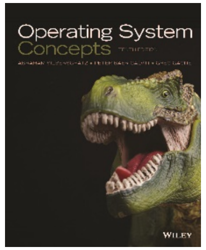
Amirkabir University of Technology
(Tehran Polytechnic)
Department of Computer Engineering

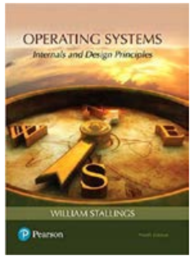
Operating Systems Course

Hamid R. Zarandi
h_zarandi@aut.ac.ir

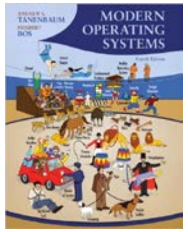
Textbooks



Silberschatz et. al. ***Operating System Concepts***
9th Edition, Wiley, 2018.



Stallings, *Operating Systems: Internal and Design Principles*
9th Edition, Pearson, 2018.



Tanenbaum, *Modern Operating Systems*
4th Edition, Prentice-Hall, 2014.

Course highlights & grading

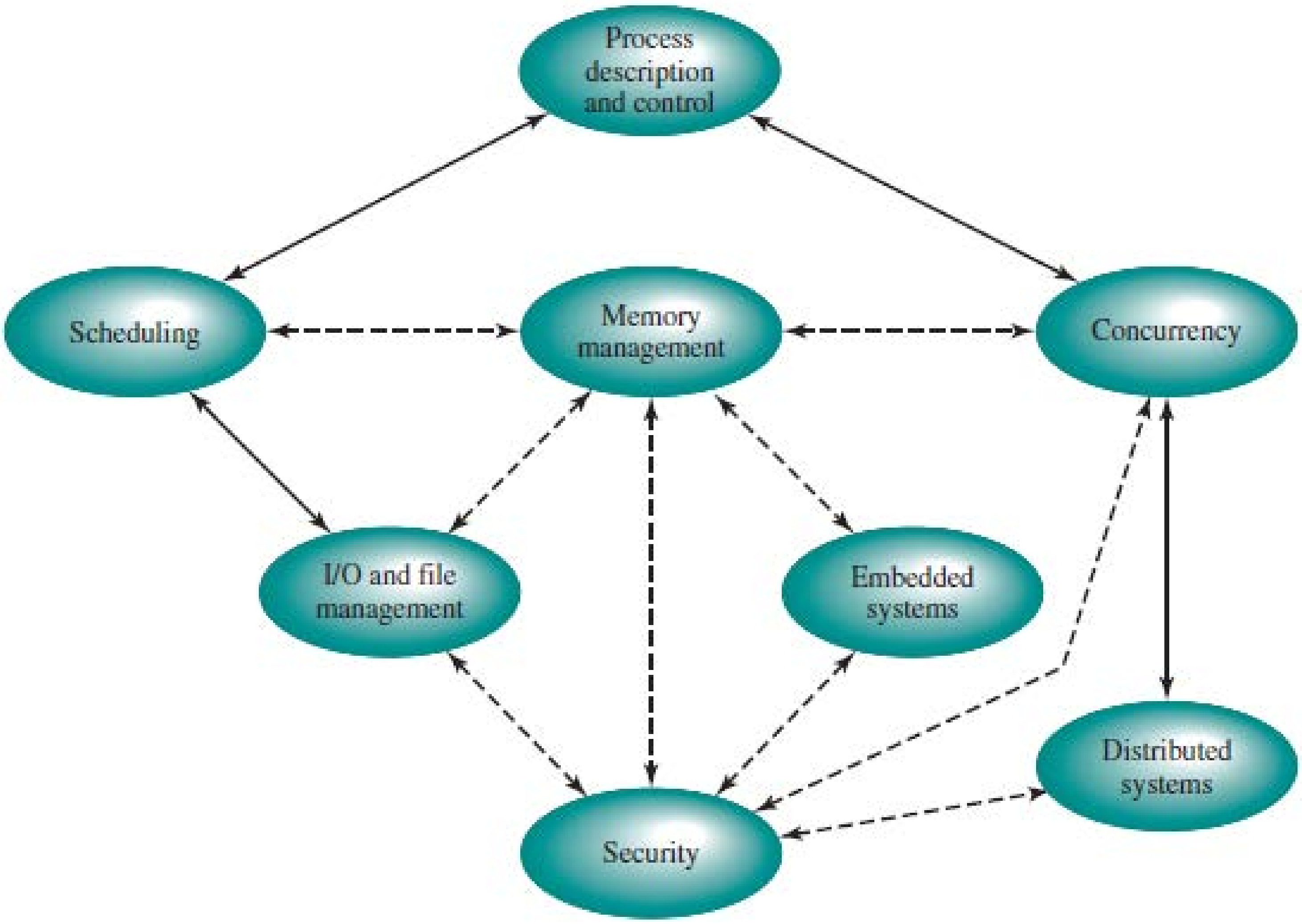
➢Grading details

- Mid-term+Quiz (6)
- Final + Quiz (9)
- Homeworks (3)
- Project(s) (1)
- Presence (1)

➢Highlights

- 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

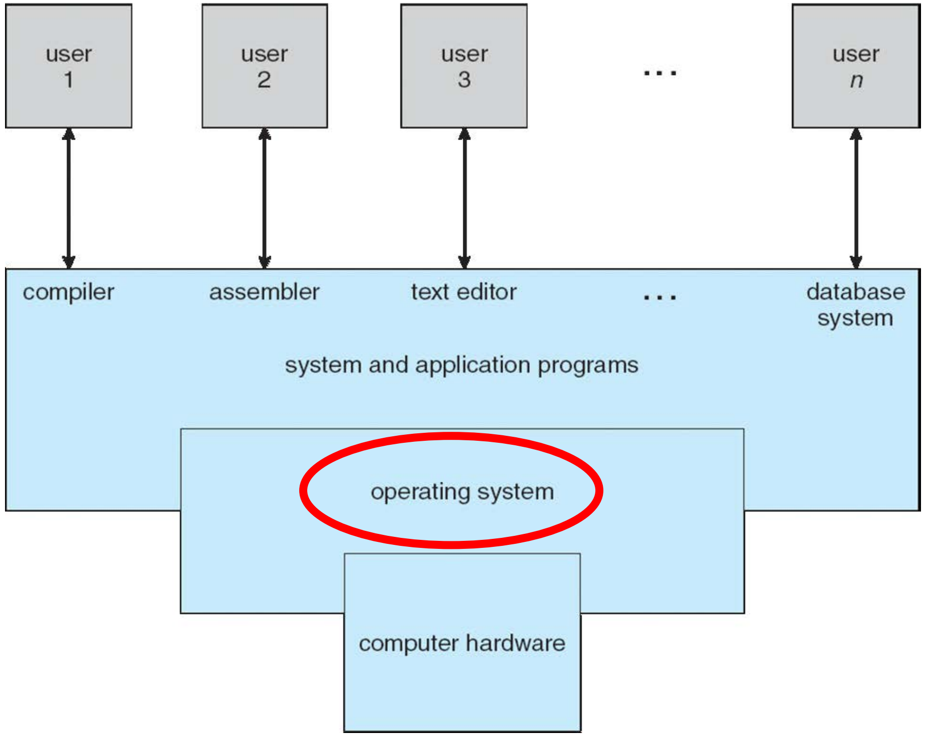
OS topics


Process description and control
Scheduling
Memory management
Concurrency
I/O and file management
Embedded systems
Distributed systems
Security

Introductions to Operating Systems

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware
  - User can execute programs conveniently & efficiently



What operating systems do?

- 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
- 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
- Some computers have little or no user interface, such as embedded computers in devices and automobiles

Operating system definition

- OS is a resource allocator
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use

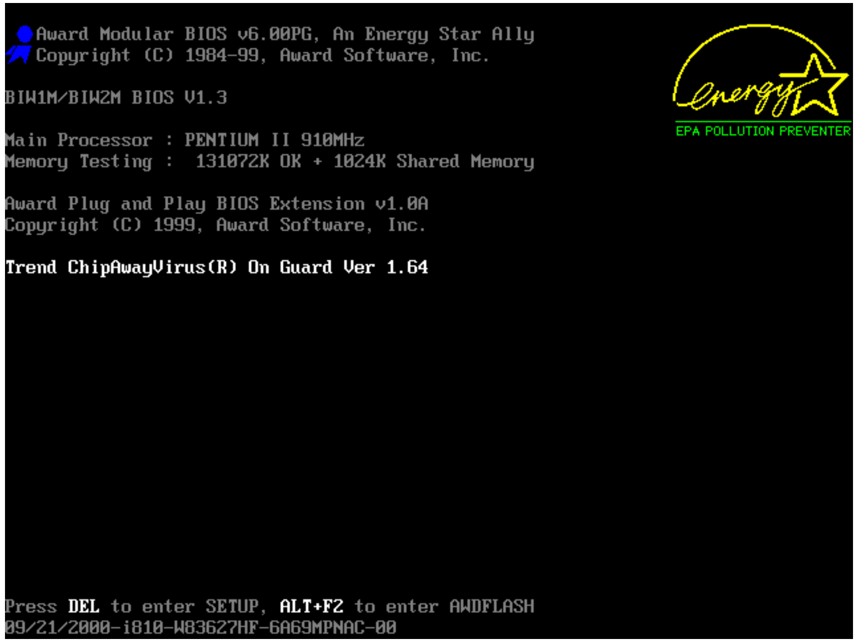
- OS is a control program
  - Controls execution of programs to prevent errors and improper use of the computer

No uniform definition!

- **No universally accepted definition!**
- **"The one program running at all times on the computer" is the kernel.**

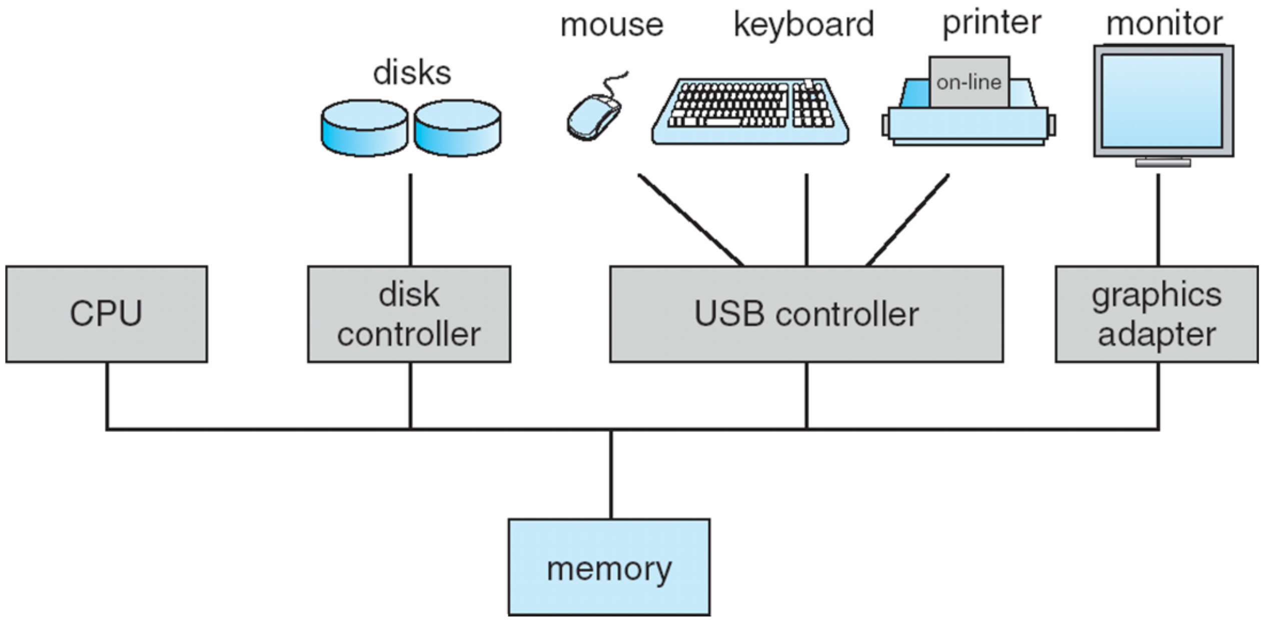
Computer startup

- 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

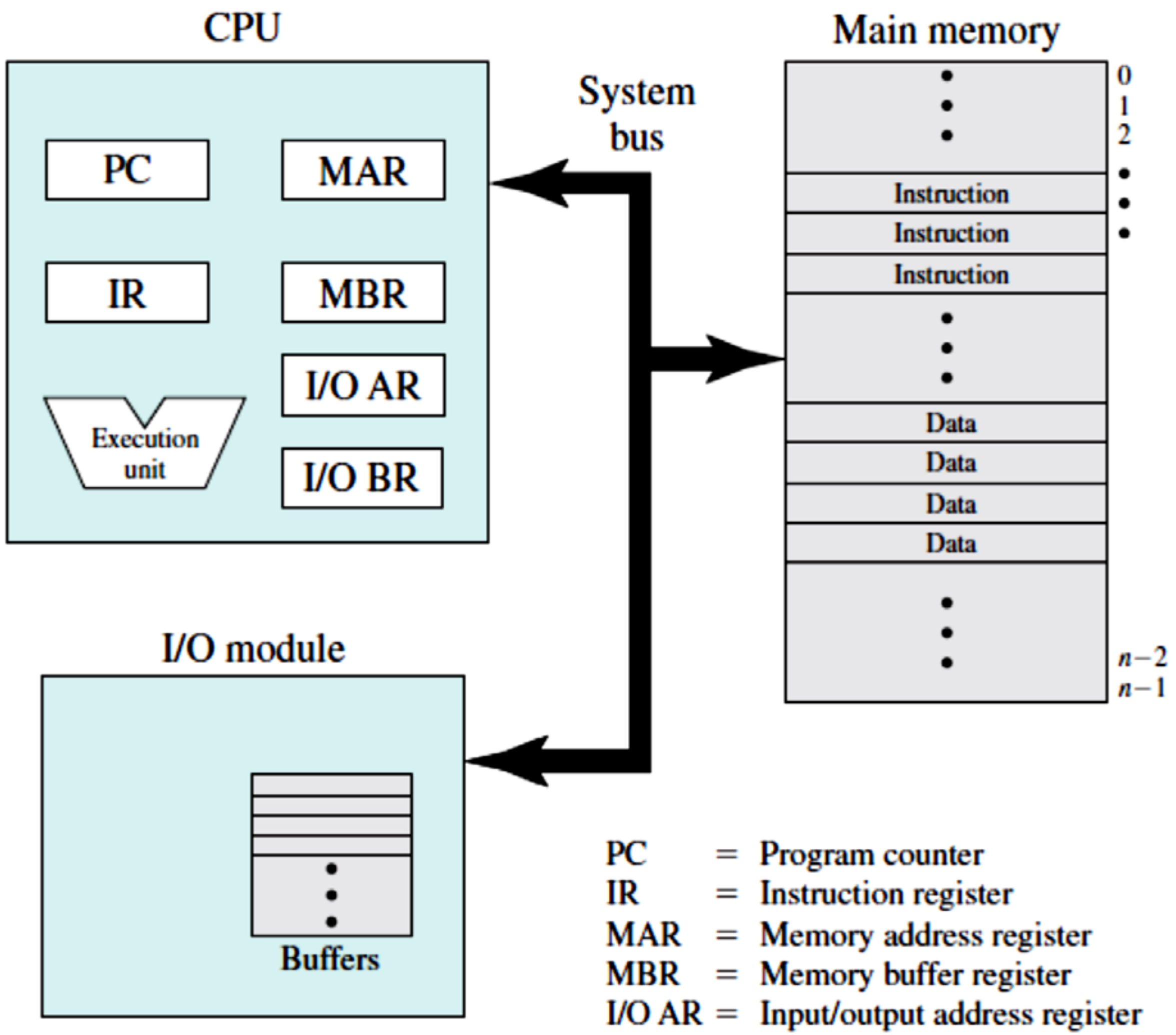


Computer system organization

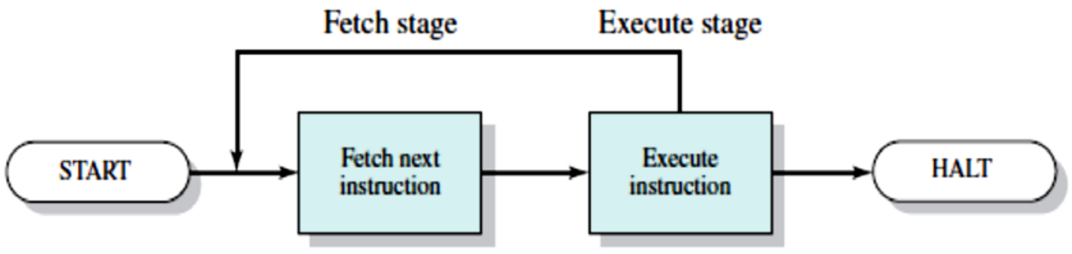
- **Computer-system operation**
  - **One or more CPUs, device controllers connect through common bus providing access to shared memory**
  - **Concurrent execution of CPUs and devices competing for memory cycles**



Computer system, basic elements



PC = Program counter
IR = Instruction register
MAR = Memory address register
MBR = Memory buffer register
I/O AR = Input/output address register
I/O BR = Input/output buffer register



Computer-system operation

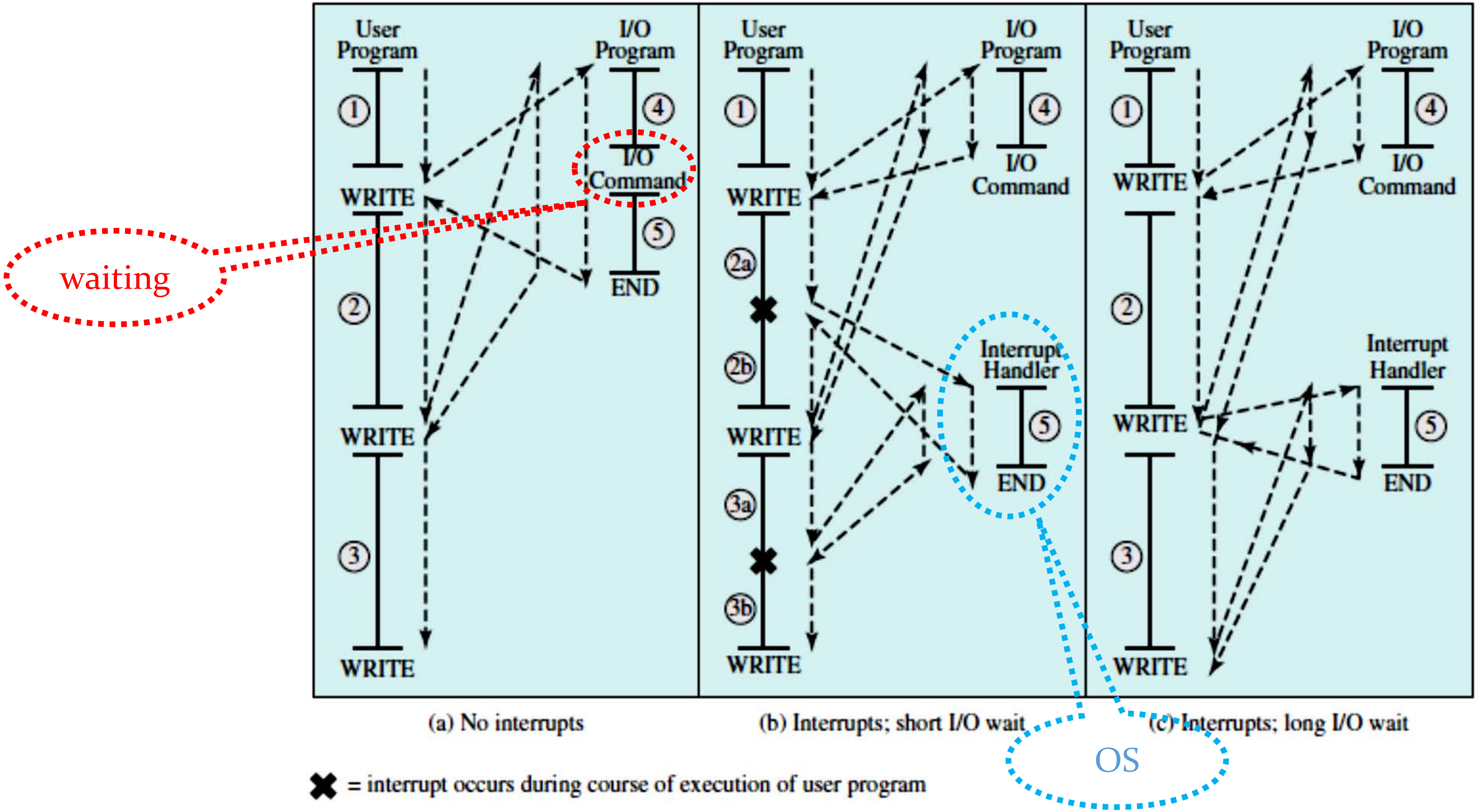
- I/O devices and the CPU can execute concurrently
- Each device controller is in charge of a particular device type
- Each device controller has a local buffer
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller
- Device controller informs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

OS is interrupt driven!

Interrupts

Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.
I/O	Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.
Hardware failure	Generated by a failure, such as power failure or memory parity error.

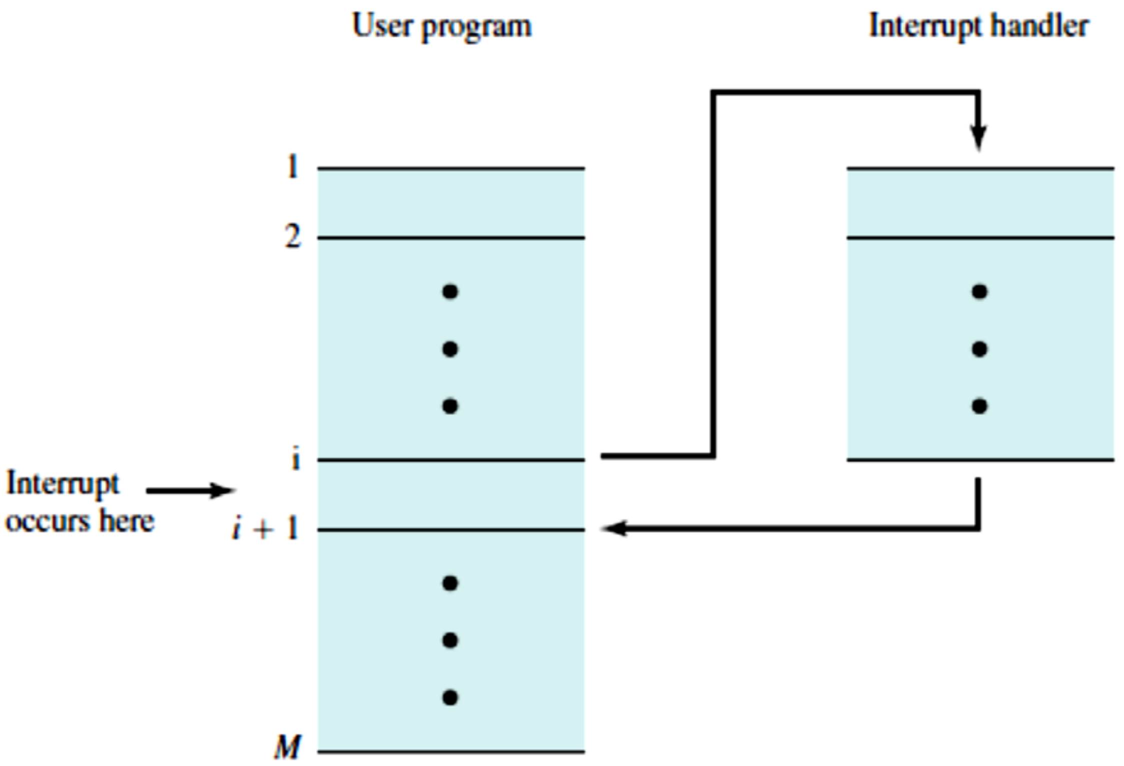
Control flow (w/wo interrupts)

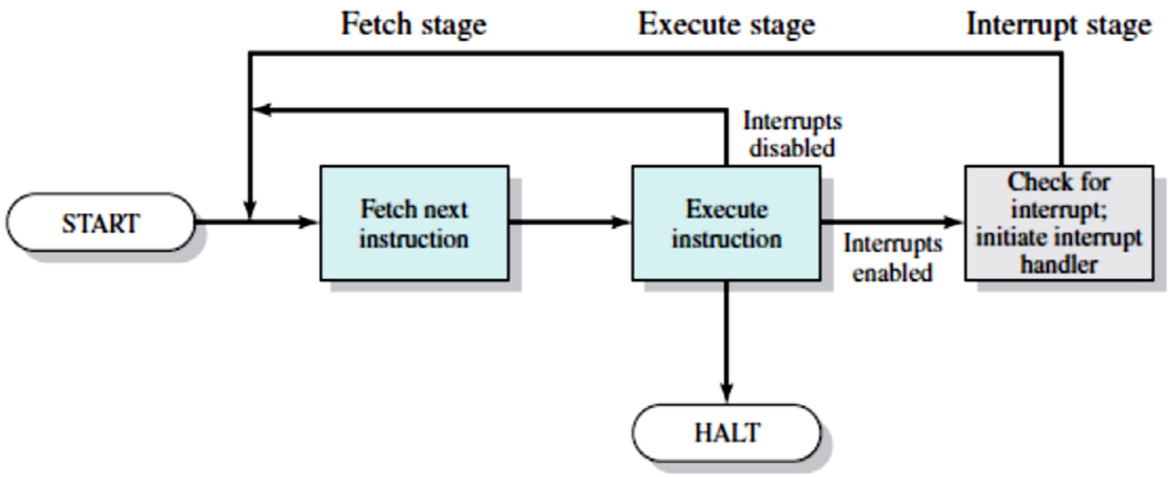


(a) No interrupts (b) Interrupts; short I/O wait (c) Interrupts; long I/O wait

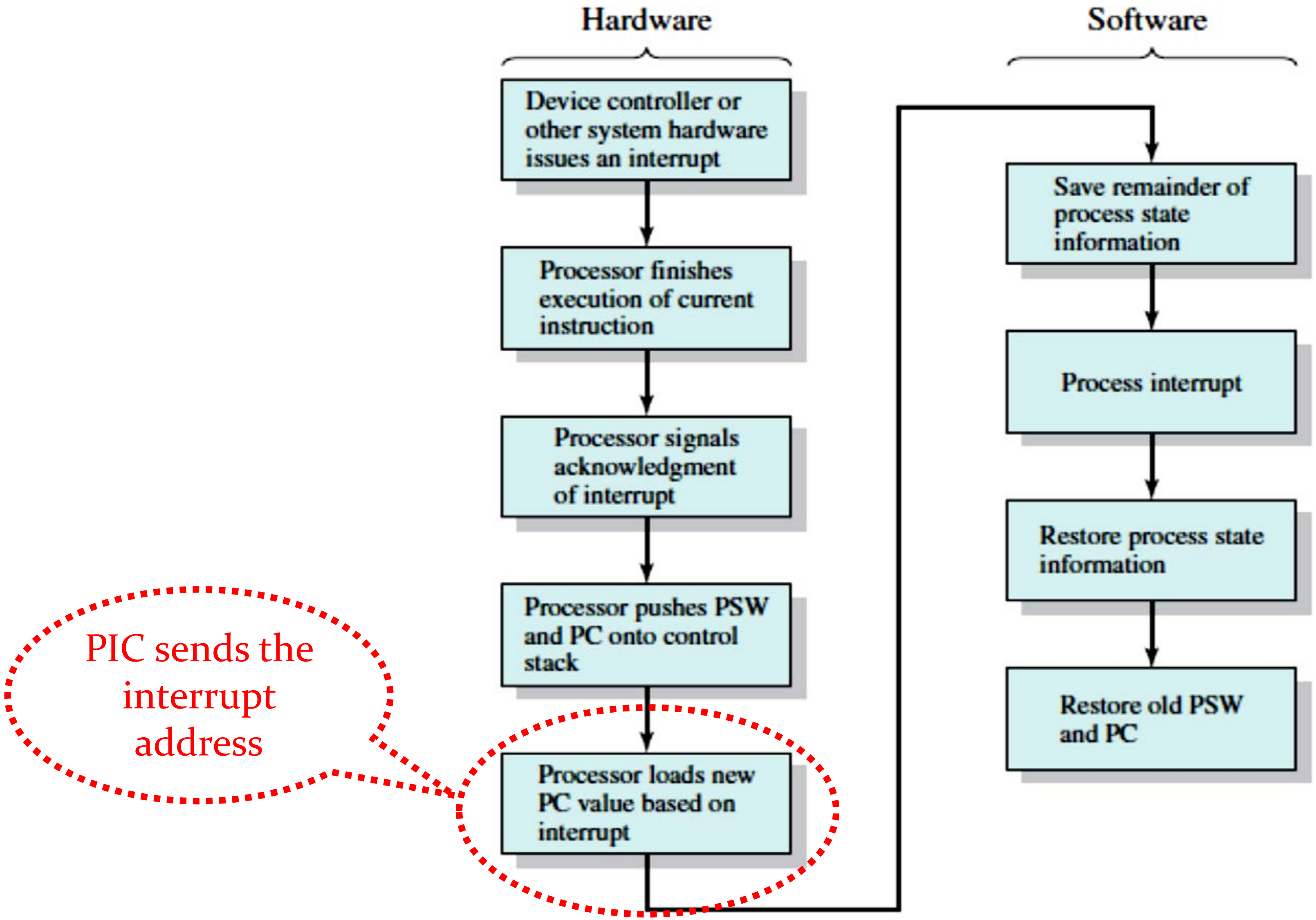
✖ = interrupt occurs during course of execution of user program

Handling interrupts

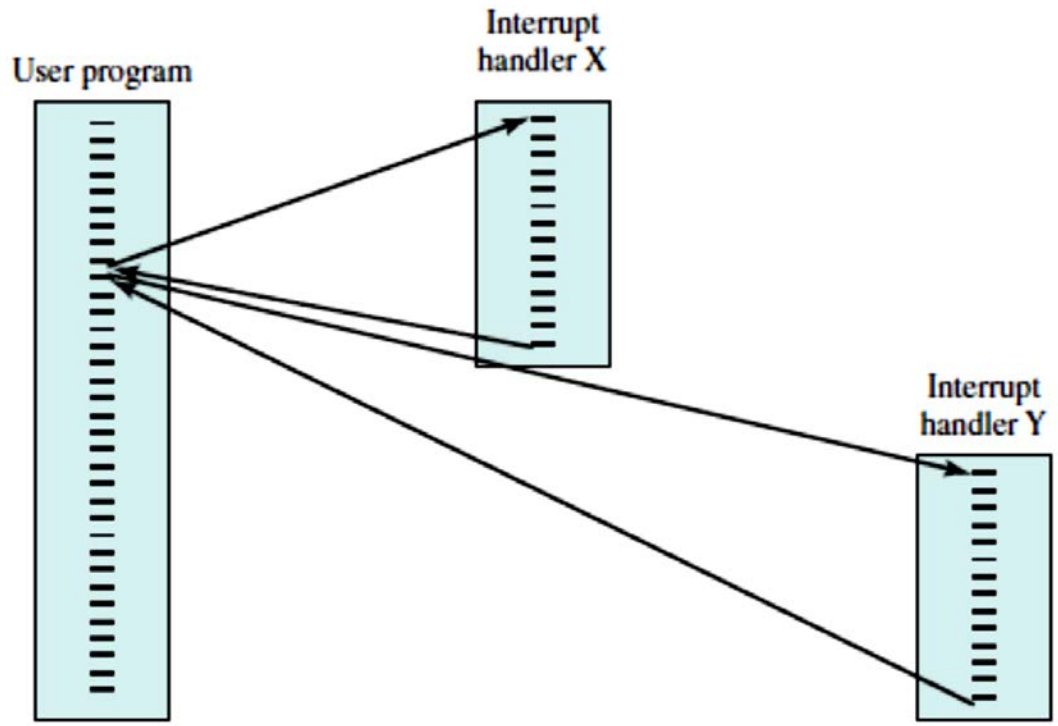




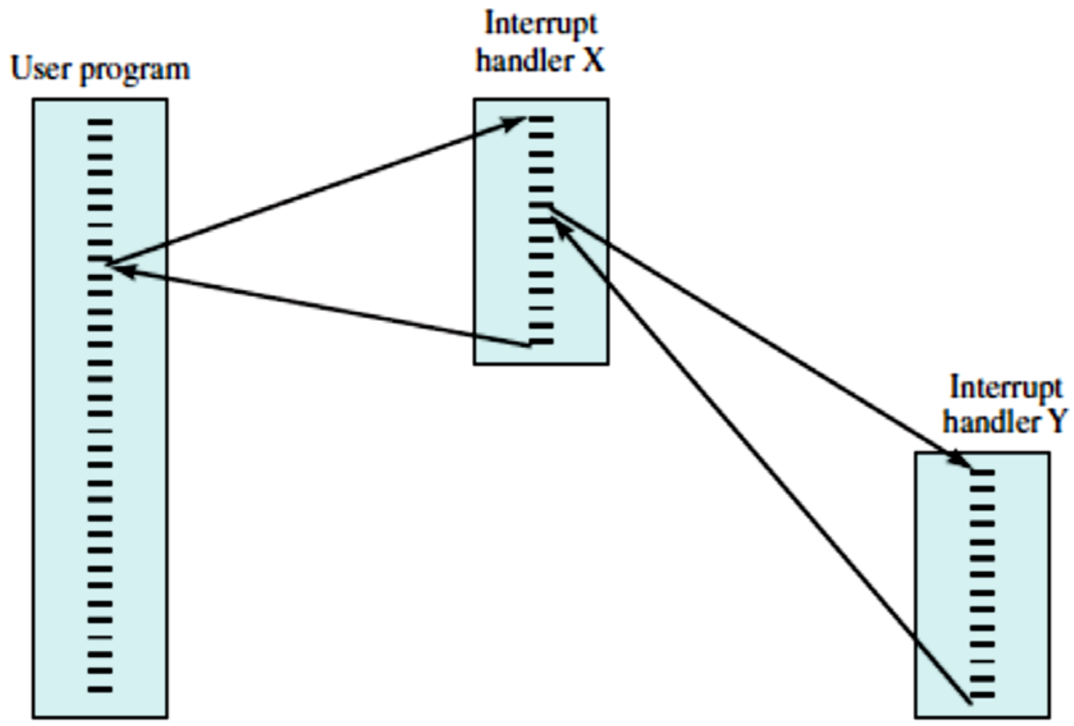
Interrupt handling



Multiple interrupts



(a) Sequential interrupt processing



(b) Nested interrupt processing

Disabling interrupt can be done to prevent future ones.

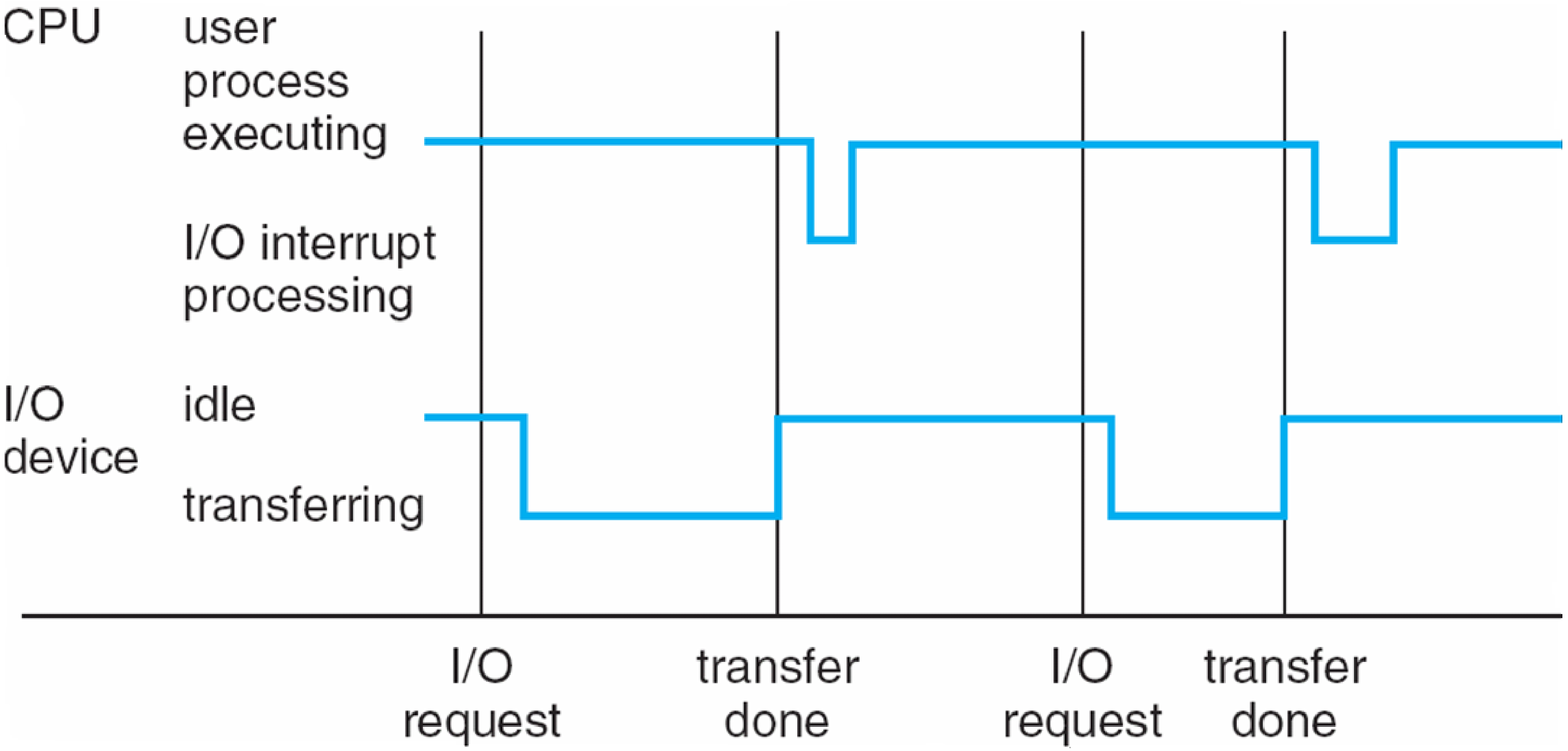
Common functions of interrupts

- 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 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



Storage Definitions and Notation Review

Storage definitions and notation review

The basic unit of computer storage is the **bit**. A bit can contain one of two values, 0 and 1. All other storage in a computer is based on collections of bits. Given enough bits, it is amazing how many things a computer can represent: numbers, letters, images, movies, sounds, documents, and programs, to name a few. A **byte** is 8 bits, and on most computers it is the smallest convenient chunk of storage. For example, most computers don't have an instruction to move a bit but do have one to move a byte. A less common term is **word**, which is a given computer architecture's native unit of data. A word is made up of one or more bytes. For example, a computer that has 64-bit registers and 64-bit memory addressing typically has 64-bit (8-byte) words. A computer executes many operations in its native word size rather than a byte at a time.

Computer storage, along with most computer throughput, is generally measured and manipulated in bytes and collections of bytes.
A **kilobyte**, or **KB**, is 1,024 bytes
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

Computer manufacturers often round off these numbers and say that a megabyte is 1 million bytes and a gigabyte is 1 billion bytes. Networking measurements are an exception to this general rule; they are given in bits (because networks move data a bit at a time).

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
- 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

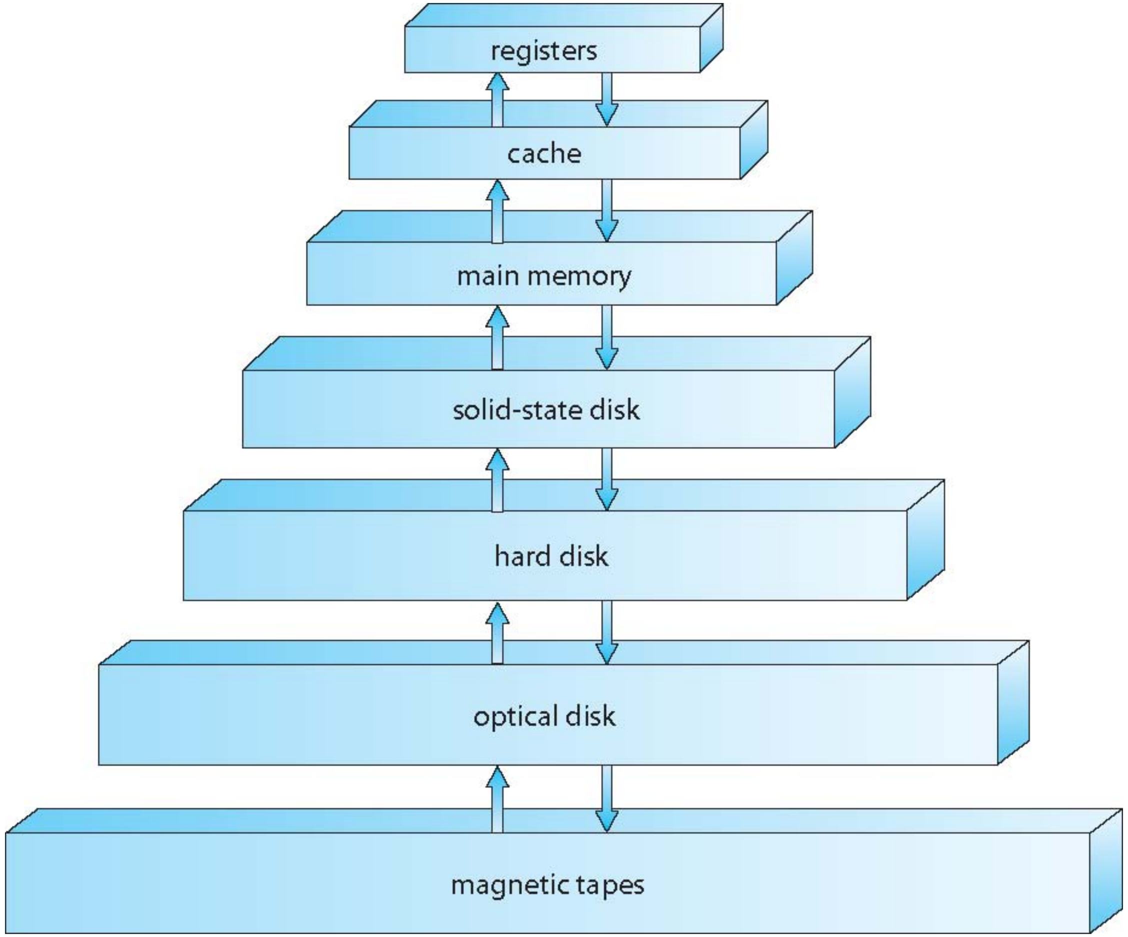
Storage hierarchy

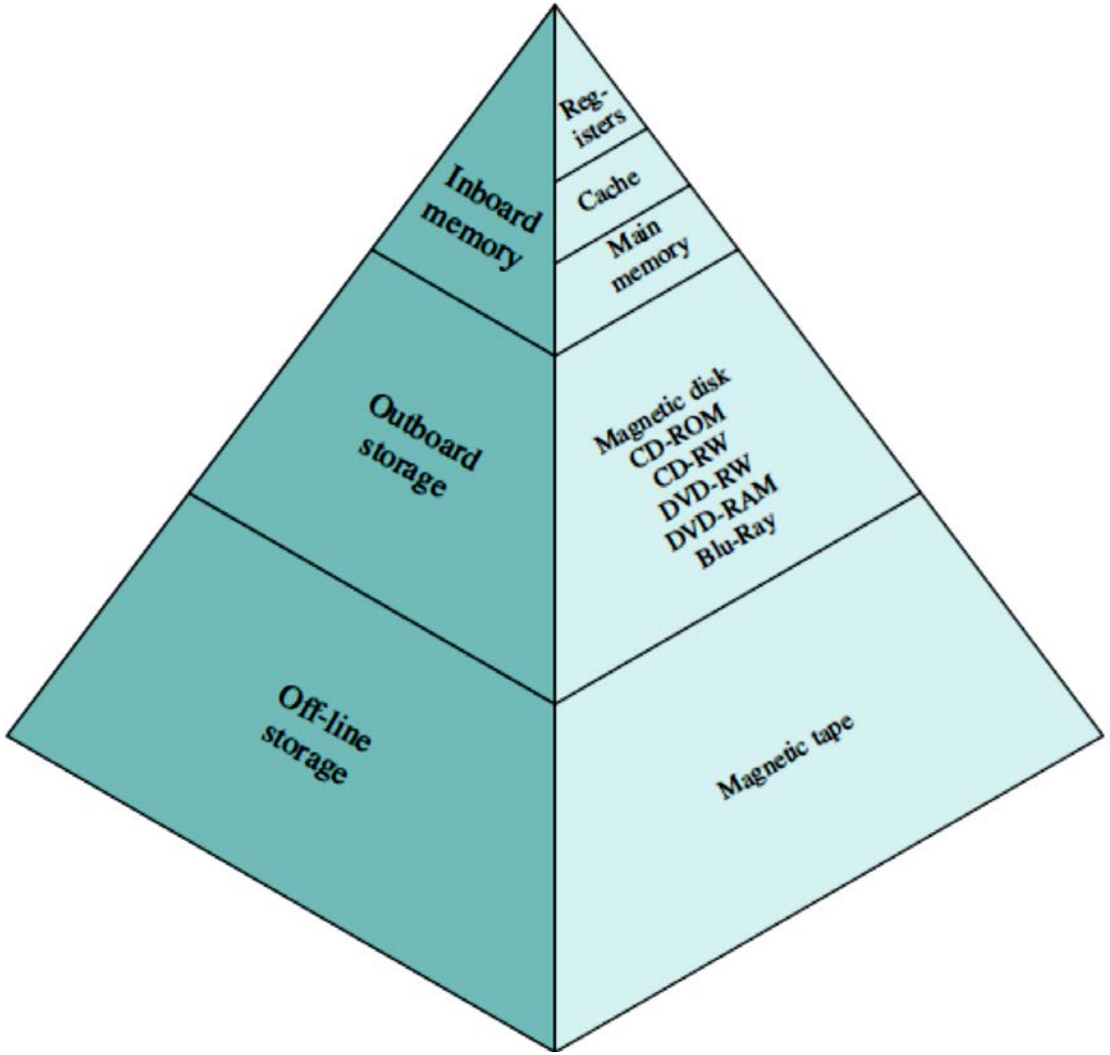
- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility

- 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-device hierarchy





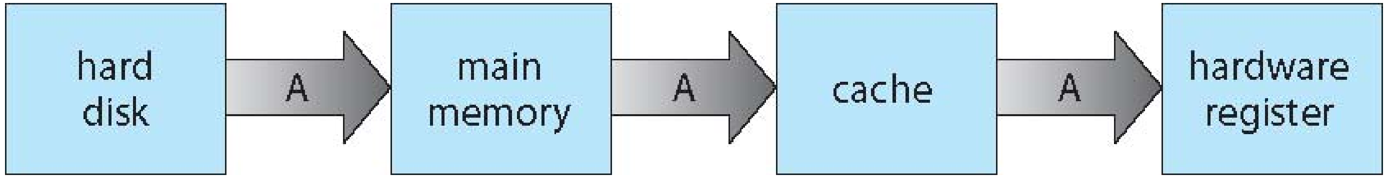
Performance of various levels of storage

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

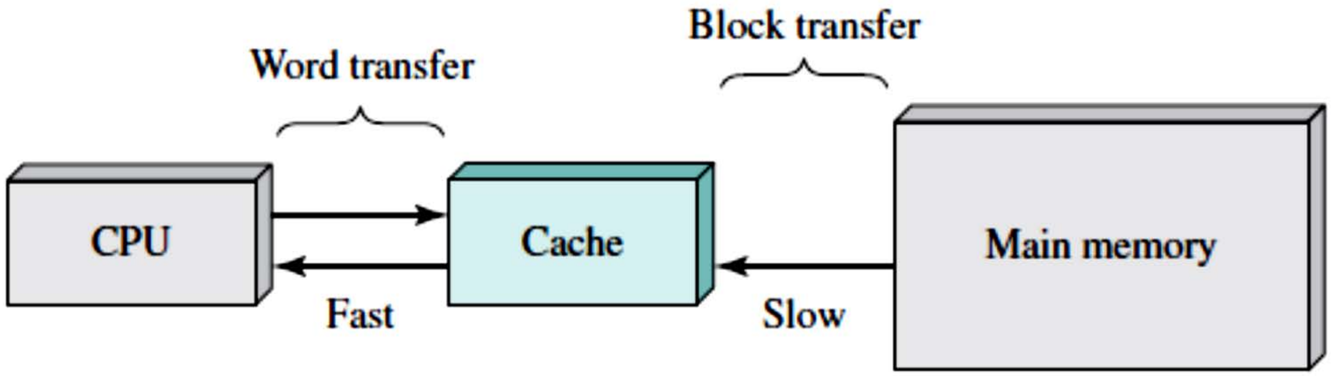
Migration of data "A" from disk to register

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

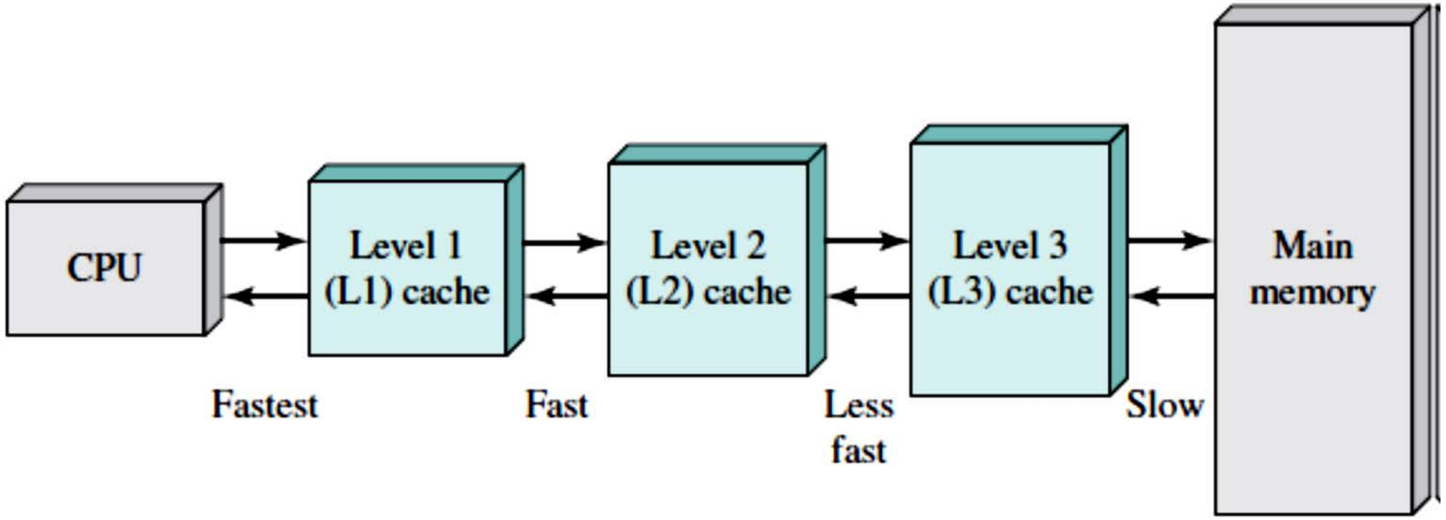


- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist
  - Various solutions covered in Chapter 17

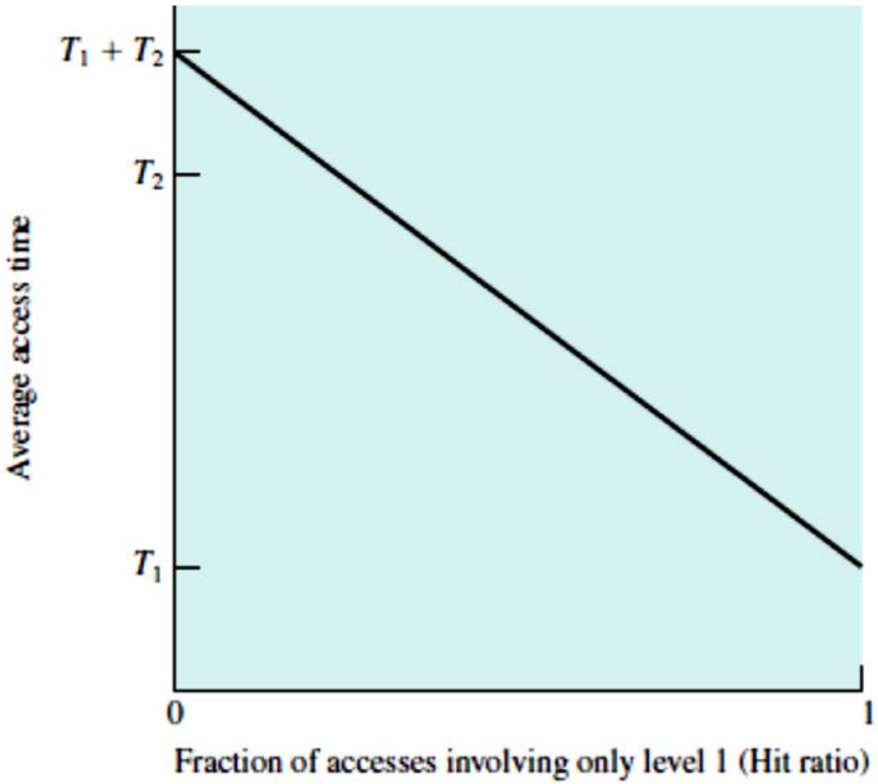
Cache memory



(a) Single cache

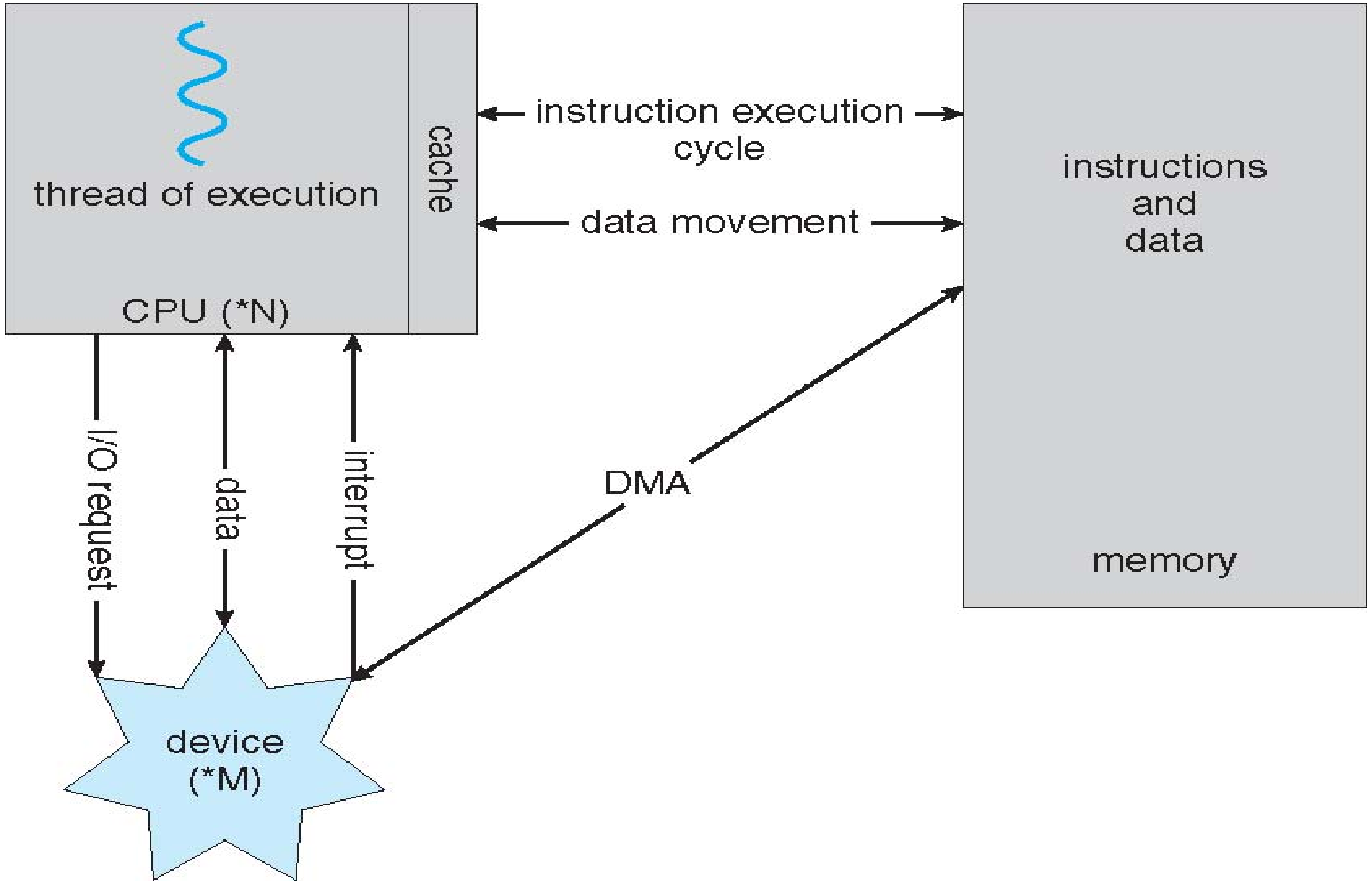


(b) Three-level cache organization



DMA: direct memory access

- IO operations
  - Programmed IO (polling)
  - Interrupt-driven
  - DMA
    - Whether a read or write is requested
    - The address of the I/O device
    - The starting location in memory to read data/write data
    - The number of words



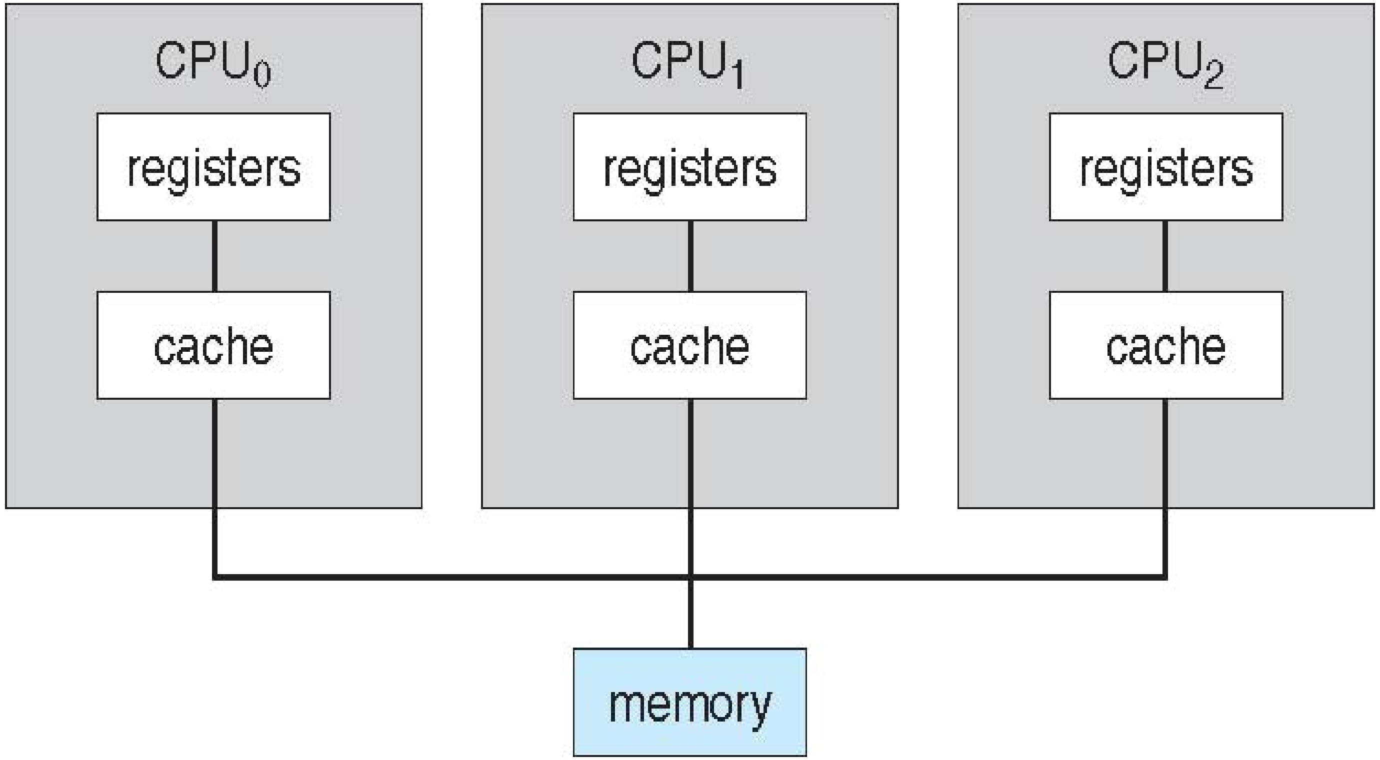
Computer System Architecture

Computer-system architecture

- Most systems use a single general-purpose processor
  - Most systems have special-purpose processors as well

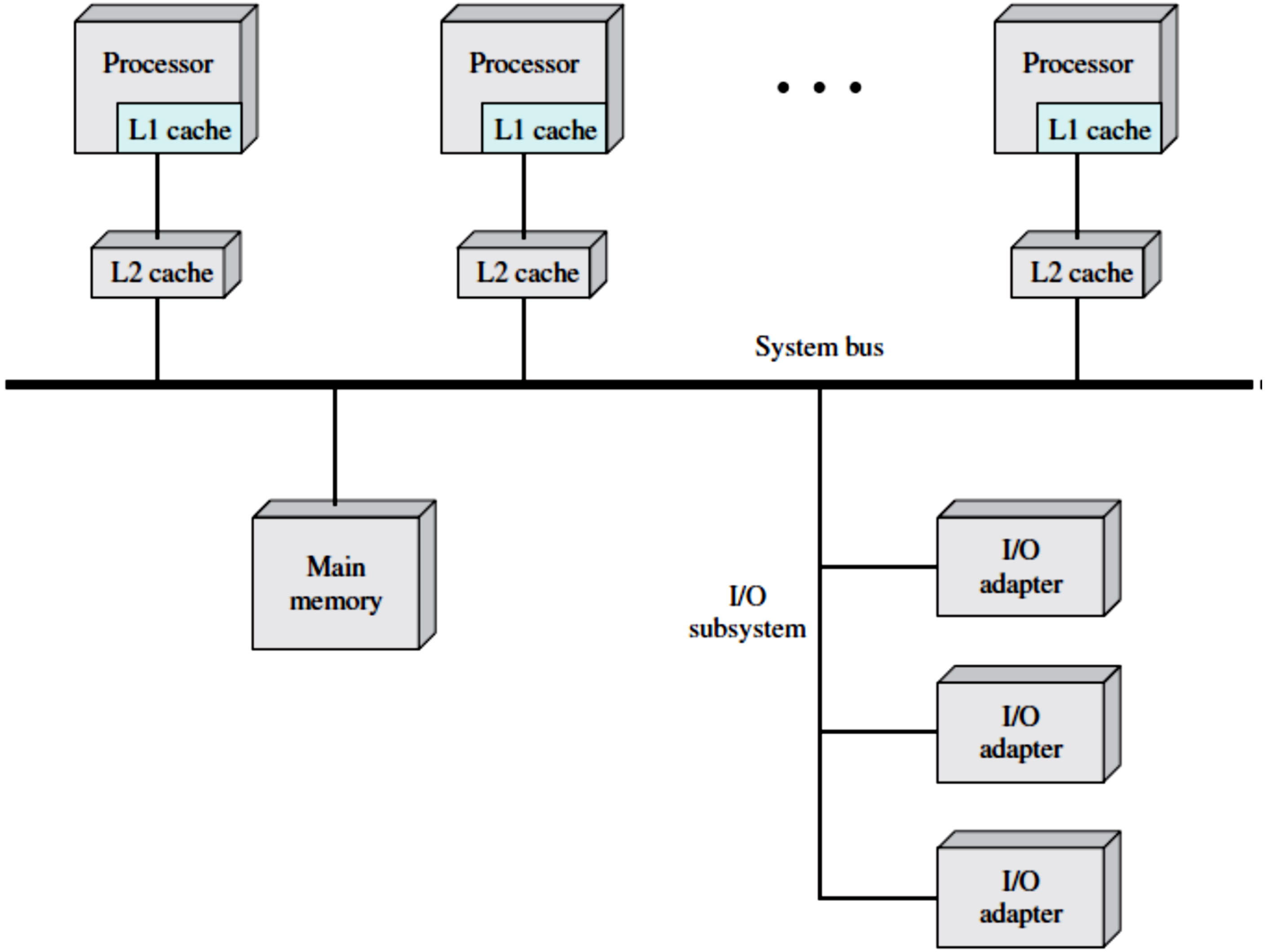
- Multiprocessors systems growing in use and importance
  - Also known as parallel systems, tightly-coupled systems
  - Advantages include:
    1. Increased throughput
    2. Economy of scale – multiprocessors vs. multiple single processor
    3. Increased reliability – graceful degradation or fault tolerance
  - Two types:
    1. Asymmetric Multiprocessing – each processor is assigned a specie task (boss-worker).
    2. Symmetric Multiprocessing – each processor performs all tasks

Symmetric multiprocessing architecture



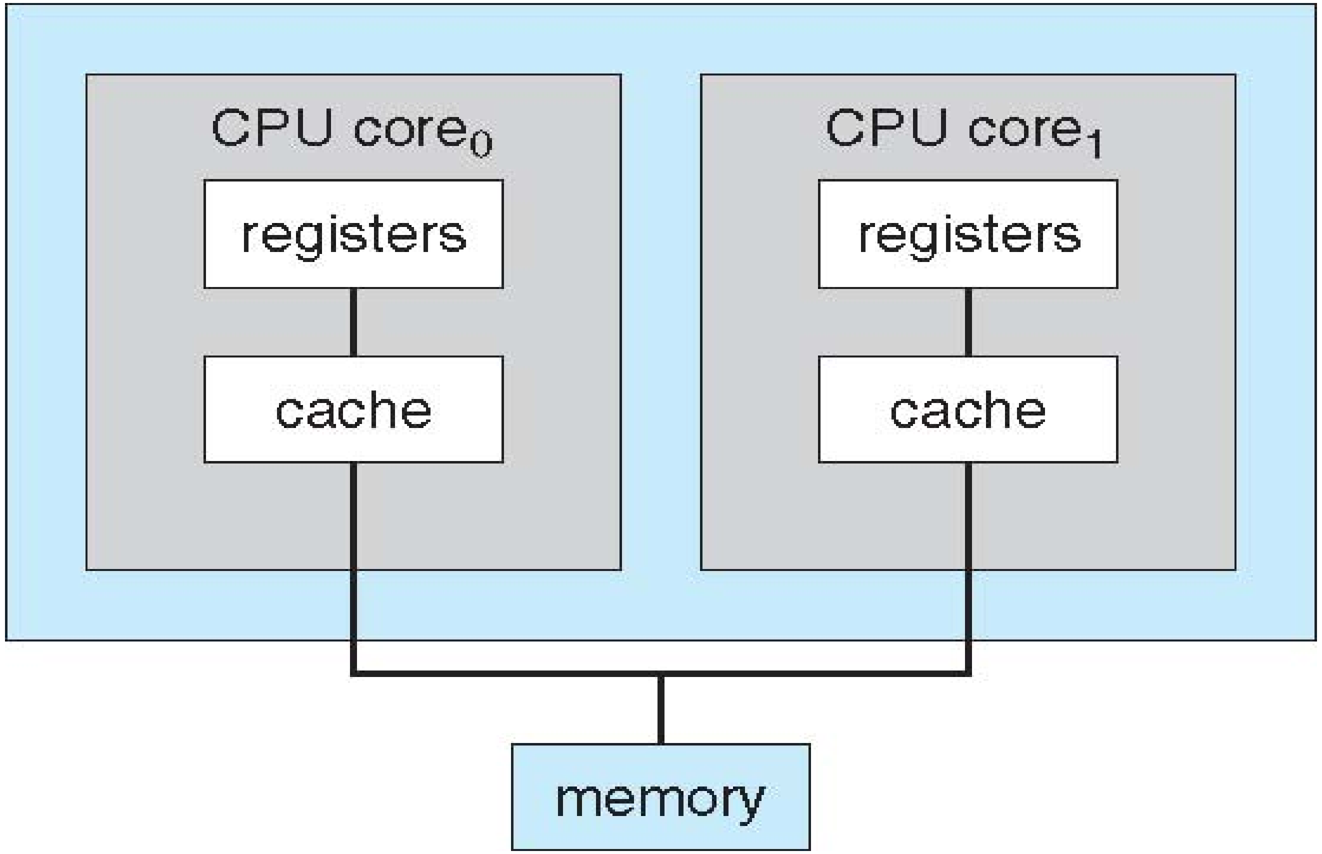
- **Multiprocessors memory access divisions:**
  - UMA (uniform memory access)
  - NUMA (non-uniform memory access)

Symmetric multiprocessing organization

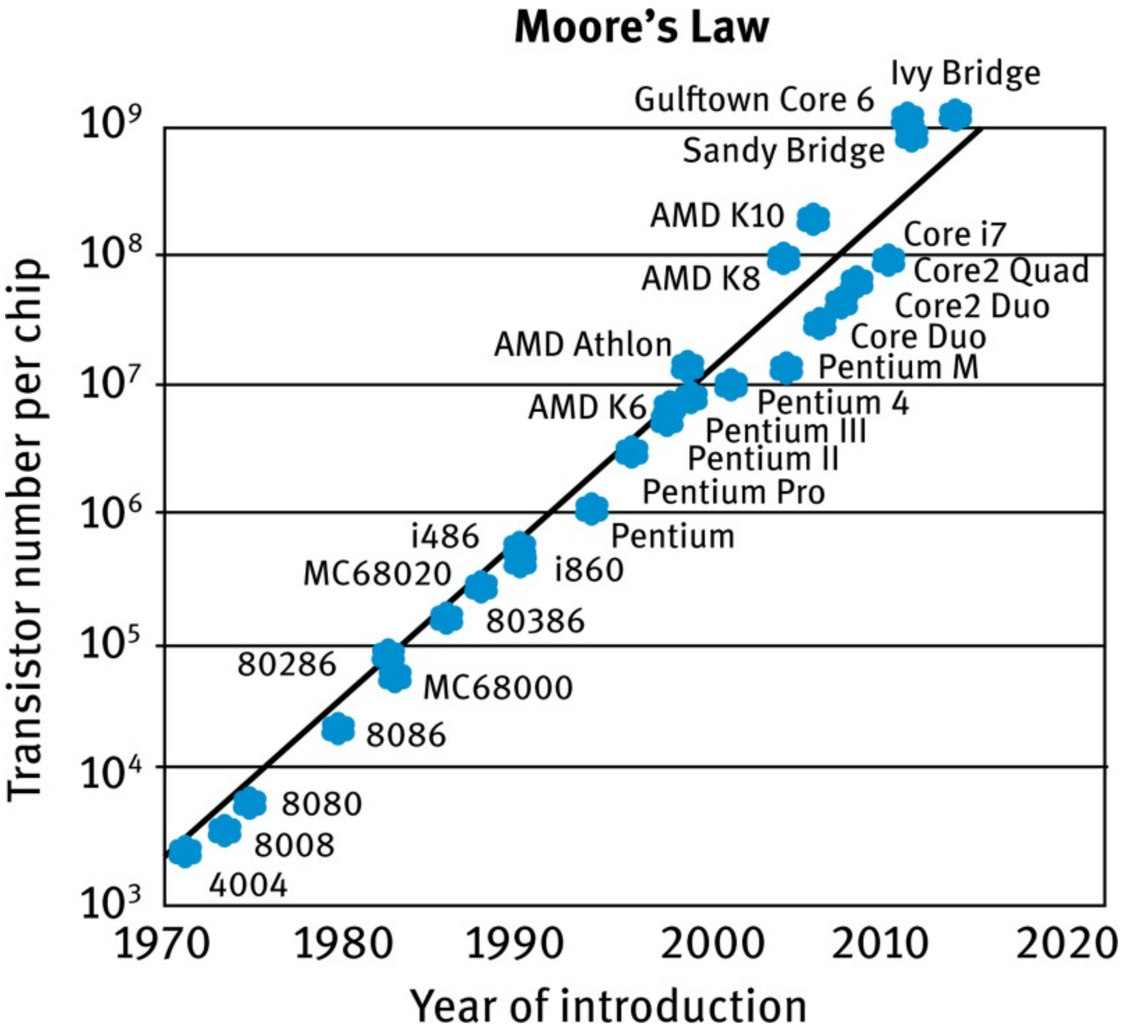


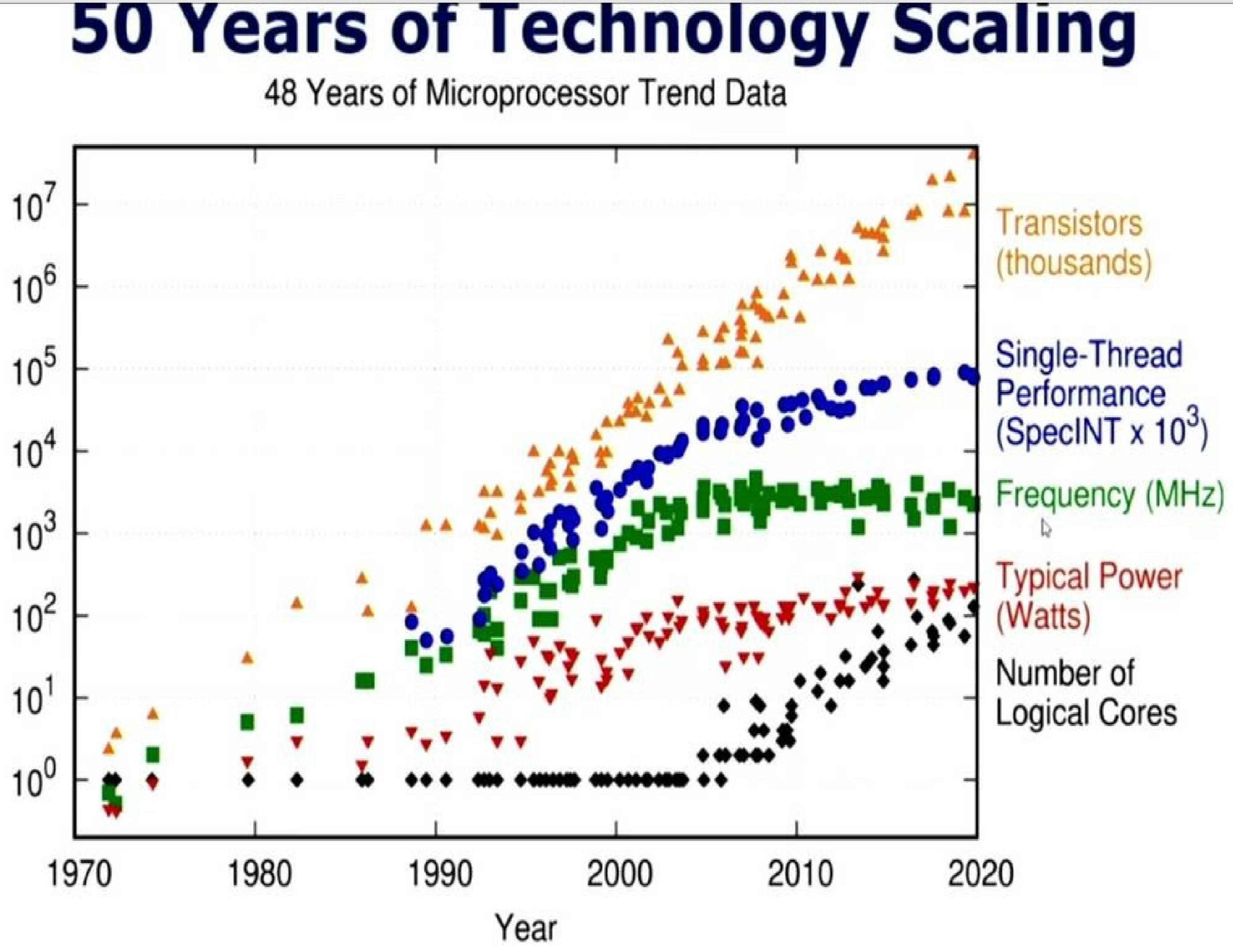
A dual-core design

- Multi-chip and multicore
- Systems containing all chips
  - Chassis containing multiple separate systems
- Advantage:
  - Faster communications
  - Less power consumption
- Disadvantage:
  - Performance gap of CPU utilization by software

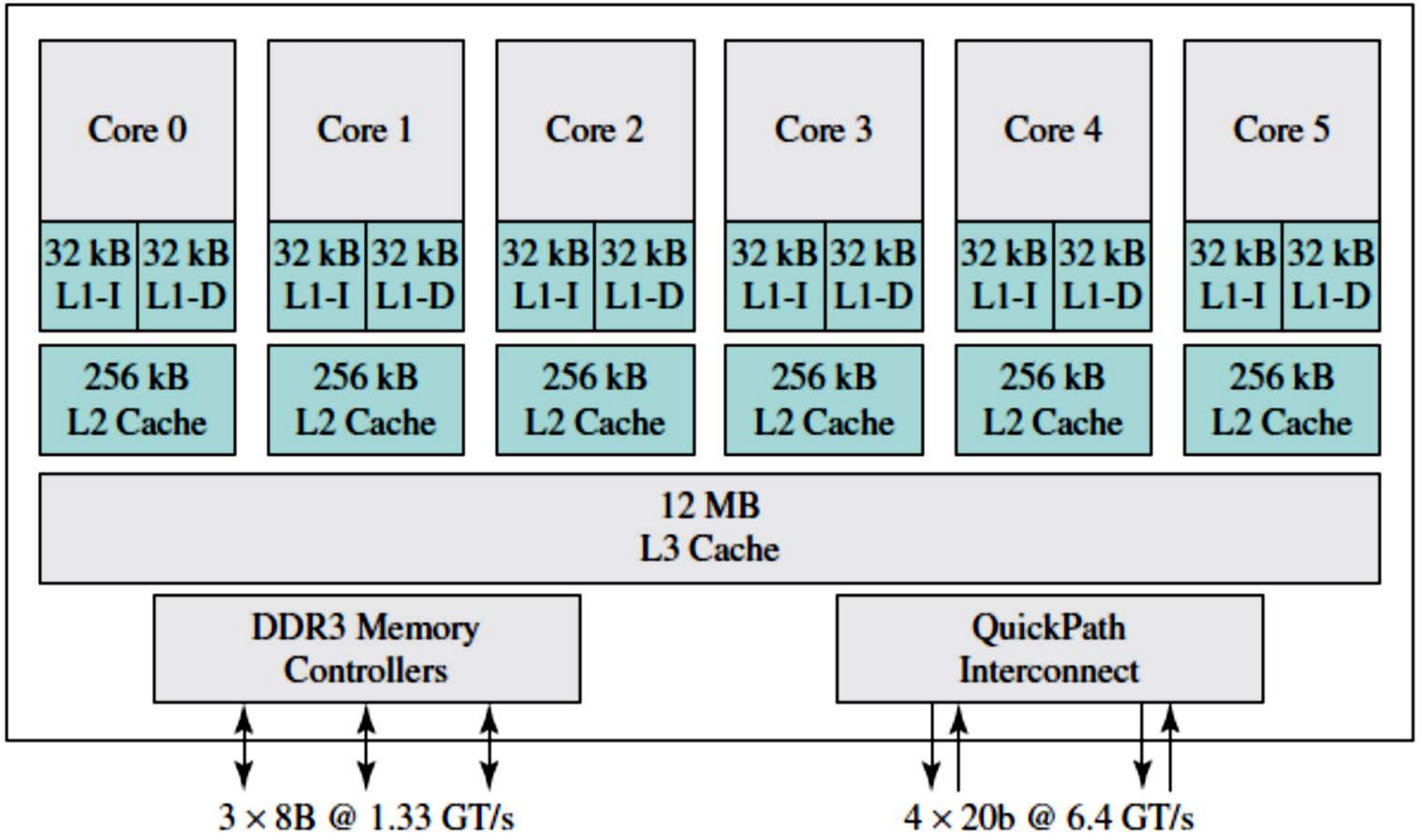


Moore’s law


Moore’s Law
Transistor number per chip
Year of introduction
10^3
10^4
10^5
10^6
10^7
10^8
10^9
1970
1980
1990
2000
2010
2020
4004
8008
8080
8086
80286
MC68000
80386
MC68020
i486
i860
Pentium
Pentium Pro
Pentium II
Pentium III
AMD K6
Pentium 4
AMD Athlon
Pentium M
Core Duo
Core2 Duo
Core2 Quad
Core i7
AMD K8
AMD K10
Sandy Bridge
Gulftown Core 6
Ivy Bridge


50 Years of Technology Scaling
48 Years of Microprocessor Trend Data
10^0
10^1
10^2
10^3
10^4
10^5
10^6
10^7
1970
1980
1990
2000
2010
2020
Year
Transistors (thousands)
Single-Thread Performance (SpecINT x 10^3)
Frequency (MHz)
Typical Power (Watts)
Number of Logical Cores

Intel core i7-990X block diagram



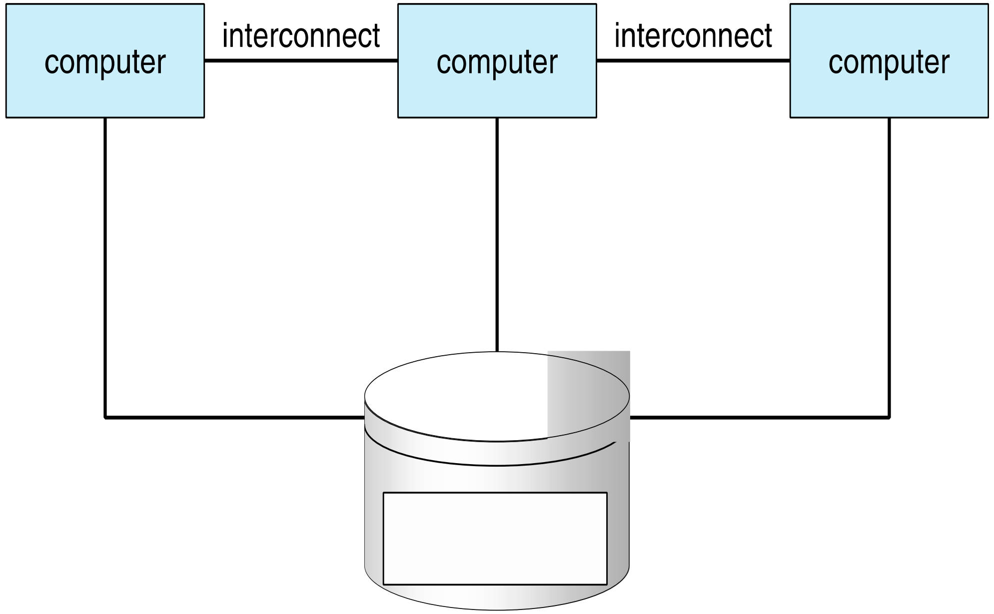
Blade servers

- **Multiple** processor boards, I/O boards, and networking boards are placed in the same chassis.

- Each blade-processor
  - boots independently
  - runs its own OS!

Clustered systems

- Like multiprocessor systems, but multiple systems working together (loosely-coupled systems)
- 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



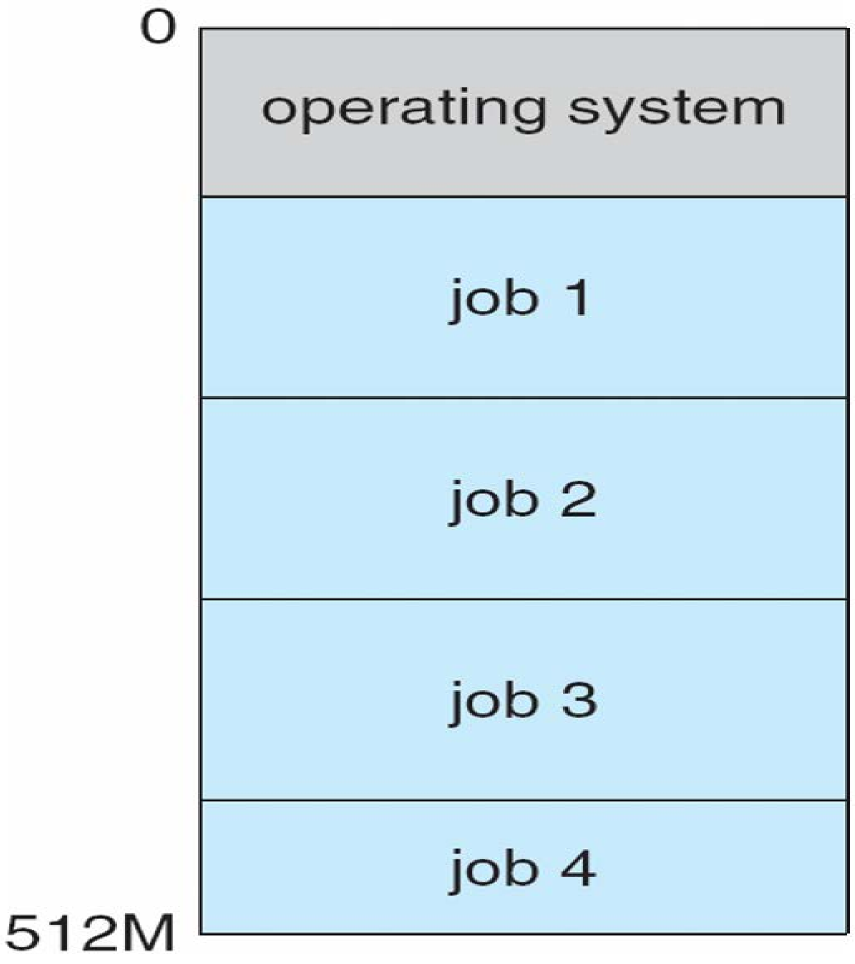
Operating System Structure

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

- **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

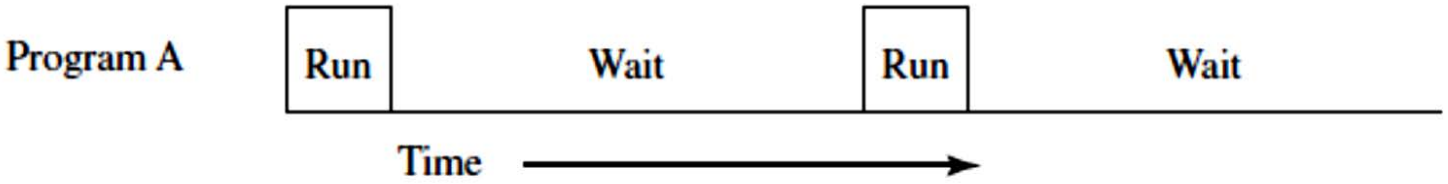


- **Job pool** on a disk!
  - Physical memory
  - Virtual memory

System utilization example

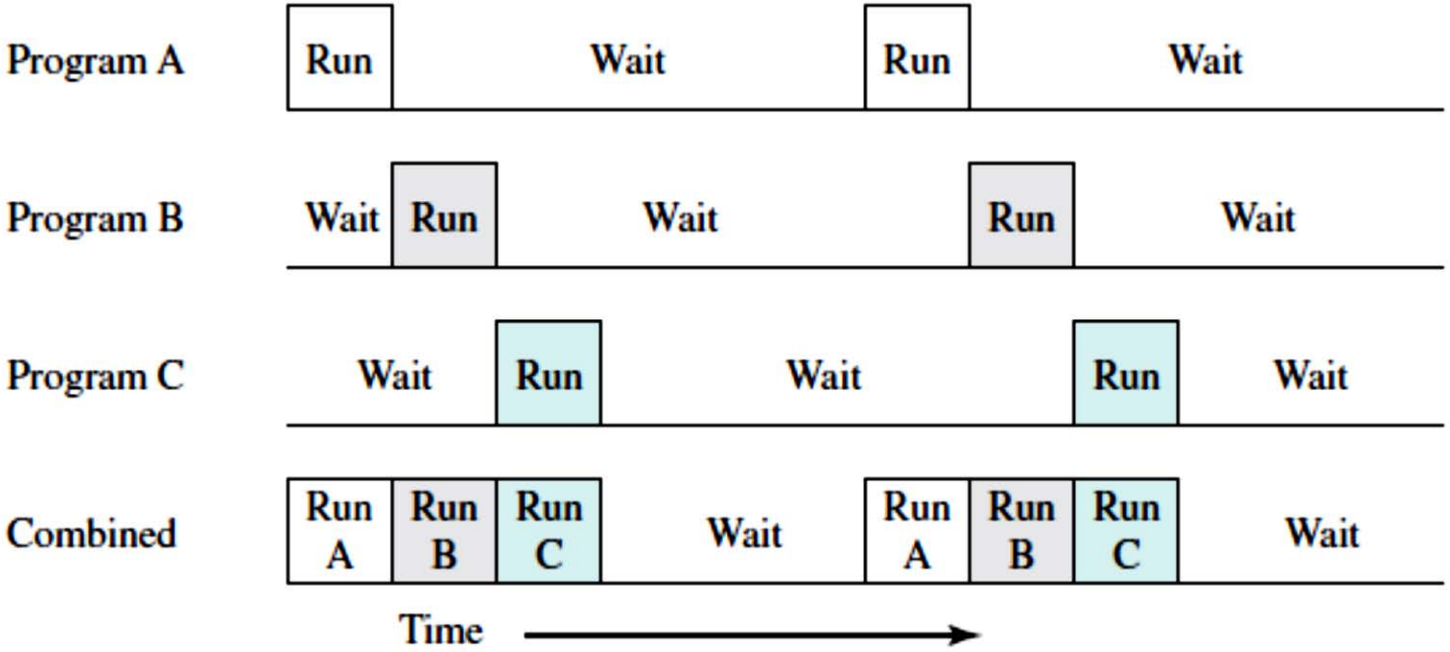
Read one record from file	15 μs
Execute 100 instructions	1 μs
Write one record to file	15 μs
Total	31 μs

$$\text{Percent CPU utilization} = \frac{1}{31} = 0.032 = 3.2\%$$



(a) Uniprogramming

Increasing system utilization example



(c) Multiprogramming with three programs

- OS idle state is done by HLT assembly instruction (x86 opcode 0xF4)

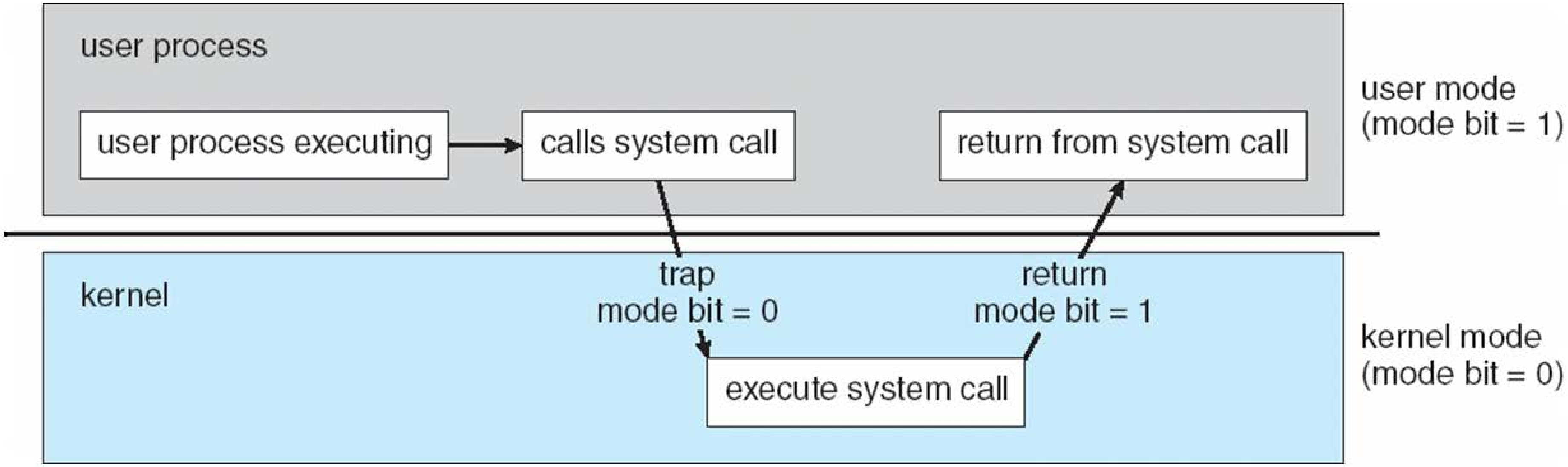
Operating-system operations

- **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 processors



Time management

- Timer to prevent infinite loop / process hogging resources (Watchdog timer)
  - 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

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
- 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 user, some operating system running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads

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

- 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 de-allocating 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 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

- 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)

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 and security

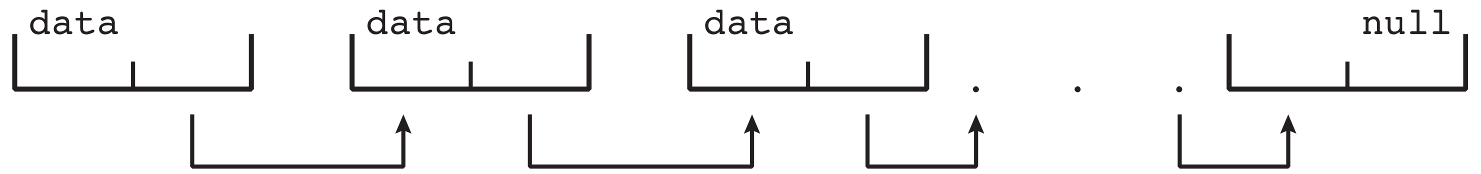
- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS

- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service

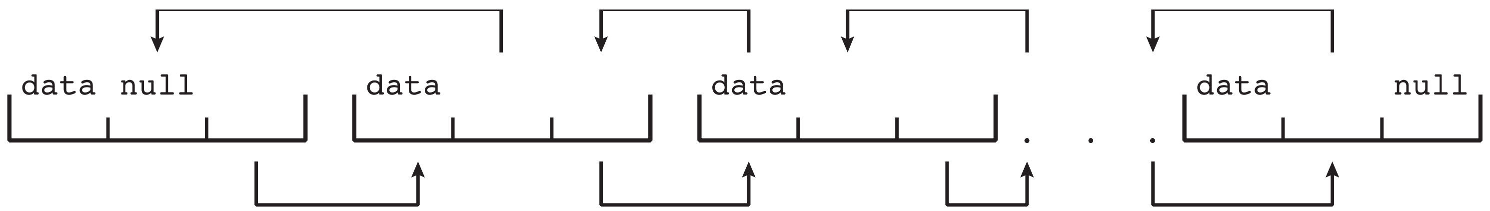
- Systems generally first distinguish among users, to determine who can do what
  - User identities (user IDs, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (group ID) 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

Kernel data structures

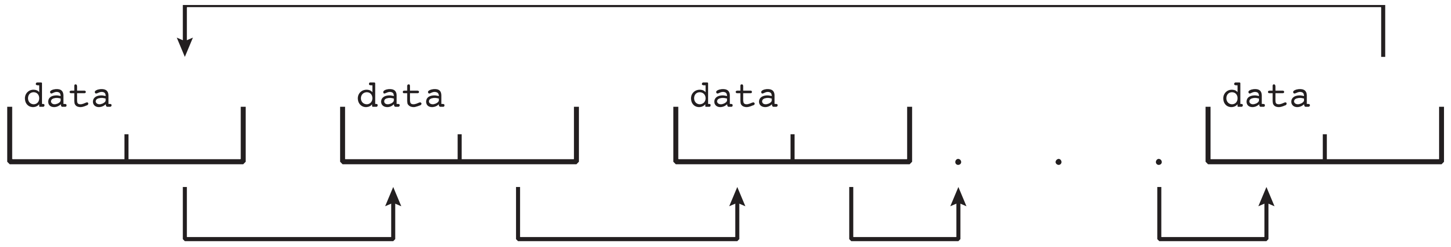
- Many similar to standard programming data structures
- *Singly linked list*



- *Doubly linked list*

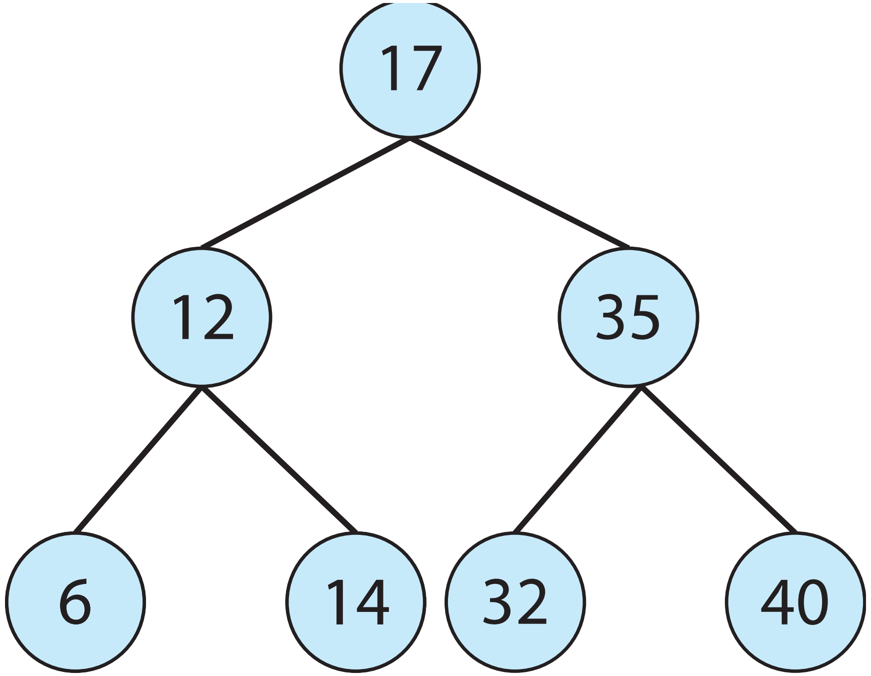


- *Circular linked list*



Kernel data structures

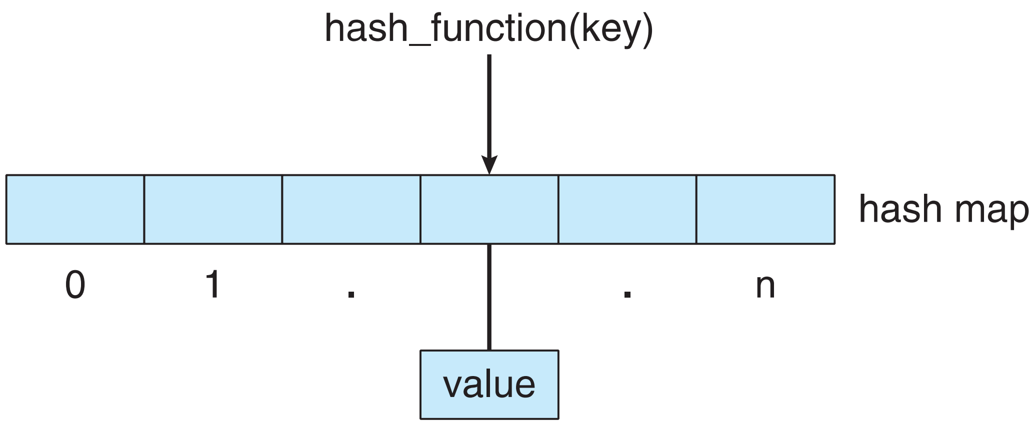
- Binary search tree
  left <= right
  - Search performance is *O(n)*
  - Balanced binary search tree is *O(lg n)*
- Example
  - Linux CPU-scheduling algorithm



Kernel data structures

- Hash function can create a hash map
  - Collision problem

Example: username, password



- Bitmap – string of *n* binary digits representing the status of *n* items (0011010111001010)
  - Example: Disk block availability
- Linux data structures defined in

  include files `<linux/list.h>, <linux/kfifo.h>, <linux/rbtree.h>`

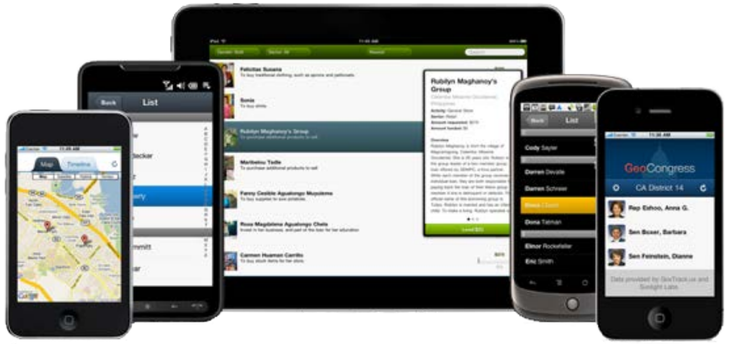
Computing Environment

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 (IoT)– even home systems use firewalls to protect home computers from Internet attacks

Computing environments (Mobile)

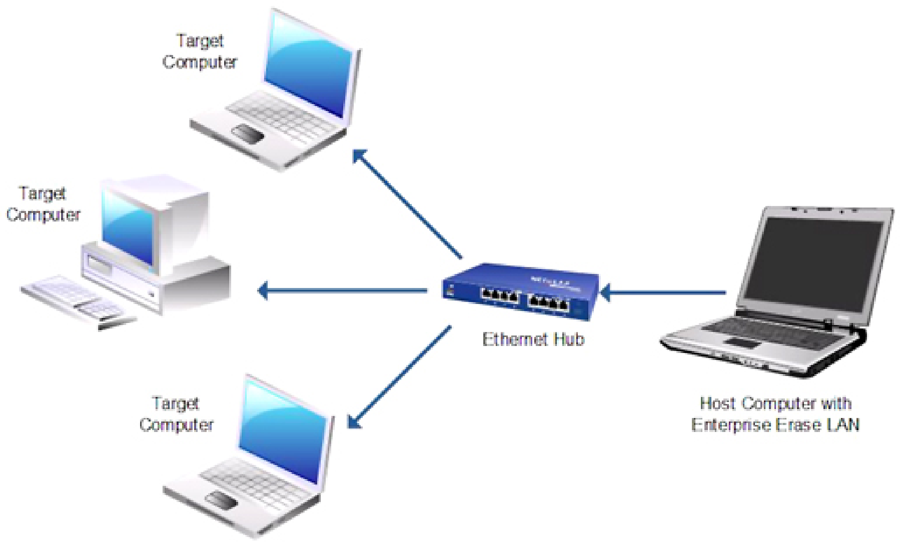
- Handheld smartphones, tablets, etc
- What is the functional difference between them and a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of apps like *augmented reality*
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are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



Computing environments (distributed)

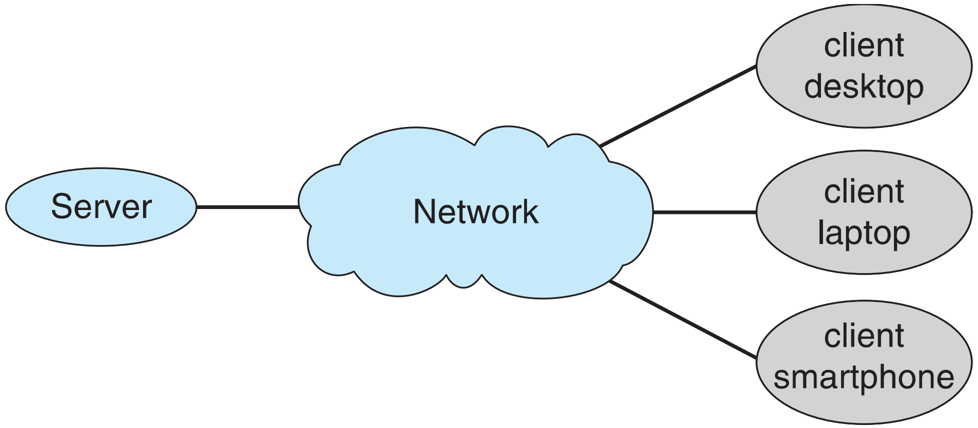
➢ Distributed computing

- 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



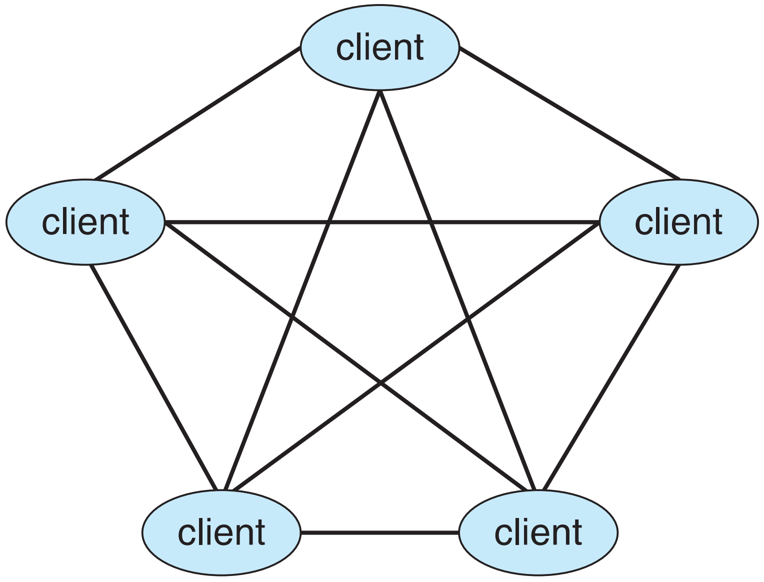
Computing environments (Client-Server)

- **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
  - No bottleneck server!
  - 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



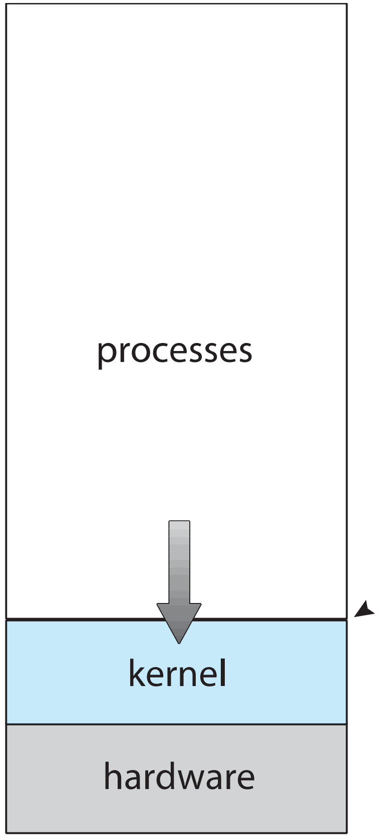
Computing environments (Virtualization)

- Allows operating systems to run applications within other OSes
  - 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 OSes 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)

- **Virtualization vs. Emulation (both are SW)**
  - OS to OS vs. CPU to CPU
- **Use cases involve laptops and desktops running multiple OSes 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)



(a)

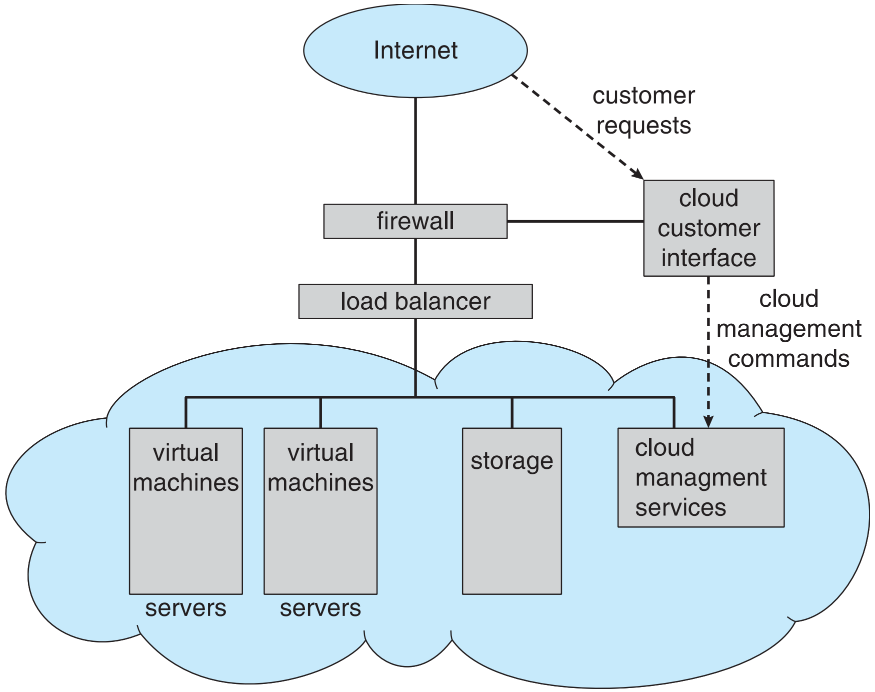
achine

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)

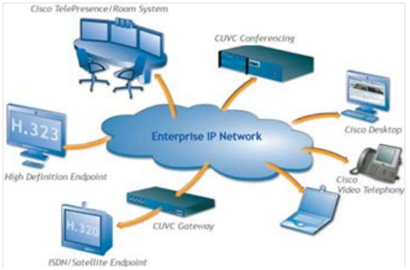
- Cloud computing environments composed of traditional OSes, plus VMMs, plus cloud management tools
  - Internet connectivity requires security like firewalls
  - Load balancers spread traffic across multiple applications



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 (soft, firm, hard)
  - 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
- All book materials are available at: www.os-book.com

Questions?