Operating System
CS-2006
Lecture 1

Mahzaib Younas

Lecturer Department of Computer Science

FAST NUCES CFD

What is 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

Structure of Computer System

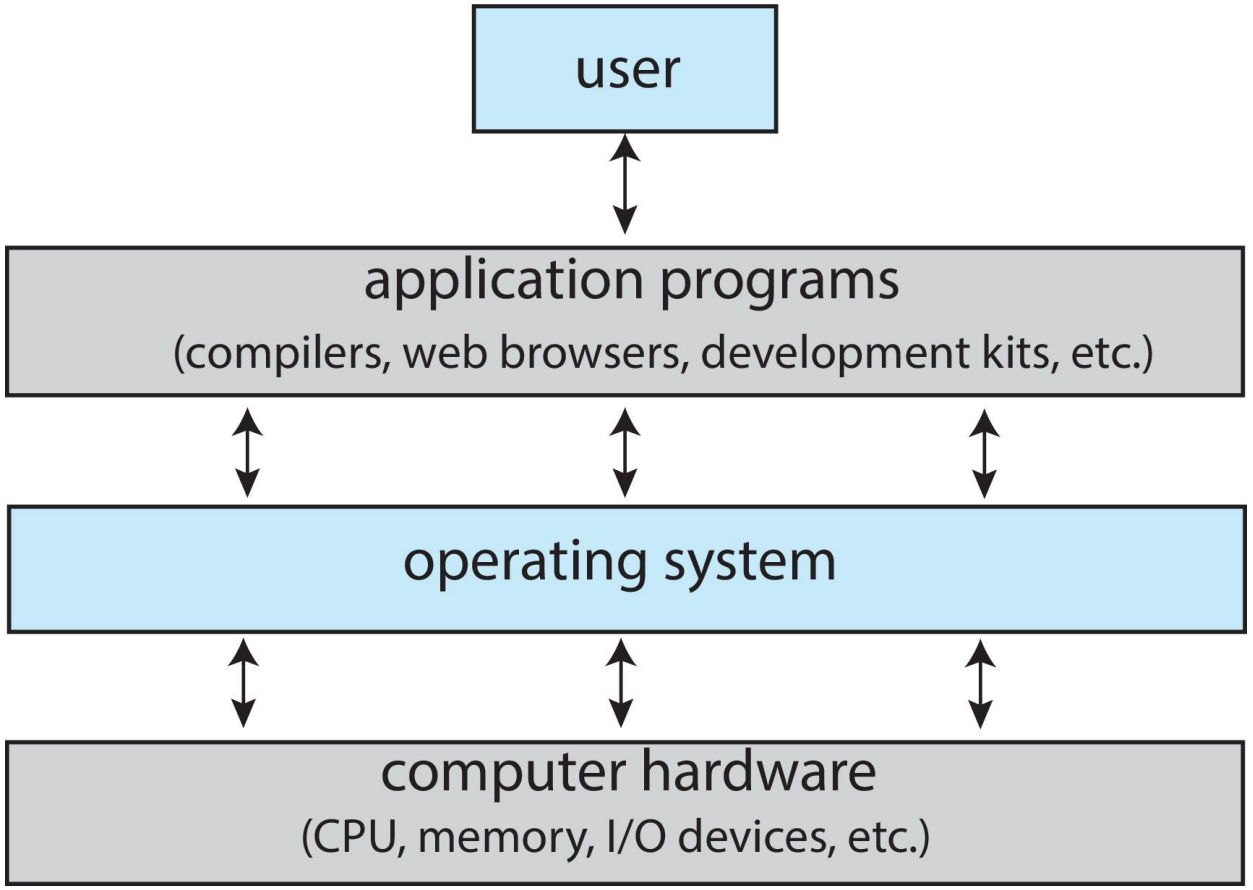
- Computer system can be divided into four components:

1. Hardware
2. Operating System
3. Users
4. Application Programs

Structure of Computer System

- **Hardware**
  - Provide basic Computing resources
  - Example
    - CPU
    - Memory
    - I/O
- **Operating Systems**
  - Controls and coordinates use of hardware among various applications and users
  - Example
    - Linux
    - Windows
- **User**
  - Who interact with computer
  - Example
    - People
    - Machine
    - Other Computers
- **Application Programs**
  - define the ways in which the system resources are used to solve the computing problems of the users
  - Example
    - Word Processor
    - Compiler
    - Database

Abstract View of Computer System



What an Operating System Can do?

- Uses/Purpose of Operating Systems
  - **User point of view**
    - In user point of view users want convenience in term of
    - Good performance (not care about resource utilization)
  - **Shared Point of view**
    - In shared Computer such as mainframe or minicomputer
    - Important to make all users happy
    - OS make efficient use of resource allocator and control program
  - **Dedicated Path**
    - In dedicated path such as workstation
    - Frequently provide the resources from server without any delay

What an Operating System Can do?

- Uses/Purpose of Operating Systems
  - **Mobile Devices**
    - In mobile devices such as smart phones and tablets
    - Optimized for usability of battery life
    - User Interface Like touch screen
  - **Embedded Systems** [perform a specific task for a device]
    - No user interface system Embedded System in device
    - Run primarily without user intervention

Basic Operating System Definition/Terms

1. ***Kernel***
   - Program that run all the time on computer
   - Primary Interface between hardware and process

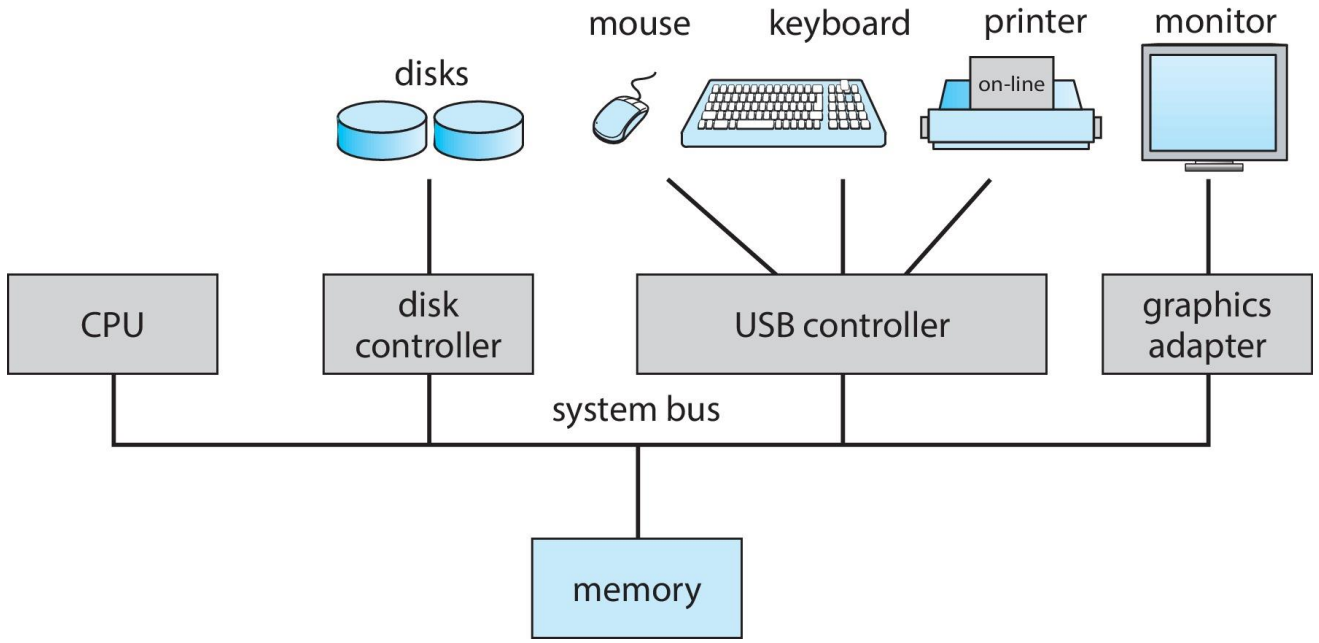
2. ***System Program***
   - Associated with operating system but not part of kernel

3. ***Application Program***
   - all programs not associated with the operating system

4. ***Middleware***
   - Middleware is software that lies between an operating system and the applications running on it.
     - Example: databases, multimedia, graphics

Computer System Operations

- Basic Operations
  - Device Controller
  - Concurrent Execution



device controllers connect through common bus providing access to shared memory

Concurrent execution of CPUs and devices competing for memory cycles

Computer System Operations (Cont…)

- Concurrent Operations
  1. I/O and CPU can execute concurrently
  2. CPU moves data from/to main memory to/from local buffers

- Device Controller
  1. Each device controller is in ***charge of a particular device type***
  2. Each device controller has a ***local buffer***
  3. Each device controller type has ***an operating system device driver*** to manage it

Buffer

The buffer is an area in the main memory used to store or hold the data temporarily.

Interrupts

An interrupt is a signal emitted by hardware or software when a process or an event needs immediate attention

- Operating system is an interrupt driven.
- Basic Terms of Interrupt
  1. **Interrupt Vector:**
     - Transfer control to interrupt service
     - Contain address of all services
  2. **Interrupt architecture**
     - Save the address of interrupted instruction
  3. **Interrupt trap/execution**
     - software-generated interrupt caused either by an error or a user request

Trap

A trap is a synchronous interrupt triggered by an exception in a user process to execute functionality.

Interrupt

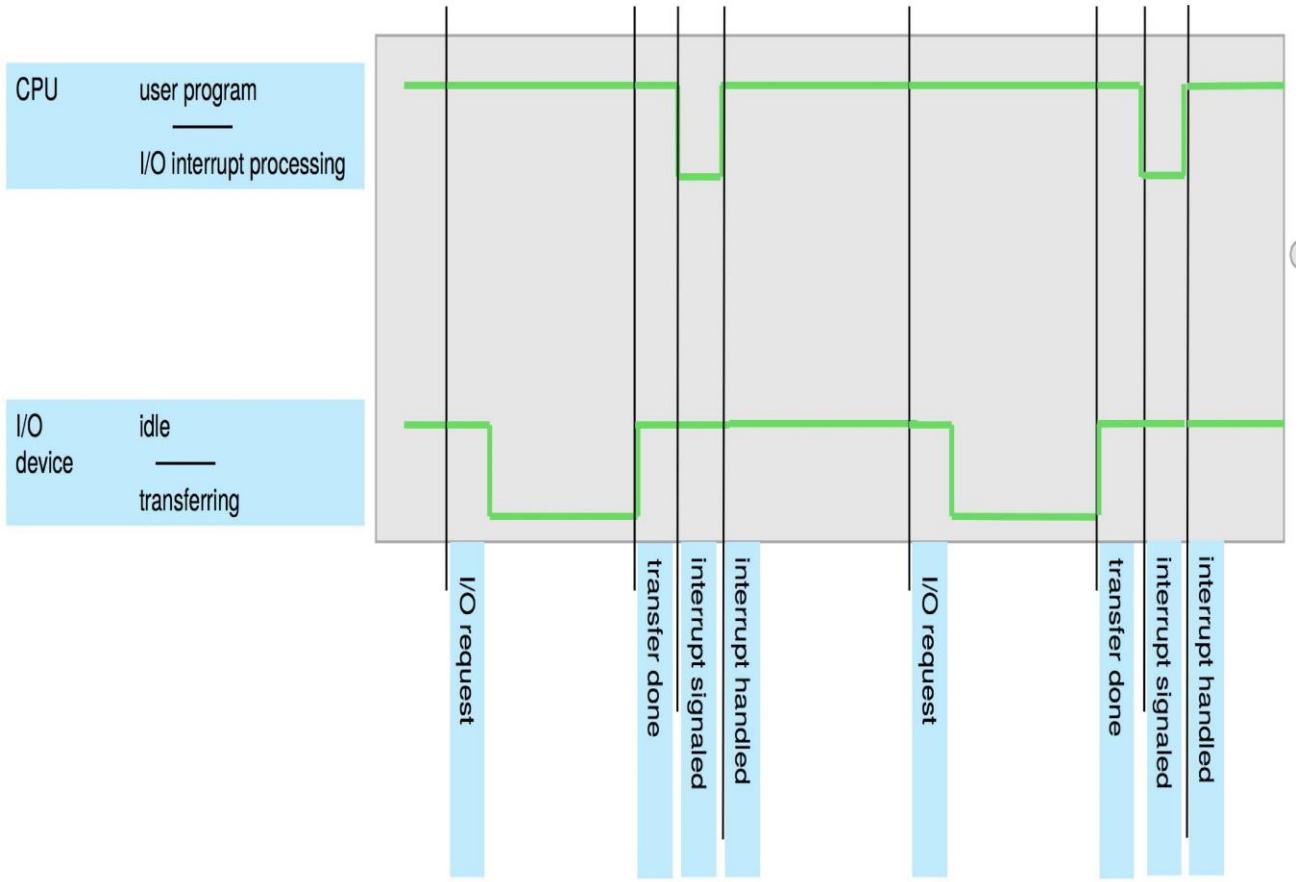
An interrupt is a hardware or software signal that demands instant attention by an OS

Interrupt Timeline

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

2. Determines which type of interrupt has occurred:

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



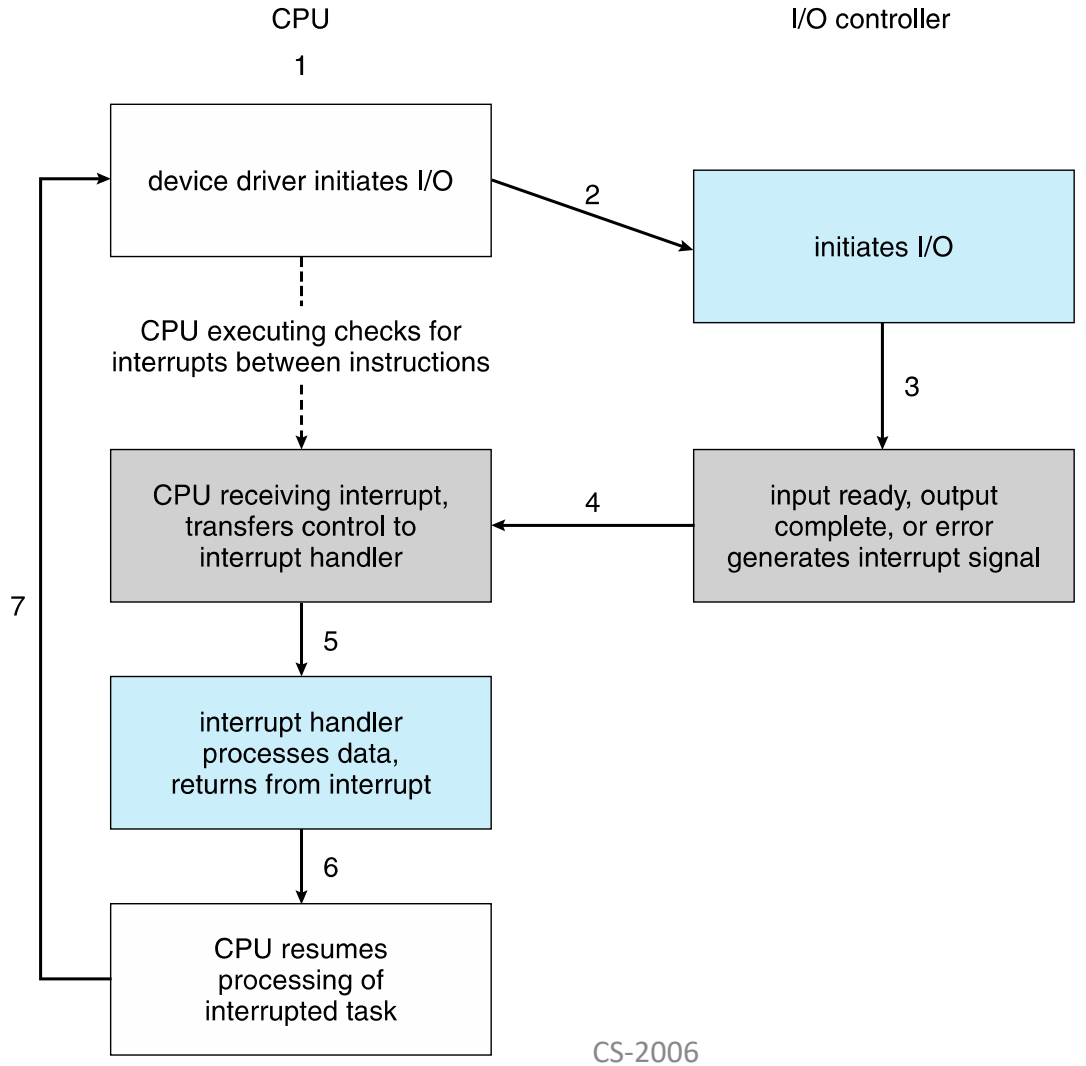
Interrupt and Trap

Interrupt and trap generated to get OS attention? Is it true of false.

Solution/Reason:

Interrupt is hardware generated signal. So when hardware wants to send data it generates signal that get the attention of OS in order to complete the transfer. On the other hand trap is software generated signal that inform the operating system about certain errors, or some services to be required.

Interrupt Driven I/O cycle



I/O Structure

Method 1: wait for completion

After I/O starts, control returns to user program only upon I/O completion

1. Wait instruction idles the CPU until the next interrupt
2. Wait loop (contention for memory access)
3. At most one I/O request is outstanding at a time, no simultaneous I/O processing

Method 2: without wait for completion

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

1. **System call** – request to the OS to allow user to wait for I/O completion
2. **Device-status table** contains entry for each I/O device indicating its type, address, and state
3. OS indexes into I/O device table to determine device status and to modify table entry to include interrupt

Bootstrap

- Bootstrap means to power on the system, which
  - Initialize all basic aspects/software of system
  - Load operating system, kernel and start execution

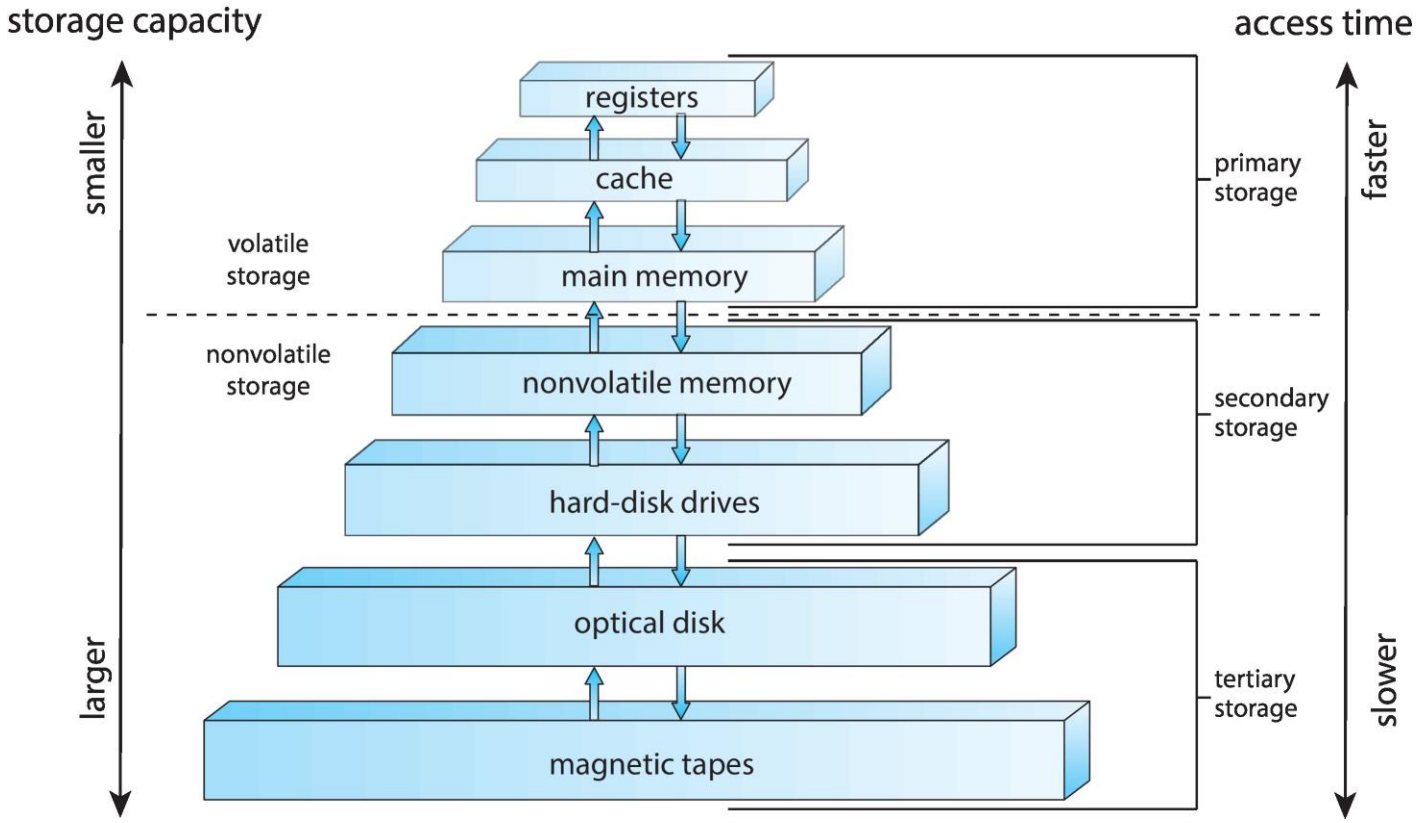
- Firmware
  - The firmware is embedded in the hardware and may not be changed.
  - Its non-volatile.

Storage Structure

- Main Memory
  - only large storage media that the CPU can access directly
  - Random access
  - Typically volatile
  - Typically, in the form of Dynamic Random-access Memory (DRAM)

- Secondary Storage
  - extension of main memory
  - provides large nonvolatile storage capacity

Storage Structure

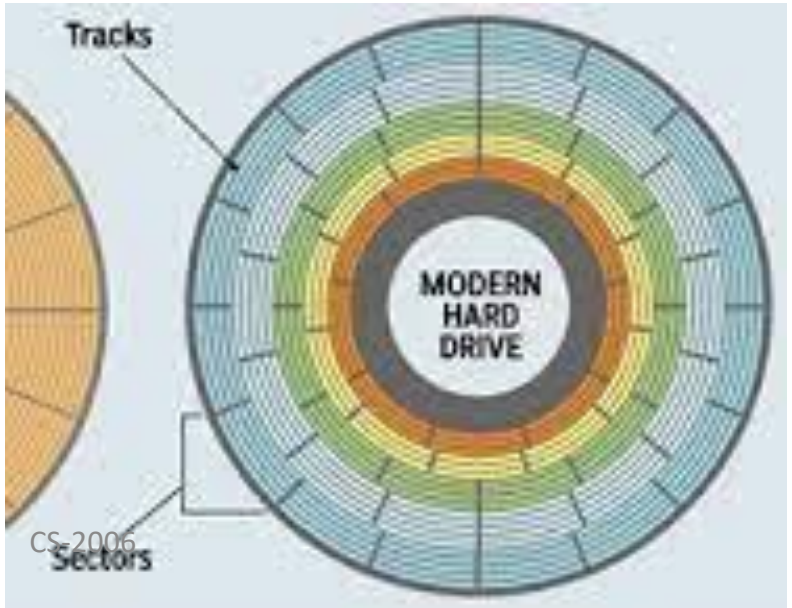


Storage Structure (Cont…)

- **Registers**
  - Fastest type of storage in a computer system, but they are also the smallest.
  - Used to hold data that the CPU needs to access quickly
- **Cache**
  - Small amount of high-speed memory that is used to temporarily store frequently accessed data
- **Main Memory**
  - primary storage used by a computer system.
  - larger than cache memory, but it is also slower.
- **Non-Volatile Memory**
  - Type of computer memory that can retain stored information even after power is removed.
  - Becoming more popular as capacity and performance increases, price drops

Storage Structure (Cont…)

- Hard disk drivers: Type of data storage device that is used in laptops and desktop computers
  - **Tracks**
    - Each platter of a hard disk is divided into a number of tracks.
  - **Sectors**
    - Each track is divided into a number of sectors, each of which can store the same amount of data. A sector is the smallest physical storage unit on the disk, and on most file systems it is fixed at 512 bytes in size

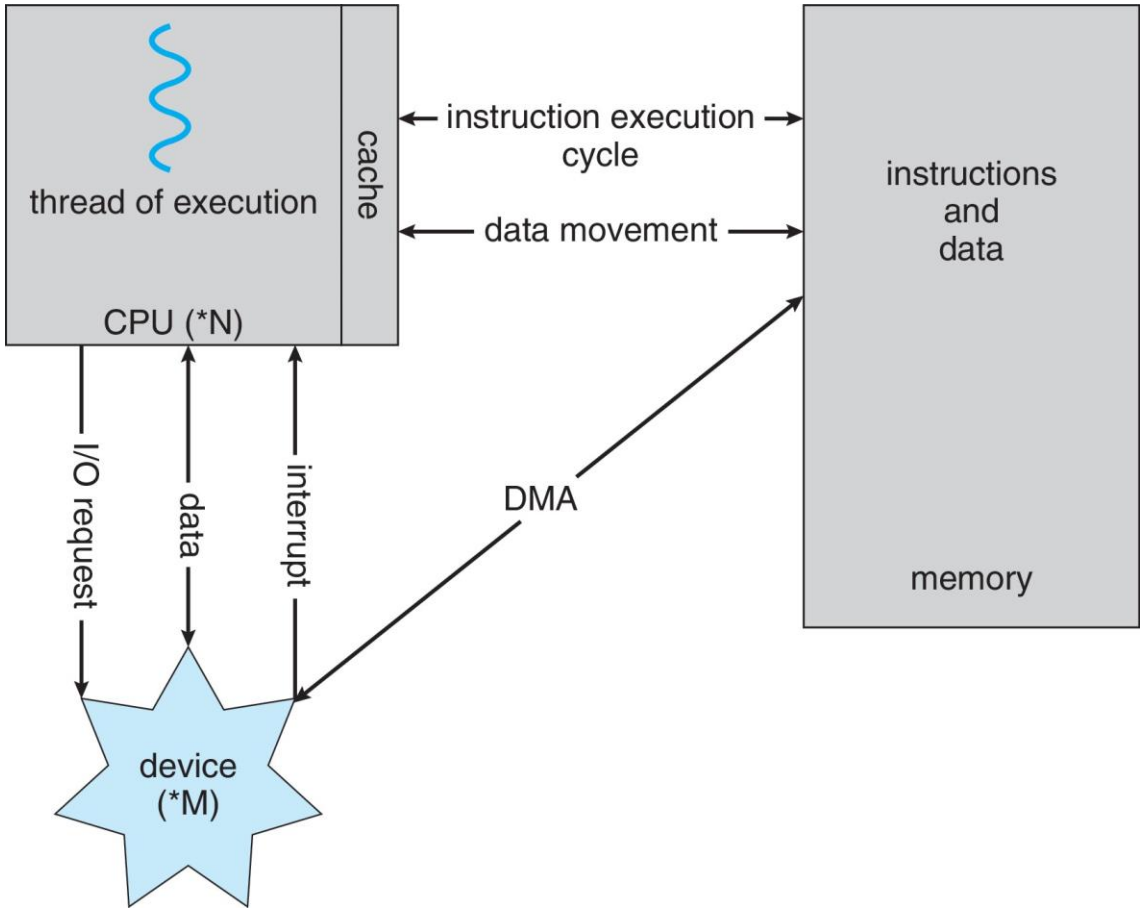


Storage Structure (Cont…)

- **Optical Disk**
  - An optical disk is an electronic data storage medium that can be written to and read from using a low-powered laser beam
  - CD, DVD, Blu-ray, etc., are the examples of optical disks.

- **Magnetic Disk**
  - Magnetic disks are flat circular plates of metal or plastic, coated on both sides with iron oxide.
  - Hard Disk, Floppy Disk, Magnetic Tape, etc. are examples of magnetic disks

How Modern Computer Work?



Direct Memory Access Structure (DMA)

- Direct memory access (DMA) is the process of transferring data without the involvement of the processor itself.
  - Only one interrupt is generated per block, rather than the one interrupt per byte

- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention

Operating System Operations

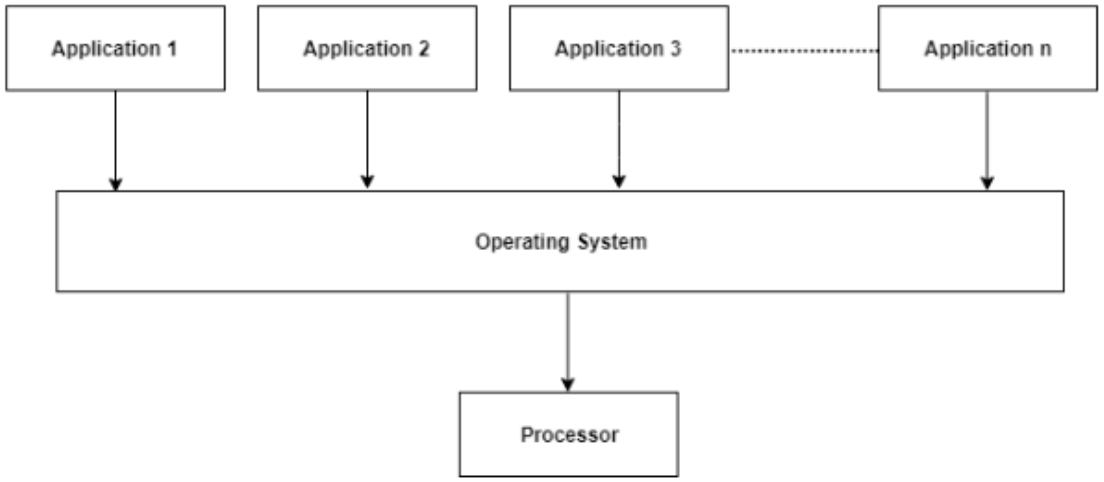
- **Bootstrap Program**
  - simple code to initialize the system, load the kernel
- **Kernel Mode**
  - some privileged instructions that can only be executed in kernel mode.
  - Example: interrupt instructions, input output management
- **System Daemons**
  - Daemon is a type of computer program that runs in the background, performing various tasks without direct interaction from the user.
- **Interrupt Driven**
  - Hardware interrupt by one device
  - Software interrupt are
    - Software error (e.g., division by zero)
    - Request for operating system service – **system call**
    - Other process problems include infinite loop, processes modifying each other or the operating system

Computer System Architecture

- Single-Processor System
- Multiprocessor System
- NUMA

Single Processor Unit

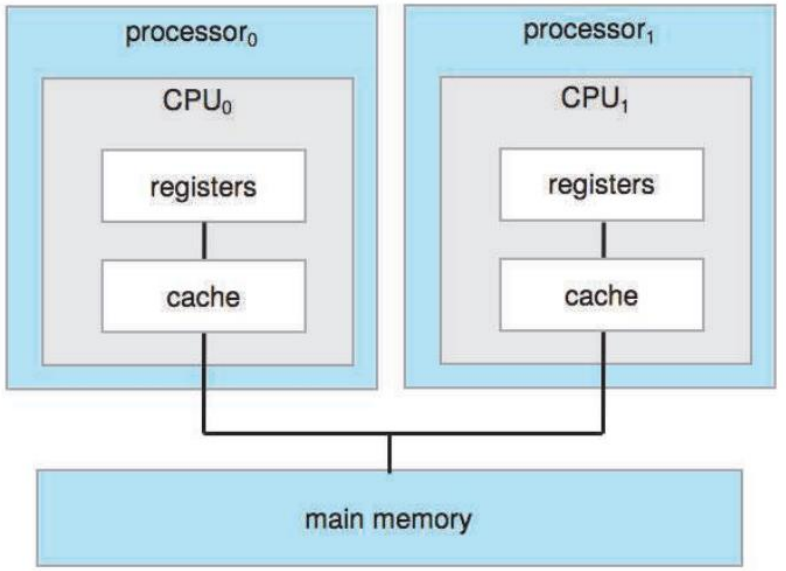
- System with single processor that contain one CPU with Single processing core.
- A single processor system can be further described using the diagram below



Single Processor System

Multi Processor Unit

- systems have multiple processors working in parallel that share the computer clock, memory, bus, peripheral devices etc.
- Most common multi processor system is SMP (Symmetric Multi Processing)



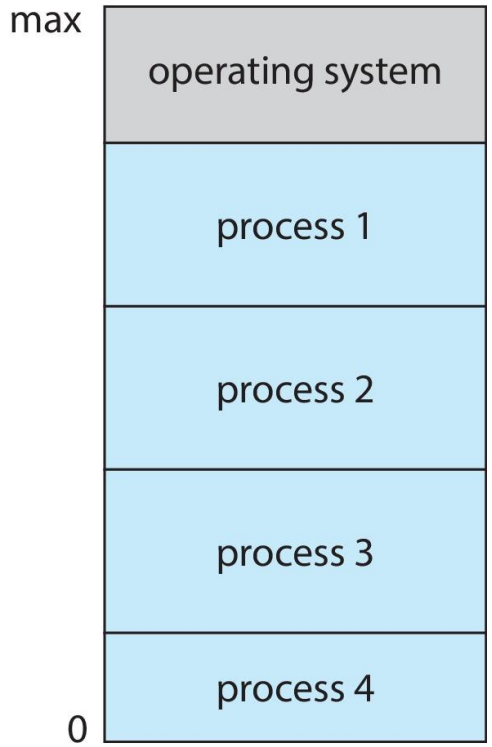
Operating System Operations

- Multi Programming (Batch System)
- Multi Tasking Operations (Time Sharing System)
- Dual Mode operations
- Timer

Multi Programming (Batch System)

- A multiprogramming operating system is one that can execute multiple programs simultaneously on a single processor.
- Single user cannot always keep CPU and I/O devices busy
- One job selected and run via **job scheduling**
- When job has to wait (for I/O for example), OS switches to another job

- Job Scheduling
  - Job scheduling is the process of allocating system resources to many different tasks by an operating system (OS).



Multi Tasking (Time sharing System)

1. Multi-Tasking Operating System is capable of executing multiple application simultaneously without slowing down the system
2. share similar processing resources like a **CPU**.
3. Used CPU scheduling
4. If processes don't fit in memory, ***swapping moves*** them in and out to run
5. Virtual memory allows execution of processes not completely in memory

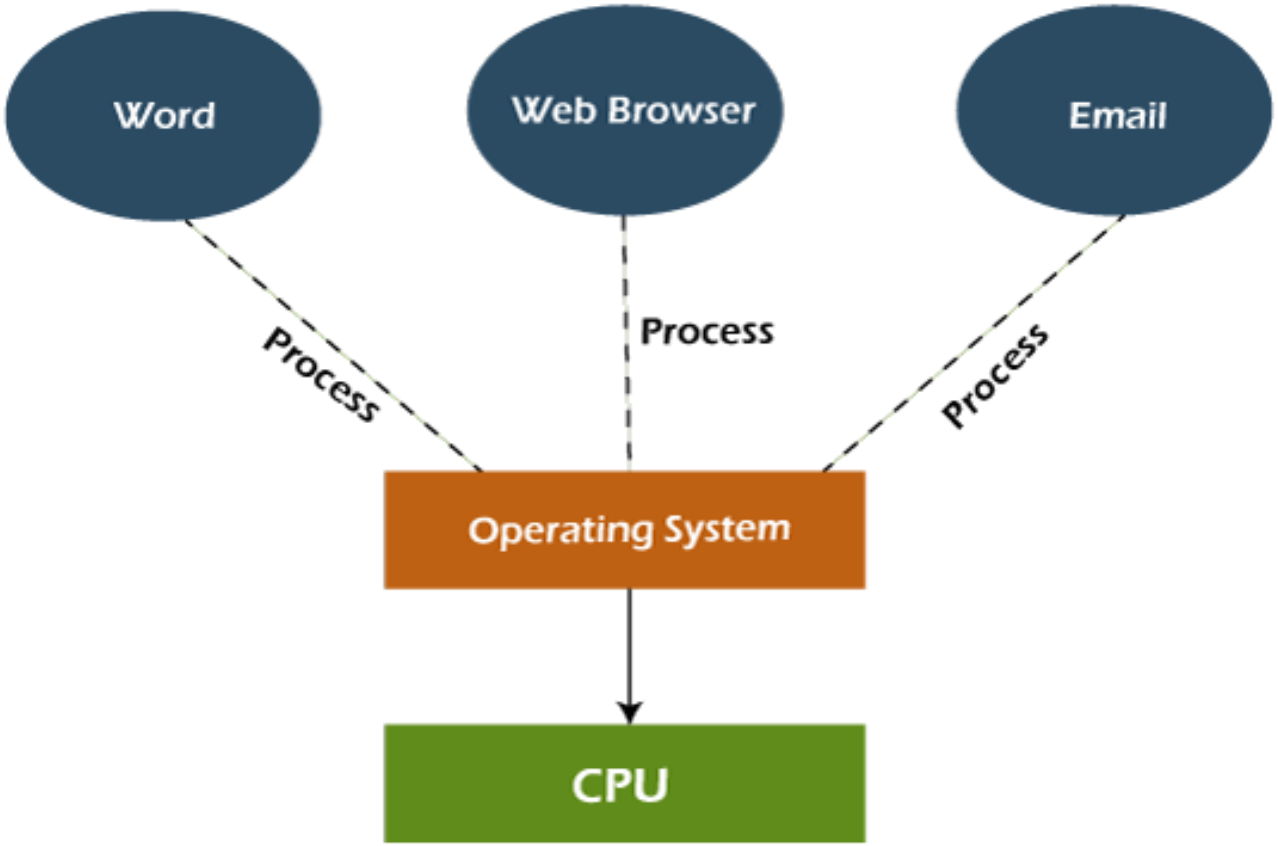
CPU scheduling

CPU Scheduling is a process of determining which process will own CPU for execution while another process is on hold.

Virtual Memory

Virtual memory uses both hardware and software to enable a computer to compensate for physical memory shortages, temporarily transferring data from random access memory (RAM) to disk storage.

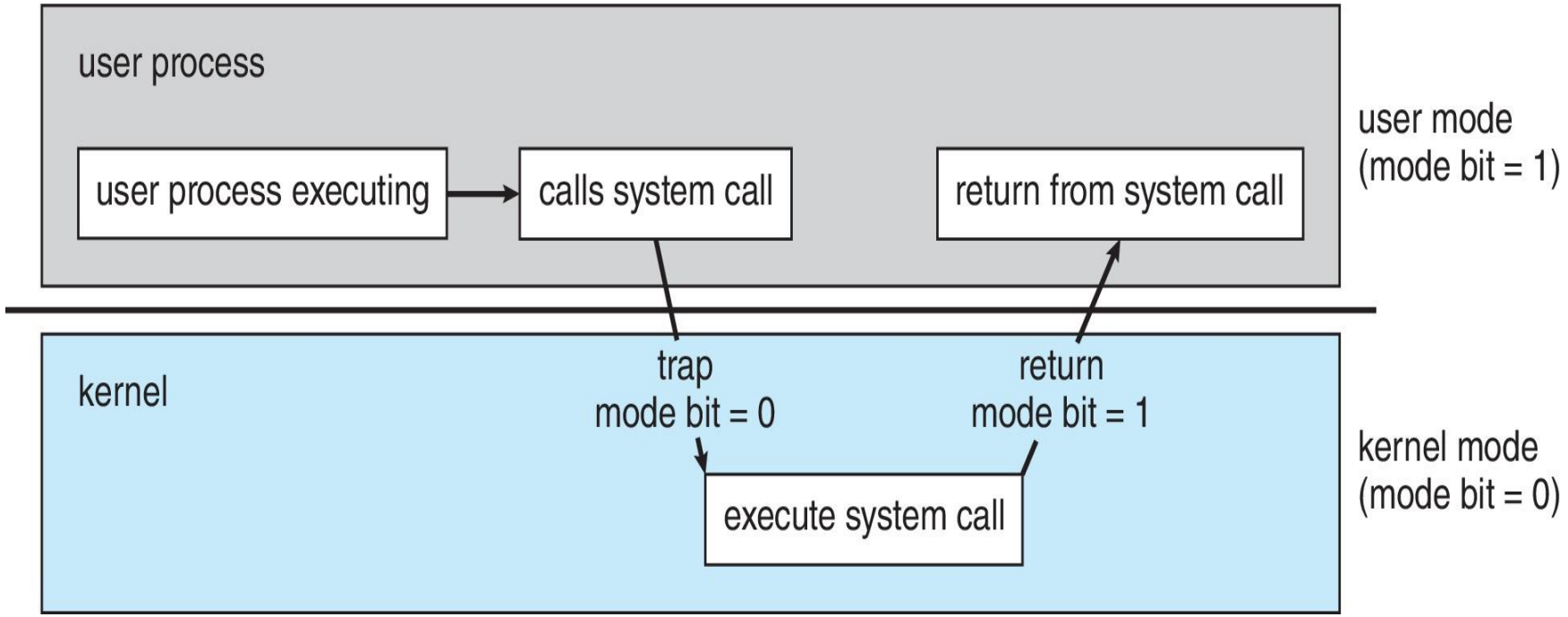
Multi Tasking (Time sharing System) Cont…)



Dual Mode System

- Dual-mode operation is designed to provide a layer of protection and stability to computer systems by separating user programs.
- operating system into two modes
  - **user mode**
    - application program do not have direct access to system resources
  - **kernel mode** (also called supervisor mode, system mode, privileged mode)
    - the processor mode that enables software to have full and unrestricted access to the system and its resources.
  - **Mode Bit**
    - Used to indicate the current mode of the system
      - 0 represent the kernel mode
      - 1 represent the user mode

Dual Mode System (Cont….)



Timer

- Timer to prevent infinite loop (or 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

Practice:

Identify the following services are privilege or not privilege

1. Generate any trap instruction
2. Clear the memory or remove process from memory
3. Set Timer
4. Sending final printout
5. I/O Instruction
6. Context Switching
7. Reading the date and time

Practice:

Identify the following services are privilege or not privilege

Instruction	Solution
Generate any trap instruction	Non-Privilege
Clear the memory or remove process from memory	Privilege
Set Timer	Privilege
I/O Instruction	Privilege
Sending final printout	Non-Privilege
Reading the date and time	Non-Privilege
Context Switching	Non-Privilege