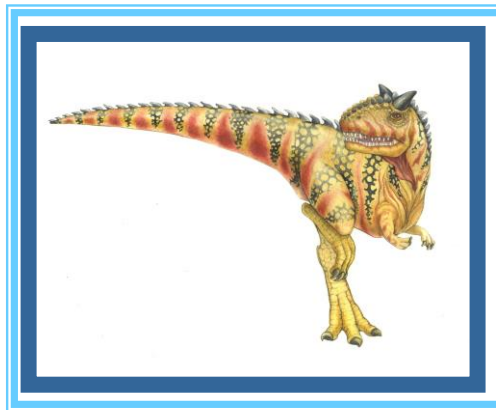


Chapter 1: Introduction





Chapter 1: Introduction

- ◆ What Operating Systems Do
- ◆ Computer-System Organization
- ◆ Computer-System Architecture
- ◆ Types of Operating-System
- ◆ Operating System Operations
- ◆ Operating System Components
- ◆ Computing Environments
- ◆ Open-Source Operating Systems





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
- ◆ To explore several open-source operating systems





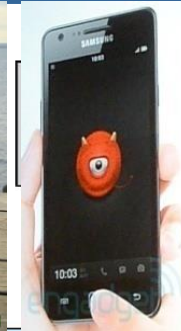
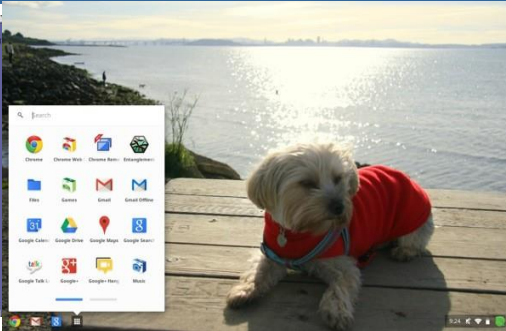
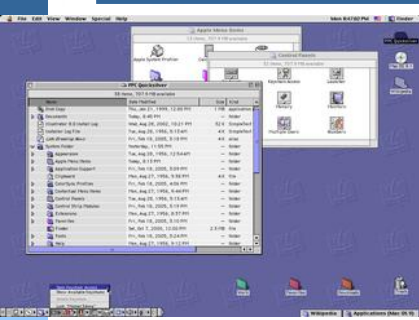
How do you use a computer ?

A box containing CPU, with a keyboard, a mouse and a monitor.



- ◆ What is the magic from your action of mouse clicking to really using the computer to do something?
 - Some one sits between you and the computer.(same to smart phone, sensors, ipod,.....).
 - It is the **manager** of the computer system!
- ◆ **The manager is Operating System.**





compiler

assembler

text editor

...

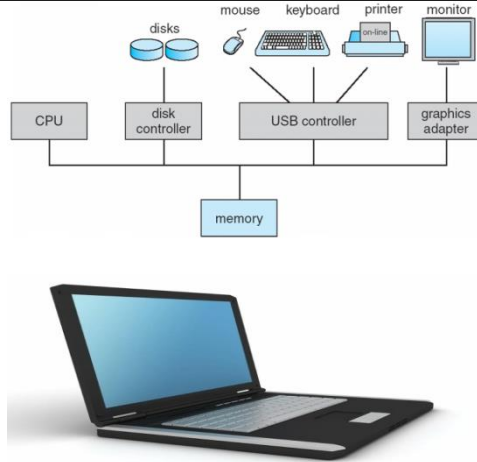
database system

system and application programs

OS=F()

Operating System

A program that acts as an intermediary between a user of a computer and the computer hardware.





What is an Operating System?

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

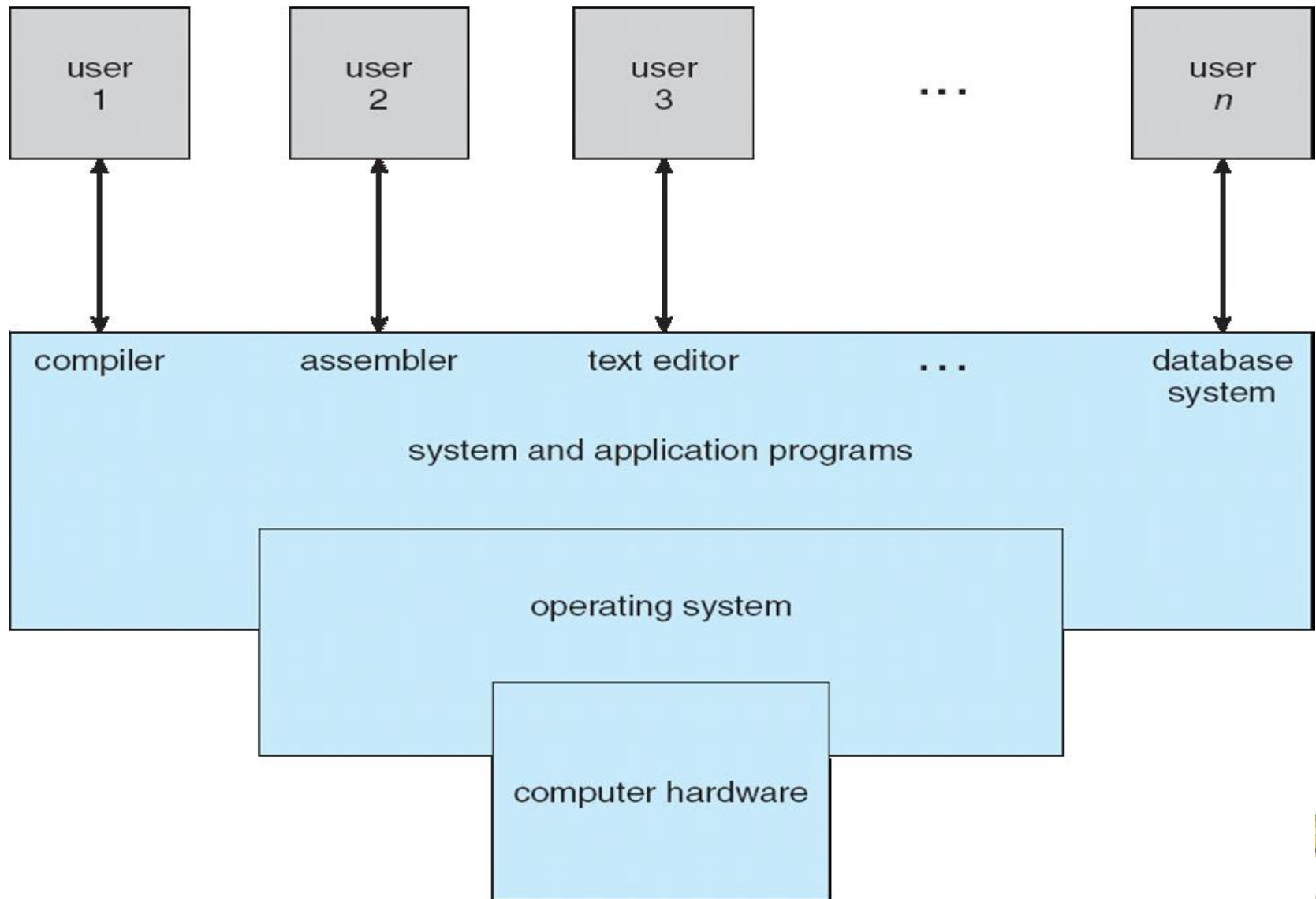
Users

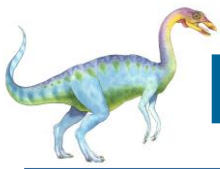
- ▶ People, machines, other computers



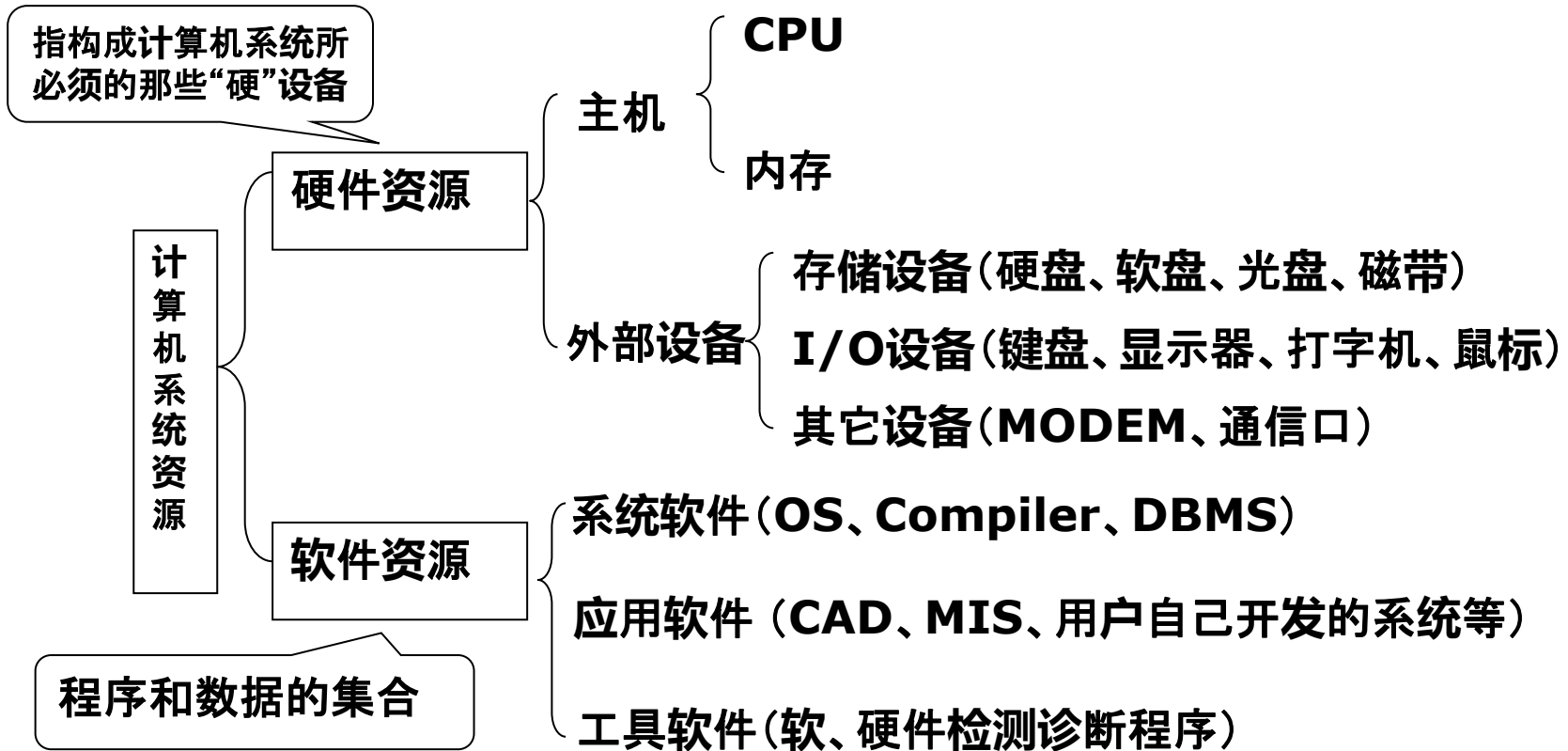


Four Components of a Computer System





Resources in computer system





Operating System Definition

- ◆ No universally accepted definition
- ◆ “Everything a vendor ships when you order an operating system” is a 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.





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

CPU Management	Allocating and controlling CPU
Memory Management	Allocating and deallocating memory space
Device Management	Allocating and controlling I/O devices
File Management	In charge of file access, sharing and protection





Operating System Definition (Cont)

Definition from Wikipedia

- ◆ A computer program that manages the hardware and software resources of a computer.
- ◆ At the foundation of all system software, it performs basic tasks such as **controlling** and **allocating memory**, prioritizing system **requests**, controlling input and output **devices**, facilitating **networking**, and managing **files**.





User mode and kernel mode

Dual-mode operation allows OS to protect itself and other system components (类似机构管理)

- ◆ **User mode**: a user mode process can only execute normal instructions, but not privileged instructions.
- ◆ **Kernel mode**: a kernel mode process can execute all instructions. Only OS processes should be executing in kernel mode.
 - ▶ It is also called system, privileged or supervisor mode.





User mode and kernel mode

Hardware provides a **mode bit**.

Provides ability to distinguish when system is running **user code** or **kernel code**

- ◆ A process running with user mode bit **on** is a user process.
- ◆ A process running with user mode bit **off** is a kernel or system process.

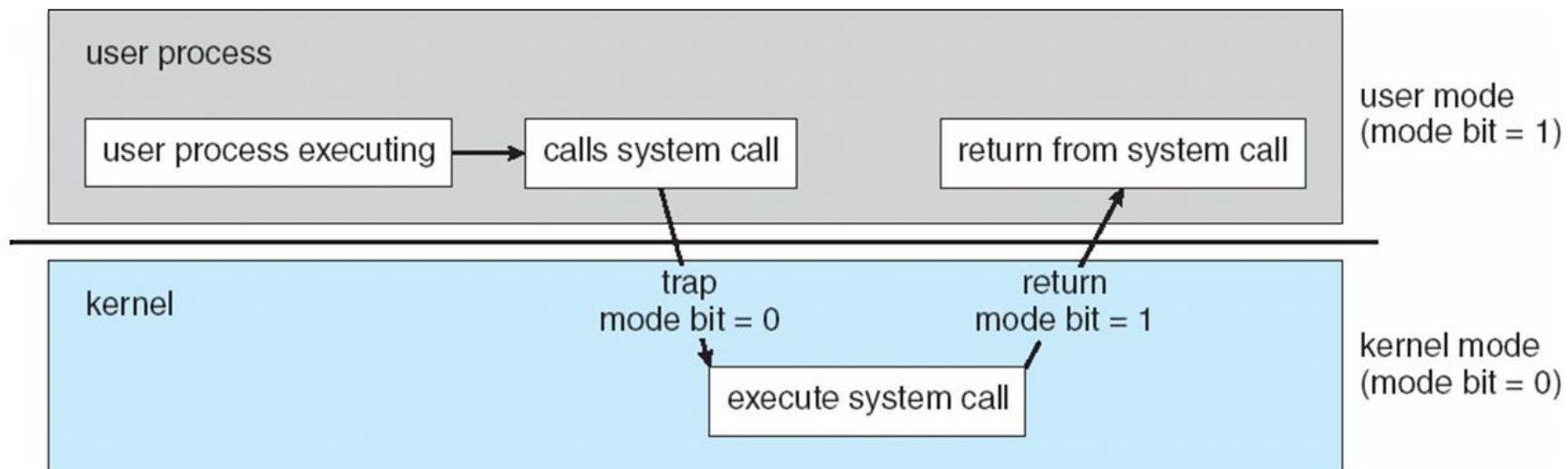




User mode and kernel mode

If a user process executes in user mode, how can it do I/O that needs to execute privileged instructions?

- ◆ Achieved through **System Calls**.
- ◆ A system call is an entry port to the OS.
- ◆ A system call is actually a **trap** that changes **from user mode to kernel mode**, executes the privileged I/O command, returns from system call and reverts **back to user mode** to continue execution.





Computer Startup

bootstrap program is loaded at power-up or reboot

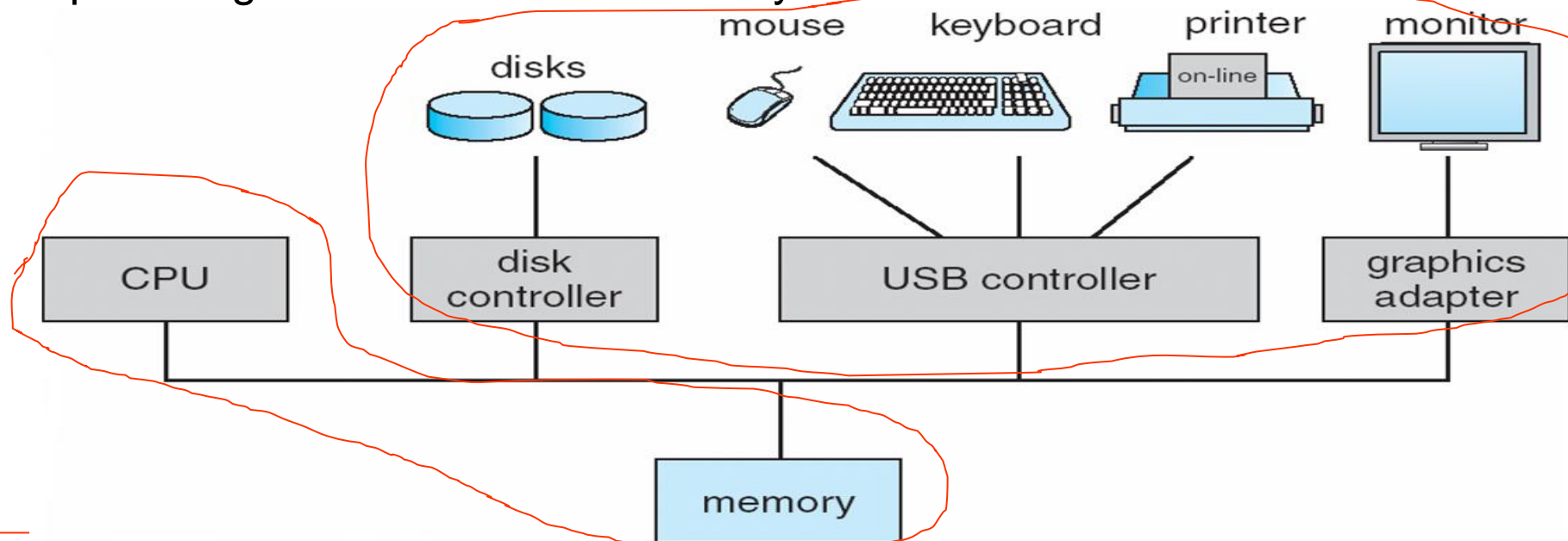
- ◆ Typically stored in ROM or EPROM, generally known as **firmware**
- ◆ Initializes all aspects of system
- ◆ Loads operating system kernel and starts execution





Computer-System Operation

- One or more CPUs, device controllers connect through common bus providing access to shared memory



- ◆ Each device controller is in charge of a particular device type, which has a local **buffer**, so I/O devices and the CPU can execute concurrently;
- ◆ 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**;





Interrupt (OS是由中断驱动的)

- ◆ OS is itself a program and that a process is a program in execution.
 - So OS will present itself as one or more processes in the computer system.
 - When there are multiple processes executing, how can the OS control and manage them?
- ◆ The OS relies on a special alarm system, called the **interrupt processing mechanism**.





Interrupt

An interrupt is a signal to the CPU to tell it about the occurrence of a major event.

- An error in calculation (illegal instruction, division by zero).
- Completion of an I/O.
- Hardware alarm.
- User-generated event (very often called a **trap**).
- When an interrupt occurs, there is a chance for OS to grasp the CPU and make decision and arrangement to control the system.





Interrupt

Two types of interrupts:

Maskable interrupt: interrupt that may be ignored or handled later. A lower priority interrupt is maskable.

Non-maskable interrupt: interrupt that cannot be ignored.

- ▶ The CPU must handle this interrupt immediately. A non-maskable interrupt may be “over”-interrupted by another non-maskable higher priority one.

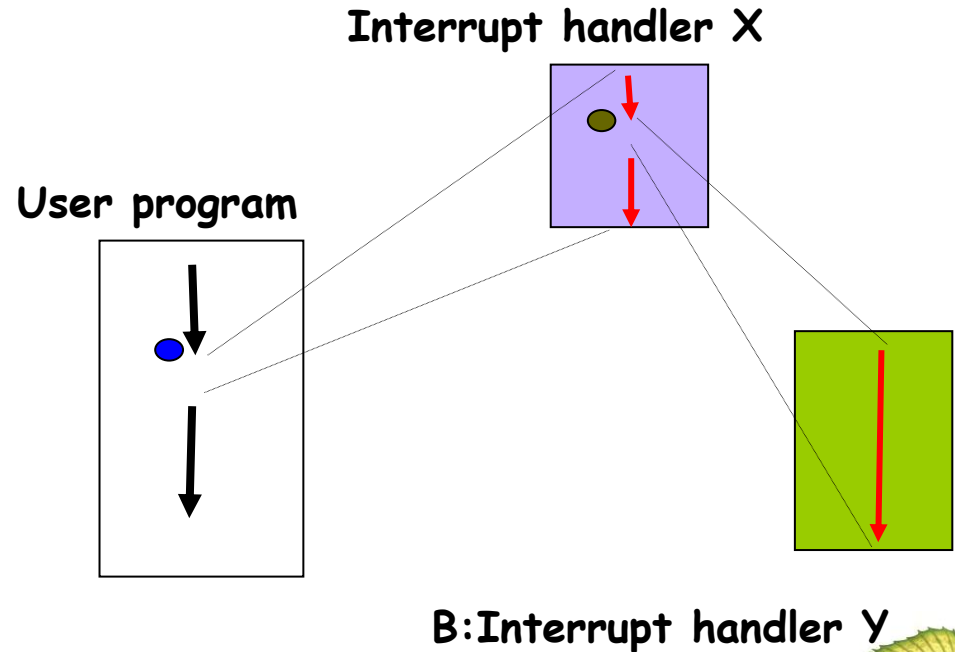
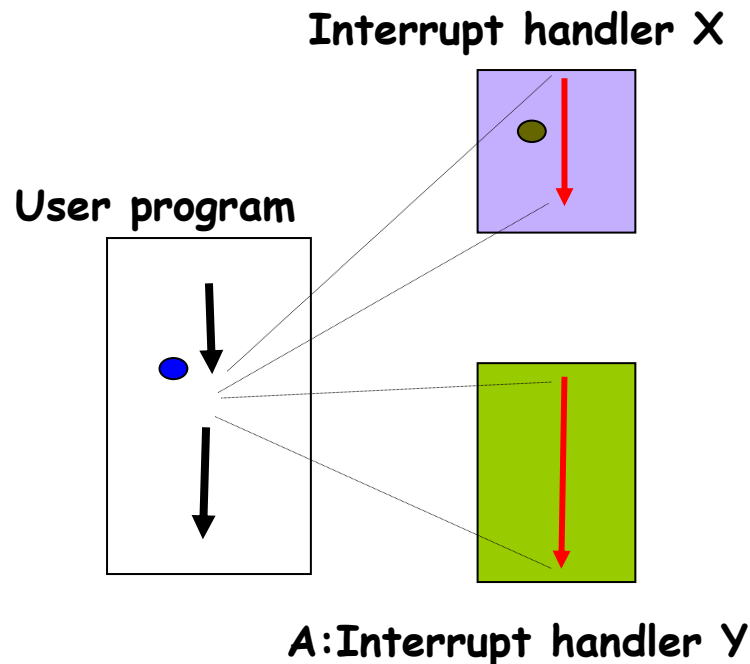




Interrupt Handling

It is possible that when an interrupt is being serviced, another interrupt may arrive.

- ◆ If the incoming new interrupt Y has a lower priority, it will wait till the first one X completes. (A)
- ◆ If the new interrupt Y has a higher priority, it will seize the CPU from the current interrupt handler X. (B)





Interrupt

Depending on the cause of interrupts, we could classify them into 3 categories.

- ◆ **Program interrupt:**

- ◆ software-generated user interrupt caused either by a request or an error, e.g. illegal instruction, overflow, division by zero, memory access violation, special instruction.(called **traps**).

- ◆ **I/O interrupt:**

- ◆ Caused by I/O related events, e.g. I/O completion or device errors.

- ◆ **Timer interrupt:**

- ◆ Caused by the hardware timer of the system to handle time-related activities.

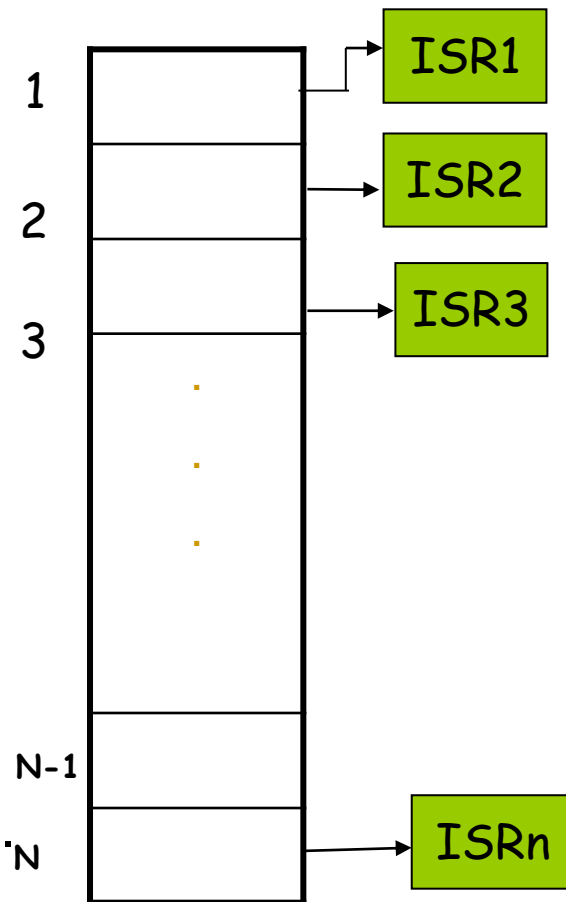




Interrupt Handling

- ◆ After an (non-maskable) interrupt occurs, CPU will put aside the process it is executing, by saving the program counter and register values.
- ◆ It looks up an **interrupt table**, using the **interrupt number** as an index, for a procedure for execution.
- ◆ This procedure is called the **interrupt handler**, or interrupt service routine(ISR). It is executed in response to the interrupt to serve the interrupt.
- ◆ CPU then jumps to interrupt handler and executes it.
- ◆ After the interrupt is serviced, the CPU resumes the suspended program and returns to the next instruction pointed to by the saved PC.

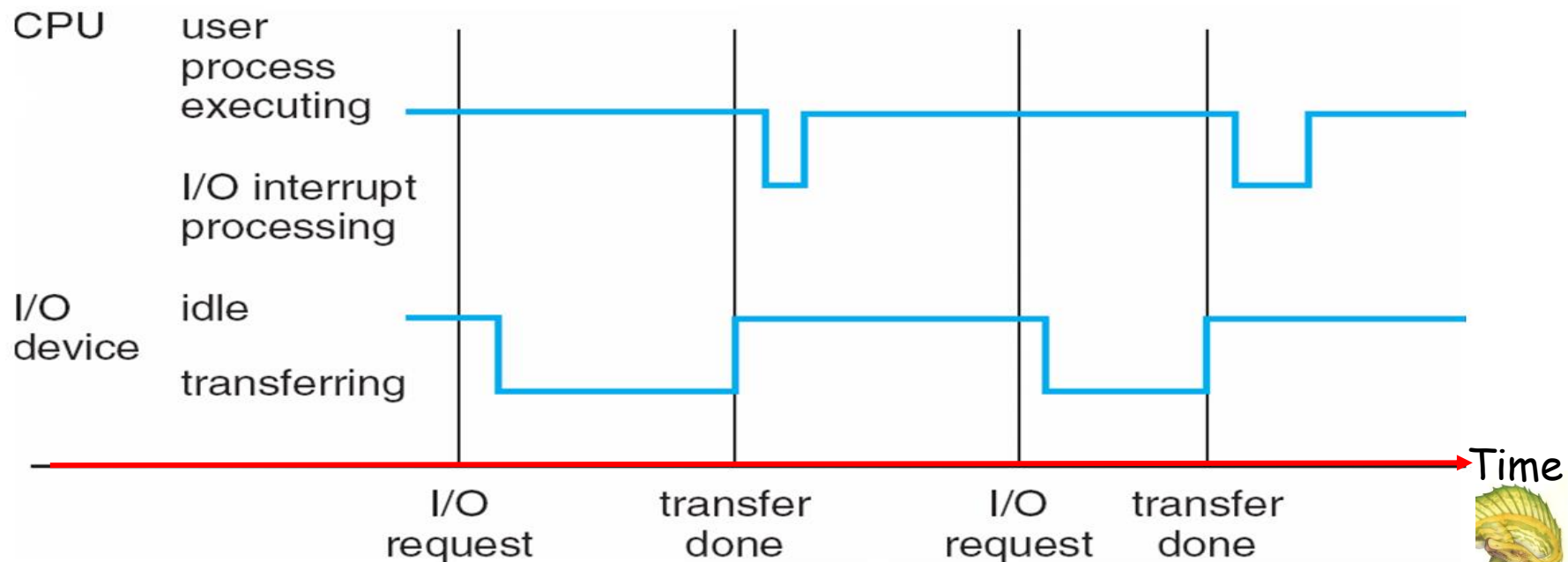
Interrupt table





Interrupt Timeline

- ◆ One can perform I/O when CPU is doing useful work .
- ◆ When I/O is completed, OS needs to put aside its current work and looks at I/O device for next I/O.
 - The event that I/O is completed causes an **interrupt**.
 - The suspension (interruption) of current work to handle the event is called **interrupt processing**.



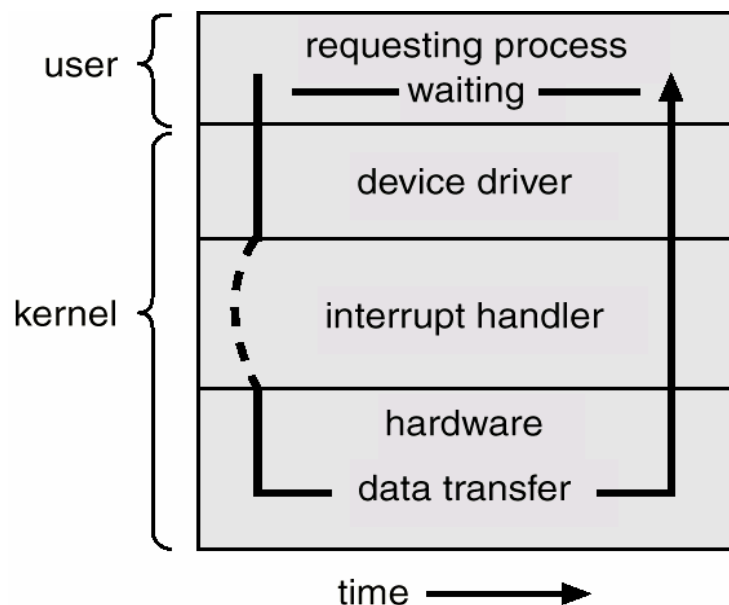


I/O Processing

Two methods for I/O processing

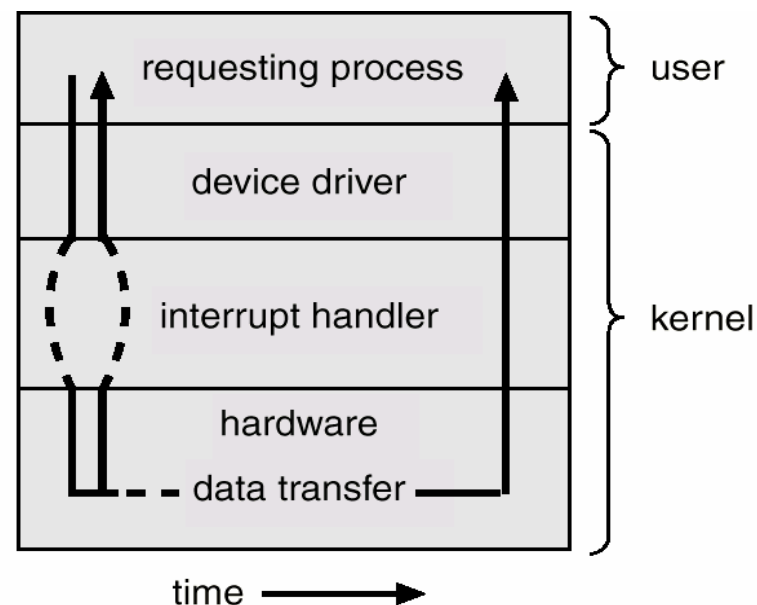
Synchronous

Asynchronous



Synchronous

(a)



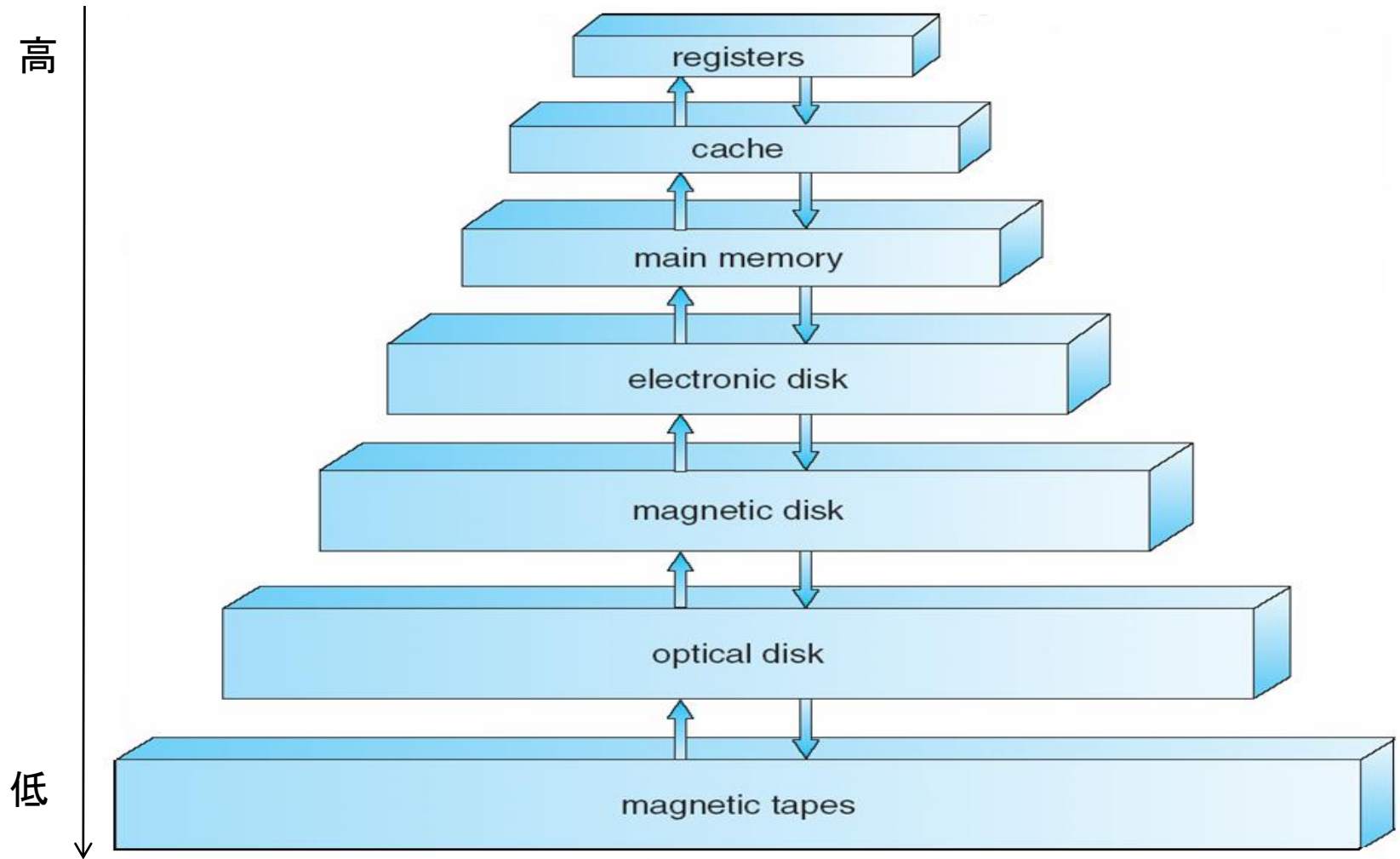
(b) **Asynchronous**





Storage-Device Hierarchy

Storage systems organized in hierarchy: Speed / Cost / Volatility



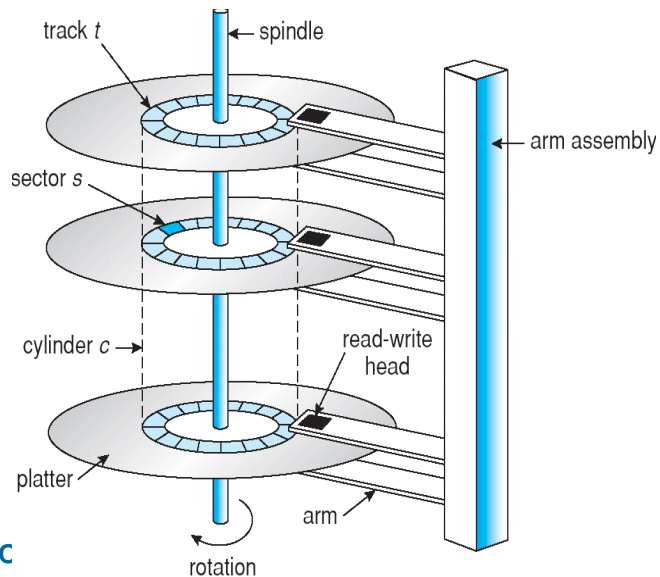


Storage Structure

Main memory – only large storage media that the CPU can access directly

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





Storage Structure

- ◆ Electronic disk is a data storage device using integrated circuit assemblies as memory to store data persistently(such as solid-state drive (SSD) or solid-state disk。存储介质:FLASH芯片或DRAM)





The diagram illustrates a write-back cache system. On the left is the CPU, in the center is the cache, and on the right is the main memory (主存). The cache is divided into two sections: 'cache 标记区' (cache tag area) at the top and 'cache 缓冲区' (cache buffer area) at the bottom. A horizontal line labeled '主存地址' (main memory address) runs from the CPU to the main memory. Data flow is indicated by arrows: a solid arrow labeled '命中' (hit) goes from the cache buffer to the CPU; a dashed arrow labeled '未命中' (miss) goes from the cache tag area to the main memory; a solid arrow labeled '数据 (块)' (data block) goes from the main memory to the cache buffer; and a solid arrow labeled '数据 (字)' (data word) goes from the main memory to the CPU. A double asterisk (**) is placed near the bottom right of the main memory.

- ◆ **Cache**是位于CPU和主存之间速度快的存储器, 减少CPU等待时间, 提高系统性能(小但是快);
- ◆ **Buffer**用于存储速度不同步的设备或优先级不同的设备之间传输数据, 减少实际的I/O操作次数;
- ◆ main memory can be viewed as a last *cache* for secondary storage



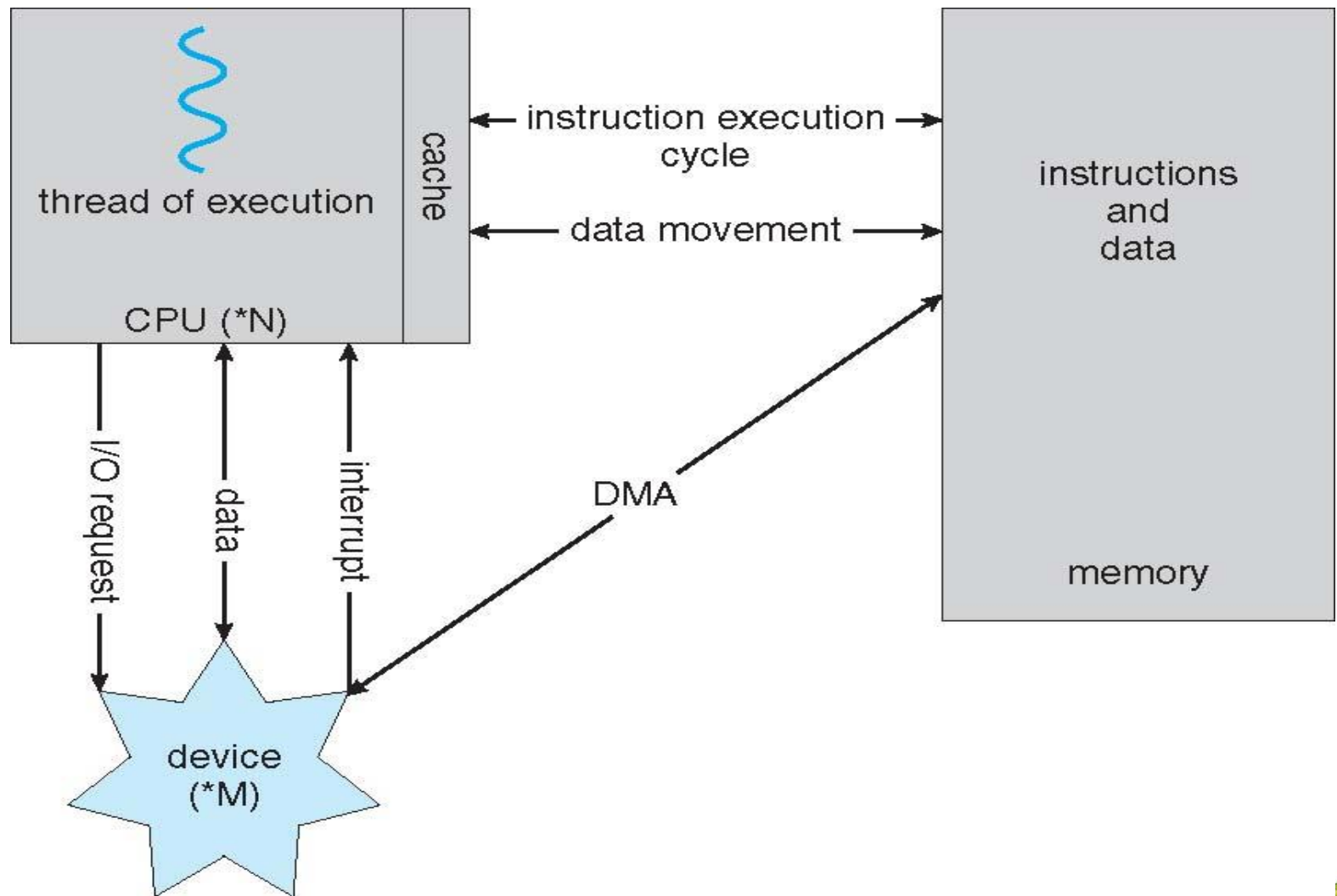
Caching(命中率)

- ◆ 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





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
 - Multiprocessors are usually for high-performance computing (HPC)





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

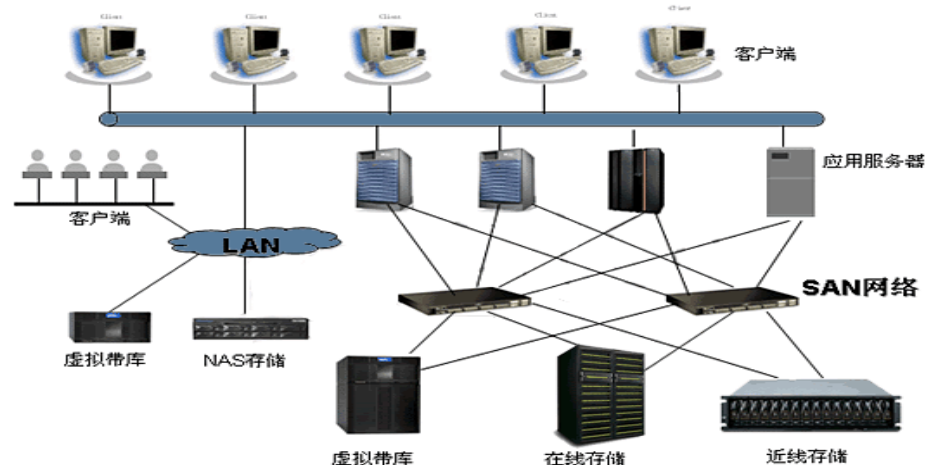
- ▶ **Asymmetric clustering** has one machine in hot-standby mode
- ▶ **Symmetric clustering** has multiple nodes running applications, monitoring each other

Some clusters are for **HPC**

- ▶ Applications must be written to use **parallelization**



图2 典型的SAN环境





Types of Operating System

- ◆ Simple Batch processing system
- ◆ Multiplied Batch processing system
- ◆ Time-sharing system
- ◆ Real time system
- ◆ Parallel Systems
- ◆ Distributed system

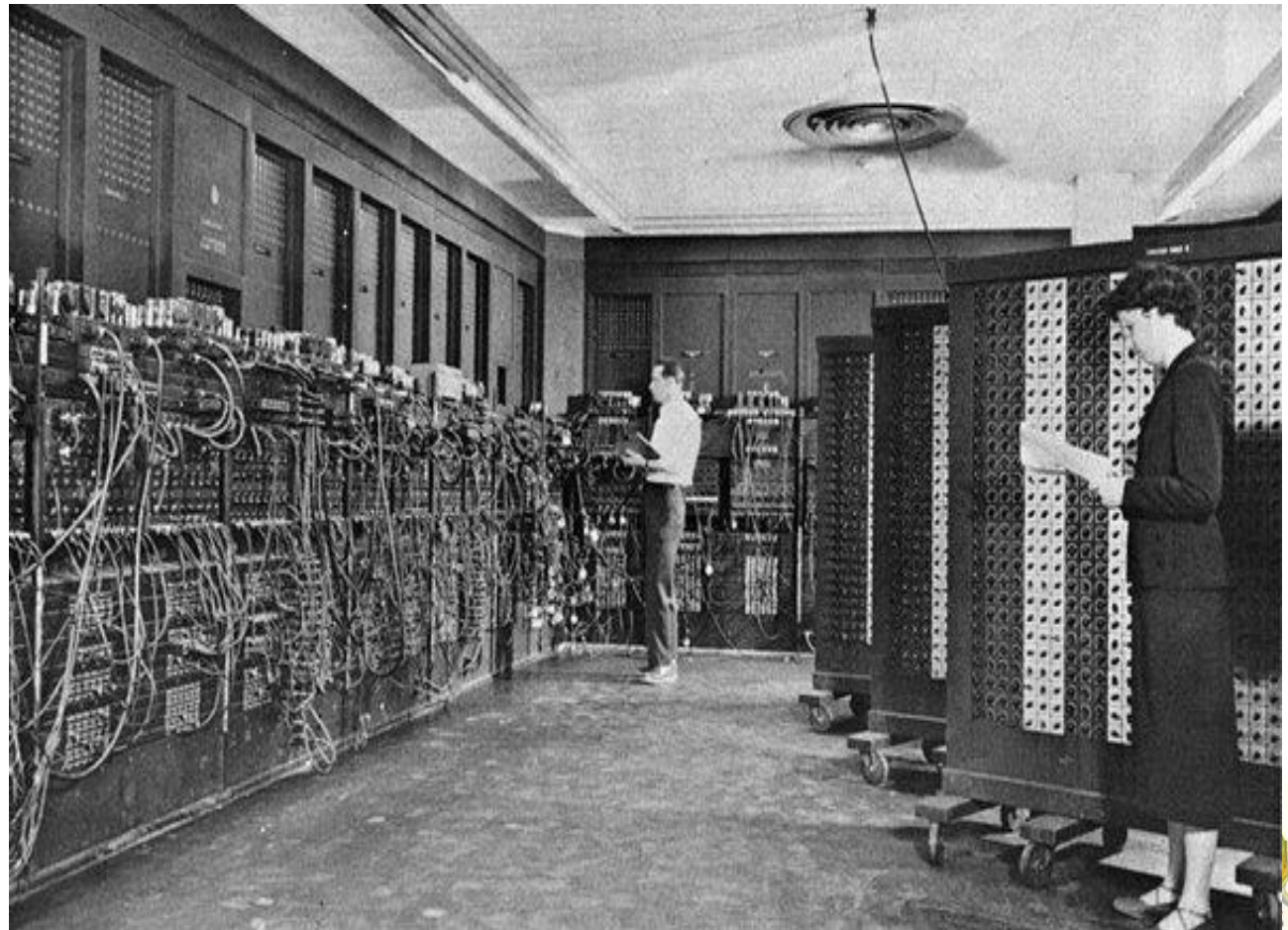




ENIAC

1946年2月14日，第一台通用电子计算机 ENIAC (Electronic Numerical Integrator And Computer)，它是图灵完全的电子计算机，能重新编程，解决各种计算问题。

使用了18800 个真空管，占地 1500 平方英尺，重 30 吨(约1间半教室)，ENIAC 耗电巨大，每次开机，整个费城西区的电灯都为之黯然失色。



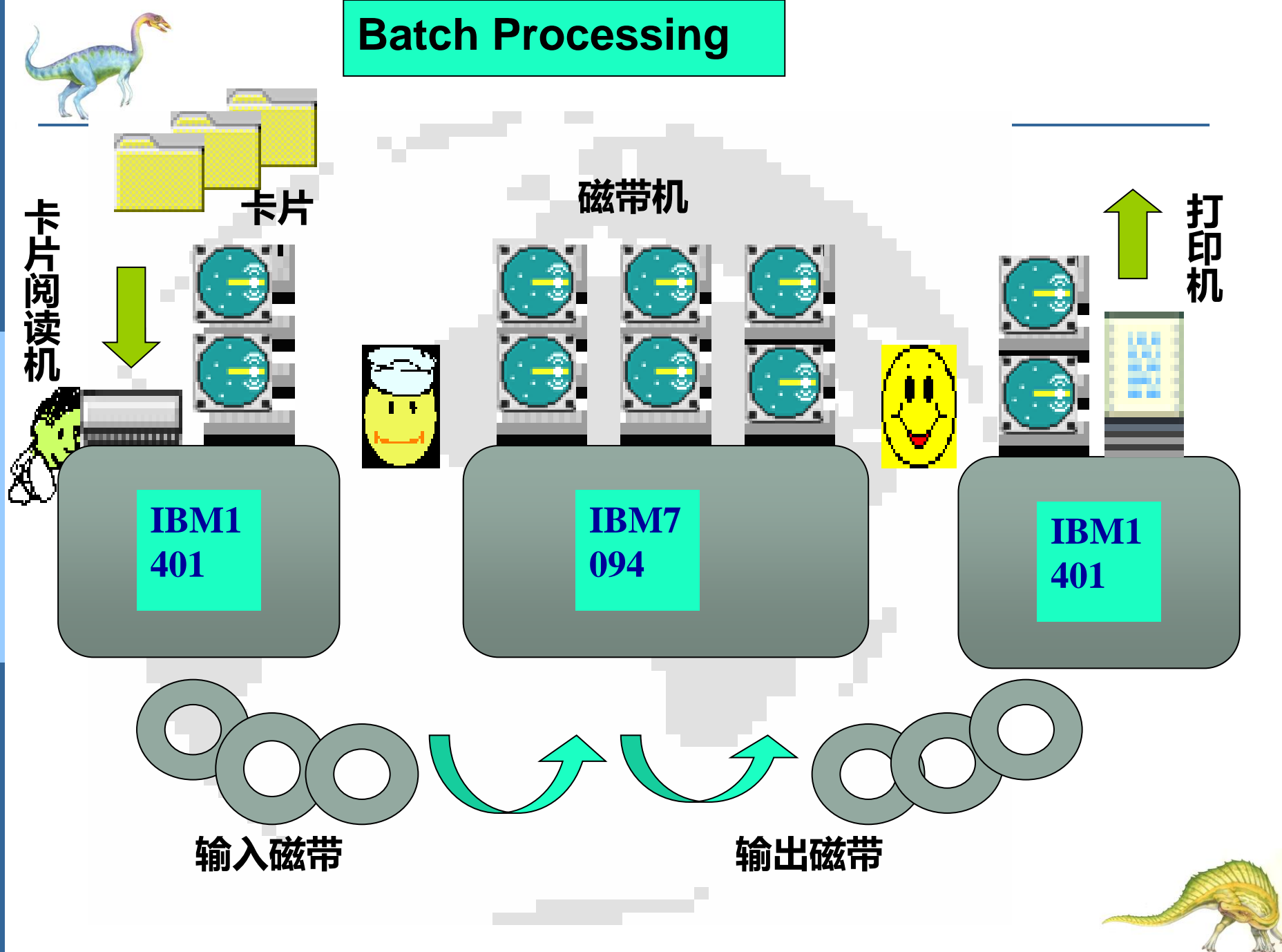


Simple Batch processing

- ◆ Reduce setup time by batching similar jobs
- ◆ Automatic job sequencing – automatically transfers control from one job to another.
- ◆ Hire an operator (User \neq operator)
- ◆ Add a card reader
- ◆ Common systems in 60's like IBM JCL-related (job control language) systems.

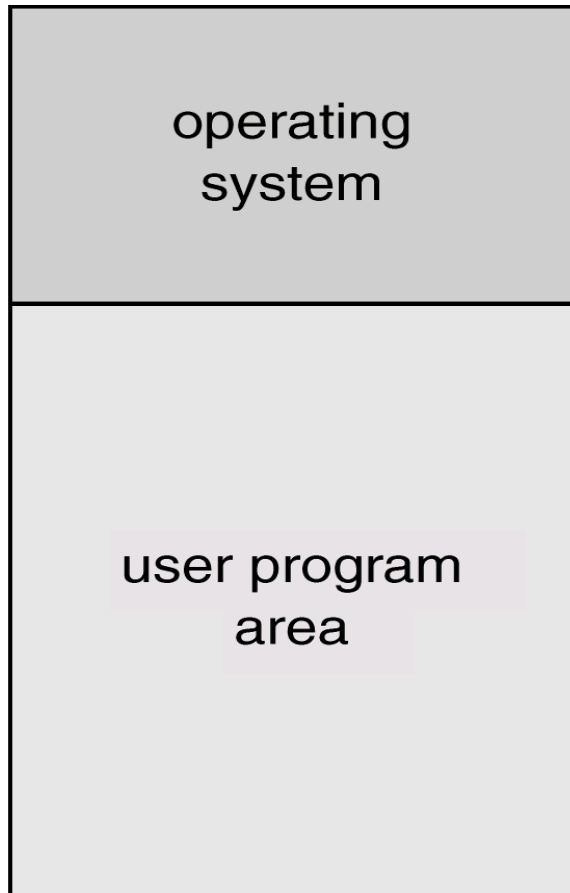


Batch Processing





Memory Layout for a Simple Batch System



Simple Batch System examples

- IBM FMS (FORTRAN Monitor System), for IBM7094;
- IBM IBSYS, for IBM7090 and 7094;
- UM UMES, for IBM7094;





Types of Operating Systems

◆ Multiprogramming(in 70's)

- Programs are executed in interleaved manner by the CPU.
- CPU is given to another program when the current program is waiting for I/O.

◆ OS Features Needed for Multiprogramming

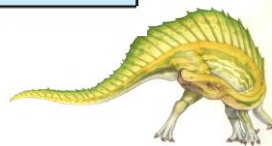
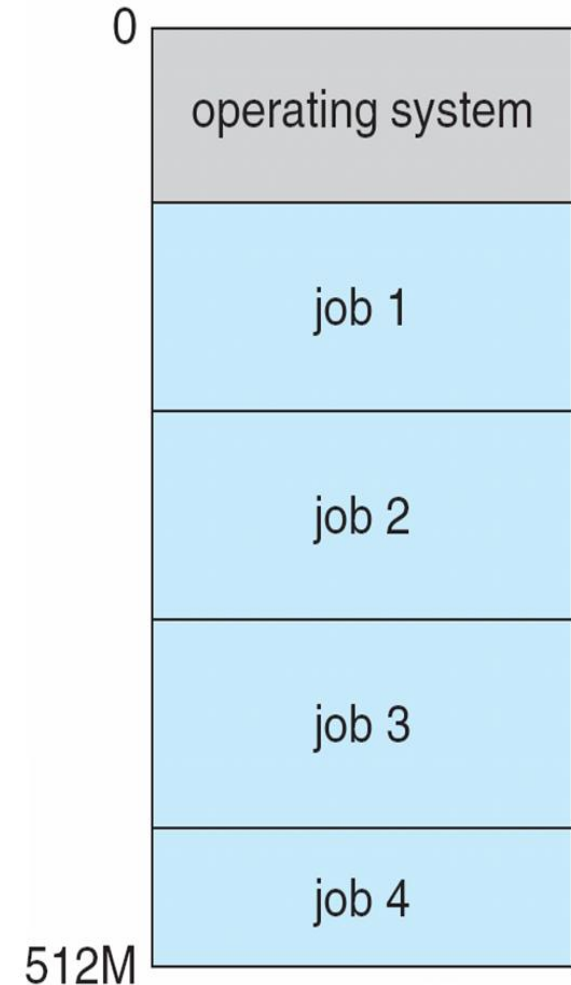
- I/O routine supplied by the system
- Memory management and protection
- CPU scheduling
- Allocation of devices





Multiplied Batch Processing System

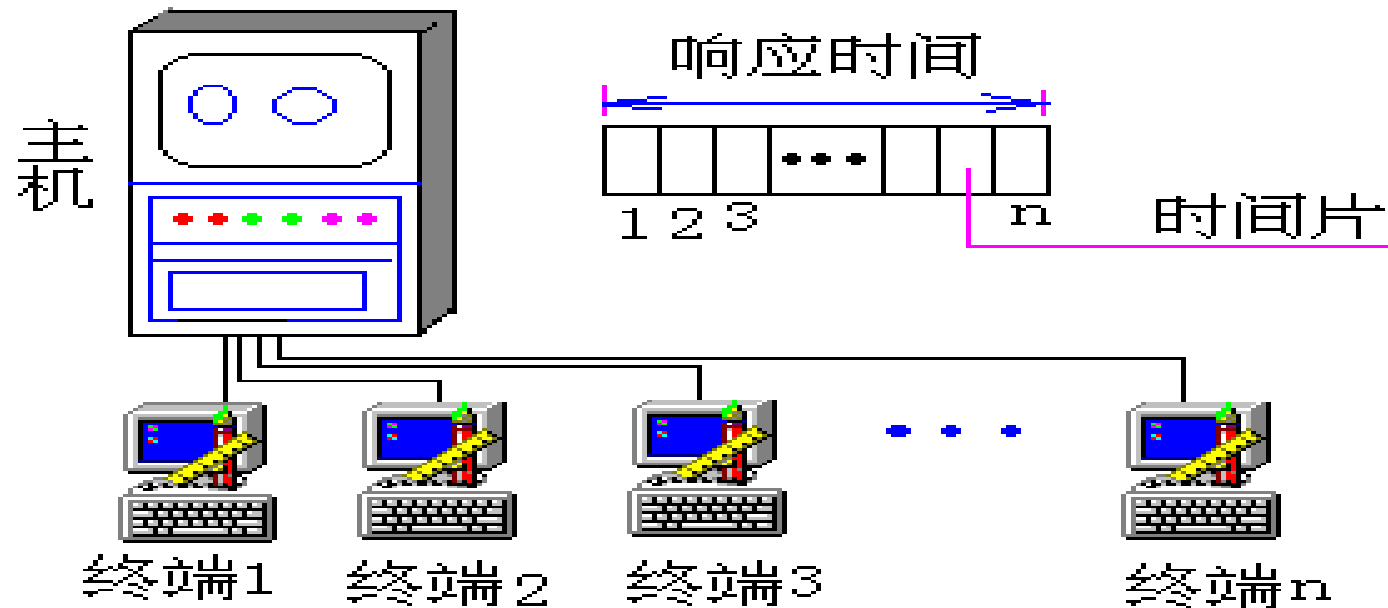
- ◆ Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.
- ◆ System Parameter
 - Throughput(吞吐量)
 - Turnaround time(周转时间)





Time-sharing System

- ◆ Processes are executed in interleaved manner by the CPU.
- ◆ Same as multiprogramming, but system is **interactive**, so each process must get CPU without waiting for too long.





Time-sharing System

- ◆ **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





Time-sharing System

- ◆ Key issue: in time
- ◆ How to: Time slice

Famous examples: MULTICS, UNIX

MULTICS

- ▶ MIT , Bell Lab of AT&T, DEC
- ▶ 1964-1969, PL/1

UNIX

- ▶ 1970
- ▶ Bell Lab of AT&T





*nix Operating Systems

- ◆ Unix and Linux are the most influential operating systems and form the largest installation base behind Windows.
 - They are often referred to as the *nix family.
- ◆ The first useful Unix was developed in 1970 at AT&T Bell Lab, written in assembly and C language.
 - **C language** was invented because of a need to develop Unix.





*nix Operating Systems

- ◆ Two major flavors for Unix:
 - BSD Unix developed by UC Berkeley for academic use.
 - ▶ Solaris is developed from BSD and is major Unix installation.
 - ▶ FreeBSD is also a very common Unix installation.
 - ▶ Mac OS X is designed to be POSIX-compliant.
 - **Portable Operating System Interface**, a standard for Unix-like OS.
 - System V by AT&T, which had led to IBM AIX.
 - Each flavor is packaged by different distributions into Linux.
 - ▶ BSD Unix is packaged under RedHat (e.g. Fedora), Novell (e.g. OpenSUSE), Ubuntu derived from Debian.





Real-time System

- ◆ Special-purpose OS
- ◆ A time-sharing system where processes have completion deadlines that must be met.
- ◆ Important in **high-valued** and **mission critical** applications.
 - Banking transactions, airline reservation, space shuttle control etc.





Real-Time System

■ Hard real-time systems have firm deadlines.

- ◆ Guarantees that critical tasks be completed on time
- ◆ Requires that all delays in the system be bounded
- ◆ Missing a deadline makes a process useless, e.g. examination.

■ Soft real-time systems have soft deadlines.

- ◆ A less restrictive type of real-time system, where a critical real-time task gets priority over other tasks, and retains that priority until it completes
- ◆ Missing a deadline reduces value of a process, e.g. assignment.





Distributed Systems

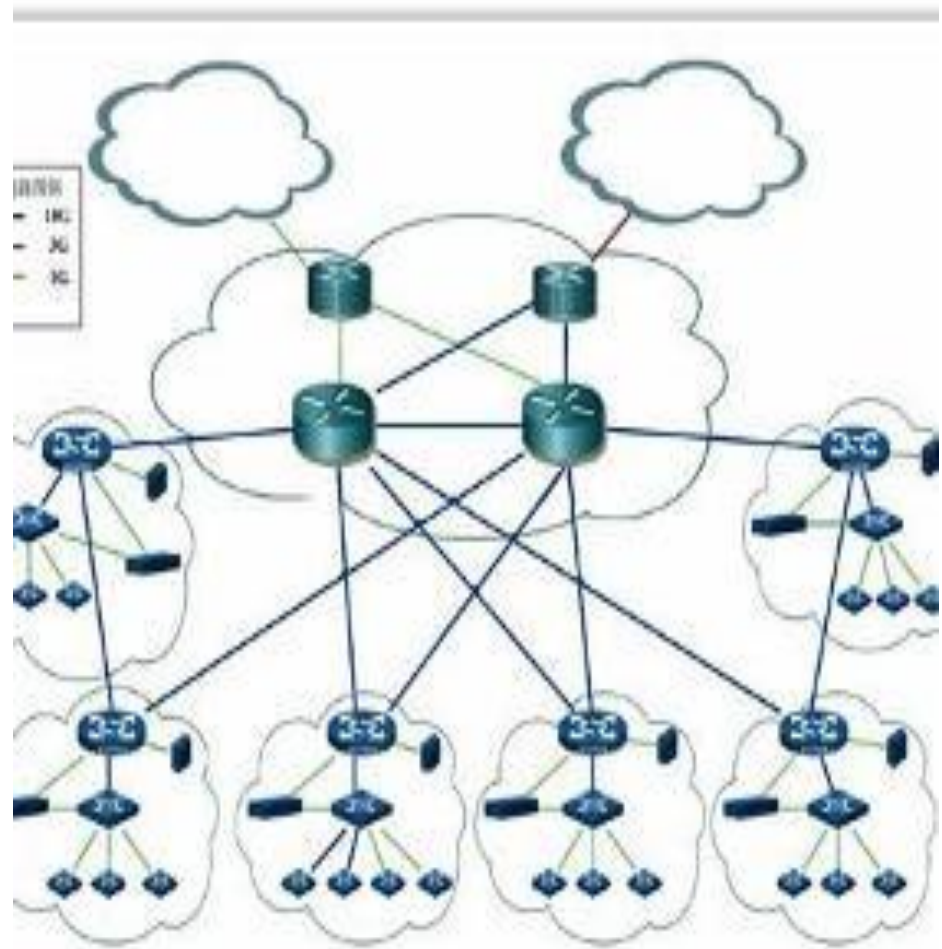
- ◆ Operated upon a collection of computers scattered (分散) across the network.
- ◆ Advanced from network operating systems.
- ◆ Need to coordinate the share of resource and execution of processes to achieve a common goal.
- ◆ Examples:
 - Client/Server system, P2P, internet computing system





Distributed Systems

- ◆ Loosely coupled system
 - each processor has its own local memory
 - processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- ◆ Services and data are just **transparent** with respect to location and access (**Users do not see the existence of the network.**)





Distributed Systems

◆ Distributed OS example:

- Data sharing via Google+ or iCloud.

◆ Advantages

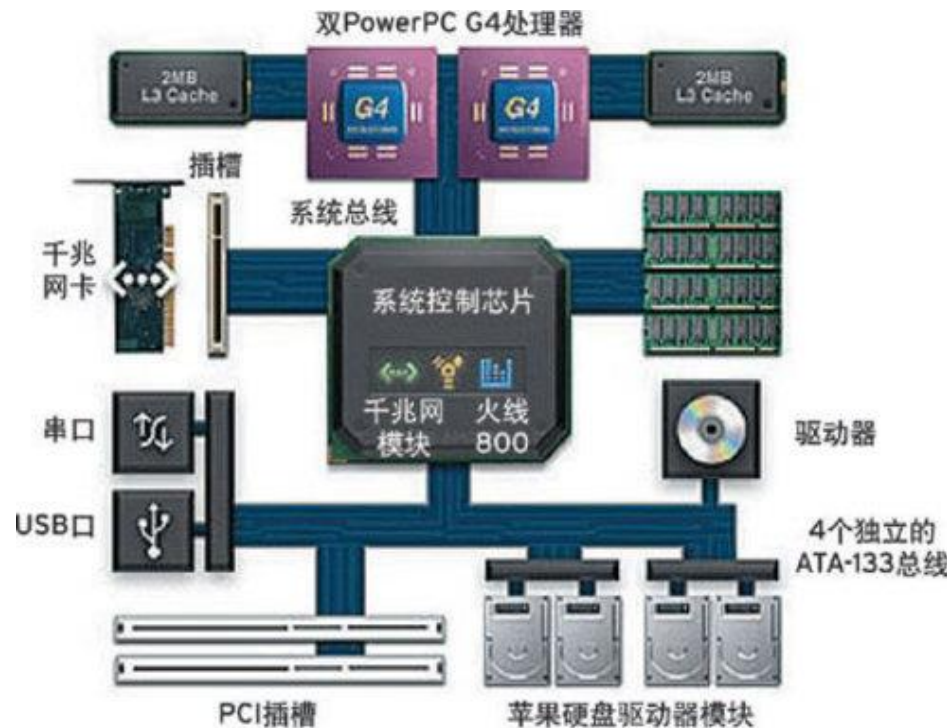
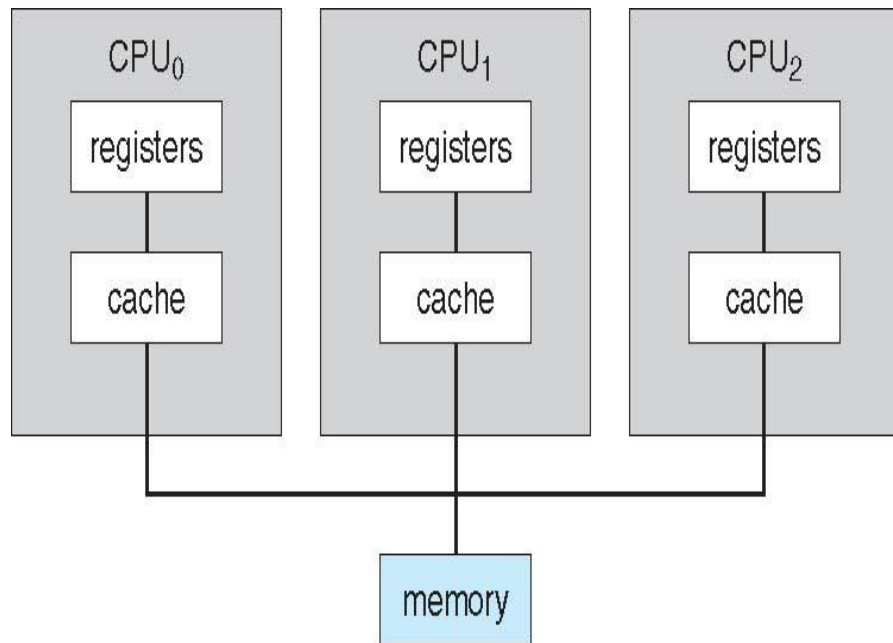
- Resources sharing
- Computation speed up – load sharing
- Reliability
- Communications





Parallel Systems

- ◆ Multiprocessor systems with more than one CPU in close communication.
- ◆ Tightly coupled system – processors share memory and a clock; communication usually takes place through the shared memory.



Power多处理器

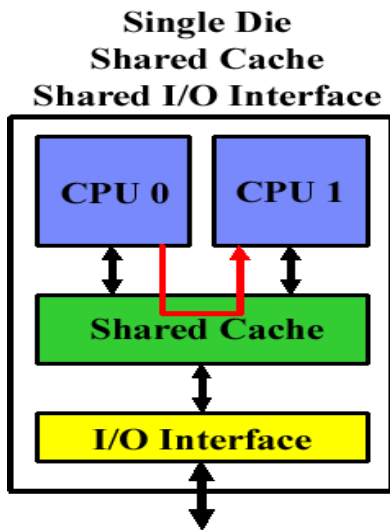




Multicore

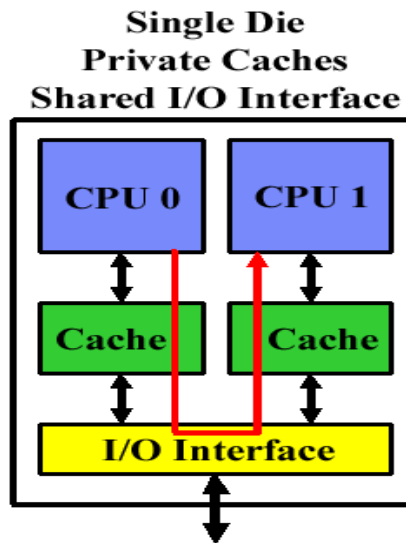
- ◆ CMP——Chip Multi-Processing 芯片多处理技术，即多核处理器，为线程并行化而设计的。
- ◆ 3 Designs of Multicore

1) Shared Cache



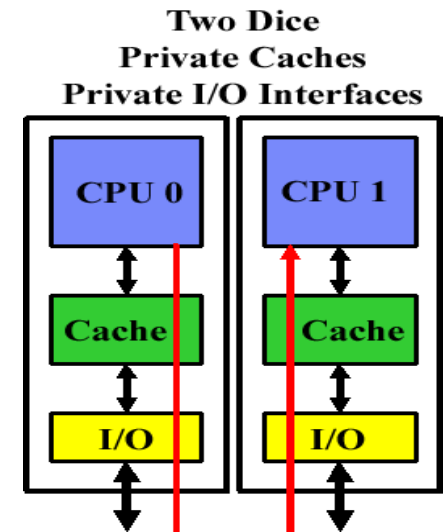
Production:
IBM POWER4/5 family
Sun UltraSPARC-IV
Fujitsu SPARC64-VI
Sun Niagara
Intel Yonah/Merom family

2) Shared I/O Interface



Production:
Intel Itanium2
AMD dual-core
Opteron

3) Shared data packet



Production:
Intel Pentium D





Parallel Systems

◆ Advantages

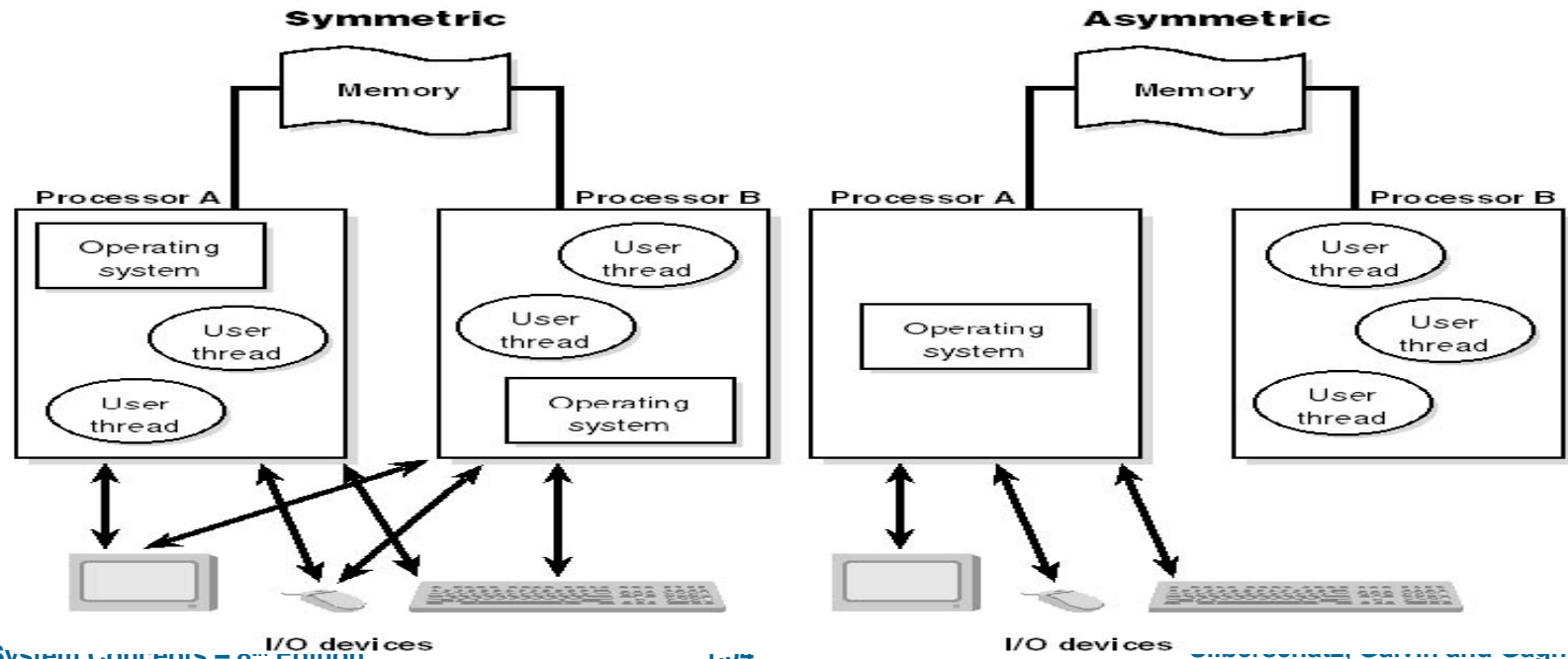
- Increased throughput
- Economical
- Increased reliability
- Graceful degradation
- Fail-soft system





Parallel Systems

- ◆ Symmetric multiprocessing (SMP)
 - Each processor runs an identical copy of the operating system
 - Most modern operating systems support SMP
- ◆ Asymmetric multiprocessing
 - Each processor is assigned a specific task; master processor schedules and allocates work to slave processors.
 - More common in extremely large systems





Network Operating System

- ◆ Add the networking capability to an operating system.
 - Network communication
 - Network services
- ◆ Unix allows you to write programs to communicate with another program on another computer.
- ◆ Network OS Example:
 - Unix, Linux, Windows 8.





Embedded Operating System

What is an Embedded System?

Consumer Products

Application Products

others

**Smartphone
Pocket PC**



**Smart
payer**



**Vehicle
device**



Gateway



**POS
ATM**



**Security
device**



**Industry control
Medical device
OA**





Embedded Operating System

◆ According to IEEE:

- Embedded Systems are “devices used to control, monitor, or assist(协助) the operation of equipment, machinery or plants”

◆ Embedded OS: Working in embedded systems

- Special purpose system
- Should be customized
- Size can be cut out
- Low energy consumption
- Real time

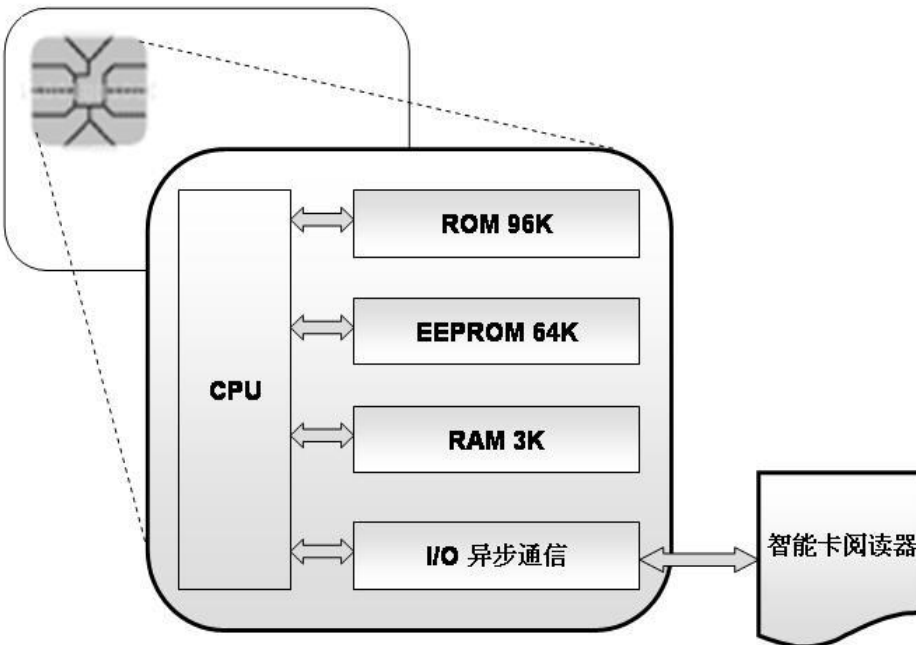
◆ Micro sensor with Tiny OS

◆ UC Berkeley





Smart card OS



- Communication mode: command-response
- Reader send command to the card ;
- OS: decrypt, verify and interpret ;
- OS then calls application to process the command, and sends the encrypted response to the reader





Operating System Components

- ◆ Four important components within OS.

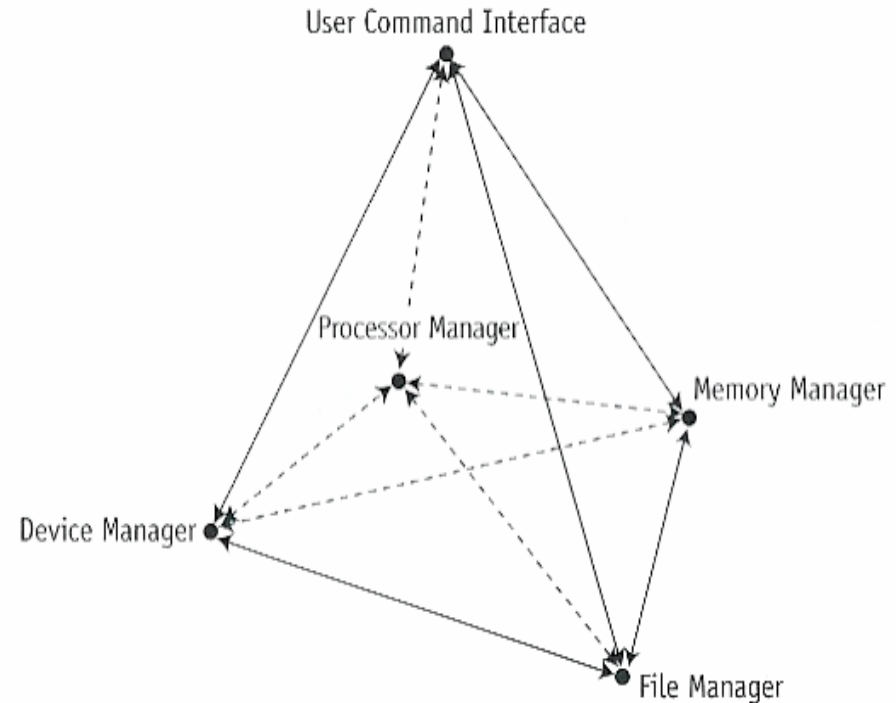
- Processor manager
- Memory manager
- File manager
- Device manager

- ◆ These components interact with one another.

- Generic issues like protection and security

- ◆ They also interact with and are controlled by users.

- User command interface





Protection and Security

- ◆ Other common functions needed of OS include protection and security.
- ◆ Protection(授权访问)
 - A mechanism for **controlling access** of processes or users to resources defined by the system.
 - Provide ways for users to specify the control and to enforce it.
- ◆ Security(防护攻击)
 - The **defense** of the computer system **against** internal and external **attacks**.
 - Types of attacks: worms, viruses, denial-of-service, identity theft, theft of service.
 - OS can only protect against some of the attacks(有限防护). Others should be managed by human.
 - ▶ For example, if you give away passwords easily, the OS cannot protect you, but the use of biometrics may help.





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*** (增强现实)
- ◆ 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

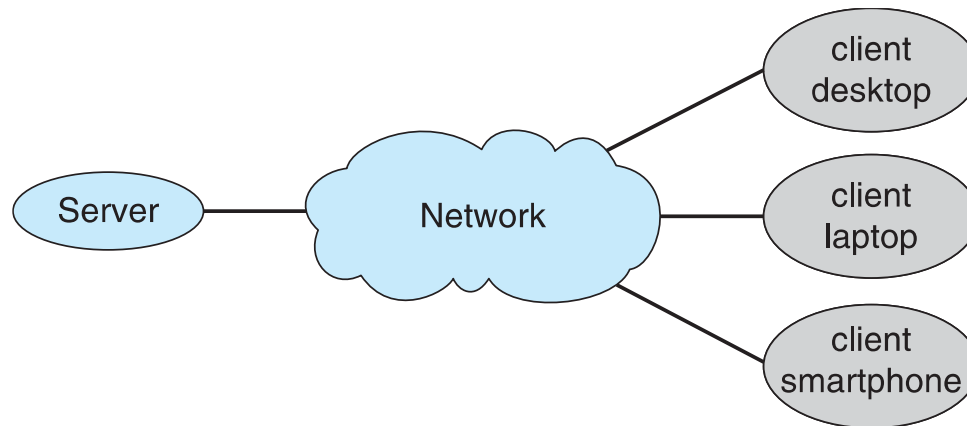




Computing Environments – Client-Server

◆ 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 – Cloud Computing

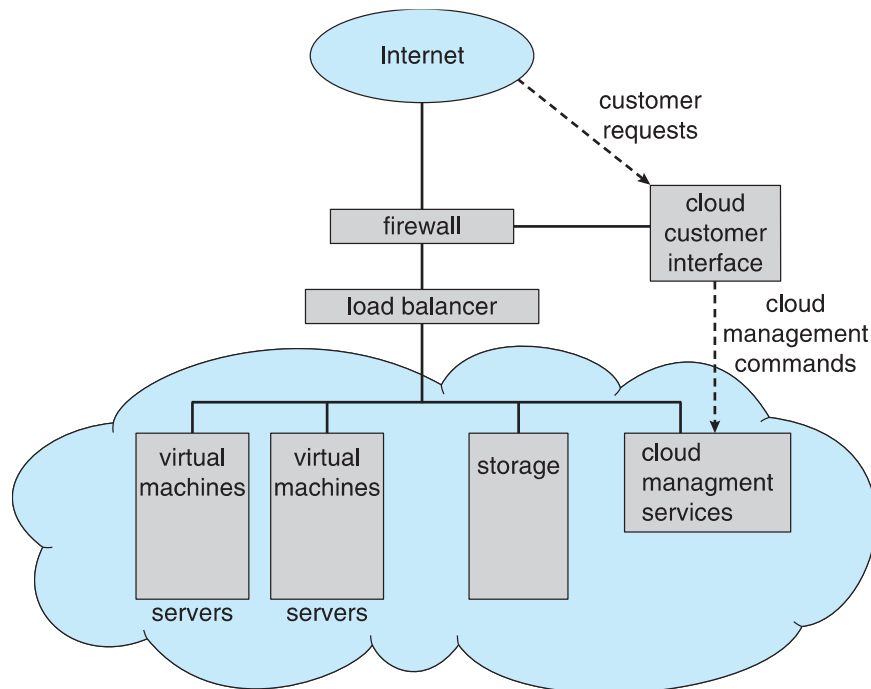
- ◆ Delivers computing, storage, even apps as a service across a network
- ◆ Logical extension of virtualization because it uses virtualization as the base for its functionality.
 - Amazon **EC2** has thousands of servers, millions of virtual machines, petabytes of storage available across the Internet,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

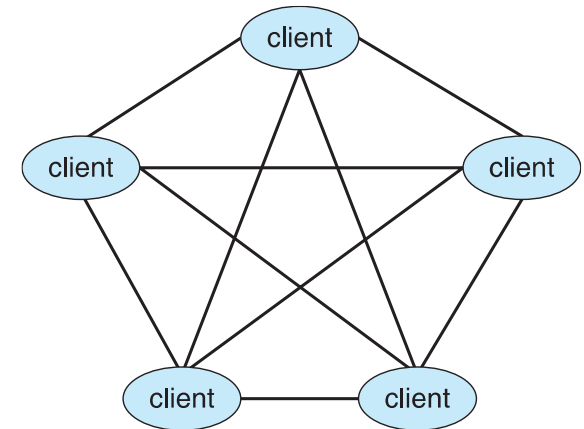
- ◆ Cloud computing environments composed of traditional OSeS, plus VMMs, plus cloud management tools
 - Internet connectivity requires security like firewalls
 - Load balancers spread traffic across multiple applications





Computing Environments - 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, (BLOCKCHEN)





Distributed Computing Paradigm

- **Message passing**
- **Remote Procedure Call**
- **Client-server**
- **Peer-to-Peer**
- **Message system:**
 - **Point-to-point;**
 - **Publish/Subscribe**
- **Distributed objects:**
 - **Remote method invocation**
 - **Network services**
 - **Object Request Broker**
 - **Object space**
 - **Component Based Technologies**
- **Mobile agents**
- **Collaborative 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





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) (GNU是一个自由的操作系统, “GNU ‘s Not Unix!”, GPL: GNU通用公共许可证: 要求软件以源代码的形式发布, 并规定任何用户能够以源代码的形式将软件复制或发布给别的用户)
- ◆ 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

- ◆ “copyleft”:  
Copyright Copyleft



End of Chapter 1

