# Introduction to Operating Systems


**Sunitha C S**
**Software Training and Development Centre, Kochi**

# Text Books

Operating Systems Principles by Abraham Silberschatz, Peter Galvin & Greg Gagne / Wiley

https://os-book.com/OS9/index.html

Unix Concepts and Applications by Sumitabha Das / McGraw Hill

**References:**

- Modern operating Systems by Andrew Tanenbaum & Herbert Bos/ Pearson
- Principles of Operating Systems by Naresh Chauhan / Oxford University Press
- Beginning Linux Programming by Neil Matthew & Richard Stones / Wrox
- Operating System : A Design-Oriented Approach by Charles Crowley / McGraw Hil

# What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the  computer hardware
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer hardware system convenient to use
  - Use the computer hardware in an efficient manner

# What Operating Systems Do

- The operating system **acts as the interface** between the user and the computer hardware. It allows users to enter data, process, and access the results. Moreover, users can interface with computers via the operating system to perform various functions like arithmetic calculations and other important tasks.
- The operating system **allows a user-friendly graphical interface** for all clients since it provides different menus, symbols, catches, and more for simple routes.
- Operating systems enable **data and relevant information to be shared** with other users through Fax Machines, Printers, and Modems. Furthermore, a single user may transfer the same data with several users simultaneously through email. Also, many apps, photos, and media files can be moved from a PC to another device using an operating system.
- An operating system can **manage several jobs** at the same time. It enables users to perform several tasks simultaneously.

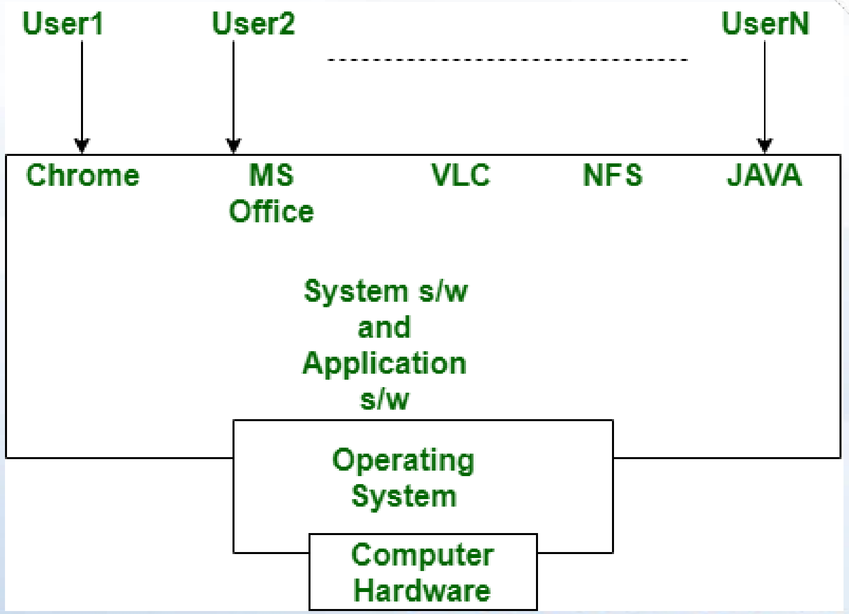
Depends on the point of view

- Users want convenience, **ease of use** and **good performance**
  - Don’t care about **resource utilization**
- But shared computer such as **mainframe** or **minicomputer** must keep all users happy
  - Operating system is a **resource allocator** and **control program** making efficient use of HW and managing execution of user programs
- Users of dedicate systems such as **workstations** have dedicated resources but frequently use shared resources from **servers**
- Mobile devices like smartphones and tables are resource poor, optimized for usability and battery life
  - Mobile user interfaces such as touch screens, voice recognition
- Some computers have little or no user interface, such as embedded computers in devices and automobiles
  - Run primarily without user intervention

# Difference between OS and other Application Software

- An operating system is system software that acts as an interface between the user and the hardware, whereas application software is a program that performs a specific task.
- It is impossible to install the application software on a computer system without an operating system.
- The entire process or functionality of computer system depends on the operating system.
- In simple terms, Operating software is like the ground or foundation where the application software is the building.

Eg: When the user commands the application's software, say VLC media player, the application software communicates the information to the Operating system, and then the OS uses hardware resources (speaker) to play the music.



# OS is hardware dependent

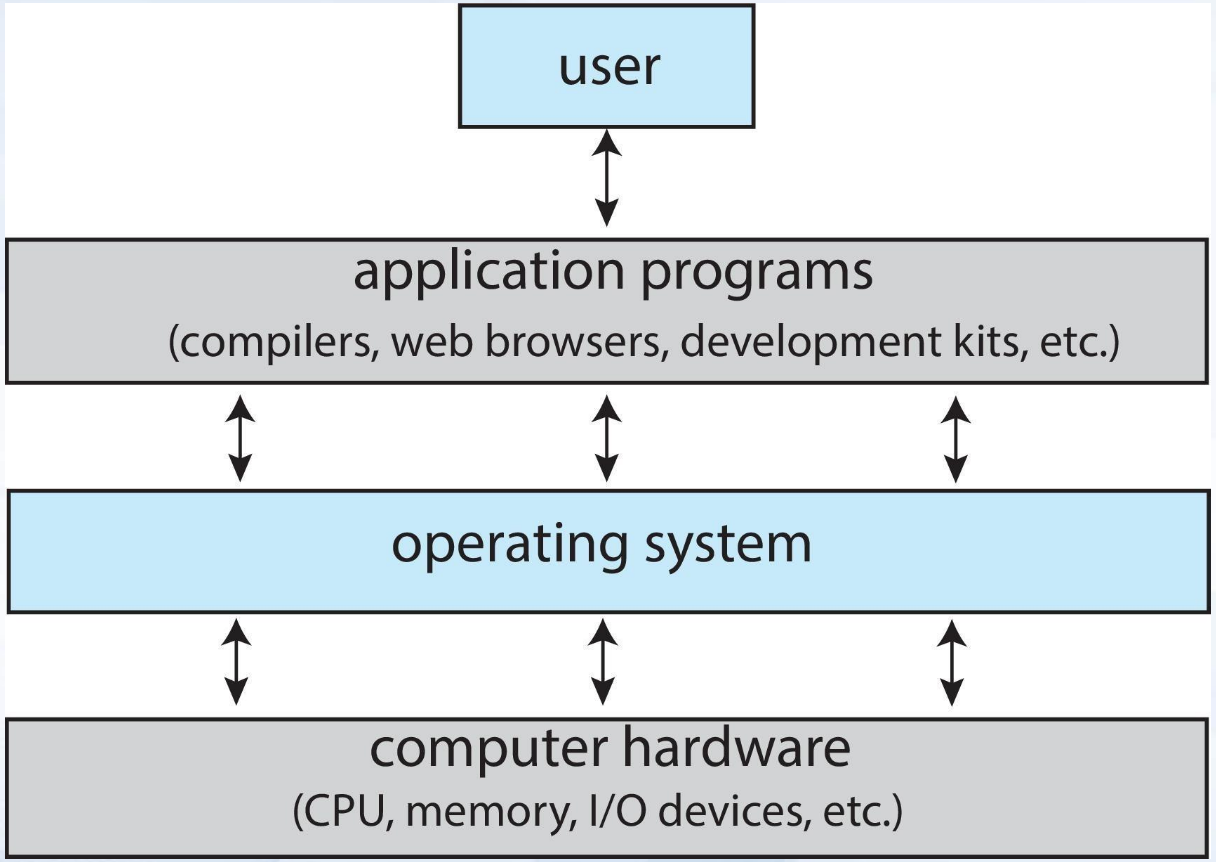
- Most of the OS is always bound to the hardware.
- Any operating system is at least depending on one piece of hardware: the CPU. There are different CPUs, each working differently and having a different "native language". Since an OS is "just a program" which needs to run on the CPU, it must be written in the CPUs native language is thus dependent on it.

# Computer Organization

# Computer System Structure

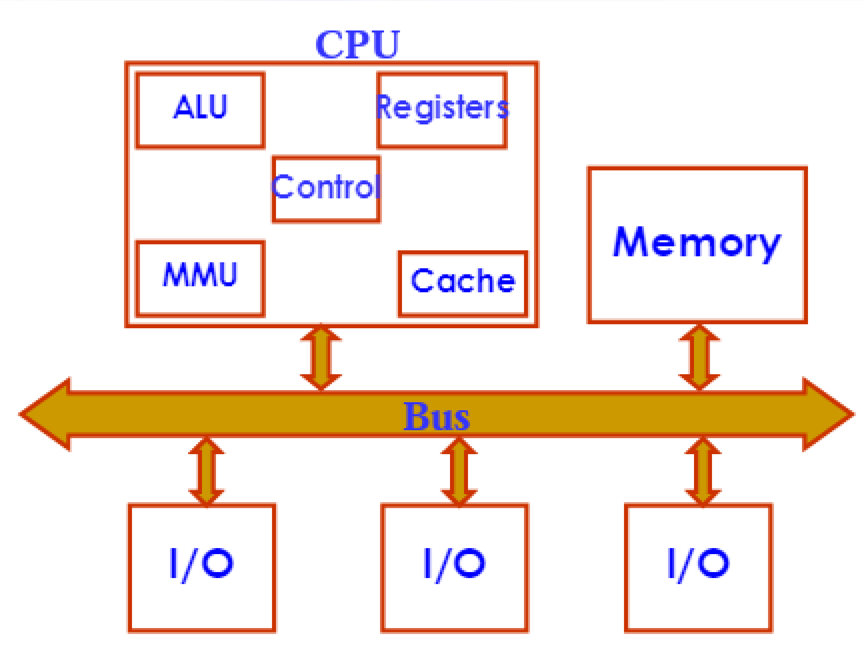
- Computer system can be divided into four components:
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system 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

# Abstract View of Components of Computer



# Computer Organisation

- Main Parts of a Computer System
  - Processor – Executes programs
  - Main Memory – Holds program and Data
  - I/O Devices – For communication with outside



# Inside the Processor…

- Hardware to manage instruction execution
- Arithmetic, logic hardware
- **Registers**: small units of memory to hold data/instructions temporarily during execution
- Two kinds of registers

  1. Special purpose registers
  2. General purpose registers

# Special Purpose Registers

- **Program Counter(PC):** specifies location in memory of instruction being executed
- **Instruction Register(IR):** holds that instruction
- **Processor Status Register**: holds status information about current state of processor, such as whether an arithmetic overflow has occurred, etc

# General Purpose Registers

- Available for use by programmer, possibly for keeping frequently used data
- Why? – Since there is a large speed disparity between processor and main memory
  - 1 GHz Processor: 1 nanosecond time scale
  - Memory: ~ 50 -100 nsec time scale
- Instruction operands can come from registers or from main memory

# How instruction is executed

- A typical instruction–execution cycle, is 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 has been executed, the result may be stored back in memory.

# Main Memory

- Holds instructions and data
- View as sequence of locations, each referred to by a **unique memory address**
- If size of each memory location is 1 Byte, we call the memory **byte addressable**
- This is quite typical, as smallest data (character) is represented in 1 Byte
- Larger data items are stored in contiguous memory locations, e.g., a 4Byte integer would occupy 4 consecutive memory locations

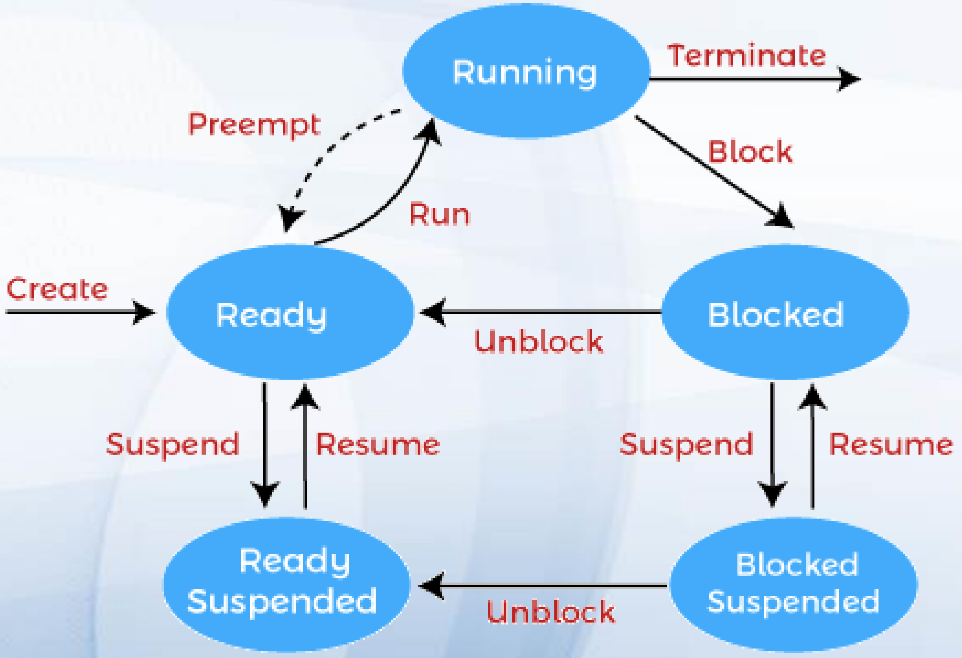
# Functions of Operating Systems

# Functions of Operating System

- An operating system is a large and complex system that can only be created by partitioning into small parts. These pieces should be a well-defined part of the system, carefully defining inputs, outputs, and functions.
- The components of an operating system play a key role to make a variety of computer system parts work together. There are different components of an operating system, such as:
  - Process Management
  - File Management
  - Network Management
  - Main Memory Management
  - Secondary Storage Management
  - I/O Device Management
  - Security Management
  - Command Interpreter System

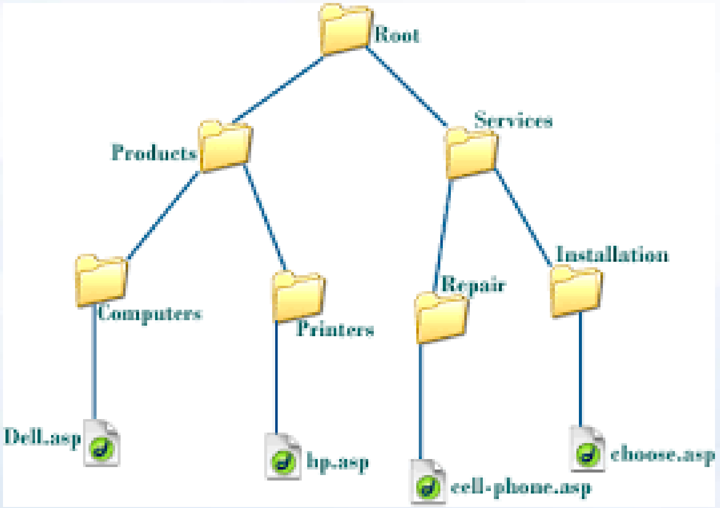
# Process Management

- The process management component is a procedure for managing many processes running simultaneously on the operating system. Every running software application program has one or more processes associated with them.
- For example, when you use a search engine like Chrome, there is a process running for that browser program.
- 



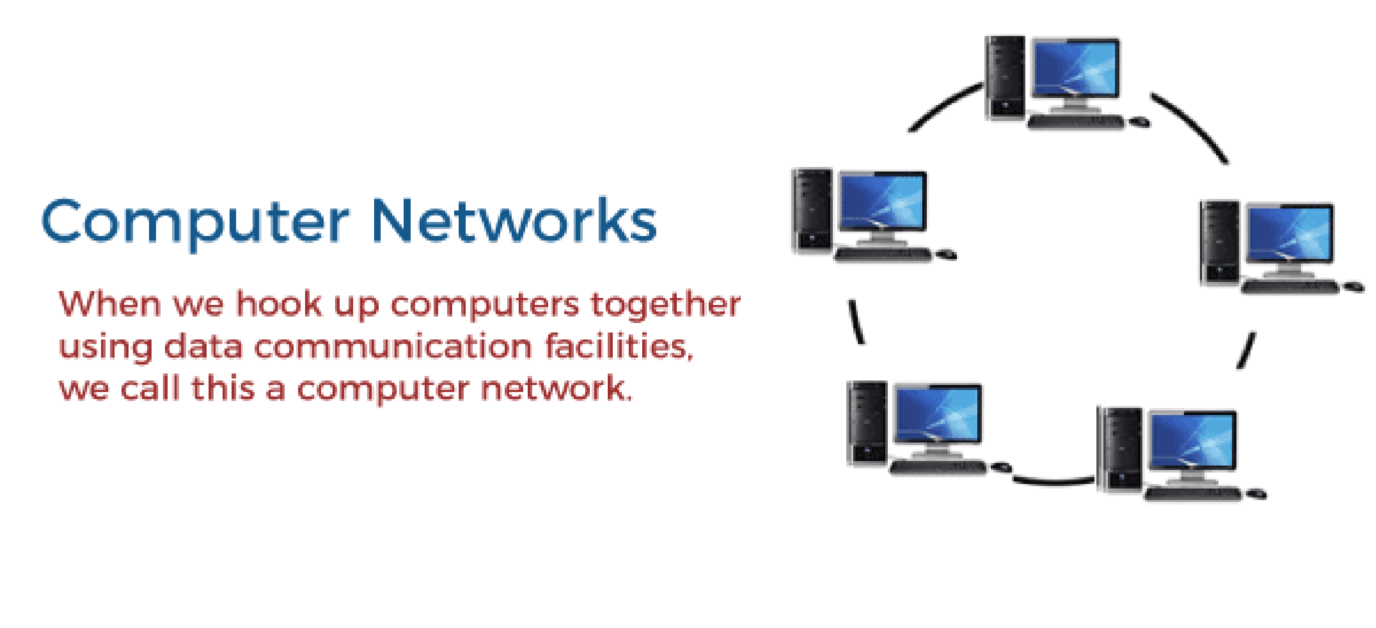
# File Management

- A file is a set of related information defined by its creator. It commonly represents programs (both source and object forms) and data. Data files can be alphabetic, numeric, or alphanumeric.
- The operating system has the following important activities in connection with file management:
    - File and directory creation and deletion.
    - For manipulating files and directories.
    - Mapping files onto secondary storage.
    - Backup files on stable storage media.



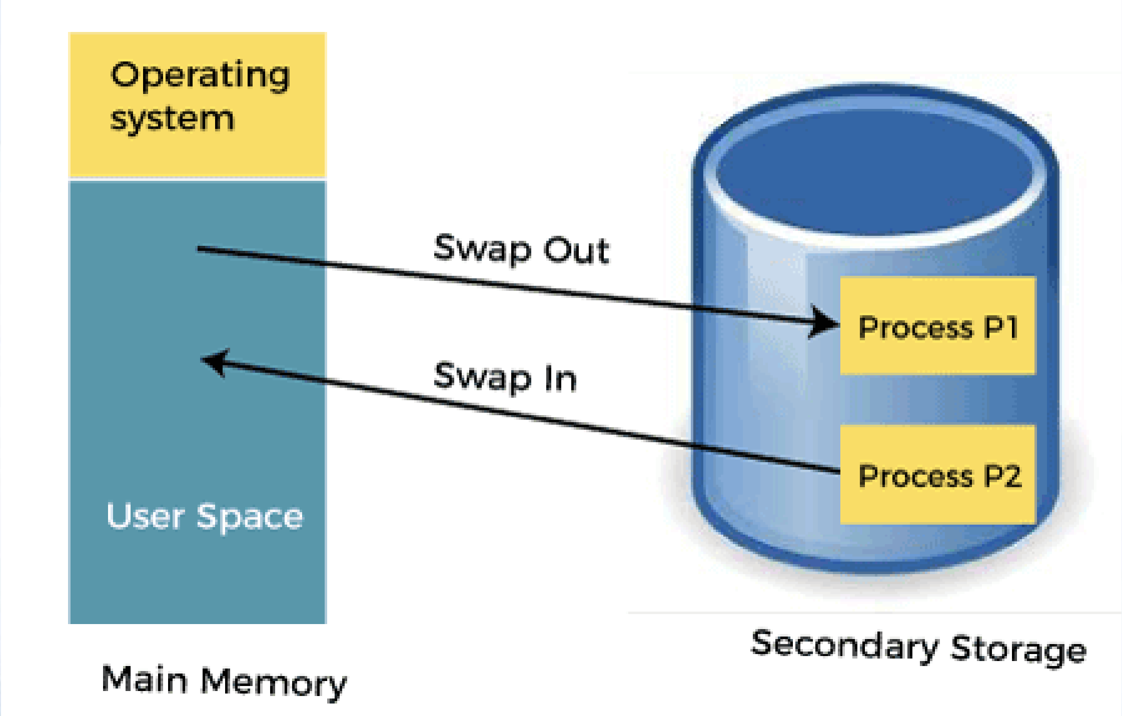
# Network Management

- Network management administers and manages computer networks. Its services include performance management, fault analysis, network provisioning, and service quality management.



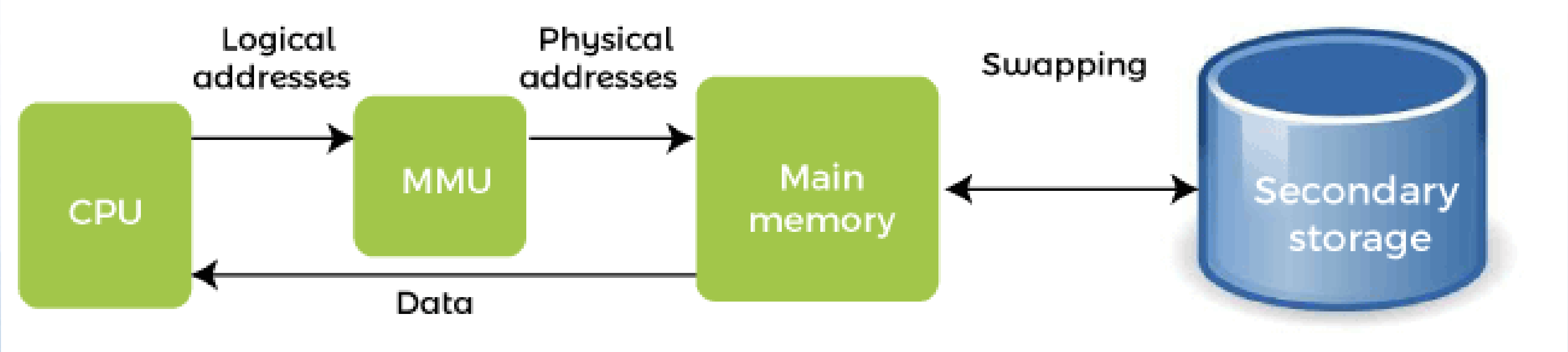
# Main Memory Management

- Main memory is a large array of storage or bytes, which has an address. The memory management process is conducted by using a sequence of reads or writes of specific memory addresses.



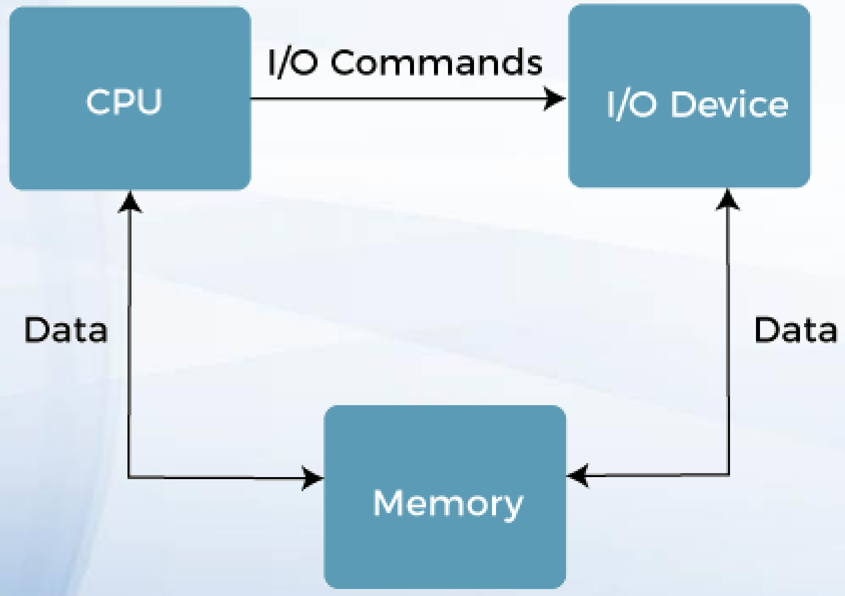
# Secondary Storage Management

- The most important task of a computer system is to execute programs. These programs help you to access the data from the main memory during execution. This memory of the computer is very small to store all data and programs permanently. The computer system offers secondary storage to back up the main memory.
- Today, modern computers use hard drives/SSD as the primary storage of both programs and data. However, the secondary storage management also works with storage devices, such as USB flash drives and CD/DVD drives.



# I/O Device Management

- One of the important use of an operating system that helps to hide the variations of specific hardware devices from the user.

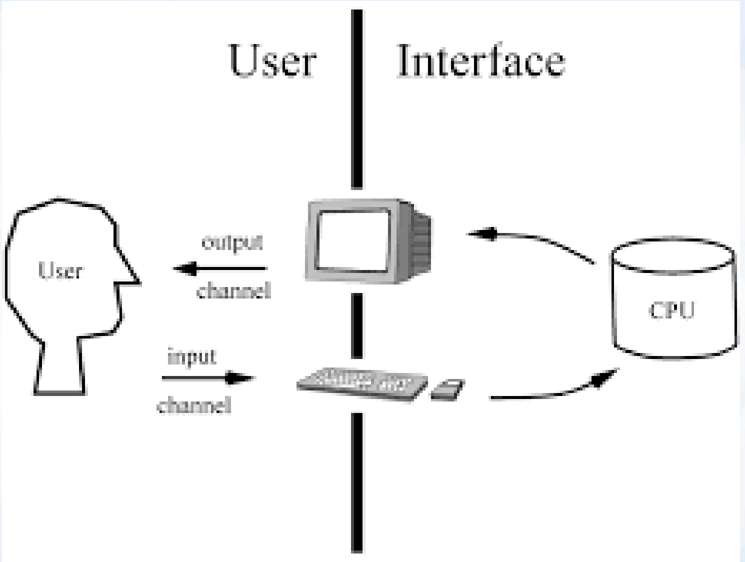


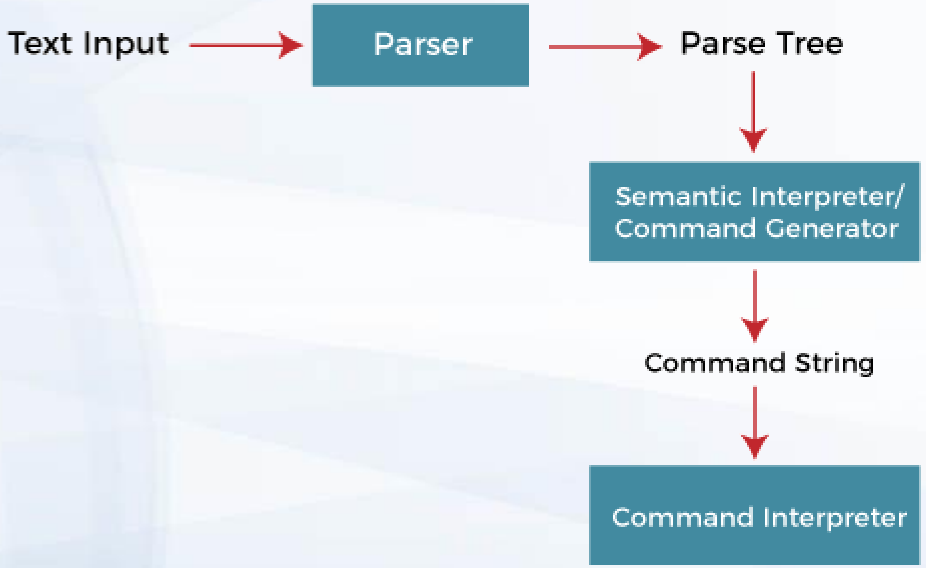
# Security Management

- The various processes in an operating system need to be secured from other activities. Therefore, various mechanisms can ensure those processes that want to operate files, memory CPU, and other hardware resources should have proper authorization from the operating system.
- Security refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by computer controls to be imposed, together with some means of enforcement.

# Command Interpreter System

- One of the most important components of an operating system is its command interpreter. The command interpreter is the primary interface between the user and the rest of the system.



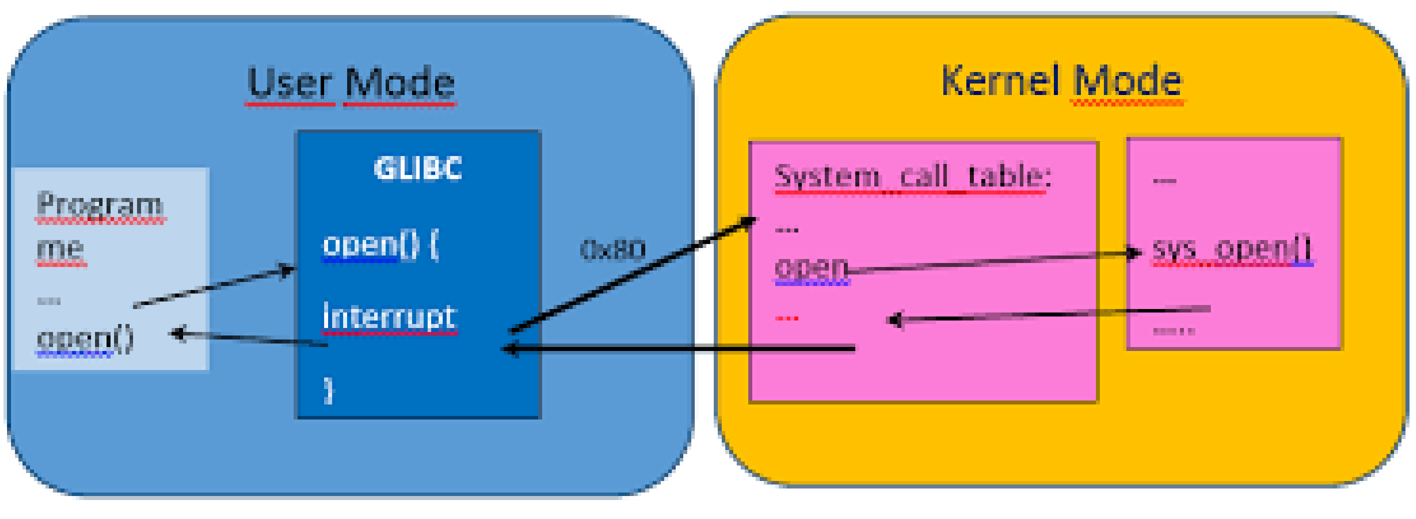



# Examples of well known OS

# Different types of OS

- Desktop OS – The desktop OS is the environment where the user controls a personal computer (Desktop, Notebook PC). It aids in the management of computer hardware and software resources.
- Mobile OS – A **mobile OS** allows application software to operate on mobile devices. Some most famous mobile operating systems are Android , iOS etc..
- Embedded system OS – An embedded operating system is a type of computer OS that is developed to increase the functional efficiency of an embedded system. OS for set-top boxes, ATMs, Smart Tv etc
- Real Time OS – A real time operating system time interval to process and respond to inputs is very small. Examples: Military Software Systems, Space Software Systems are the Real time OS example.
- Server machine OS – It is an operating system that is designed to be used on server. It is used to provide services to multiple client. It can serve multiple client at a time and is very advanced operating system.

# User and Kernel space and Mode



# Dual mode of operation

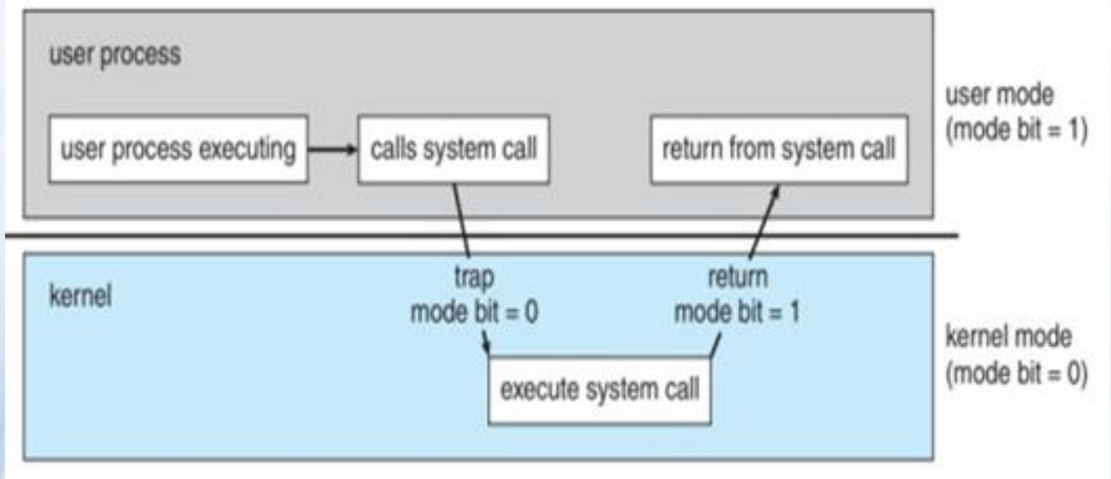
- In order to ensure the proper execution of the operating system, two separate modes of operation are present:

- **User mode** and **Kernel mode** (also called **Supervisor mode**, **System mode**, or **Privileged mode**).

- A bit, called the mode bit, is added to the hardware of the computer to indicate the current mode:

  **kernel (0) or user (1).**

- With the mode bit, we can distinguish between a task that is executed on behalf of the operating system and one that is executed on behalf of the user.



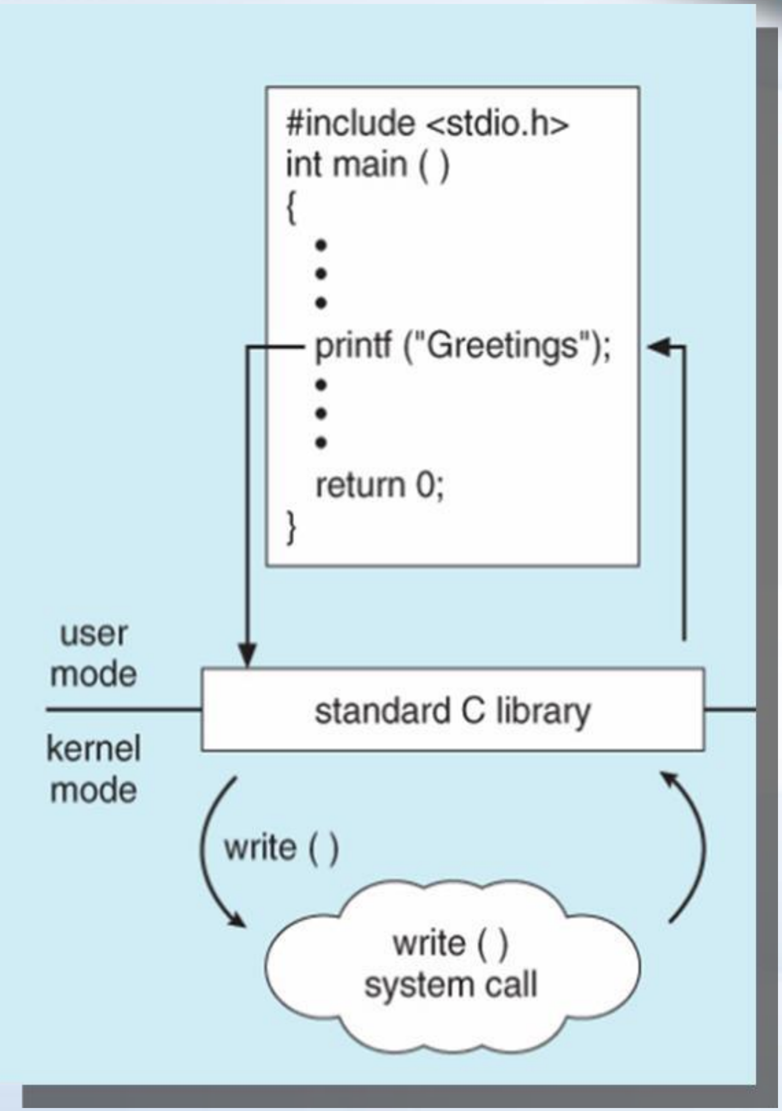

- Whenever a **trap or interrupt occurs**, the hardware switches from user mode to kernel mode

| User mode | Kernel mode |
| --- | --- |
| The operating system puts the CPU in user mode when a user program is in execution | The operating system puts the CPU in kernel mode when it is executing in the kernel |
| When system runs a user application like creating a text document or using any application program, then the system is in user mode. | At system boot time, the hardware starts in kernel mode. on as well.<br>The operating system is then loaded and starts user applications in user mode. |
| When CPU is in user mode, the programs don't have direct access to memory and hardware resources. | The CPU can execute certain instruction only when it is in the kernel mode. **These instruction are called privilege instruction**<br>To provide protection to the hardware, we have privileged instructions which execute only in kernel mode |
| system will be in a safe state even if a program in user mode crashes. | If a program crashes in kernel mode, the entire system will be halted. |

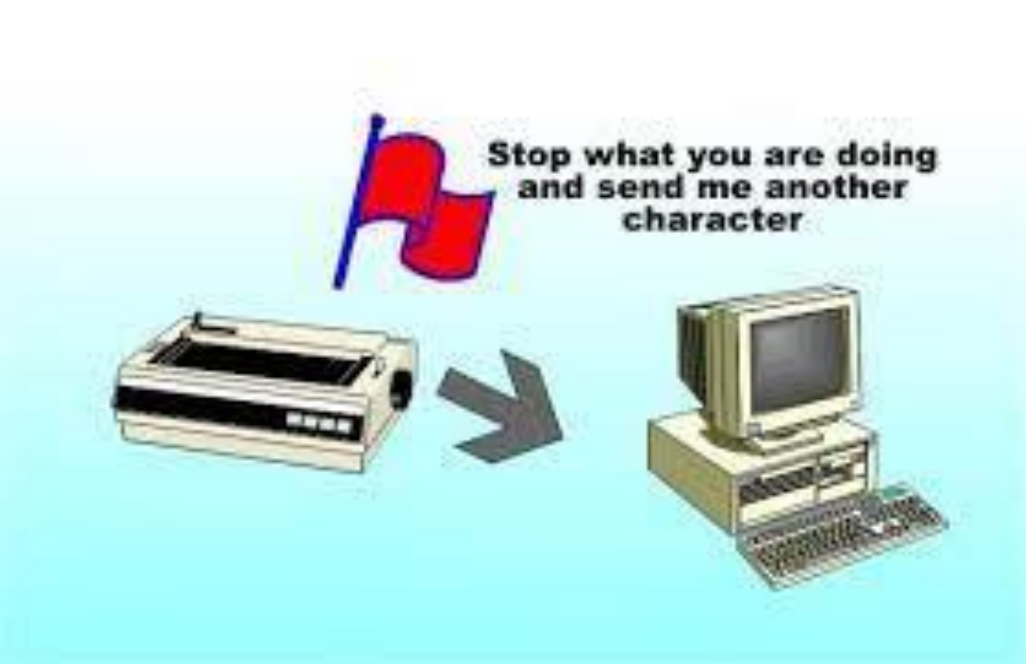
# System Calls

- A system call is the programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on
- When a program in user mode requires access to RAM or hardware resource, it must ask the kernel to provide access to that resource. This is done via something called system call.
- System calls are the only entry points into the kernel system.

- System calls are made by the user level programs in the following situations:
  - Creating, opening, closing and deleting files in the file system.
  - Creating and managing new processes.
  - Creating a connection in the network, sending and receiving packets.
  - Requesting access to a hardware device, like a mouse or a printer.



# Interrupts



# 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
- Each device controller type has an operating system **device driver** to manage it
- 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 Interrupts

- 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
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request
- An operating system is **interrupt driven**

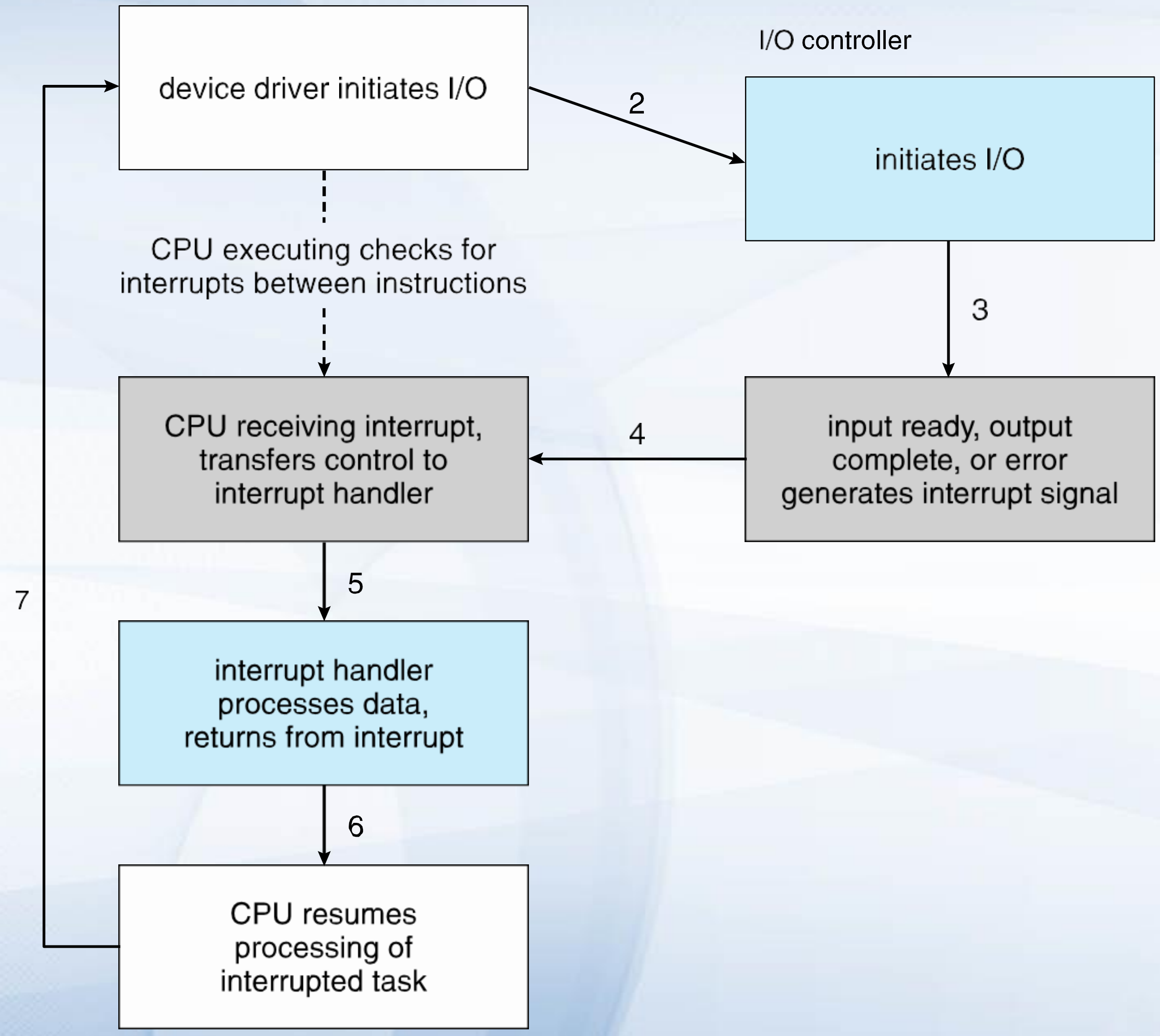
# Interrupt- driven I/O cycle Process

- To start an I/O operation, the device driver loads the appropriate registers in the device controller.
- The device controller, in turn, examines the contents of these registers to determine what action to take.
- The device controller raises an interrupt by asserting a signal on the interrupt request line, the CPU catches the interrupt and dispatches it to the interrupt handler, and the handler clears the interrupt by servicing the device.
  - **Interrupt** is a signal emitted by hardware or software when a process or an event needs immediate attention.
  - **Interrupt request l**ine: The CPU hardware has a wire called the interrupt-request line that the CPU senses after  executing every instruction.
  - **Interrupt handler routine**: When the CPU detects that a controller has asserted a signal on the interrupt-  request line, it reads the interrupt number and jumps to the interrupt-handler routineby using that interrupt  number as an index into the interrupt vector. .

# Interrupt Handling

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

# Interrupt-drive I/O Cycle



# I/O Structure

- Two methods for handling I/O
  - After I/O starts, control returns to user program only upon I/O completion
  - After I/O starts, control returns to user program without waiting for I/O completion

# I/O Structure (Cont.)

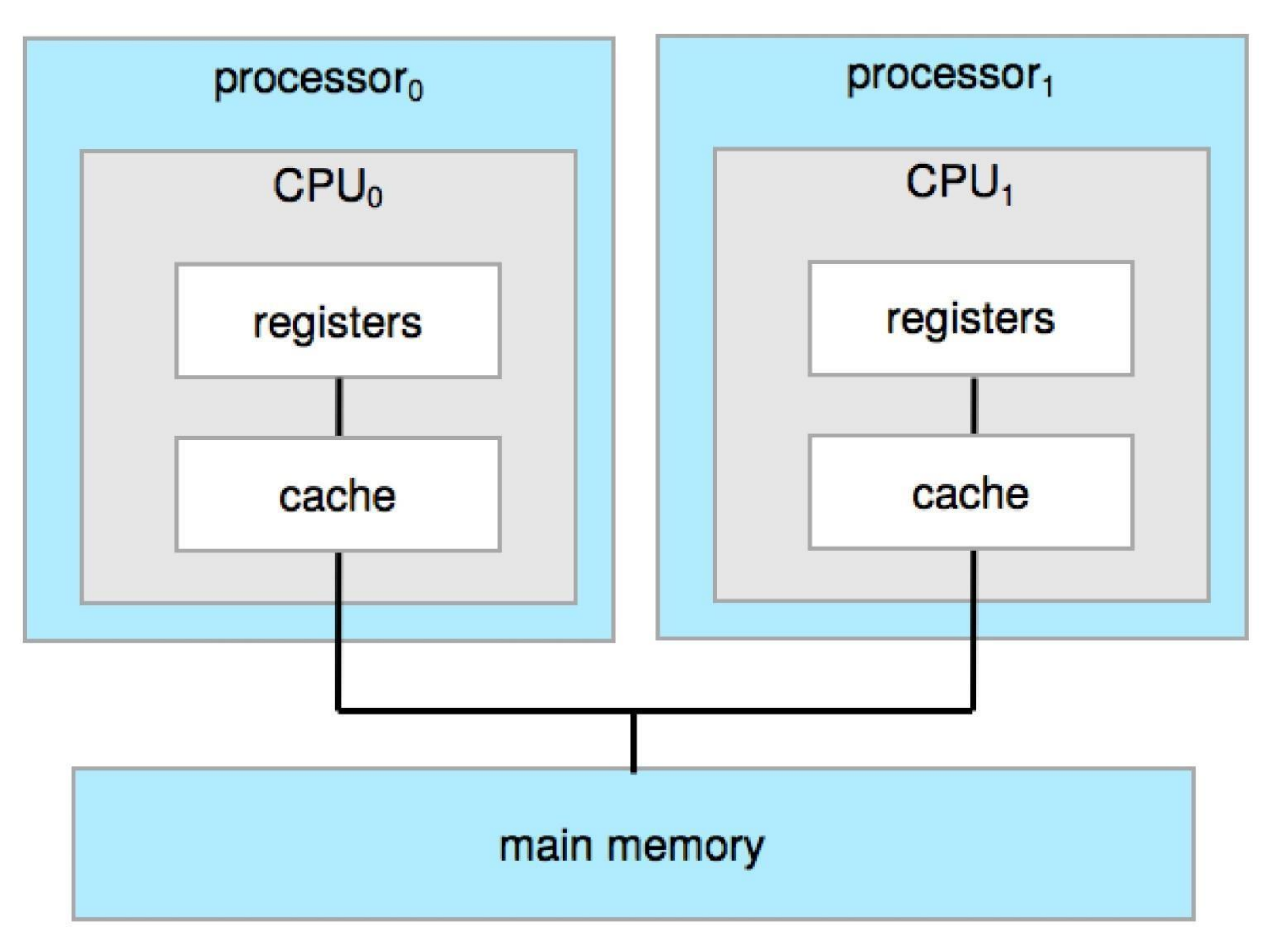
- After I/O starts, control returns to user program only upon I/O completion
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access)
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing
- After I/O starts, control returns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

# Computer System Architecture

# Computer-System Architecture

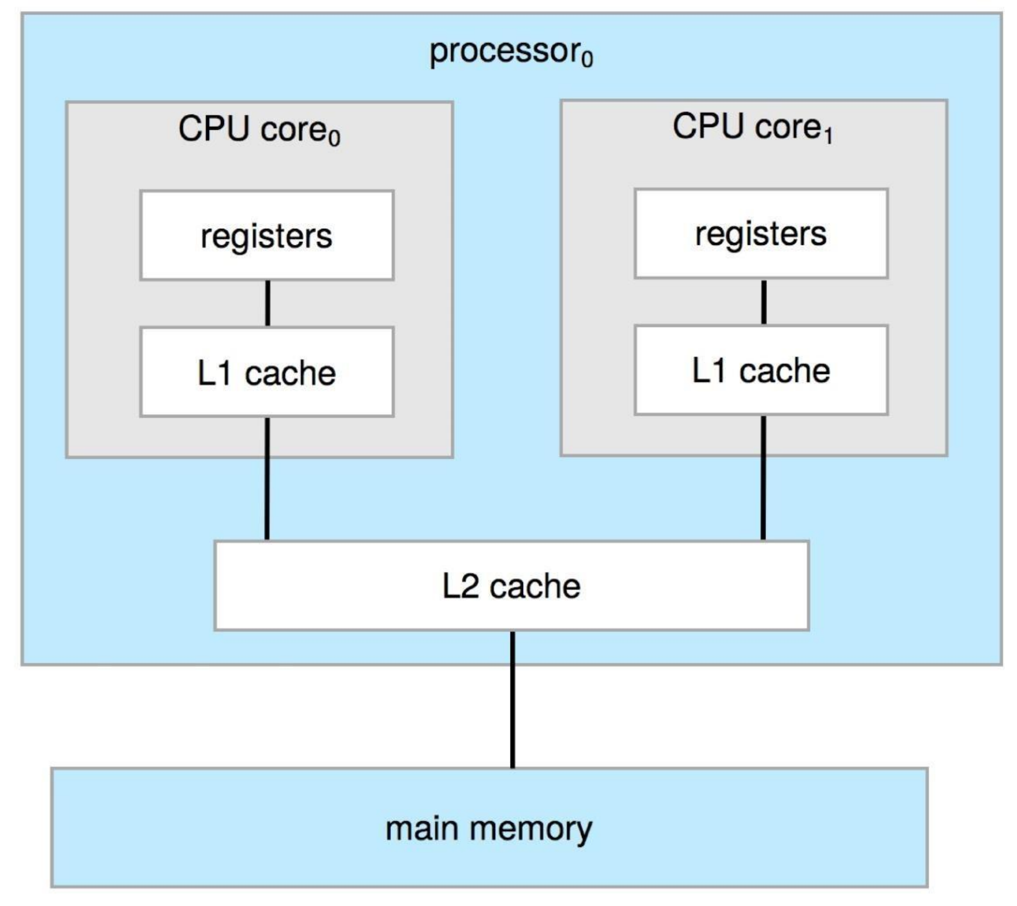
- Most systems use a single general-purpose processor
  - Most systems have special-purpose processors as well
- **Multiprocessors** systems growing in use and importance
  - Also known as **parallel systems**, **tightly-coupled systems**
  - Advantages include:
    1. **Increased throughput**
    2. **Economy of scale**
    3. **Increased reliability** – graceful degradation or fault tolerance
  - Two types:
    1. **Asymmetric Multiprocessing** – each processor is assigned a specie task.
    2. **Symmetric Multiprocessing** – each processor performs all tasks

# Symmetric Multiprocessing Architecture

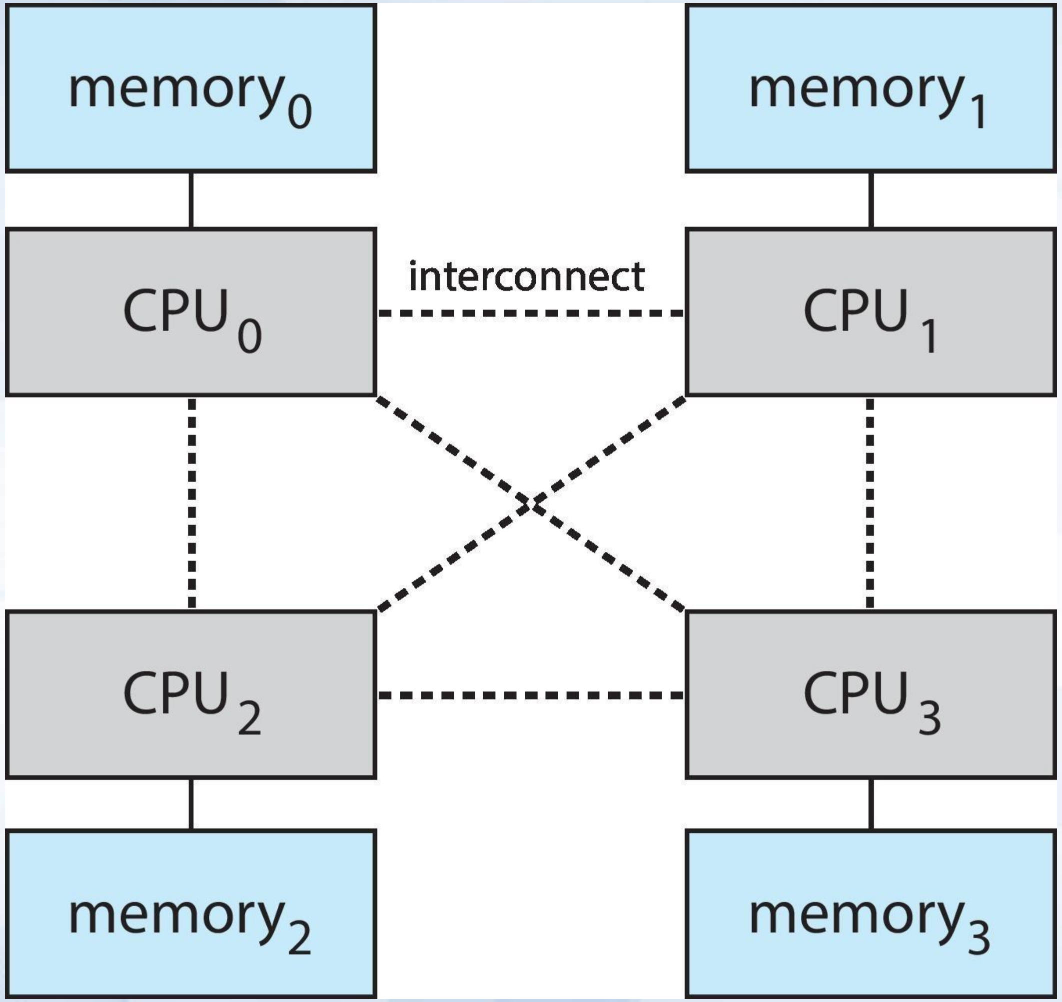


# Dual-Core Design

- Multi-chip and **multicore**
- Systems containing all  chips
  - Chassis containing multiple separate systems



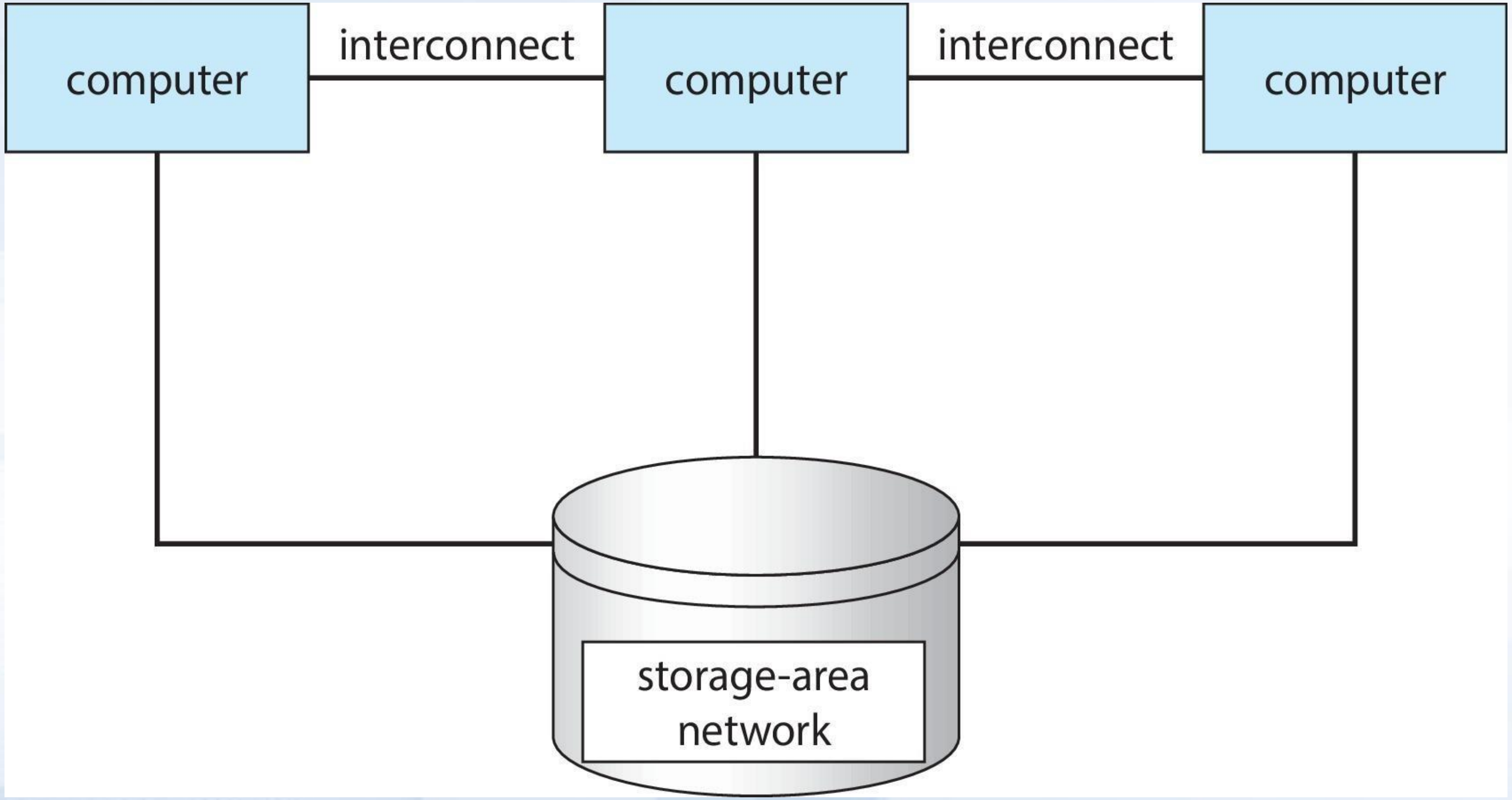
# Non-Uniform Memory Access System



# Clustered Systems

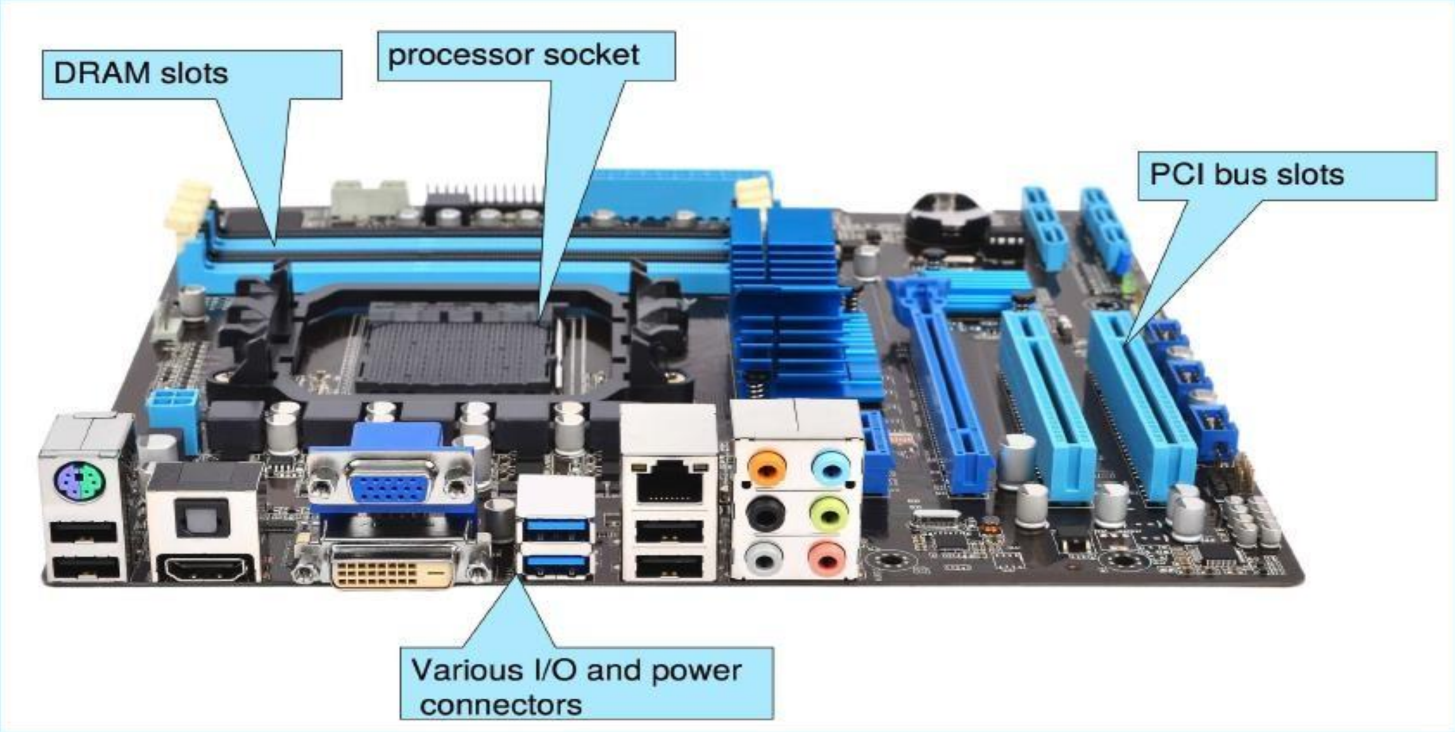
- Like multiprocessor systems, but multiple systems working together
  - Usually sharing storage via a **storage-area network** (**SAN**)
  - Provides a **high-availability** service which survives failures
    - **Asymmetric clustering** has one machine in hot-standby mode
    - **Symmetric clustering** has multiple nodes running applications, monitoring each other
  - 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

# Clustered Systems



# PC Motherboard

Consider the desktop PC motherboard with a processor socket shown below:




This board is a fully-functioning computer, once its slots are populated. It consists of a processor socket containing a CPU, DRAM sockets, PCIe bus slots, and I/O connectors of various types. Even the lowest-cost general-purpose CPU contains multiple cores. Some motherboards contain multiple processor sockets. More advanced computers allow more than one system board, creating NUMA systems.

# Computer System Environments

# Computing Environments

- Traditional
- Mobile
- Client Server
- Pear-to-Pear
- Cloud computing
- Real-time Embedded

# Traditional

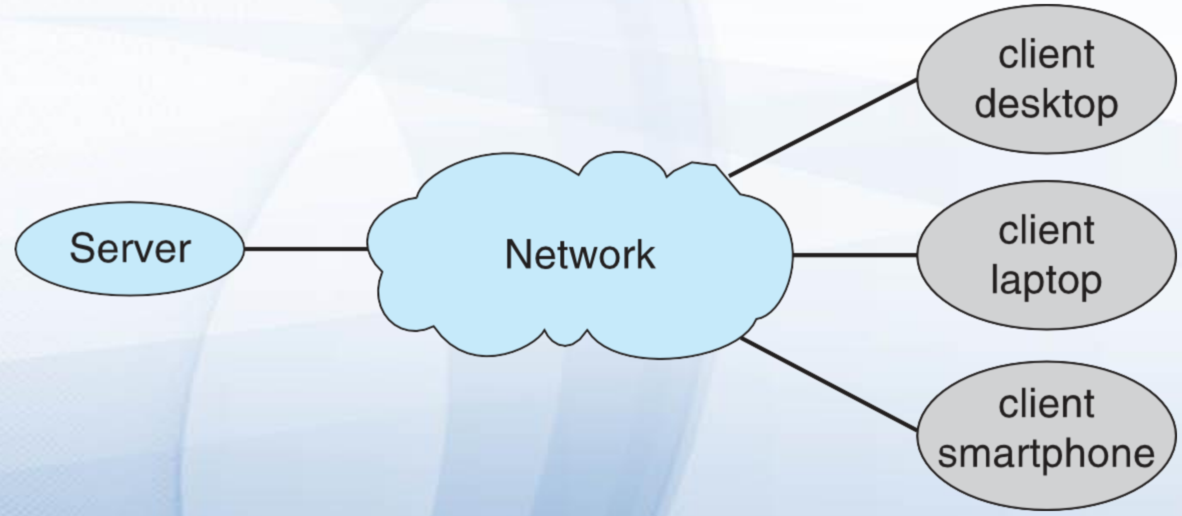
- Stand-alone general-purpose machines
- But blurred as most systems interconnect with others (i.e., the Internet)
- **Portals** provide web access to internal systems
- **Network computers** (**thin clients**) are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking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

# 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**

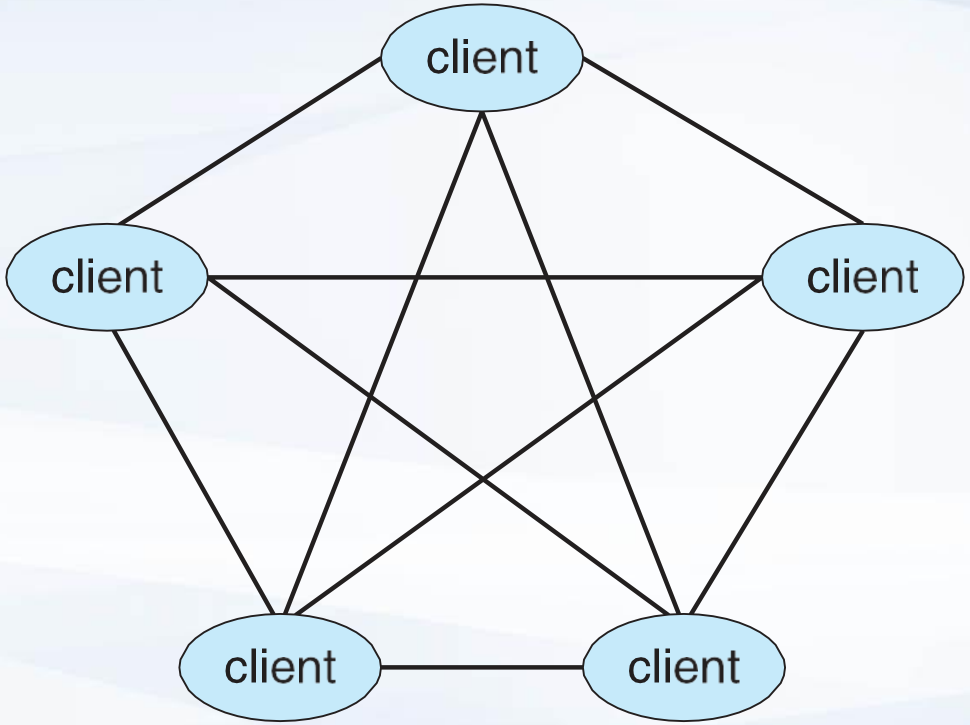
# Client Server

- 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



# 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
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via ***discovery protocol***
  - Examples include Napster and Gnutella, **Voice over IP** (**VoIP**) such as Skype



# Cloud Computing

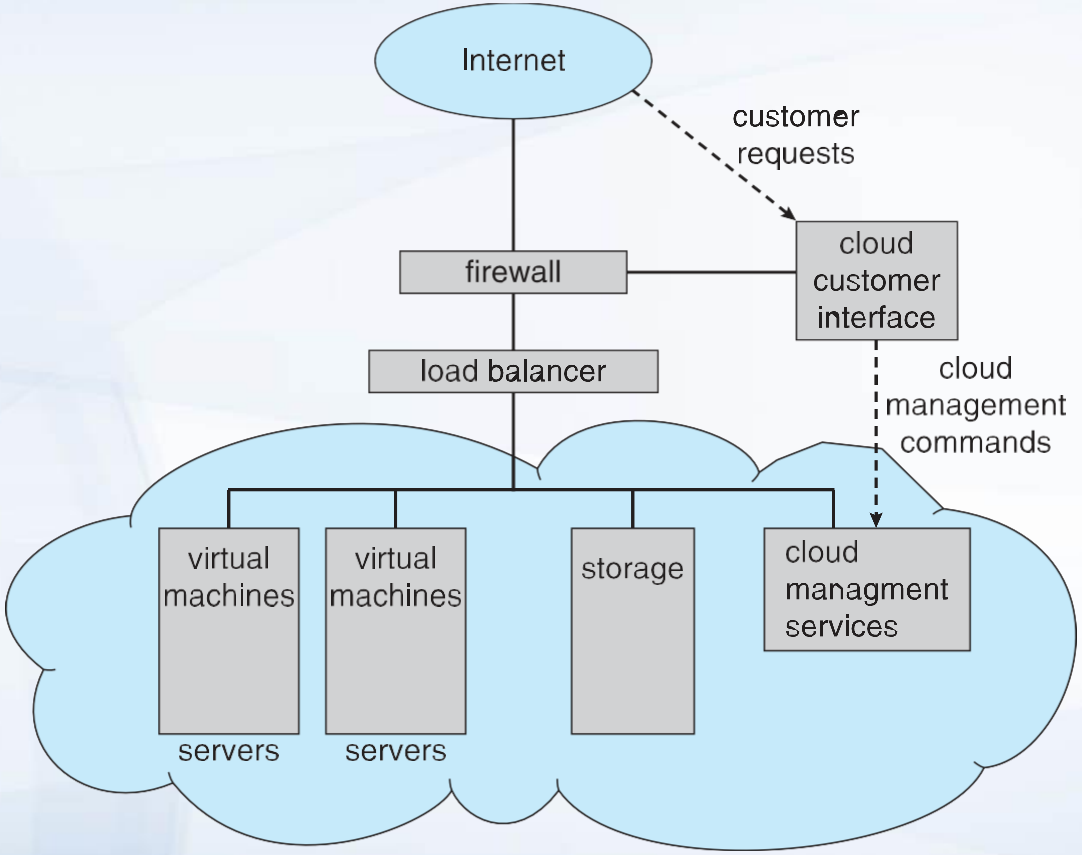
- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization because it uses virtualization as the base for it functionality.
  - Amazon **EC2** has thousands of servers, millions of virtual machines, petabytes of storage available across the Internet, pay based on usage

# Cloud Computing (Cont.)

- Many types
  - **Public cloud** – available via Internet to anyone willing to pay
  - **Private cloud** – run by a company for the company's own use
  - **Hybrid cloud** – includes both public and private cloud components
  - Software as a Service (**SaaS**) – one or more applications available via the Internet (i.e., word processor)
  - Platform as a Service (**PaaS**) – software stack ready for application use via the Internet (i.e., a database server)
  - Infrastructure as a Service (**IaaS**) – servers or storage available over Internet (i.e., storage available for backup use)

# Cloud Computing (cont.)

- Cloud computing environments composed of traditional OSes, plus VMMs, plus cloud management tools
  - Internet connectivity requires security like **firewalls**
  - **Load balancers** spread traffic across multiple applications



# 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

# Doubts?