Operating System

Organizational Information

- Instructor
  - Tianhai Zhao (赵天海)
- Contact info
  - zhaoth@nwpu.edu.cn
- Office
  - 301,Yongzi building, YouYi campus
- Online discussion group
  - Use your student ID and name

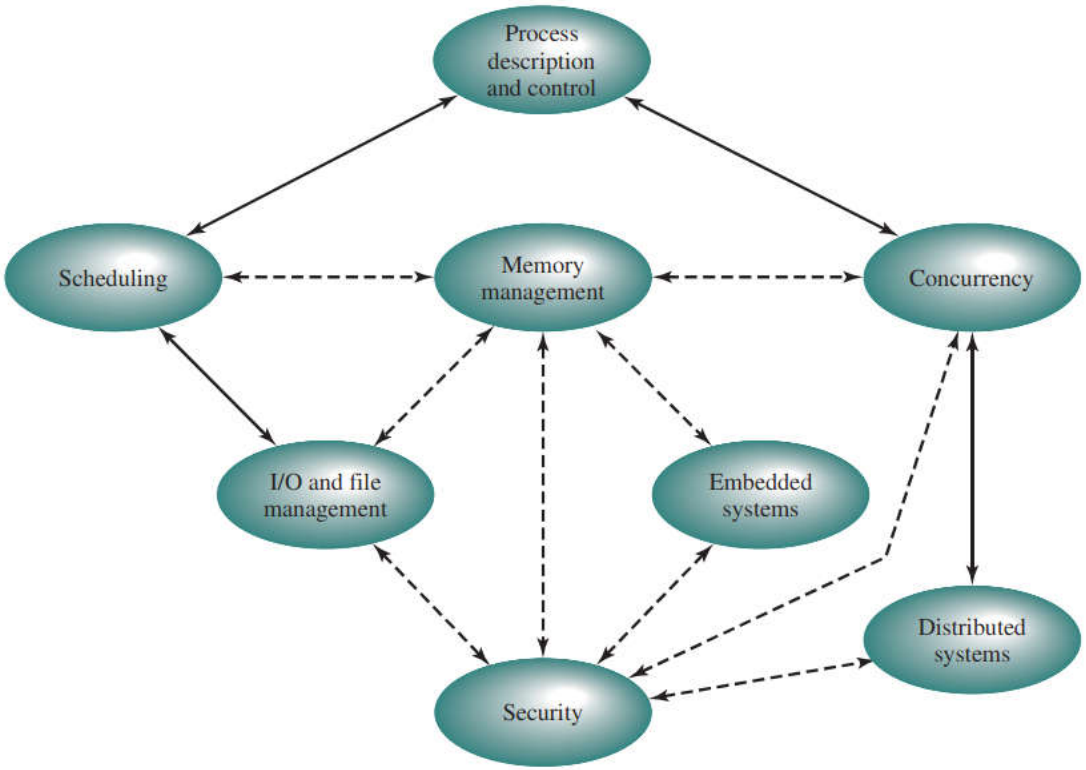
群名称：Operating System 2021
群 号：713426437

Basic information of this course

- Course ID: U10M12007.01
- Class room: East building JB309
  - Mon. and Wed.
- Credit: 3.5
- Class Hours: 48
- Weeks: 1-14

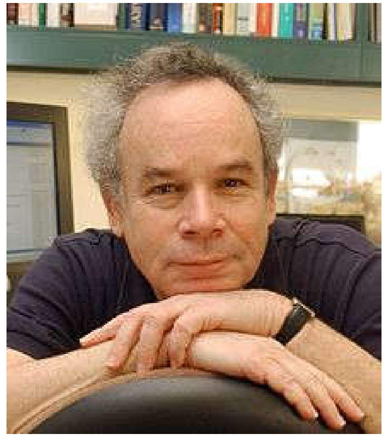
Main Topic

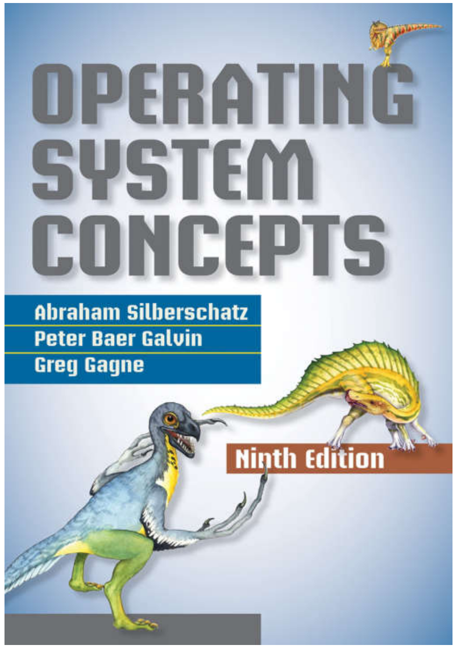
- Overview
- Process management
- Memory Management
- Storage management
- Protection and security



TextBook

- Operating System Concepts(Ninth Edition)
  - Avi Silberschatz
  - Peter Baer Galvin
  - Greg Gagne
- John Wiley & Sons, Inc.
- ISBN 978-1-118-06333-0





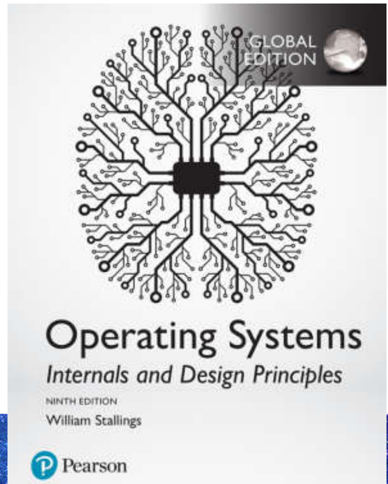
Avi Silberschatz is the Sidney J. Weinberg Professor of Computer Science at Yale University.

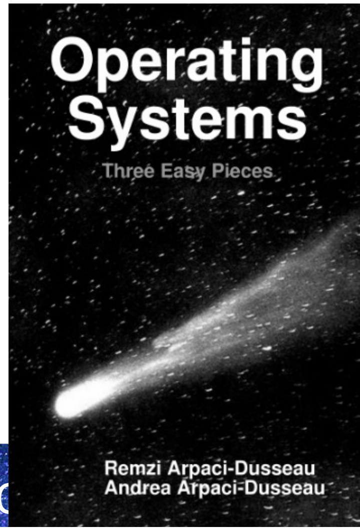
On-line course

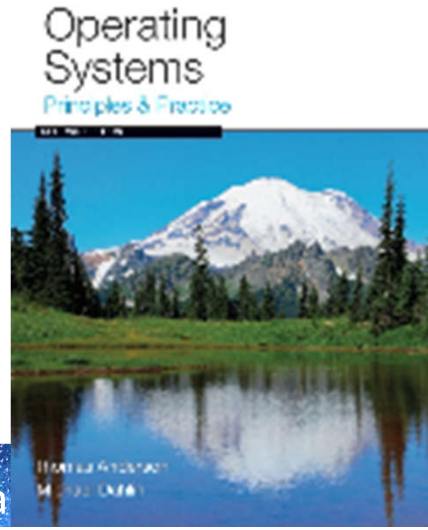
- CS 162: Operating Systems and System Programming
  - https://cs162.eecs.berkeley.edu/
- Online video
  - From bilibili or youtube
  - https://www.bilibili.com/video/BV1e7411B7Ja?from=search&seid=3572888976071900748

Refence book

- **Operating Systems: Internal and Design Principles**
  - William Stallings
- **Operating Systems: Three Easy Pieces**
  - Remzi H. Arpaci-Dusseau
  - Andrea C. Arpaci-Dusseau (University of Wisconsin-Madison)
- **Operating Systems: Principles and Practice**
  - Anderson and Dahlin







Prerequisite course

- Principles of Computer Organization
- Program Design & Development
- Data Structure
- Compliers

Grading

- 60% *final exam*
  – In-class exams
- 15% projects
- 15% homework
- 10% participation

Why take this course?

- Some of you will actually design and build operating systems or components of them.
  - Perhaps more now than ever
- Many of you will create systems that utilize the core concepts in operating systems.
  - Whether you build software or hardware
  - The concepts and design patterns appear at many levels
- All of you will build applications, etc. that utilize operating systems
  - The better you understand their design and implementation, the better use you'll make of them.

How to learn this course

“学而时习之 不亦说乎。 “

To learn something and timely practice it – it is a pleasure

--孔子《论语》

"不闻不若闻之，闻之不若见之，见之不若知之，知之不若行之；学至于行之而止矣。 “

Not hearing is not as good as hearing, hearing is not as good as seeing, seeing is not as good as knowing, knowing is not as good as acting; true learning continues until it is put into action.

--荀子《儒效篇》

From Tsinghua OS course

Chapter 1: Introduction

Chapter 1: Introduction

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture
- Process Management
- Memory Management
- Storage Management
- Computing Environments

Goals for Today

- What is an Operating System?
  - And – what is it not?

Interactive is important!

Ask Questions!

Objectives

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

Introductions to Operating Systems

What is an Operating System?

Application Software and Operating System

- runs or executes as per user request.
- java, c, c++ etc are used to develop
- Without an operating system application software can not be installed

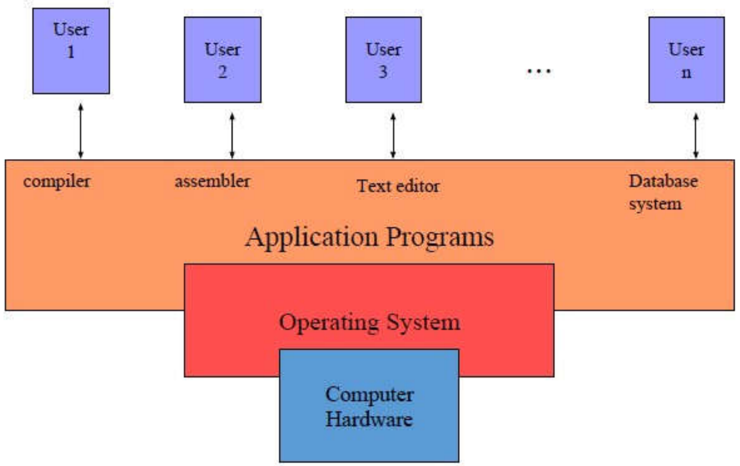
- maintain the system resources and provide environment for application software to run.
- Low level languages are used to write the system software.
- Without system software, system can't run.

What is an Operating System?(cont.)

- 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 system convenient to use
  – Use the computer hardware in an efficient manner

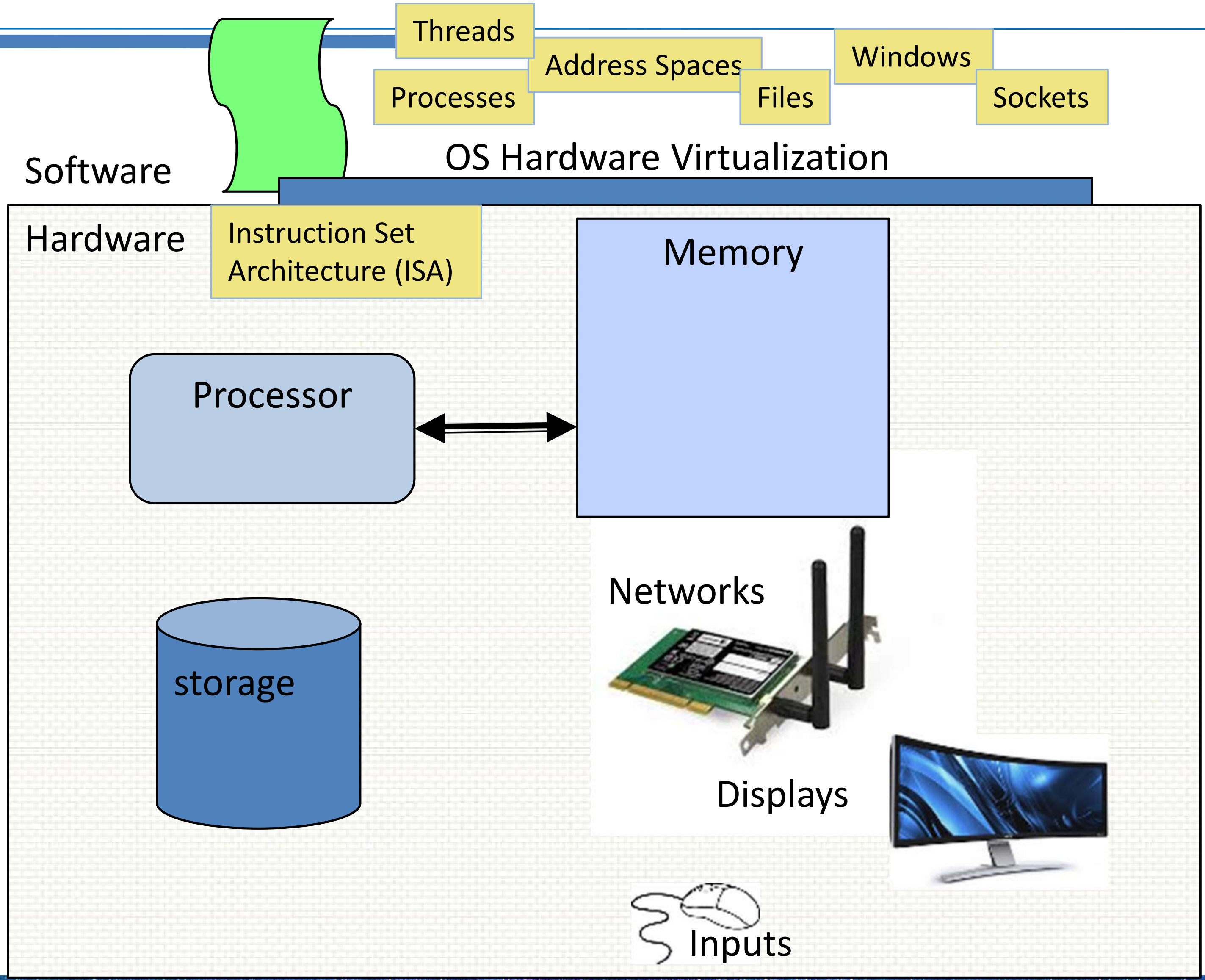
Computer System Structure

- 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

OS Basics: "Virtual Machine" Boundary



What Operating Systems Do

- Depends on the point of view
- Users want convenience, **ease of use**
  - Don't care about **resource utilization**
- But shared computer such as 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(System view)

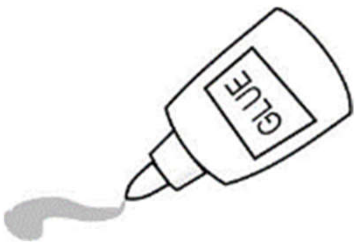
- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use

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

An Operating System?

- Referee
  - Manage sharing of resources, Protection, Isolation
    - Resource allocation, isolation, communication

- Illusionist
  - Provide clean, easy to use abstractions of physical resources
    - Infinite memory, dedicated machine
    - Higher level objects: files, users, messages
    - Masking limitations, virtualization



- Glue
  - Common services
    - Storage, Window system, Networking
    - Sharing, Authorization

Operating System Definition (Cont.)

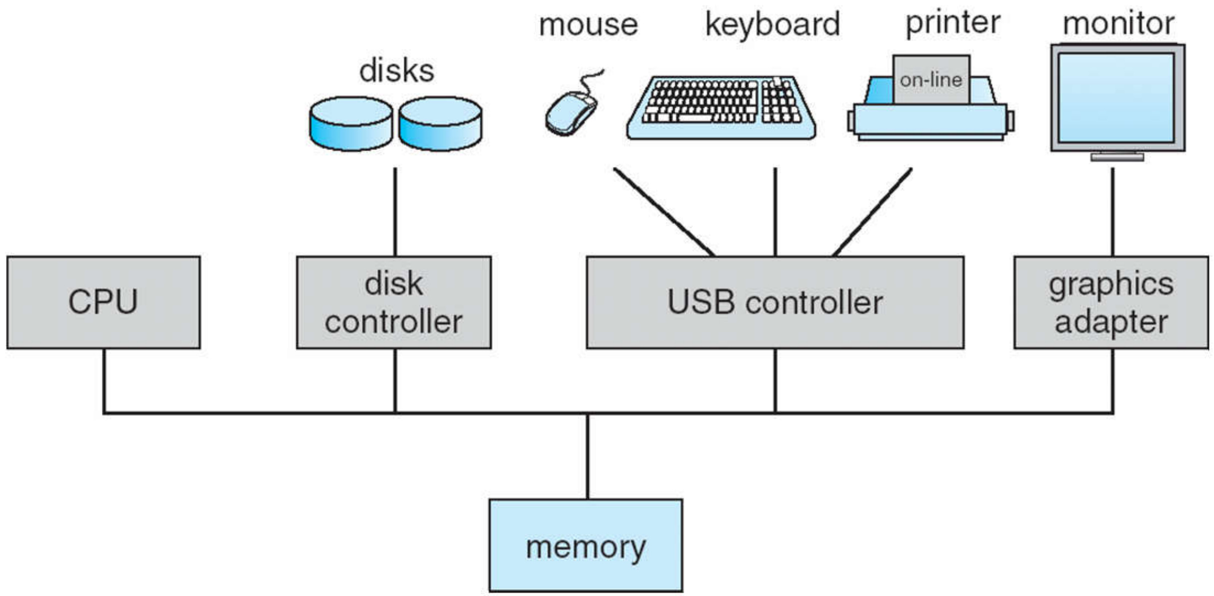
- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is good approximation
  - But varies wildly
- “The one program running at all times on the computer” is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program.

An operating system (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs.

Computer System Organization

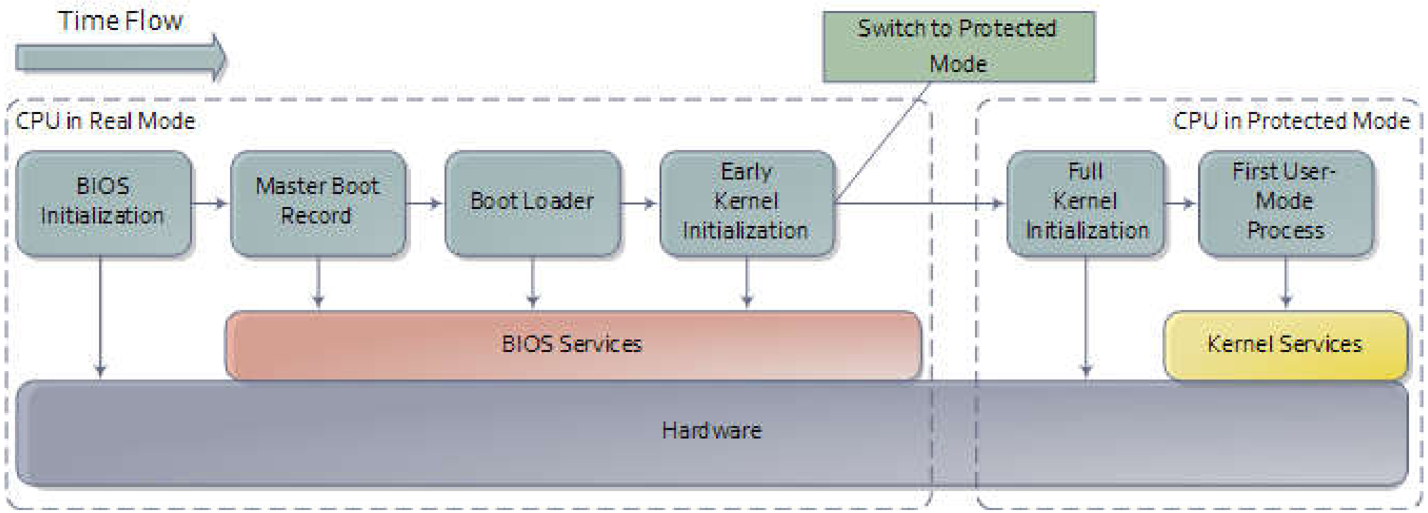
Computer System Organization

- 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

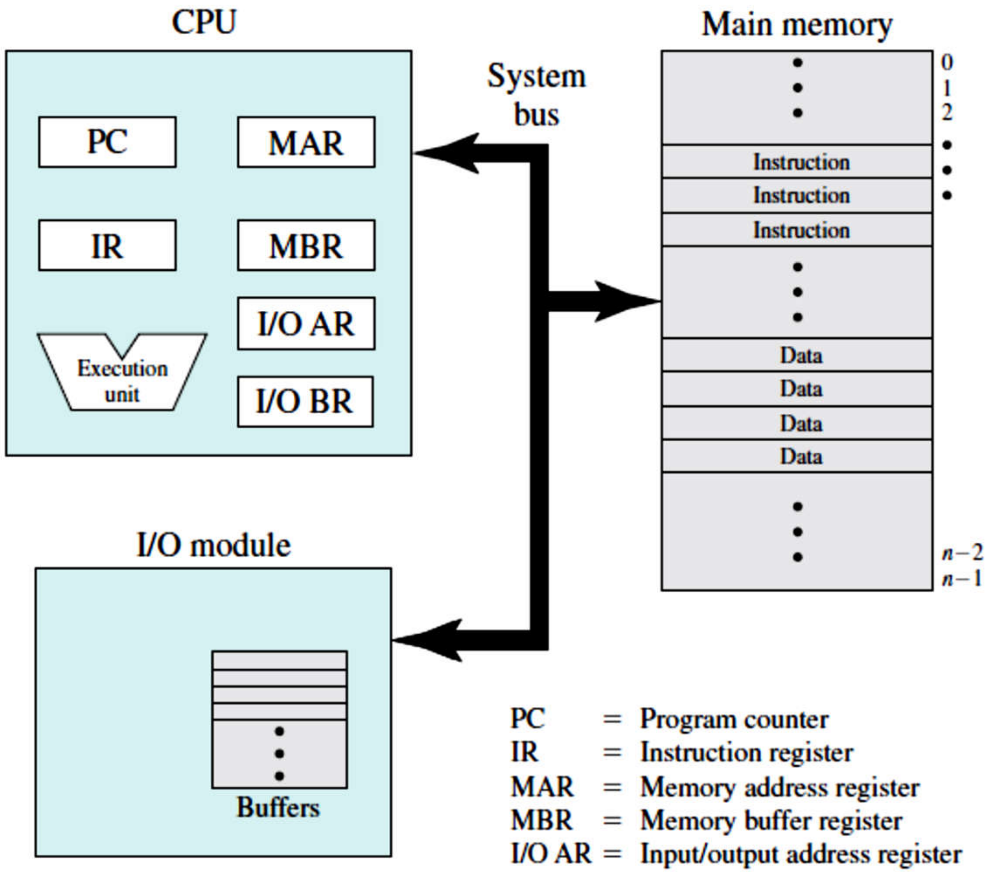


Computer Startup

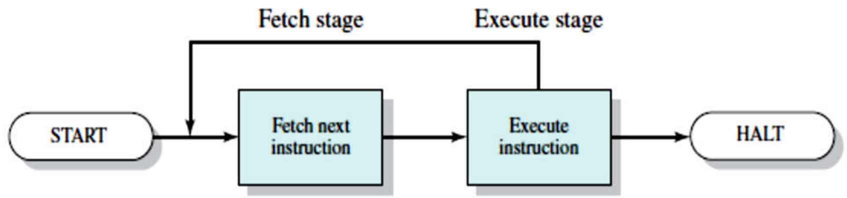
- **bootstrap program** is loaded at power-up or reboot
  – Typically stored in ROM or EPROM, generally known as **firmware**
  – Initializes all aspects of system
  – Loads operating system kernel and starts execution



Computer system, 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

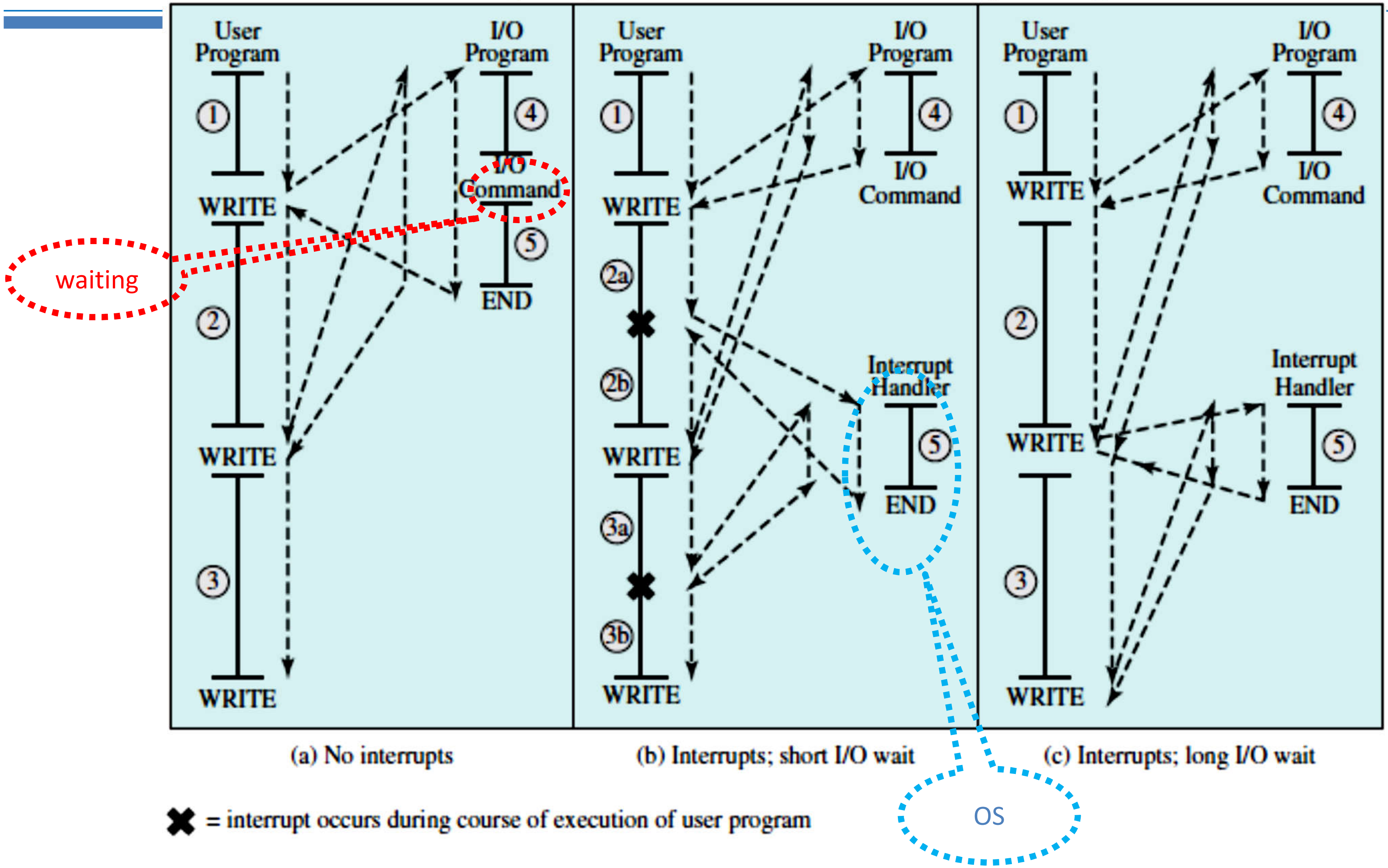
Interrupts

Interrupts are provided primarily as a way to improve processor utilization.

Table 1.1 Classes of 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, or 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)

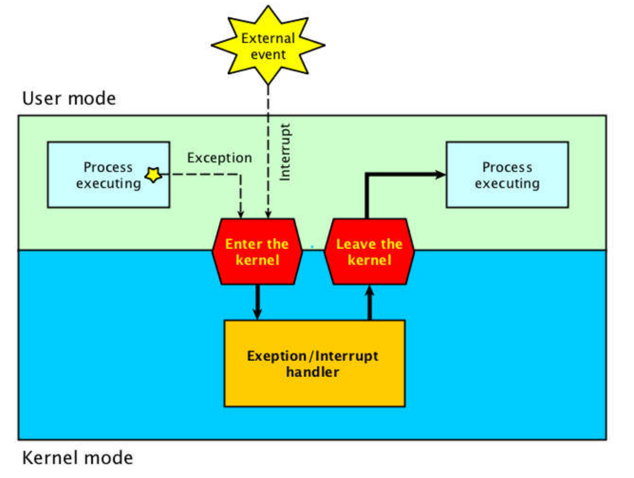


(a) No interrupts
(b) Interrupts; short I/O wait
(c) Interrupts; long I/O wait

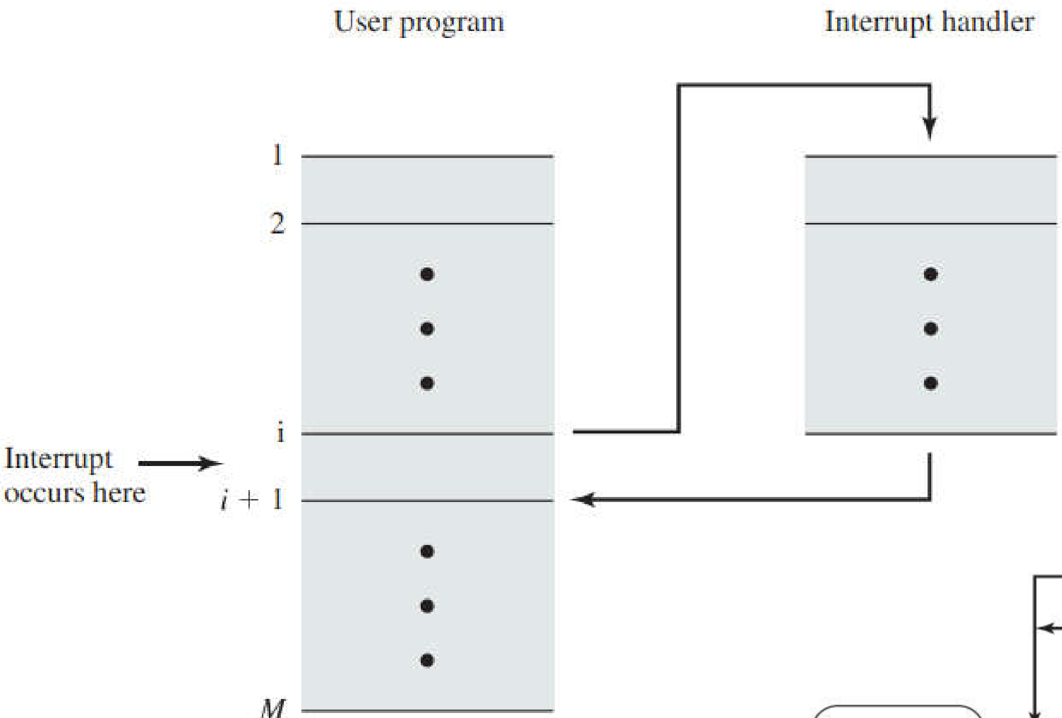
✖ = interrupt occurs during course of execution of user program

Common Functions of Interrupts

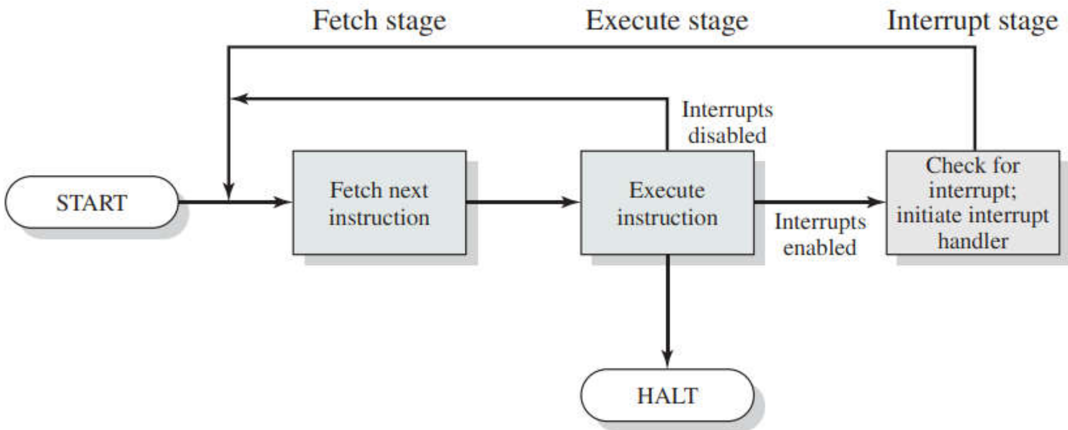
- Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines
- an interrupt handler, also known as an interrupt service routine or ISR, is a special block of code associated with a specific interrupt condition.
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request
- Interrupt architecture must save the address of the interrupted instruction
- An operating system is **interrupt driven**



Interrupt Handling

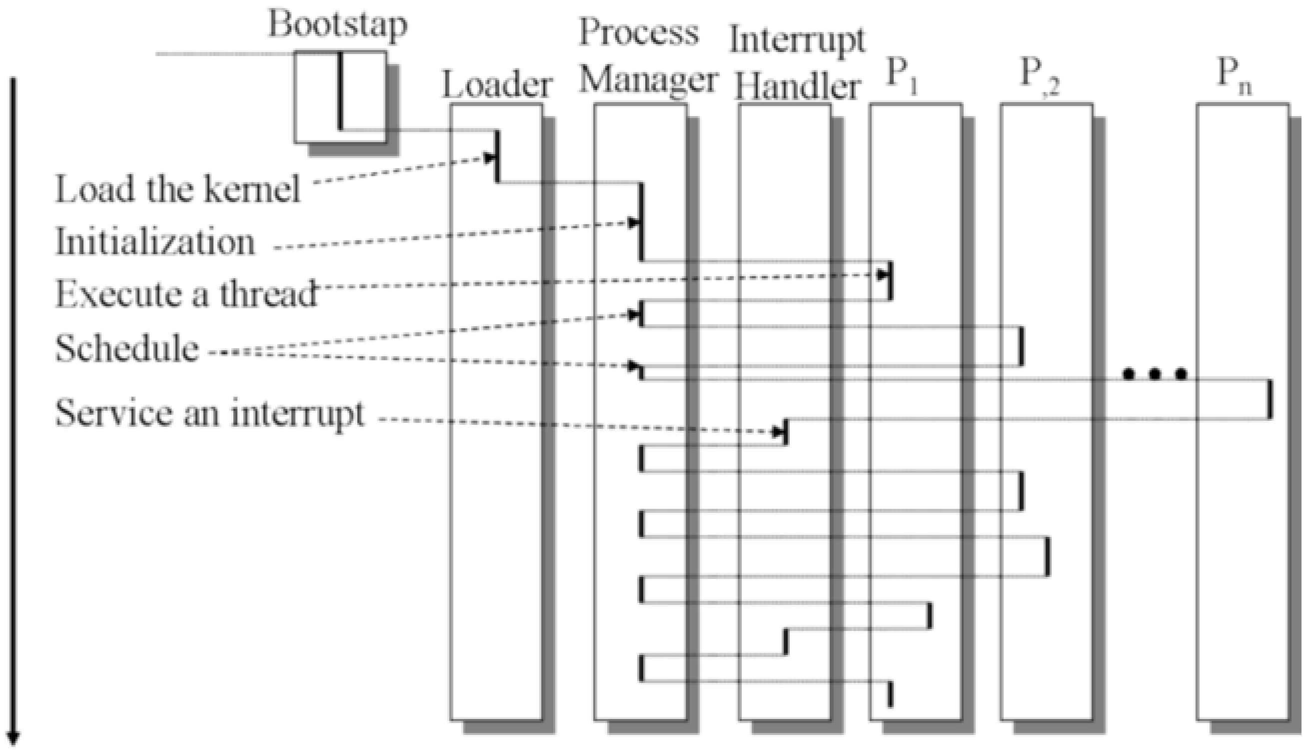


Transfer of Control via Interrupts



Instruction Cycle with Interrupts

Interrupt Timeline



As the computer runs, processing switches between user processes and the operating system as hardware and software interrupts are received.

Storage Structure

- Main memory – only large storage media that the CPU can access directly
  - **Random access**
  - Typically **volatile**

- Secondary storage – extension of main memory that provides large **nonvolatile** storage capacity
- Magnetic disks – rigid metal or glass platters covered with magnetic recording material

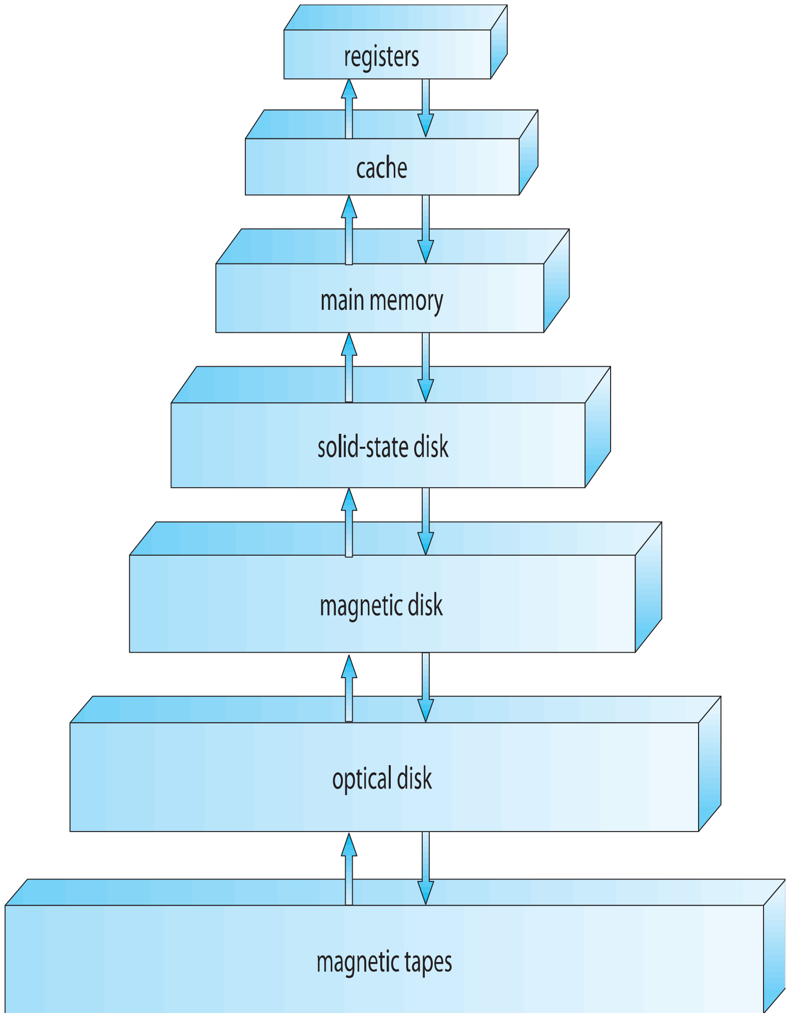
  - Disk surface is logically divided into **tracks**, which are subdivided into **sectors**
  - The **disk controller** determines the logical interaction between the device and the computer
- **Solid-state disks** – faster than magnetic disks, nonvolatile
  - Various technologies
  - Becoming more popular

Storage Hierarchy

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

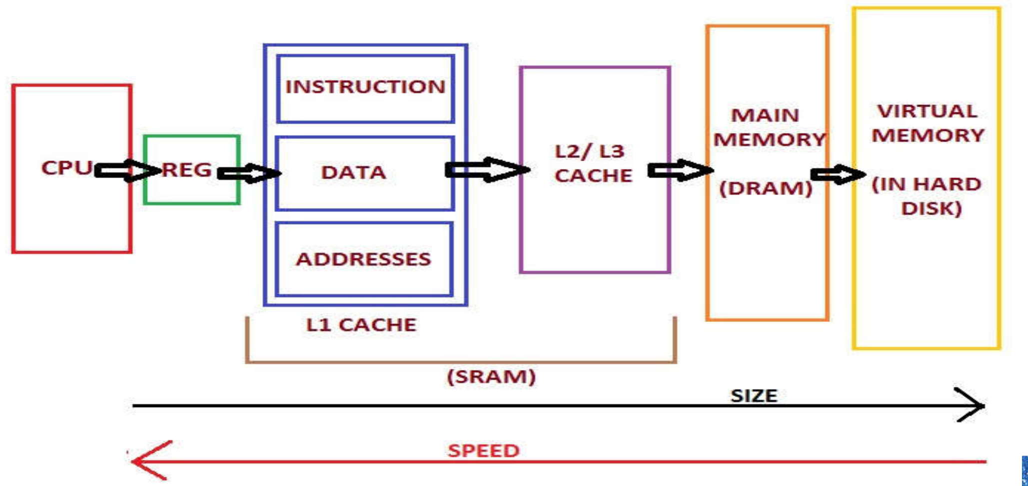
- **Caching** – copying information into faster storage system; main memory can be viewed as a cache for secondary storage

- **Device Driver** for each device controller to manage I/O
  - Provides uniform interface between controller and kernel



Caching

- Principle of locality is the phenomenon where the same value or related storage locations are frequently accessed by the processor.
- 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

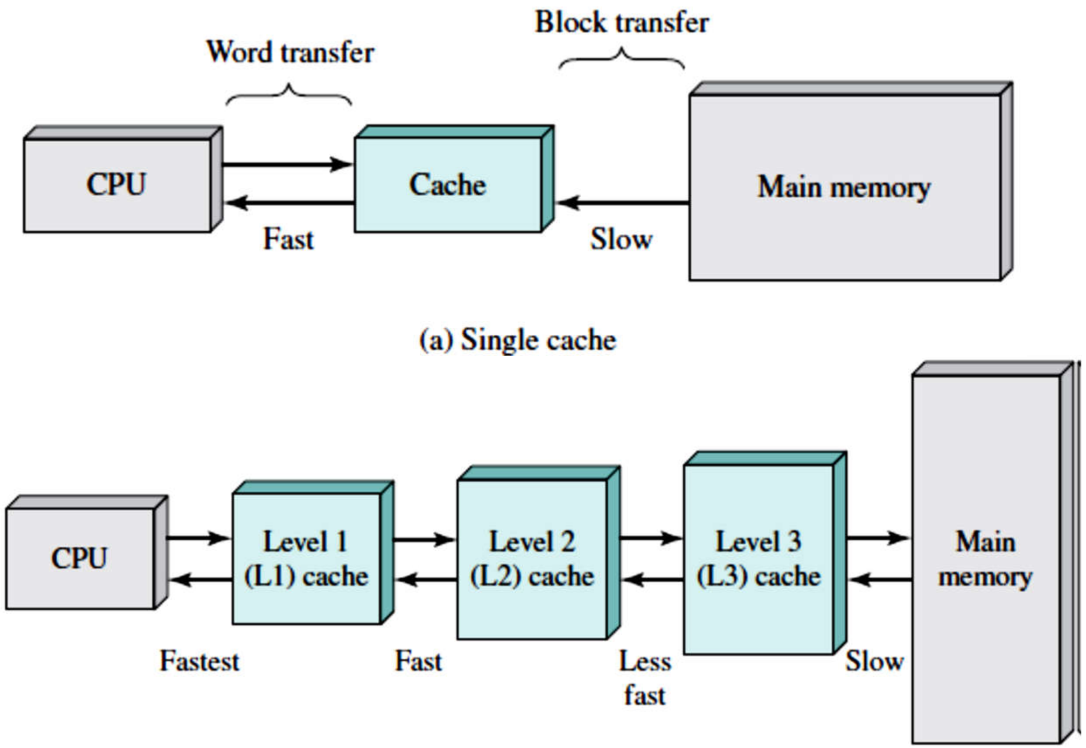


Performance of Various Levels of Storage

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

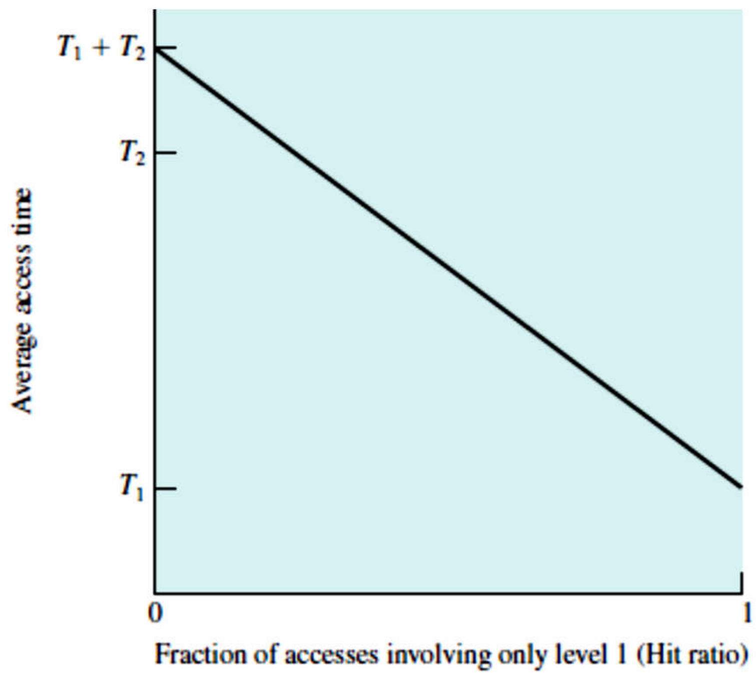
- Movement between levels of storage hierarchy can be explicit or implicit

Cache memory



(a) Single cache

(b) Three-level cache organization



suppose the processor has access to two levels of memory. Level 1 has an access time of 0.1 *us*, level 2 has an access time of 1 *us*. suppose 95% of the memory accesses are found in the cache (*H* = 0.95). Then, the average time to access a byte can be expressed as

$$(0.95)(0.1\ \mu s) + (0.05)(0.1\ \mu s + 1\ \mu s) = 0.095 + 0.055 = 0.15\ \mu s$$

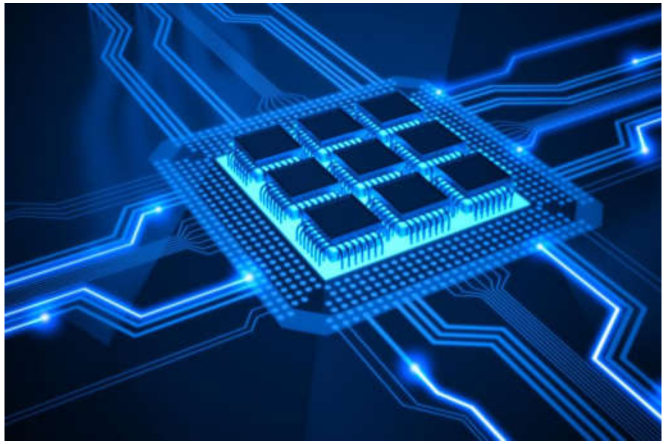
I/O Structure

- 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

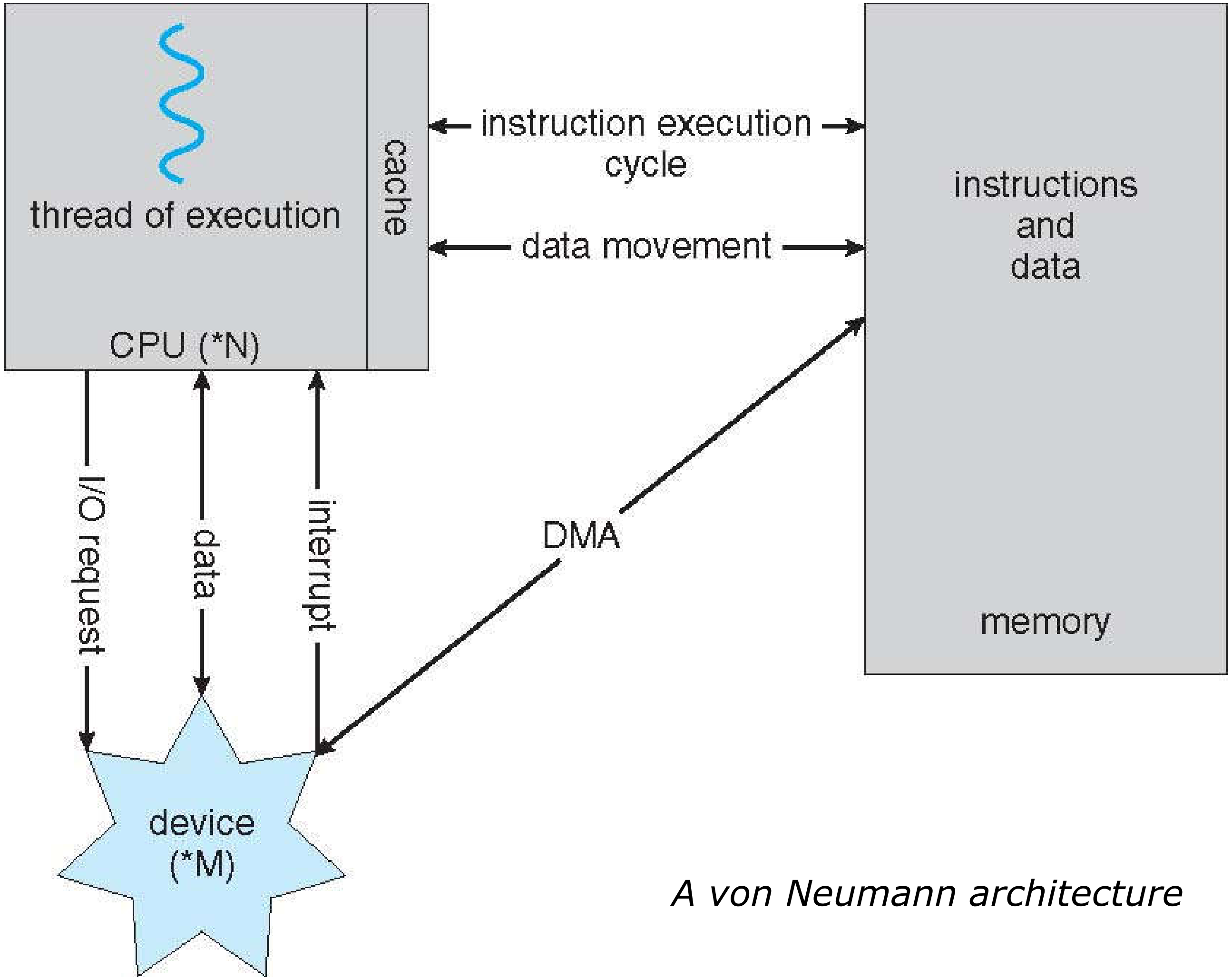
Direct Memory Access Structure

- 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 the one interrupt per byte

Computer System Architecture



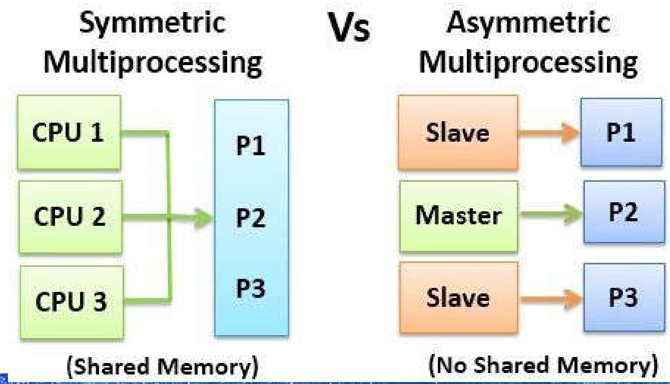
How a Modern Computer Works



A von Neumann architecture

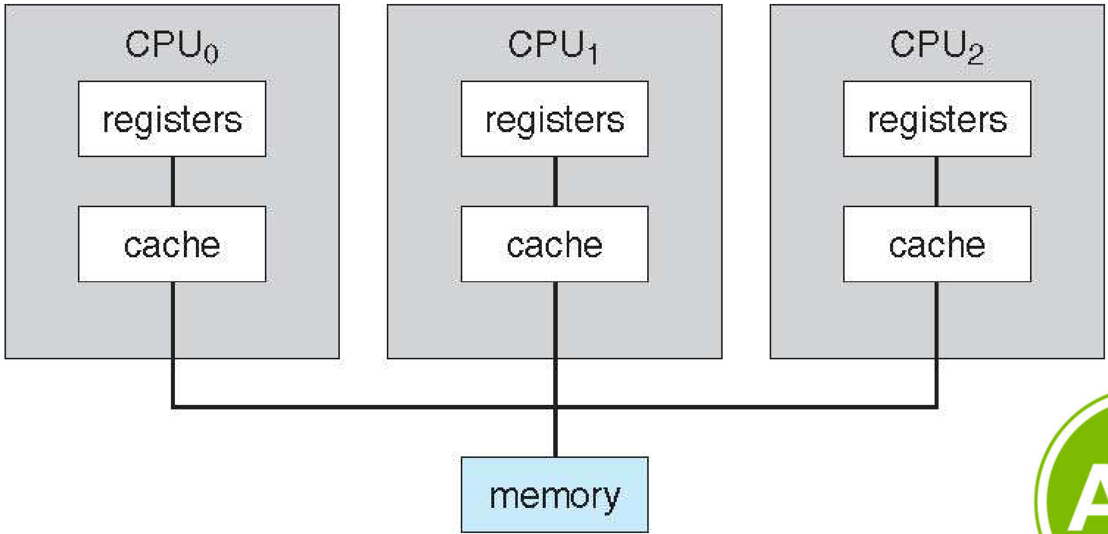
Computer-System Architecture

- single general-purpose processor
  - Most systems have special-purpose processors as well
- **Multiprocessors** systems growing in use and importance
  - Also known as **parallel systems**, **multicore systems**
  - Advantages include:
    1. **Increased throughput**
    2. **Economy of scale** – multiprocessors vs. multiple single processor
    3. **Increased reliability –** graceful degradation vs fault tolerance
  - Two types:
    1. **Asymmetric Multiprocessing**
    2. **Symmetric Multiprocessing**



Symmetric Multiprocessing Architecture

- **multiprocessor systems** have begun to dominate the landscape of computing
- each processor has its own set of registers, as well as a private—or local—cache.
- all processors share physical memory.

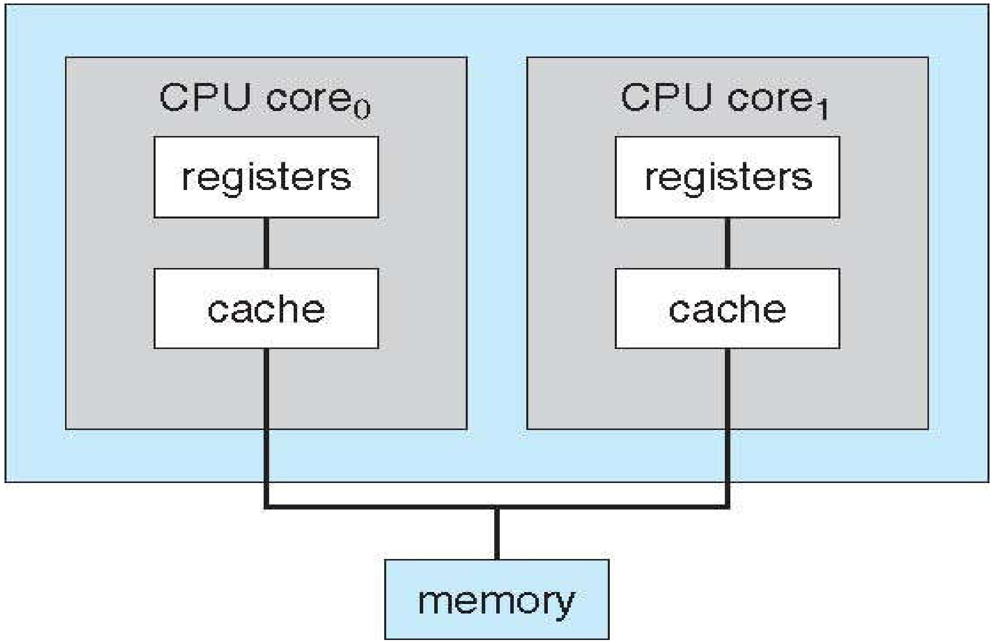


- Multiprocessors memory access divisions:
  - UMA (uniform memory access)
  - NUMA (non-uniform memory access)

a commercial version of UNIX designed by IBM

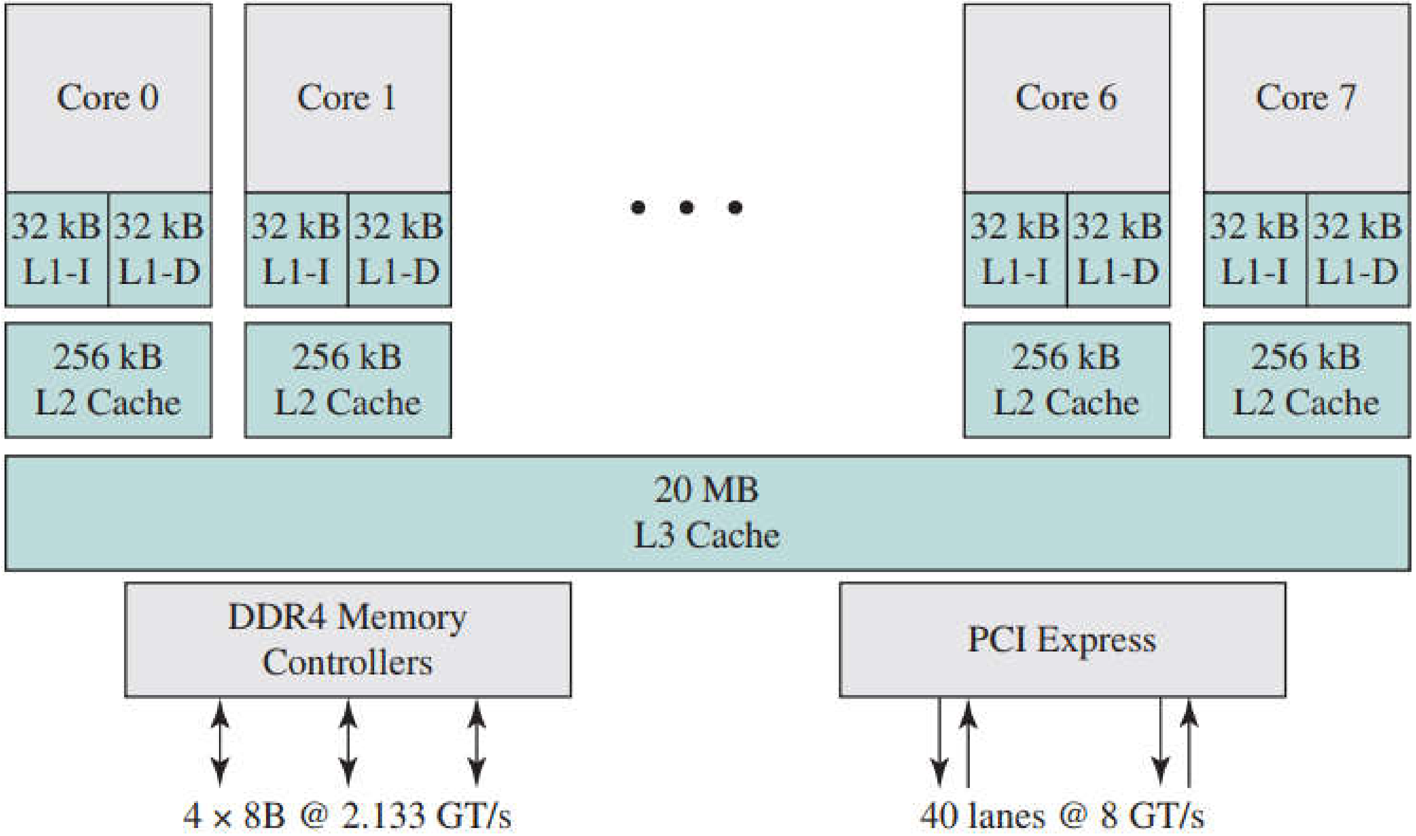
A dual-core design

- 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

Intel core i7-5960X block diagram



Intel Core i7-5960X Block Diagram

Blade servers

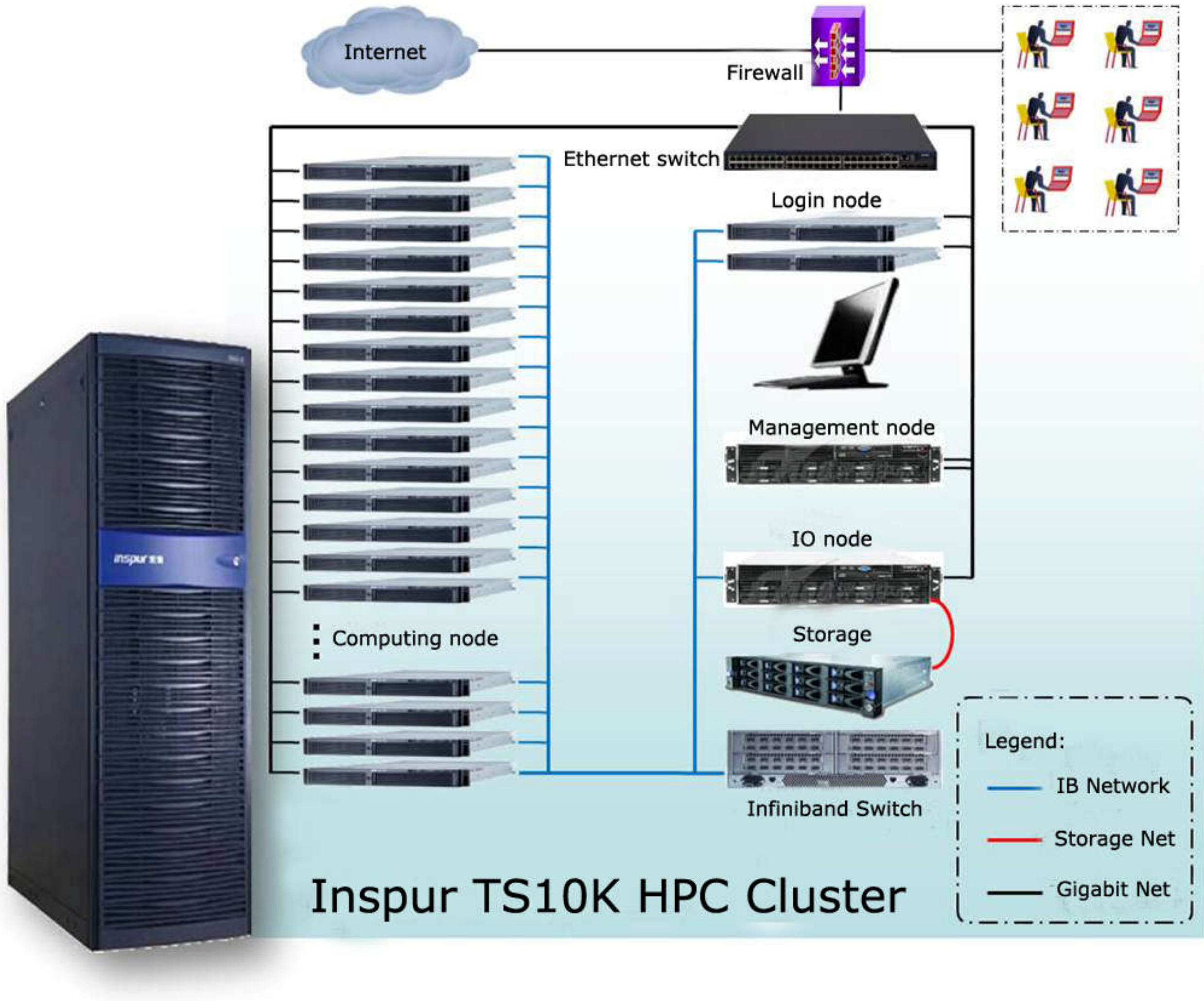
- Multiple processor boards, I/O boards, and networking boards are placed in the same chassis.

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

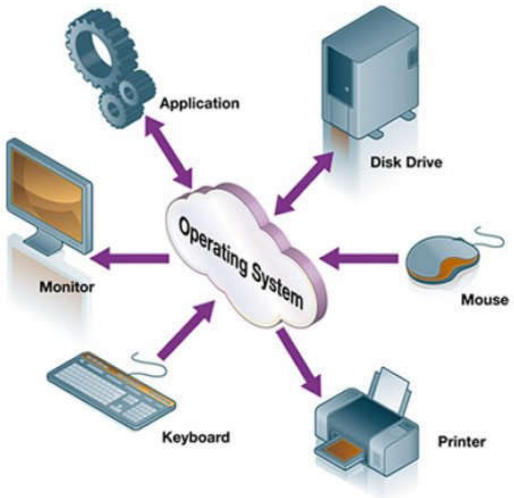
Clustered Systems

- Like multiprocessor systems, but multiple systems working together
  - 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



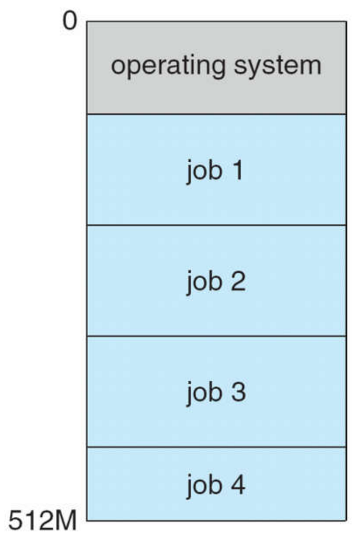
Operating System Structure



Operating System Structure

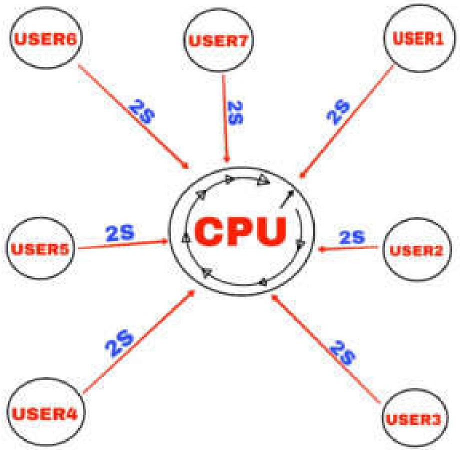
- Multiprogramming (Batch system) needed for efficiency
  - Single program cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via job scheduling
  - When it has to wait (for I/O for example), OS switches to another job

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



Operating System Structure(Cont.)

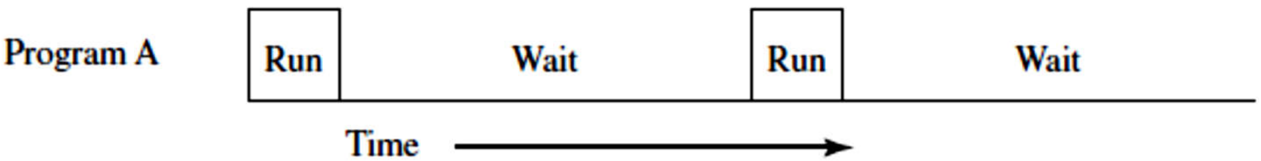
- **Timesharing (multitasking)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨ **process**
  - 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



System utilization example

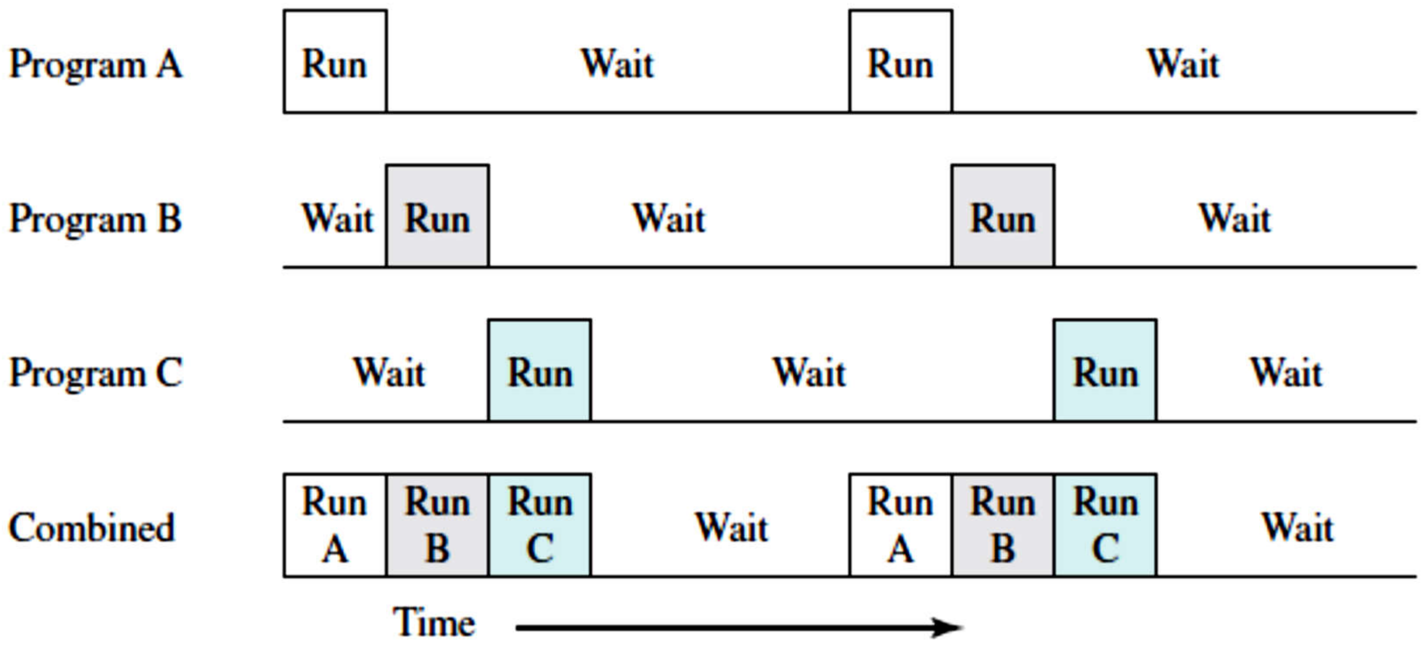
Read one record from file	15 μs
Execute 100 instructions	1 μs
Write one record to file	15 μs
Total	31 μs

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



(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

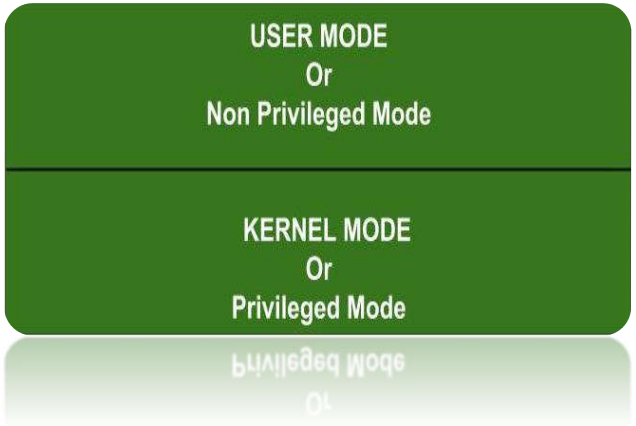
- Interrupt driven (hardware and software)
  - Hardware interrupt by one of the devices
  - Software interrupt (exception or trap):
    - Software error (e.g., division by zero)
    - Request for operating system service
    - Other process problems include infinite loop, processes modifying each other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

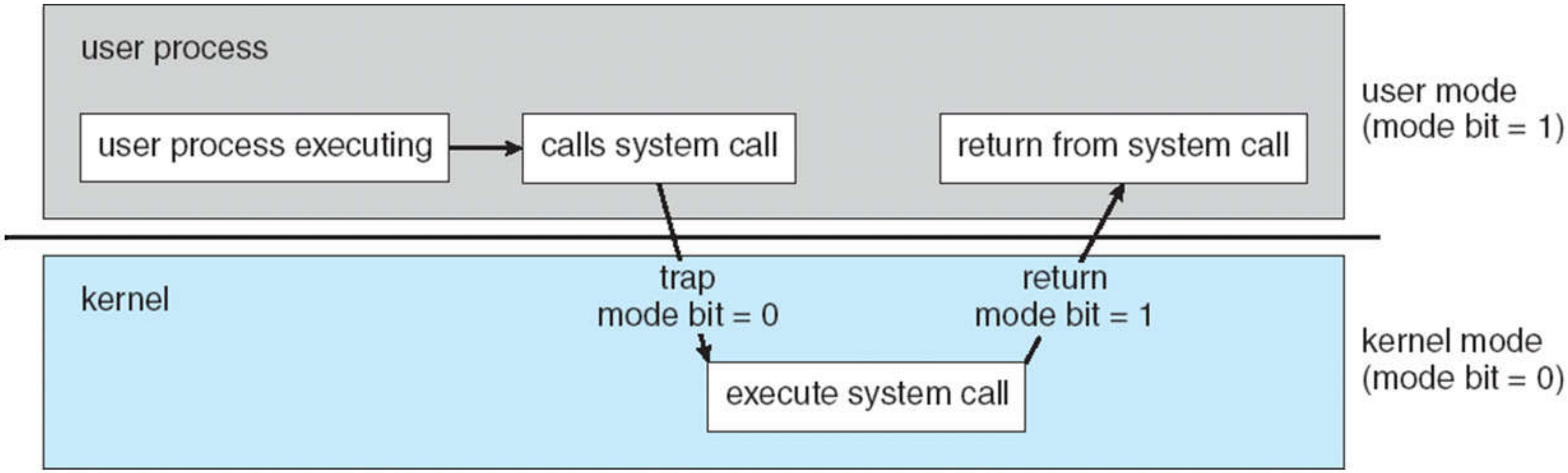
Privileged and Non-Privileged Instructions

- Privileged Instructions :
  - ✓ I/O instructions and Halt instructions
  - ✓ Turn off all Interrupts
  - ✓ Set the Timer
  - ✓ Context Switching
  - ✓ Modify entries in Device-status table
  - ➢ …
- Non-Privileged Instructions include
  - ✓ Reading the status of Processor
  - ✓ Reading the System Time
  - ✓ Generate any Trap Instruction
  - ✓ …



Transition from User to Kernel Mode

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



Process Management

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

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  – Optimizing CPU utilization and computer response to users
- Memory management activities
  – Keeping track of which parts of memory are currently being used and by whom
  – Deciding which processes (or parts thereof) and data to move into and out of memory
  – Allocating and deallocating memory space as needed

Storage Management

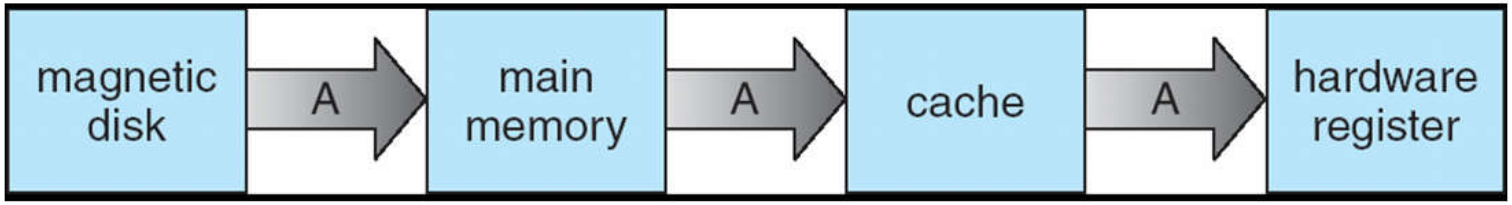
- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and dirs
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Mass-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Free-space management
  - Storage allocation
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Still must be managed – by OS or applications
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

Migration of Integer A from Disk to Register

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



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

I/O Subsystem

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

Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

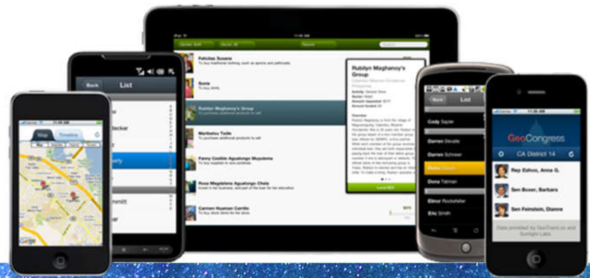
Computing Environments

Computing Environments - Traditional

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

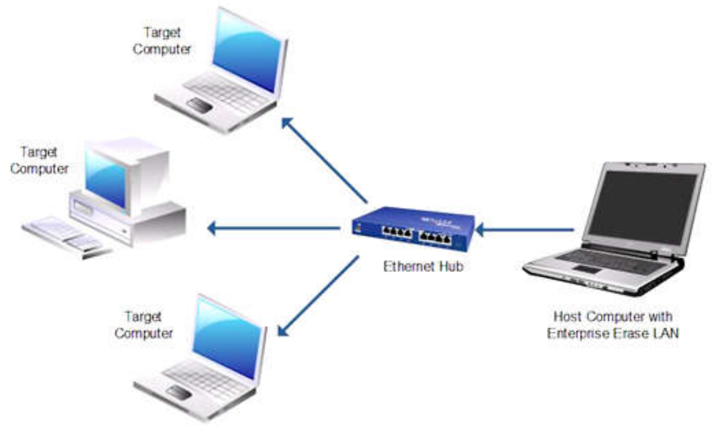
Computing Environments - Mobile

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



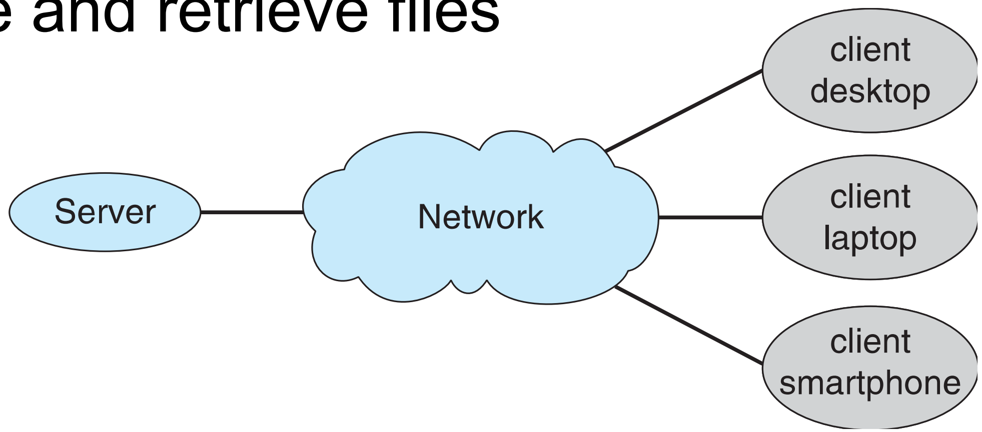
Computing Environments – Distributed

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

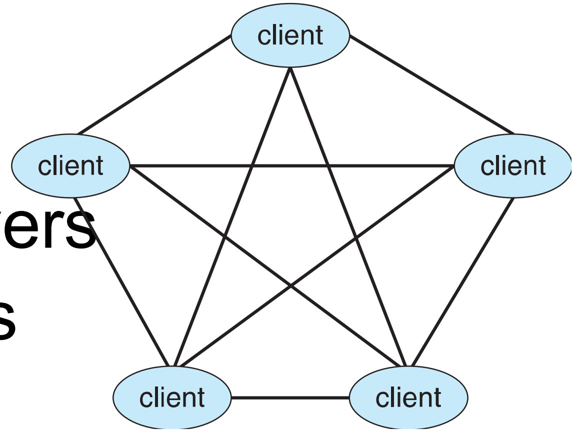


Computing Environments – Client-Server

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



Computing Environments - Peer-to-Peer



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

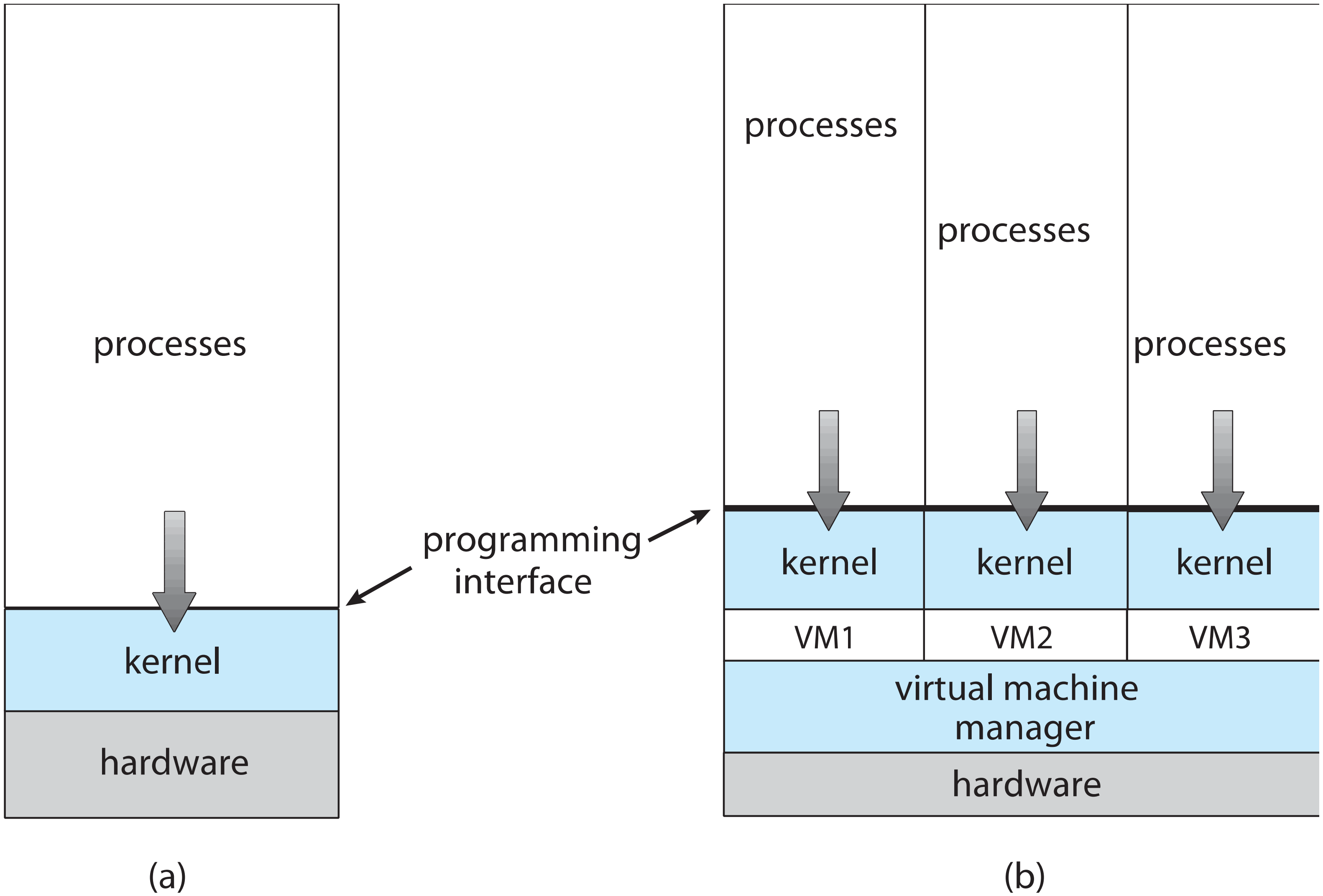
Computing Environments - Virtualization

- 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** provides virtualization services

Computing Environments - Virtualization

- 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

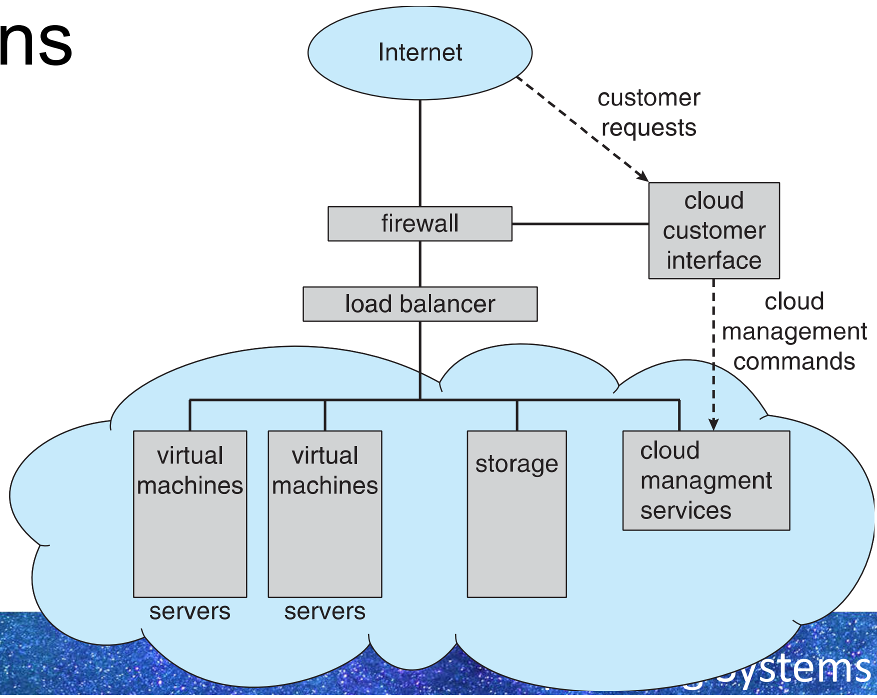


Computing Environments – Cloud Computing

- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization as based on virtualization
  - Amazon **EC2** has thousands of servers, millions of VMs, PBs 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 compute 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
  - Correct operation only if constraints met

In conclusion…

- Operating systems provide a virtual machine abstraction to handle diverse hardware
  - Operating systems simplify application development by providing standard services
- Operating systems coordinate resources and protect users from each other
  - Operating systems can provide an array of fault containment, fault tolerance, and fault recovery

End of Chapter 1