東南大學
SOUTHEAST UNIVERSITY

OPERATING SYSTEM CONCEPTS

Chapter 1. Introduction

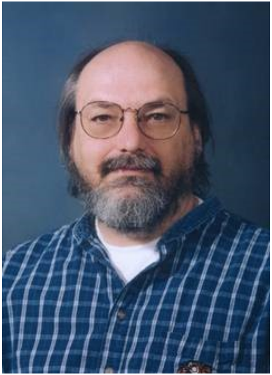
A/Prof. Kai Dong

Warm-up

OS Archaeology

Ken Thompson & Dennis Ritchie

- Multics ⟶ AT&T Unix ⟶ BSD Unix ⟶ Ultrix, SunOS, NetBSD, . . .



Warm-up

OS Archaeology (contd.)

Linus Torvalds

- Linux → Android OS, Chrome OS
- Linux → RedHat, Ubuntu, Fedora, Debian, Suse, . . .

Warm-up

OS Archaeology (contd.)

Steve Jobs

- Mach + BSD ⟶ NextStep ⟶ XNU ⟶ Apple OSX, iphone iOS

Warm-up

OS Archaeology (contd.)

Bill Gates

- CP/M ⟶ QDOS ⟶ MS-DOS ⟶ Windows 3.1 ⟶ NT ⟶ 95 ⟶ 98 ⟶ 2000 ⟶ XP ⟶ Vista ⟶ 7 ⟶ 8 ⟶ 10 ⟶ phone, ...

Warm-up

Discussion

- What Operating System(s) are you familiar with?
- What is the best Operating System in your opinion? And for what features?

Objectives

- To describe the basic organization of computer systems.
- To explain the evolution of operating system structures.
- 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.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

What Operating Systems Do

What is an Operating System?

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

What Operating Systems Do

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:

What Operating Systems Do

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.

What Operating Systems Do

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.

What Operating Systems Do

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.
  - Use the computer hardware in an efficient manner.

What Operating Systems Do

Computer System Structure

Computer system can be divided into four components:

What Operating Systems Do

Computer System Structure

Computer system can be divided into four components:

- **Hardware** — provides basic computing resources.
  - CPU, memory, I/O devices

What Operating Systems Do

Computer System Structure

Computer system can be divided into four components:

- **Hardware** — provides basic computing resources.
  - CPU, memory, I/O devices
- **Users**
  - People, machines, other computers

What Operating Systems Do

Computer System Structure

Computer system can be divided into four components:

- **Hardware** — provides basic computing resources.
  - CPU, memory, I/O devices
- **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

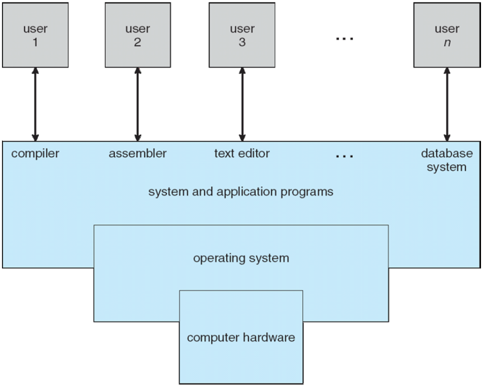
Computer System Structure

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

Computer System Structure (contd.)



- We can explore operating systems from two viewpoints: that of the user and that of the system.

What Operating Systems Do

User View

- Want convenience, ease of use and good performance.
- Don't care about resource utilization.
  - But *shared computer* such as *mainframe* or *minicomputer* must keep all users happy.
  - Users of dedicate systems such as *workstations* have dedicated resources but frequently use shared resources from *servers*.
  - Handheld computers are resource poor, optimized for usability and battery life.
  - Some computers have little or no user interface, such as embedded computers in devices and automobiles.

What Operating Systems Do

System View

- OS is a resource allocator.

What Operating Systems Do

System View

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

What Operating Systems Do

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.

What Operating Systems Do

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.

What Operating Systems Do

Defining Operating Systems

- No universally accepted definition.

What Operating Systems Do

Defining Operating Systems

- No universally accepted definition.
- "Everything a vendor ships when you order an operating system"
  - is a good approximation,
  - but varies wildly.

What Operating Systems Do

Defining Operating Systems

- 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"
  - defines the kernel,
  - everything else is either
    - a system program (ships with the operating system) , or
    - an application program.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Computer-System Organization

Computer Startup

- Bootstrap program is loaded at power-up or reboot.

Computer-System Organization

Computer Startup

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

Computer-System Organization

Computer 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.
- Think of:
  - What happens when a program runs / How instructions are executed?

Computer-System Organization

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.
- Think of:
  - What happens when a program runs / How instructions are executed?
  - Millions/billions of times a second,
    - the processor fetches an instruction from memory,
    - decodes it,
    - executes it,
    - moves on to the next till end.

Computer-System Organization

Von Neumann Model

- We have just described the basics of the Von Neumann model of computing.
- "… an electronic digital computer with parts consisting of

Computer-System Organization

Von Neumann Model

- We have just described the basics of the Von Neumann model of computing.
- "… an electronic digital computer with parts consisting of
  - a processing unit containing an arithmetic logic unit and processor registers;

Computer-System Organization

Von Neumann Model

- We have just described the basics of the Von Neumann model of computing.
- "… an electronic digital computer with parts consisting of
  - a processing unit containing an arithmetic logic unit and processor registers;
  - a control unit containing an instruction register and program counter;

Computer-System Organization

Von Neumann Model

- We have just described the basics of the Von Neumann model of computing.
- "… an electronic digital computer with parts consisting of
  - a processing unit containing an arithmetic logic unit and processor registers;
  - a control unit containing an instruction register and program counter;
  - a memory to store both data and instructions;

Computer-System Organization

Von Neumann Model

- We have just described the basics of the Von Neumann model of computing.
- "... an electronic digital computer with parts consisting of
  - a processing unit containing an arithmetic logic unit and processor registers;
  - a control unit containing an instruction register and program counter;
  - a memory to store both data and instructions;
  - external mass storage; and

Computer-System Organization

Von Neumann Model

- We have just described the basics of the Von Neumann model of computing.
- "… an electronic digital computer with parts consisting of
  - a processing unit containing an arithmetic logic unit and processor registers;
  - a control unit containing an instruction register and program counter;
  - a memory to store both data and instructions;
  - external mass storage; and
  - input and output mechanisms."

Computer-System Organization

Von Neumann Model

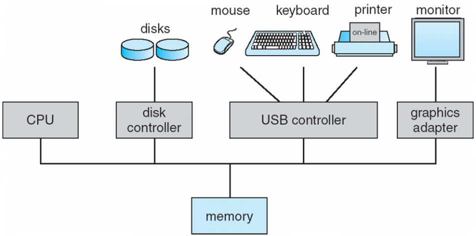
- We have just described the basics of the Von Neumann model of computing.
- "… an electronic digital computer with parts consisting of
  - a processing unit containing an arithmetic logic unit and processor registers;
  - a control unit containing an instruction register and program counter;
  - a memory to store both data and instructions;
  - external mass storage; and
  - input and output mechanisms."

  — Wikipedia "Von Neumann architecture".

Computer-System Organization

Von Neumann Model (contd.)

The above Von Neumann Model defines/describes:

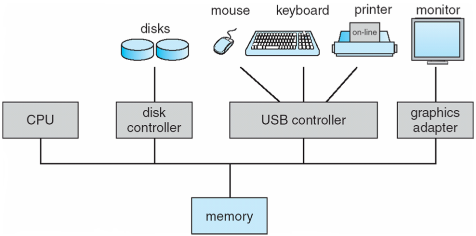


Computer-System Organization

Von Neumann Model (contd.)

The above Von Neumann Model defines/describes:

- How is a computer-system organized
  - One or more CPUs, device controllers connect through common bus providing access to shared memory.

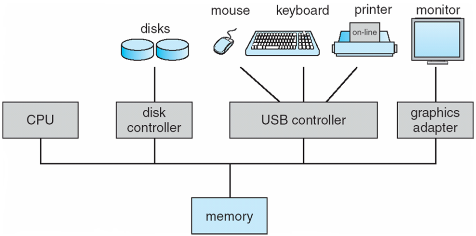


Computer-System Organization

Von Neumann Model (contd.)

The above Von Neumann Model defines/describes:

- How is a computer-system organized
  - One or more CPUs, device controllers connect through common bus providing access to shared memory.
- How is a computer-system operating
  - Concurrent execution of CPUs and devices competing for memory cycles.



Computer-System Organization

Computer-System Operation

- I/O devices and the CPU can execute concurrently.

Computer-System Organization

Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- But how?

Computer-System Organization

Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- But how?
  - 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.

Computer-System Organization

Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- But how?
  - 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.
- By now communication is enabled, but how to cooperate?

Computer-System Organization

Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- But how?
  - 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.
- By now communication is enabled, but how to cooperate?
  - Device controller informs CPU that it has finished its operation by causing an **interrupt**.

Computer-System Organization

Common Functions of Interrupts

- An interrupt is an input signal to the processor indicating an event that needs immediate attention.

Computer-System Organization

Common Functions of Interrupts

- An interrupt is an input signal to the processor indicating an event that needs immediate attention.
- Interrupt transfers **control** to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines.

Computer-System Organization

Common Functions of Interrupts

- An interrupt is an input signal to the processor indicating an event that needs immediate attention.
- Interrupt transfers **control** to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.

Computer-System Organization

Common Functions of Interrupts

- An interrupt is an input signal to the processor indicating an event that needs immediate attention.
- Interrupt transfers **control** to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request.

Computer-System Organization

Common Functions of Interrupts

- An interrupt is an input signal to the processor indicating an event that needs immediate attention.
- Interrupt transfers **control** to the interrupt service routine generally, through the **interrupt vector**, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request.
- An operating system is **interrupt driven**.

Computer-System Organization

Interrupt Handling

- The operating system preserves the state of the CPU by storing registers and the program counter.

Computer-System Organization

Interrupt Handling

- The operating system preserves the state of the CPU by storing registers and the program counter.
- Determines which type of interrupt has occurred:
  - polling, or
  - vectored interrupt system

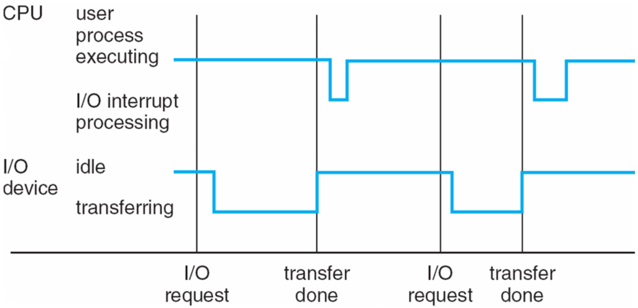
Computer-System Organization

Interrupt Handling

- The operating system preserves the state of the CPU by storing registers and the program counter.
- Determines which type of interrupt has occurred:
  - polling, or
  - vectored interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt.

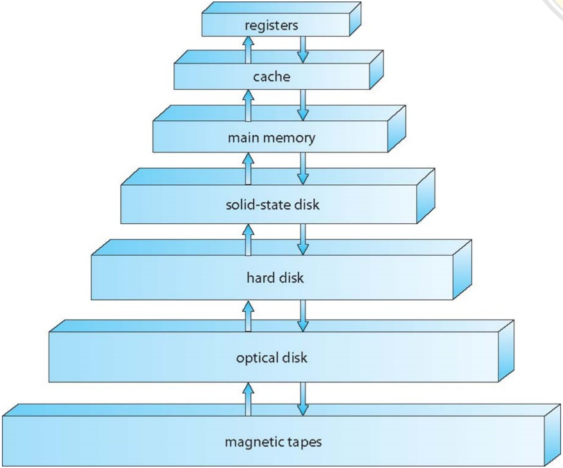
Computer-System Organization

Interrupt Timeline



Computer-System Organization

Storage-Device Hierarchy



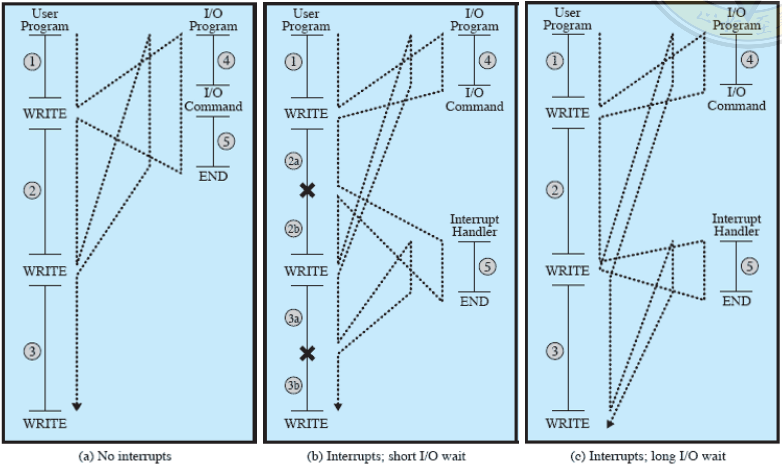
Computer-System Organization

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.

Computer-System Organization

An Example of I/O Interrupts



Computer-System Organization

I/O Structure — No Interrupts

- After I/O starts, control returns to user program only upon I/O completion
  - Wait instruction idles the CPU.
  - Wait loop (contention for memory access).
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing.

Computer-System Organization

I/O Structure — Interrupt Driven

- After I/O starts, control returns to user program without waiting for I/O completion

Computer-System Organization

I/O Structure — Interrupt Driven

- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** — request to the OS to allow user to wait for I/O completion.

Computer-System Organization

I/O Structure — Interrupt Driven

- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** — request to the OS to allow user to wait for I/O completion.
  - Device-status table contains entry for each I/O device indicating its type, address, and state.

Computer-System Organization

I/O Structure — Interrupt Driven

- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** — request to the OS to allow user to wait for I/O completion.
  - Device-status table contains entry for each I/O device indicating its type, address, and state.
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt.

Computer-System Organization

I/O Structure — Interrupt Driven

- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** — request to the OS to allow user to wait for I/O completion.
  - Device-status table contains entry for each I/O device indicating its type, address, and state.
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt.
- Who returns the control?

Computer-System Organization

I/O Structure — Interrupt Driven

- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** — request to the OS to allow user to wait for I/O completion.
  - Device-status table contains entry for each I/O device indicating its type, address, and state.
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt.
- Who returns the control?
  - Device Driver for each device controller to manage I/O. Provides uniform interface between controller and kernel.

- **Direct Memory Access** Structure

Computer-System Organization

I/O Structure — Direct Memory Access

- **Direct Memory Access** Structure
  - Used for high-speed I/O devices able to transmit information at close to memory speeds.

Computer-System Organization

I/O Structure — Direct Memory Access

- **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.

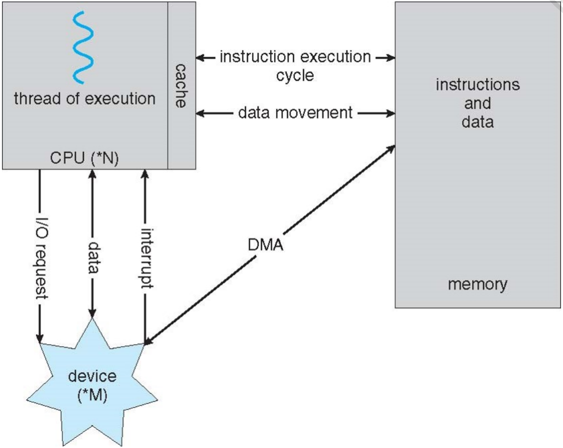
Computer-System Organization

I/O Structure — Direct Memory Access

- **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 Organization

How a Modern Computer Works



Contents

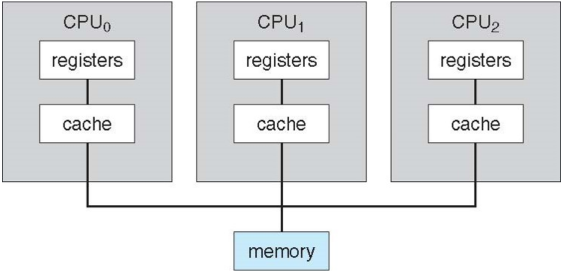
1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Computer-System Architecture

- Most systems use a single general-purpose processor until 10 years ago.
  - Most systems have special-purpose processors as well.
- Multiprocessor systems growing in use and importance.
  - Also known as parallel systems, tightly-coupled systems.
  - Advantages include:
    1. Increased throughput.
    2. Economy of scale.
    3. Increased reliability — graceful degradation or fault tolerance.
  - Two types:
    1. **Asymmetric Multiprocessing** — each processor is assigned a specific task.
    2. **Symmetric Multiprocessing** — each processor performs all tasks.

Computer-System Architecture

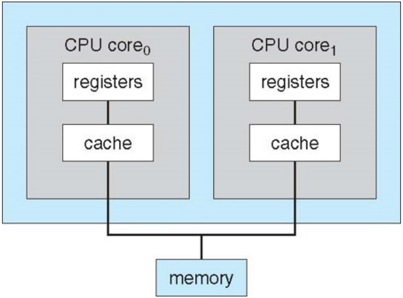
Symmetric Multiprocessing Architecture



Computer-System Architecture

A Dual-Core Design

- Multi-chip and multicore.
  - Why multicore?
  - On-chip communication is faster, uses significantly less power.
- Blade Servers: all in one chassis.
  - Chassis containing multiple separate systems.



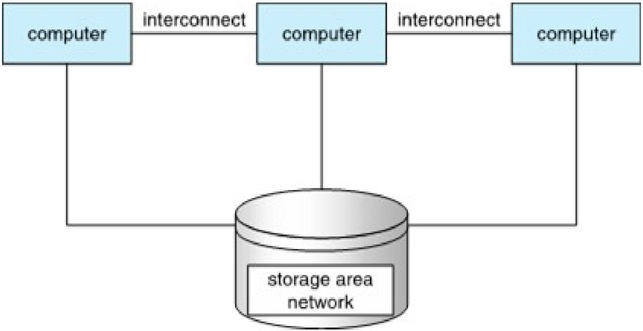
Computer-System Architecture

Clustered Systems

- Like multiprocessor systems, but multiple systems working together.
- Loosely-coupled systems.
  - Usually sharing storage via a storage-area network (SAN).
  - Provides a high-availability service which survives failures.
    - Asymmetric clustering has one machine in hot-standby mode.
    - Symmetric clustering has multiple nodes running applications, monitoring each other.
  - Some clusters are for high-performance computing (HPC).
    - Applications must be written to use parallelization.
  - Some have distributed lock manager (DLM) to avoid conflicting operations.

Computer-System Architecture

Clustered Systems (contd.)



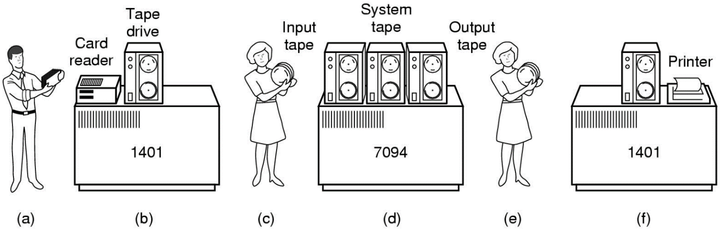
Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Operating-System Structure

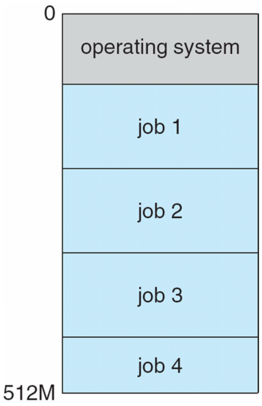
Some History

- A typical simple Batch system
  - bring cards to IBM 1401
  - IBM 1401 read cards to tape
  - put tape on IBM 7094 which does computing
  - put tape on IBM 1401 which prints output



Operating-System Structure

Some History (contd.)



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

Operating-System Structure

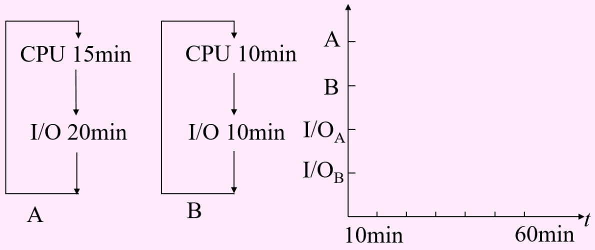
Some History (contd.)

- **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** (**short term 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.

In Class Exercise

Multiprogramming

- Consider a multiprogramming environment, two programs *A* and *B* share the system simultaneously, and run as follows. Suppose *B* runs first, I/O_A and I/O_B are different devices, and the time of switching between *A* and *B* can be ignored. Draw the timeline for these two programs, and calculate CPU utilization within 60 minutes.
- What if I/O_A and I/O_B are the same device?



Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

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

Dual Mode

- **Dual-mode** operation allows OS to protect itself and other system components.

Operating-System Operations

Dual Mode

- **Dual-mode** operation allows OS to protect itself and other system components.
  - **User mode** and **kernel mode**.

Operating-System Operations

Dual Mode

- **Dual-mode** operation allows OS to protect itself and other system components.
  - **User mode** and **kernel mode**.
  - Mode bit provided by hardware.

Operating-System Operations

Dual Mode

- **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.

Operating-System Operations

Dual Mode

- **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.

Operating-System Operations

Dual Mode

- **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.

Operating-System Operations

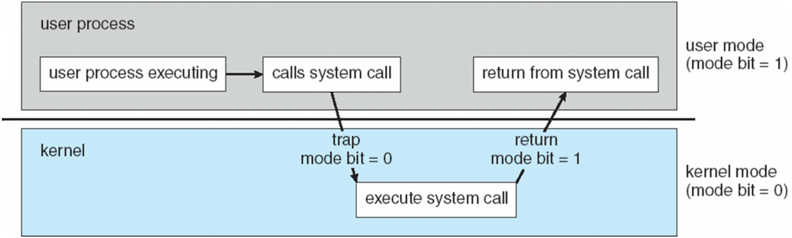
Dual Mode

- **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.

Operating-System Operations

Dual Mode (contd.)

Transition from User to Kernel Mode



Operating-System Operations

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.

Operating-System Operations

What Happens When the Computer System is Booting?

OS @boot (kernel mode)	**Hardware**
initialize trap table	
	remember addresses of ...
	system call handler
	timer handler
	illegal instruction handler
start interrupt timer	
	start timer; interrupt after X ms

Operating-System Operations

What Happens When the Computer System is Running?

OS @run (kernel mode)	**Hardware**	**Program (user mode)**
to start process *A*:		
return-from-trap (into *A*)		
	move to *user mode*	
		process *A* runs:
		fetch instruction
		execute instruction
		...
	timer interrupt	
	move to *kernel mode*	
	jump to interrupt handler	

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Process Management

Processes

- 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.
- 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.

Process Management

Process Management Activities

- The operating system is responsible for the following activities in connection with process management:
  - Scheduling processes and threads on the CPUs.
  - Creating and deleting both user and system processes.
  - Suspending and resuming processes.
  - Providing mechanisms for process synchronization.
  - Providing mechanisms for process communication.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. **Memory Management**
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

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.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. **Storage Management**
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Storage Management

- OS provides a uniform, logical view of information storage.
- Abstracts physical properties to logical storage unit — **file**.
- Each medium is controlled by a device (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.

Storage Management

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).

Storage Management

Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software).
- 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.

Storage Management

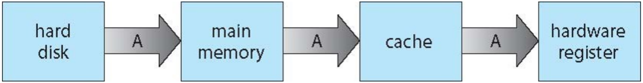
Performance of Various Levels of Storage

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

Storage Management

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.



Storage Management

I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user.
- I/O subsystem consists of:
  - 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.

Process/Memory/Storage Management

Too Many Terms?

- Just remember three key ideas for now:

Process/Memory/Storage Management

Too Many Terms?

- Just remember three key ideas for now:
  - **Virtualization**: The OS takes a physical resource (such as the processor, or memory, or a disk) and transforms it into a more general, powerful, and easy-to-use virtual form of itself.

Process/Memory/Storage Management

Too Many Terms?

- Just remember three key ideas for now:
  - **Virtualization**: The OS takes a physical resource (such as the processor, or memory, or a disk) and transforms it into a more general, powerful, and easy-to-use virtual form of itself.
  - **Concurrency**: Many problems arise, and must be addressed, when working on many things at once (i.e., concurrently) in the same program.

Process/Memory/Storage Management

Too Many Terms?

- Just remember three key ideas for now:
  - **Virtualization**: The OS takes a physical resource (such as the processor, or memory, or a disk) and transforms it into a more general, powerful, and easy-to-use virtual form of itself.
  - **Concurrency**: Many problems arise, and must be addressed, when working on many things at once (i.e., concurrently) in the same program.
  - **Persistence**: The OS makes information persist, despite computer crashes, disk failures, or power outages.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

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.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Kernel Data Structures

You Should Have Known the Concepts of ...

- lists, stacks, and queues,
- trees,
- hash functions and maps,
- bitmaps.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

Computing Environments

Traditional Computing

- 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 Computing

- Hand-held smart-phones, 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

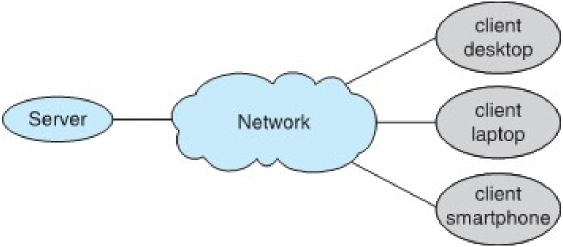
Distributed Systems

- 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 (such as file sharing) between systems across network.
  - Communication scheme allows systems to exchange messages.
  - Illusion of a single system.

Computing Environments

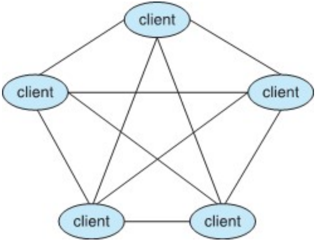
Client-Server Computing

- 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 Computing



- 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 .

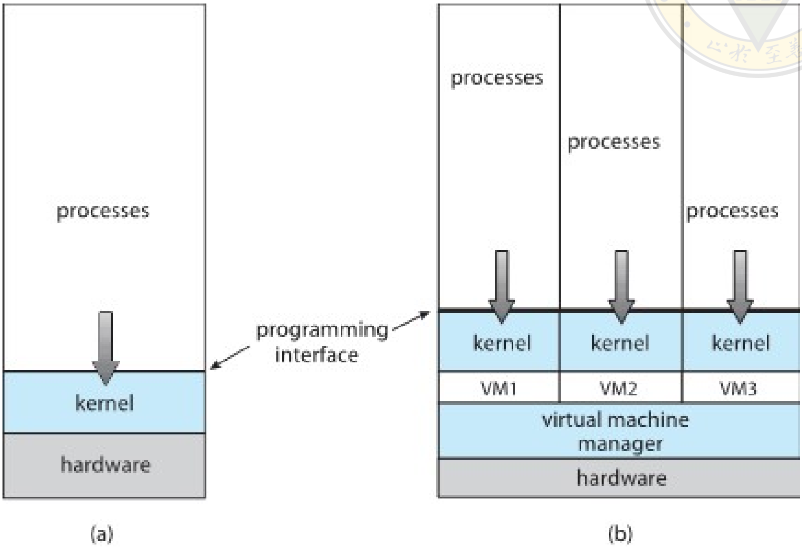
Computing Environments

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.
  - Virtual machine manager (VMM) provides virtualization services.

Computing Environments

Virtualization (contd.)



Computing Environments

Cloud Computing

- Delivers computing, storage, apps as a service across a network.
- Logical extension of virtualization because it uses virtualization as the base for it functionality.
- 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

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.

Contents

1. What Operating Systems Do
2. Computer-System Organization
3. Computer-System Architecture
4. Operating-System Structure
5. Operating-System Operations
6. Process Management
7. Memory Management
8. Storage Management
9. Protection and Security
10. Kernel Data Structures
11. Computing Environments
12. Open-Source Operating Systems

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.