Introduction to Operating Systems

CMSC 125: Operating Systems

Victor M. Romero II

Division of Natural Sciences and Mathematics

Objectives

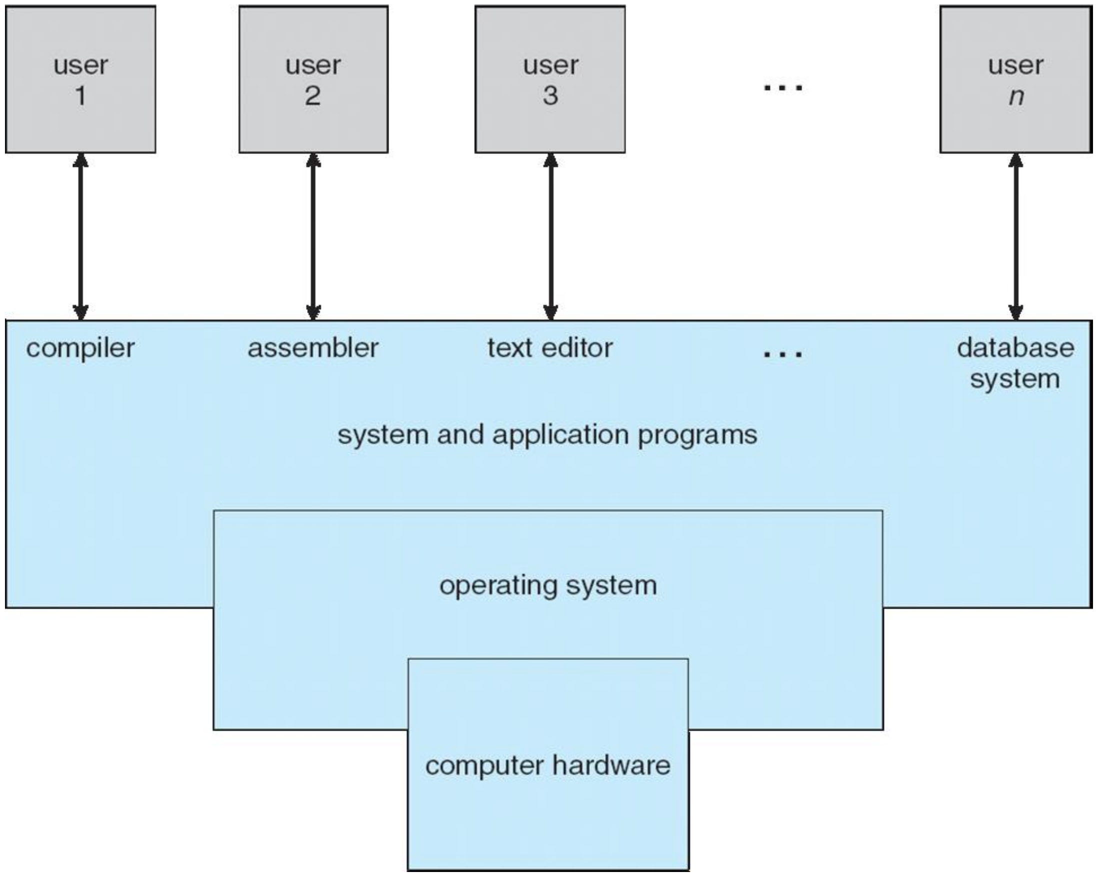
- Describe the basic organization of computer systems
- Provide a grand tour of the major components of operating systems
- Give an overview of the many types of computing environments
- Explore several open source operating systems

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 computer system convenient to use
  - Use the computer hardware efficiently

Computer System Structure

- Computer system can be divided into four components:
    1. Hardware
    2. Operating system
    3. Application programs
    4. Users



What Operating Systems Do

- Depends on context.
- Users want **convenience**, **ease of use**, and **good performance**
  - Ignorant to resource utilization
- **Shared-computer** environments such as mainframe must keep all users happy
- Users of **dedicated 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
- Some computers have little or no user interface, e.g., **embedded systems**.

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

Operating System Definition

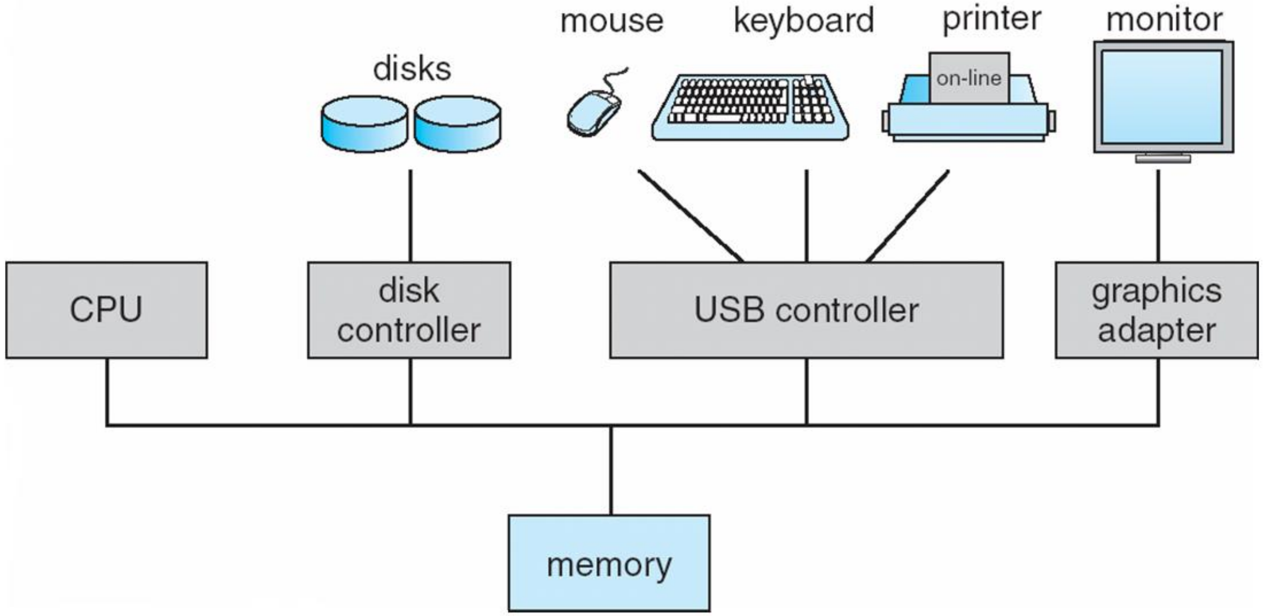
- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is a good approximation
- The one program running at all times is the **kernel**
- Everything else is either
  - A system program
  - An application program

Computer Startup

- A **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware/BIOS**
  - Initializes all aspects of the system
  - Loads operating system kernel and starts execution

Computer System Organization

- 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.



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**

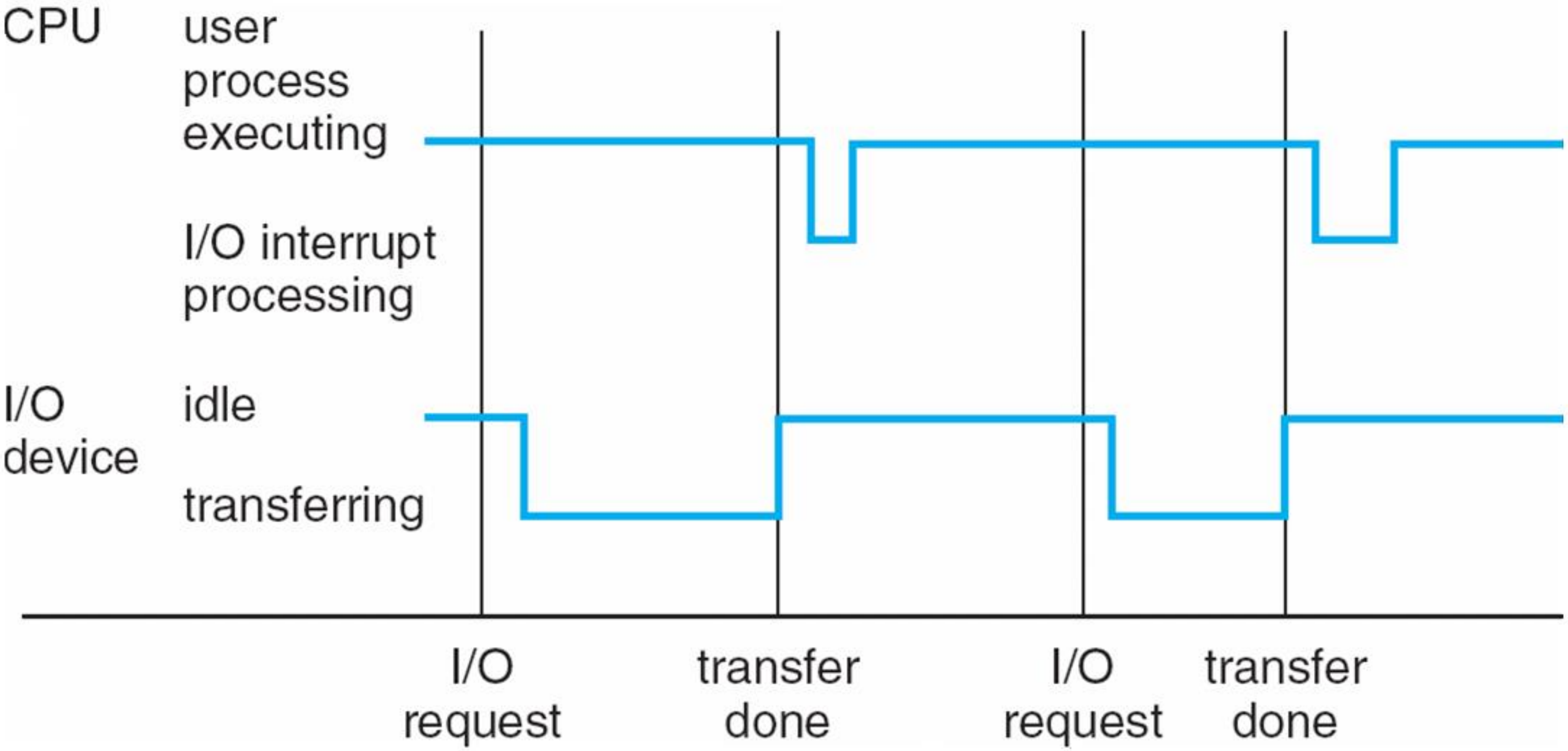
Common Interrupt Functions

- 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 by an error or a user request
- An operating system is **interrupt driven**

Interrupt Handling

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

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 type, address, and state
  - OS indexes into I/O device table

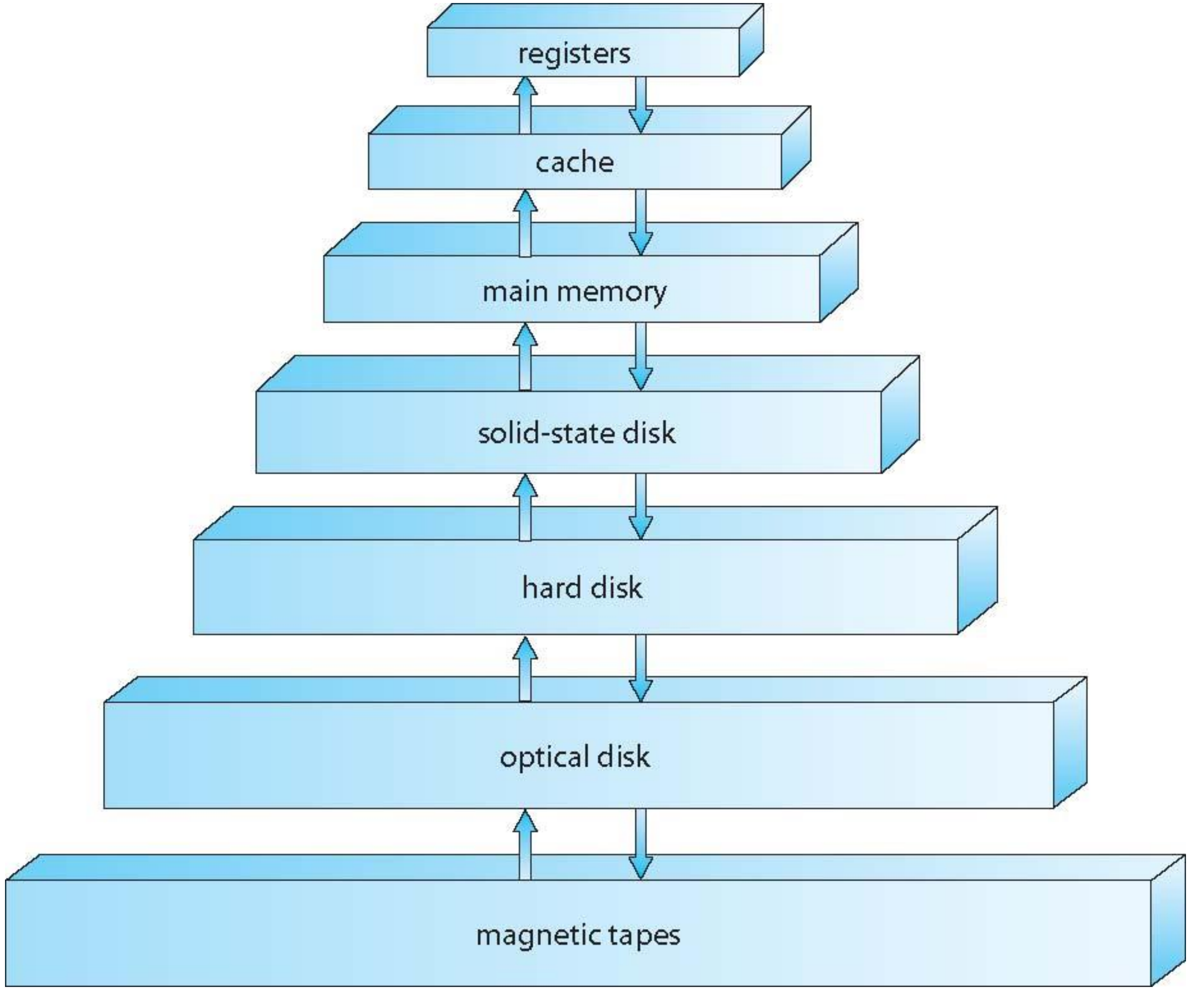
Storage Structure

- **Main memory** – only large storage media directly accessible by the CPU
  - **Random Access**, **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 logically divided into **tracks** that are subdivided to **sectors**
- **Solid-state device** – faster than hard disk, nonvolatile
  - Various technologies, Growing popularity

Storage Hierarchy

- Storage systems organized in hierarchy
  - Speed, cost, volatility
- **Caching** – copying information into faster storage system.
  - e.g., main memory as cache for secondary storage
- **Device Driver** for each device controller to manage I/O
  - Provides uniform interface between controller and kernel

Storage-Device Hierarchy



Caching

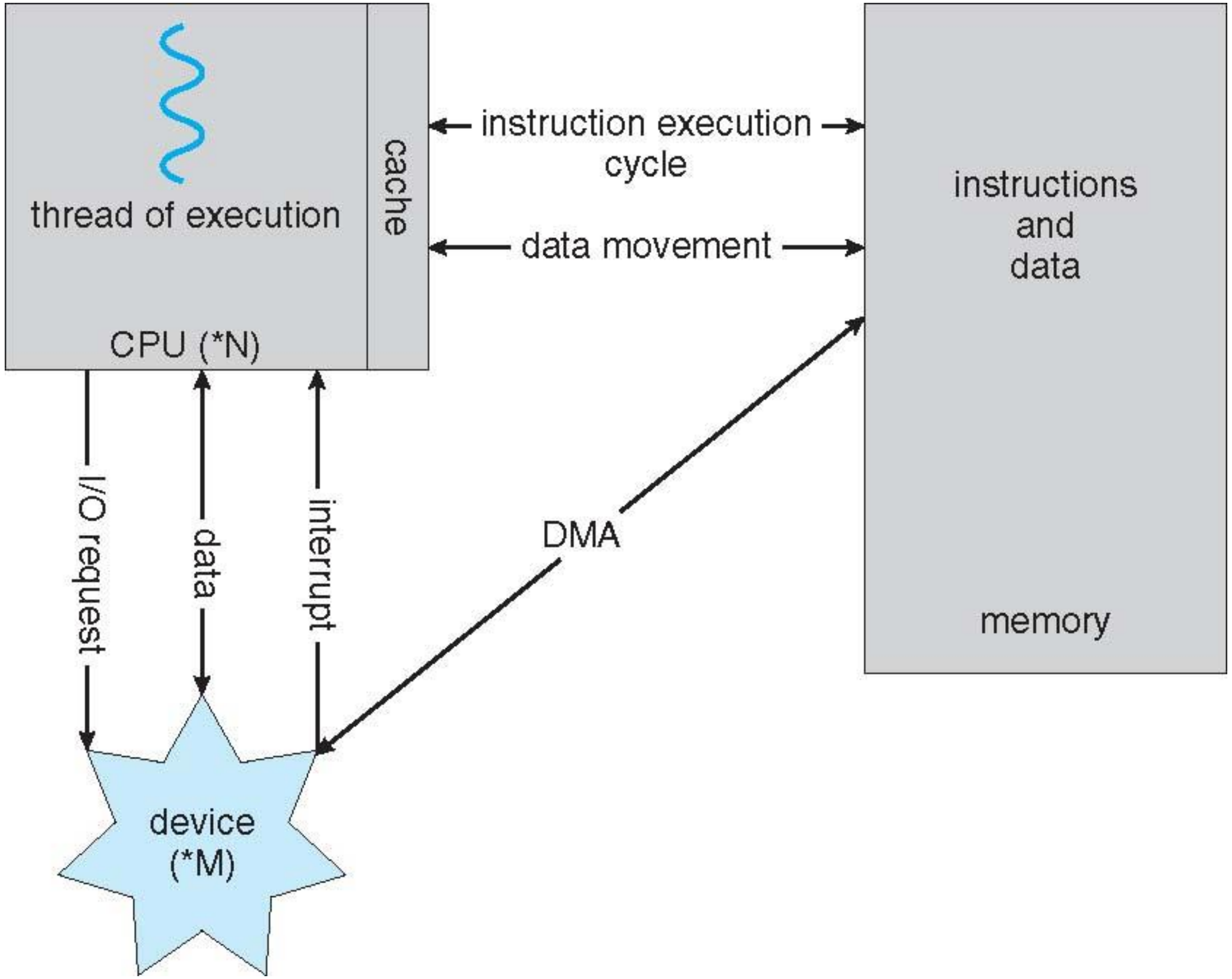
- Important principle, performed at many levels in a computer
- Information in use is copied from slower to faster storage **temporarily**
- Faster storage (cache) is checked first to determine if information is immediately available
  - If not available, data is copied to cache and used
- Cache is smaller than storage being cached
  - Cache management is an important design problem
  - Cache size and replacement policy

Direct Memory Access

- 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 one interrupt per byte.

How a Modern Computer Works

A von Neumman architecture

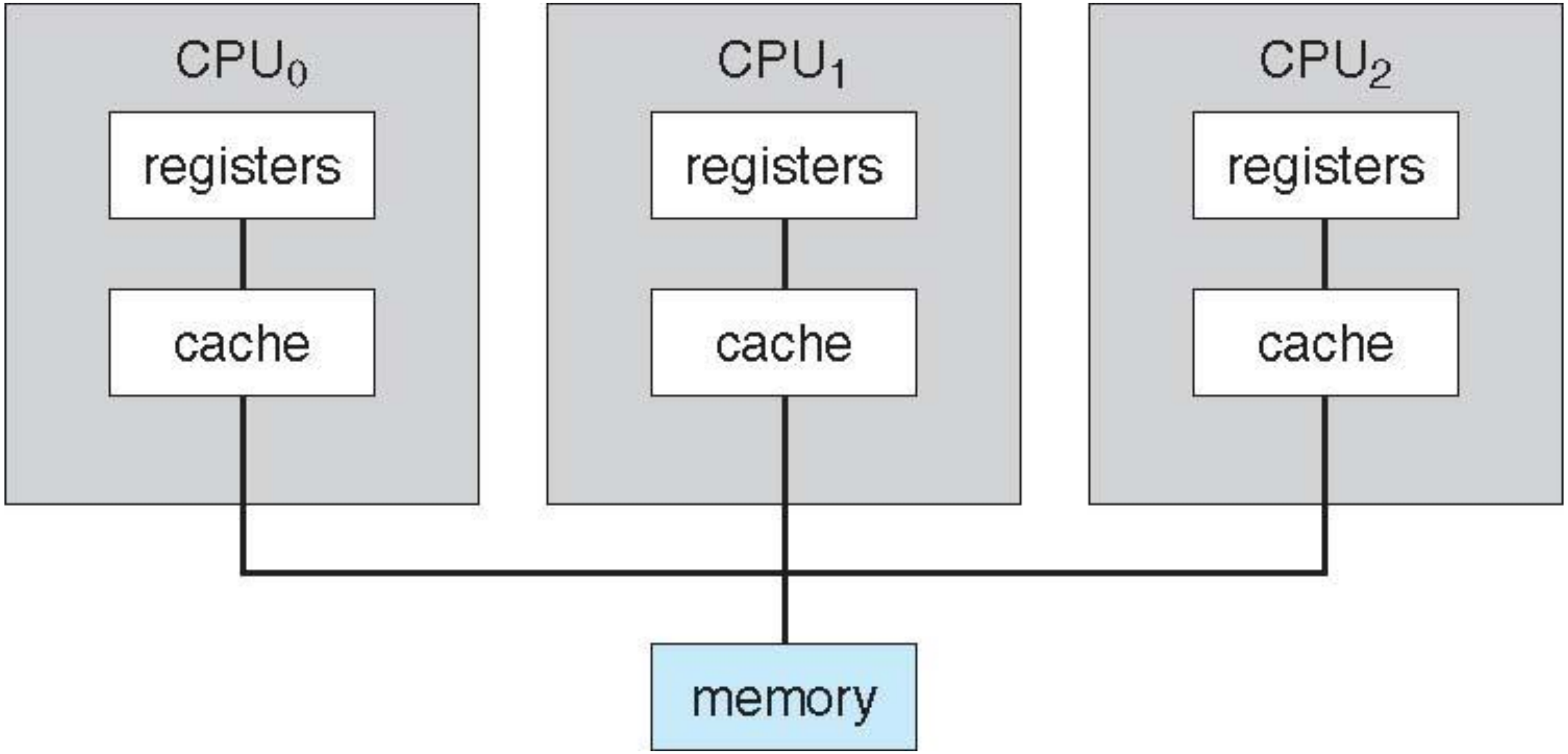


Computer System Architecture

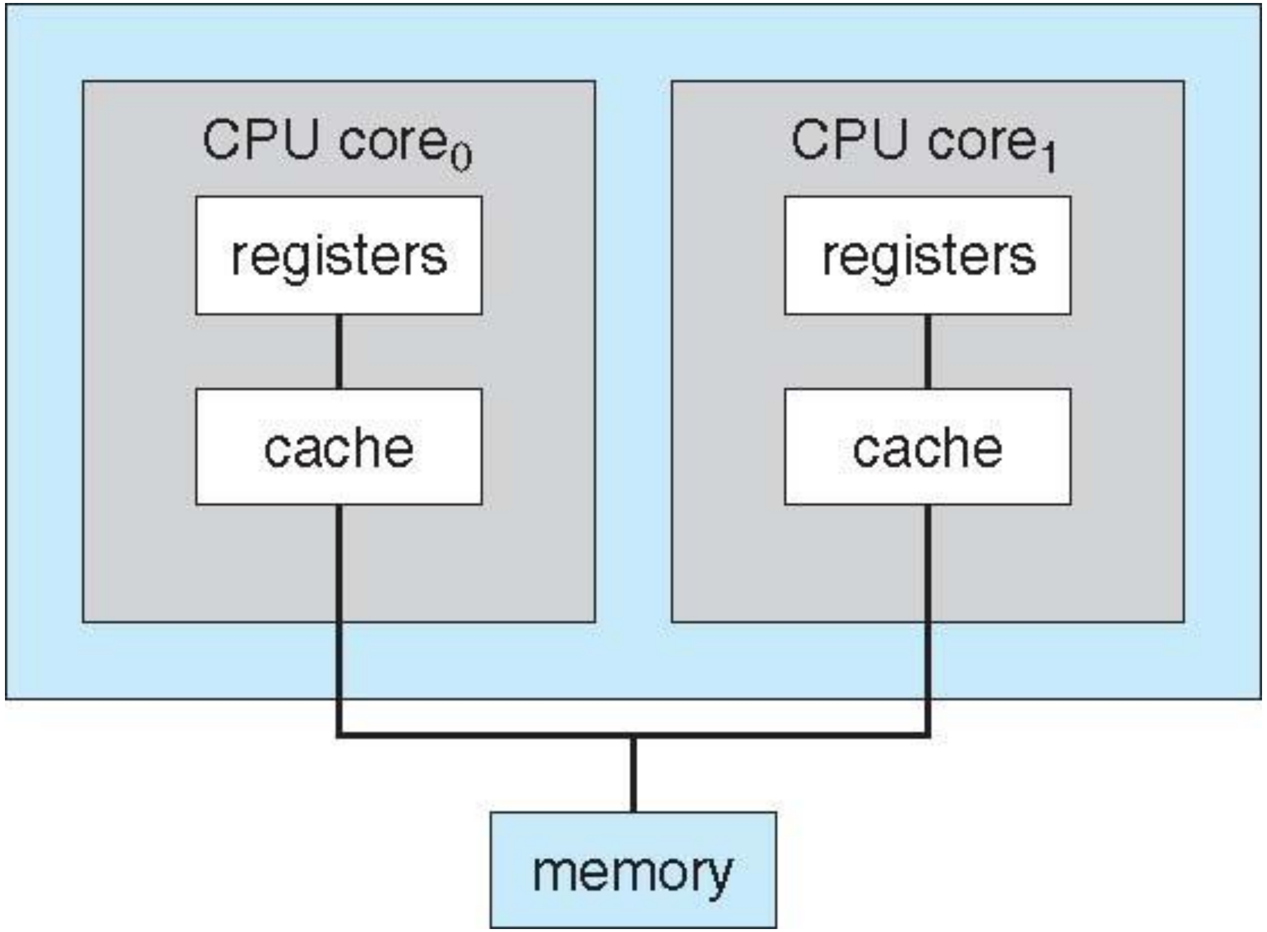
Computer-System Architecture

- Previously, most systems use a single general-purpose processor
  - Most systems have **special purpose processors** as well
- **Multiprocessor** systems largely popular nowadays, predecessors almost if not already obsolete.
  - Also known as **parallel** systems, **tightly-coupled** systems
  - Advantages:
    - Increased **throughput**, economy of **scale**, increased **reliability**.
  - Two types:
    - **Asymmetric** or **Symmetric**

Symmetric Multiprocessing Architecture



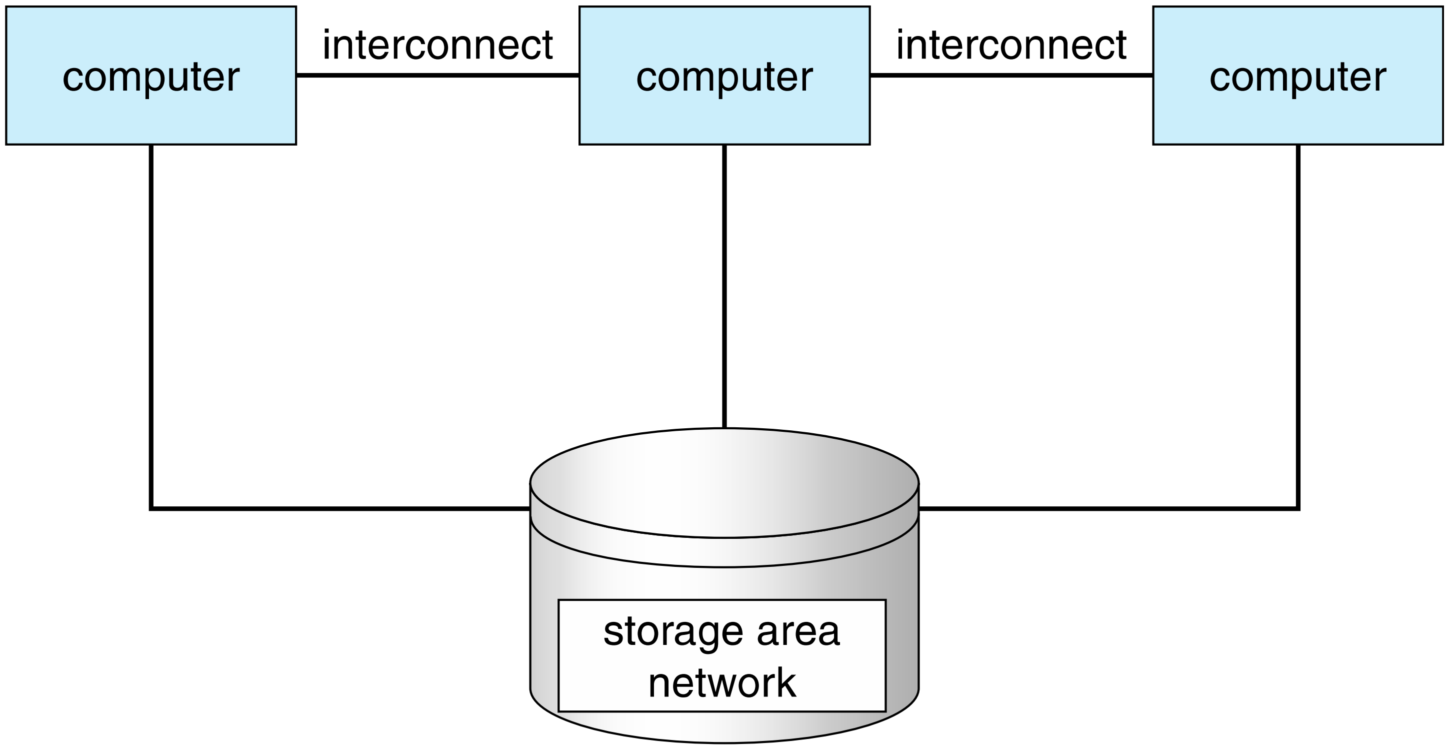
A Dual-Core Design



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
  - May be **asymmetric** or **symmetric**
  - Some clusters are for **high-performance computing** (**HPC**)
  - Some have **distributed lock manager** (DLM) to avoid conflicting operations

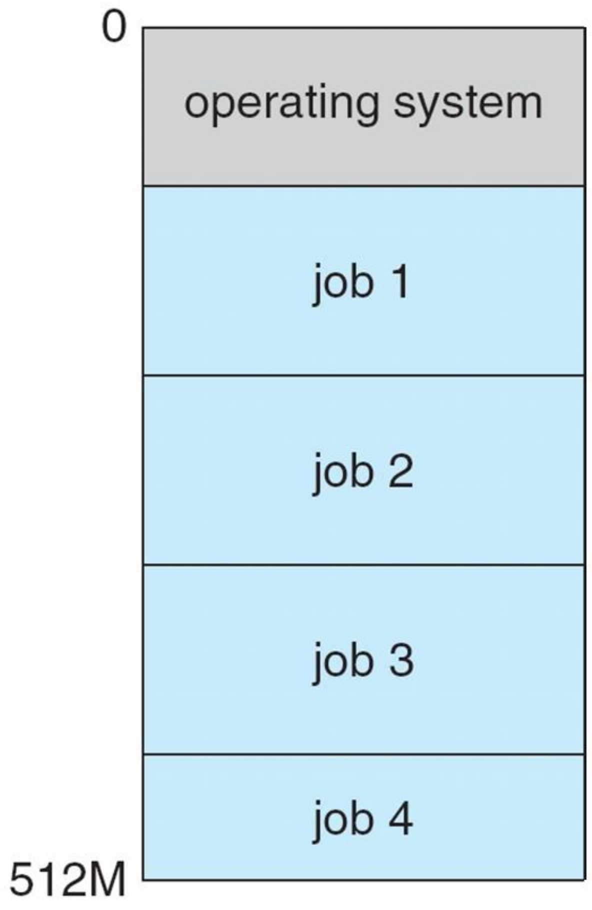
Clustered Systems



Operating Systems Structure

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



Operating System Structure

- **Timesharing** (**multitasking**) is logical extension in which CPU switches jobs so frequently 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 running in memory → **process**
  - If several jobs ready to run at the same time → **CPU scheduling**
  - If process do not fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Operating System Operations

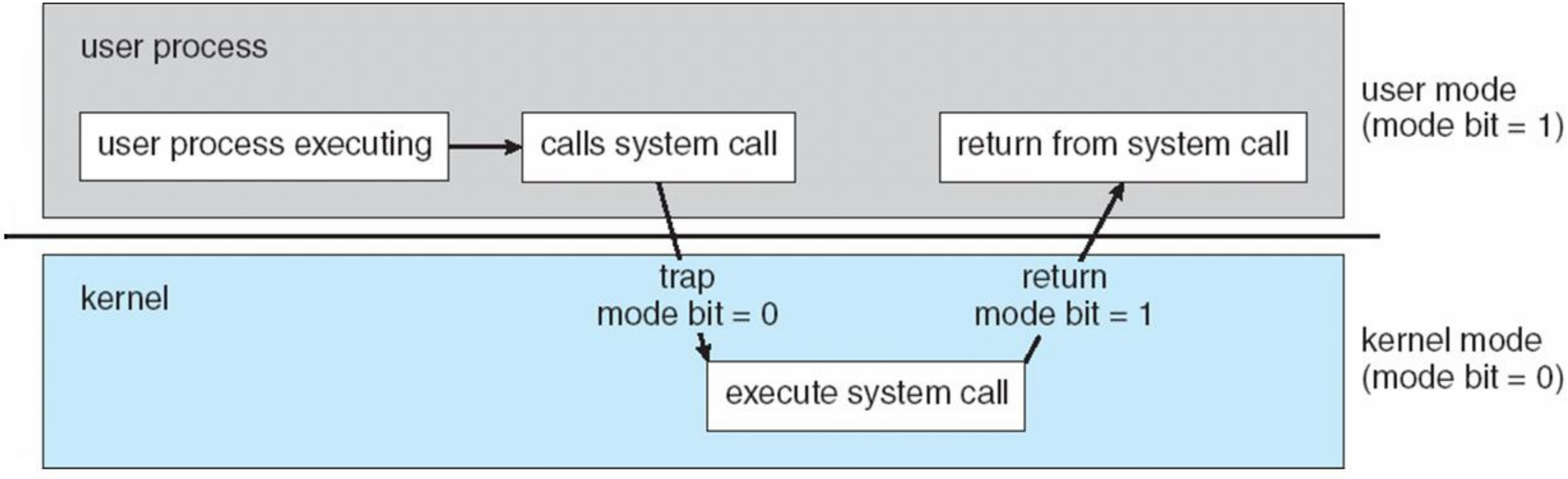
Operating Systems Operations

- **Interrupt driven** (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (**execution** 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

Operation System Operations

- **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
- Increasing CPUs support multi mode operations
  - i.e., **virtual machine manager** (**VMM**) mode for guest VMs

Transition from User to Kernel Mode



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
  - When counter zero, generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time

Resource Management

Process Management

- A **process** is a program in execution. It is a unit of work within the system.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- 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 users, some OS running concurrently on one or more CPUs

Process Management

- 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

- To execute a program all (or part) of the instructions must be in memory
- All (or part) of the data that is needed by the program must be in memory
- Memory management determines what is in memory and 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)
- 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 directories
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Mass Storage Management

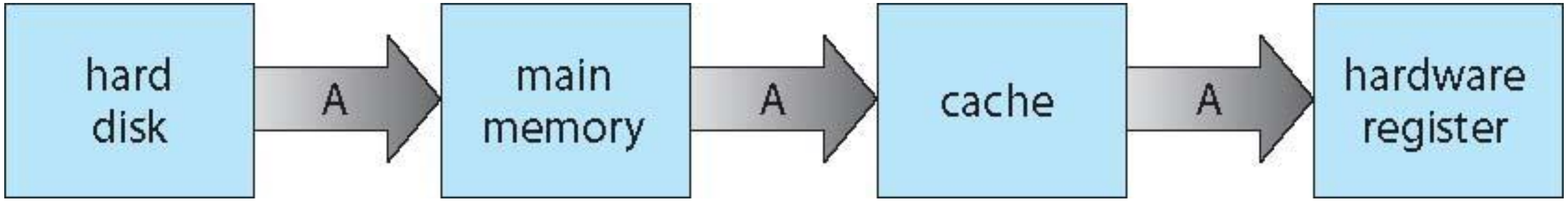
- Usually, disks used to store data that does not fit in main memory or data that must be kept for a long period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities:
    - **Free-space management**, **Storage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

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

Migration of data “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
- Distributed environment situation even more complex
    - Several copies of a datum can exist



I/O Subsystem

- One purpose of OS is to hide the 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 others)
  - 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 DOS, 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
  - User ID then associated with files, processes
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, and also associated with processes and files
  - **Privilege escalation** allows user to change to effective ID with more rights

Computing Environments

Computing Environments - Traditional

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

Computing Environment - Mobile

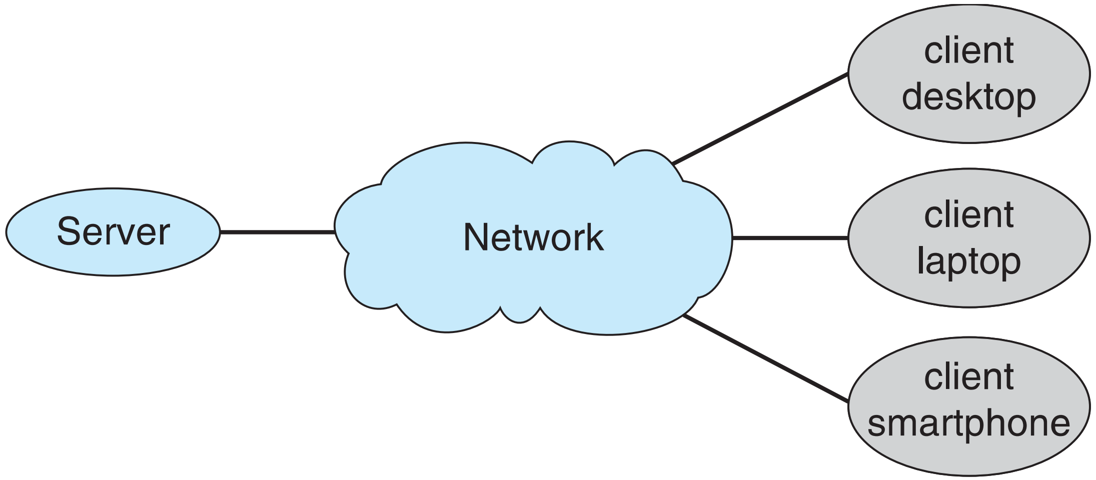
- Handheld smartphones, tablets, etc.
- What is the functional difference between them and a traditional laptop
- Extra OS features (GPS, gyroscope)
- Allows new types of apps like **augmented reality**
- Use IEEE 802.11 wireless, or cellular data networks for connectivity

Computing Environment - Distributed

- Distributed Computing
  - Collection of separate, possibly heterogeneous, systems networked together
    - **Network** is a communications path
      - **Local Area Network** (**LAN**)
      - **Wide Area Network** (**WAN**)
      - **Metropolitan Area Network** (**MAN**)
      - **Personal Area Network** (**PAN**)
  - **Network Operating Systems** 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 system now **servers**, responding to request 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 – P2P

- Another model of distributed systems
- 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

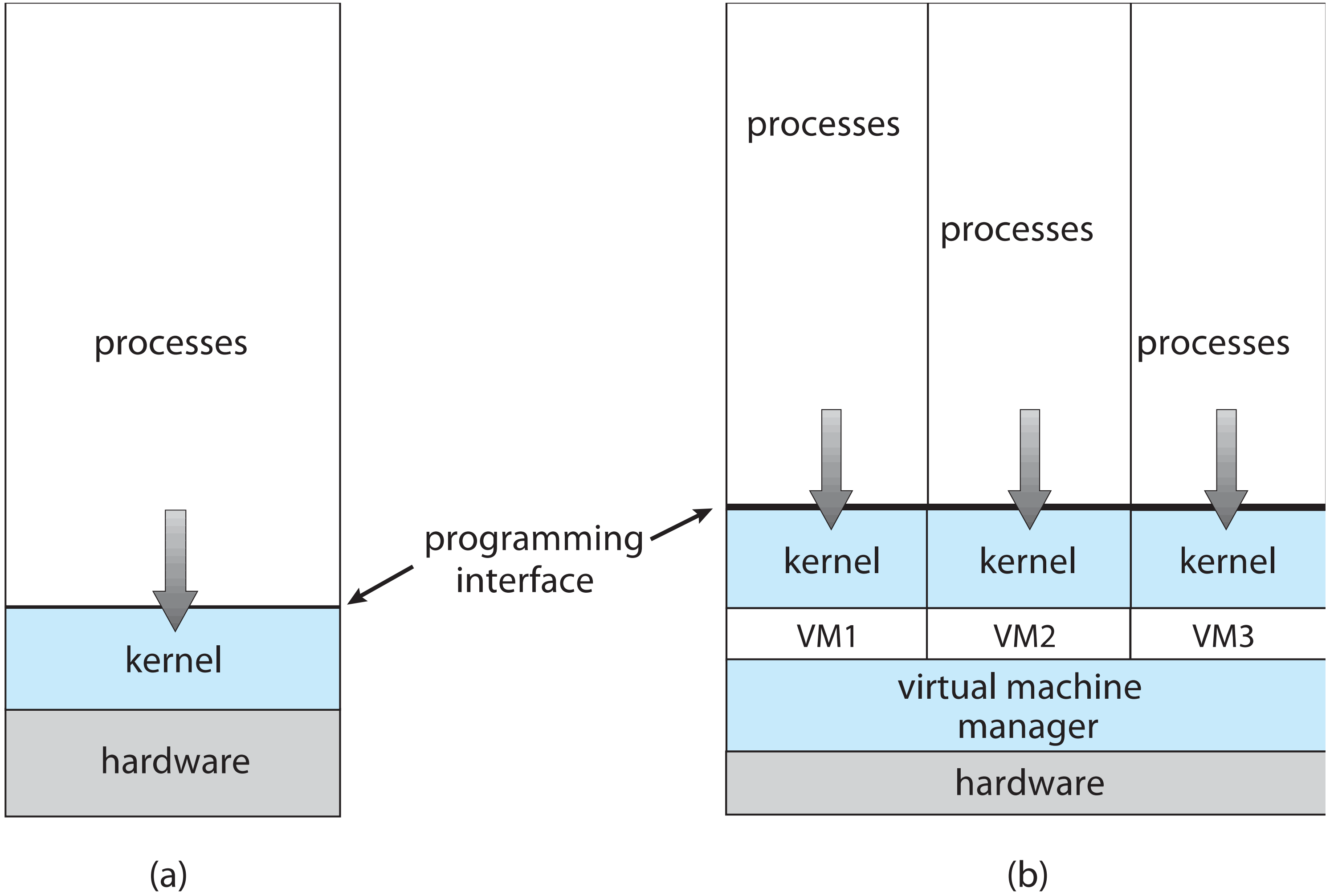
Computing Environments - Virtualization

- Allows operating systems to run applications within other OS
- **Emulation** used when source CPU type different from target type (i.e., PowerPC to Inter x86)
  - Generally slowest method
  - When computer language not compiled to native code – **interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OS 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 for exploration or compatibility
  - Apple laptop running MAC OS X host, Windows as a guest
  - Developing apps for multiple OS 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

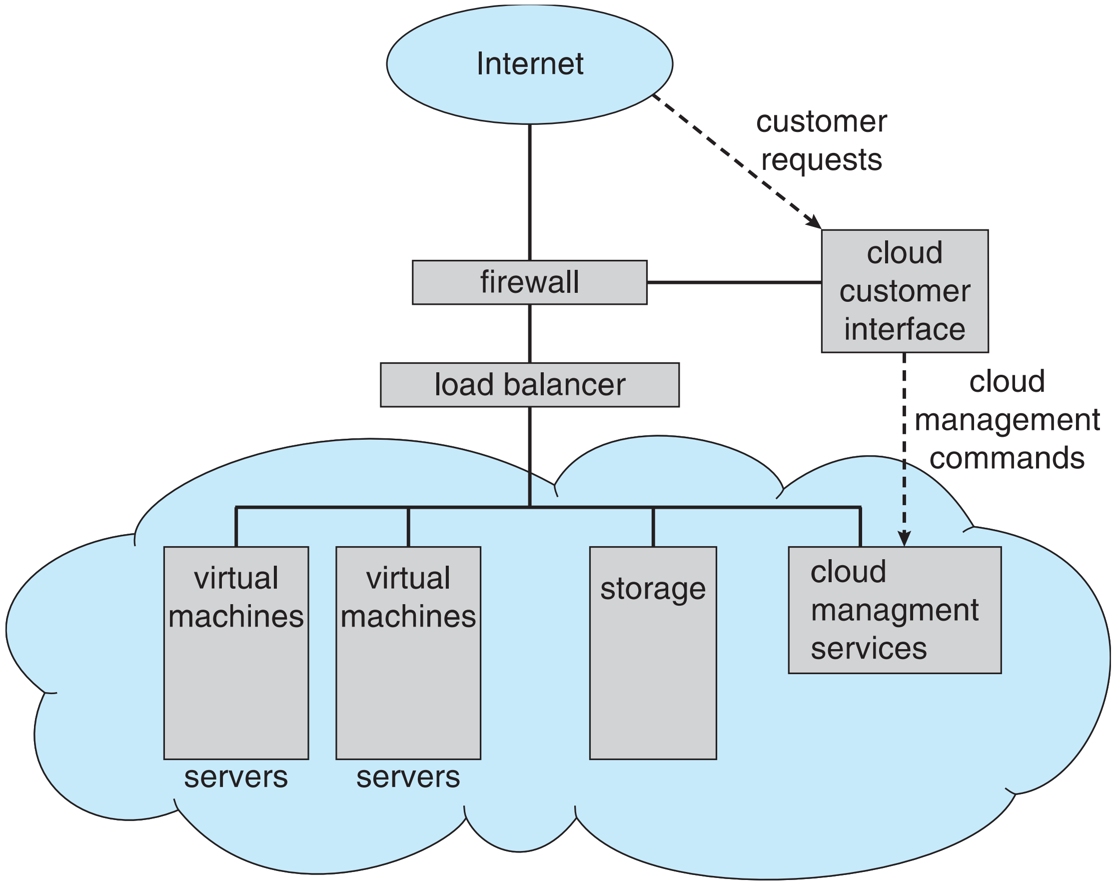


Computing Environments – Cloud

- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization because it uses virtualization as the base for its functionality
- Many types:
  - Public cloud
  - Private cloud
  - Hybrid Cloud
  - Software-as-a-Service
  - Platform-as-a-Service
  - Infrastructure-as-a-Service

Computing Environments – Cloud

- **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 Environment – 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 environment as well
  - Some have OS, 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 Managements (DRM) movement
- Started Free Software Foundation (FSF), which has “copyleft” GNU Public License (GPL)
- Examples include GNU/Linux and BSD Linux (including core of Mac OS X), and many more
- Can use VMM like Vmware Player (Free on Windows), Virtualbox