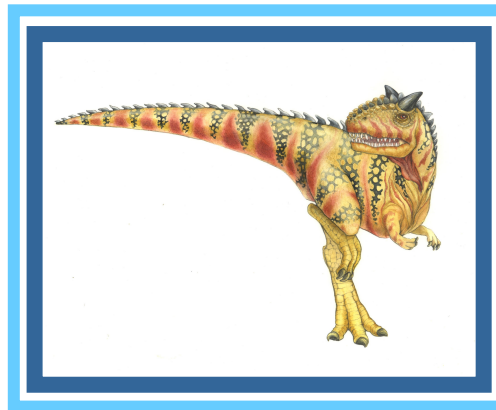


Chapter 1: Introduction





Chapter 1: Introduction

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture
- Operating-System Structure
- Operating-System Operations
- Process Management
- Memory Management
- Storage Management
- Protection and Security
- Kernel Data Structures
- 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





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 (resource management)





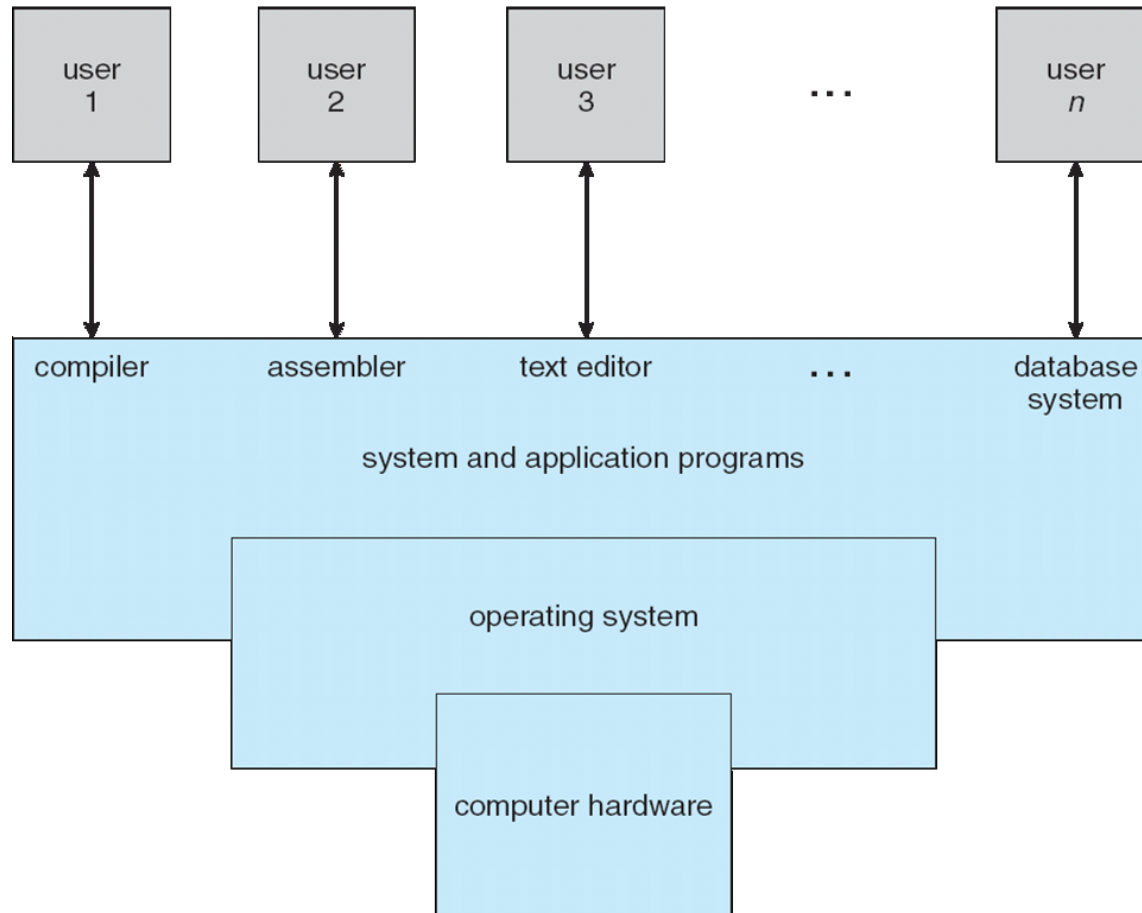
Computer System Structure

- Computer system can be divided into four components:
 - Hardware – provides basic computing resources
 - ▶ CPU, memory, disk, 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





Four Components of a Computer System





What Operating Systems Do

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





Operating System Definition

- OS is a **resource allocator**
 - Manages all resources
 - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
 - Controls execution of programs to prevent errors and improper use of the computer (protection)





Operating System Definition (Cont.)

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is a good approximation
 - But varies widely
- “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.





Computer Startup

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

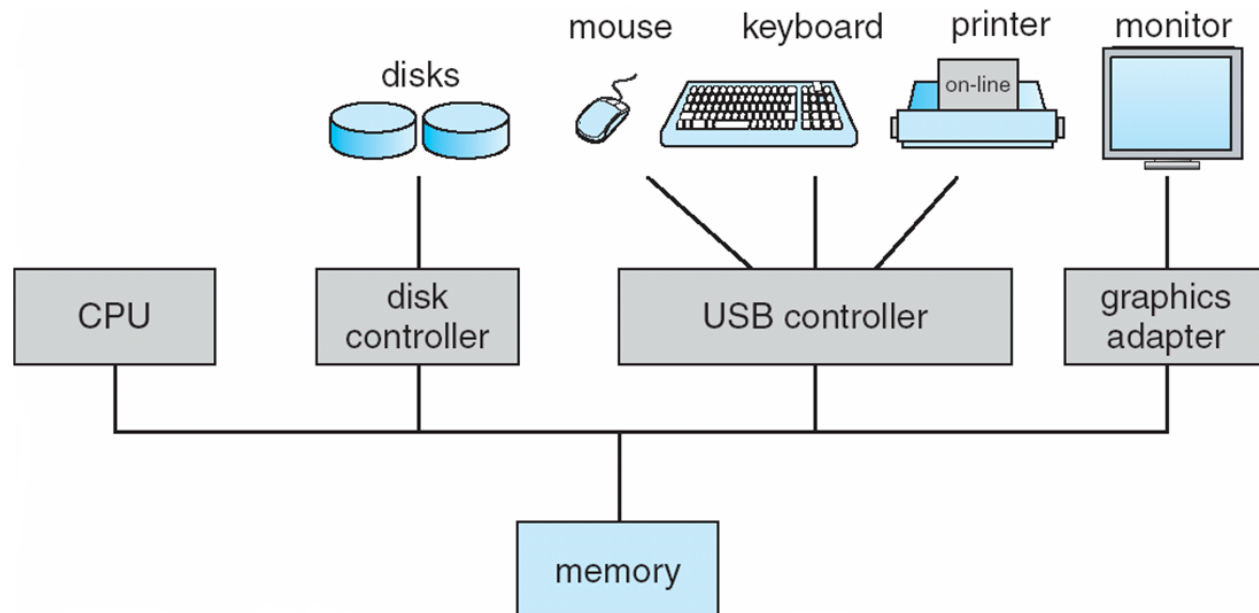




Computer System Organization

■ Computer-system operation

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





Computer-System 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

- 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**





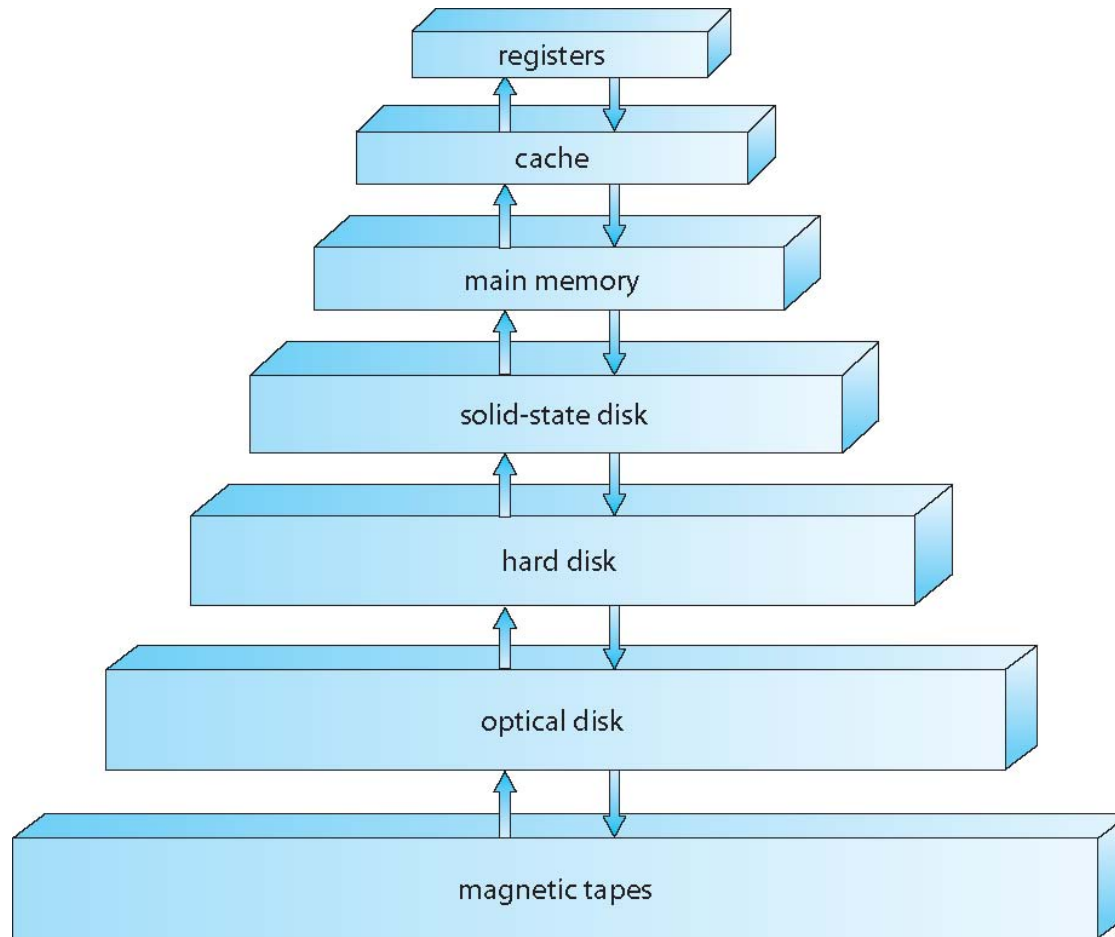
Storage Structure

- 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-Device Hierarchy





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 is smaller than storage being cached
 - Cache management is an important design problem
 - Cache size and replacement policy





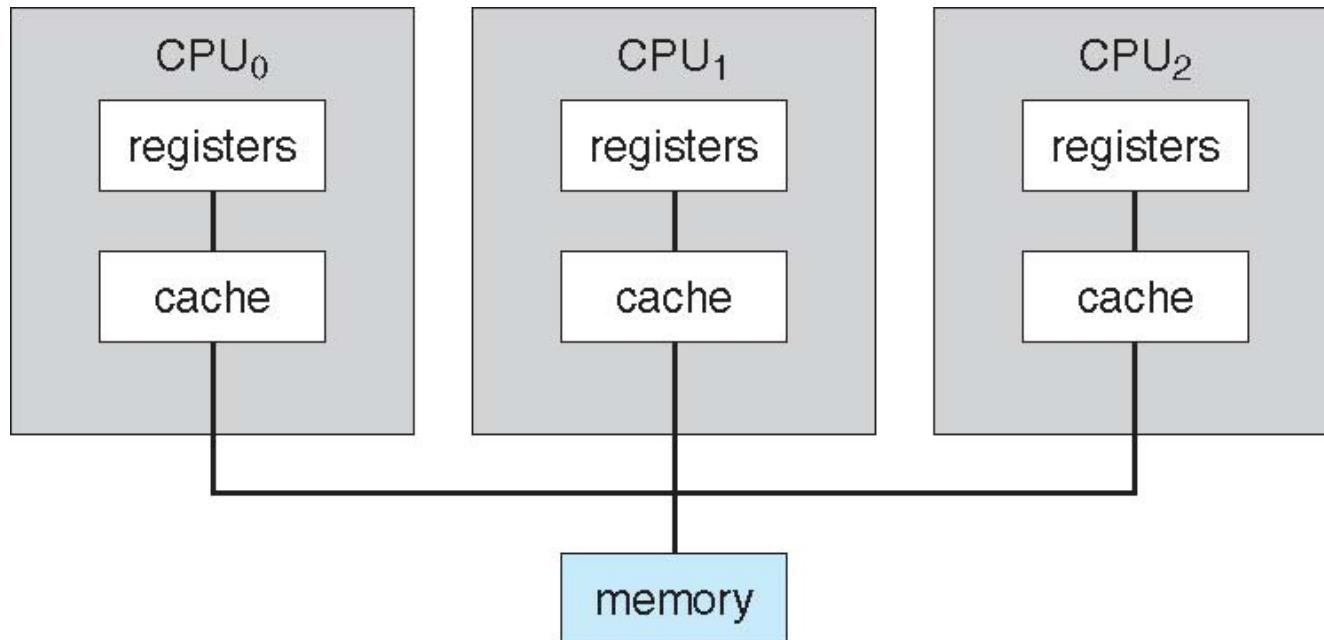
Computer-System Architecture

- Most systems use a single general-purpose processor
 - 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





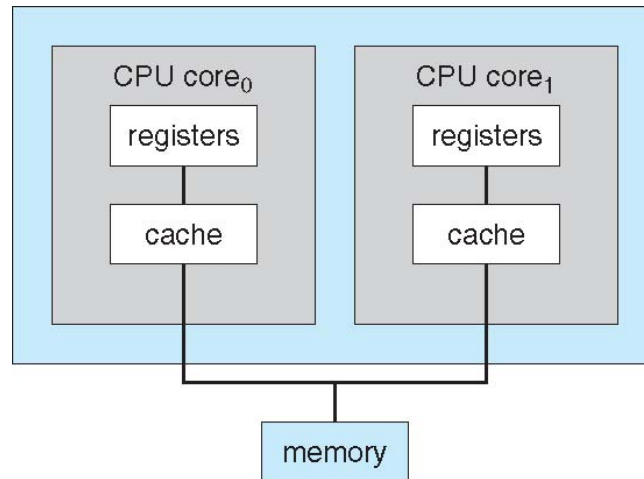
Symmetric Multiprocessing Architecture





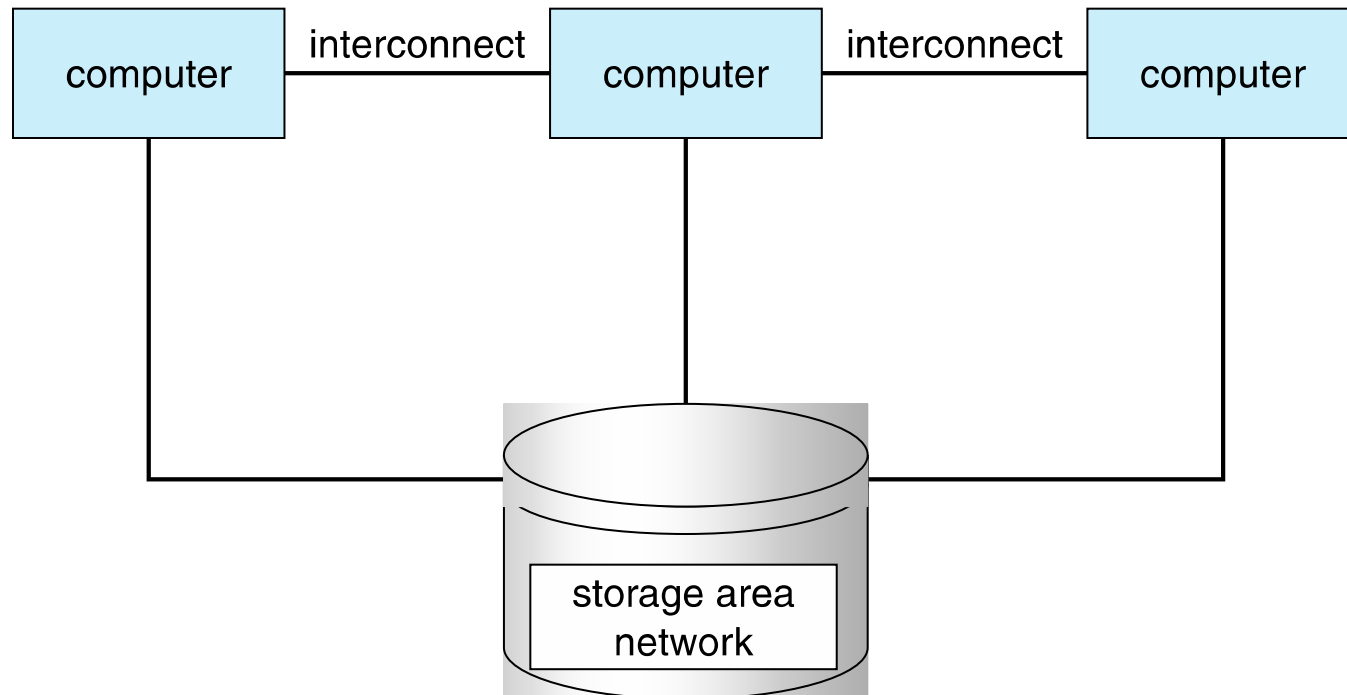
A Dual-Core Design

- Multi-chip and **multicore**
- Systems containing all chips





Clustered Systems





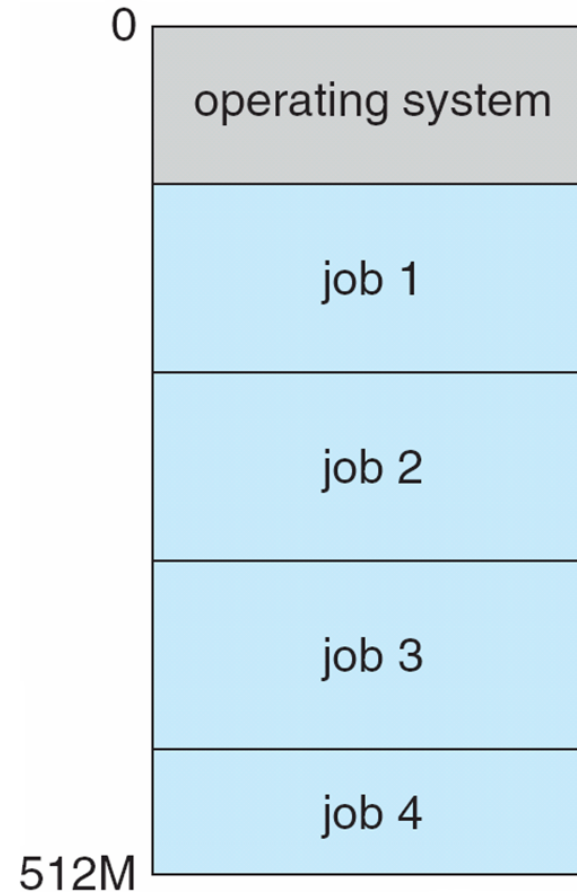
Operating System Structure

- **Multiprogramming (Batch system)** needed for efficiency
 - Single user cannot keep CPU and I/O devices busy at all times
 - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
 - A subset of total jobs in system is kept in memory
 - One job selected and run via **job scheduling**
 - When it has to wait (for I/O for example), OS switches to another job
- **Timesharing (multitasking)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
 - **Response time** should be < 1 second
 - Each user has at least one program executing in memory □ **process**
 - If several jobs ready to run at the same time □ **CPU scheduling**
 - If processes don't fit in memory, **swapping** moves them in and out to run
 - **Virtual memory** allows execution of processes not completely in memory





Memory Layout for Multiprogrammed System





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, invalid memory access)
 - ▶ Request for operating system service (system call)
 - ▶ Time slice (quantum) of a process expires
- An interrupt service routine is provided to deal with the interrupt
- Interrupts are used to prevent processes from modifying each other or the operating system





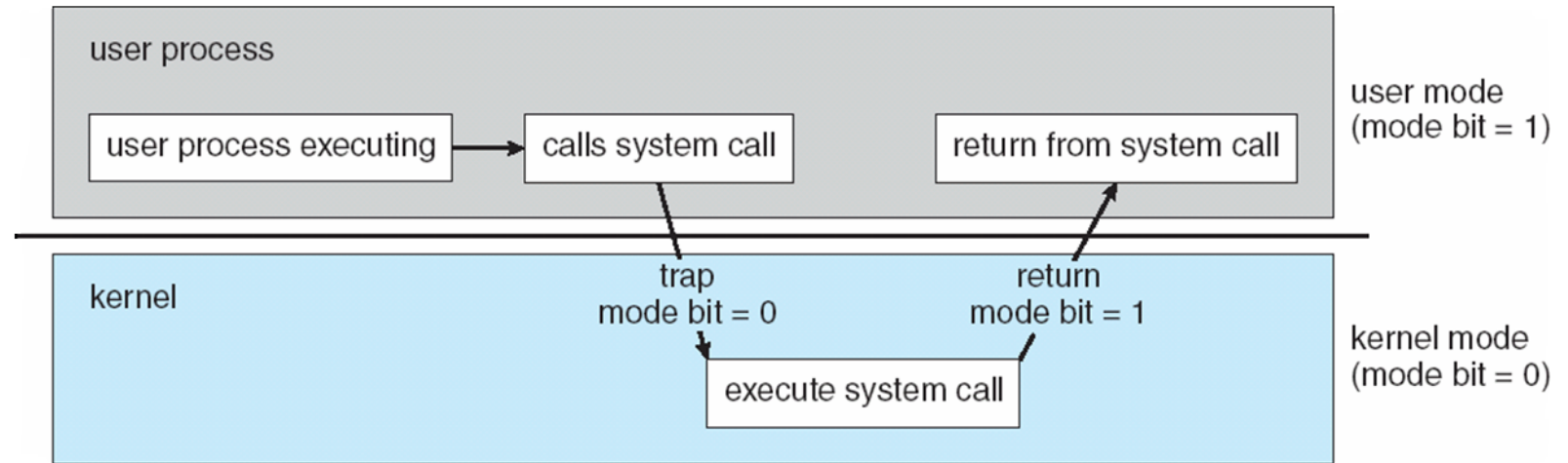
Operating-System Operations (cont.)

- **Dual-mode** operation allows OS to protect itself and other system components
 - **User mode** and **kernel mode**
 - **Mode bit** provided by hardware
 - ▶ Provides ability to distinguish between user code and kernel code
 - ▶ Some instructions designated as **privileged**, only executable in kernel mode
 - ▶ System call changes mode to kernel, return resets it to user
- Lack of HW-supported dual mode can cause serious problems (e.g. DOS on Intel 8088)
- Some modern machines support multiple modes (e.g third mode for virtual machine manager)





Transition from User to Kernel Mode





Interrupts and Timers

- 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 sets the counter (privileged instruction)
 - When counter reaches zero, generate an interrupt
 - Set up before scheduling process to regain control or terminate program that exceeds allotted time





Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity***, process is an ***active entity***.
- Process needs resources to accomplish its task
 - CPU, memory, I/O, files
 - Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one **program counter** specifying location of next instruction to execute
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
 - Concurrency by multiplexing the CPUs among processes / threads





Process Management Activities

The operating system is responsible for the following activities in connection with process management:

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling





Memory Management

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





Storage Management

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





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





I/O Subsystem

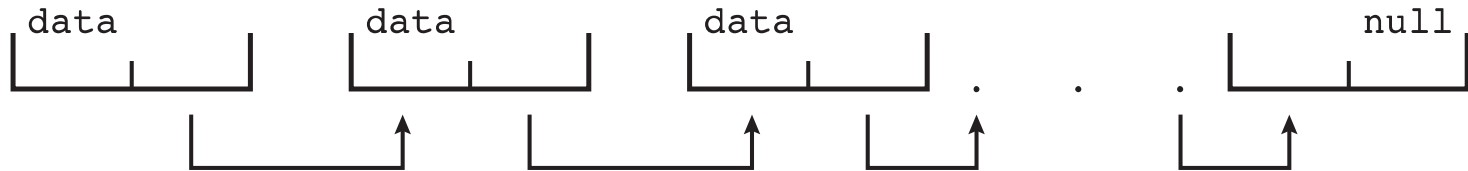
- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
 - General device-driver interface
 - Drivers for specific hardware devices
 - Memory management of I/O



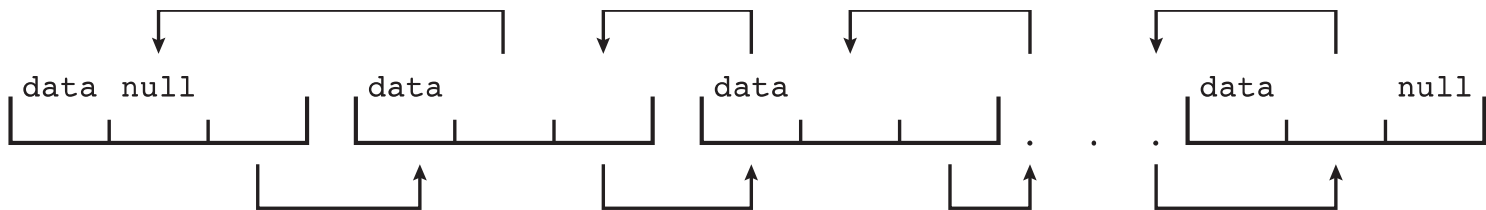


Kernel Data Structures

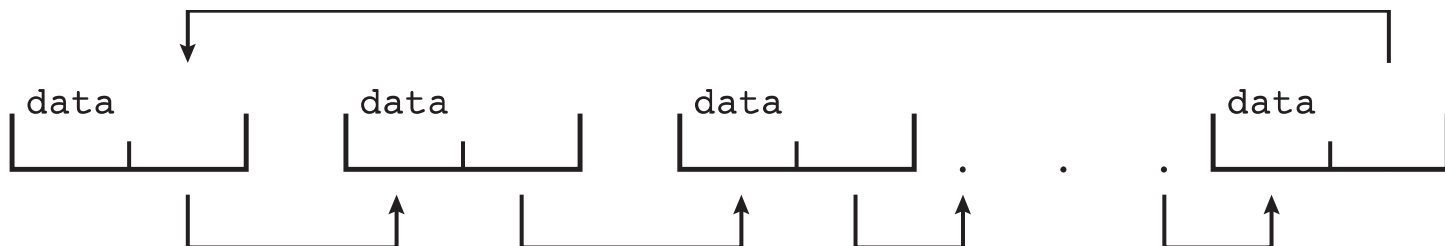
- Many similar to standard programming data structures
- ***Singly linked list***



- ***Doubly linked list***



- ***Circular linked list***



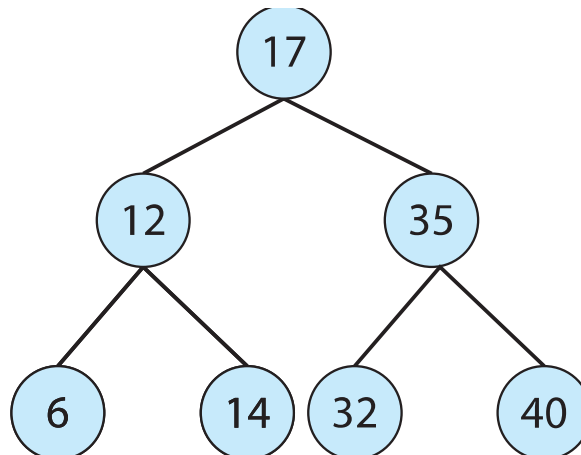


Kernel Data Structures

■ Binary search tree

left \leq right

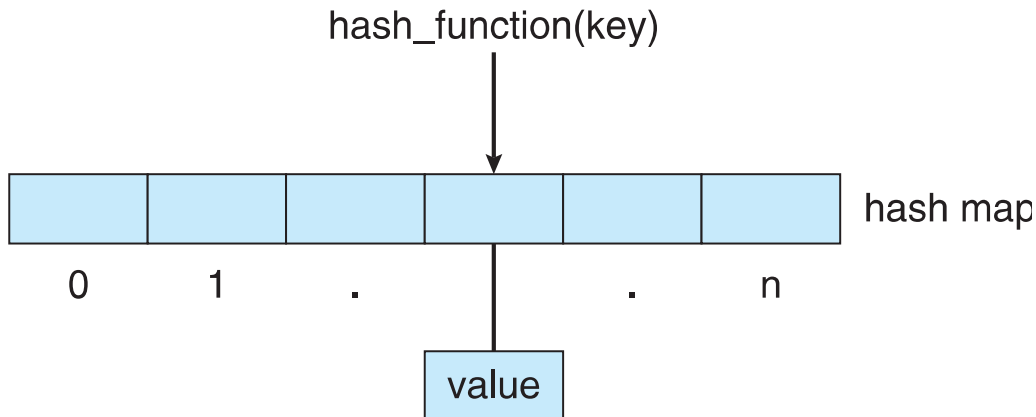
- Search performance is $O(n)$
- **Balanced binary search tree** is $O(\lg n)$





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>`





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**), **Open Solaris** and many more
- Good for academic purposes (education and research)



End of Chapter 1

