Module I

INTRODUCTION TO OPERATING SYSTEM

What is an Operating System?

An operating system is a system software that acts as an intermediary between a user of a computer and the computer hardware.
It is a software that manages the computer hardware.
Os allows the user to execute programs in a convenient and efficient manner.

Operating system goals:
- Make the computer system convenient to use. It hides the difficulty in managing the hardware.
- Use the computer hardware in an efficient manner
- Provide an environment in which user can easily interface with computer.
- It is a **resource allocator**

Computer System Structure (Components of Computer System)
Computer system mainly consists of four components-

- Hardware – provides basic computing resources
  - ✓ CPU, memory, I/O devices
- Operating system
  - ✓ Controls and coordinates use of hardware among various applications and users
- Application programs – define the ways in which the system resources are used to solve the computing problems of the users
  - ✓ Word processors, compilers, web browsers, database systems, video games
- Users
  - ✓ People, machines, other computers
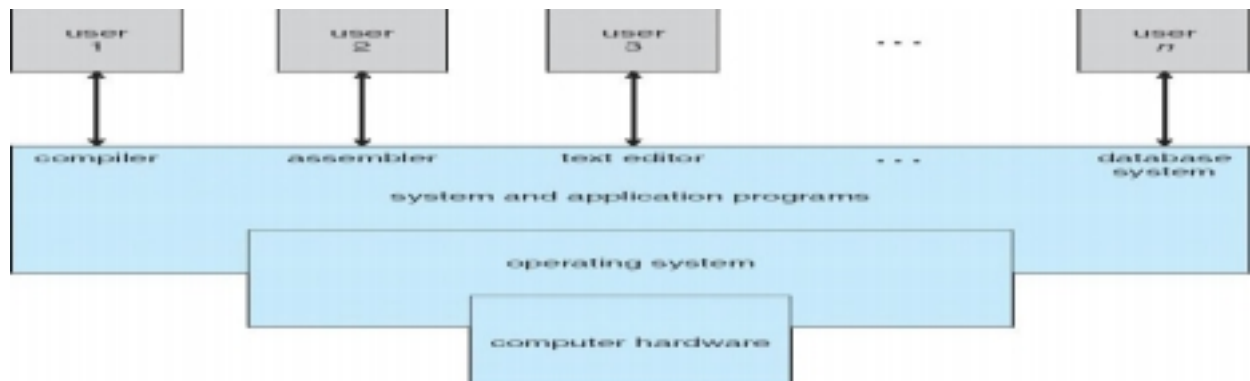
The basic hardware components comprises of CPU, memory, I/O devices. The application program uses these components. The OS controls and co-ordinates the use of hardware, among various application programs (like compiler, word processor etc.) for various users.

The OS allocates the resources among the programs such that the hardware is efficiently used.

The operating system is the program running at all the times on the computer. It is usually called as the kernel.

Non Kernel

OS (User necessary functions)

Core of OS

Kernel

(Sys. necessary functions)
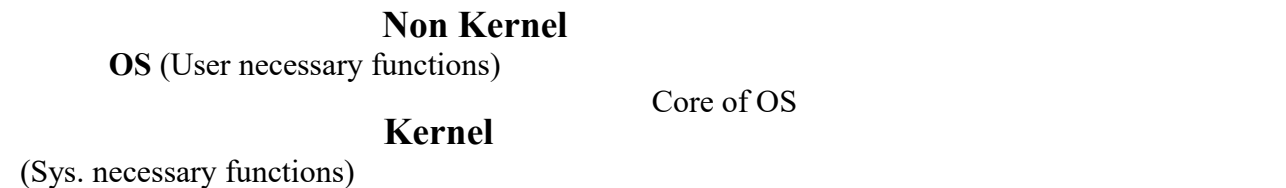
Kernel functions are used always in system, so always stored in memory. Non kernel functions are stored in hard disk, and it is retrieved whenever required.

<u>Views of OS</u>

Operating System can be viewed from two viewpoints–
    User views & System views

1. User Views:-
    The user's view of the operating system depends on the type of user.
        i. If the user is using **standalone** system, then OS is designed for <u>ease of use</u> and <u>high performances</u>. Here resource utilization is not given importance.

        ii. If the users are at different **terminals** connected to a mainframe or minicomputers, by sharing information and resources, then the OS is designed to <u>maximize resource utilization</u>. OS is designed such that the CPU time, memory and i/o are used efficiently and no single user takes more than the resource allotted to them.

        iii. If the users are in **workstations**, connected to networks and servers, then the user have a system unit of their own and shares resources and files with other systems. Here the OS is designed for both <u>ease of use</u> and <u>resource availability (files)</u>.

        iv. Users of **hand held** systems, expects the OS to be designed for <u>ease of use</u> and <u>performance</u> per amount of battery life.

        v. Other systems like embedded systems used in home devies (like washing m/c) & automobiles do not have any user interaction. There are some LEDs to show the status of its work.

2. System Views:-
Operating system can be viewed as a **resource allocator** and **control program**. i.
Resource allocator - The OS acts as a manager of hardware and software resources. CPU time, memory space, file-storage space, I/O devices, shared files etc. are the different resources required during execution of a program. There can be conflicting request for these resources by different programs running in same system. The OS assigns the resources to the requesting program depending on the priority.
ii. **Control Program** – The OS is a control program and manage the execution of user program to prevent errors and improper use of the computer.

Computer System Organization

Computer-system operation
One or more CPUs, device controllers connect through common bus providing access to shared memory. Each device controller is in-charge of a specific type of device. To ensure orderly access to the shared memory, a memory controller is provided whose function is to synchronize access to the memory. The CPU and other devices execute concurrently competing for memory cycles. Concurrent execution of CPUs and devices competing for memory cycles
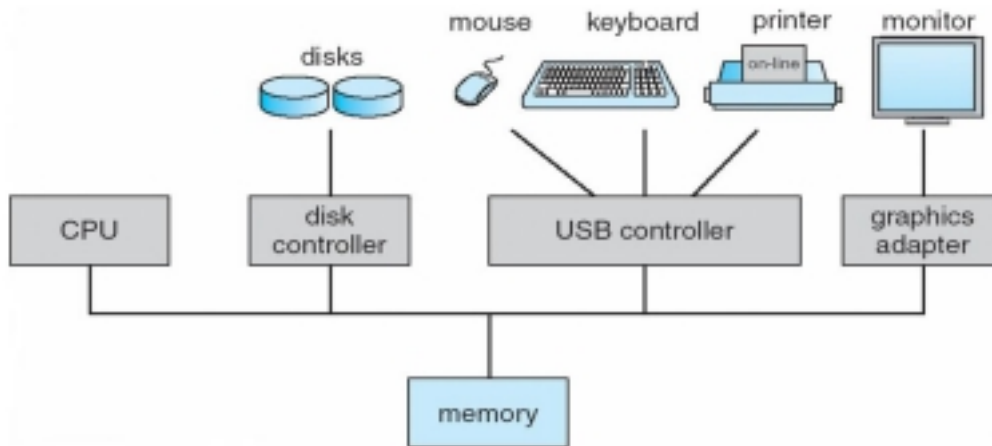
When system is switched on, '**Bootstrap'** program is executed. It is the initial program to run in the system. This program is stored in read-only memory (ROM) or in electrically erasable programmable read-only memory(EEPROM). It initializes the CPU registers, memory, device controllers and other initial setups. The program also locates and loads, the OS kernel to the memory. Then the OS starts with the first process to be executed (ie. 'init' process) and then wait for the interrupt from the user.

Switch on 'Bootstrap' program
- Initializes the registers, memory and I/O devices
- Locates & loads kernel into memory
- Starts with 'init' process
- Waits for interrupt from user.

Interrupt handling –

The occurrence of an event is usually signaled by an **interrupt.** The interrupt can either be from the hardware or the software. Hardware may trigger an interrupt at any time by sending a signal to the CPU. Software triggers an interrupt by executing a special operation called a **system call** (also called a **monitor call).**

When the CPU is interrupted, it stops what it is doing and immediately transfers execution to a fixed location. The fixed location (Interrupt Vector Table) contains the starting address where the service routine for the interrupt is located. After the execution of interrupt service routine, the CPU resumes the interrupted computation.

Interrupts are an important part of computer architecture. Each computer design has its own interrupt mechanism, but several functions are common. The interrupt must transfer control to the appropriate interrupt service routine

interrupt

Processor

IVT

Interrupt Service Routine

Stored at a fixed

location
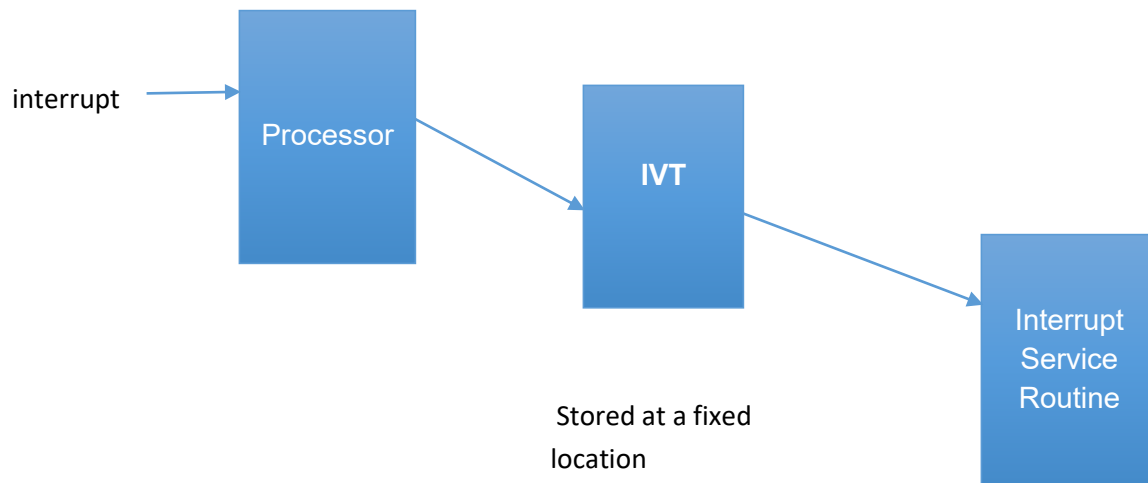
Storage Structure

Computer programs must be in main memory (**RAM)** to be executed. Main memory is the large memory that the processor can access directly. It commonly is implemented in a semiconductor technology called **dynamic random-access memory (DRAM).** Computers provide Read Only Memory(ROM), whose data cannot be changed.

All forms of memory provide an array of memory words. Each word has its own address. Interaction is achieved through a sequence of load or store instructions to specific memory addresses.

A typical instruction-execution cycle, as executed on a system with a **Von Neumann** architecture, first fetches an instruction from memory and stores that instruction in the **instruction register.** The instruction is then decoded and may cause operands to be fetched from memory and stored in some internal register. After the instruction on the operands has been executed, the result may be stored back in memory.

Ideally, we want the programs and data to reside in main memory permanently. This arrangement usually is not possible for the following two reasons:

1. Main memory is usually too small to store all needed programs and data permanently. 2. Main memory is a *volatile* storage device that loses its contents when power is turned off.

Thus, most computer systems provide **secondary storage** as an extension of main memory. The main requirement for secondary storage is that it will be able to hold large quantities of data permanently.

The most common secondary-storage device is a **magnetic disk,** which provides storage for both programs and data. Most programs are stored on a disk until they are loaded into memory. Many programs then use the disk as both a source and a destination of the information for their processing.
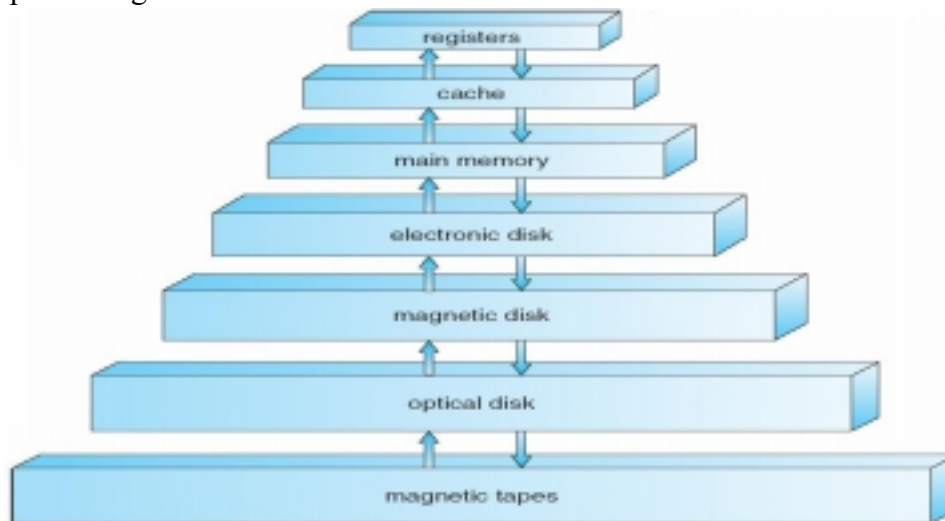
The wide variety of storage systems in a computer system can be organized in a hierarchy as shown in the figure, according to speed, cost and capacity. The higher levels are expensive, but they are fast. As we move down the hierarchy, the cost per bit generally decreases, whereas the access time and the capacity of storage generally increases.

In addition to differing in speed and cost, the various storage systems are either volatile or nonvolatile. **Volatile storage** loses its contents when the power to the device is removed. In the absence of expensive battery and generator backup systems, data must be written to **nonvolatile storage** for safekeeping. In the hierarchy shown in figure, the storage systems above the electronic disk are volatile, whereas those below are nonvolatile.

An **electronic disk** can be designed to be either volatile or nonvolatile. During normal operation, the electronic disk stores data in a large DRAM array, which is volatile. But many electronic-disk devices contain a hidden magnetic hard disk and a battery for backup power. If external power is interrupted, the electronic-disk controller copies the data from RAM to the magnetic disk. Another form of electronic disk is flash memory.

I/O Structure

A large portion of operating system code is dedicated to managing I/O, both because of its importance to the reliability and performance of a system and because of the varying nature of the devices.

Every device have a device controller, maintains some local buffer and a set of special purpose registers. The device controller is responsible for moving the data between the peripheral devices.

The operating systems have a **device driver** for each device controller.

To start an I/O operation, the device driver loads the registers within the device controller. The device controller, examines the contents of these registers to determine what action to take (such as "read a character from the keyboard"). The controller starts the transfer of data from the device to its local buffer. Once the transfer of data is complete, the device
controller informs the device driver(OS) via an interrupt that it has finished its operation. The device driver then returns control to the operating system, and also returns the data. For other operations, the device driver returns status information.

This form of interrupt-driven I/O is fine for moving small amounts of data, but very difficult for bulk data movement. To solve this problem, **direct memory access (DMA)** is used.
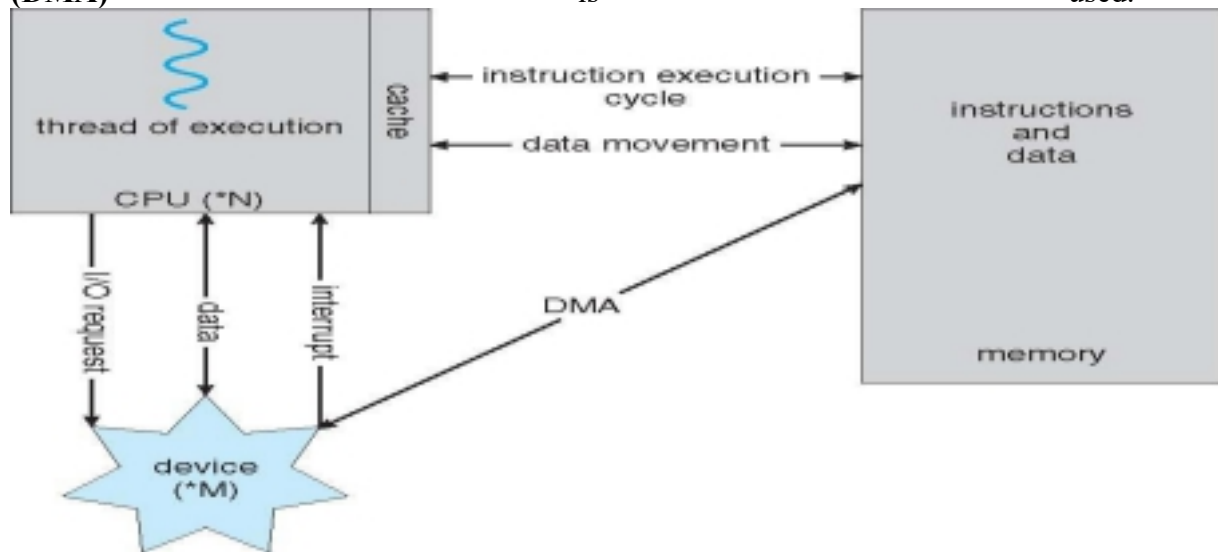
- DMA is used for high-speed I/O devices, able to transmit information at close to memory speeds
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention
- Only one interrupt is generated per block, rather than the one interrupt per byte

Computer System Architecture

Categorized roughly according to the number of general-purpose processors used –

Single-Processor Systems –

Most systems use a single processor. The variety of single-processor systems range from PDAs through mainframes. On a single-processor system, there is one main CPU capable of executing instructions from user processes. It contains special-purpose processors, in the form of device-specific processors, for devices such as disk, keyboard, and graphics controllers.

All special-purpose processors run limited instructions and do not run user processes. These are managed by the operating system, the operating system sends

them information about their next task and monitors their status.

For example, a disk-controller processor, implements its own disk queue and scheduling algorithm, thus reducing the task of main CPU. Special processors in the keyboard, converts the keystrokes into codes to be sent to the CPU.

The use of special-purpose microprocessors is common and does not turn a single processor system into a multiprocessor. If there is only one general-purpose CPU, then the system is a single-processor system.

Multiprocessor Systems (parallel systems or **tightly coupled systems) –** Systems that have two or more processors in close communication, sharing the computer bus, the clock, memory, and peripheral devices are the multiprocessor systems.

Multiprocessor systems have three main advantages:
1. **Increased throughput -** In multiprocessor system, as there are multiple processors execution of different programs take place simultaneously. Even if the number of   processors is increased the performance cannot be simultaneously increased. This is due to the overhead incurred in keeping all the parts working correctly and also due to the competation for the shared resources. The speed-up ratio with N processors is not N, rather, it is less than N. Thus the speed of the system is not has expected.

2. **Economy of scale -** Multiprocessor systems can cost less than equivalent number of   many single-processor systems. As the multiprocessor systems share peripherals, mass storage, and power supplies, the cost of implementing this system is economical. If several processes are working on the same data, the data can also be shared among them.

3. **Increased reliability-** In multiprocessor systems functions are shared among several processors. If one processor fails, the system is not halted, it only slows down. The job of the failed processor is taken up, by other processors.
    Two techniques to maintain 'Increased Reliability' - graceful degradation & fault tolerant
    Graceful degradation – As there are multiple processors when one processor fails other process will take up its work and the system goes down slowly. **Fault tolerant –** When one processor fails, its operations are stopped, the system failure is then detected, diagnosed, and corrected.
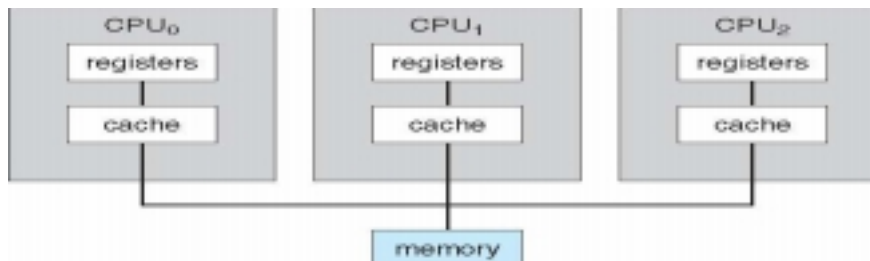
The HP NonStop system uses both hardware and software duplication to ensure continued operation despite faults. The system consists of multiple pairs of CPUs. Both processors in the pair execute same instruction and compare the results. If the results differ, then one CPU of the pair is at fault, and both are halted. The process that was being executed is then moved to another pair of CPUs, and the instruction that failed is restarted. This solution is expensive, since it involves special hardware and considerable hardware duplication.

There are two types of multiprocessor systems –
 • **Asymmetric multiprocessing**
  • **Symmetric multiprocessing**

1) **Asymmetric multiprocessing – (Master/Slave architecture)** Here each processor is assigned a specific task, by the master processor. A master processor controls the other processors in the system. It schedules and allocates work to the slave processors.

2) **Symmetric multiprocessing (SMP) –** All the processors are considered as peers. There is no master-slave relationship. All the processors have its own registers and CPU, only memory is shared.

The benefit of this model is that many processes can run simultaneously. N processes can run if there are N CPUs—without causing a significant deterioration of performance. Operating systems like Windows, Windows XP, Mac OS X, and Linux—now provide support for SMP.

A recent trend in CPU design is to include multiple computing **cores** on a single chip. Such systems are called **multicore systems** . They are more efficient than multiple chips with single core, since the communication between processors within a chip is more faster than communication between two single processors.
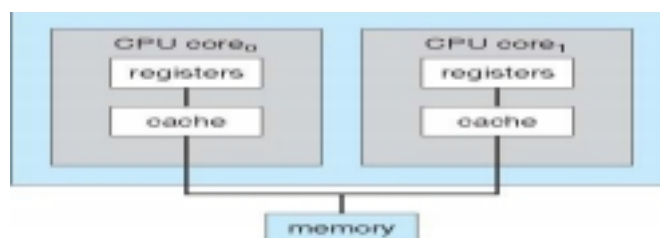
Clustered Systems

Clustered systems are two or more individual systems connected together via network and sharing software resources. Clustering provides **high-availability** of resources and services. The service will continue even if one or more systems in the cluster fail. High availability is generally obtained by storing a copy of files (s/w resources) in the system.

There are two types of Clustered systems – **asymmetric** and **symmetric**

In **asymmetric clustering –** one system is in **hot-stand by mode** while the others are running the applications. The hot-standby host machine does nothing but monitor the active server. If that server fails, the hot-standby host becomes the active server.

In **symmetric clustering –** two or more systems are running applications, and are monitoring each other. This mode is more efficient, as it uses all of the available hardware. If any system fails, its job is taken up by the monitoring system.

Other forms of clusters include parallel clusters and clustering over a wide-area network (WAN). Parallel clusters allow multiple hosts to access the same data on the shared storage. Cluster technology is changing rapidly with the help of **SAN(storage-area networks)**. Using SAN resources can be shared with dozens of systems in a

cluster, that are separated by miles.

Operating-System Structure

One of the most important aspects of operating systems is the ability to multiprogram. A single user cannot keep either the CPU or the I/O devices busy at all times. **Multiprogramming** increases CPU utilization by organizing jobs, so that the CPU always has one to execute.
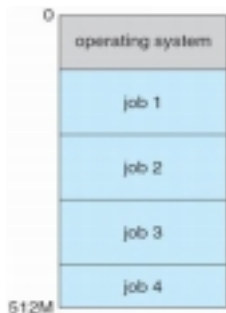
The operating system keeps several jobs in memory simultaneously as shown in figure. This set of jobs is a subset of the jobs kept in the job pool. Since the number of jobs that can be kept simultaneously in memory is usually smaller than the number of jobs that can be kept in the job pool(in secondary memory). The operating system picks and begins to execute one of the jobs in memory. Eventually, the job may have to wait for some task, such as an I/O operation, to complete. In a non-multiprogrammed system, the CPU would sit   idle. In a multiprogrammed system, the operating system simply switches to, and executes, another job. When *that* job needs to wait, the CPU is switched to *another* job, and so on. Eventually, the first job finishes waiting and gets the CPU back. Thus the CPU is never idle.
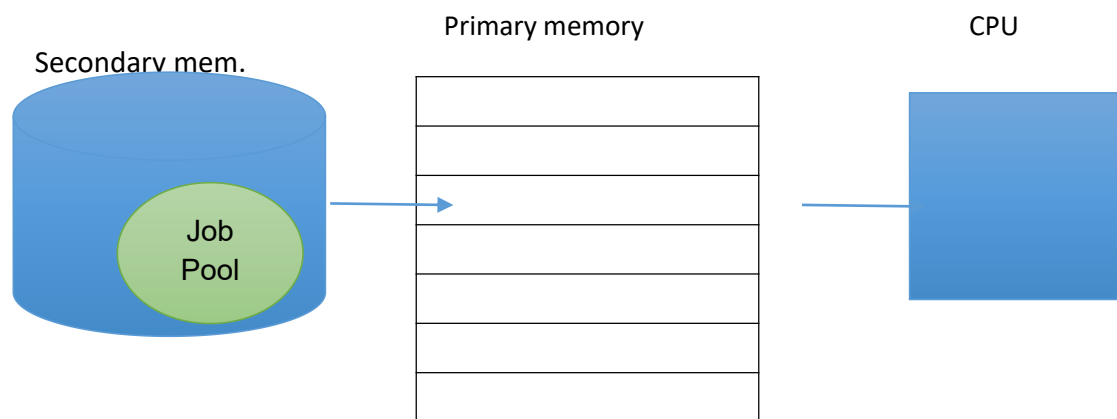
Multiprogrammed systems provide   an environment in which the various system resources (for   example, CPU, memory, and peripheral devices) are utilized effectively, but they do not provide for user interaction with the computer system.

In **Time sharing** (or **multitasking) systems,** a single CPU executes multiple jobs by switching among them, but the switches occur so frequently that the users can interact with each program while it is running. The user feels that all the programs are

being executed at the same time. Time sharing requires an **interactive** (or **hands-on) computer system,** which provides direct communication between the user and the system. The user gives instructions to the operating system or to a program directly, using a input device such as a keyboard or a mouse, and waits for immediate results on an output device. Accordingly, the **response time** should be short—typically less than one second.

A time-shared operating system allows many users to share the computer simultaneously. As the system switches rapidly from one user to the next, each user is given the impression that the entire computer system is dedicated to his use only, even though it is being shared among many users.

A **multiprocessor system** is a computer system having two or more CPUs within a single computer system, each sharing main memory and peripherals. Multiple programs are executed by multiple processors parallel.

Distributed Systems

Individual systems that are connected and share the resource available in network is called Distributed system. Access to a shared resource increases computation speed, functionality, data availability, and reliability.

A **network** is a communication path between two or more systems. Distributed systems depend on networking for their functionality. Networks vary by the protocols used, the distances between nodes, and the transport media. TCP/IP is the most common network protocol. Most operating systems support TCP/IP.

Networks are characterized based on the distances between their nodes. A **local-area network (LAN)** connects computers within a room, a floor, or a building. A **wide-area network (WAN)** usually links buildings, cities, or countries. A global company may have a WAN to connect its offices worldwide. A **metropolitan-area network** (MAN) links buildings within a city. A **small-area network** connects systems within a several feet using wireless technology. Eg. BlueTooth and 802.11.

The media to carry networks also vary - copper wires, fiber strands, and wireless transmissions between satellites, microwave dishes, and radios.

A **network operating system** is an operating system that provides features such as file sharing across the network and that allows different processes on different computers to exchange messages.A computer running a network operating system acts autonomously from all other computers on the network, although it is aware of the network and is able to communicate with other networked computers.

Operating-System Operations

Modern operating systems are **interrupt driven.** If there are no processes to execute, no I/O devices to service, and no users to whom to respond, an operating system will sit quietly, waiting for something to happen. Events are signaled by the occurrence of an interrupt or a trap. A **trap (or** an **exception)** is a software-generated

interrupt. For each type of interrupt, separate segments of code in the operating system determine what action should be taken. An interrupt service routine is provided that is responsible for dealing with the interrupt.

a) Dual-Mode Operation

Since the operating system and the user programs share the hardware and software resources of the computer system, it has to be made sure that an error in a user program cannot cause problems to other programs and the Operating System running in the system.

The approach taken is to use a hardware support that allows us to differentiate among various modes of execution.

The system can be assumed to work in two separate **modes** of operation:
   • **user mode** and
   • **kernel mode (supervisor mode, system mode, or privileged mode).**

A hardware bit of the computer, called the **mode bit**, is used to indicate the current mode: kernel (0) or user (1). With the mode bit, we are able to distinguish between a task that is executed by the operating system and one that is executed by the user.

When the computer system is executing a user application, the system is in user mode. When a user application requests a service from the operating system (via a system call), the transition from user to kernel mode takes place.
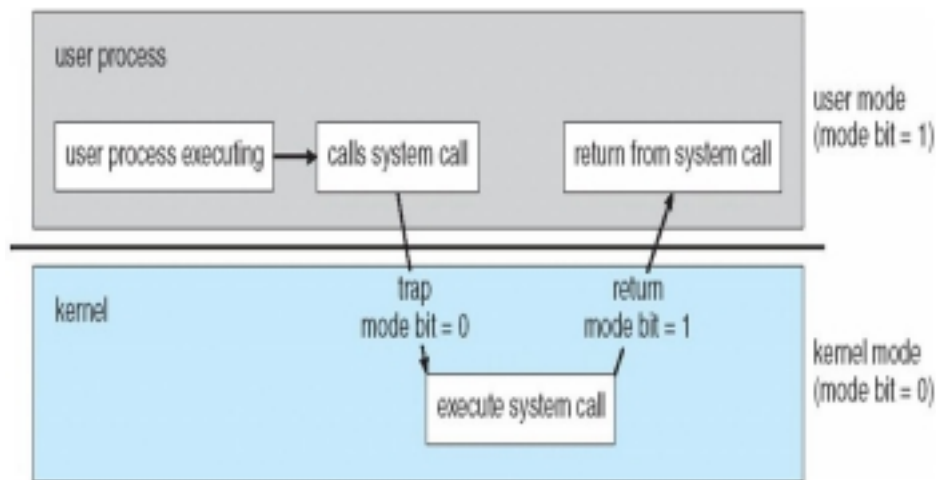
At system boot time, the hardware starts in kernel mode. The operating system is then loaded and starts user applications in user mode. Whenever a trap or interrupt occurs, the hardware switches from user mode to kernel mode (that is, changes the mode bit from 1 to 0). Thus, whenever the operating system gains control of the computer, it is in kernel mode.

The dual mode of operation provides us with the means for protecting the operating system from errant users—and errant users from one another.

The hardware allows privileged instructions to be executed only in kernel mode. If an attempt is made to execute a privileged instruction in user mode, the hardware does not execute the instruction but rather treats it as illegal and traps it to the operating system. The instruction to switch to user mode is an example of a privileged instruction.

Initial control is within the operating system, where instructions are executed in kernel mode. When control is given to a user application, the mode is set to user mode. Eventually, control is switched back to the operating system via an interrupt, a trap, or a system call.

b) Timer

Operating system uses timer to control the CPU. A user program cannot hold CPU for a long time, this is prevented with the help of timer.

A timer can be set to interrupt the computer after a specified period. The period may be **fixed** (for example, 1/60 second) or **variable** (for example, from 1 millisecond to 1 second).

Fixed timer – After a fixed time, the process under execution is interrupted.

Variable timer – Interrupt occurs after varying interval. This is implemented using a fixed-rate clock and a counter. The operating system sets the counter. Every time the clock ticks, the counter is decremented. When the counter reaches 0, an interrupt occurs.

Before changing to the user mode, the operating system ensures that the timer is set to interrupt. If the timer interrupts, control transfers automatically to the operating system, which may treat the interrupt as a fatal error or may give the program more time.

Process Management

A program under execution is a process. A process needs resources like CPU time, memory, files, and I/O devices for its execution. These resources are given to the process when it is created or at run time. When the process terminates, the operating system reclaims the resources.

The program stored on a disk is a **passive entity** and the program under execution is an **active entity**. A single-threaded process has one **program counter** specifying the next instruction to execute. The CPU executes one instruction of the process after another, until the process completes. A multithreaded process has multiple program counters, each pointing to the next instruction to execute for a given thread.

The operating system is responsible for the following activities in connection with process management:
- Scheduling process and threads on the CPU
- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication

Memory Management

Main memory is a large array of words or bytes. Each word or byte has its own address. Main memory is the storage device which can be easily and directly accessed by the CPU. As the program executes, the central processor reads instructions and also

reads and writes data from main memory.

To improve both the utilization of the CPU and the speed of the computer's response to its users, general-purpose computers must keep several programs in memory, creating a need for memory management.

The operating system is responsible for the following activities in connection with memory management:

> • Keeping track of which parts of memory are currently being used by user. • Deciding which processes and data to move into and out of memory.
>
> • Allocating and deallocating memory space as needed.

Storage Management

There are three types of storage management i) File system management ii) Mass-storage management iii) Cache management.

File-System Management

File management is one of the most visible components of an operating system. Computers can store information on several different types of physical media. Magnetic disk, optical disk, and magnetic tape are the most common. Each of these media has its own characteristics and physical organization. Each medium is controlled by a device, such as a disk drive or tape drive, that also has its own unique characteristics.

A file is a collection of related information defined by its creator. Commonly, files represent programs and data. Data files may be numeric, alphabetic, alphanumeric, or binary. Files may be free-form (for example, text files), or they may be formatted rigidly (for example, fixed fields).

The operating system implements the abstract concept of a file by managing mass storage media. Files are normally organized into directories to make them easier to use. When multiple users have access to files, it may be desirable to control by whom and in what ways (read, write, execute) files may be accessed.

The operating system is responsible for the following activities in connection with file management:

> • Creating and deleting files
> • Creating and deleting directories to organize files
> • Supporting primitives for manipulating files and directories
> • Mapping files onto secondary storage
> • Backing up files on stable (nonvolatile) storage media

Mass-Storage Management

As the main memory is too small to accommodate all data and programs, and as the data that it holds are erased when power is lost, the computer system must provide secondary storage to back up main memory. Most modern computer systems use disks as the storage medium for both programs and data.

Most programs—including compilers, assemblers, word processors, editors, and formatters—are stored on a disk until loaded into memory and then use the disk as both the source and destination of their processing. Hence, the proper management of

disk storage is of central importance to a computer system. The operating system is responsible for the following activities in connection with disk management:

- Free-space management
- Storage allocation
- Disk scheduling

As the secondary storage is used frequently, it must be used efficiently. The entire speed of operation of a computer may depend on the speeds of the disk. Magnetic tape drives and their tapes, CD, DVD drives and platters are **tertiary storage** devices. The functions that operating systems provides include mounting and unmounting media in devices, allocating and freeing the devices for exclusive use by processes, and migrating data from secondary to tertiary storage.

Caching

Caching is an important principle of computer systems. Information is normally kept in some storage system (such as main memory). As it is used, it is copied into a faster storage system— the cache—as temporary data. When a particular piece of information is required, first we check whether it is in the cache. If it is, we use the information directly from the cache; if it is not in cache, we use the information from the source, putting a copy in the cache under the assumption that we will need it again soon.

Because caches have limited size, **cache management** is an important design problem. Careful selection of the cache size and page replacement policy can result in greatly increased performance.

The movement of information between levels of a storage hierarchy may be either explicit or implicit, depending on the hardware design and the controlling operating-system software. For instance, data transfer from cache to CPU and registers is usually a hardware function, with no operating-system intervention. In contrast, transfer of data from disk to memory is usually controlled by the operating system.

In a hierarchical storage structure, the same data may appear in different levels of the storage system. For example, suppose to retrieve an integer A from magnetic disk to the processing program. The operation proceeds by first issuing an I/O operation to copy the disk block on which A resides to main memory. This operation is followed by copying A to the cache and to an internal register. Thus, the copy of A appears in several places: on the magnetic disk, in main memory, in the cache, and in an internal register.
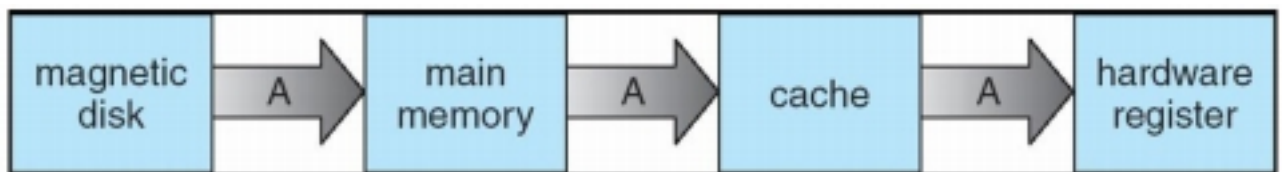
In a multiprocessor environment, in addition to maintaining internal registers, each of the CPUs also contains a local cache. In such an environment, a copy of A may exist simultaneously in several caches. Since the various CPUs can all execute concurrently, any update done to the value of A in one cache is immediately reflected in all other caches where A resides. This situation is called **cache coherency,** and it is usually a hardware problem (handled below the operating-system level).

I/O Systems

One of the purposes of an operating system is to hide the peculiarities of specific hardware devices from the user. The I/O subsystem consists of several components: • A memory-management component that includes buffering, caching, and spooling

- • A general device-driver interface
- • Drivers for specific hardware devices

Only the device driver knows the peculiarities of the specific device to which it is assigned.

Protection and Security

If a computer system has multiple users and allows the concurrent execution of multiple processes, then access to data must be regulated. For that purpose, mechanisms ensure that files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system.

If a computer system has multiple users and allows the concurrent execution of multiple processes, then access to data must be regulated. For that purpose, there are mechanisms which ensure that files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system.

For example, memory-addressing hardware ensures that a process can execute only within its own address space. The timer ensures that no process can gain control of the CPU for a long time. Device-control registers are not accessible to users, so the integrity of the various peripheral devices is protected.

Protection is a mechanism for controlling the access of processes or users to the resources defined by a computer system. This mechanism must provide means for specification of the controls to be imposed and means for enforcement.

Protection improves reliability. A protection-oriented system provides a means to distinguish between authorized and unauthorized usage. A system can have adequate protection but still be prone to failure and allow inappropriate access.

Consider a user whose authentication information is stolen. Her data could be copied or deleted, even though file and memory protection are working. It is the job of **security** to defend a system from external and internal attacks. Such attacks spread across a huge range and include viruses and worms, denial-of service attacks etc.

Protection and security require the system to be able to distinguish among all its users. Most operating systems maintain a list of user names and associated **user identifiers (user IDs).** When a user logs in to the system, the authentication stage determines the appropriate user ID for the user.

Distributed Systems

A distributed system is a collection of systems that are networked to provide the users with access to the various resources in the network. Access to a shared resource increases computation speed, functionality, data availability, and reliability.

A **network** is a communication path between two or more systems. Networks vary by the protocols used(TCP/IP,UDP,FTP etc.), the distances between nodes, and the transport media(copper wires, fiber-optic,wireless).

TCP/IP is the most common network protocol. The operating systems support

of protocols also varies. Most operating systems support TCP/IP, including the Windows and UNIX operating systems.

Networks are characterized based on the distances between their nodes. A **local-area network (LAN)** connects computers within a room, a floor, or a building. A **wide-area network (WAN)** usually links buildings, cities, or countries. A global company may have a WAN to connect its offices worldwide. These networks may run one protocol or several protocols. A **metropolitan-area network** (MAN) connects buildings within a city. BlueTooth and 802.11 devices use wireless technology to communicate over a distance of several feet, in essence creating a **small-area network** such as might be found in a home.

The transportation media to carry networks are also varied. They include copper wires, fiber strands, and wireless transmissions between satellites, microwave dishes, and radios. When computing devices are connected to cellular phones, they create a network.

Special-Purpose Systems

There are different classes of computer systems, whose functions are more limited and specific and it deal with limited computation domains. The systems can be classified as Real-Time Embedded Systems, Multimedia Systems and Handheld Systems.

Real-Time Embedded Systems

Embedded computers are the most prevalent form of computers in existence. These devices are found everywhere, from car engines and manufacturing robots to VCRs and microwave ovens. They tend to have very specific tasks. Usually, they have little user interface, and more time is spent for monitoring and managing hardware devices, eg. automobile engines and robotic arms.

The Operating Systems, in these embedded systems vary considerably. Some systems have standard operating systems—such as UNIX—with special-purpose applications. Others have special-purpose embedded operating system providing just the functionality desired.

Embedded systems always run **real-time operating systems.** A real-time system is used when there is restricted time for an operation or for the flow of data. A real-time system functions

correctly only if it returns the correct result within its time constraints. Sensors bring data to the computer. The computer must analyze the data and perform certain action.

Some medical imaging systems, automobile-engine fuel-injection systems, home appliance controllers, and weapon systems are real-time systems. A real-time system has well defined, fixed time constraints. Processing **mustbe** done within the defined constraints, or the system will fail. For instance, the robot arm should be

halted before it has smashed into the car, it was building.

Entire houses can be computerized, so that a computer —can control heating and lighting, alarm systems, and even coffee makers. Web access can enable a home owner to tell the house to heat up before she arrives home.

Multimedia Systems

Multimedia data consist of audio and video files as well as conventional files. These data differ from conventional data in that multimedia data—such as frames of video— must be delivered (streamed) according to certain time restrictions (for example, 30 frames per second).

Multimedia describes a wide range of applications like audio files - MP3, DVD movies, video conferencing, and short video clips of movie previews or news. Multimedia applications may also include live webcasts of speeches or sporting events and even live webcams. Multimedia applications can be either audio or video or combination of both. For example, a movie may consist of separate audio and video tracks.

Handheld Systems

Handheld systems include personal digital assistants (PDAs), such as Palm and Pocket-PCs, and cellular telephones. Developers of these systems face many challenges, due to the limited memory, slow processors and small screens in such devices.

The amount of physical memory in a handheld depends upon the device, the operating system and applications must manage memory efficiently. This includes returning all allocated memory back to the memory manager when the memory is not being used. A second issue of concern to developers of handheld devices is the speed of the processor used in the devices. Processors for most handheld devices run at faster speed than the processor in a PC. Faster processors require more power and so, a larger battery is required. Another issue is the usage of I/O devices.

Generally, the limitations in the functionality of PDAs are balanced by their convenience and portability. Their use continues to expand as network connections become more available and other options, such as digital cameras and MP3 players, expand their utility.

Computing Environments

The different computing environments are -

Traditional Computing

The current trend is toward providing more ways to access these computing environments. Web technologies are stretching the boundaries of traditional computing. Companies establish **portals,** which provide web accessibility to their internal servers. **Network computers** are essentially terminals that understand web-based computing. Handheld computers can synchronize with PCs to allow very portable use of company information. Handheld PDAs can also connect to **wireless networks** to use the company's web portal. The fast data connections are allowing home computers to serve up web pages and to use networks. Some homes even have **firewalls** to protect their networks.

In the latter half of the previous century, computing resources were scarce. Years before, systems were either batch or interactive. Batch system processed jobs in bulk,

with predetermined input (from files or other sources of data). Interactive systems waited for input from users. To optimize the use of the computing resources, multiple users shared time on these systems. Time-sharing systems used a timer and scheduling algorithms to rapidly cycle processes through the CPU, giving each user a share of the resources.

Today, traditional time-sharing systems are used everywhere. The same scheduling technique is still in use on workstations and servers, but frequently the processes are all owned by the same user (or a single user and the operating system). User processes, and system processes that   provide services to the user, are managed so that each frequently gets a slice of computer time.

Client-Server Computing

Designers shifted away from centralized system architecture to - terminals connected to centralized systems. As a result, many of today's systems act as **server systems to** satisfy requests generated by **client systems.** This form of specialized distributed system, called **client server** system.
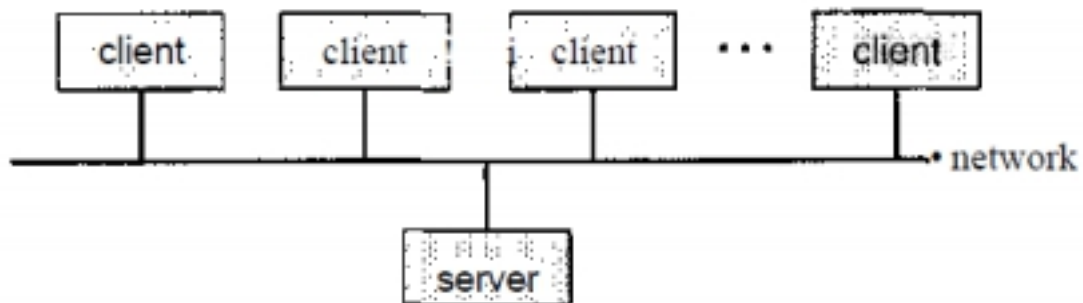
General Structure of Client – Server System

Server systems can be broadly categorized as compute servers and file servers: • The **compute-server system** provides an interface to which a client can send a request to perform an action (for example, read data); in response, the server executes the action and sends back results to the client. A server running a database that responds to client requests for data is an example of such a svstem.
• The **file-server system** provides a file-system interface where clients can create, update, read, and delete files. An example of such a system is a web server that delivers files to clients running the web browsers.

Peer-to-Peer Computing

In this model, clients and servers are not distinguished from one another; here, all nodes within the system are considered peers, and each may act as either a client or a server, depending on whether it is requesting or providing a service.
In a client-server system, the server is a bottleneck, because all the services must be served by the server. But in a peer-to-peer system, services can be provided by several nodes distributed throughout the network.

To participate in a peer-to-peer system, a node must first join the network of peers. Once a node   has joined the network, it can begin providing services to—and requesting services from—other nodes in the network. Determining what services are

available is accomplished in one of two general ways:

- When a node joins a network, it registers its service with a centralized lookup service on the network. Any node desiring a specific service first contacts this centralized lookup service to determine which node provides the service. The remainder of the communication takes place between the client and the service provider.
- A peer acting as a client must know, which node provides a desired service by broadcasting a request for the service to all other nodes in the network. The node (or nodes) providing that service responds to the peer making the request. To support this approach, a *discovery protocol* must be provided that allows peers to discover services provided by other peers in the network.

Web-Based Computing

Web computing has increased the importance on networking. Devices that were not previously networked now include wired or wireless access. Devices that were networked now have faster network connectivity.

The implementation of web-based computing has given rise to new categories of devices, such as **load balancers,** which distribute network connections among a pool of similar servers. Operating systems like Windows 95, which acted as web clients, have evolved into Linux and Windows XP, which can act as web servers as well as clients. Generally, the Web has increased the complexity of devices, because their users require them to be web-enabled.

The design of an operating system is a major task. It is important that the goals of the new system be well defined before the design of OS begins. These goals form the basis for choices among various algorithms and strategies.