Chapter 1: Introduction (Part 1)

Chapter 1: Introduction

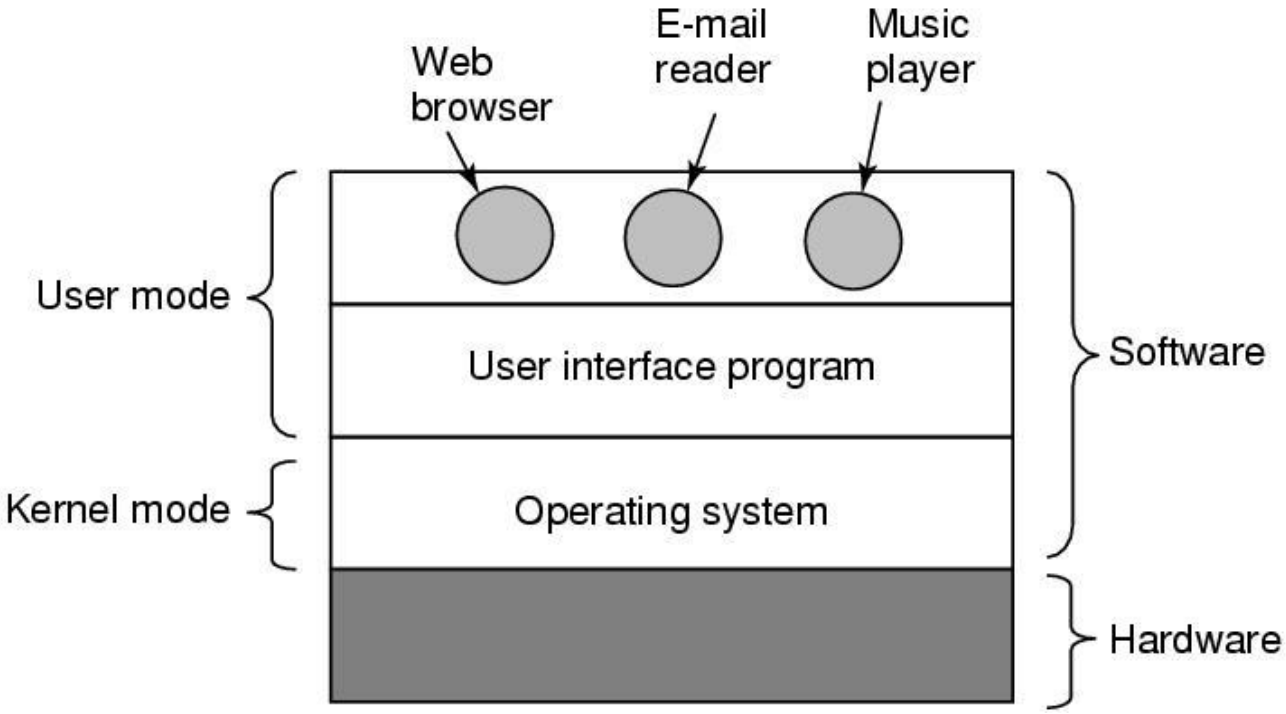
- **What Operating Systems Do**
- Computer-System Organization
- Computer-System Architecture
- Operating-System Structure
- Operating-System Operations
- Process Management
- Memory Management
- Storage Management

What is an Operating System?

- A modern computer consists of:
  - One or more processors
  - Main memory
  - Disks
  - Printers
  - Various input/output devices.
- Managing all these varied components requires a layer of software – the **Operating System (OS).**

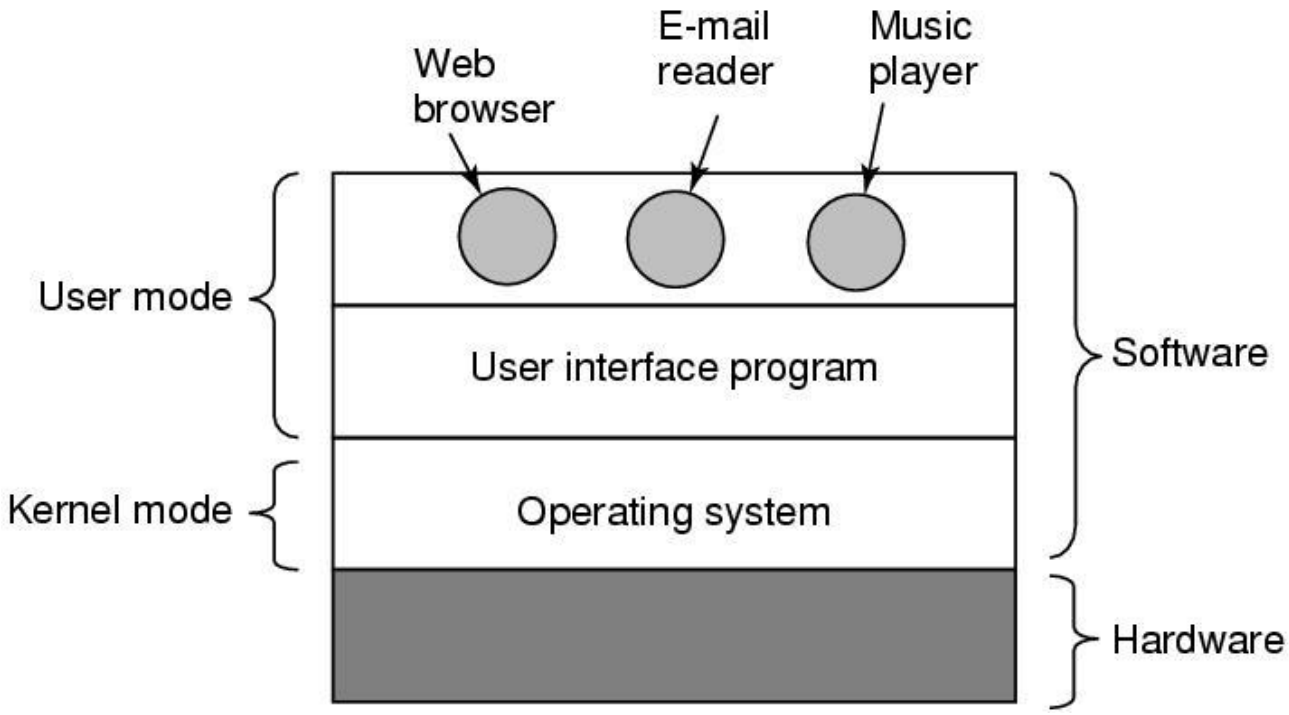
What is an Operating System?

- **A program that acts as an intermediary between a user of a computer and the computer's hardware**
  - **Manage** the hardware resources
  - Support applications; games, business, programs, …, etc
  - Designed to **optimize** hardware utilization



What is an Operating System?

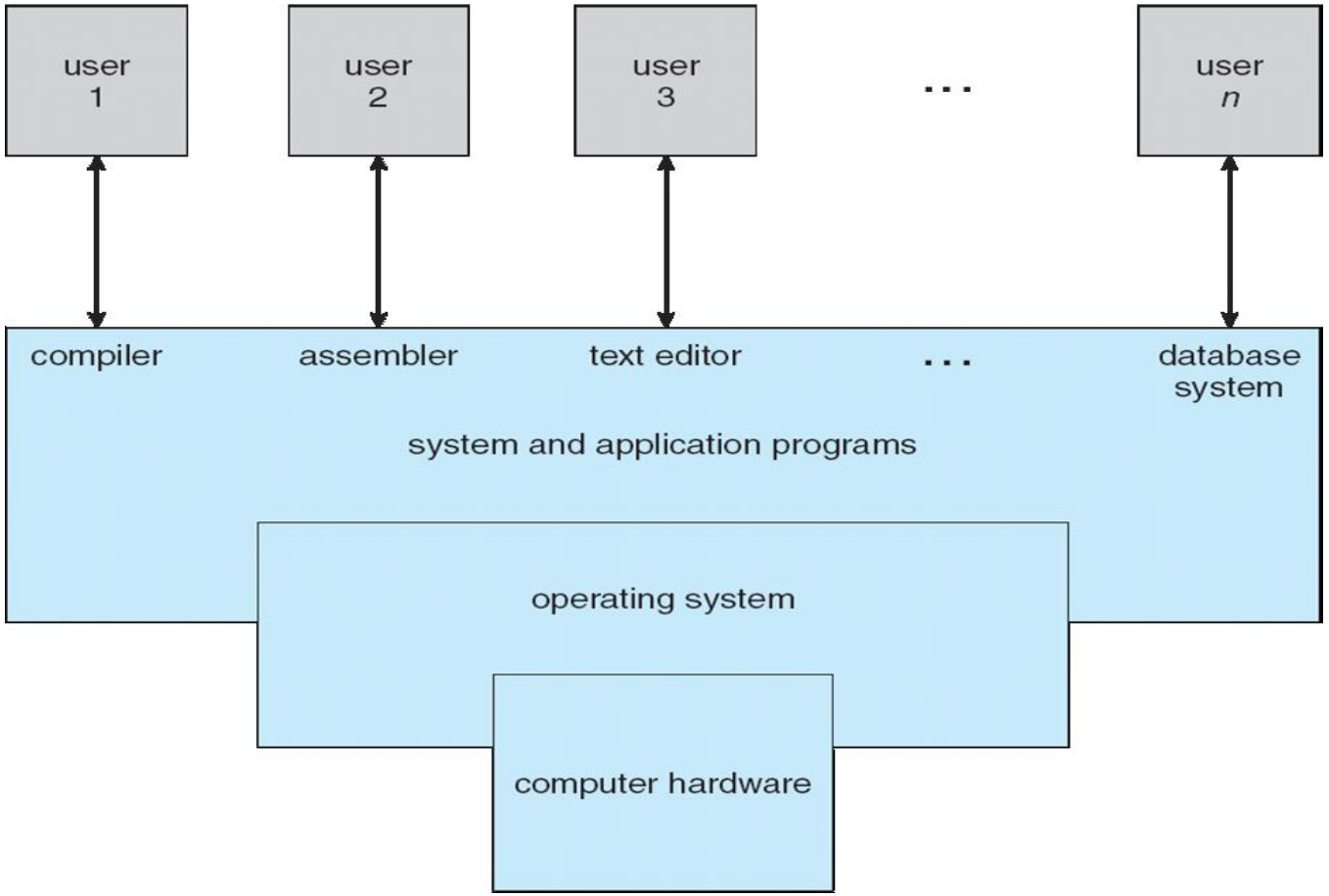
- **Operating system goals:**
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner



Computer System Structure

- Computer system can be divided into four components:
  - **Hardware** – provides basic computing resources
    - CPU, memory, I/O devices (printer, keyboard, monitor, etc)
  - **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
- **Computer system = hardware, software, and data**.
  - Operating system provides means for proper uses of these resources

Four Components of a Computer System



What Operating Systems Do?

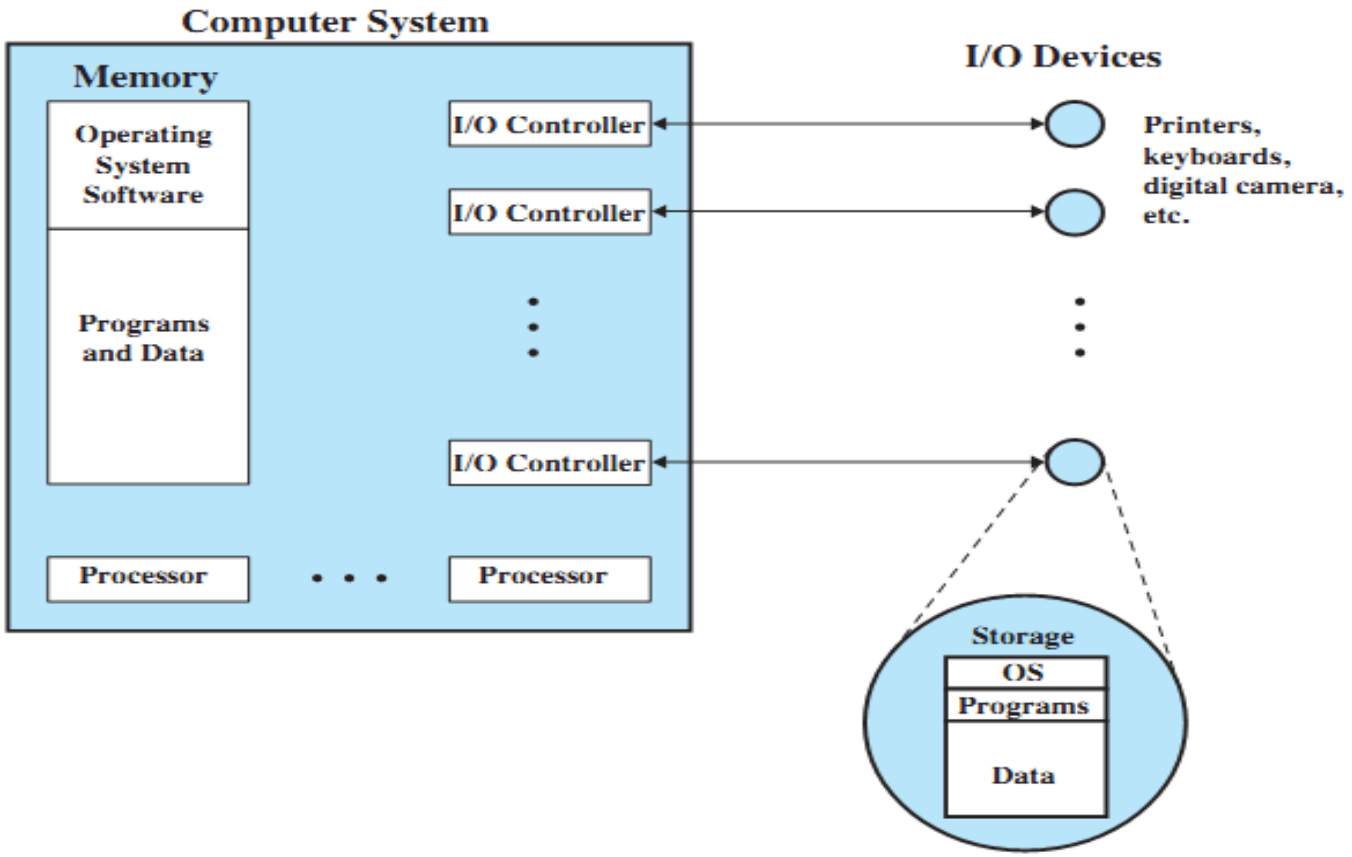
- Users want convenience, **ease of use** and **good performance**
  - Users don't care about **resource utilization**
  - OS maximizes resource utilization, assures efficient use of resources
- Users of dedicated systems such as **workstations** have dedicated resources but frequently use shared resources 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 automobiles etc.

Operating System Definition

- OS is a **resource allocator**
  - Manages all resources
    - CPU time, memory space, I/O devices, …, etc
  - Decides between conflicting requests for efficient and fair resource use
    - When many users (processes, programs) access to the same resources
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer
    - For example: Fetch-and-Execute Cycle of the CPU

OS as a Resource Manager

- **Resource Manager:**
  - Manages and protects multiple computer resources: CPU, Processes, Internal/External memory, Tasks, Applications, Users, Communication channels, etc…



Operating System Definition (Cont.)

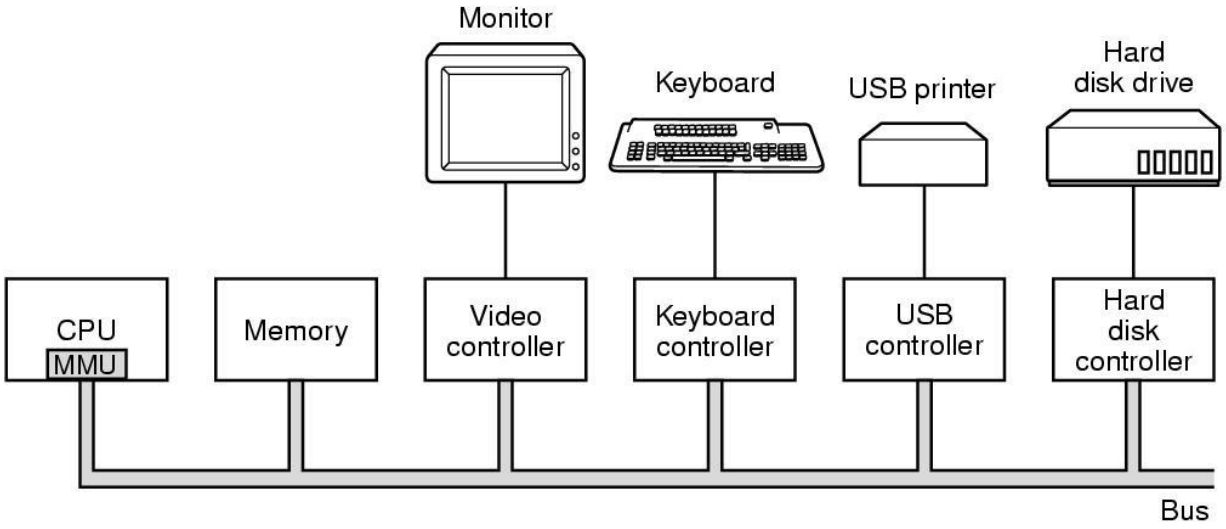
- No universally accepted definition
  - OSs exist to create a usable computing system
- “Everything a vendor ships when you order an operating system” is a good approximation
- “The one program running at all times on the computer” is the **kernel**.
- Everything else is either
  - a system program (ships with the operating system) , or
  - an application program.

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

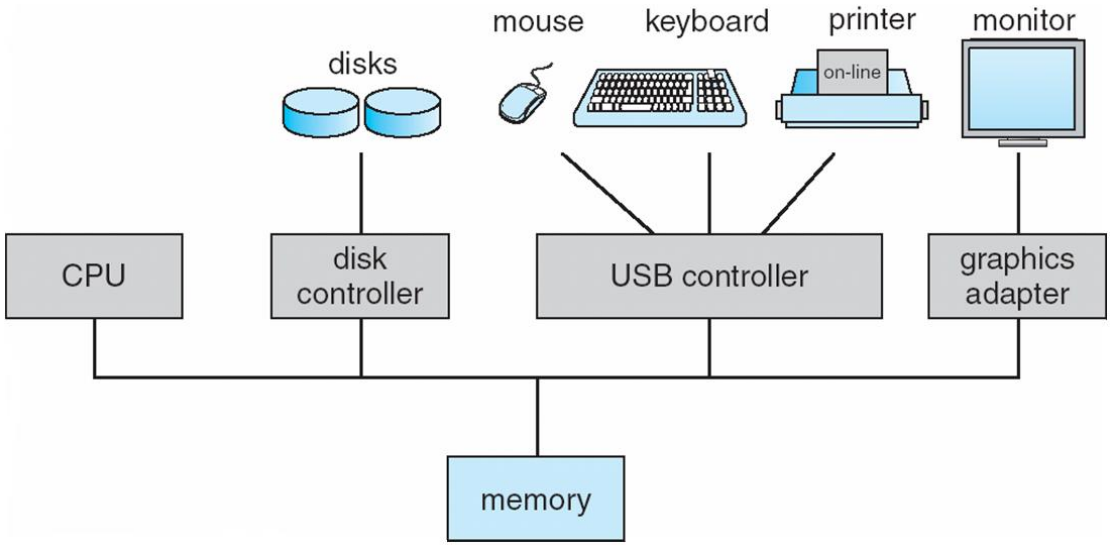
Computer System Organization

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



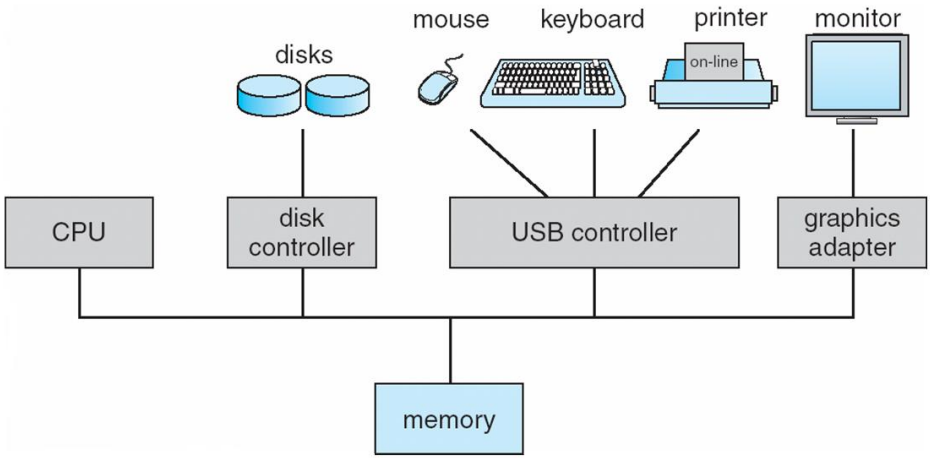
Computer System Organization

- Computer-system operation
  - **Processor:** Controls the operation of the computer, performs the data processing functions, referred to as the Central Processing Unit (CPU)
  - **I/O Modules:** Moves data between the computer and external environments such as, hard drive, communication equipment, terminals.
  - **System Bus:** Provides communication among processors, main memory, and I/O modules

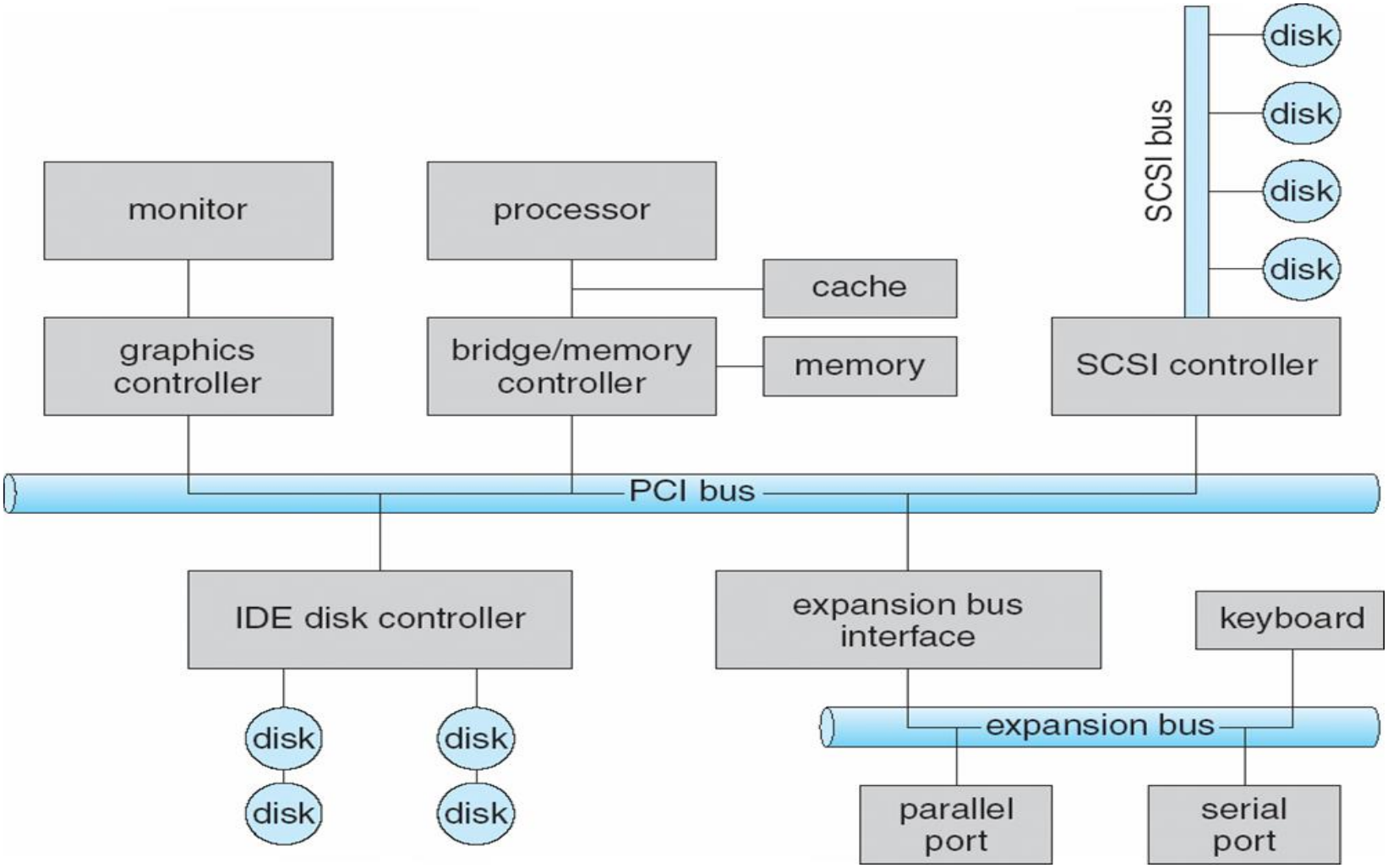


Computer System Organization

- Computer-system operation
  - **Main Memory:** Volatile, Contents of the memory is lost when the computer is shut down, also referred to as **real memory** or **primary memory**
  - Concurrent execution of CPUs and devices competing for memory cycles



A Typical PC Bus Structure

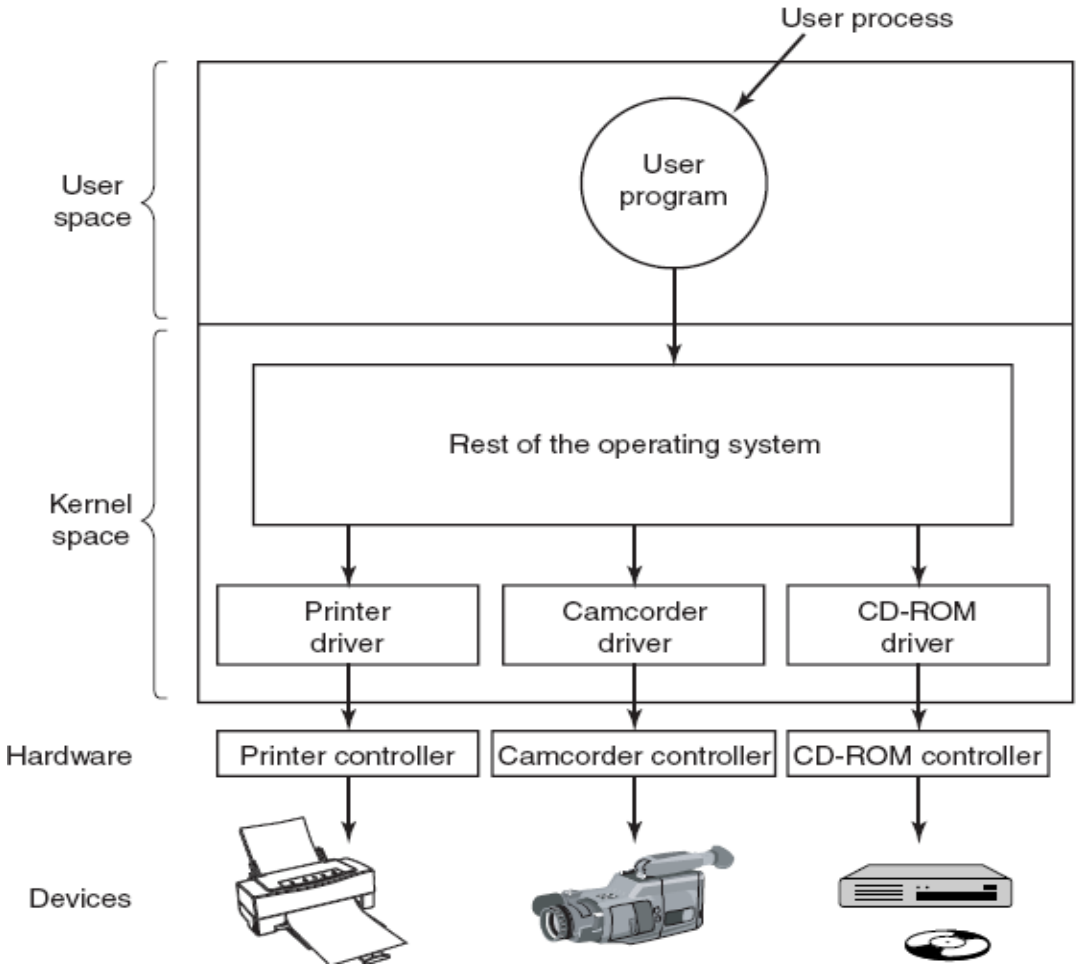


Devices and Device controller

- A computer system contains a multitude of I/O devices and their respective controllers, e.g., network card, graphics adapter, disk controller, DVD-ROM controller etc.
- **I/O devices** generally consists of two parts:
  - An Electronic component: Device Controller
  - A Mechanical component: The device itself
- Each device controller is in charge of a specific device
- It maintains
  - Some local storage buffer
  - A set of special purpose register
  - Accepts command from the Operating System

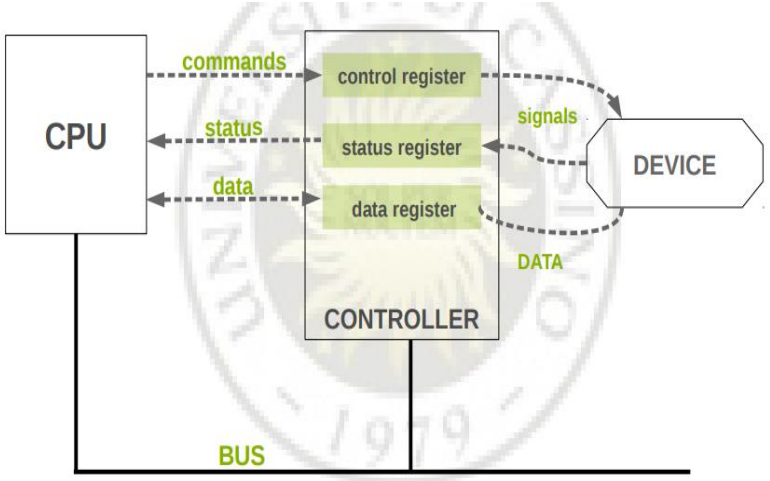
Devices and Device controller

- Every I/O device is managed by the OS through its controller



Devices and Device controller

- Interacting with Device Controllers
  - command registers (write-only);
  - status registers (read-only)→Busy, idle, malfunction
  - data registers (read/write).
- For example, to read data from the hard drive.
  - Device controller might accept a command such as
  - Read
  - Sector 11,206
- The device controller would:
  - Determine the current position of the head
  - Move the head to the required location
  - Accept data bit by bit
  - Store in a local buffer
  - Perform checksum (error checking) on the data
  - Controllers contain small embedded programs to carry out all this work.

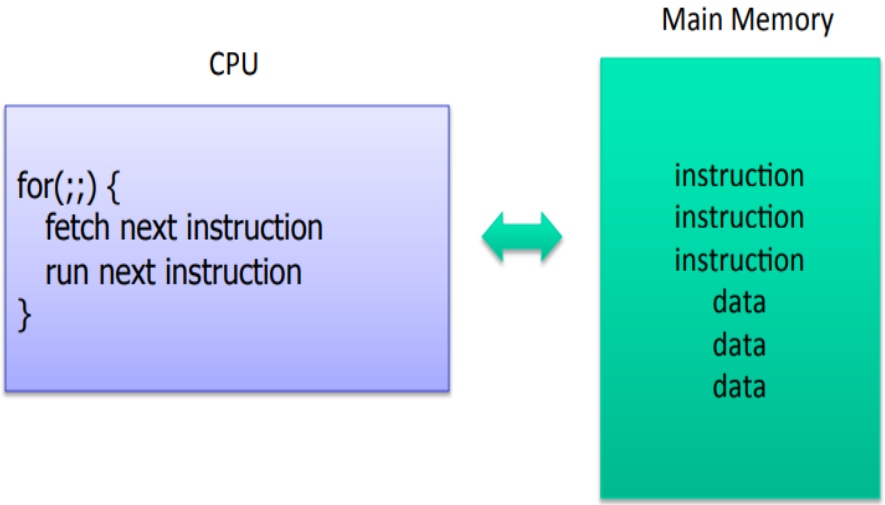


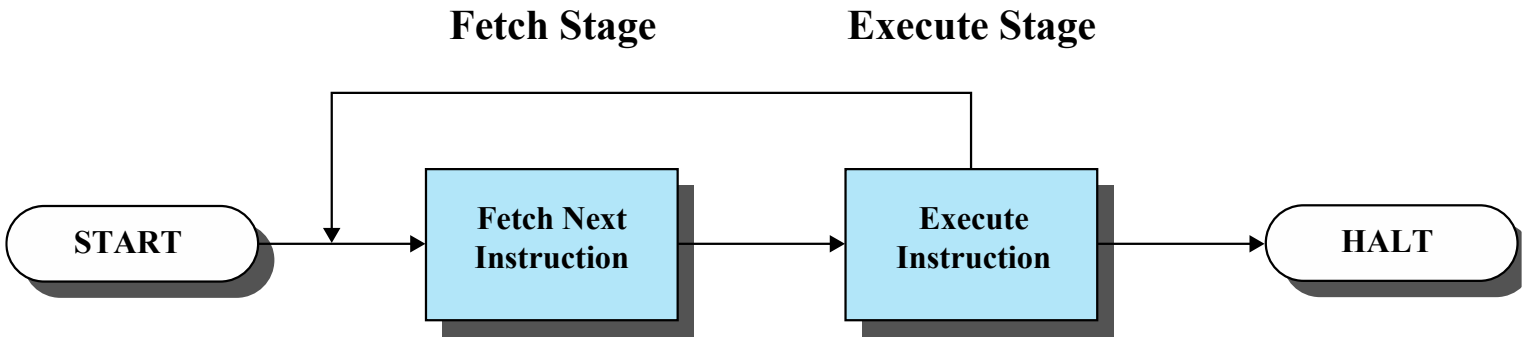
Device Drivers

- ***Device Driver*** is a software that initiates the ***device controller***.
- Interaction with a ***device controller*** is managed through a ***device driver***.
- Device drivers are part of the operating system, but not necessarily part of the OS kernel.
- In some cases, the operating system buffers data that are transferred between a device and a userspace program. This usually increases performance, but not always.
- ***Different device drivers*** for different types of ***device controllers***.

CPU Normal Execution (Fetch and Execute Cycle)

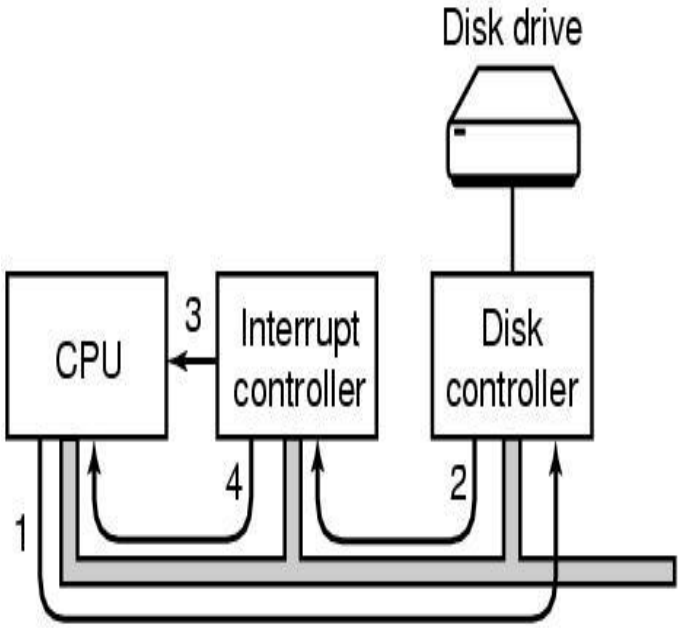
- The CPU is the arithmetic and logic engine that **executes user applications**.
- A program consists of a set of instructions stored in memory
  - processor reads (fetches) instructions from memory
  - processor executes each instruction





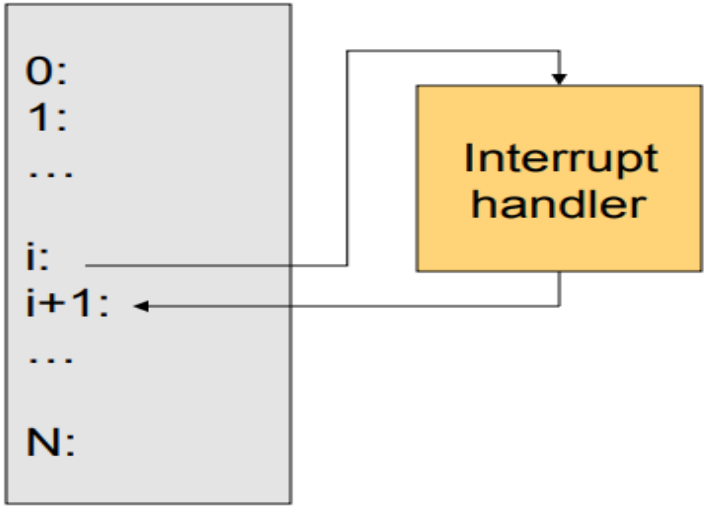
Interrupts

- Interrupts plays an important role
- I/O devices and the CPU can execute **concurrently**
- Each device controller has a **local buffer**
  - Data movement (I/O) between device and local buffer (**by device**)
  - Data movement between memory and local buffer (**by CPU**)
- Device controller informs CPU that it has finished its current operation or it has something
  - How does I/O device inform CPU?
    - by causing an interrupt



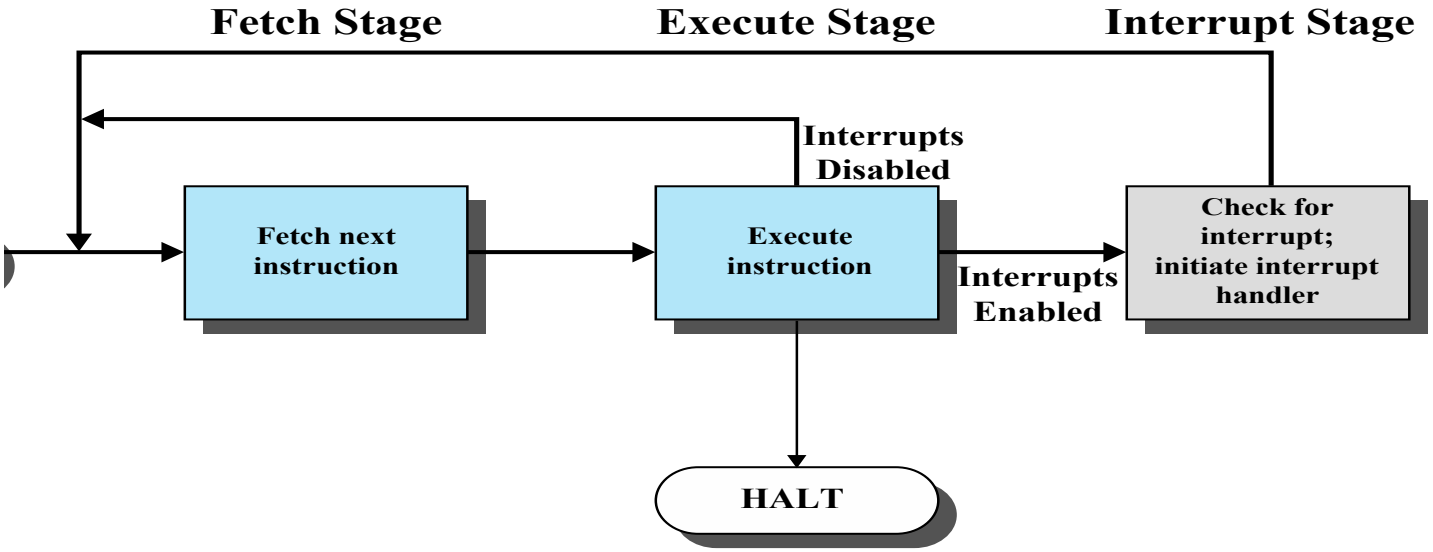
Interrupts

- **Definition:** Interrupts are signals sent to the CPU by I/O devices. They tell the CPU to stop its current activities and execute the appropriate part of the operating system (handler, aka **interrupt service routine** or ISR).
- An ***event*** external to the currently executing process that causes a change in the normal flow of instruction execution
- Raised by external events
- Interrupt handler is in the kernel
  - Switch to another process
  - Overlap I/O with CPU
- Why Interrupts?
  - Isn't it expensive to stop a process, switch to another process, and then resume the suspended process?
    - Interrupts are important because they give the user better control over the computer.



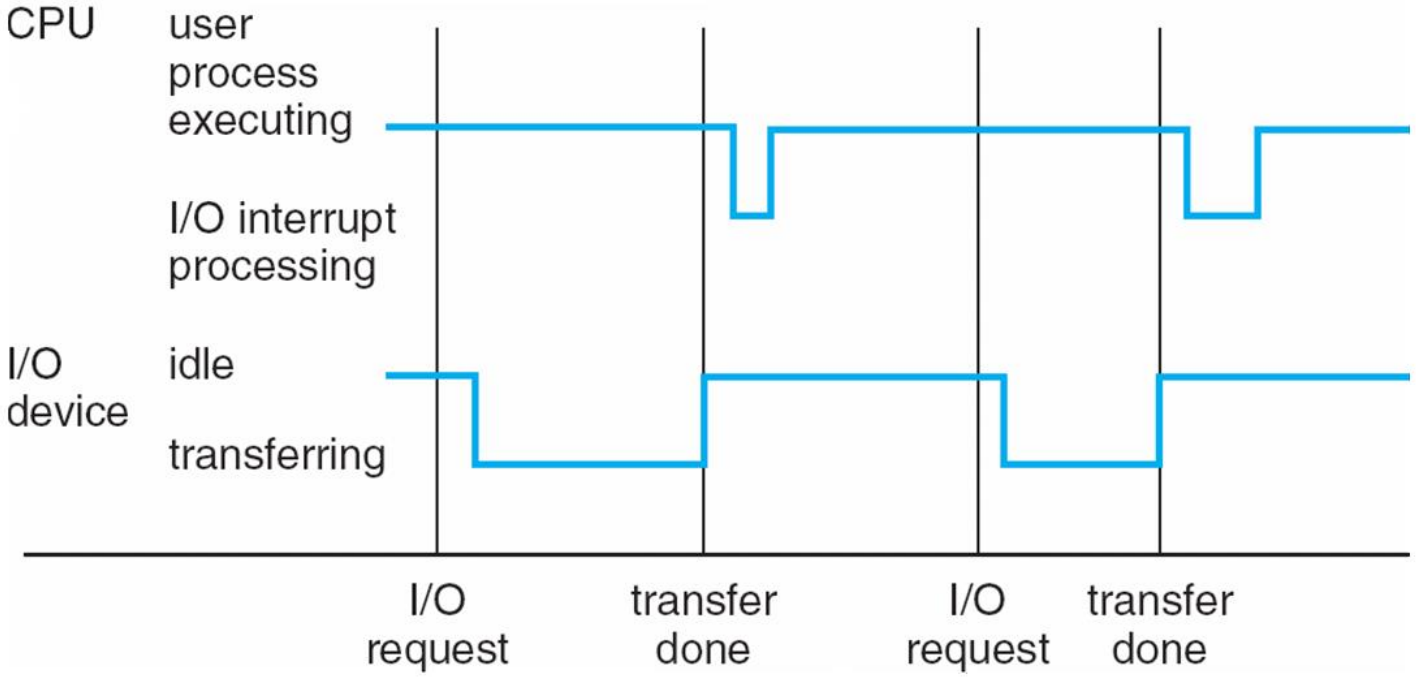
Instruction cycle with Interrupts

- The ***interrupt-handler*** routine is generally part of the OS.
- Typically, this routine determines the nature of the interrupt and performs whatever actions are needed.
- The handler determines which I/O module generated the interrupt and may branch to a program that will write more data out to that I/O module.
- When the interrupt-handler routine is completed, the processor can resume execution of the user program at the point of interruption.



Interrupt Timeline

- I/O device controller moves data between the device and its local buffer
- To start an I/O operation:
  - CPU loads the appropriate registers in the device controller
  - The device controller examines these registers to determine what actions to take (e.g., **read request** → start transferring data from device to buffer, **write request** → start transferring data from buffer to device)
  - When action is completed, the device controller informs the CPU via an interrupt



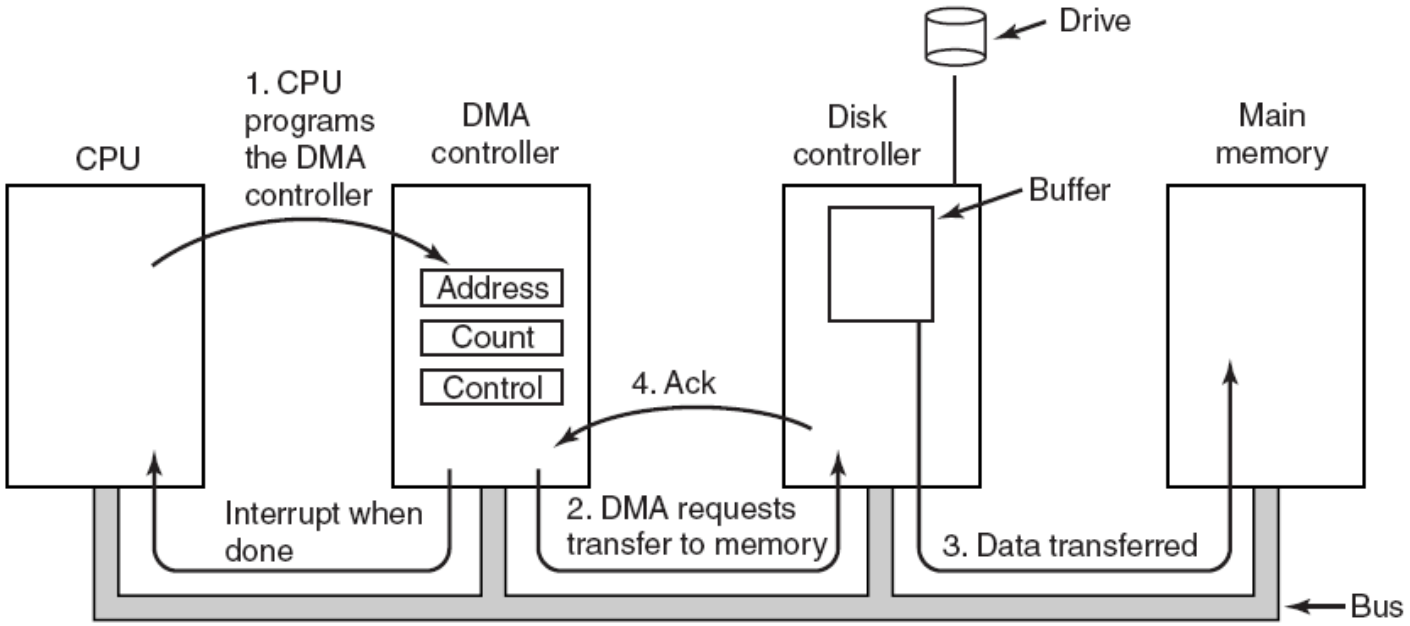
Direct Memory Access (DMA)

- DMA is used by smart high-speed I/O devices able to transmit information at close to memory speeds.
- A ***DMA controller*** allows devices to transfer data to or from the main memory without the intervention of the processor.
- DMA 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 one interrupt per byte.
- Hence processor is involved only at the beginning and end of the transfer
- An interrupt is sent when the task is complete

Direct Memory Access Structure

- **Simple keyboard processing:**
  - User types character at keyboard
  - Keyboard controller sends interrupt to CPU
  - CPU must complete current instruction, then save state
  - Control is transferred to interrupt service routine (handler), which:
    - Stores char in a buffer & increments buffer pointer
    - Sets flag to notify OS that input is available (can be transferred to requesting program)
  - Must restore CPU state and resume processing
- **For high-speed I/O devices, the combined overhead of the interrupts may be too costly**
  - **Direct Memory Access:** a device driver assigns a specific memory segment to the device controller
  - Device controller can transfer an entire block of data directly to/from main memory without CPU intervention.
  - Only one interrupt is generated per block, rather the one interrupt per byte

Direct Memory Access (DMA)



- A DMA controller has:
  - Memory address register
  - Byte count register
  - Control registers to specify the I/O port to use, and the direction of the transfer such as reading or writing from/to the I/O device.

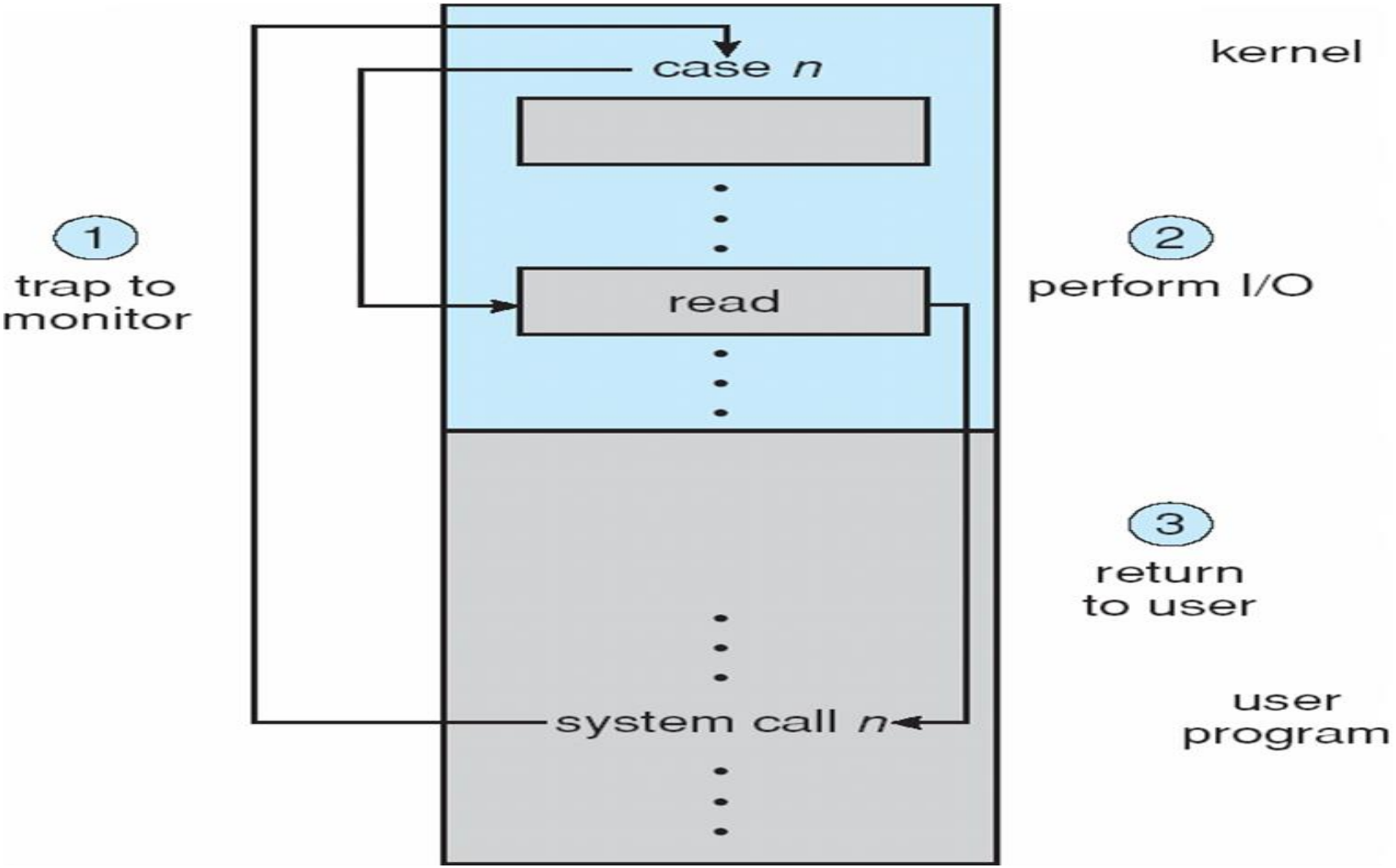
I/O Protection

- User process may accidentally or purposefully attempt to disrupt normal operation of system via illegal I/O instructions.
- All I/O devices need to be protected from wrongdoing by the user processes.
- All I/O instructions need to be privileged instructions.
- Given that the I/O instructions are privileged, how does the user program perform I/O?
- **Solution:** System Calls (from programs).

System Call

- The *method* used by a user program to request action by the operating system, e.g., read(), write() etc.
- After system call parameter preparations, it uses the trap instruction to transfer control to the requested service routine in the OS.
- The system verifies that the parameters are correct and legal, and executes the request.
- Returns control to the instruction following the system call.

System Call to Perform I/O



System Call

- User processes cannot perform privileged operations themselves
- Must request OS to do so on their behalf by issuing system calls
- Basic concepts (today), more details on how done (later)

Types of Interrupts

- Modern Operating systems are interrupt-driven.
- There are three types of interrupts:
  - Hardware Interrupts (External Interrupts) are generated by hardware devices to signal that they need some attention from the OS. They may have just received some data (e.g., keystrokes on the keyboard or an data on the Ethernet card); or they have just completed a task which the operating system previous requested, such as transferring data between the hard drive and memory.
  - Software Interrupts are generated by programs when they want to request a ***system call*** to be performed by the operating system.
  - Traps (Exceptions) are generated by the CPU itself to indicate that some error or condition occurred for which assistance from the operating system is needed.