CSE-3211, Operating System

Chapter 1: Introduction

Md Saidur Rahman Kohinoor
Lecturer, Dept. of Computer Science
Leading University – Sylhet
kohinoor_cse@lus.ac.bd

Overview

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture
- Operating-System Structure
- Operating-System Operations
- Process Management
- Memory Management
- Storage Management
- Protection and Security
- Kernel Data Structures
- Computing Environments
- Open-Source Operating Systems

Objectives

- To describe the basic organization of computer systems
- To provide a grand tour of the major components of operating systems
- To give an overview of the many types of computing environments
- To explore several open-source operating systems

Operating System

- A program that controls the execution of application programs and acts as an intermediary between a user of a computer and the computer hardware

- Operating system goals:
  - **Simplicity:** Execute user programs and make solving user problems easier. Provide an user interface and simplify the computer system.
  - **Convenience:** Make the computer system convenient to use
  - **Efficiency:** Use the computer hardware in an efficient manner
  - **Ability to Evolve:** Permit the system to effective development, testing and introducing new system functions without interfering the existence service.

Computer System Structure

- Computer system can be divided into 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

Four Components of a Computer System

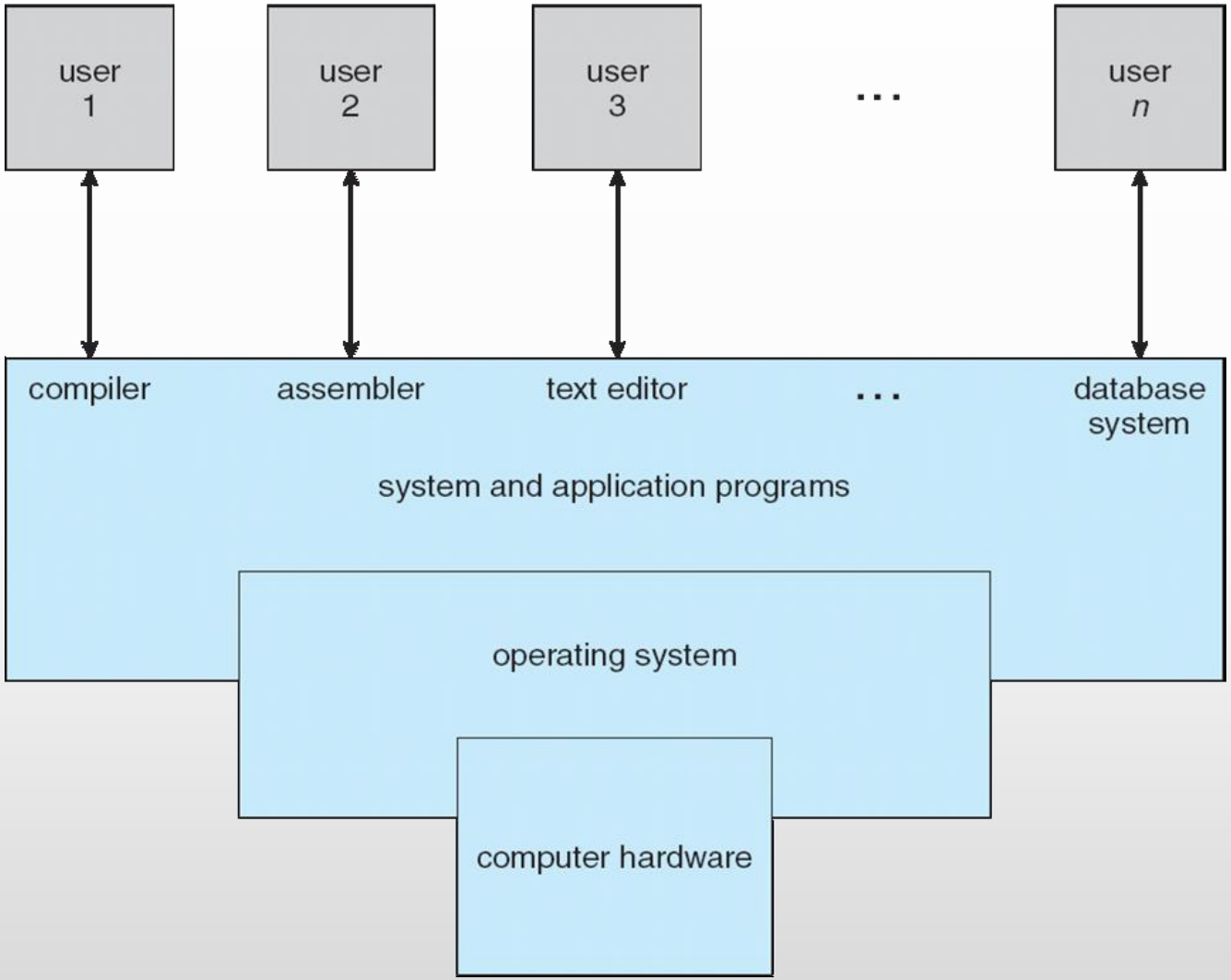


Fig 1.1: Conceptual view of a computer system

What Operating Systems Do

- Depends on the point of view
- Users want convenience, **ease of use**
  - Don't care about **resource utilization**
- But shared computer such as **mainframe** or **minicomputer** must keep all users happy
- Users of dedicate systems such as **workstations** have dedicated resources but frequently use shared resources from **servers**
- Handheld computers are resource poor, optimized for usability and battery life (smart phone)
- Some computers have little or no user interface, such as embedded computers in devices and automobiles (microwave oven)
- Others: Security, Error detecting aids, Process Management, Memory management, etc.

Operating System Definition

- No universally accepted definition
- 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 operating system is the one program running at all times on the computer". This is usually called the **kernel**. Everything else is either a system program (ships with the operating system) or an application program.

Kernel

In computing, the **kernel** is a computer program that manages I/O requests from software, and translates them into data processing instructions for the CPU and other electronic components of a computer. The **kernel** is a core part of an **operating system** which acts like a bridge between application and hardware of the computer. It is one of the first programs loaded on start-up (after the Bootloader).

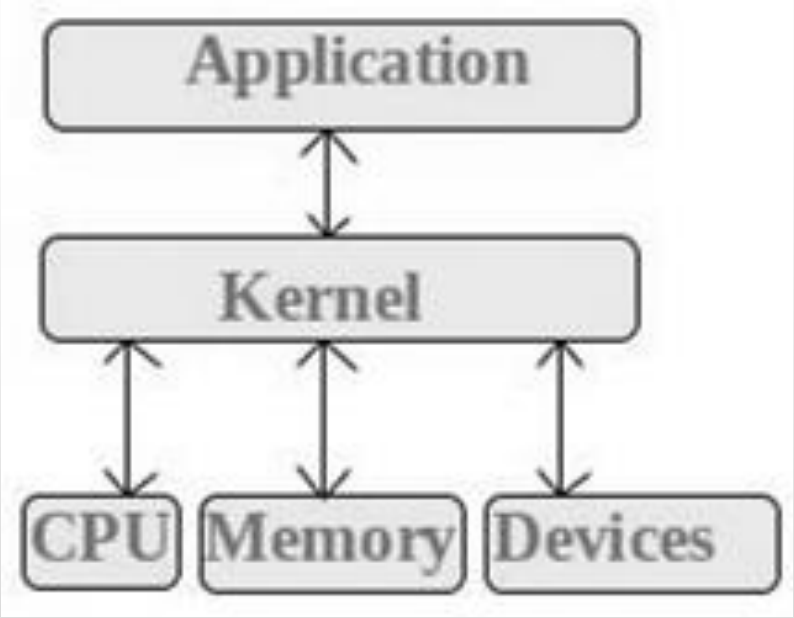


Fig 1.2: Overview

Examples of Operating System

- Windows (GUI based, PC)
- GNU/Linux (Personal, Workstations, ISP, File and print server, Three-tier client/Server)
- macOS (Macintosh), used for Apple's personal computers and workstations (MacBook, iMac).
- Android (Google's OS for smartphones/tablets/smartwatches)
- iOS (Apple's OS for iPhone, iPad, and iPod Touch)

Note:

- **GNU** is a Unix-like operating system. That means it is a collection of many programs: applications, libraries, developer tools, even games. The development of **GNU**, started in January 1984, is known as the **GNU** Project.
- **UNIX** is originally spelled UNICS (UNiplexed Information & Computing System).
- **Linux** is the **full** name of the kernel. It was created as a combination of the phrase "Linus' **Unix**", after the original author, Linus Torvalds.

Types of Operating System

Some widely used operating systems are –

- Batch Operating System
- Time-sharing Operating System
- Distributed Operating System
- Network Operating System
- Real-time Operating System

Batch Operating System

- This type of OS does not interact with the computer directly. There is an operator which takes similar jobs having the same requirement and group them into batches. It is the responsibility of the operator to sort jobs with similar needs.
- Examples: Payroll System, Bank Statements

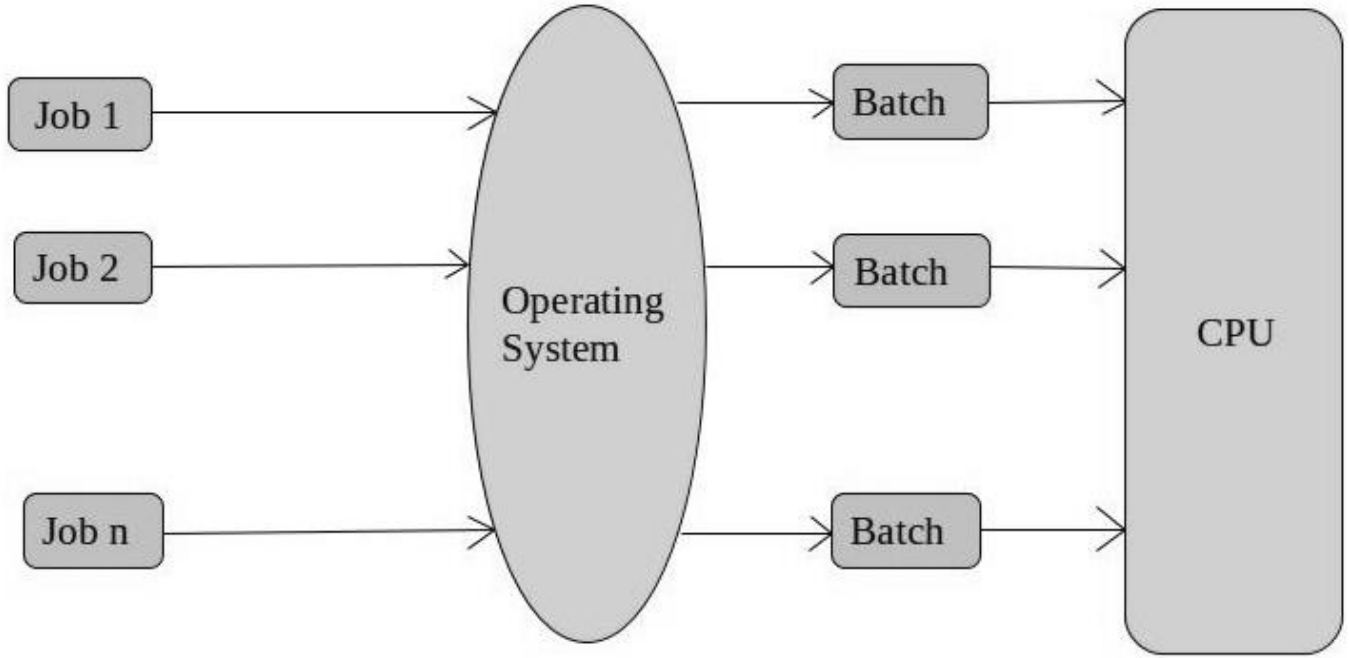


Fig 1.3: Batch Operating System

Time-sharing Operating System

- This type of OS allows many users to share the computer resources. Here, each task is given some time to execute so that all the tasks work smoothly. It is also known as **multitasking system** which can ensure the maximum utilization of the resources.
- Examples: Multics, Unix

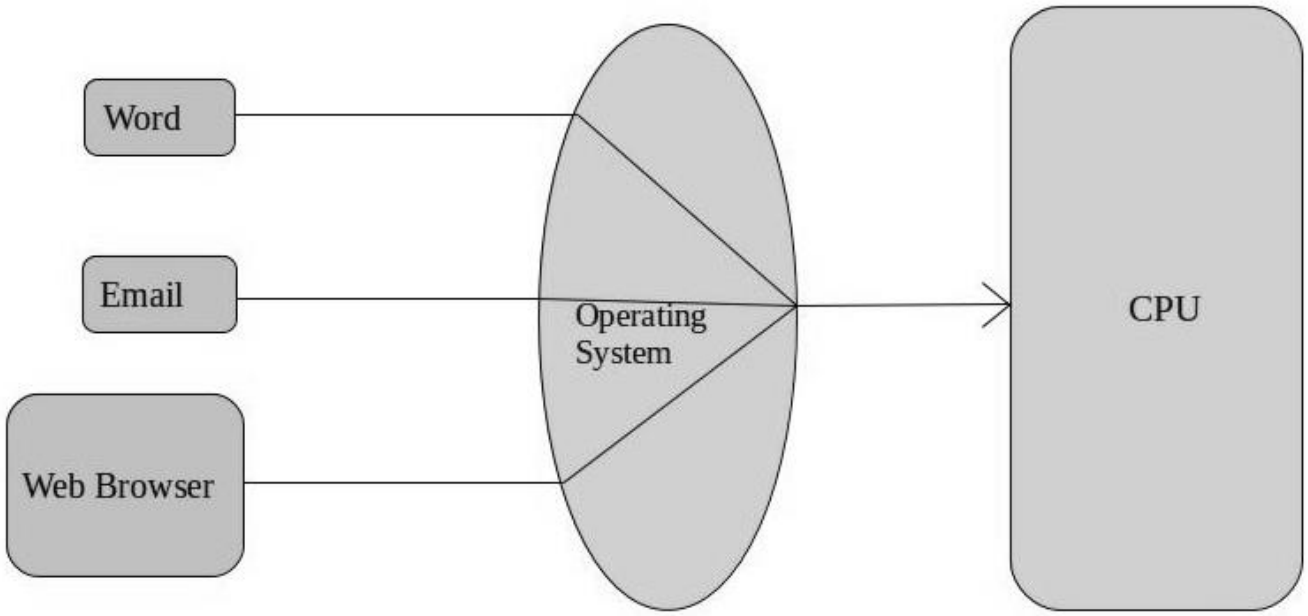


Fig 1.4: Time-sharing Operating System

Distributed Operating System

- This type of OS manages a group of different computers and makes appear to be a single computer. It is also known as **loosely coupled system**.
- Examples: LOCUS

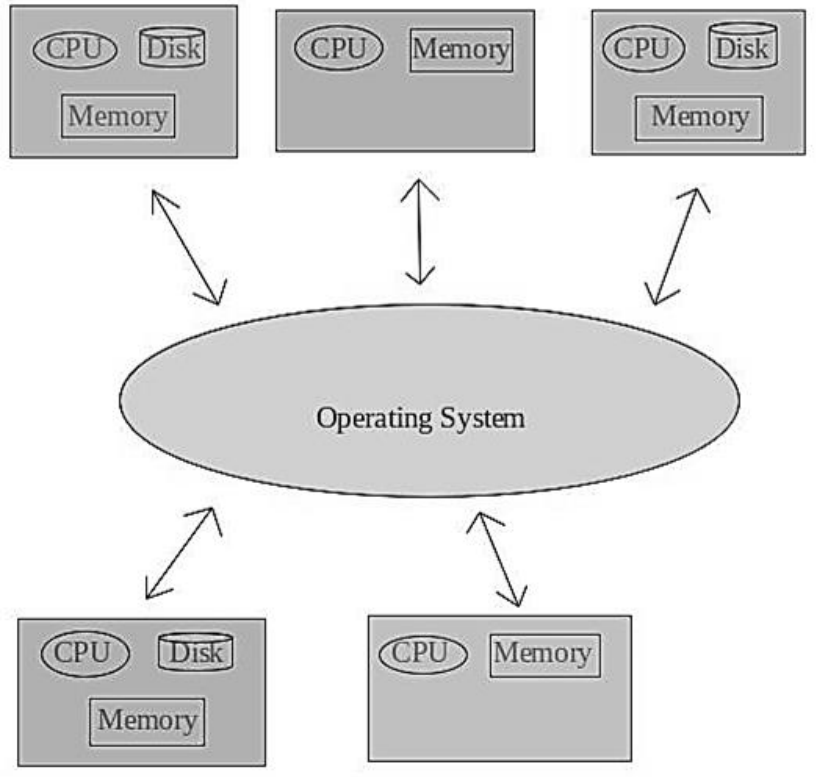


Fig 1.5: Distributed Operating System

Network Operating System

- This type of OS runs on a server and provides the capability to manage data, users, groups, security, applications, and other networking functions. It allows shared access of files, printers, and other networking functions over a small private network. It is also known as **tightly coupled system** and use for security purposes.
- Examples: Microsoft Windows Server (2003, 2008), UNIX, Linux, Mac OS X

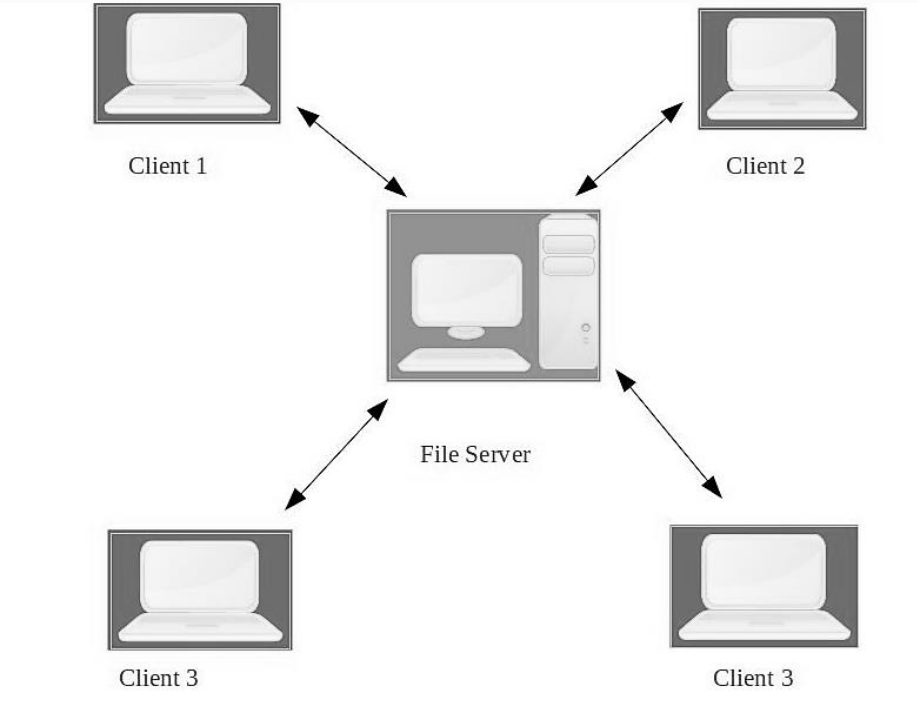


Fig 1.6: Network Operating System

Real-time Operating System

- This type of OS serves real-time systems where time requirements are very strict like medical imaging systems, air traffic control systems, weapon system, robots, etc.

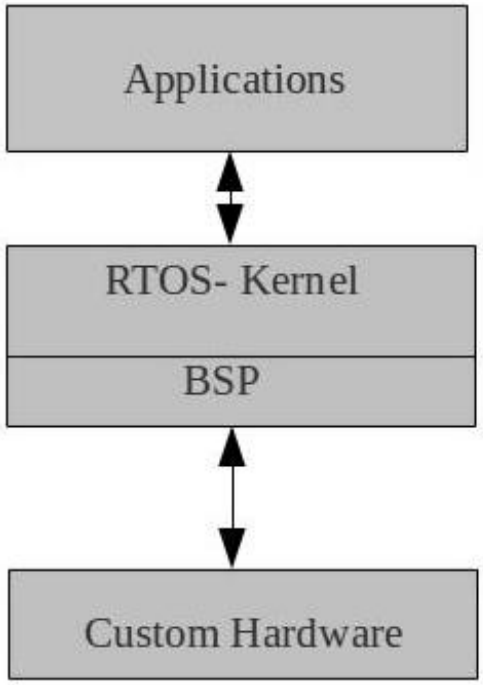


Fig 1.7: Real-time Operating System

Computer Startup

- **POST** (Power On Self Test) happens each time when computer is turning on. It initializes the various hardware devices and checks the bios.
- Boot Process starts with the **POST** and ends when the Bios searches for the **MBR** on the Hard Drive.
- **MBR** (Master Boot Record) is a small program that starts when the computer is booting, in order to find the operating system. Means, **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
- Initializes all aspects of system.
- Loads operating system kernel and starts execution

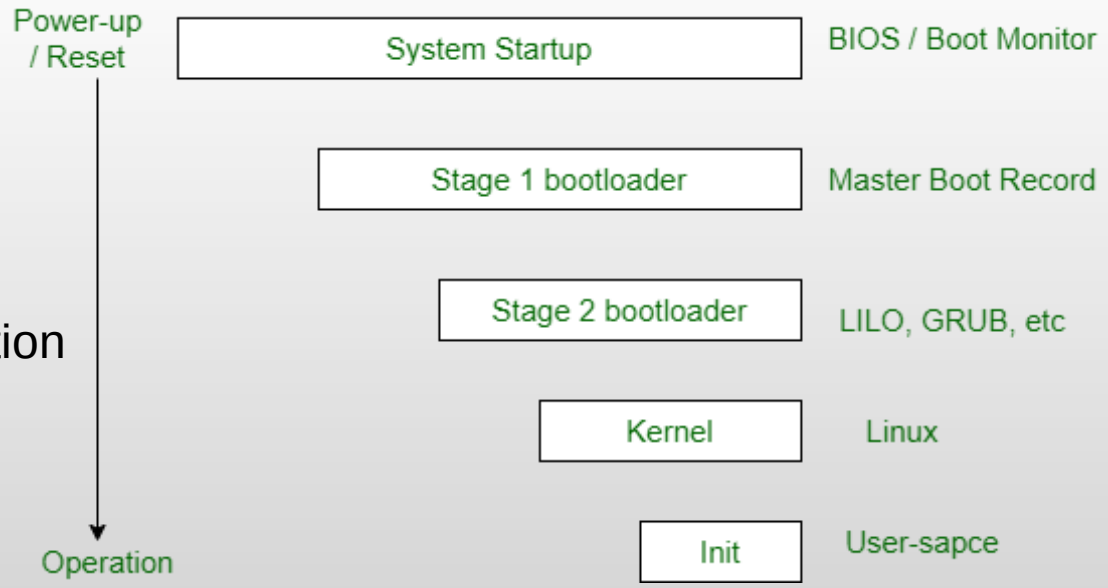


Fig 1.8: An overview of the boot process

Computer System Operation

- One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles

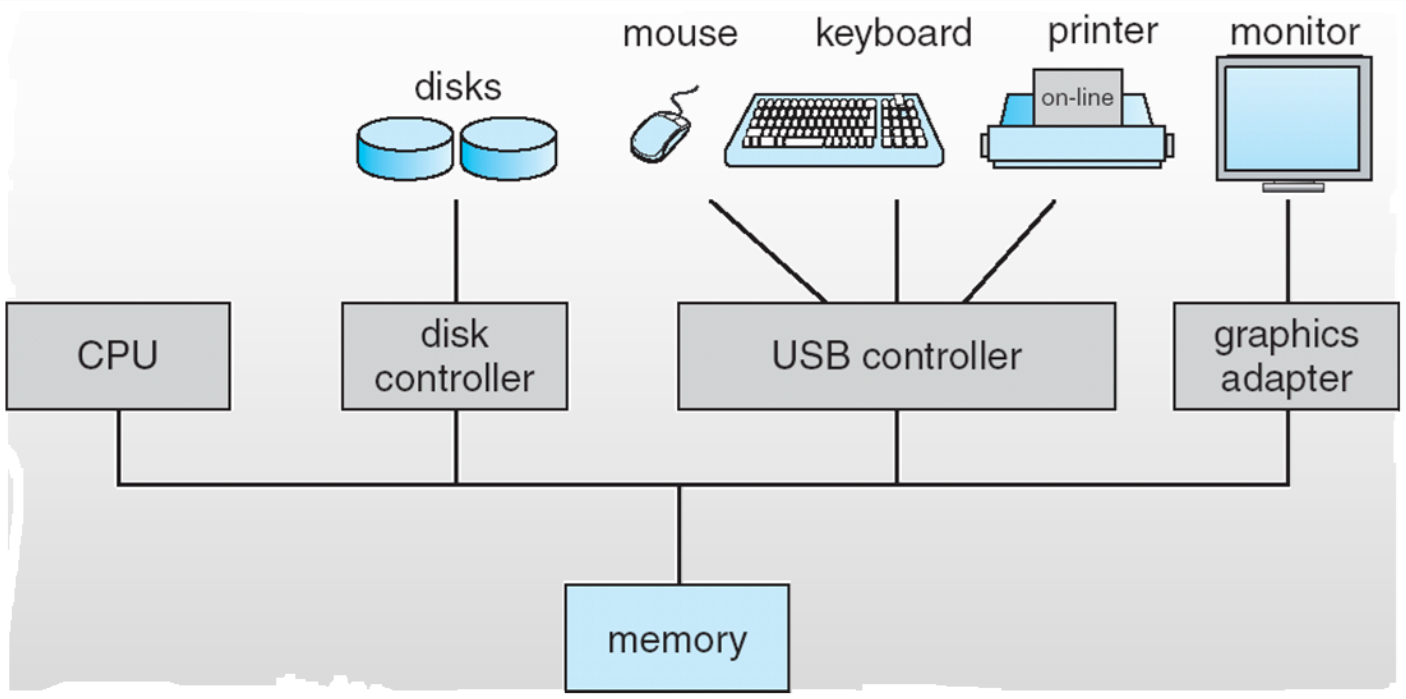


Fig 1.9: Device Controllers in Computer System

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

Common Functions of Interrupt

- **Interrupt Handling**:
  - The operating system preserves the state of the CPU by storing registers and the program counter
  - Separate segments of code determine what action should be taken for each type of interrupt

- The interruption is temporary, and, after the interrupt handler finishes, the processor resumes normal activities.

- There are two types of **interrupt**s: hardware interrupts and software interrupts.
  - Hardware interrupts are used by devices to communicate that they require attention from the operating system.
  - A trap or exception is a software-generated interrupt caused either by an error or a user request.

Common Functions of Interrupt

- Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines
- Interrupt architecture must save the address of the interrupted instruction
- An operating system is **interrupt driven.**

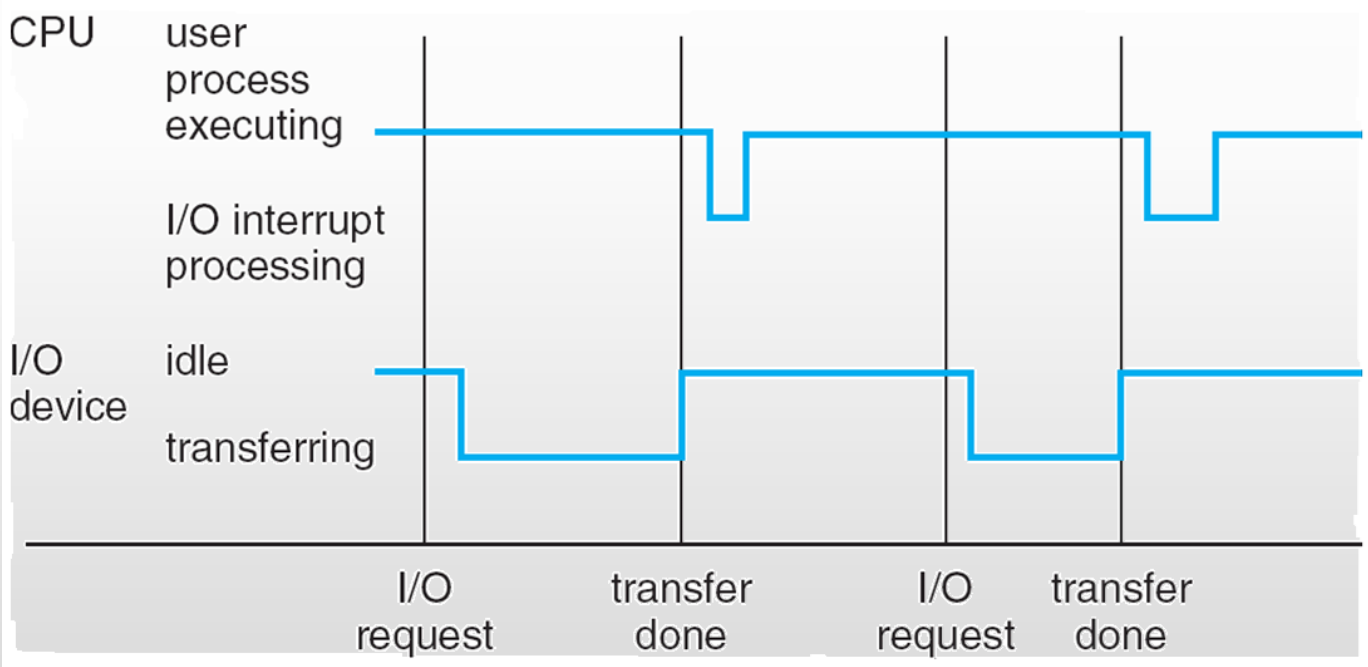


Fig 1.10: Interrupt Timeline

I/O Structure

- After I/O starts, control returns to user program only upon I/O completion
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access)
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing

- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** – request to the OS to allow user to wait for I/O completion
  - **Device-status table** contains entry for each I/O device indicating its type, address, and state
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt

Storage Structure

- **Main memory** – only large storage media that the CPU can access directly
  - Random access, Typically volatile
- **Secondary storage** – extension of main memory that provides large nonvolatile storage capacity
  - **Magnetic disks** – rigid metal or glass platters covered with magnetic recording material
    - Disk surface is logically divided into tracks, which are subdivided into sectors
    - The **disk controller** determines the logical interaction between the device and the computer
  - **Solid-state disks (SSD)** – It uses integrated circuit assemblies to store data persistently, typically using flash memory.
    - faster than magnetic disks & becoming more popular
- **Caching** – copying information into faster storage system; main memory can be viewed as a cache for secondary storage
- **Device Driver** for each device controller to manage I/O
  - Provides uniform interface between controller and kernel

Storage Structure

- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility & Capacity

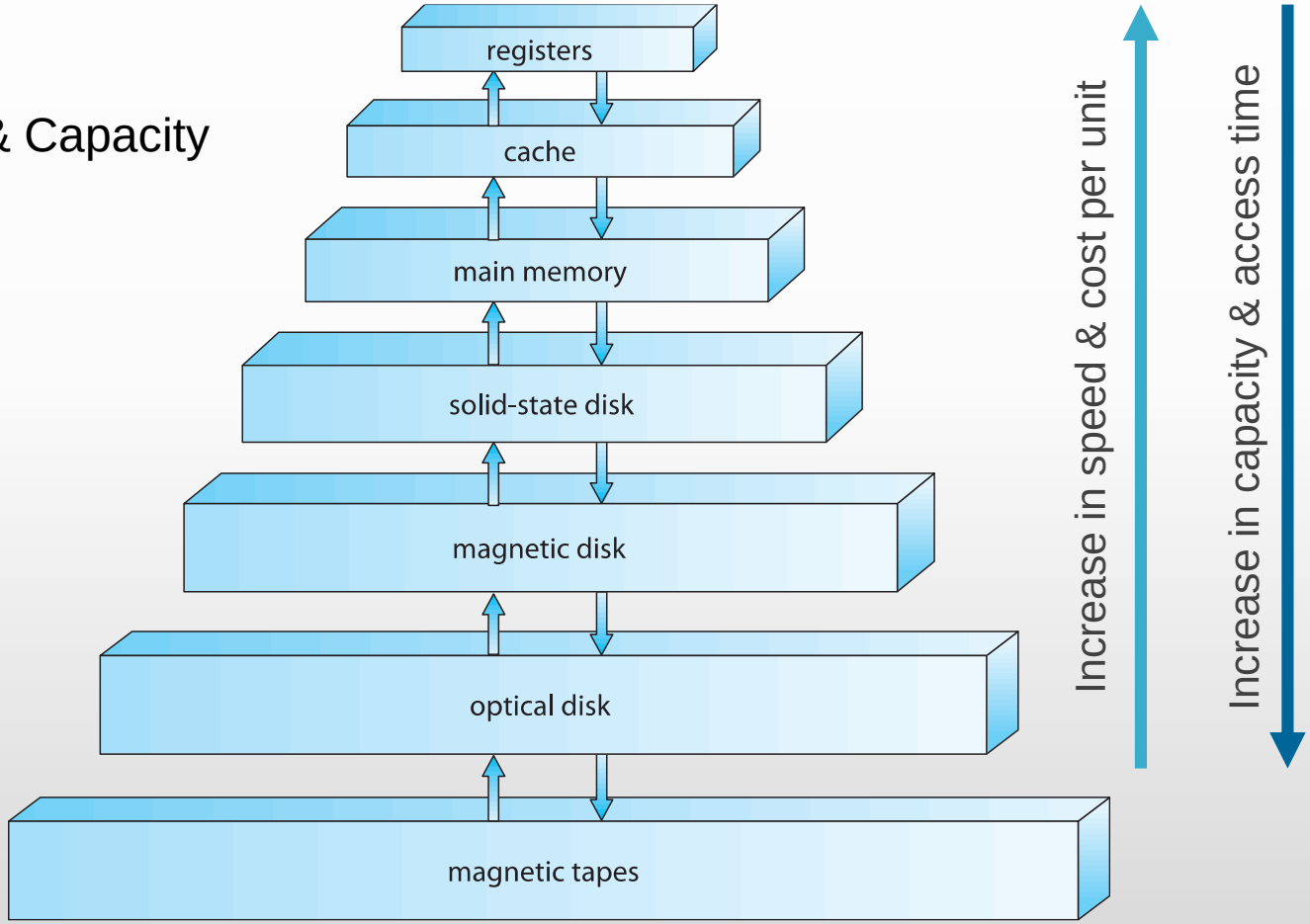


Fig 1.11: Storage Hierarchy Design

How a Modern Computer Works

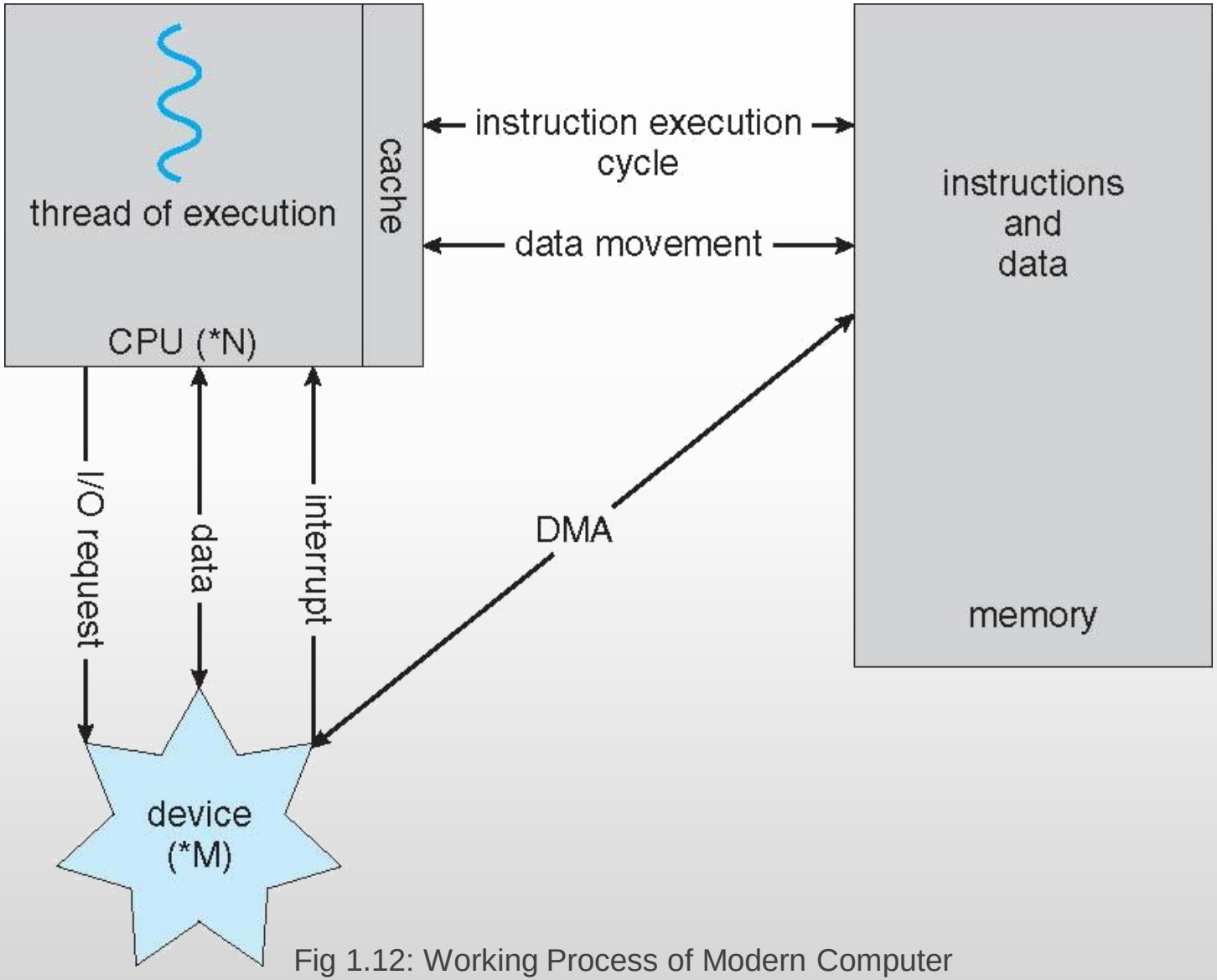


Fig 1.12: Working Process of Modern Computer

Computer System Architecture

- **Single Processor System**
  - It has **single general-purpose processor** (PDAs through mainframes)
  - Most systems use single processor systems.
  - They perform only one process at a given time, and it carries out the next process in the queue only after the current process is completed.
  - OS monitors the status of them and also sends them next executable instruction.
  - It relieves CPU of disk scheduling and other tasks.
  - It is suitable for general purpose computers , as it cannot run multiple processes in parallel.

Computer System Architecture

- **Multi Processor System**
  - Also known as parallel systems, tightly-coupled systems as they can run multiple process in parallel to each other efficiently.
  - Two or more processors will be in close communication with each other with shared memory, storage and power supply.
  - **Advantages:**

1. Increased throughput: As there are a number of processors, more work can be done in less time. These multiple processors run parallel to each other increasing the performance of the system.

1. Economy of scale: Multi Processor Systems cost less than a number of individual single processor system. In the case of multi processor system expenditure for system cabinet, memory power supply, accessories are saved as these systems share resources like power supply, memory and also space.

1. Reliability and failure-free: Failure of any processor will not affect the functionality of the system, as there are a number of processors. We can expect failure free service from multi-processor system.

Computer System Architecture

- **Multi Processor System**

  Two Types

  - **Asymmetric Multiprocessing**
  - **Symmetric Multiprocessing**

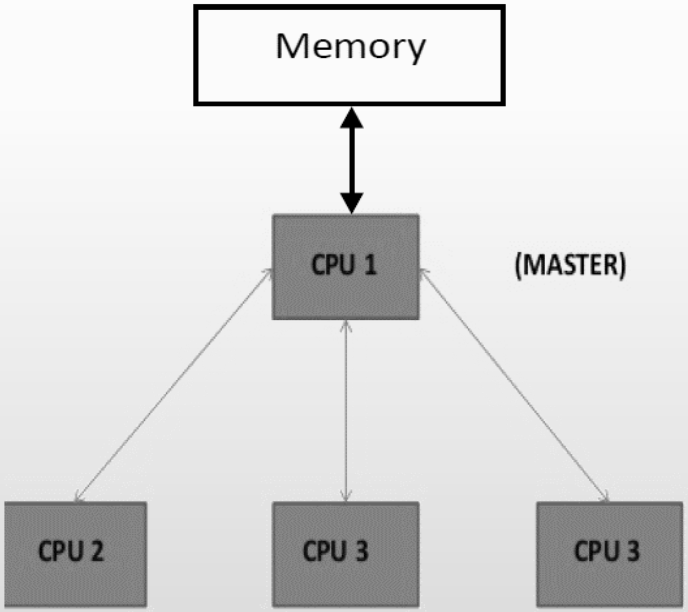


Fig 1.13: Asymmetric Multiprocessing

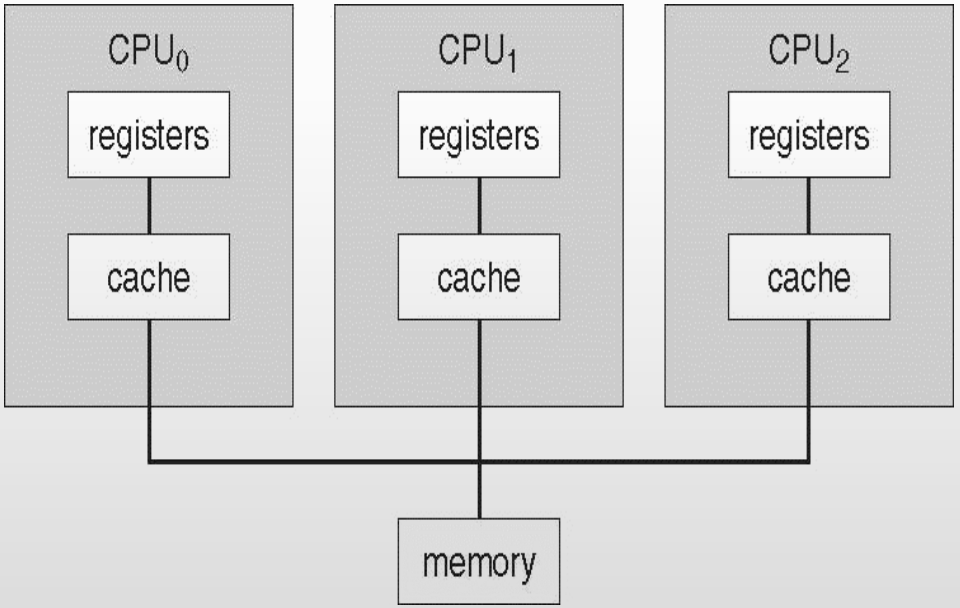


Fig 1.14: Symmetric Multiprocessing

Computer System Architecture

- **Clustered System**
  - Like multiprocessor systems, but multiple systems working together
  - Usually sharing storage via a **storage-area network (SAN)**
  - Provides **a high-availability** service which survives failures
  - Some clusters are for **high-performance computing (HPC):** Applications must be written to use parallelization
  - Some have **distributed lock manager (DLM)** to avoid conflicting operations

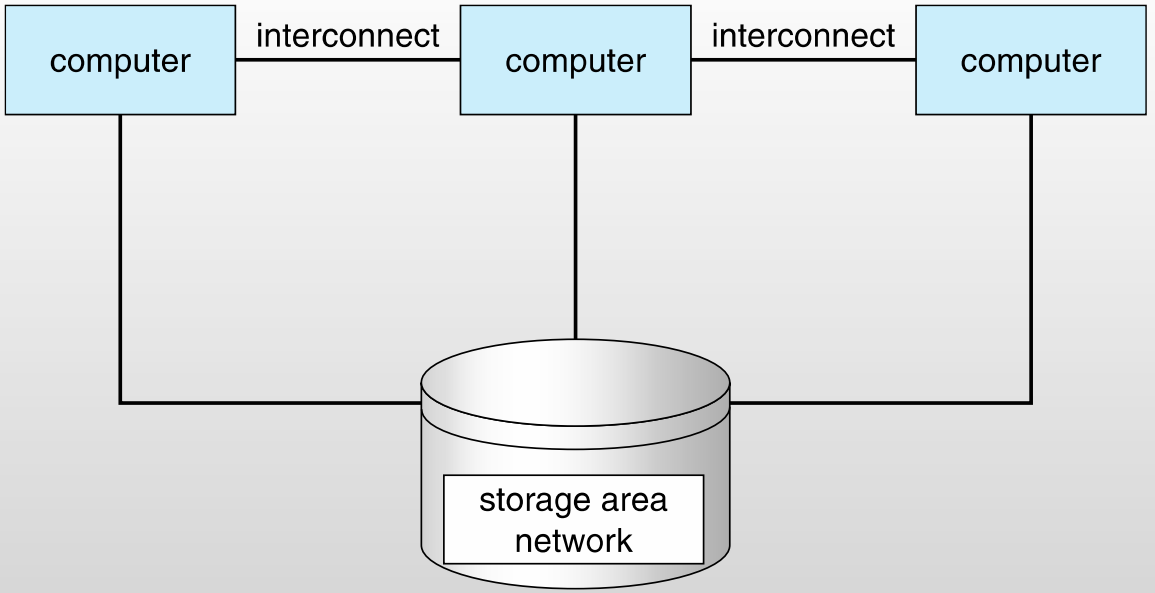


Fig 1.15: General structure of a clustered system

Operating System Structure

- **Multiprogramming** needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via job scheduling
  - The processor executes the job until it is interrupted by some external factor or it goes for an I/O task.
  - When it has to wait (for example I/O), OS switches to another job.
  - Means a computer can run more than one program at a time in multiprogramming system.

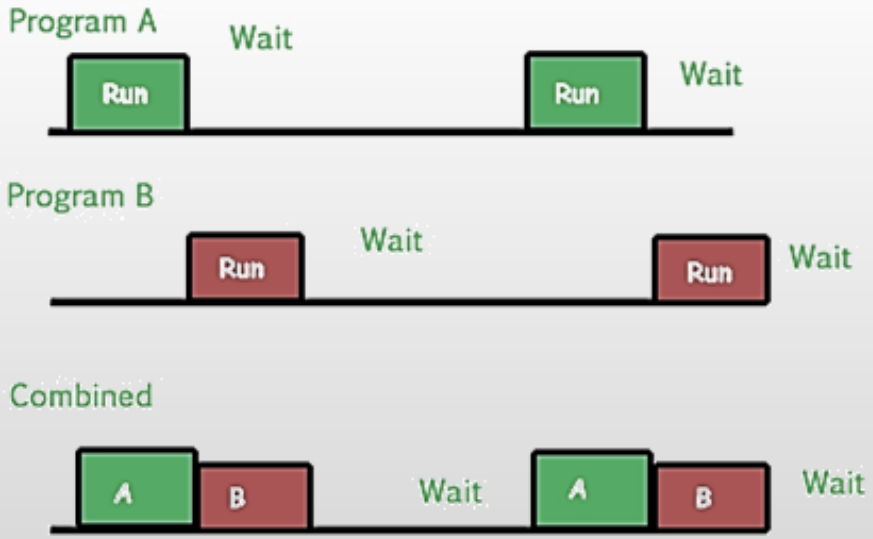


Fig 1.16: Graphical View of Multiprogramming System

Operating System Structure

- **Multitasking (Timesharing)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing.
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨**process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

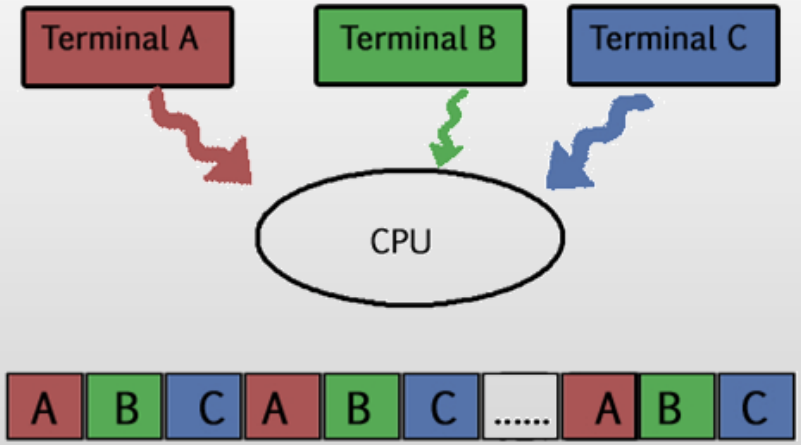


Fig 1.17: Graphical View of Multitasking System

Operating System Structure

- **Multithreading** allows sub-processes (threads) to run concurrently or parallelly.
  - A **thread** is called a lightweight process. Or we may say a process can be divided into sub-processes which are known as threads.
  - For example- VLC media player, where one thread is used for opening the VLC media player, one thread for playing a particular song and another thread for adding new songs to the playlist.

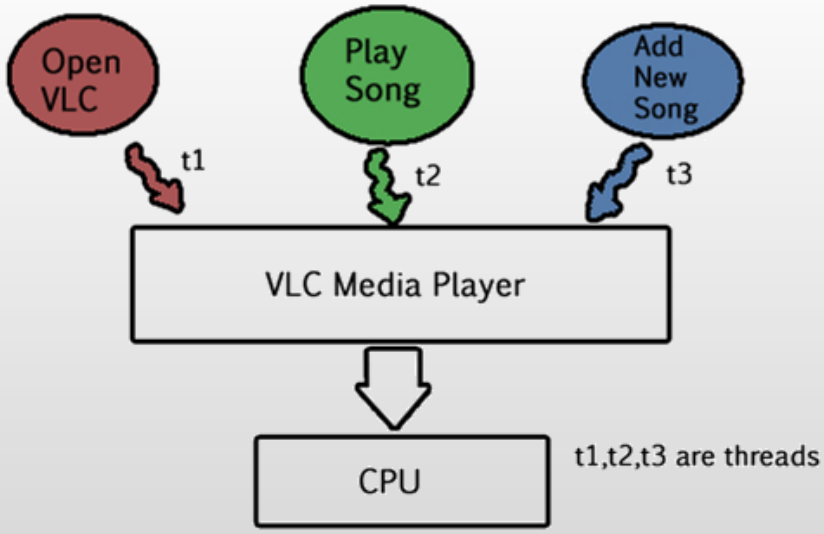


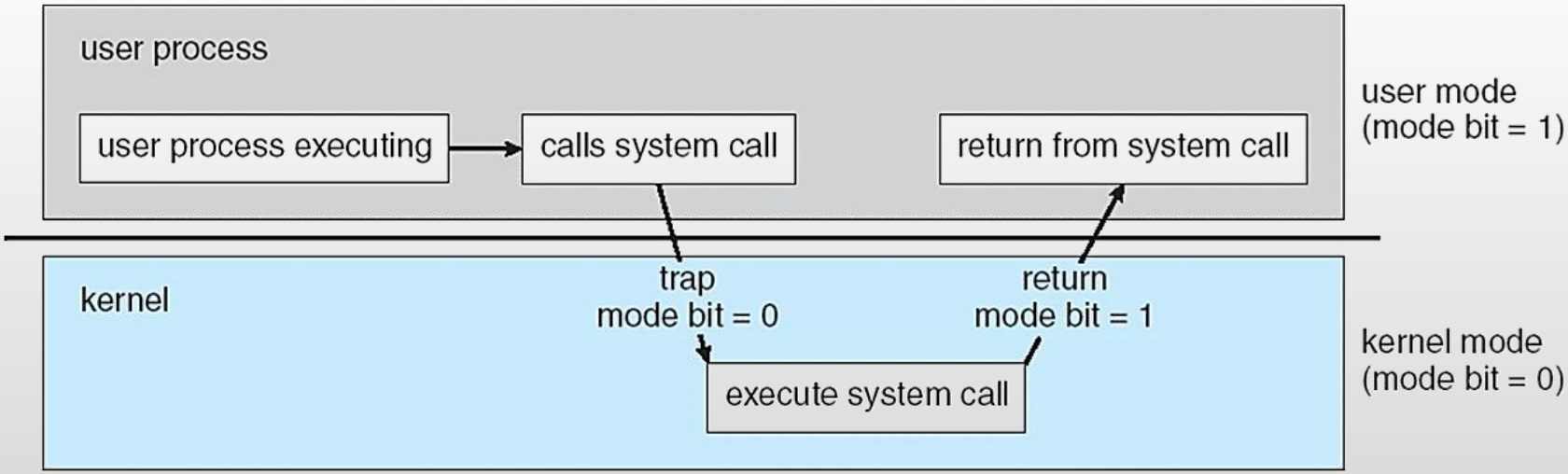
Fig 1.17: Example of Multithreading System

Operating-System Operations

- **Interrupt driven** by hardware
- Software error or request creates **exception** or **trap**
  - Division by zero, request for operating system service
  - Other process problems include infinite loop, processes modifying each other or the operating system
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
  - Provides ability to distinguish when system is running user code or kernel code
  - Some instructions designated as **privileged**, only executable in kernel mode
  - System call changes mode to kernel, return from call resets it to user

Transition from User to Kernel Mode

- **Timer** to prevent infinite loop / process hogging resources
  - Set interrupt after specific period
  - Operating system decrements counter
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time



Process Management

- A process is a program in execution. It is a unit of work within the system.
- Program is a ***passive entity***, process is an ***active entity***.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources

- **Process Management Activities**

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

Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  - Optimizing CPU utilization and computer response to users

- **Memory Management Activities**
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
  - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - 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

Mass-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - 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)

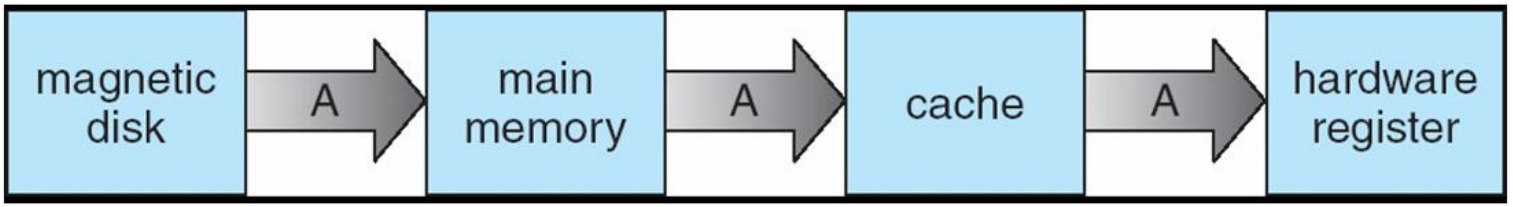
Performance of Various Levels of Storage

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

- Movement between levels of storage hierarchy can be explicit or implicit

Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy



- Multiprocessor environment must provide **cache coherency** in hardware such that all CPUs have the most recent value in their cache

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user

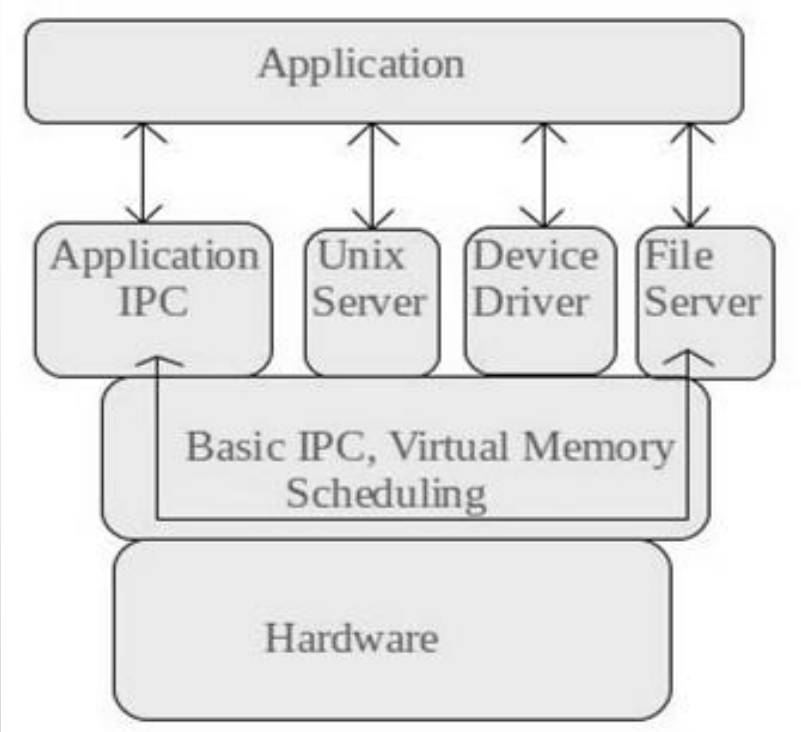
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

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

Microkernel

Microkernel is one of the classification of the kernel. Being a kernel it manages all system resources. But in a microkernel, the **user services** and **kernel services** are implemented in different address space. The user services are kept in **user address space**, and kernel services are kept under **kernel address space**, thus also reduces the size of kernel and size of operating system as well.



Computing Environments - Traditional

- Stand-alone general purpose machines
- But blurred as most systems interconnect with others (i.e. the Internet)
- **Portals** provide web access to internal systems
- **Network computers (thin clients)** are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks

Computing Environments - Mobile

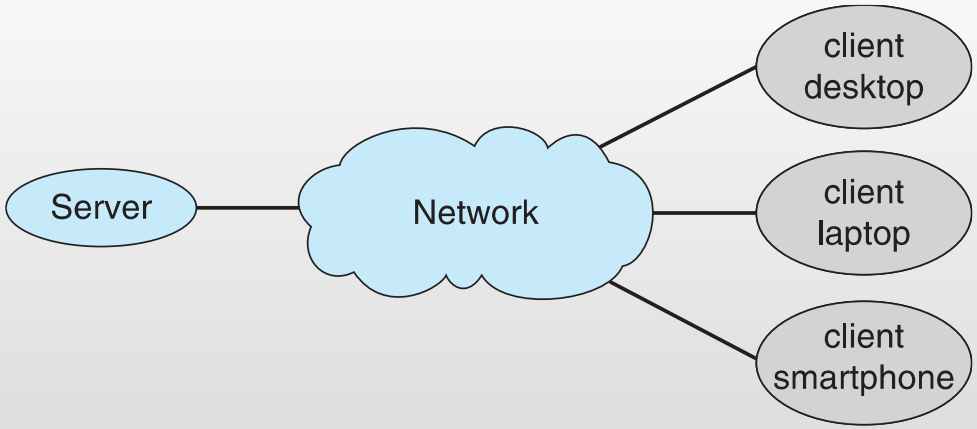
- Handheld smartphones, tablets, etc
- What is the functional difference between them and a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of apps like ***augmented reality***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**

Computing Environments – Distributed

- Distributed
  - Collection of separate, possibly heterogeneous, systems networked together
  - **Network** is a communications path, **TCP/IP** most common
    - **Local Area Network (LAN)**
    - **Wide Area Network (WAN)**
    - **Metropolitan Area Network (MAN)**
    - **Personal Area Network (PAN)**
  - **Network Operating System** provides features between systems across network
  - Communication scheme allows systems to exchange messages
  - Illusion of a single system

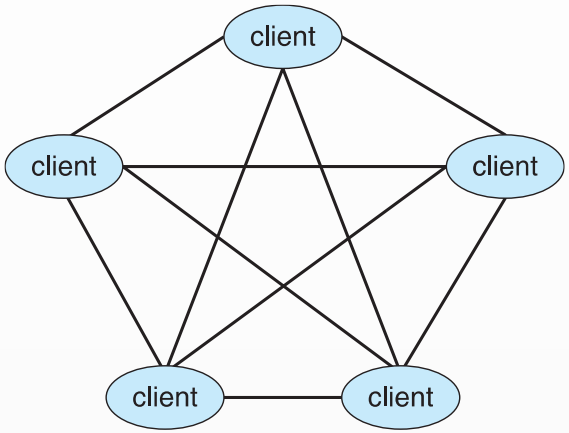
Computing Environments – Client-Server

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server system** provides an interface to client to request services (i.e., database)
    - **File-server system** provides interface for clients to store and retrieve files



Computing Environments - Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both
  - Node must join P2P network



Computing Environments – Real-Time Embedded Systems

- Real-time embedded systems most prevalent form of computers
  - Vary considerable, special purpose, limited purpose OS, **real-time OS**
  - Use expanding
- Many other special computing environments as well
  - Some have OSes, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
  - Processing ***must*** be done within constraint
  - Correct operation only if constraints met

Open-Source Operating Systems

- Operating systems made available in source-code format rather than just binary **closed-source**
- Counter to the **copy protection** and **Digital Rights Management (DRM)** movement
- Started by **Free Software Foundation (FSF)**, which has “copyleft” **GNU Public License (GPL)**
- Examples include **GNU/Linux** and **BSD UNIX** (including core of **Mac OS X**), and many more
  - Can use VMM like VMware Player (Free on Windows), Virtualbox (open source and free on many platforms - http://www.virtualbox.com)
  - Use to run guest operating systems for exploration

Few Other Topics

- 32-bit and 64-bit operating systems
- UEFI(Unified Extensible Firmware Interface) and BIOS (basic input/output system)
- Dual Mode operations in OS

Question & Discussion

Task Assign

Thank You

kohinoor_cse@lus.ac.bd