DESIGN OF OPERATING SYSTEMS
CSE-4049

Lecture-1

Introduction

Nitesh Kumar Jha

Assistant Professor

ITER S'O'A (Deemed to be University)

Course Info

- **Course website**: https://niteshjha.ml/os.html
- **Contact**:

  email - niteshjha@soa.ac.in

  cabin – C-227

  webpage - niteshjha.ml
- **Lecture Materials**:
  - Lecture Slides
  - Additional readings
- **Reference books**:
- Abraham Silberschatz, Peter Baer Galvin and Greg gagne, Operating System Concepts 9th Edition, 2013: John Wiley & Sons, Inc®.
- Understanding the Linux kernel, Third Edition, 2006 by Daniel P. Bovet and Marco Cesati, O'Reilly Media Inc®.

Tentative Syllabus

- Introduction to OS
- Process Concept
- Threads
- Synchronization
- Process Scheduling
- Deadlock
- Memory Management Strategies
- Virtual Memory
- File Management System
- I/O Management System

Grading Pattern

Grading Pattern 1	**ATTENDANCE**	**5**
	ASSIGNMENTS	20
	MID TERM	15
	TOTAL INTERNAL	**40**
	THEORY EXAM	60
	TOTAL EXTERNAL	**60**
	TOTAL	**100**

What is Operating System?

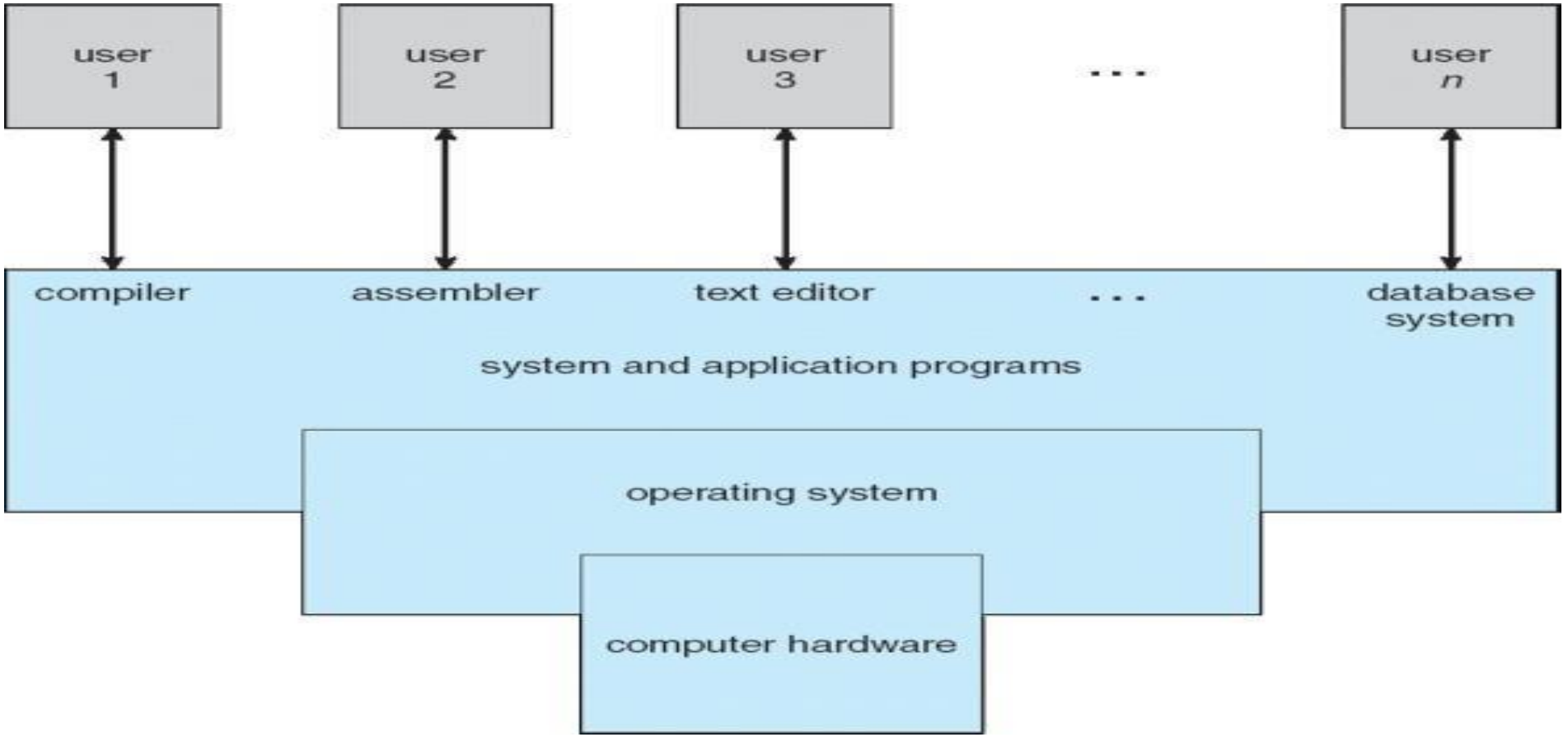
- A program that acts as an intermediary between a user and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Provides the basis/environment for application programs to run
  - Manages the computer hardware in an efficient manner
  - Make the computer system convenient to use

OPERATING SYSTEMS

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



Operating System 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 one program running at all times on the computer" is the kernel.
- No universally accepted definition

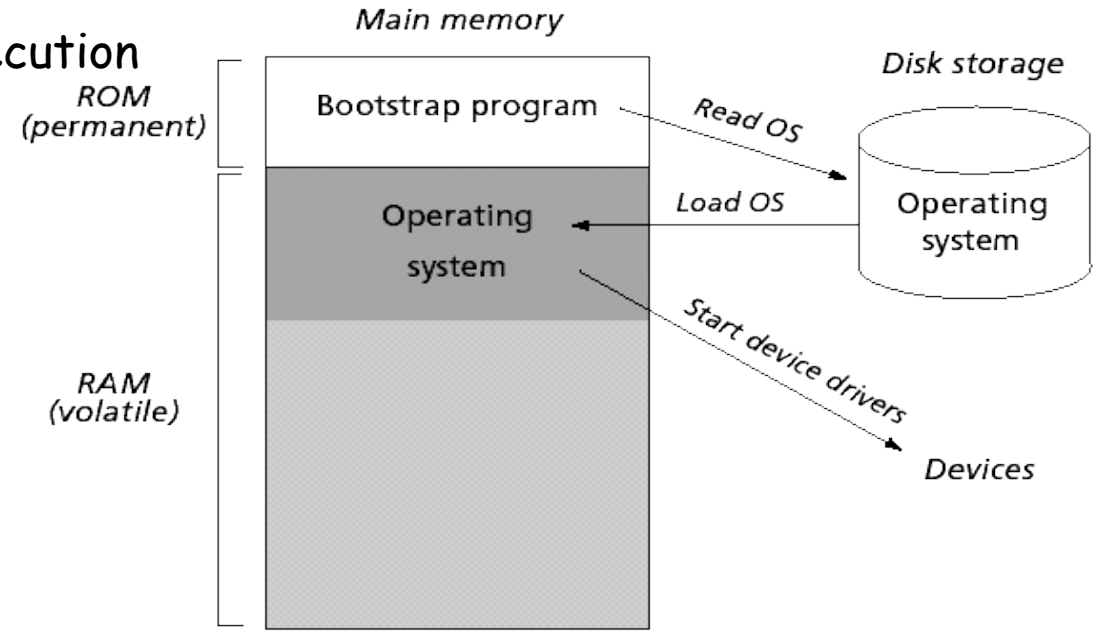
- Everything else is either a system program (ships with the operating system) , or an application program.

Computer Startup

- The process of loading the **operating system** into main memory is called *booting* (originally this was *bootstrapping* which refers to the process of pulling yourself up "by your bootstraps").
- **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

Is it that simple ?

What actually happens when we switch on a Computer ?



What happens when we switch on a computer?

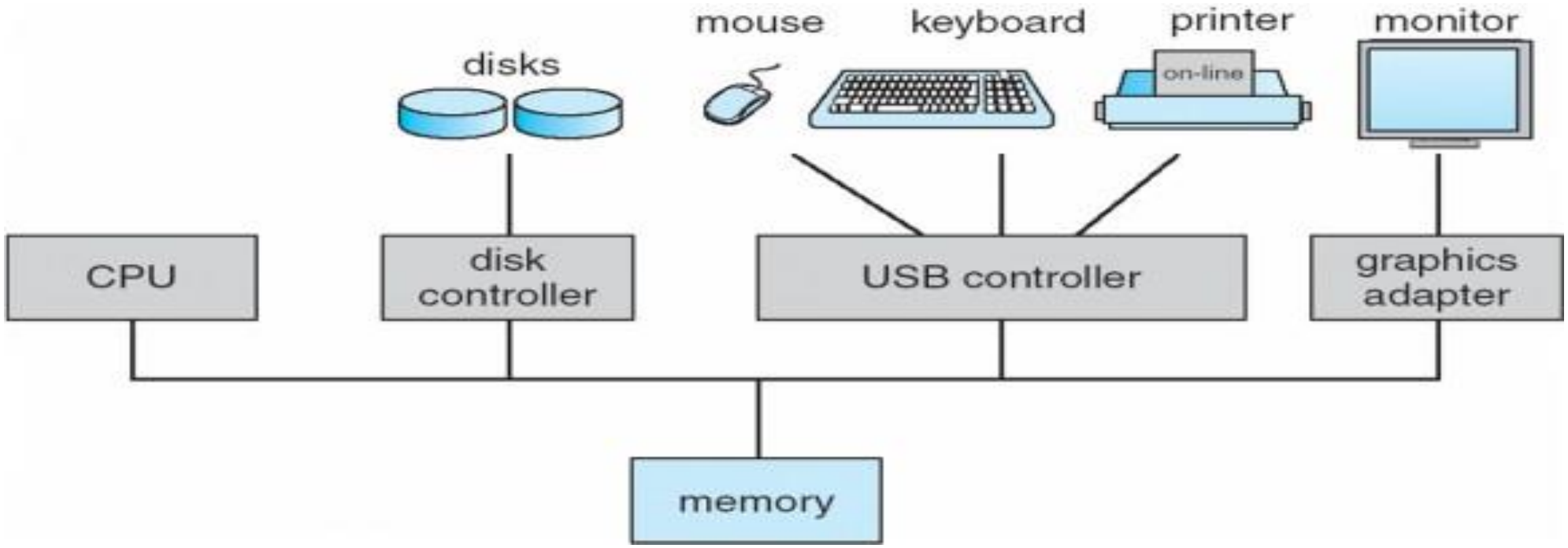
- All the booting instructions are built into chip called **BIOS(Basic Input Output System)** chip. The program in BIOS looks for a program called **Boot Loader** which is generally present in Boot Disk. In Linux Boot Loader is generally called **GRUB or LILO**. Boot Loader is bought to main memory and it is the duty of Boot Loader to load the OS into RAM.
- Loader does this by looking for **Kernel** which is a core of OS and has control over everything that occurs in system. Kernel when loaded looks for the hardware, and prepare them to run programs. It does this by poking the I/O ports. This process is called **Autoprobing**.
- Loading the Kernel into the memory isn't the end of Booting. It's just Stage-1. After this Kernel gives the control to a special process called **init**. It is the first *process* started during booting of the computer system and runs continuously until the system is shut down.

What happens when we switch on a computer?

- Init's job is to ensure the disks in our system are OK .The next job of it is to start **daemons**. Daemon is which runs as a background process, rather than being under the direct control of an interactive user.
- The next step is to prepare the system for users. Init starts a program called **Getty** which looks after your screen and keyboard. Now a days they're using several copies of Getty so that we can have multiple consoles.
- The last step is to start daemons that controls networking and other services. The main one among them is **X server** which is responsible for managing graphical interface, keyboard and mouse.
- At last we will see our graphical login screen which was produced by a program called **Display Manager**.

Computer System Organization

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



Interrupts

- The occurrence of an event is usually signaled by an interrupt from either the hardware or the software.
- Hardware may trigger an interrupt at any time by sending a signal to the CPU.
- Software may trigger an interrupt by executing a special operation called a system call (also called a monitor call).
- When the CPU is interrupted, it stops current execution and immediately transfers execution to a fixed location.
- The fixed location usually contains the starting address where the service routine for the interrupt is located.
- On completion, the CPU resumes the interrupted computation.

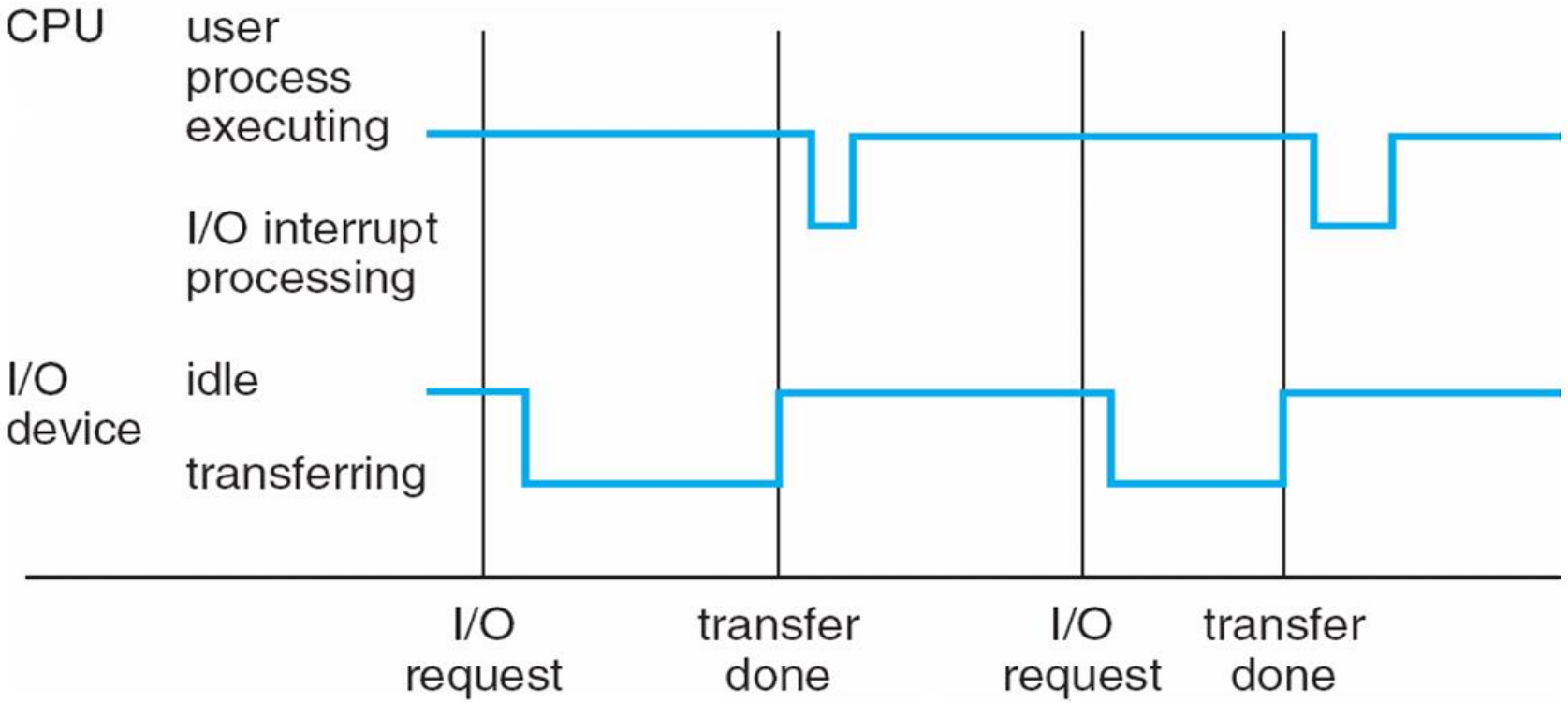
Importance of Interrupts

- It helps us to perform a task which is of higher importance when the processor is busy with another task.
- Interrupt alerts the processor to a high priority condition the interruption of current code the processor is executing.
- Processor responds by suspending its current activities, saving its state and executing a function called interrupt handler.

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 Timeline



Storage Unit

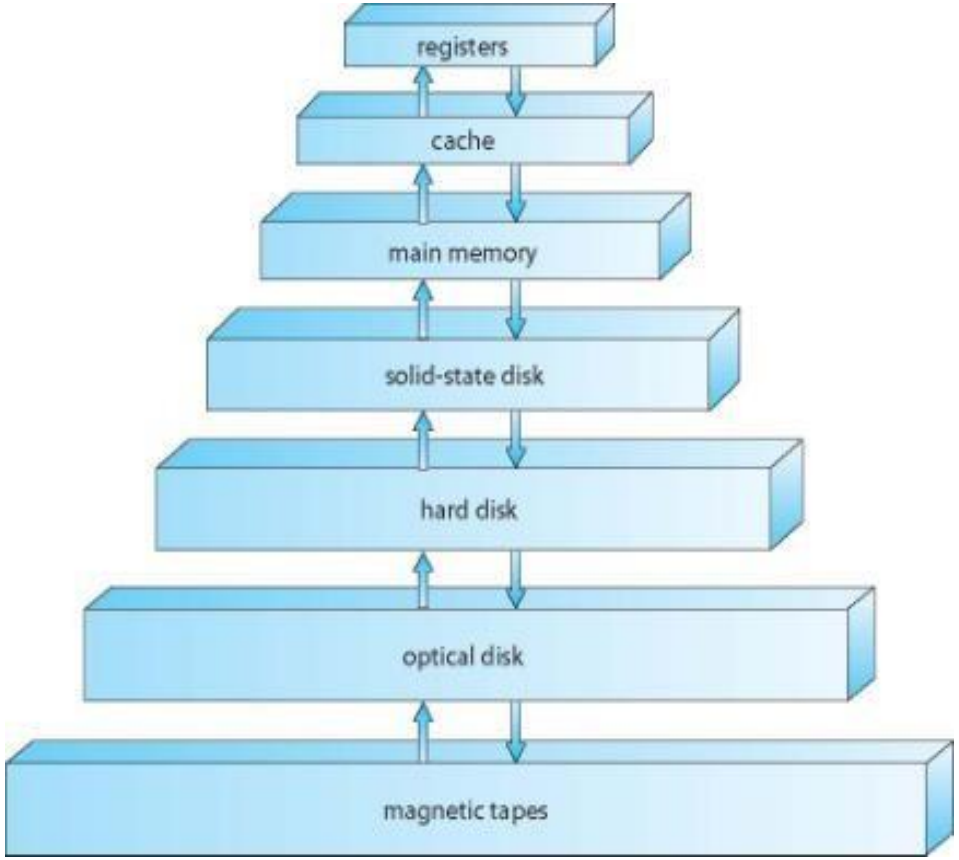
- Basic unit of computer storage is the **bit**
  - A bit can contain one of two values, 0 and 1
- A **byte** is 8 bits, and on most computers it is the smallest convenient chunk of storage.
- A **word**, which is a given computer architecture's native unit of data.
  - Example: A computer that has 64-bit registers and 64-bit memory addressing typically has
  - 64-bit (8-byte) words.
- Computer storage measured in collection of bytes.
  - A **kilobyte**, or **KB**, is 1,024 bytes (2^{10})
  - a **megabyte**, or **MB**, is $1{,}024^2$ bytes
  - a **gigabyte**, or **GB**, is $1{,}024^3$ bytes
  - a **terabyte**, or **TB**, is $1{,}024^4$ bytes
  - a **petabyte**, or **PB**, is $1{,}024^5$ bytes

Storage Structure

- Main memory – only storage media that the CPU can access directly. Too small to store all programs/data needed.
  - Random access
  - Typically volatile
- Secondary storage – extension of main memory that provides large nonvolatile storage capacity
- Hard 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 – faster than hard disks, nonvolatile
  - Various technologies
  - Becoming more popular

Storage-Device Hierarchy

- Storage systems organized in hierarchy according to their Speed, Cost, and Volatility



Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- 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 there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

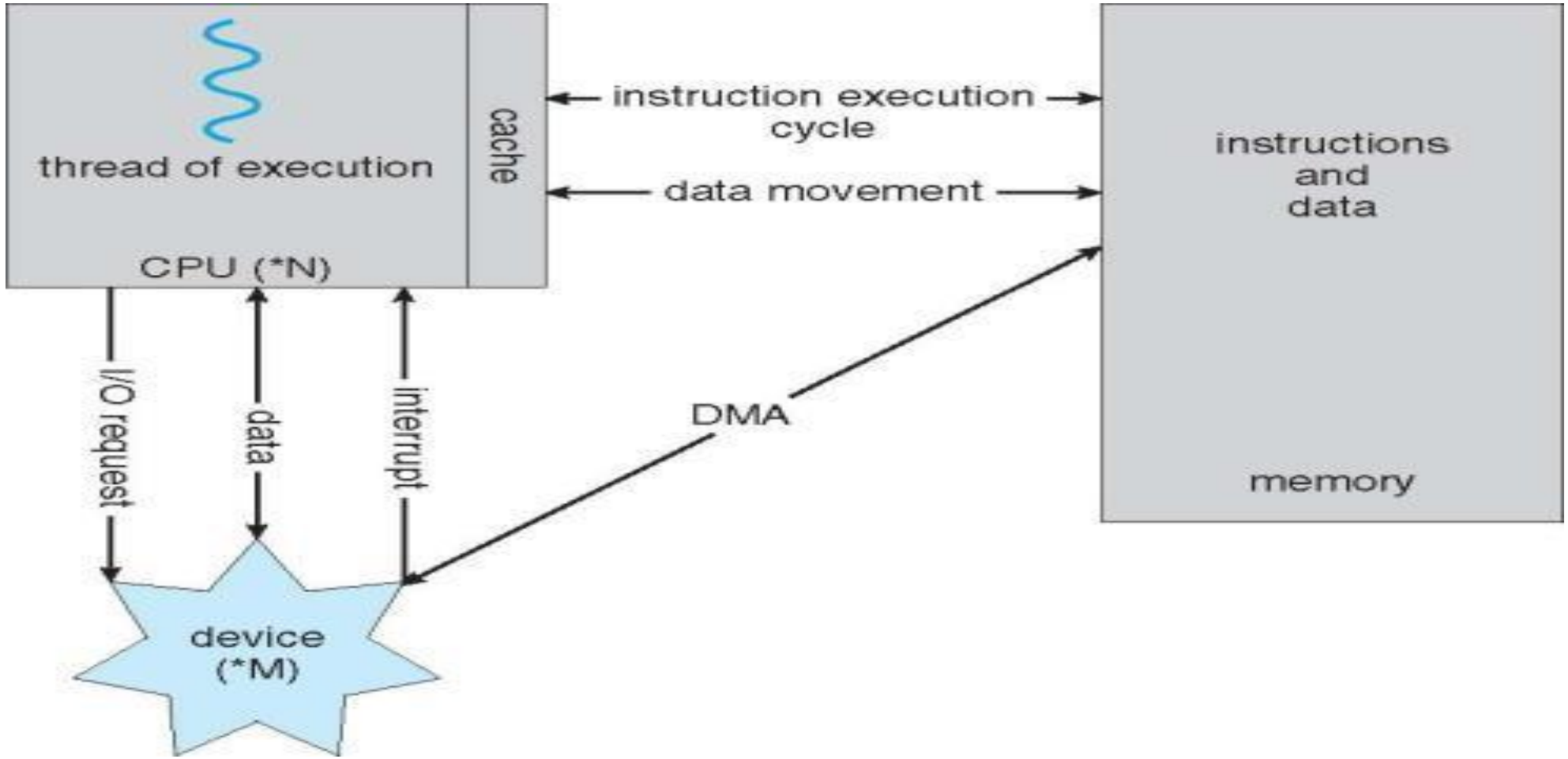
I/O Structure

- Each device controller is in charge of a specific device and is responsible for moving the data between the peripheral device and the local buffer.
- OS have a device driver for each device controller
- To start an I/O, device driver loads the appropriate register within the device controller.
- The controller starts transfer from device to the local buffer.
- Once complete, it informs the device driver via an interrupt and the driver then returns the control to the OS.
- It is called interrupt driven data transfer.

I/O Structure - DMA

- The interrupt driven data transfer produce high overhead when used for bulk data transfer as the processor is blocked.
- To avoid 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 entire 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

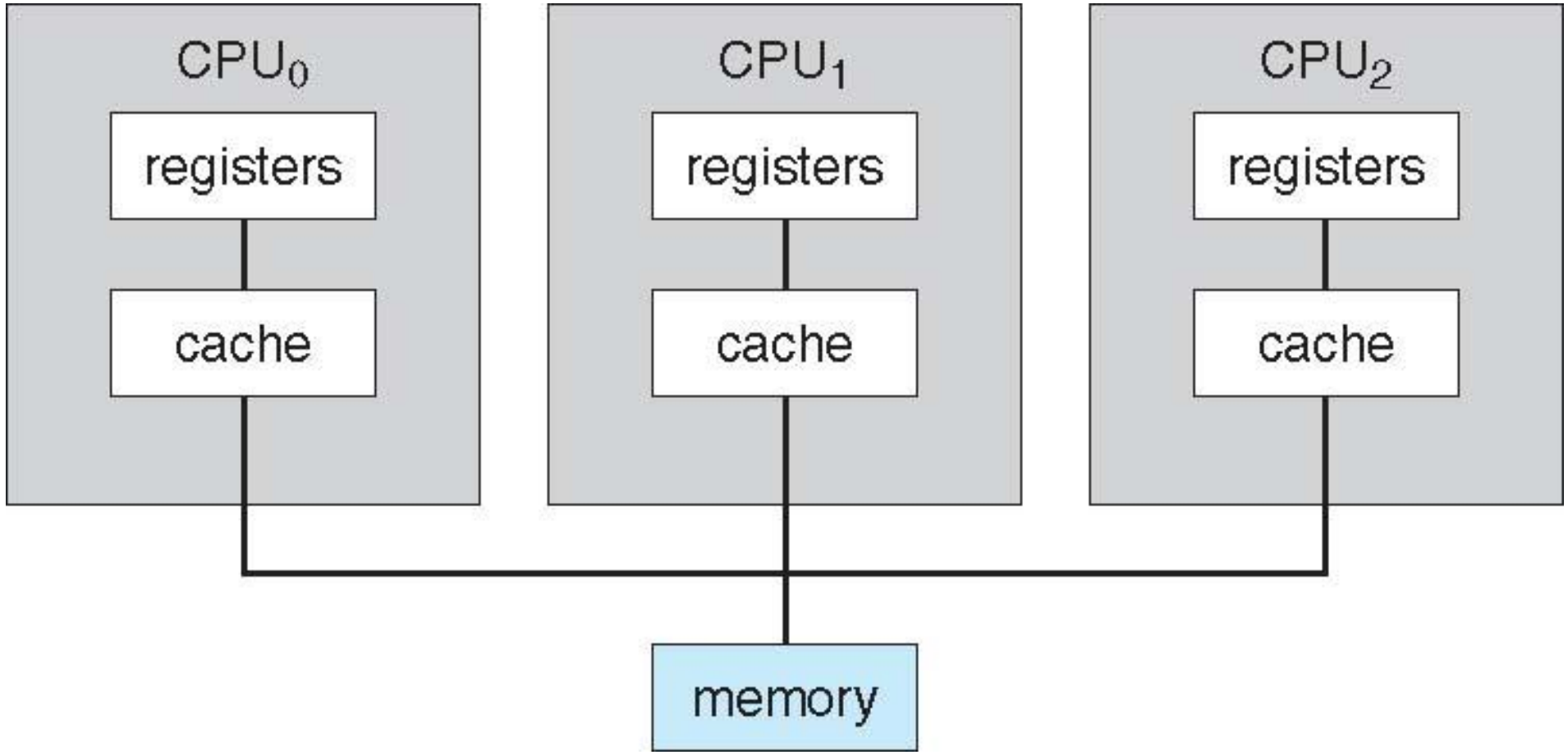
How a Modern Computer Works



Computer-System Architecture

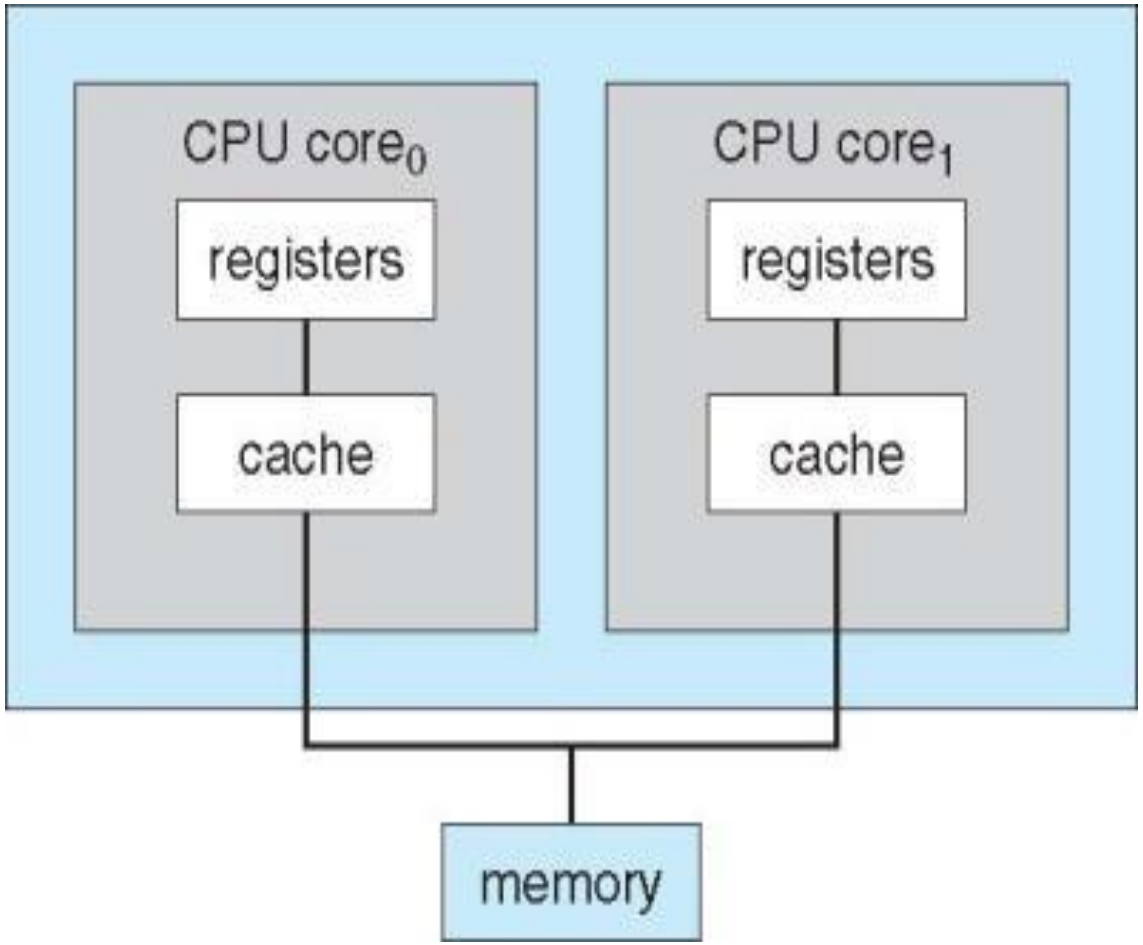
- Most systems use a single general-purpose processor
  - Most systems have special-purpose processors as well
- **Multiprocessors** systems growing in use and importance
  - Also known as **parallel systems**, **tightly-coupled systems**
  - Advantages include:
    1. **Increased throughput**
    2.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

Symmetric Multiprocessing Architecture



A Dual-Core Design

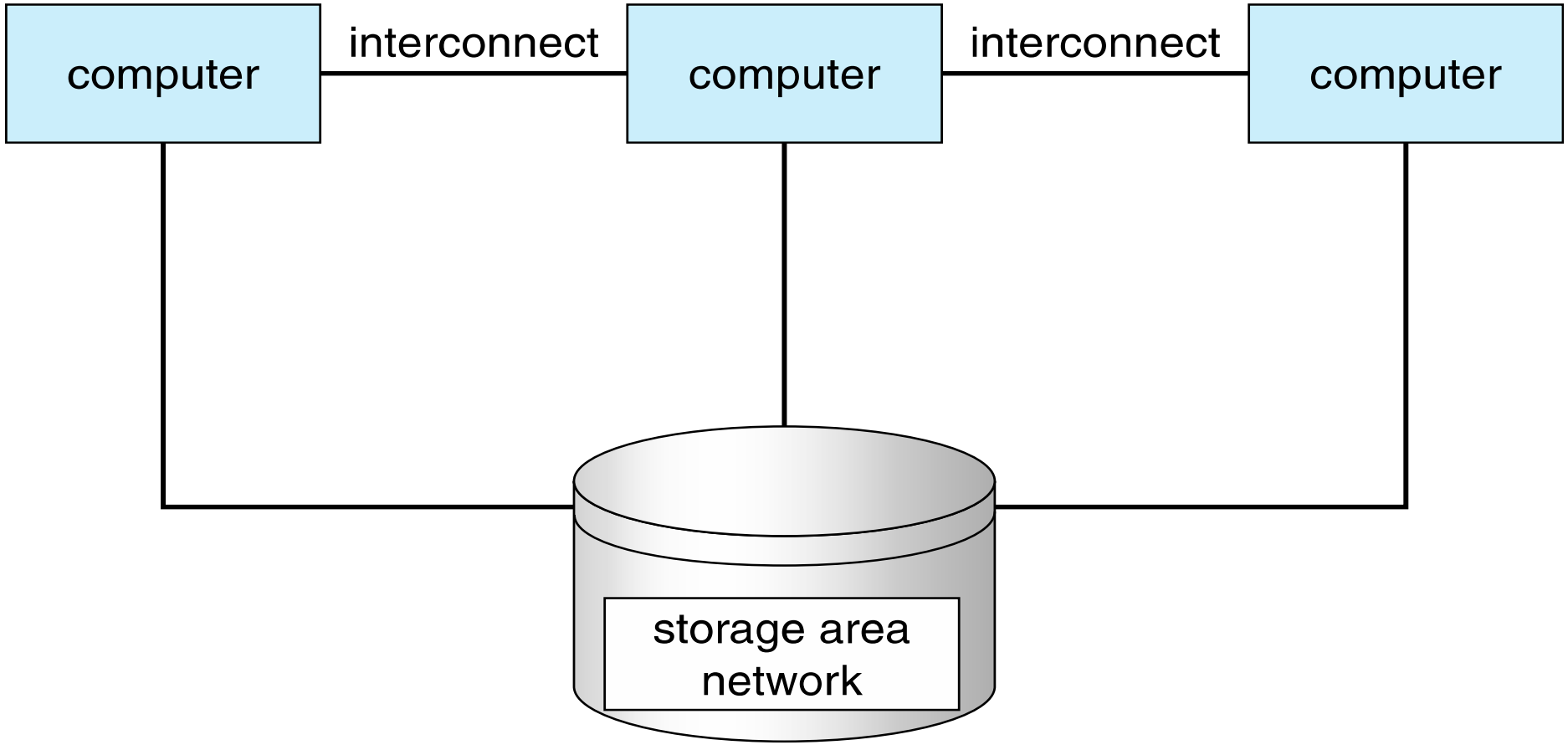
- Multiple computing cores on a single chip.
- These are more efficient as on chip communication is faster.
- It consumes less power.
- Suitable for database and web servers



Clustered Systems

- 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 used for **high-performance computing (HPC)**
    - Applications must be written to use **parallelization**
  - Some have **distributed lock manager** (DLM) to avoid conflicting operations

Clustered Systems

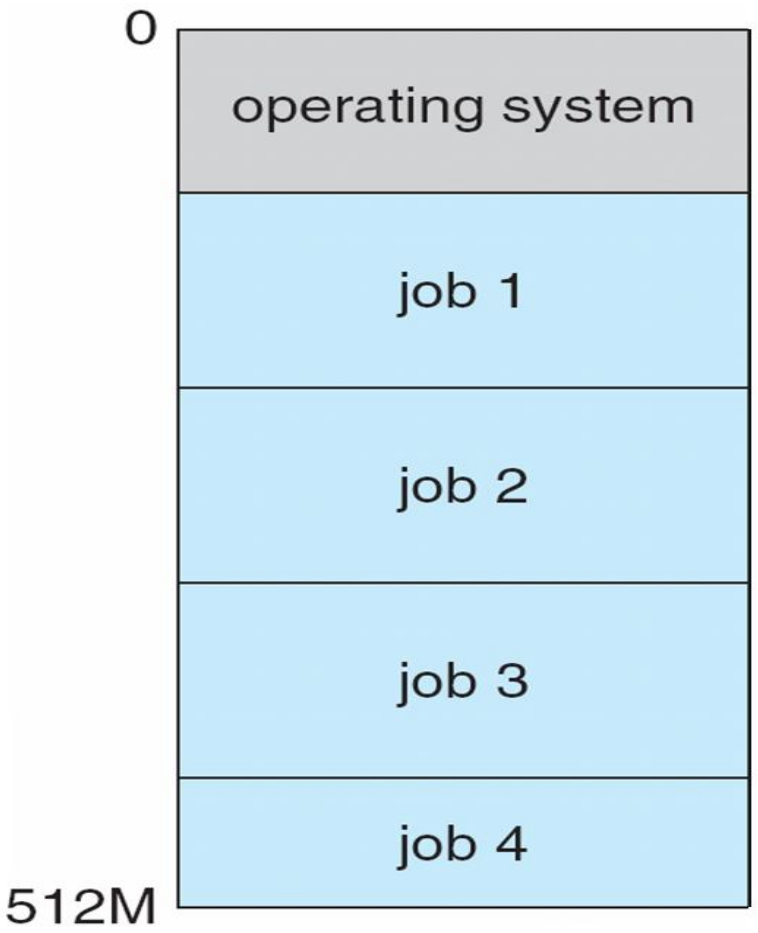


Operating System Structure

- **Multiprogramming** (**Batch system**) 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**
  - When it has to wait (for I/O for example), OS switches to another job

- **Timesharing** (**multitasking**) 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

Memory Layout for Multiprogrammed System



Operating-System Operations

- **Interrupt driven** (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (**exception** or **trap):**
    - Software error (e.g., division by zero)
    - Request for operating system service
    - Other process problems include infinite loop, processes modifying each other or the operating system

Operating-System Operations (cont.)

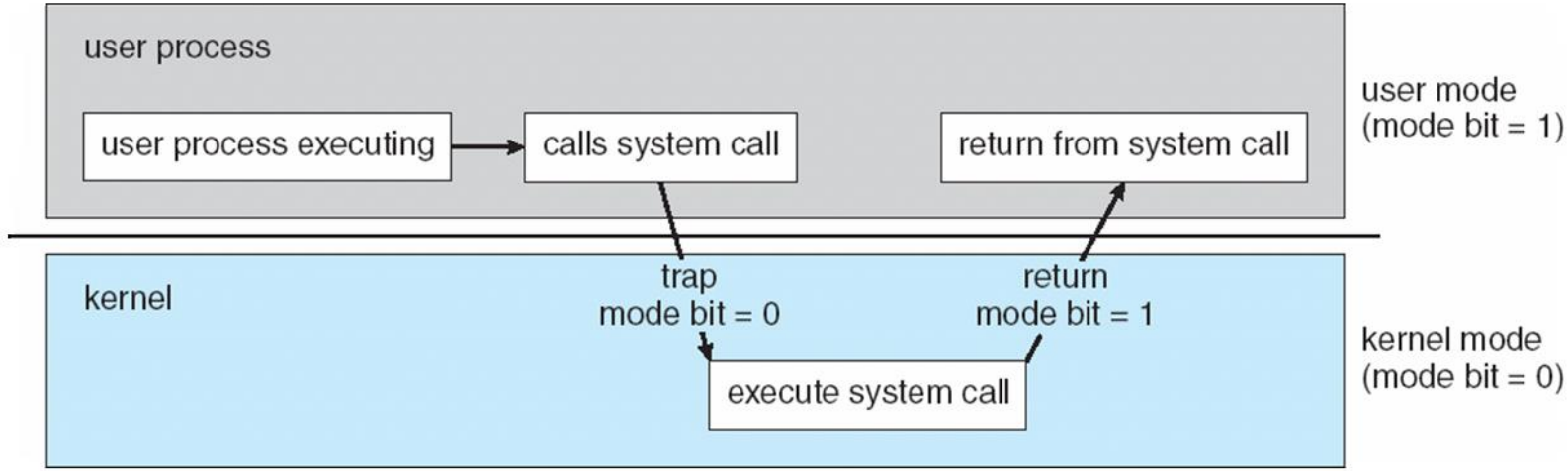
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user
- Increasingly CPUs support multi-mode operations
  - i.e. **virtual machine manager** (**VMM**) mode for guest **VMs**

Dual Mode Operation

- At system boot the hardware starts in kernel mode. The OS is then loaded and starts user applications in user mode.
- Whenever a trap or interrupt occurs, the hardware switches from user mode to kernel mode and vice versa by changing the mode bit.
- Initial control is within the operating system, where instructions are executed in kernel mode.
- When control is given to a user application, the mode
  - is set to user mode.
- Eventually, control is switched back to the operating system via an interrupt, a trap, or a system call
- The dual mode of operation provides us with the means for protecting the operating system

Transition from User to Kernel Mode

- Timer to prevent infinite loop / process hogging resources
  - Timer is set to interrupt the computer after some time period
  - Keep a counter that is decremented by the physical clock.
  - Operating system set the counter (privileged instruction)
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time



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

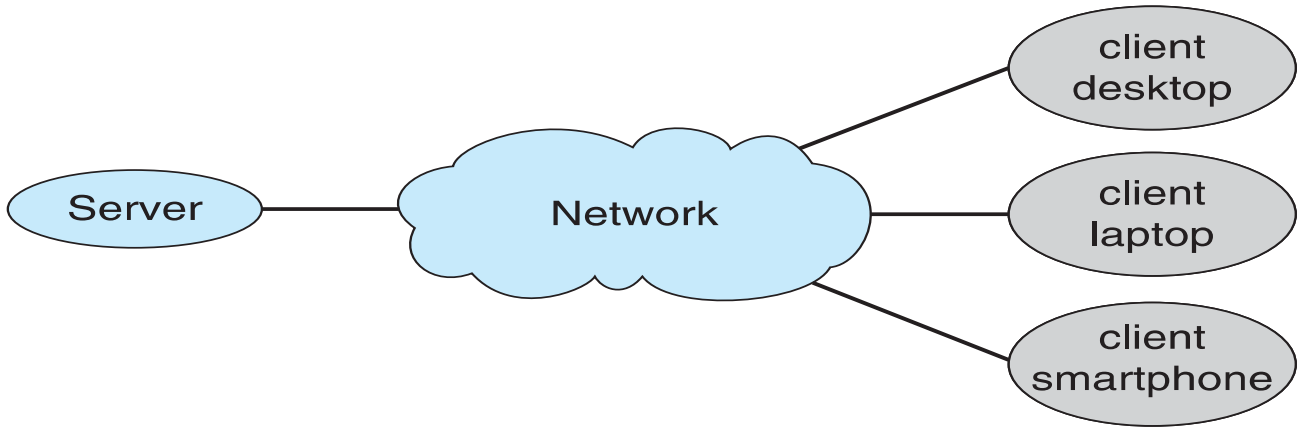
- Handheld smartphones, tablets, etc
- What is the functional difference between them and a "traditional" laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of apps like ***augmented reality***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders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

Computing Environments – Distributed

- Distributed computiing
  - 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

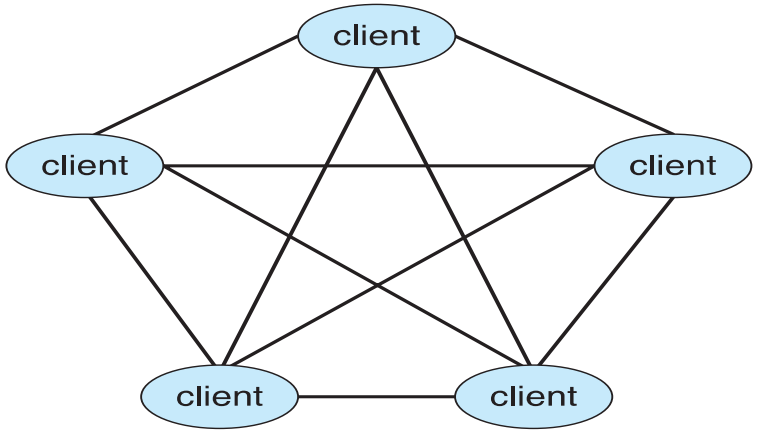
Computing Environments - Client-Server

- 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
    - 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



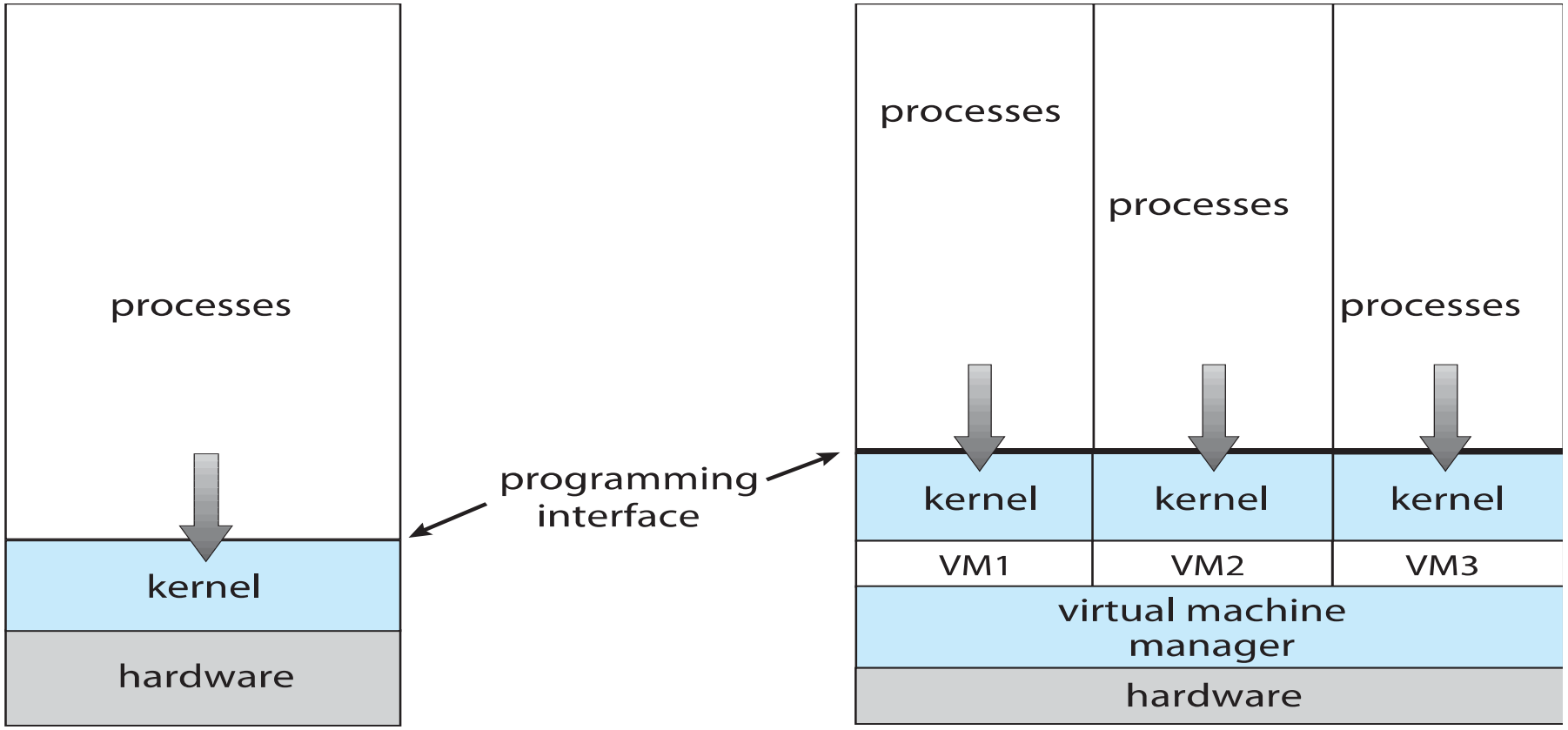
Computing Environments - Virtualization

- Allows operating systems to run applications within other OS'es
  - Vast and growing industry
- **Emulation** used when source CPU type different from target type (i.e. PowerPC to Intel x86)
  - Generally slowest method
  - When computer language not compiled to native code – **Interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OS'es also natively compiled
  - Consider VMware running WinXP guests, each running applications, all on native WinXP **host** OS
  - **VMM** (virtual machine Manager) provides virtualization services

Computing Environments - Virtualization

- Use cases involve laptops and desktops running multiple OS'es for exploration or compatibility
  - Apple laptop running Mac OS X host, Windows as a guest
  - Developing apps for multiple OSes without having multiple systems
  - QA testing applications without having multiple systems
  - Executing and managing compute environments within data centers
- VMM can run natively, in which case they are also the host
  - There is no general purpose host then (VMware ESX and Citrix XenServer)

Computing Environments - Virtualization

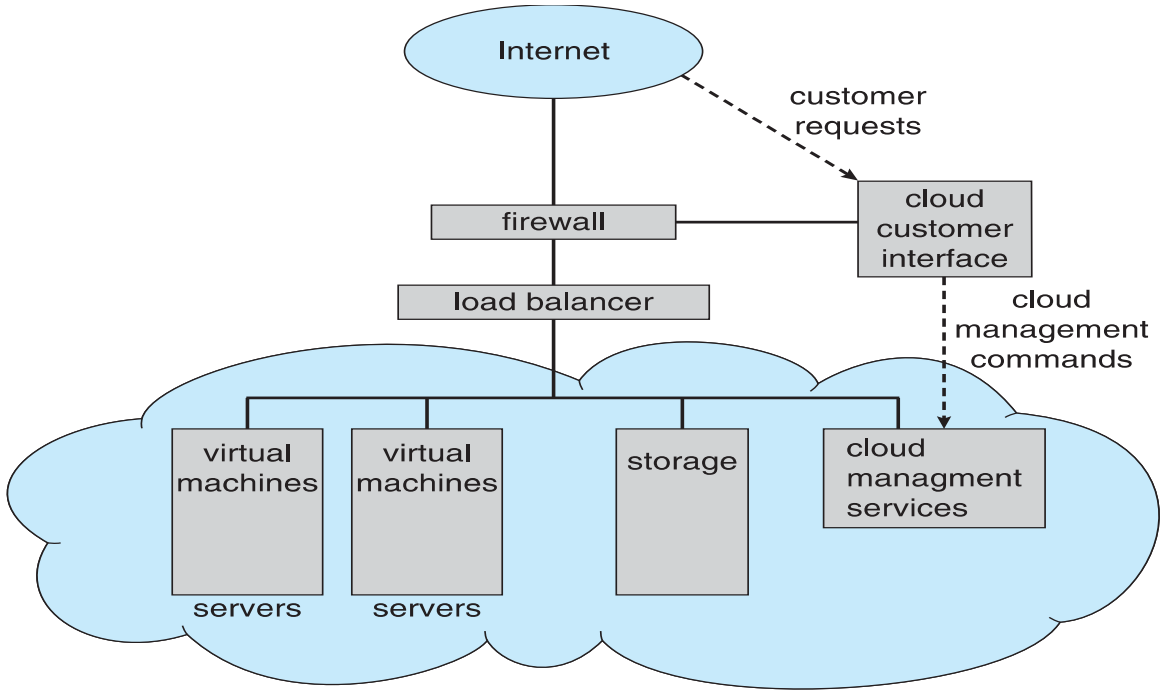


Computing Environments – Cloud Computing

- 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
- 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)

Computing Environments – Cloud Computing

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



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

End of Lecture
Thank You