ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Real-Time Systems and programming for Automation M

5. Introduction to OS

Notice

The course material includes slides downloaded from:

http://codex.cs.yale.edu/avi/os-book/

(slides by Silberschatz, Galvin, and Gagne, associated with Operating System Concepts, 9th Edition, Wiley, 2013)

and

http://retis.sssup.it/~giorgio/rts-MECS.html

(slides by Buttazzo, associated with Hard Real-Time Computing Systems, 3rd Edition, Springer, 2011)

which have been edited to suit the needs of this course, and slides provided by Laura Carnevali used by her course on Software Engineering for Embedded Systems at the University of Florence.

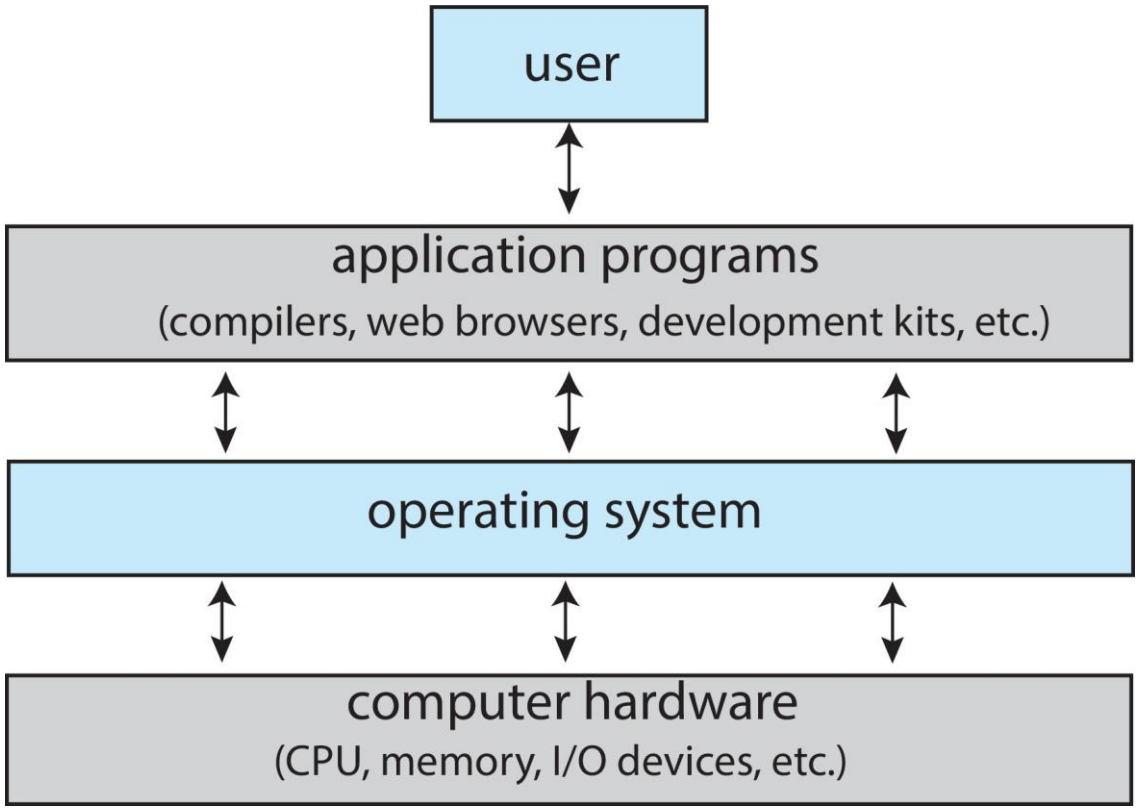
The slides are authorized for personal use only.

Any other use, redistribution, and any for profit sale of the slides (in any form) requires the consent of the copyright owners.

What is an Operating System?

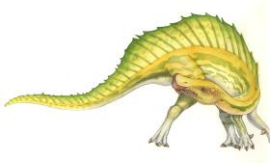
- A program that acts as an **intermediary** between a **user** of a computer and the computer **hardware**
- User's view / system's view
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware efficiently
  - Ensure a computer system's predictable behavior

Abstract View of Computer Components



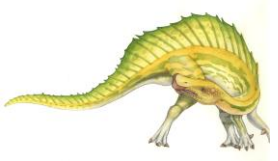
Operating System Definition

- 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
  - Focus on I/O devices

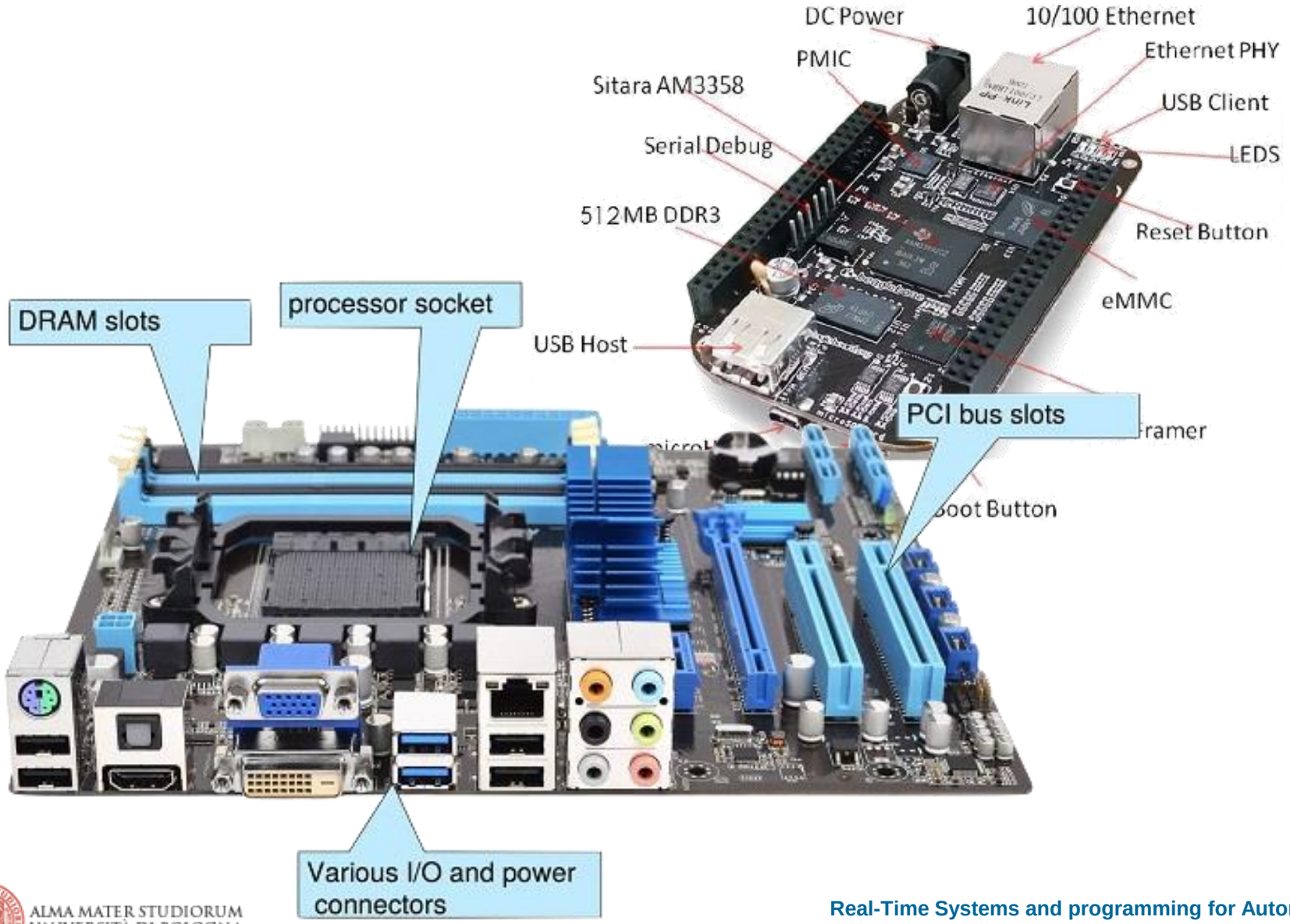


Operating System Definition

- No universally accepted definition
- "Everything a vendor ships when you order an operating system" is good approximation
  - But varies wildly
- "A piece of software bringing together all common functions of controlling and allocating resources on behalf of application 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

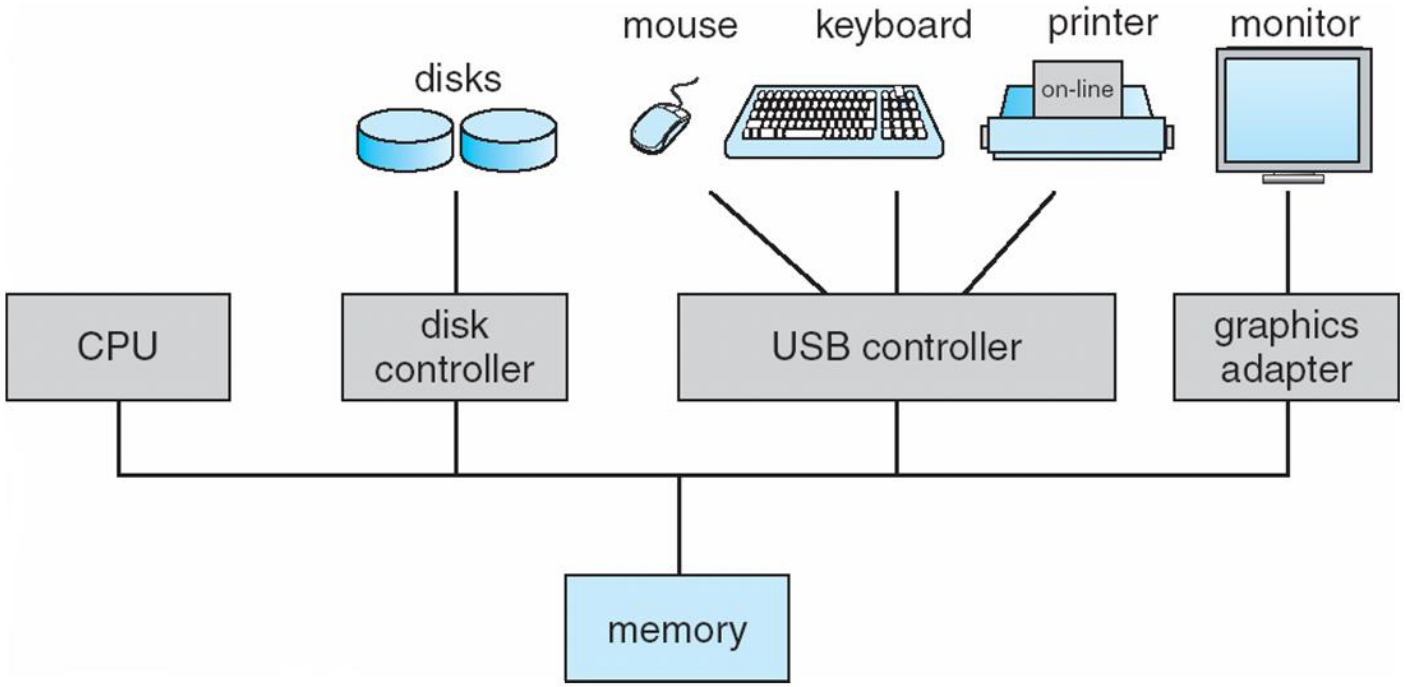


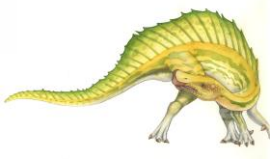
Computer System



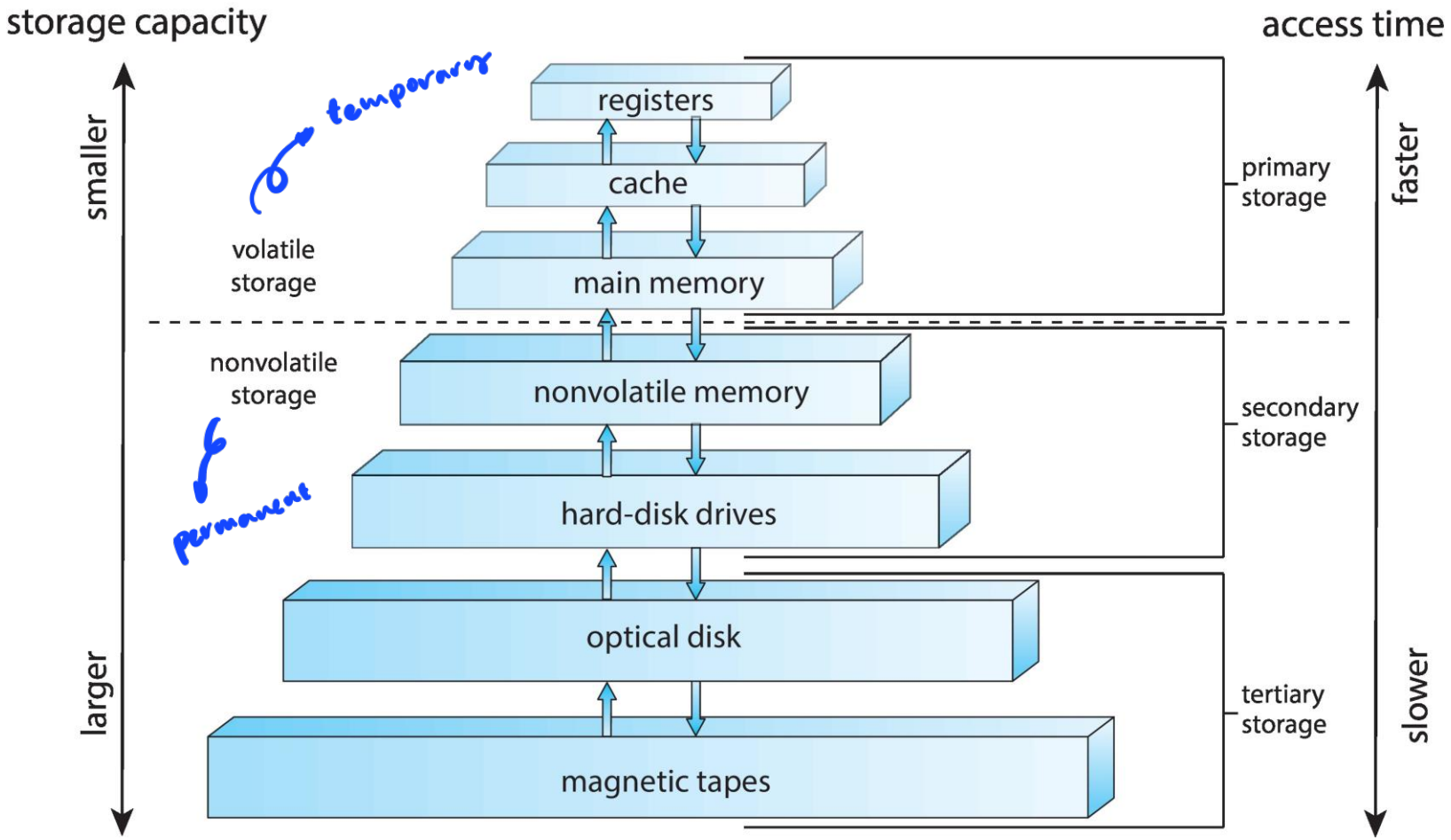
Computer System Organization

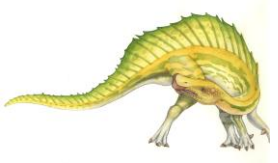
- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles





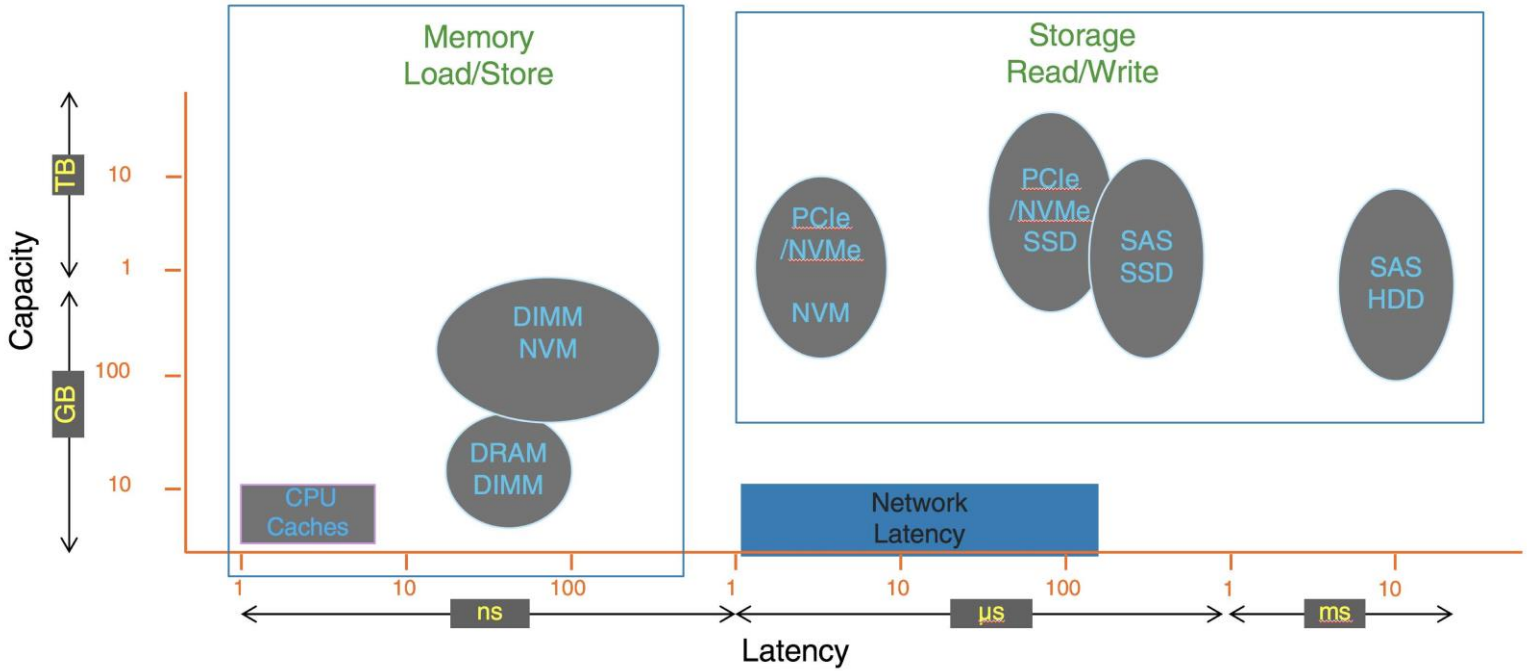
Storage-Device Hierarchy

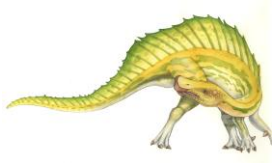




Storage Structure

- Main memory – only large storage media that the CPU can access directly
- Secondary storage – extension of main memory that provides large nonvolatile storage capacity
  - Caching – copying information into faster storage system; main memory can be viewed as a last *cache* for secondary storage





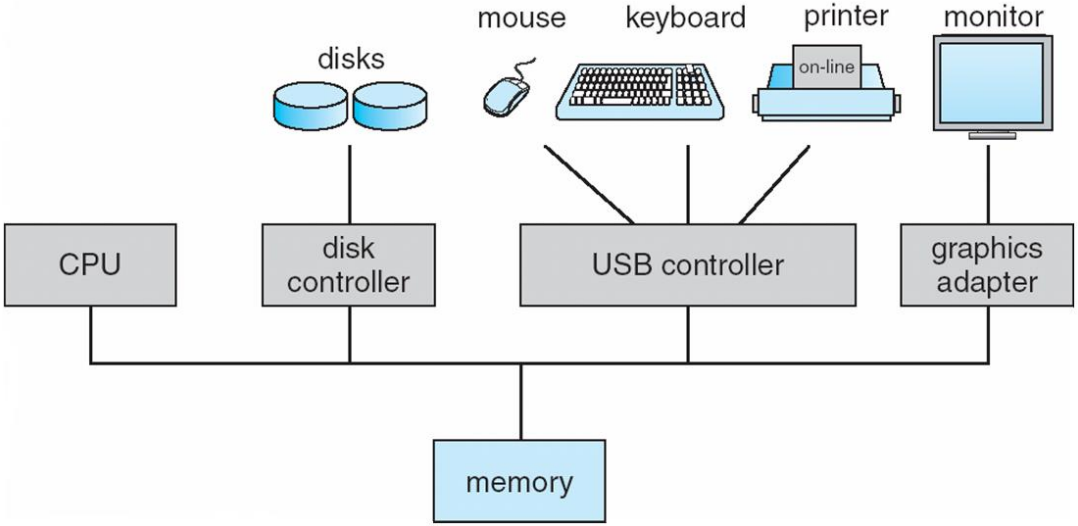
Computer Startup

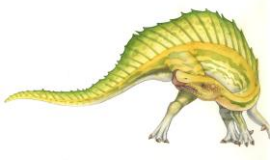
all done in kernel

- **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
  - Kernel starts first process (init) and wait for events to occur

Computer-System Operation

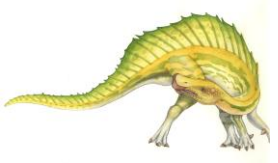
- I/O devices and the CPU can execute **concurrently**
- Each device controller
  - is in charge of a particular device type
  - has a local buffer
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller
- Device controller informs CPU that it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





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

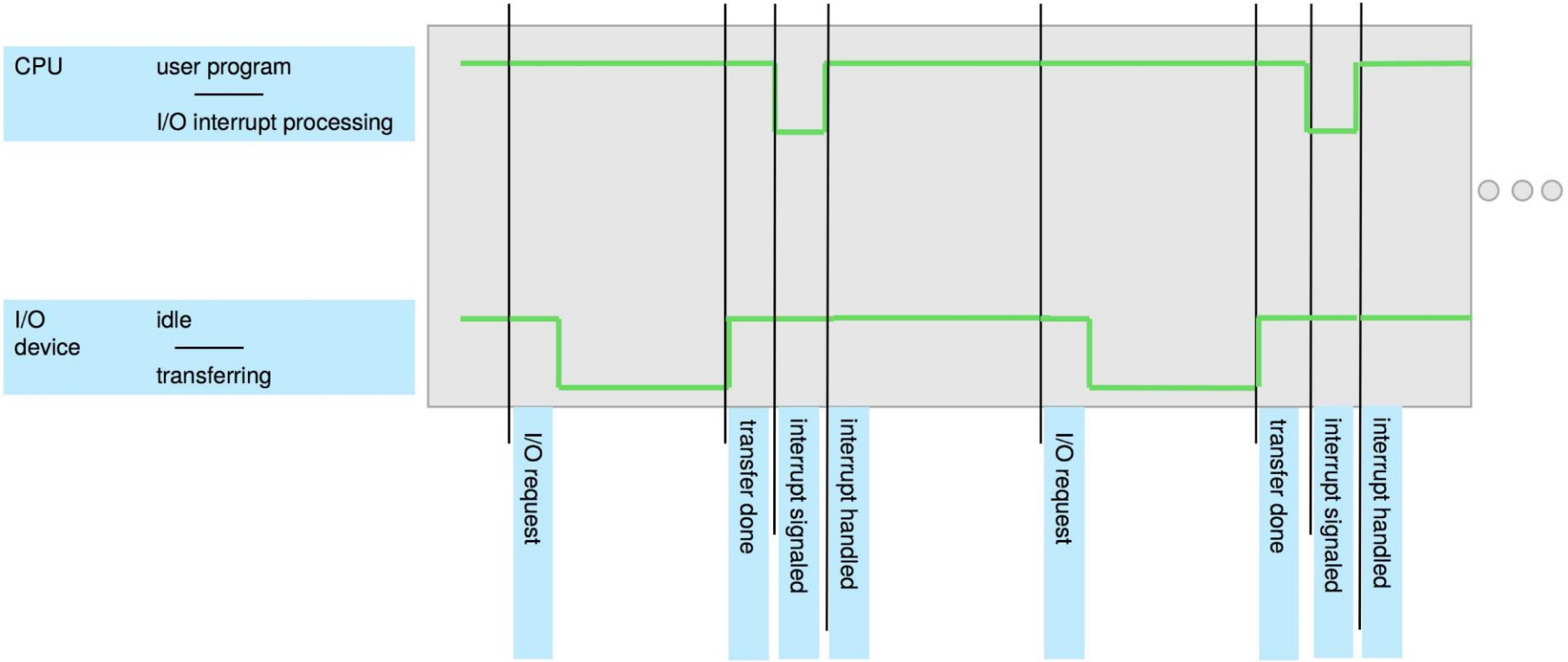


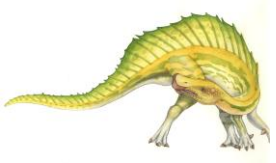
Common Functions of Interrupts

- 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
- Incoming interrupts are *disabled* while another interrupt is being processed to prevent a *lost interrupt*
- A *trap* is a software-generated interrupt caused either by an error or a user request
- An operating system is **interrupt driven**

not something that actively check things happen

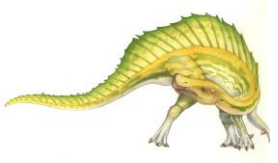
Interrupt Timeline





I/O Structure

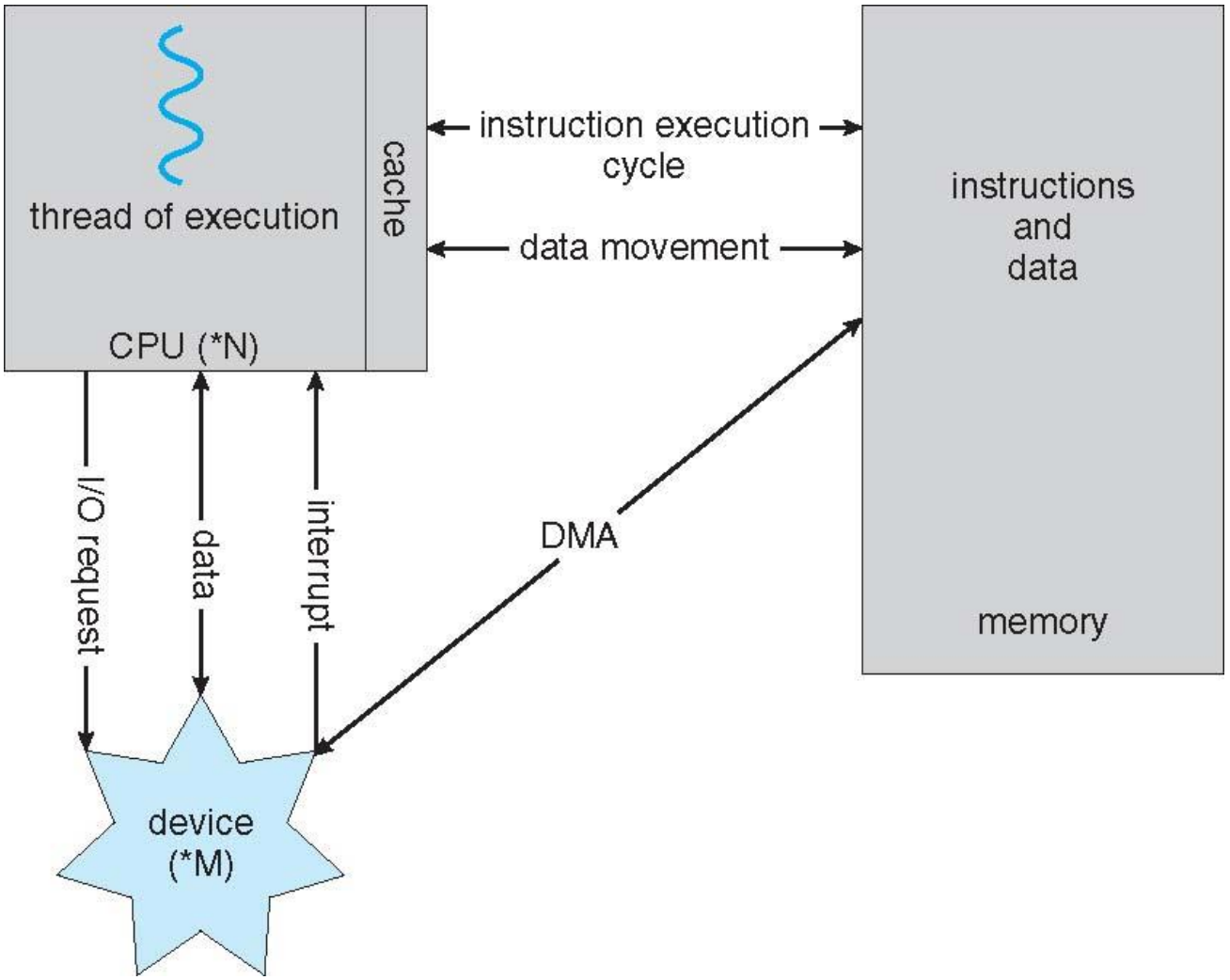
- After I/O starts, control returns to user program only upon I/O completion
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access)
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing
- After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** – request to the operating system 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
  - Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt

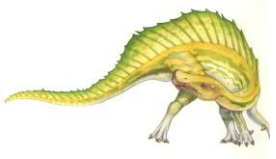


Direct Memory Access Structure

- 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

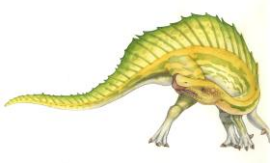
How a Modern Computer Works



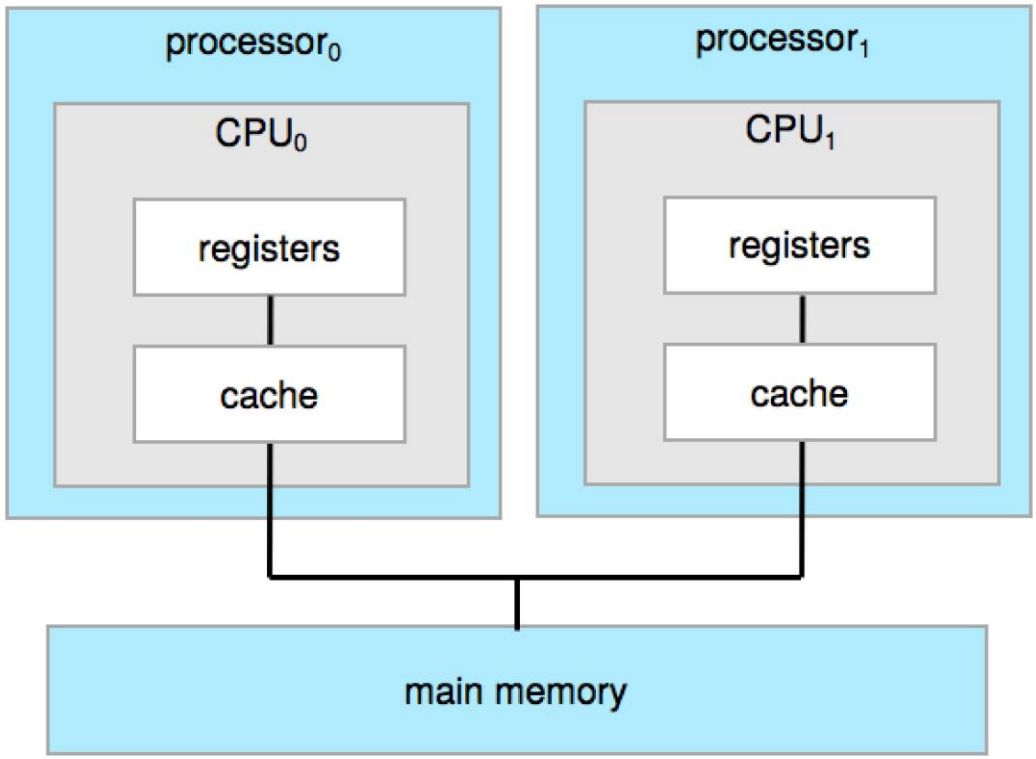


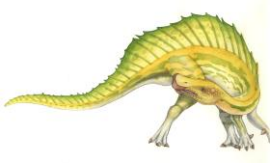
Computer-System Architecture

- Most systems use a single general-purpose processor
  - Most systems have special-purpose processors as well
- Multiprocessors systems growing in use and importance
  - Advantages include
    - Increased throughput
    - Economy of scale
    - Increased reliability – graceful degradation or fault tolerance
  - Two types
    1. Asymmetric Multiprocessing (each CPU has specific task)
    2. Symmetric Multiprocessing → executing same tasks

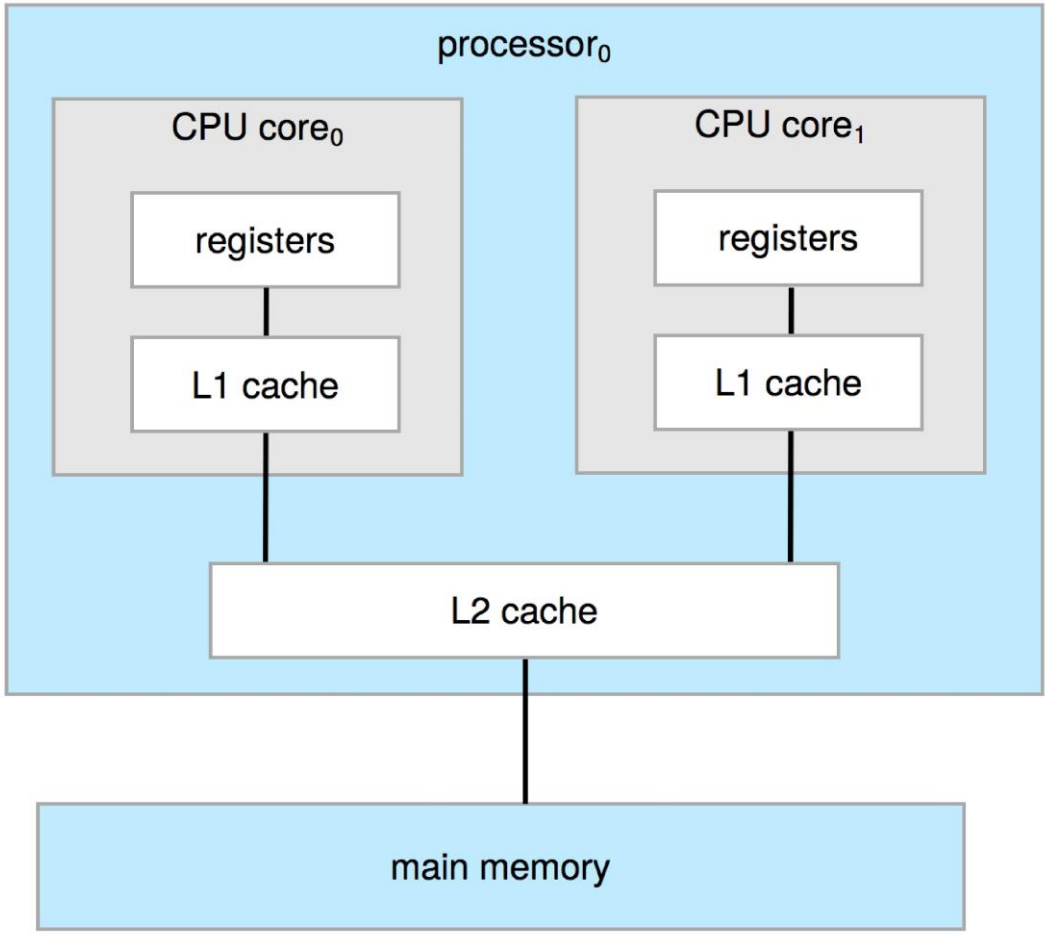


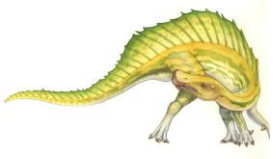
Symmetric Multiprocessing Architecture





A Dual-Core Design





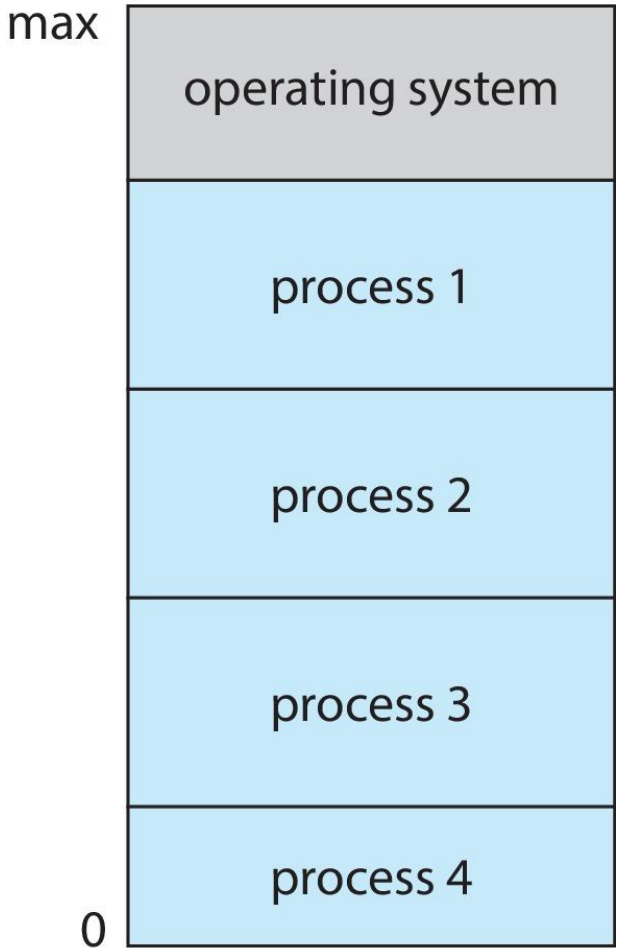
Quizzes

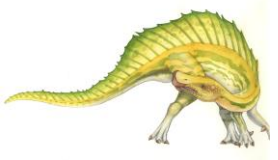
- An operating system is everything a vendor ships when you order an operating system true-ish
- The main goal of an operating system is to manage the hardware resources efficiently depends. could be easiness of use
- A trap can be generated intentionally by a user program Yes
- Direct Memory Access (DMA) is effective especially with very slow devices No
- Increased throughput, economy of scale, and increased reliability are three advantages of multiprocessor systems Yes
- Symmetric multiprocessing is only possible with computer systems that have an even number of CPUs No

Operating System Structure

- **Multiprogramming** 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

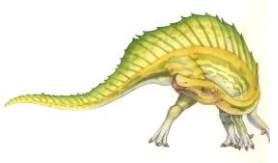
Memory Layout for Multiprogrammed System



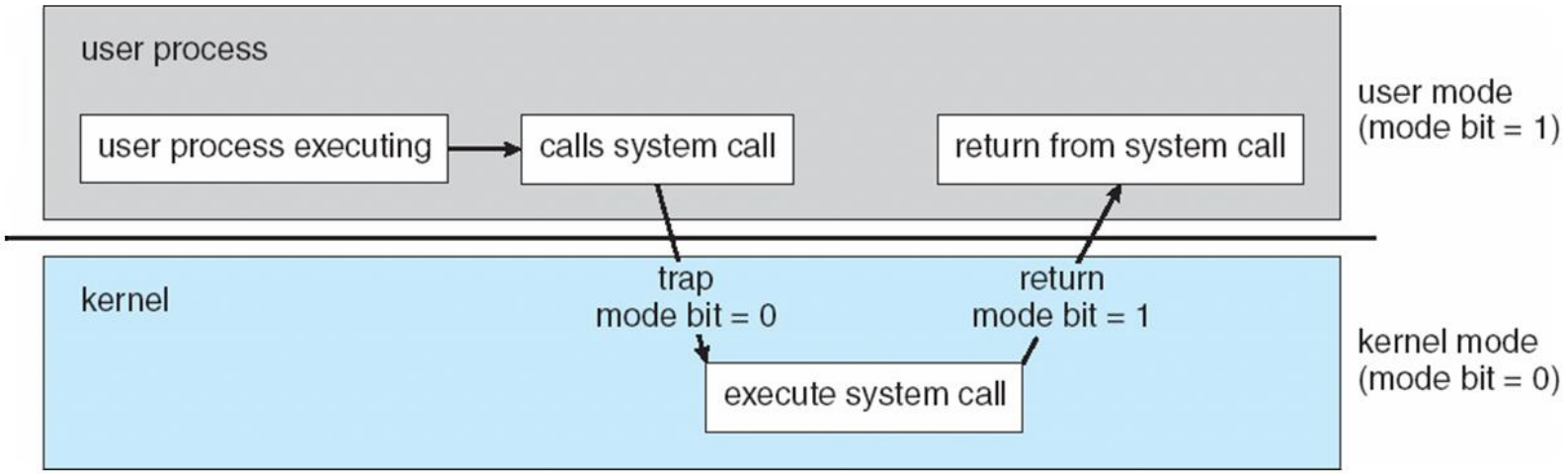


Operating-System Operations

- Interrupt driven by hardware
- Software error or request creates **exception** or **trap**
  - Division by zero, request for operating system service
- Other process problems include infinite loop, processes modifying each other's data, or code, or the operating system
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
- Increasingly CPUs support multi-mode operations
  - i.e. **virtual machine manager** (**VMM**) mode for guest **VMs**



Transition from User to Kernel Mode



- **Mode bit**
  - 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

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 zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time

Quizzes

- Which of the following instructions should be **privileged**?
  - Access I/O device ✓ → to choose which information to be shown
  - Modify entries in device-status table ✓
  - Set value of timer ✓ → to stop the process
  - Turn off interrupts ✓ → to keep the process running
  - Read the clock ✗
  - Issue a trap instruction ✗

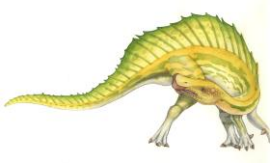
Process Management

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

Process Management Activities

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

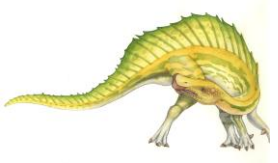
- Scheduling processes and threads on the CPUs
- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication



Memory Management

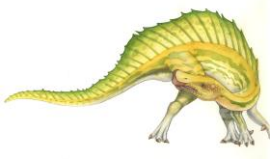
- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  - Optimizing CPU utilization and computer response to users

- Memory management activities:
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed



Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
- OS responsible for:
  - 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

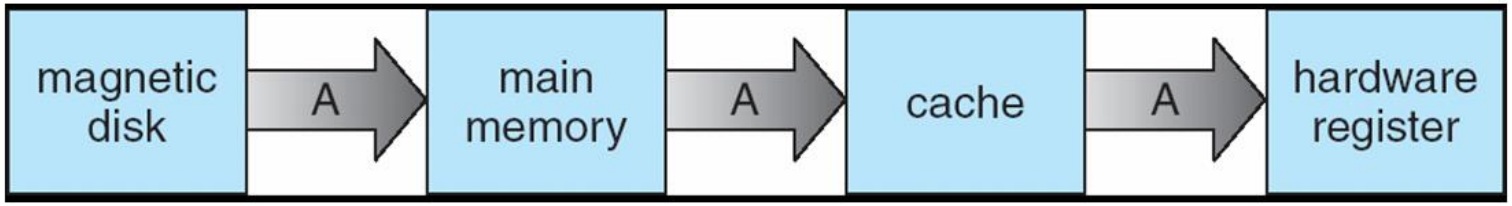


Mass-Storage Management

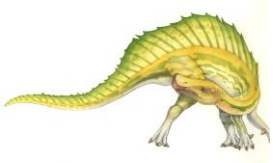
- Usually disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time
- Proper management in general is of central importance
  - Entire speed of personal computer operation may hinge on disk subsystem and its algorithms
  - Often not a central issue in embedded systems

Migration of Integer A from Disk to Register

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

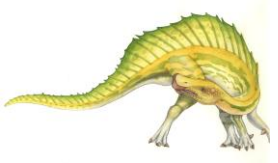


- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist



I/O System: overview

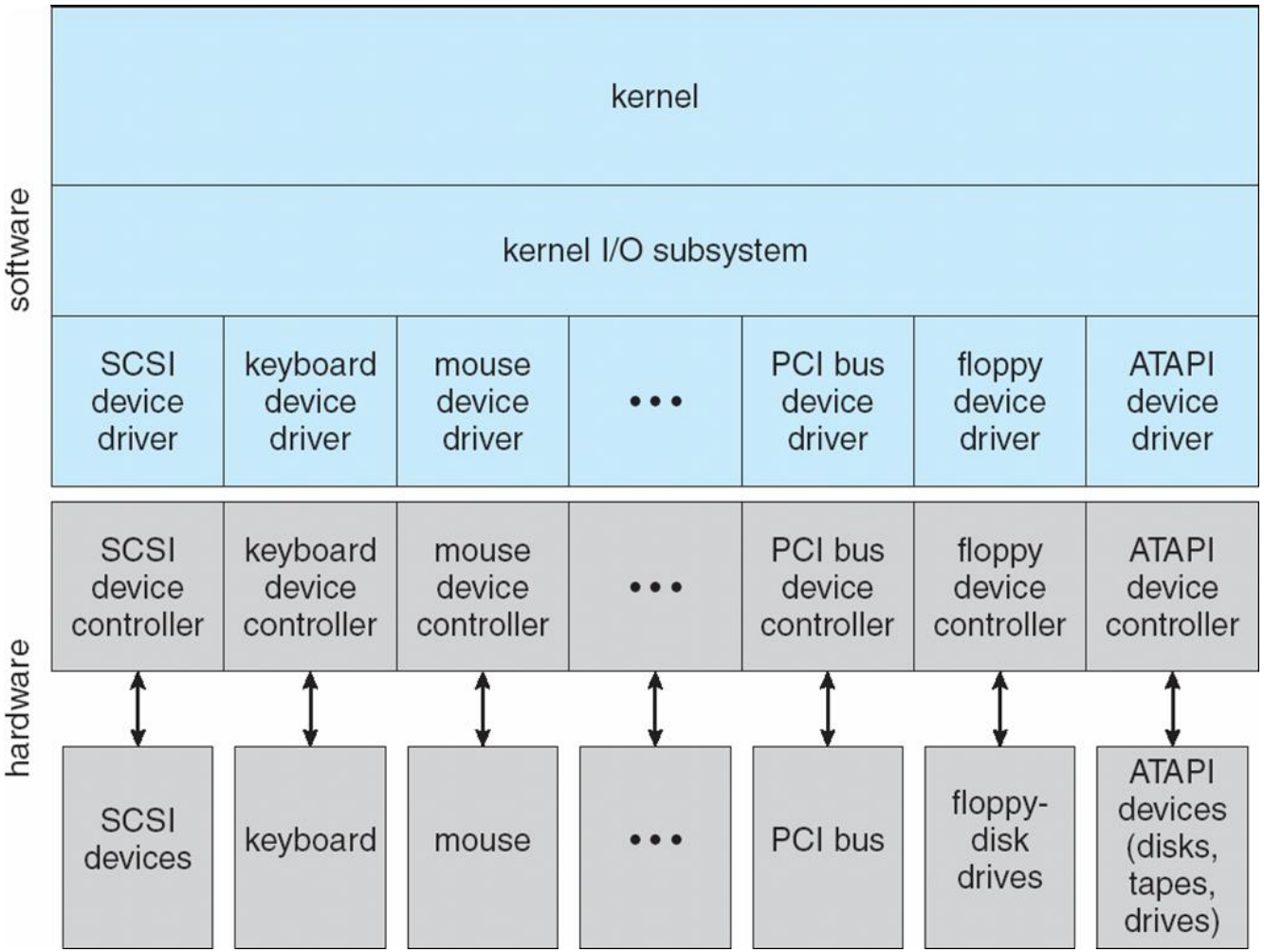
- I/O management is a major component of operating system design and operation
  - Important aspect of computer operation
  - I/O devices vary greatly. Various methods to control them
  - The OS masks device heterogeneity behind a unique interface
  - New types of devices frequent
- Hardware: ports, busses, device controllers connect to various devices
- Software: **Device drivers** encapsulate device details
  - Present uniform device-access interface to I/O subsystem



I/O Subsystem

- 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 (overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

A Kernel I/O Structure



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

Quizzes

- Multiprogramming and multitasking are the same thing
- An operating system's process management activities include suspending and resuming processes
- An operating system should never keep track of which parts of memory are currently being used and by whom
- Caching is used to increase the size of a system's memory
- Buffering is used for transferring data
- Protection is concerned with defending a system against internal and external attacks
- What matters in real-time embedded systems is performance

In the first semester…

- Processes and threads
  - Process life-cycle
  - Scheduling
  - Inter-process communication
- Process synchronization
  - Deadlocks
  - Elements of concurrent programming
  - Patterns & classical synchronization problems
- Memory management
  - Virtual memory management
  - Kernel memory management