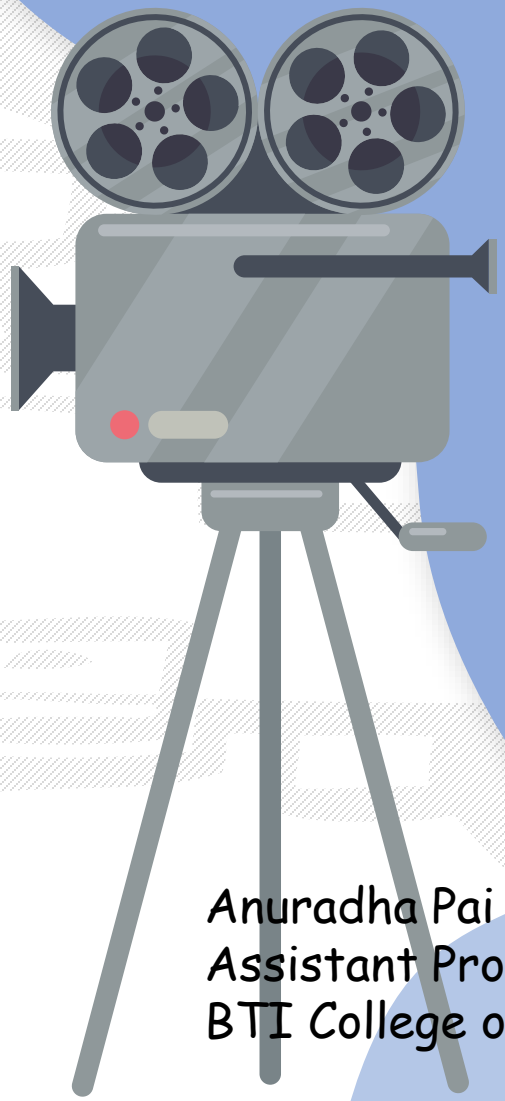
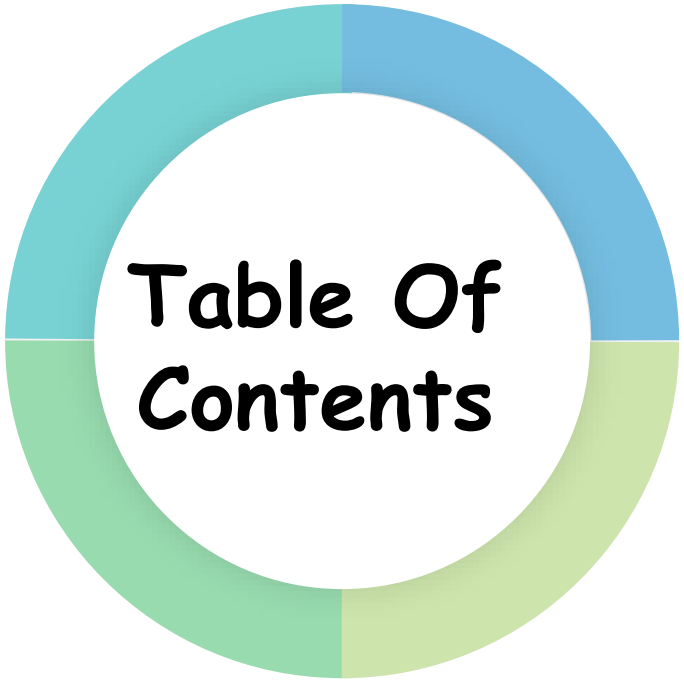
OPERATING SYSTEMS

Module-1: Introduction



Anuradha Pai
Assistant Prof
BTI College of Engineering



1. Introduction

- Computer system is a collection of software and hardware components.
- Hardware refers to all the physical components(the components that you can touch) of the computer
- Software is a set programs that direct a computer on how to carry out specific tasks. It gives life and meaning to the hardware and brings it to operational level. Without software, hardware is a useless piece of metal.
- Software is of two types
    1. Application Software
    2. System Software

1. Introduction(Contd..)

- System Software: Software that manages and controls the computer's hardware, providing a platform for other software to run. Examples include operating systems like Windows and macOS, device drivers, firmware etc.
- Application Software: Software designed to help users perform specific tasks or activities, such as word processing, web browsing, or gaming. Examples include Microsoft Word, Google Chrome etc.

TYPES OF SOFTWARE

2. What is an Operating System?

- An operating system is a system software which is an interface between the computer user and the hardware. It is a software that manages the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use. It hides the difficulty in managing the hardware.
  - Use the computer hardware in an efficient manner
  - It is a **resource allocator**

3. Computer System Structure

Computer system can be divided into four components:

1. Hardware – provides basic computing resources
   - CPU, memory, I/O devices
2. Operating system
   - Controls and coordinates use of hardware among various applications and users
3. 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
4. Users
   - People, machines, other computers

3.1 Components of a computer system

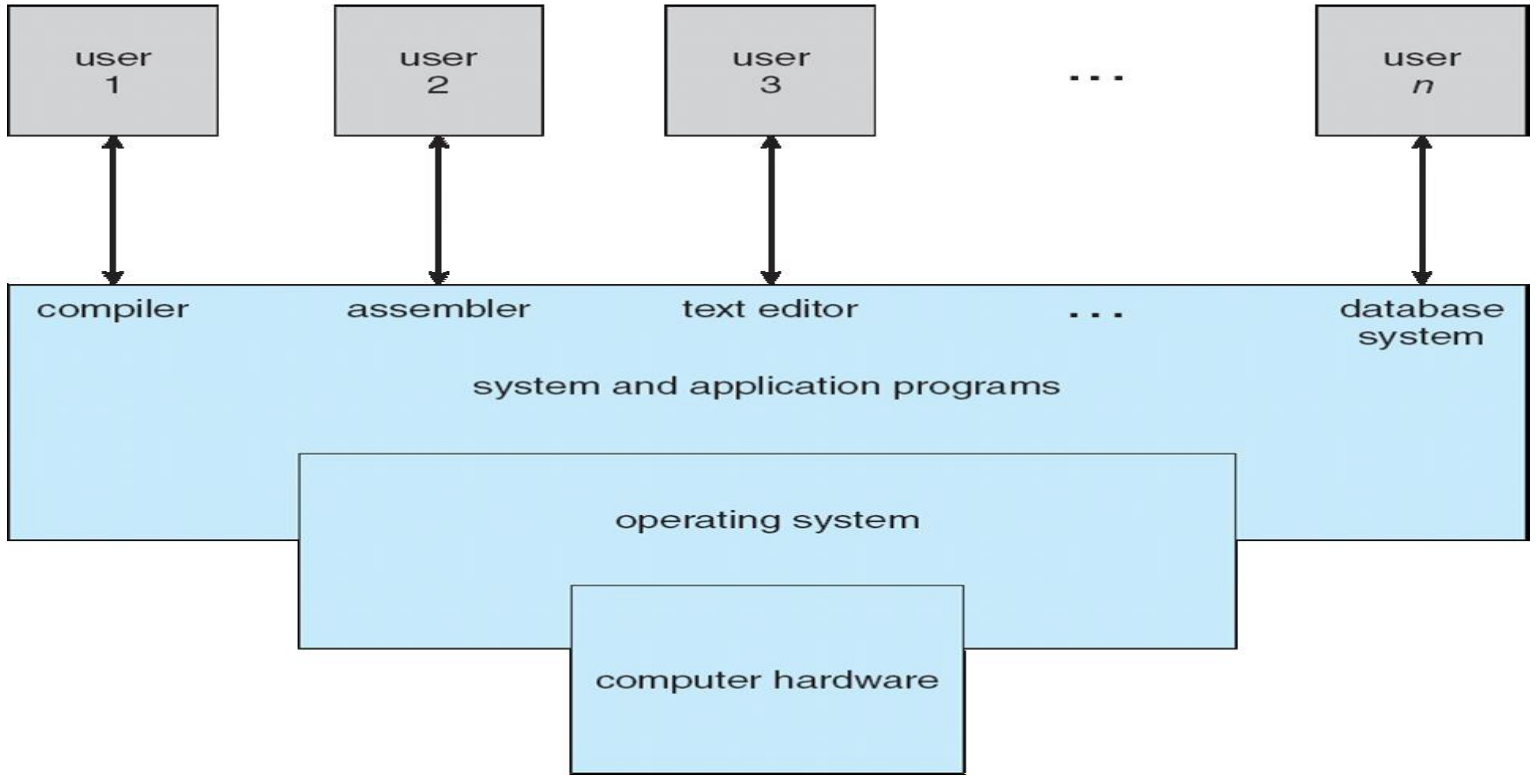


Fig: Abstract View of the components of a computer system

4. What Operating Systems do? User's View

- Depends on the point of view
- Users want convenience, **ease of use**
  - Don't care about **resource utilization**
- But shared computer such as **mainframe** must keep all users happy
- Users of dedicate systems such as **workstations** have dedicated resources but frequently use shared resources from **servers**
- Handheld devices are resource poor, optimized for usability and battery life.
- Some computers have little or no user interface, such as embedded computers in devices and automobiles

4. What Operating Systems do? System's View

- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer
- "The one program running at all times on the computer" is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program. 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.

5. Computer System Organization

- Computer-system operation
  - One or more CPUs and device controllers are connected through a common bus providing access to shared memory
  - The CPU and device controllers can operate simultaneously, which may lead to competition for memory cycles.

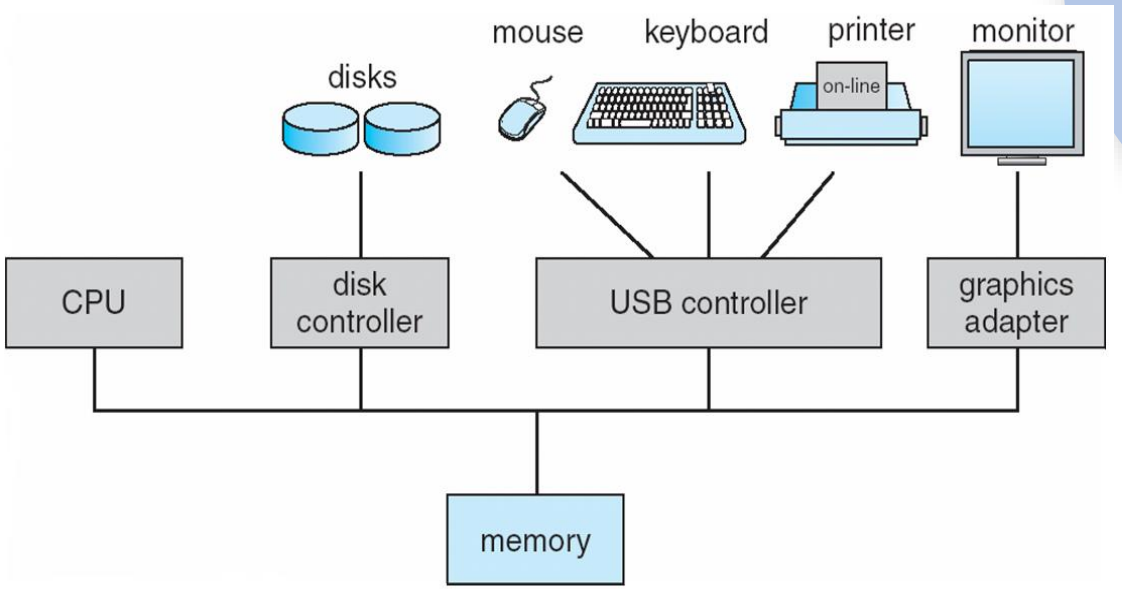


Fig: A Modern computer system

5.1 Computer System Operation: Computer Startup

- **Bootstrap program(also known as BIOS/firmware)** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM
  - Initializes all aspects of system
  - Loads operating system kernel and starts execution
- 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.

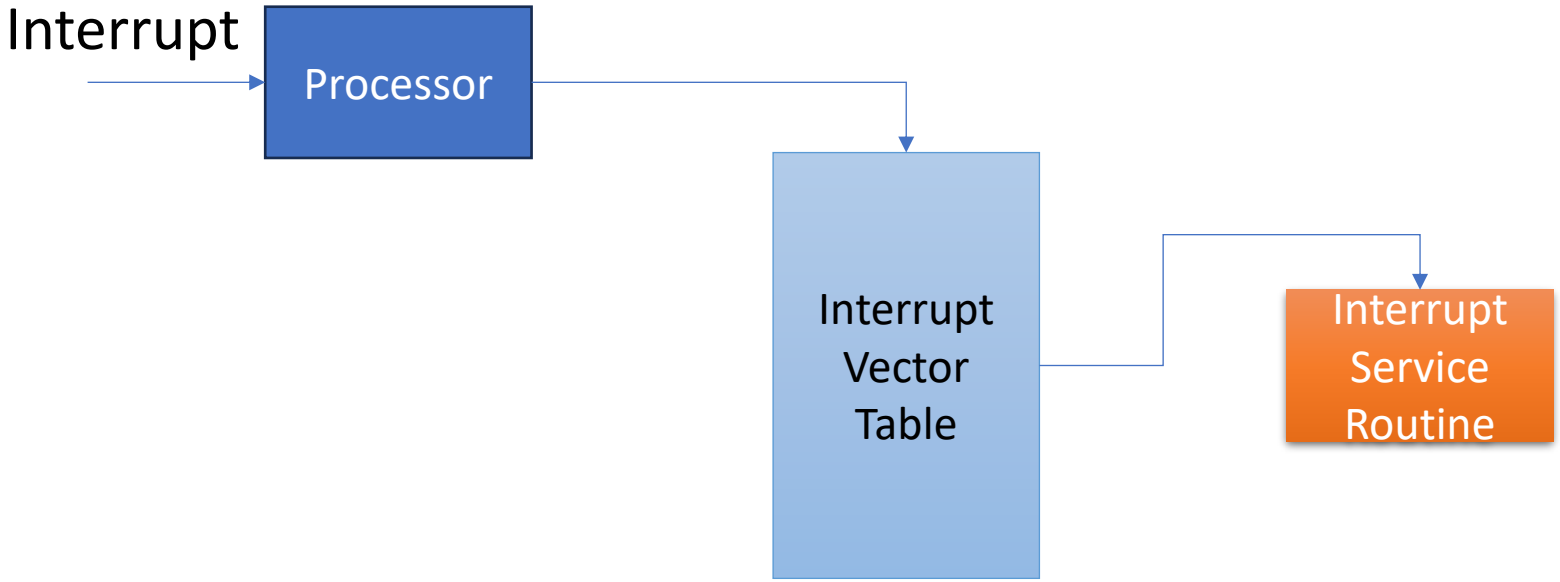
5.1 Computer System Operation

- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular device type
- Each device controller has a local buffer
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller
- Device controller informs CPU that it has finished its operation by causing an interrupt

5.1 Computer System Operation

- The occurrence of an event is usually signalled 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. A *trap* is a software-generated interrupt caused either by an error or a user request
- Interrupt architecture must save the address of the interrupted instruction
- An operating system is **interrupt driven**
- 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.

5.1 Computer System Operation: Interrupt



5.2 Storage Structure

- Computer programs must be in main memory (**RAM)** to be executed. Main memory - only large storage media that the CPU can access directly
  - **Random access**
  - Typically **volatile**
- Ideally, we want the programs and data to reside in main memory permanently. This arrangement usually is not possible for the following two reasons:
  - Main memory is usually too small to store all needed programs and data permanently.
  - 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.

5.2 Storage Structure Hierarchy

- 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(flash memory) are volatile, whereas those below are nonvolatile.

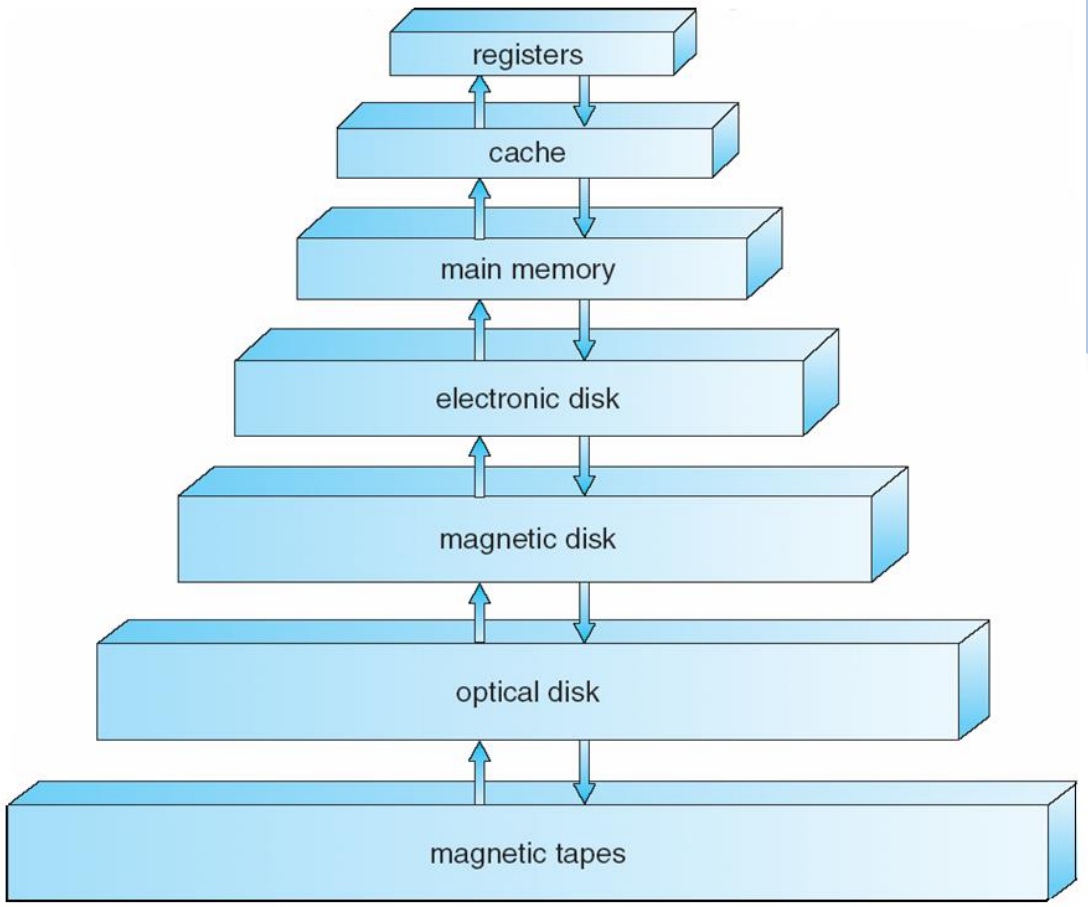


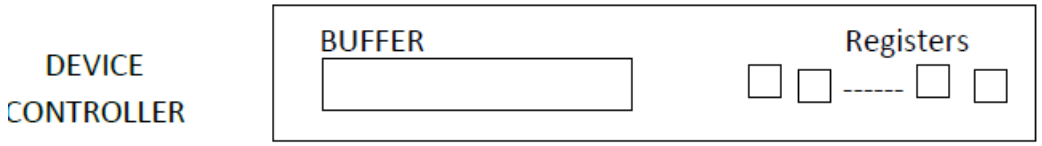
Fig: Storage Structure Hierarchy

5.2 Storage Structure: Caching

- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used

5.3 I/O Structure

- A large portion of operating system code is dedicated to managing I/O
- Every device has 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.

5.3.1 Direct Memory Access(DMA)

- 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.
  - 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

5.3.2 How modern computer works?

Von Neuman 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.

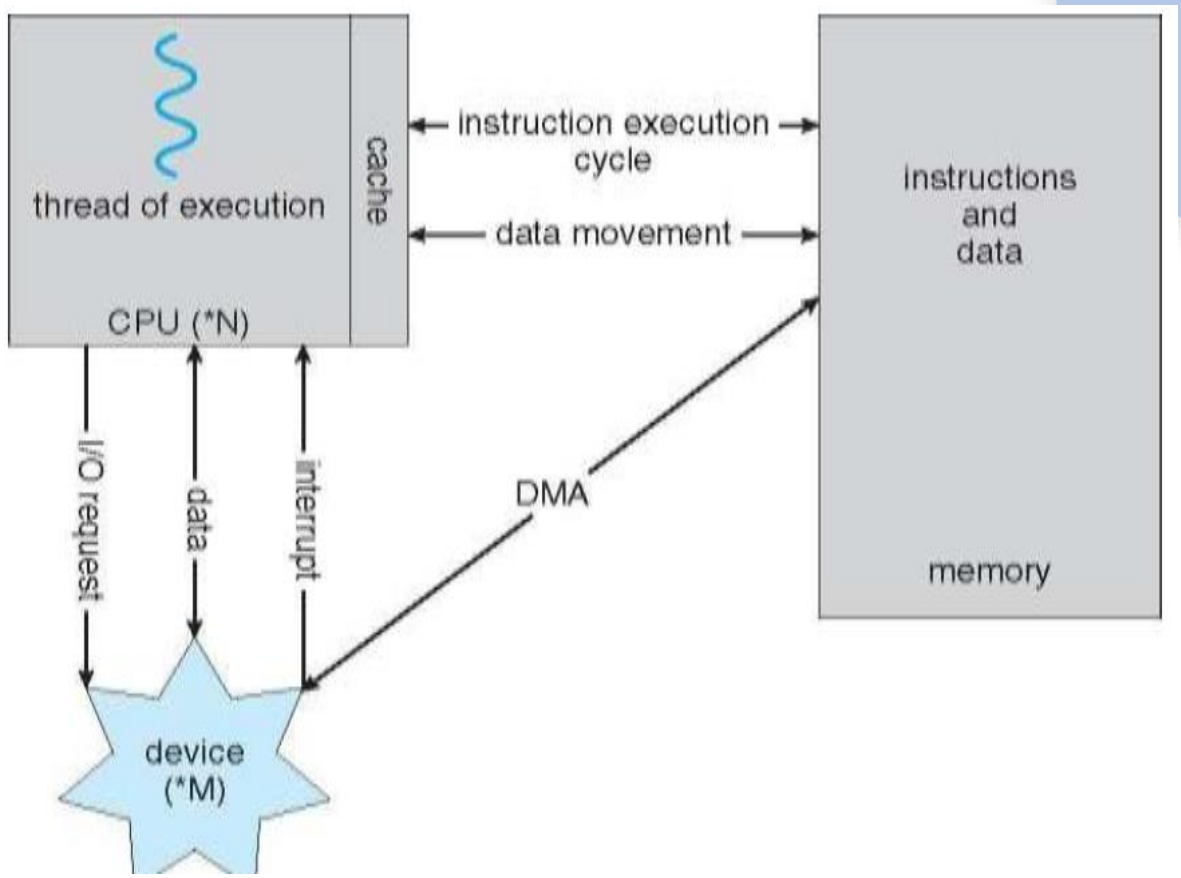


Fig: Interplay of all components of computer system

6.Computer System Architecture(CSA)

Computer systems can be categorized according to the number of general-purpose processors present in the system

1. Single-Processor Systems
2. Multiprocessor Systems
   - Symmetric Multiprocessing
   - Asymmetric Multiprocessing

6. 1 CSA: Single-Processor Systems

- There is one main CPU capable of executing a general-purpose instruction set, including instructions from user processes.
- Almost all single-processor systems have other special-purpose processors as well. Two types
    1. Device-specific processors for devices such as disk, keyboard etc.
    2. They may come in the form of more general-purpose processors, such as I/O processors that move data rapidly among the components of the system.
- Specialized processors reduce the load or effort required by the main CPU

6. 2 CSA: Multiprocessor 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. **Improved Performance**:By distributing tasks across multiple processors, a multiprocessor system can handle more processes simultaneously, leading to faster execution and improved overall performance.
    2. **Increased Reliability**: In a multiprocessor system, if one processor fails, others can take over its tasks, providing fault tolerance and making the system more reliable.
    3. **Cost Efficiency:** 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.

6. 2 CSA: Multiprocessor Systems

- 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.

6.2.1 CSA: Symmetric Multiprocessing (SMP)

- All the processors are considered as peers. There is no master-slave relationship. All the processors have its own registers and cache, 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.

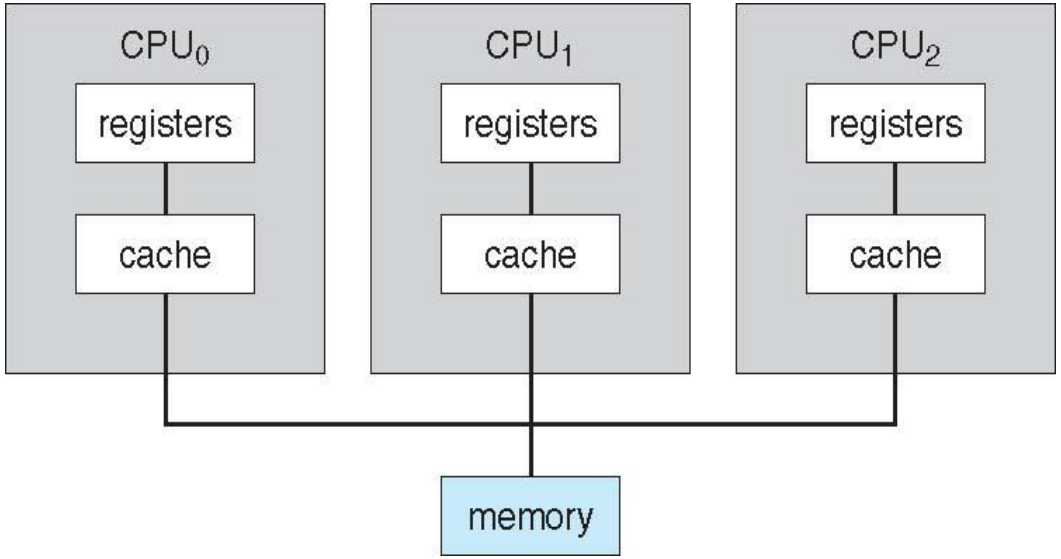


Fig: Symmetric Multiprocessing Architecture

6.2.2 CSA: 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.

6.3 CSA: Multicore Design

A recent trend in CPU design is to include multiple compute **cores** on a single chip. Such multiprocessor systems are termed **multicore**. They can be more efficient than multiple chips with single cores because on-chip communication is faster than between-chip communication. In addition, one chip with multiple cores uses significantly less power than multiple single-core chips.

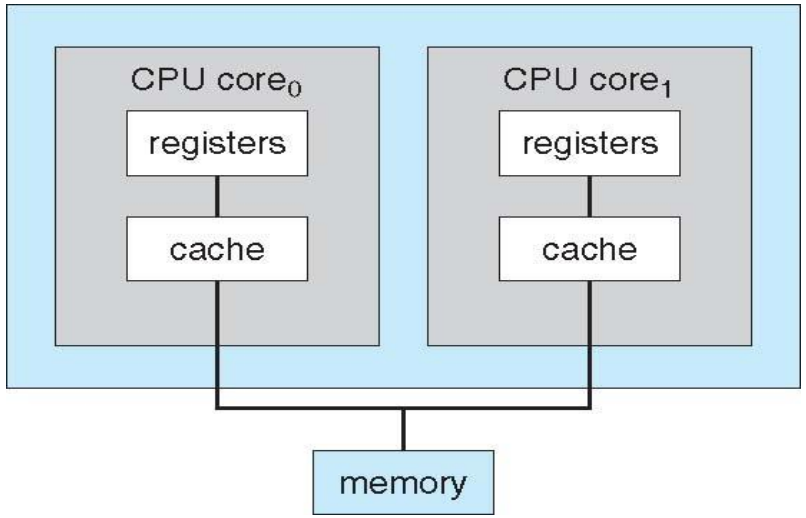


Fig: A dual-core design with two cores placed on same chip

6.3 Clustered Systems

- Like multiprocessor systems, but multiple systems working together
- 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.
- Some clusters are for **high-performance computing (HPC).** Applications must be written to use **parallelization**
- 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.

6.3 Clustered Systems

- Some cluster products support dozens of systems in a cluster, as well as clustered nodes that are separated by miles.
- **Storage-area networks (SANs)** which allow many systems to attach to a pool of storage. If the applications and their data are stored on the SAN, then the cluster software can assign the application to run on any host that is attached to the SAN. If the host fails, then any other host can take over.

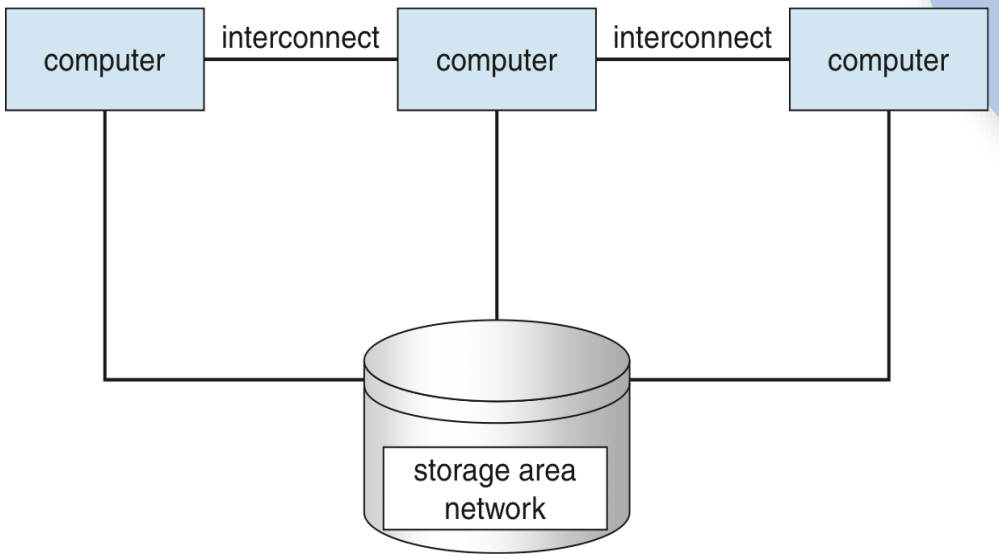


Fig: General structure of a clustered system.

7. Operating System Structure

- Single user cannot keep CPU and I/O devices busy at all times. Single users frequently have multiple programs running.
- Multiprogramming increases CPU utilization by arranging jobs(processes) in a way that ensures the CPU always has a task to execute.

7.1 Multiprogramming

The operating system keeps several jobs in memory simultaneously as shown in figure.

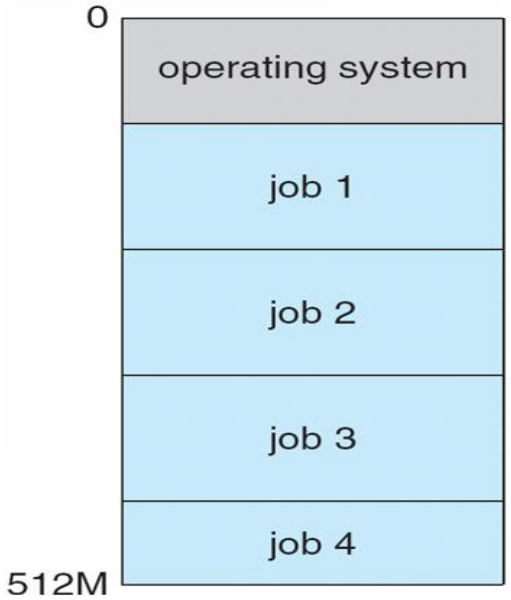
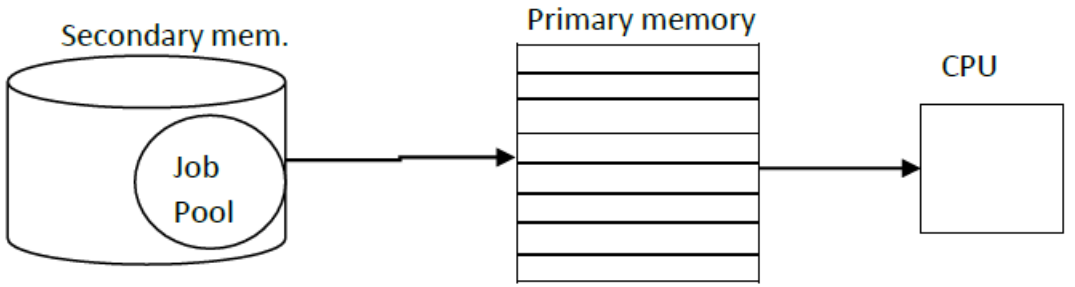


Fig: Memory Layout of Multiprogramming System

- Main memory is too small to accommodate all jobs, the jobs are kept initially on the disk in the **job pool**.
- This pool consists of all processes residing on disk awaiting allocation of main memory.
- The set of jobs in memory can be a subset of the jobs kept in the job pool.



7.1 Multiprogramming

- 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.

7.1 Multiprogramming

- Advantages
  - High and efficient CPU Utilization
  - User feels that many programs are allotted CPU simultaneously
- Disadvantages
  - CPU Scheduling is required
  - To accommodate many jobs in memory, memory management is required

7.2 Multitasking (Timesharing)

- Multitasking refers to term where 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.
- Multitasking operating systems are also called time sharing systems. A time shared operating system uses the concept of CPU scheduling and multiprogramming to provide each user a time-shared CPU
- In a time-sharing system, the operating system must ensure reasonable response time. This goal is sometimes accomplished through
  - **Swapping**, whereby processes are swapped in and out of main memory to the disk.
  - **Virtual memory**, a technique that allows the execution of a process that is not completely in memory

8. 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 example, division by zero or invalid memory access)
- 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 properly designed operating system must ensure that an incorrect (or malicious) program cannot cause other programs to execute incorrectly.

8.1 Dual Mode Operations

- **Dual-mode** operation allows OS to protect itself and other system components.
  - **User mode** and **kernel mode**
  - **Mode bit** is provided by hardware
    1. Provides ability to distinguish when system is running user code or kernel code
    2. Some instructions designated as **privileged**, only executable in kernel mode.
    3. System call changes mode to kernel, return from call resets it to user

       When a system call is executed, it is treated by the hardware as a software interrupt. Control passes to the service routine in the operating system, and the mode bit is set to kernel mode. The system-call service routine is a part of the operating system.

***Note:** System calls provide the means for a user program to ask the operating system to perform tasks reserved for the operating system on the user program's behalf*

8.1 Transition from User to Kernel Mode

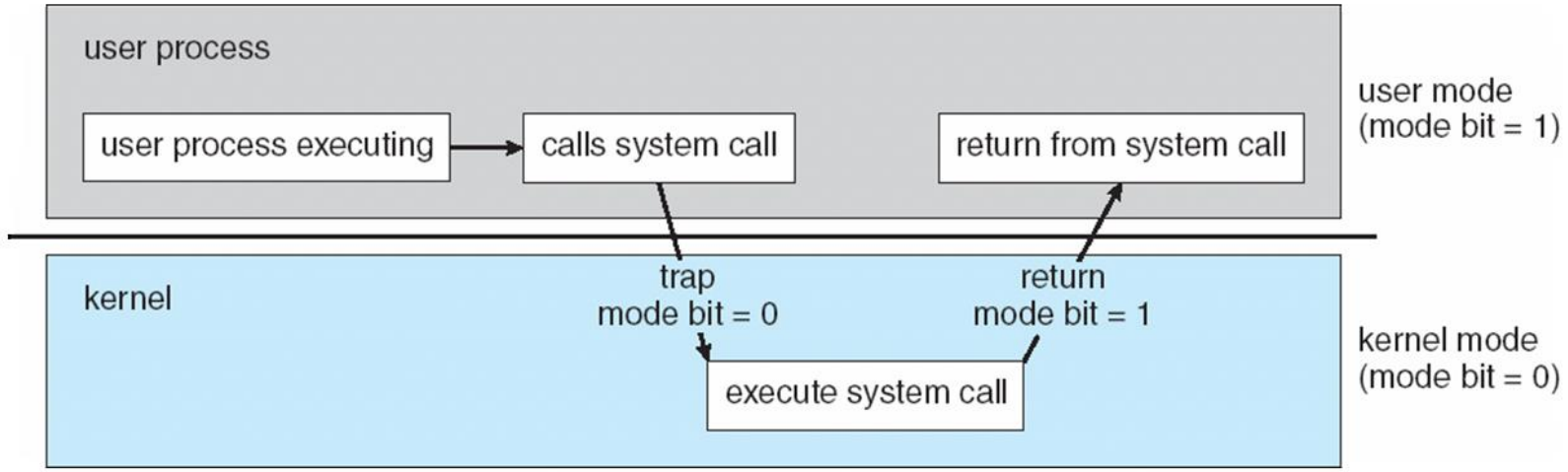


Fig: Transition from user to kernel mode

8.2 Timer

- A user program cannot hold CPU for a long time, this is prevented with the help of timer. This timer will prevent user program from running too long.
- 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).
- How does Timer work?
  - Set interrupt after specific period of time.
  - Operating system decrements counter for each clock tick
  - When counter zero generate an interrupt. This interrupt will transfer the control to operating system, thereby preventing the user programs from running too long. OS might terminate the program that exceeds the allotted time.

9. Process Management

- A program under execution is a process. A process needs resources like CPU tin memory, files, and I/O devices for its execution. These resources are given to th process when it is created or at run time. When the process terminates, th operating system reclaims the resources.
- The program stored on a disk is a **passive entity** and the program under execution is an **active entity**.
- Single-threaded process has one **program counter** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread.
- Typically system has many processes, some of which are operating-system processes (those that execute system code) and the rest of which are user processes (those that execute user code).running concurrently on one or more CPUs

9. Process Management (Contd..)

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

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

10. Memory Management

- Main memory is a large array of bytes. **Each byte has its own address.** The central processor reads instructions from main memory during the instruction-fetch cycle and both reads and writes data from main memory during the data-fetch cycle.
- The main memory is generally the only large storage device that the CPU is able to address and access directly. For example, for the CPU to process data from disk, those data must first be transferred to main memory by CPU-generated I/O calls. In the same way, instructions must be in memory for the CPU to execute them.
- 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.

11. Storage Management

- The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the **file**.
- There are three types of storage management done by OS

1. File System Management
2. Mass-storage Management
3. Caching

11.1 File System Management

- 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.

11.1 File System Management

- A file is a collection of related information defined by its creator. Files are usually organized into directories
- When multiple users have access to files, access control on most systems to determine who can access what and in what ways they can access it(read, write, execute)
- OS activities include
  - Creating and deleting files and directories
  - Primitives to manipulate files and dirs
  - Mapping files onto secondary storage
  - Backup files onto stable (non-volatile) storage media

11.2 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.
- Entire speed of computer operation hinges on disk subsystem and its algorithms

11.2 Mass-Storage Management

- The operating system is responsible for the following activities in connection with disk management:
  - Free-space management
  - Storage allocation
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Still must be managed – by OS or applications
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

11.3 Caching

- **Caching** is a fundamental concept in computer systems. Data is typically stored in a primary storage system, like main memory, but when it is accessed, a copy is temporarily stored in a faster system, called the cache. When we need specific data, we first check if it's in the cache. If it is, we use the cached version; if not, we retrieve it from the original source and store a copy in the cache, assuming it will be needed again soon.
- Due to the limited size of caches, efficient cache management is crucial. Properly choosing the cache size and implementing an effective page replacement policy can significantly boost performance.
- Data movement between different levels of the storage hierarchy can be either explicit or implicit, depending on the hardware and the operating system. For example, transferring data from cache to the CPU or registers is typically handled by hardware without operating system involvement, while transferring data from disk to memory is usually managed by the operating system.

11.3 Caching

Hierarchical Storage Structure:

- Data may exist at multiple levels within a storage system (e.g., magnetic disk, main memory, cache, and internal registers).
- Example: Retrieving an integer A involves copying it from **magnetic disk** to **main memory**, then to the **cache**, and finally 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.

11.3 Caching

- 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**
- **Cache Coherency**:
  - **Cache coherency** refers to ensuring that updates to data in one cache are immediately reflected in all other caches where the same data resides.
  - Cache coherency is typically a **hardware-level issue**, managed below the operating system.

11.4 Performance of Various Levels of Storage

Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape

12. Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

13. 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 transportation media to carry networks are also varied. They include copper wires, fiber strands, and wireless transmissions between satellites, radios etc.

13. Distributed 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.

14. 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
- Handheld Systems.

14.1 Real-Time Embedded Systems

- **Embedded Computers are Common**: These are the most common type of computers found today.
- **Found Everywhere**: They are used in things like car engines, robots, VCRs, and microwaves.
- **Do Specific Jobs**: They are designed to do specific tasks.
- **Limited User Interaction**: They don’t need much input from people and focus on controlling hardware.
- Different Operating Systems:
  - Some use regular operating systems like UNIX but with special apps.
  - Others use simple operating systems that do just what’s needed.
- A **real-time embedded system** is a specialized computing system designed to perform dedicated tasks within a strict timing constraint, where the correctness of its operation depends not only on logical results but also on the timing of those results. These systems are often embedded within larger systems to control hardware and respond to external events in real time.

14.1 Real-Time Embedded Systems

- 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 **must be** 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 xa 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.

14.2 Multimedia Systems

- **Conventional vs. Multimedia Data:**
  - Most operating systems are traditionally designed for conventional data like text files, programs, word-processing documents, and spreadsheets.
  - Multimedia data includes audio, video, and conventional files, with added complexity due to time restrictions (e.g., delivering video frames at 30 frames per second).
- **Wide Range of Multimedia Applications:**
  - Popular multimedia applications include:
    - Audio files (e.g., MP3)
    - DVD movies
    - Video conferencing
    - Short video clips (e.g., movie previews, news stories)
    - Live webcasts (e.g., speeches, sporting events)
    - Live webcams (e.g., observing a cafe in Paris from Manhattan)
- **Combination of Audio and Video:**
  - Multimedia applications often integrate both audio and video (e.g., movies with separate audio and video tracks).
- **Multimedia Beyond Desktop Computers:**
  - Multimedia applications are increasingly directed at smaller devices like PDAs and cellular phones, enabling real-time data delivery (e.g., stock quotes sent wirelessly to a PDA).
- **Operating System Requirements for Multimedia:**
  - The design of operating systems must adapt to the specific demands of multimedia data, which includes handling both time-sensitive delivery and supporting a broader range of devices.

14.3 Handheld Systems

- **Types of Handheld Systems**:
  - Personal Digital Assistants (PDAs) like Palm and Pocket-PCs.Cellular telephones.
- **Challenges for Developers:**
  - Limited Memory: The physical memory varies by device, requiring efficient memory management in the operating system and applications.
  - Slow Processors: While handheld processors may be faster than typical PC processors, they still present speed limitations.
  - Power Consumption: Faster processors demand more power, necessitating larger batteries.I/O Device Limitations: The input/output functionalities may be restricted.
- **Memory Management**:
  - Efficient allocation and deallocation of memory is crucial, including returning unused memory to the memory manager.

14.3 Handheld Systems

- **Processor Speed:**
  - Although many handheld devices have faster processors than PCs, the balance between performance and power consumption is a key concern.
- **Functionality vs. Convenience:**
  - The limited functionality of PDAs is often offset by their convenience and portability.
  - The usability of handheld systems is increasing with better network connectivity and additional features like digital cameras and MP3 players.
- **Market Trends:**
  - Expanding utility and functionalities of handheld devices are likely to lead to further adoption and integration into daily life.

15.1 Traditional Computing

Expanding Access to Computing Environments:

- There is a trend toward providing **more ways to access** computing environments, driven by **web technologies**.

Web Technologies and Traditional Computing:

- **Web technologies** are **expanding the boundaries** of traditional computing.
- Companies are creating **portals** that provide web access to their **internal servers**.

Network Computers:

- **Network computers** act as **terminals** designed specifically for **web-based computing**.

Handheld Devices:

- **Handheld computers** can **synchronize with PCs**, enabling portable access to company information.
- **PDAs** (Personal Digital Assistants) can connect to **wireless networks** and access the company's **web portal**.

Home Networking:

- **Fast data connections** are enabling home computers to both **serve web pages** and utilize networks.
- Some homes now use **firewalls** to protect their networks from external threats.

15.1 Traditional Computing

Scarcity of Computing Resources in the Past:

- In the latter half of the 20th century, **computing resources were limited**, leading to the need for efficient use of those resources.

Batch and Interactive Systems:

- **Batch systems** processed jobs in bulk with **predetermined input** from files or data sources.
- **Interactive systems** relied on **user input** to proceed.

Time-Sharing Systems:

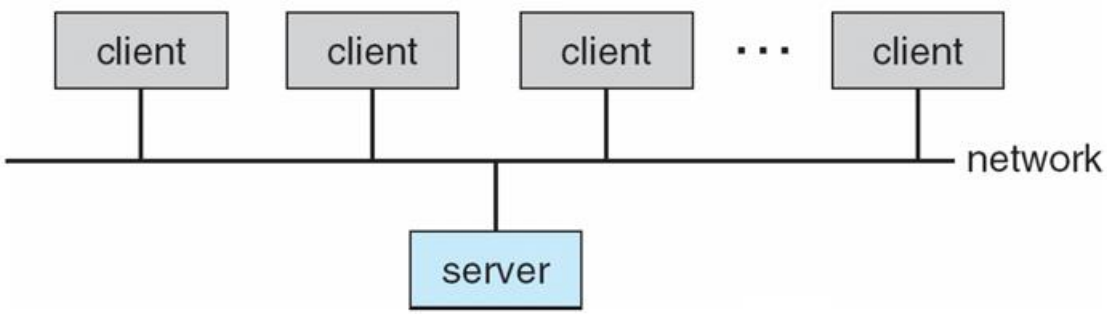
- **Time-sharing systems** allowed multiple users to share computing resources by **rapidly cycling processes** through the CPU.
- These systems used **timers** and **scheduling algorithms** to allocate CPU time to each user.

Modern Use of Time-Sharing:

- **Time-sharing systems** are still widely used today, particularly on **workstations** and **servers**.
- Although multiple users aren't always involved, the same **scheduling techniques** are used to manage **user processes** and **system processes**.
- These processes are scheduled to ensure each gets a **slice of CPU time**.

15.2 Client Server Computing

- Designers moved away from centralized system architecture towards using terminals connected to centralized systems.
- As a result, many modern systems function as servers, handling requests from client systems. This type of specialized distributed system is known as a **client-server system.**



15.2 Client Server Computing

Server systems can be broadly categorized as:

- 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.

15.3 Peer-to-Peer Computing

- In a peer-to-peer system, clients and servers are not distinguished; all nodes are considered peers.
- Each peer can act as either a client (requesting services) or a server (providing services) depending on the situation.
- **Comparison with Client-Server Systems:**
  - In a client-server system, the server often becomes a bottleneck as all services must be provided by a central server.
  - In a peer-to-peer system, services are distributed across multiple nodes, reducing reliance on a single server and increasing efficiency.
- **Joining the Network:**
  - To participate in a peer-to-peer system, a node must first join the network of peers.
  - Once joined, the node can both request services from and provide services to other peers.
- **Service Discovery:**
  - Two general methods for discovering services in P2P systems:
  - Centralized Lookup Service: When a node joins the network, it registers its services with a centralized lookup service, which other nodes consult to find service providers.
  - Broadcasting Requests: A peer can broadcast a service request to all other nodes in the network. Nodes that provide the service respond to the request.
- **Discovery Protocol:**
  - A discovery protocol is needed to allow peers to find services provided by other peers in the network, enabling efficient communication and service sharing.

15.4 Web-based Computing

- **Increased Importance of Networking:**
  - Web computing has enhanced the importance of networking.
  - Devices that were not previously networked now have wired or wireless access.
  - Devices that were already networked now benefit from faster network connectivity.
- **Emergence of New Devices:**
  - Web-based computing has led to new types of devices, like load balancers, which distribute network traffic among multiple servers.
- **Evolution of Operating Systems:**
  - Early operating systems like Windows 95 functioned primarily as web clients.
  - Modern operating systems, such as Linux and Windows XP, can now function as both web servers and clients.

15.4 Web-based Computing

- **Increased Complexity of Devices:**
  - The demand for web-enabled functionality has led to greater device complexity.
- **Designing Operating Systems:**
  - Designing an operating system is a significant challenge.
  - It's essential to define the goals of the operating system before the design process starts.
  - These goals influence the selection of algorithms and strategies during the design process.

Some succeed because they are destined.
Some succeed because they are determined.
Good luck ☺