Introduction
— Chapter 1

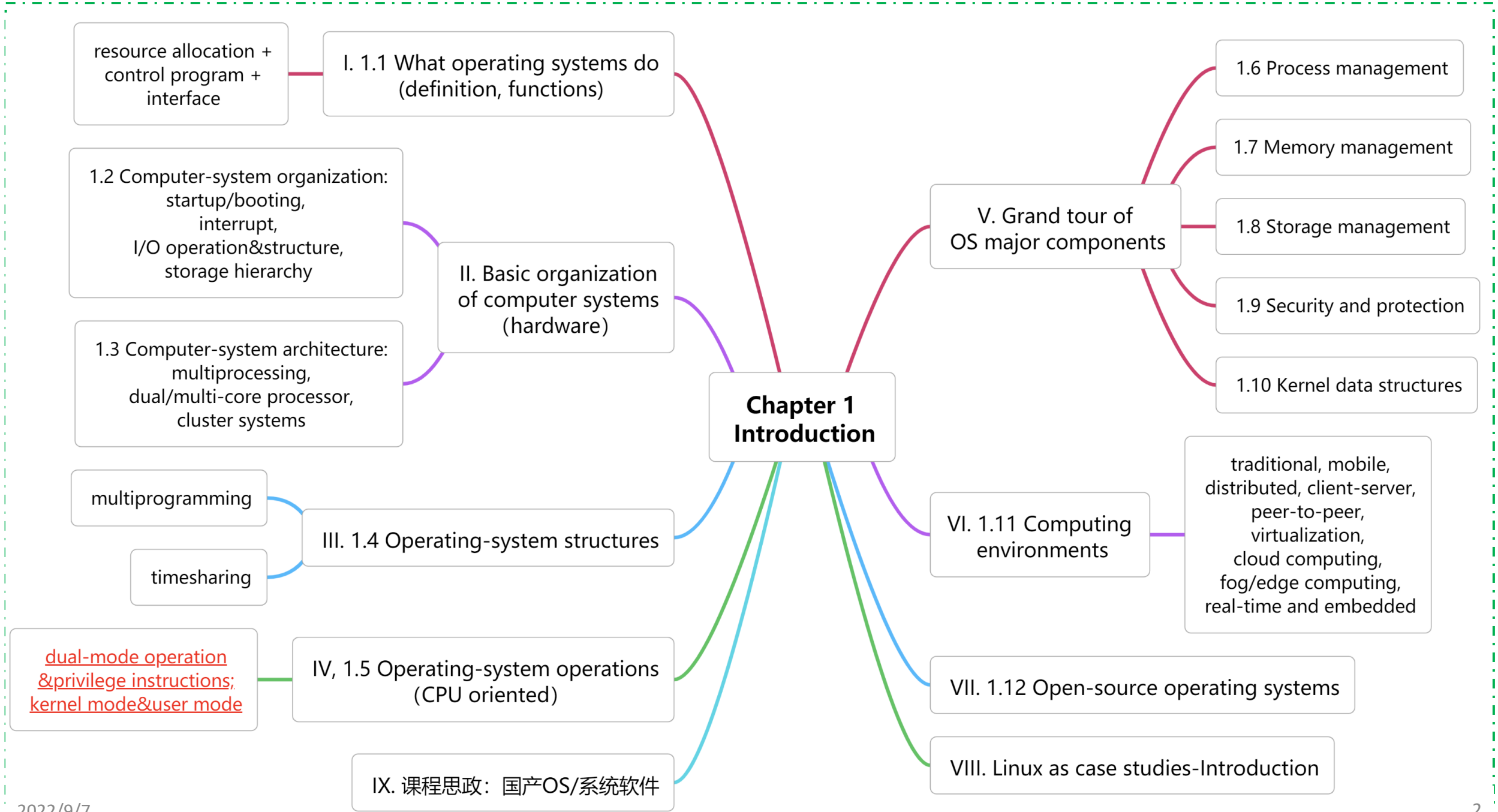
2022年9月

薛哲

xuezhe@bupt.edu.cn

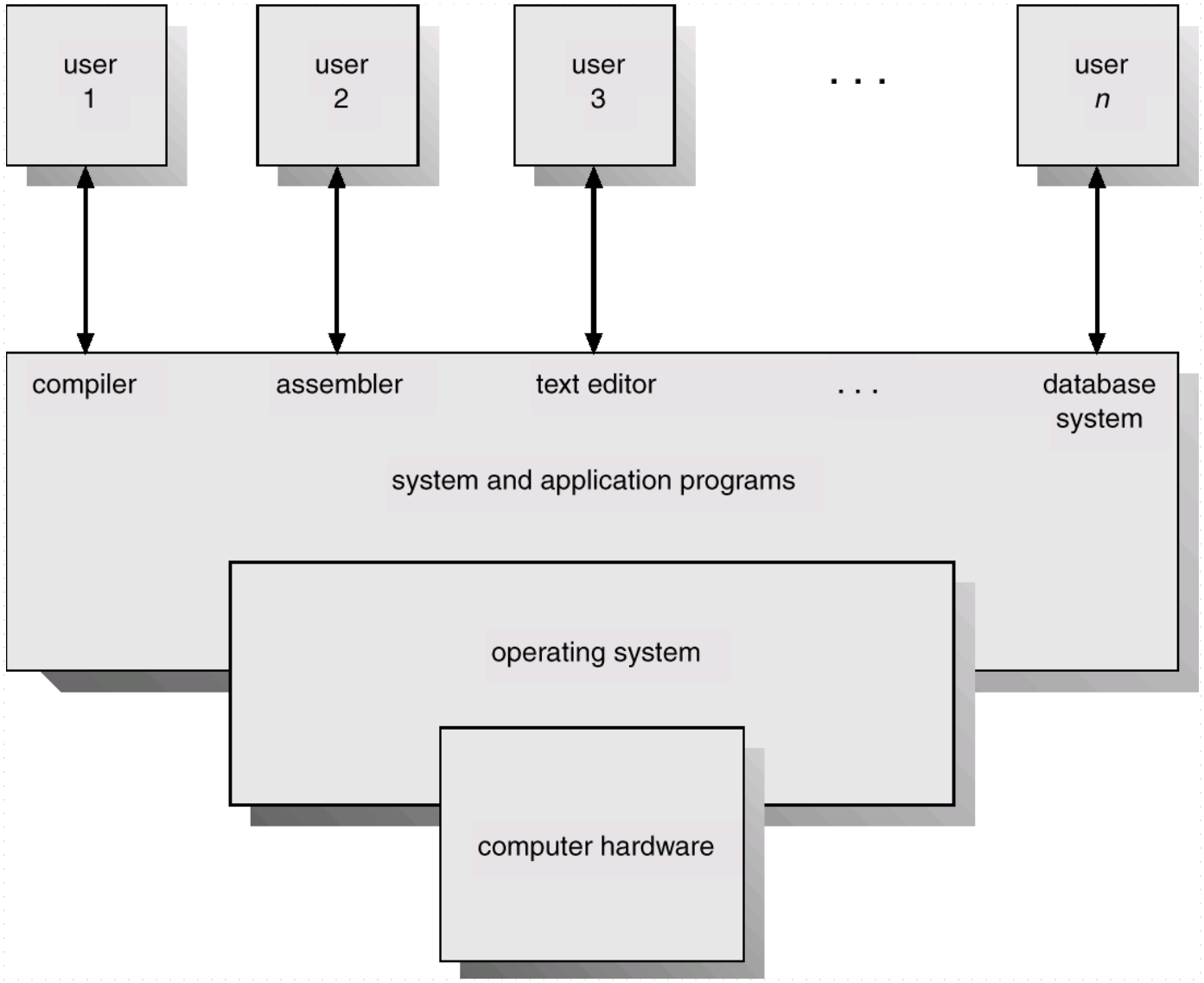
School of Computer Science (National Pilot Software Engineering School)

北京郵電大學


Chapter 1
Introduction
I. 1.1 What operating systems do
(definition, functions)
resource allocation +
control program +
interface
II. Basic organization
of computer systems
(hardware)
1.2 Computer-system organization:
startup/booting,
interrupt,
I/O operation&structure,
storage hierarchy
1.3 Computer-system architecture:
multiprocessing,
dual/multi-core processor,
cluster systems
III. 1.4 Operating-system structures
multiprogramming
timesharing
IV, 1.5 Operating-system operations
(CPU oriented)
dual-mode operation
&privilege instructions;
kernel mode&user mode
IX. 课程思政：国产OS/系统软件
V. Grand tour of
OS major components
1.6 Process management
1.7 Memory management
1.8 Storage management
1.9 Security and protection
1.10 Kernel data structures
VI. 1.11 Computing
environments
traditional, mobile,
distributed, client-server,
peer-to-peer,
virtualization,
cloud computing,
fog/edge computing,
real-time and embedded
VII. 1.12 Open-source operating systems
VIII. Linux as case studies-Introduction

1.1 What Operating Systems Do

- What is a operating system
  - a program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals
  - execute user programs and make solving user problems easier
  - make 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

What Operating Systems Do

- Depends on the point of view
  - user view, system view
- User view
  - Interface, users want convenience, **ease of use** and **good performance**
- 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
- System view
  - resource allocation
  - efficient management of hardware (CPU, memory, I/O devices, storage) and software (program, data) resources in computer
    - maximization of utilization

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
- OS provides **interface** between computer hardware and other system and application programs

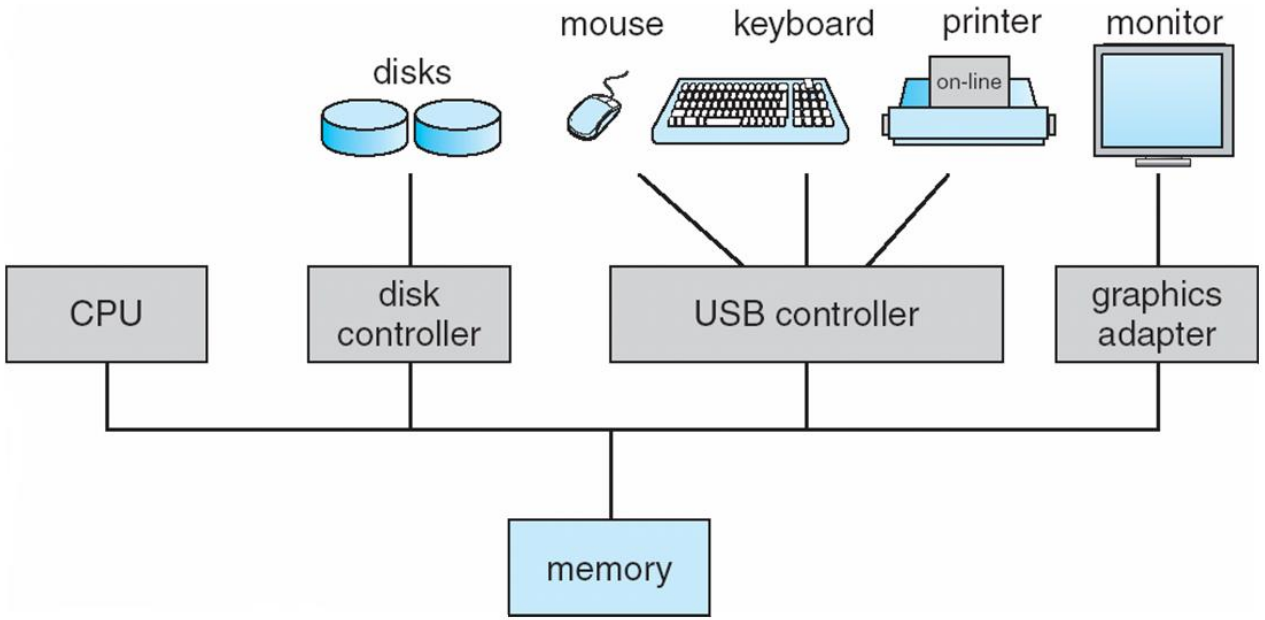
- No universally accepted definition
- OS = **kernel** + system programs
  - ″The one program running at all times on the computer″ is the **kernel**
  - everything else is either
    - a system program (ships with the operating system) , or
    - an application program.

Operating System Definition

- Summary
  - OS is a type of system software, residing between computer hardware and the other software or users in the system
  - OS provides the interfaces for the users and application programs to use computer software and hardware resources **conveniently**
  - it also **efficiently** manages and controls hardware resources, such as CPU, memory, and I/O devices, to maximize the utilization of these resources; also manage information (e.g. programs, data) resources

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

Computer System Organization

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

Common Functions of Interrupts

- An operating system is **interrupt driven**

- Interrupt transfers control to the interrupt service routine（中断服务程序） generally, through the **interrupt vector**, which contains the addresses of all the service routines
  - Interrupt architecture must save the address of the interrupted instruction
  - OS preserves the state of the CPU by storing registers and the program counter

- Separate segments of code determine what action should be taken for each type of interrupt

Interrupt

- **Interrupts in computers**
  - when some events occur, the programs currently running on the CPU are interrupted, CPU control is transferred to the *interrupt service routine* to handle the events
  - interrupt mechanism: interrupt enabling/disabling, interrupt vector, interrupt priority
- Fig. 1.3 Interrupt time line for a single process doing input, e.g. *disk reading*
  - (1) user program runs on CPU, and I/O device is idle
  - (2) user program prepares memory buffer, send a I/O request to I/O device by I/O API (e.g. *read*) and drivers, ready to doing input
  - (3) I/O device responds to the request, begins to transfer data, I/O device and the CPU execute in parallel
  - (4) after it has finished its I/O operations, I/O device informs CPU by causing an I/O ***interrupt,*** and CPU switches to *interrupt service routine* (i.e. *drivers*)to fetch the data from disk; I/O device transits into the idle state
  - (5) after I/O interrupt processing is over, CPU returns to user program

Interrupt

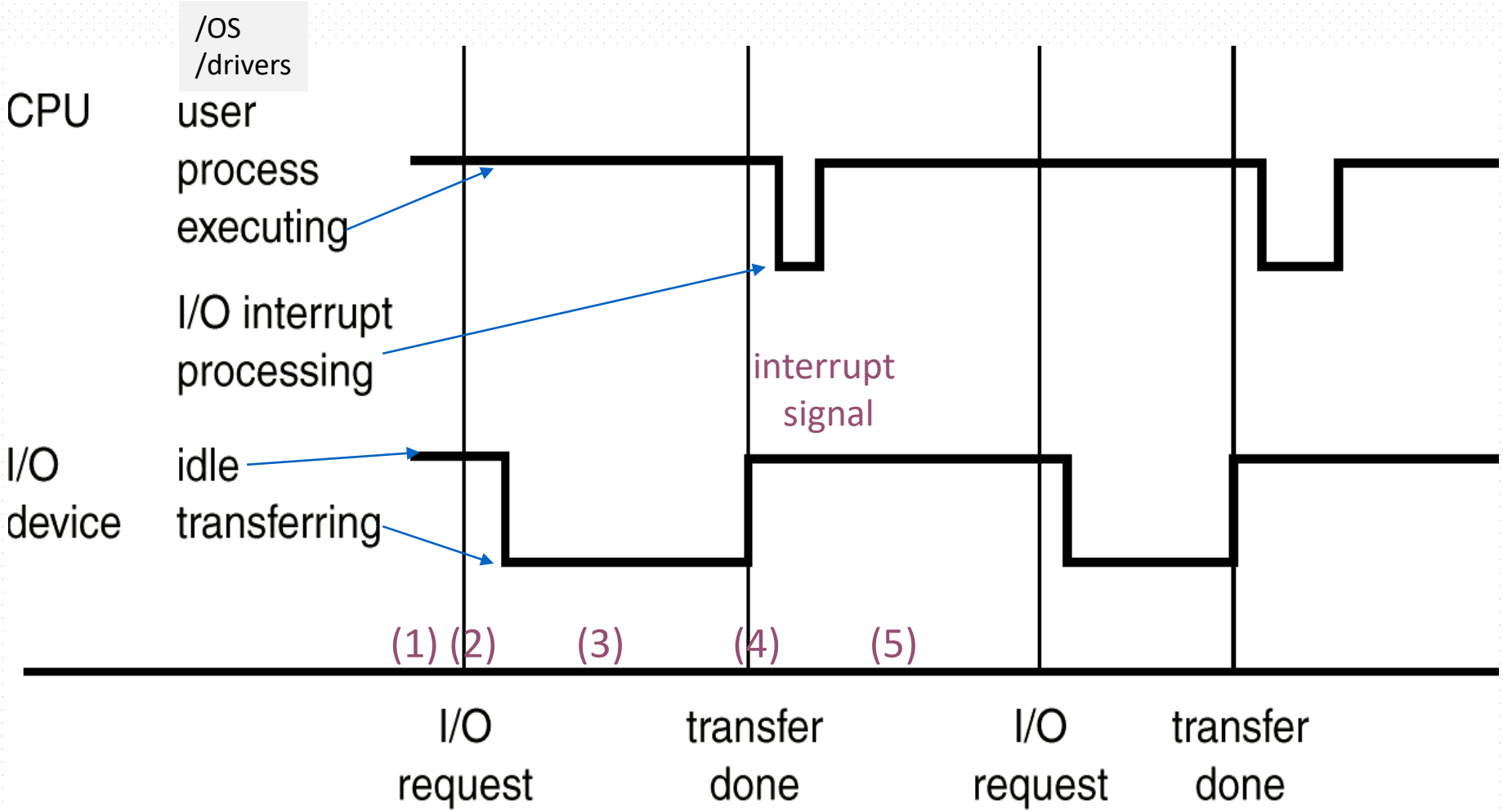
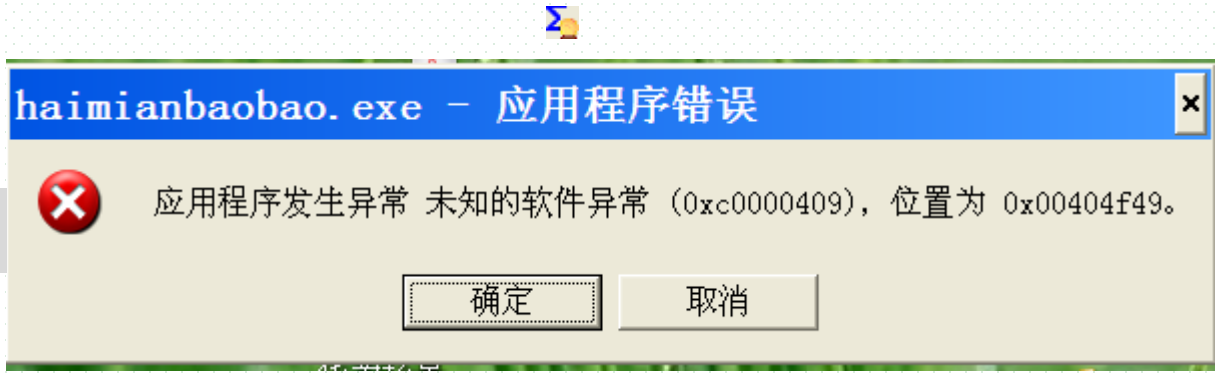


Fig.1.3 Interrupt time line for a *single* process doing output
e.g. input data from disk

Interrupt

- Three types of interrupts
- Hard interrupt
  - caused by hardware interrupt signals, trigged by external events
- System call/monitor call
  - an interrupt triggered by software
  - from user request program that a OS service be performed, e.g. **system calls** such as *read*
- Trap(陷阱) or exception(异常)
  - software-generated interrupt caused e by an error

e.g. 2+3/0



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).

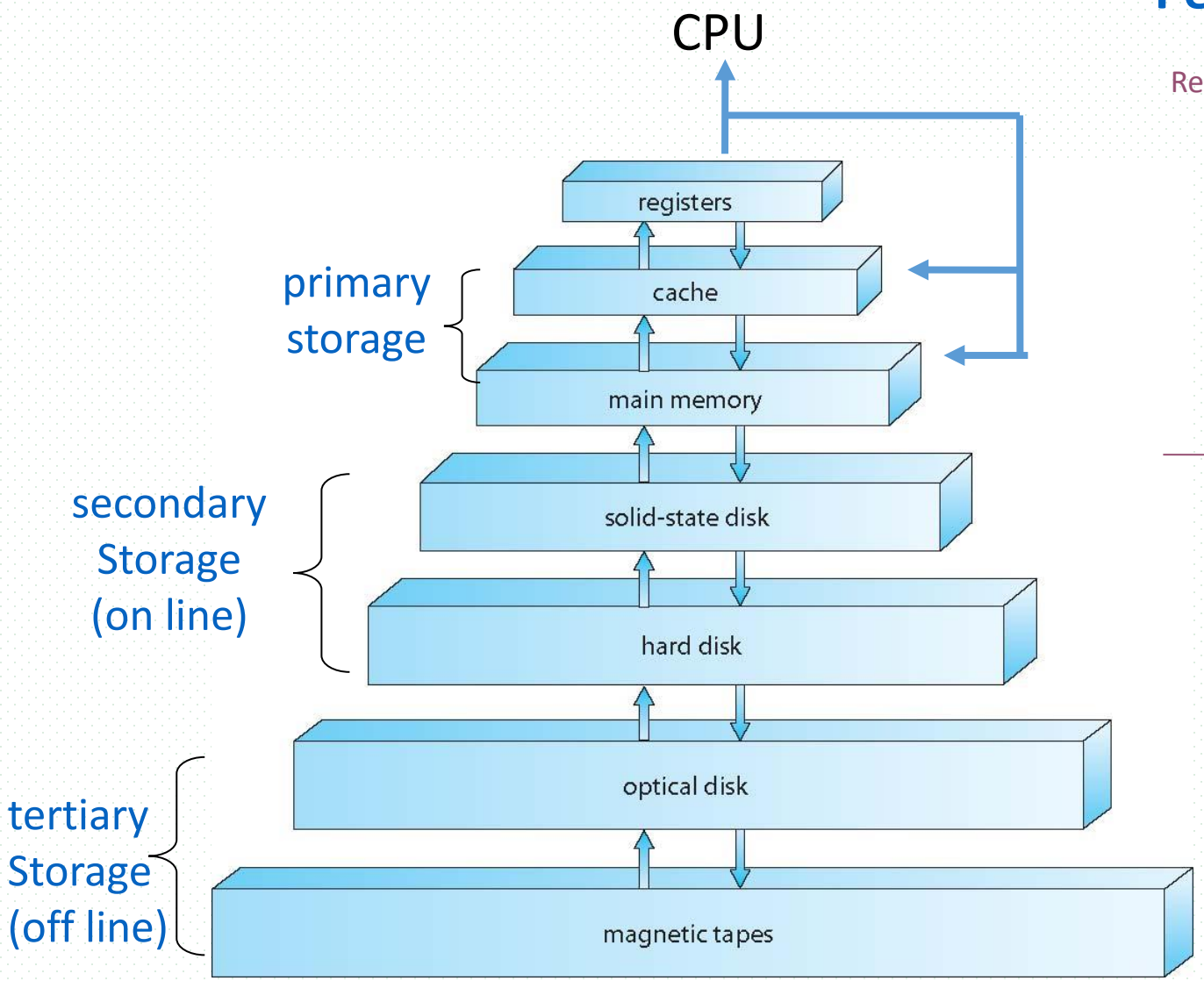
Storage Structure

- 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 Structure

- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility
- **Caching**（高速缓存）– copying information into faster storage system; main memory can be viewed as a cache for secondary storage
- **Device Driver**（设备驱动程序） for each device controller to manage I/O
  - Provides uniform interface between controller and kernel

Storage-Device Hierarchy



Performance Indicators

Reliability	Speed	Capacity	Cost /per unit of memory	Access Mode
volatile	fast	small	high	byte-based
nonvolatile	slow	large	low	block-based

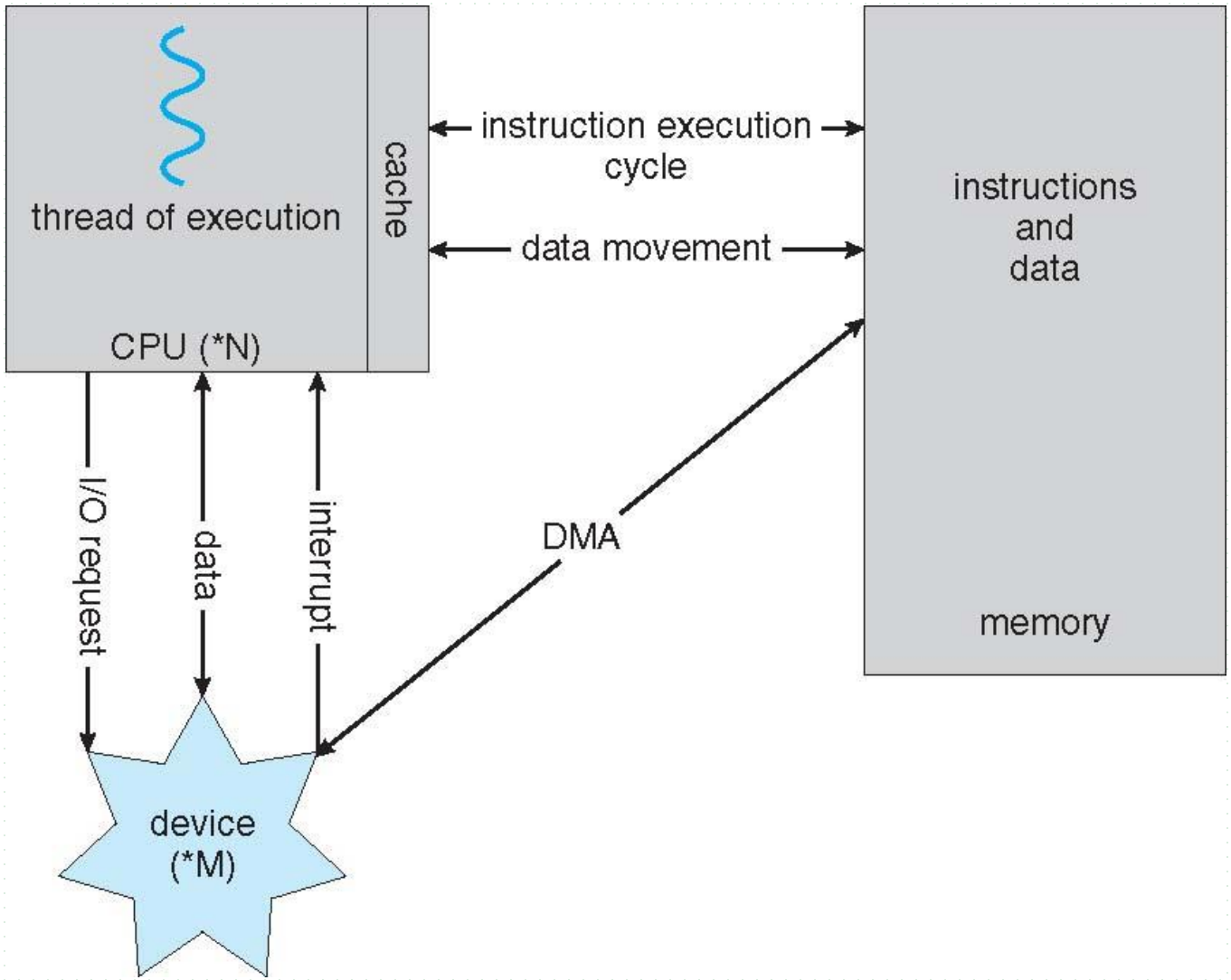
Network Storage Systems,

e.g. Network-attached storage (NAS), Storage Area network(SAN)

Caching

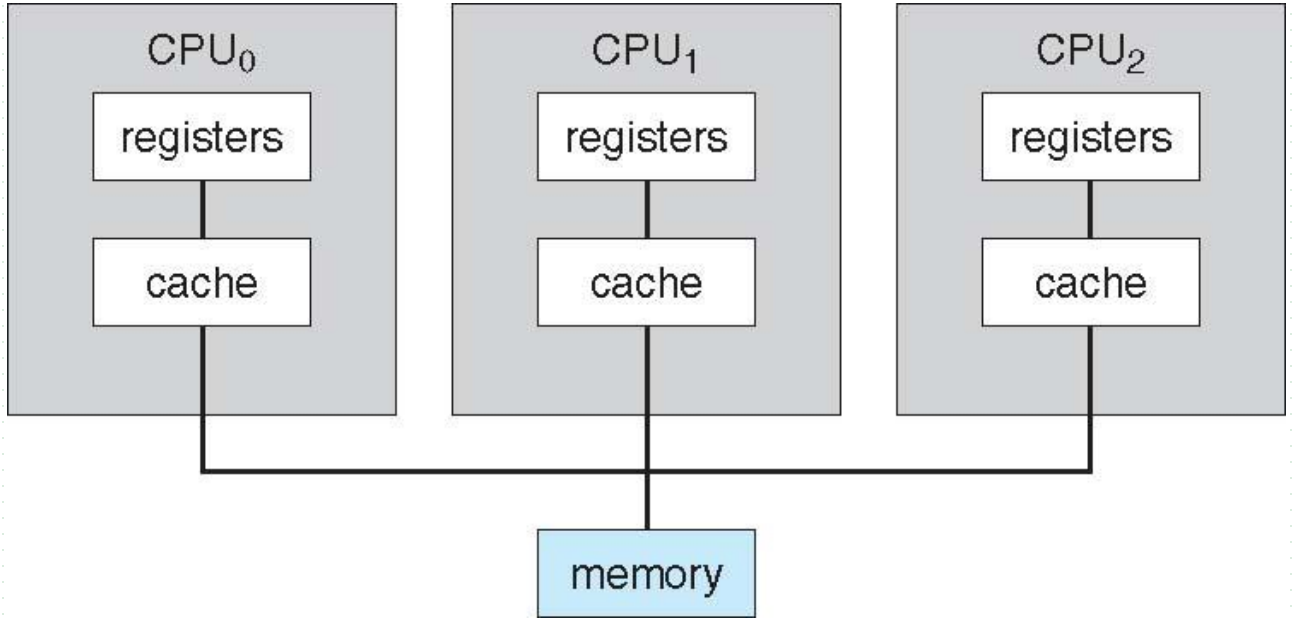
- Caching
  - Information in use copied from slower to faster storage temporarily
- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly 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



1.3 Computer-System Architecture

- Most systems use a single general-purpose processor
- Multiprocessors systems growing in use and importance
  - Also known as parallel systems, tightly-coupled systems
  - Advantages include:
    - Increased throughput
    - Economy of scale
    - Increased reliability
    - – graceful degradation or fault tolerance



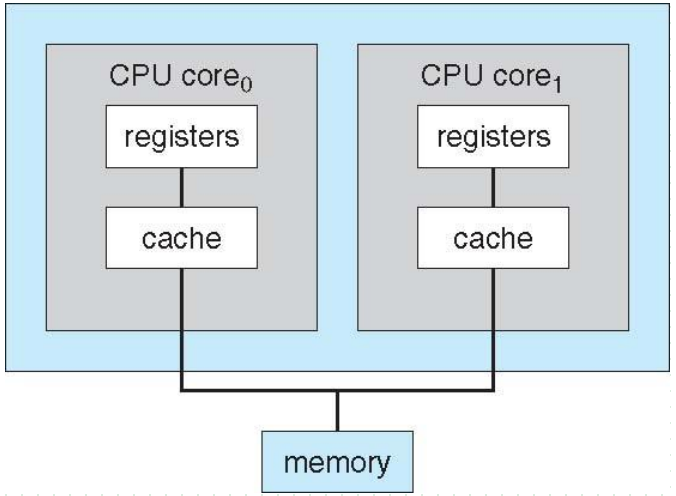
  - Two types
    - asymmetric Multiprocessing – each processor is assigned a specie task
    - symmetric Multiprocessing – each processor performs all tasks

Dual-Core/Multi-Core Design

- 双核/多核/众核CPU
- Multi-chip and **multicore**

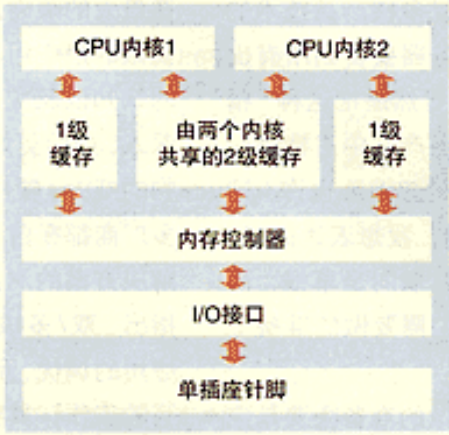
- 神威太湖之光高性能计算机
  - 申威26010众核处理器
  - 25平方厘米，260个运算核心
  - 每秒3万+亿次计算能力

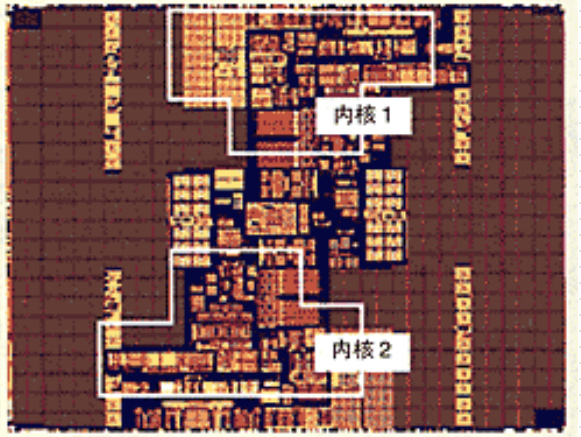
- 飞腾64核CPU芯片
  - 64 核通用CPU，ARM 架构
  - 浮点运算的峰值速度5120 亿次/秒，
  - 用于天河超级计算机管理结点



多内核架构

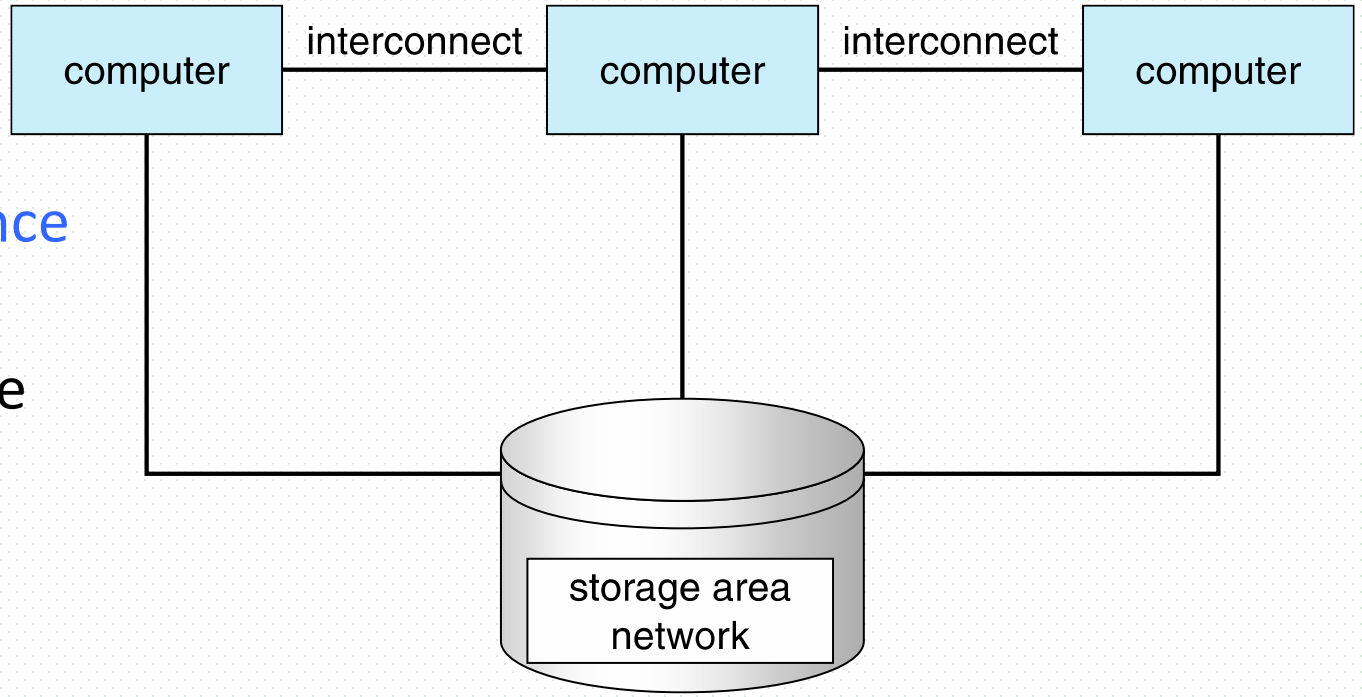
目前的通用多内核处理器设计都是基于对称多处理（SMP）技术的。SMP是一种并行架构，在这种架构中，所有的处理器运行一个操作系统副本，共享内存和一台计算机的其他资源，并且平等地访问内存、I/O及外部中断。大多数SMP架构是由下图演化而来的：





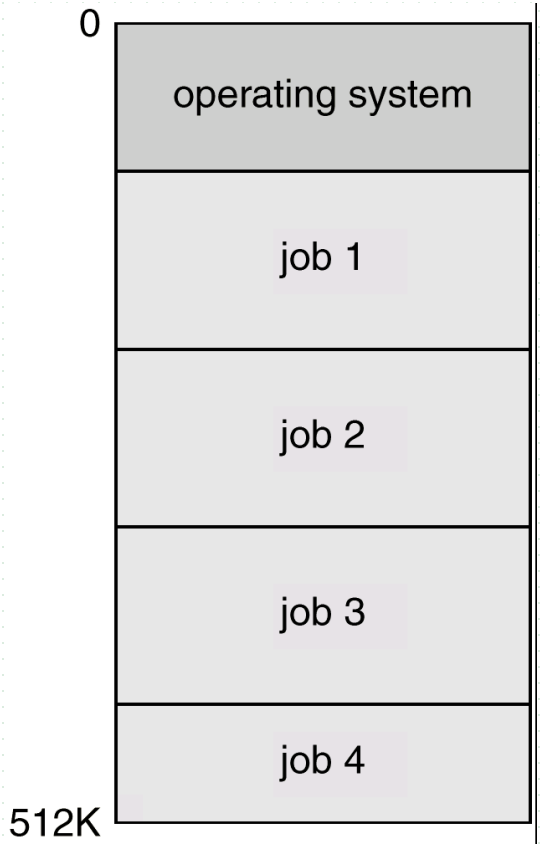
Clustered Systems

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



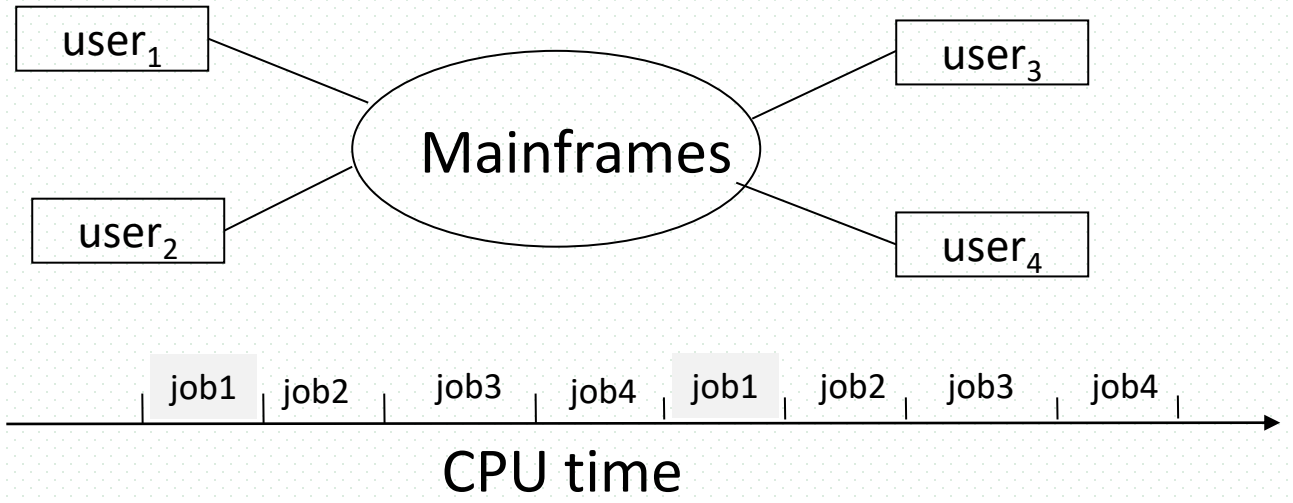
1.4 Operating System Structure

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



Operating System Structure

- Timesharing (multitasking) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing
  - CPU times are divided into timeslot (时间片)，each user/process/job/task occupies one timeslot alternatively
  - 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



1.5 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

Dual-mode operation & Privilege instructions

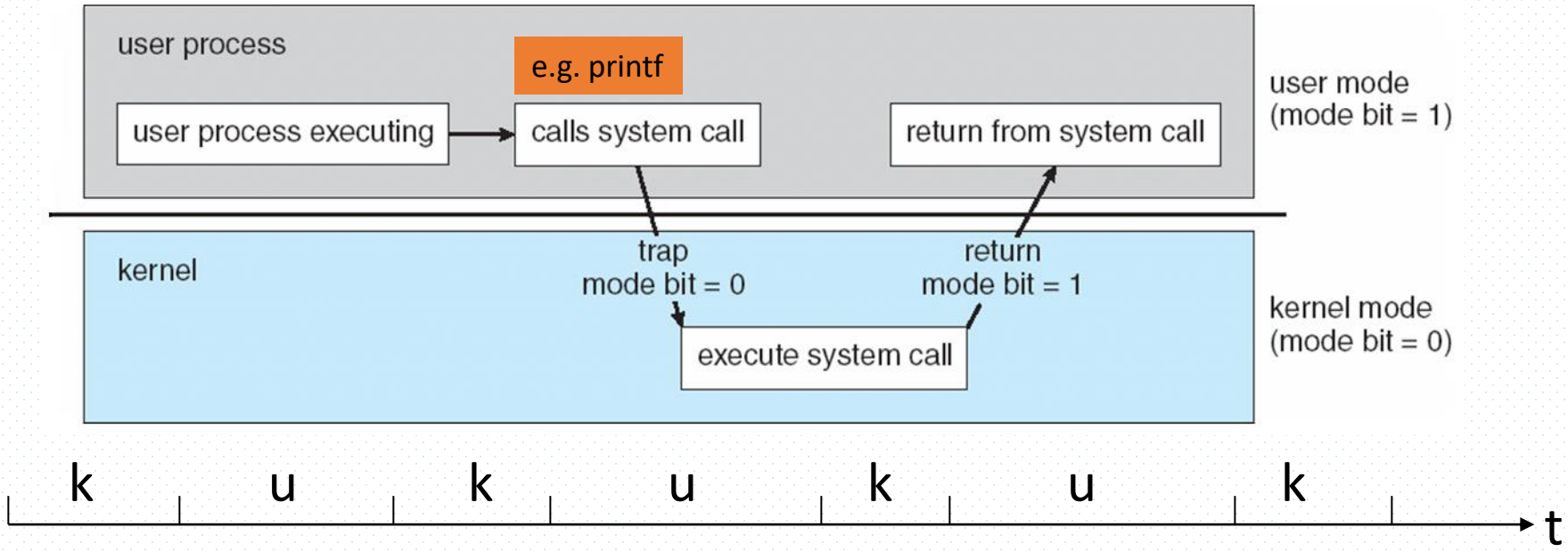
- Goal
  - protect OS and other system components from errant users (i.e., programs), or errant users from one another
- Privilege instruction
  - identify machine instructions that may cause harm, and designate these instructions as privileged instructions (特权指令，权限指令), i.e. *dangerous* instructions, e.g.
    - privileged "set value of timers",
    - non-privileged "read the clock"
- Dual-mode: User mode and kernel mode
  - mode bit provided by hardware/CPU to indicate the current operation mode
    - 0: kernel/monitor/supervisor/system/privileged mode, in which OS and privileged instructions are running on CPU
    - 1: user mode, in which user programs are executing on CPU
  - privileged instructions are only executable in kernel mode

Dual-mode operation & Privilege instructions

- Dual-mode scheme provides ability to distinguish when system is running user code or kernel code
- If user programs in user mode attempt to execute privileged instructions, the hardware does not execute these instructions, but rather treats them as illegal operations
  - a trap is generated and CPU control is passed to OS to handle this errors
    - /* 避免用户程序执行有可能导致错误的权限指令（危险指令）

Transition from User to Kernel Mode

- At system boot time, CPU starts in monitor mode. After OS is loaded into memory and begin to work in normal states, CPU switches into user mode to run user programs
- System call changes mode to kernel, return from call resets it to user



Timer

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

Dual-mode operation & Privilege instructions

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

OS Management

- 1.6 Process management
- 1.7 Memory Management
- 1.8 Storage Management
- 1.9 Protection and Security

1.6 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（复用） 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

1.7 Memory Management

- To execute a program all (or part) of the instructions must be in memory
- All (or part) of the data that is needed by the program must be in memory.
- Memory management determines what is in memory and when
  - Optimizing CPU utilization and computer response to users
- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

1.8 Storage Management: File Systems

- 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
- 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)

Caching: speeding up data access

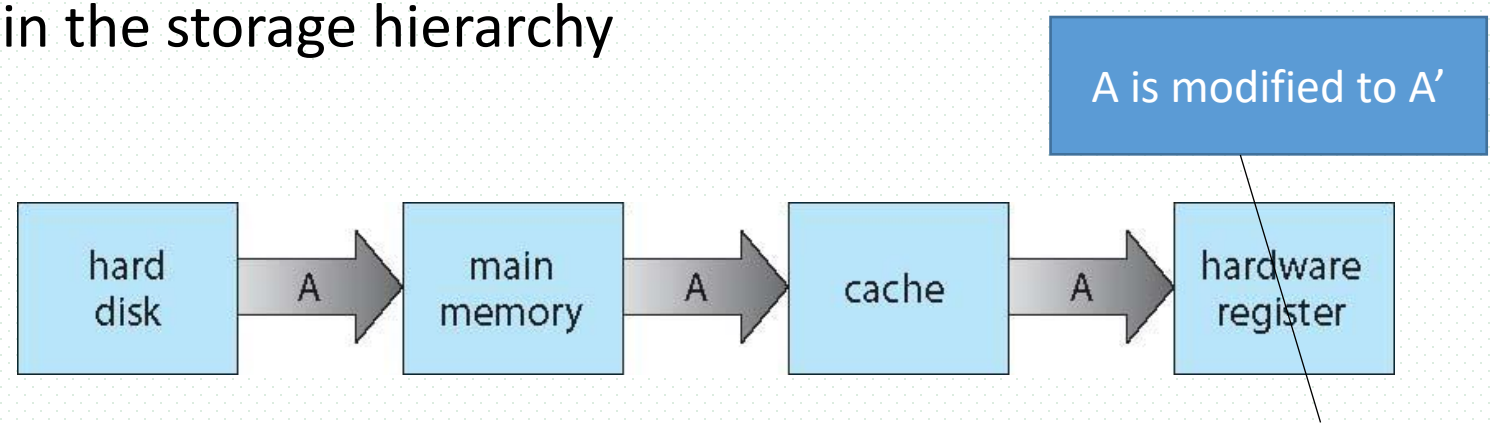
Performance of Various Levels of Storage

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

I/O Subsystem

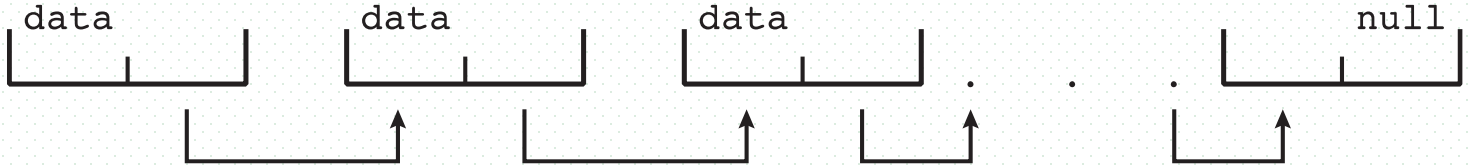
- One purpose of OS is to hide peculiarities（特性） of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the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

1.9 Protection and Security

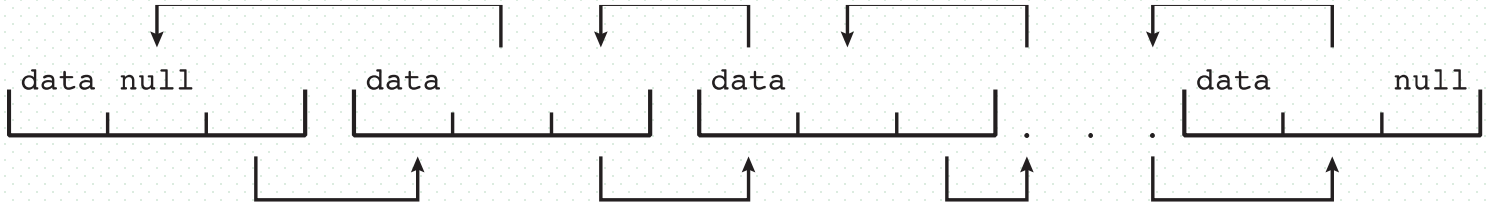
- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first 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

1.10 Kernel Data Structures【略】

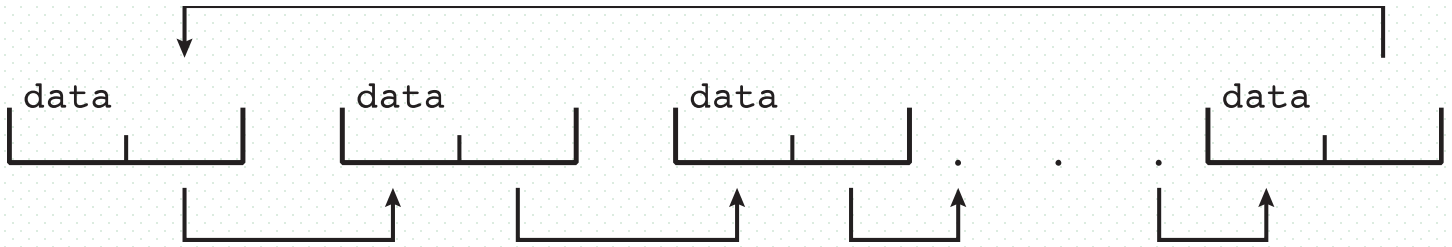
- Many similar to standard programming data structures
- ***Singly linked list***



- ***Doubly linked list***



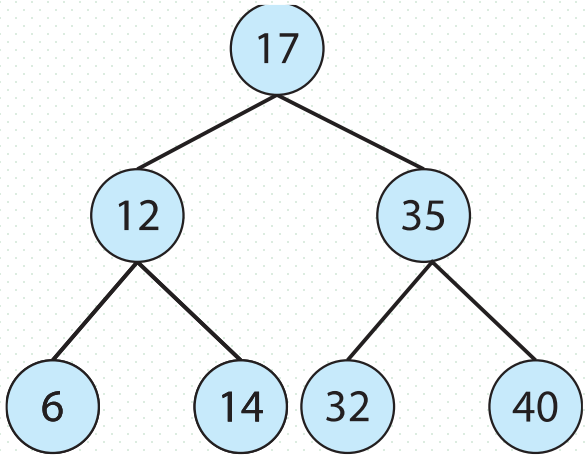
- ***Circular linked list***



- **Linked List Usage**
  - scheduling queue

Kernel Data Structures

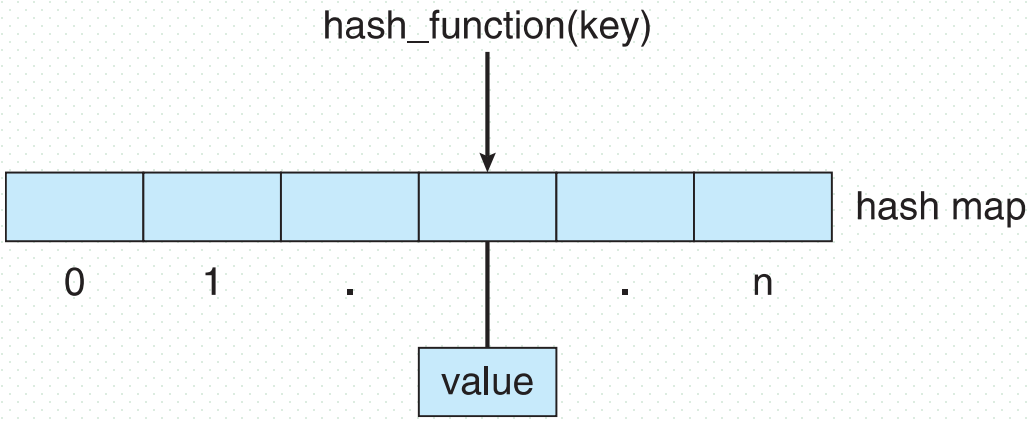
- **Binary search tree**
  left <= right
  - Search performance is *O(n)*
  - **Balanced binary search tree** is *O(lg n)*



- **红黑树rbtree**
  - rbtree.h/rbtree.c in Linux 3.0kernel
  - usage
    - CPU调度时，组织running task(调度对象)
    - 高精度定时器timer组织定时请求
    - EXT3文件系统管理目录
    - 虚拟存储管理系统管理内存空间块VMAs (Virtual Memory Areas)

Kernel Data Structures

- **Hash function** can create a **hash map**



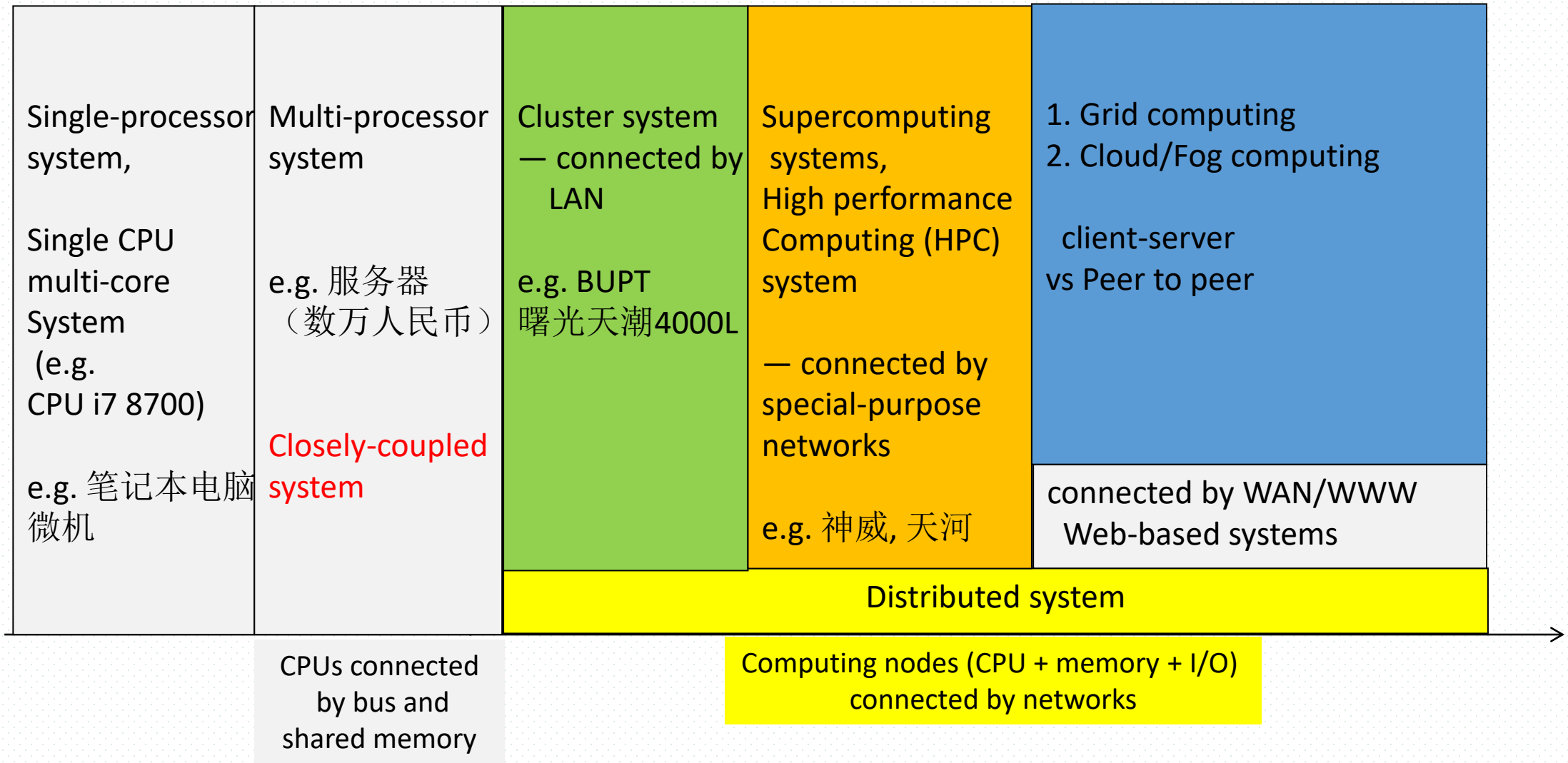
- **Bitmap** – string of *n* binary digits representing the status of *n* items
- Linux data structures defined in

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

1.11 Computing Environments

- Traditional
- Mobile
- Distributed
- Client-Server
- Peer-to-Peer
- Virtualization
- Cloud Computing
- Fog/Edge Computing
- Real-Time Embedded Systems

Evolution of computer system structures



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***
  - ***AR***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**

Computing Environments – Distributed

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

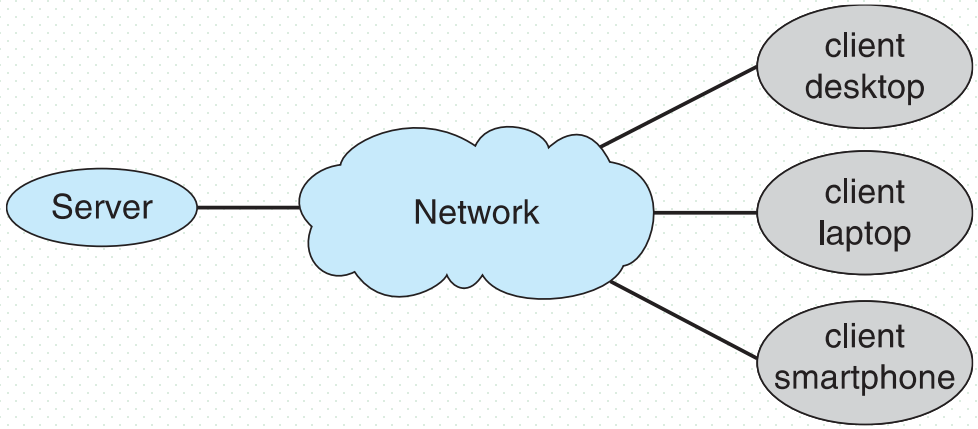
Computing Environments – Client-Server

Client-Server Computing

Dumb terminals supplanted by smart PCs

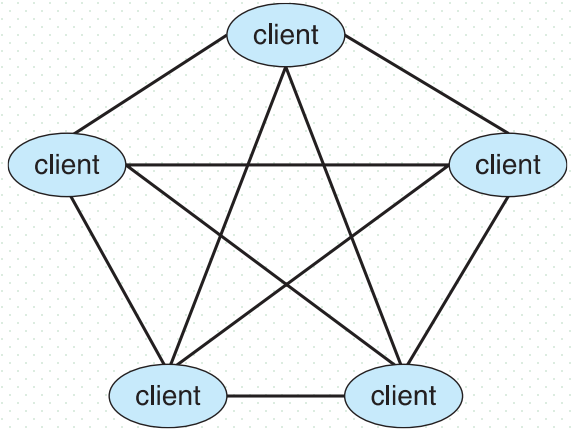
Many systems now **servers**, responding to requests generated by **clients**

- **Compute-server system** provides an interface to client to request services (i.e., database)
- **File-server system** provides interface for clients to store and retrieve files



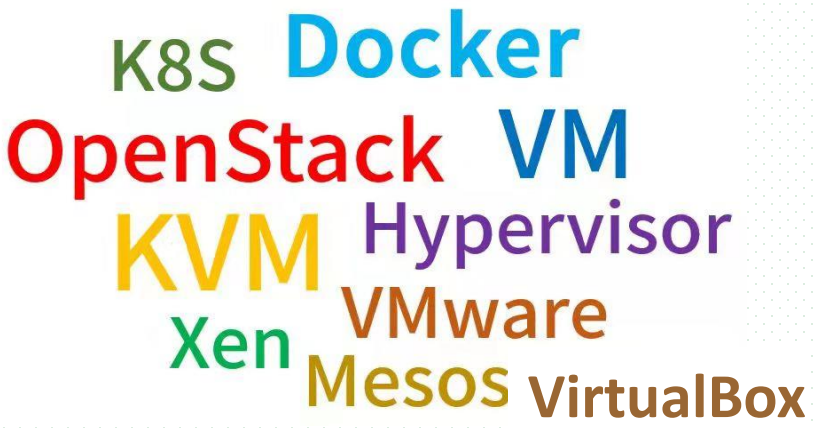
Computing Environments - Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both
  - Node must join P2P network
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via ***discovery protocol***
  - Examples include Napster and Gnutella, **Voice over IP** (**VoIP**) such as Skype

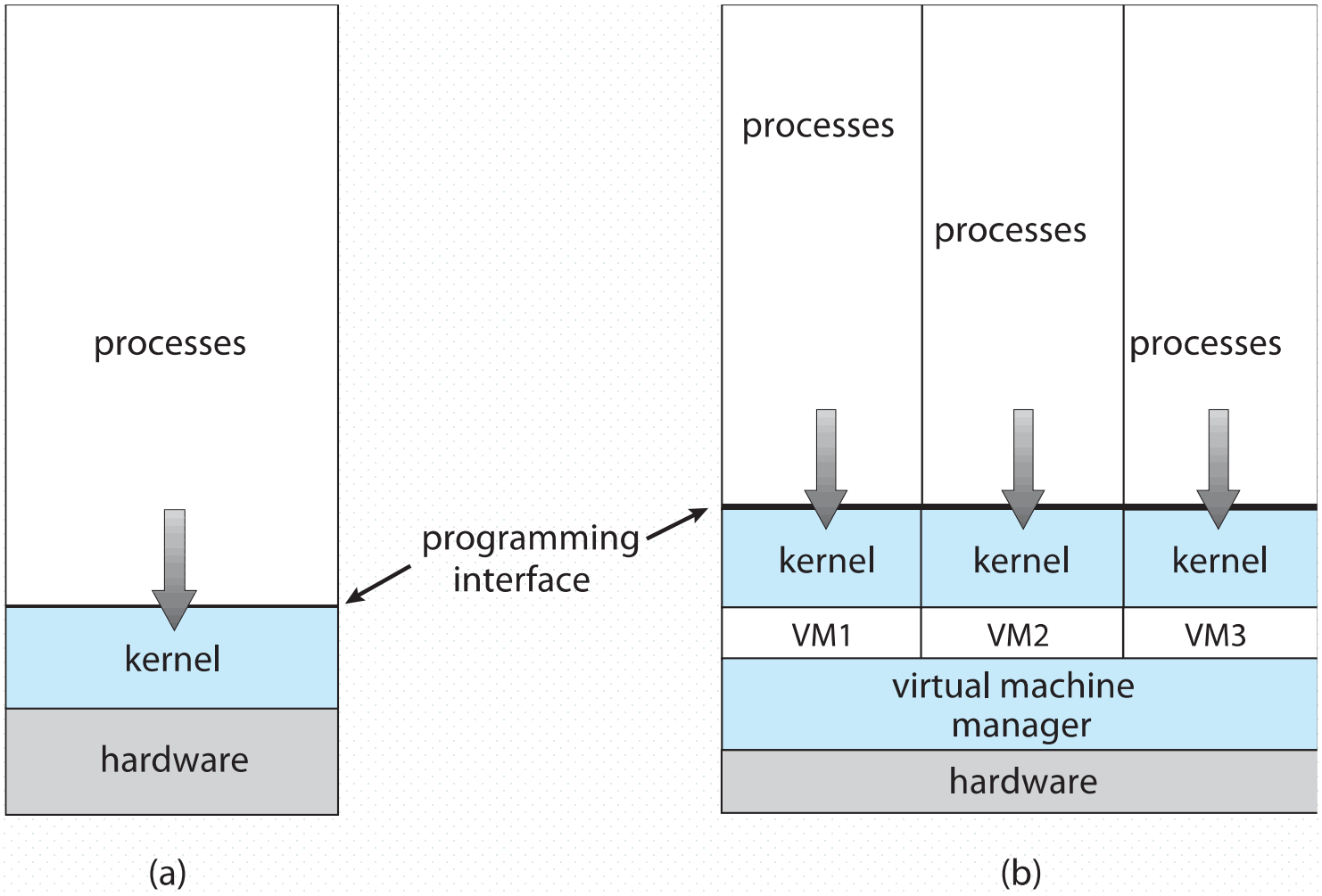


Computing Environments - Virtualization

- Allows operating systems to run applications within other 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
- Virtual Machine/Container
  - 虚拟机/容器



Computing Environments - Virtualization



Computing Environments - Virtualization

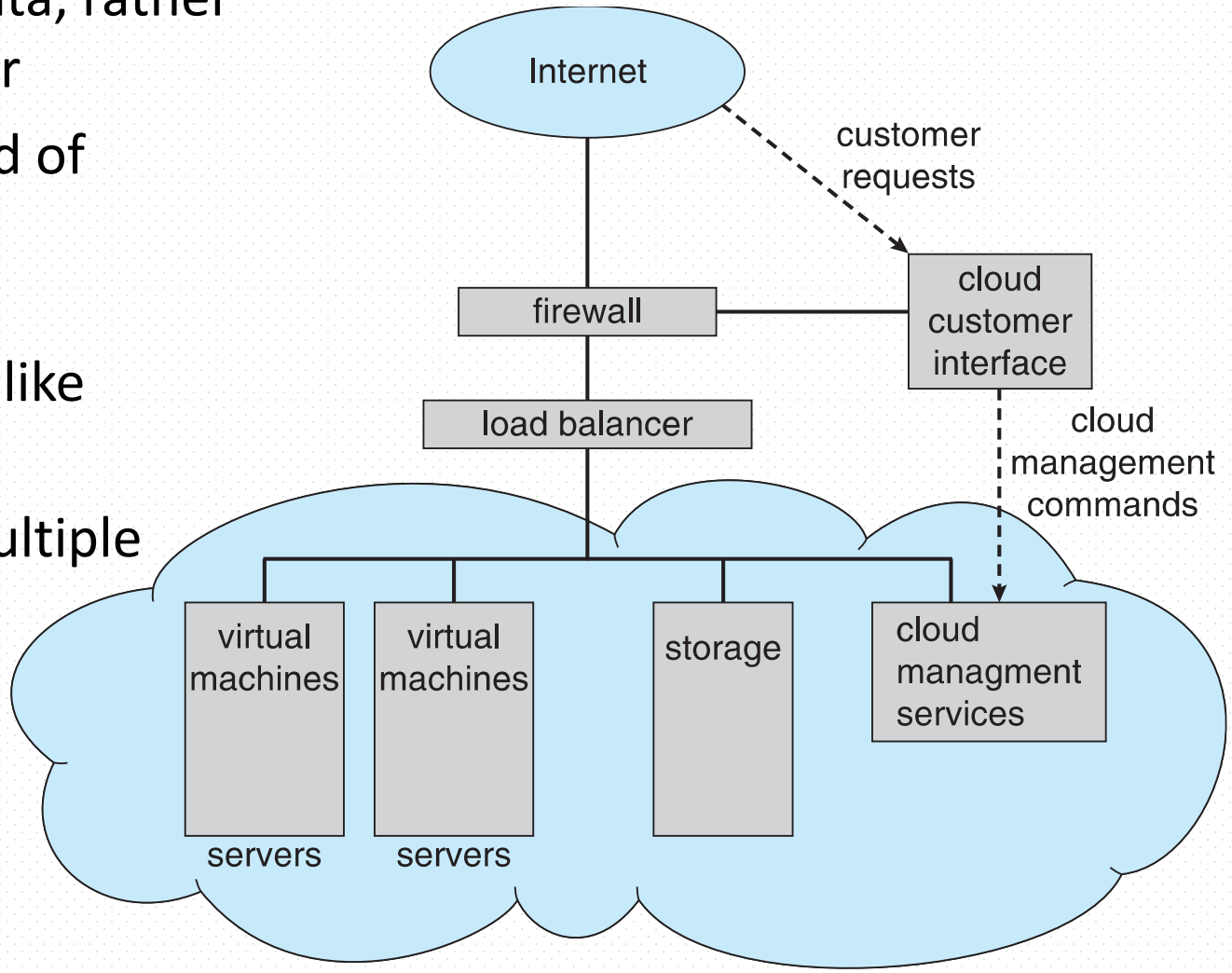
- Use cases involve laptops and desktops running multiple OSes for exploration or compatibility
  - Apple laptop running Mac OS X host, Windows as a guest
  - Developing apps for multiple OSes without having multiple systems
  - QA testing applications without having multiple systems
  - Executing and 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 – 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

- Using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer
- 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 – Fog/Edge Computing

- Also called Edge Computing
- Fog computing is intended for distributed computing where numerous "peripheral" devices connect to a cloud. (The word "fog" suggests a cloud's periphery or edge). Many of these devices will generate voluminous raw data (e.g., from sensors), and rather than forward all this data to cloud-based servers to be processed, the idea behind fog computing is to do as much processing as possible using computing units co-located with the data-generating devices, so that processed rather than raw data is forwarded, and bandwidth requirements are reduced
- An additional benefit is that the processed data is most likely to be needed by the same devices that generated the data, so that by processing locally rather than remotely, the latency between input and response is minimized. This idea is not entirely new: in non-cloud-computing scenarios, special-purpose hardware (e.g., signal-processing chips performing Fast Fourier Transforms) has long been used to reduce latency and reduce the burden on a CPU

Computing Environments – Real-Time Embedded Systems

- Real time systems
  - are computers and/or software systems that react to external events in limited time intervals before the events become obsolete /*对外部事件作出及时响应
  - dedicated application oriented computer systems, in which rigid time requirements are placed on the operations of processors or the flow of data
  - e.g. *temperature* and *pressures* alarm processing in fault management of chemical industry production line
- Well-defined fixed-time constraints
  - (task) ***deadline,*** 时限，截止期
- Embedded system
  - mini or micro computer systems, embedded in application devices, often used in a dedicated application
- Another definition *by IEE*
  - the facilities used for monitoring and controlling large-scale systems, such as machineries, devices, equipments, and factories

Real-Time Systems

- Real-Time systems may be either *hard* or *soft* real-time
- Hard real-time
  - guarantee that critical tasks in system be completed *on time*
  - secondary storage limited or absent, data stored in short term memory, or read-only memory (ROM)
- Soft real-time
  - critical task gets priority over other tasks to be handled as soon as possible
  - no deadline constraints, e.g. calling processing in telecommunications switching
  - useful in applications (multimedia, virtual reality) requiring advanced operating-system features
- OS features for real time systems
  - real-time task scheduling: chapter6
  - memory management: chapter10

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
    - depending on eal-time CPU scheduling algorithms
  - Correct operation only if constraints met

1.12 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

附录 1.1

- 1. 大国隐痛：做一个操作系统有多难?
  - https://new.qq.com/omn/20200716/20200716A0YG0M00.html

- 1

附录1.1 大国隐痛：做一个操

Thanks for your attention

北京郵電大學