Chapter: Introduction & Operating-System Structures

NARZU TARANNUM (NAT)
LECTURER
DEPT. OF CSE, BRAC UNIVERSITY

Information of Course Teachers

- Narzu Tarannum
  <narzu.tarannum@bracu.ac.bd

Recommended Text book

- Silberschatz, Peter B. Galvin, Greg Gagne, "Operating System Concepts", Wiley; 9th edition (2009) ISBN: 978-1-118-06333-0

Course Outline

- Operating Systems Overview and Structures
- Processes Management
- CPU Scheduling
- Threads
- Process Synchronization
- Deadlocks
- Memory Management/ Main memory
- Virtual Memory
- File-System Interface

Objectives

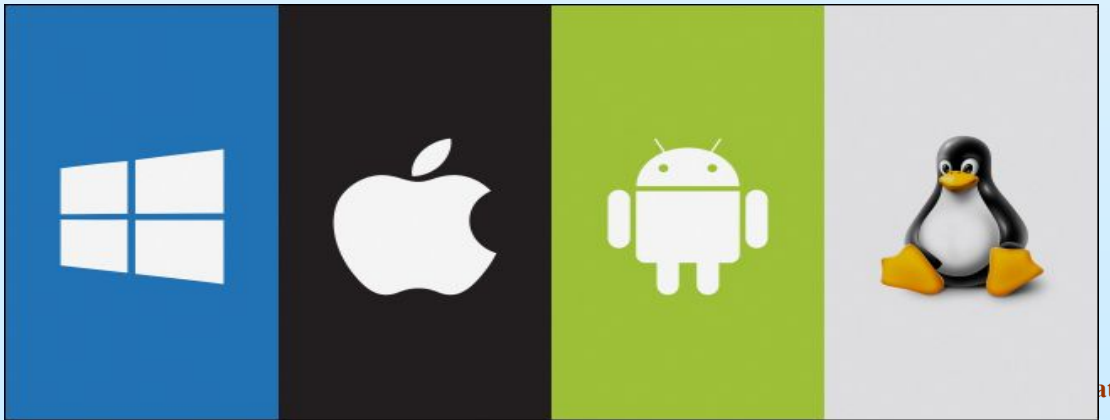
- To provide a grand tour of the major operating systems components
- To provide coverage of basic computer system organization
- To describe the services an operating system provides to users, processes, and other systems
- To discuss the different structure of operating system

Content of this chapter

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture
- Operating-System Structure
- Operating-System Operations
- Storage Structure
- System call
- Protection and Security
- Operating systems Services

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 and make the computer system convenient to use.
  - Use the computer hardware in an efficient manner.



What is an Operating Systems do?

It works between users and computer hardware.

- Resource allocator – manages and allocates resources.
- Control program – controls the execution of user programs and operations of I/O devices .
- Kernel – the one program running at all times (all else being application programs).
  - The **kernel** is the central module of an **operating system** (**OS**). It is the part of the **operating system** that loads first, and it remains in main memory. The **kernel** code is usually loaded into a protected area of memory to prevent it from being overwritten by programs or other parts of the **operating system**.

Primary Functions of an OS

- Processes-management
- Storage-memory management
- Data-file management
- Input/output devices-i/o management
- Network management
- Protection& security

Software

The two most common types of software are :

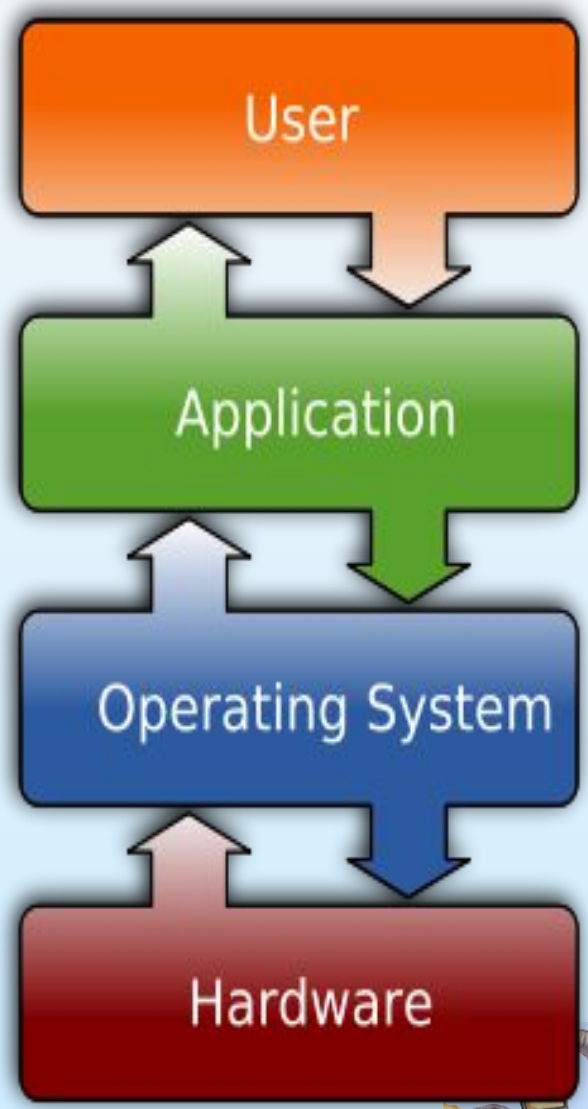
- System software
- Application software.

What is System Software?

System Software refers to the operating system and all utility programs that manage computer resources at a low level. Systems software includes compilers, loaders, linkers, and debuggers.

What is Application Software?

Applications software comprises programs designed for an end user, such as word processors, database systems, and spreadsheet programs.



Four Components of a Computer System

- Computer system can be divided into four components
  - **Hardware** – provides basic computing resources

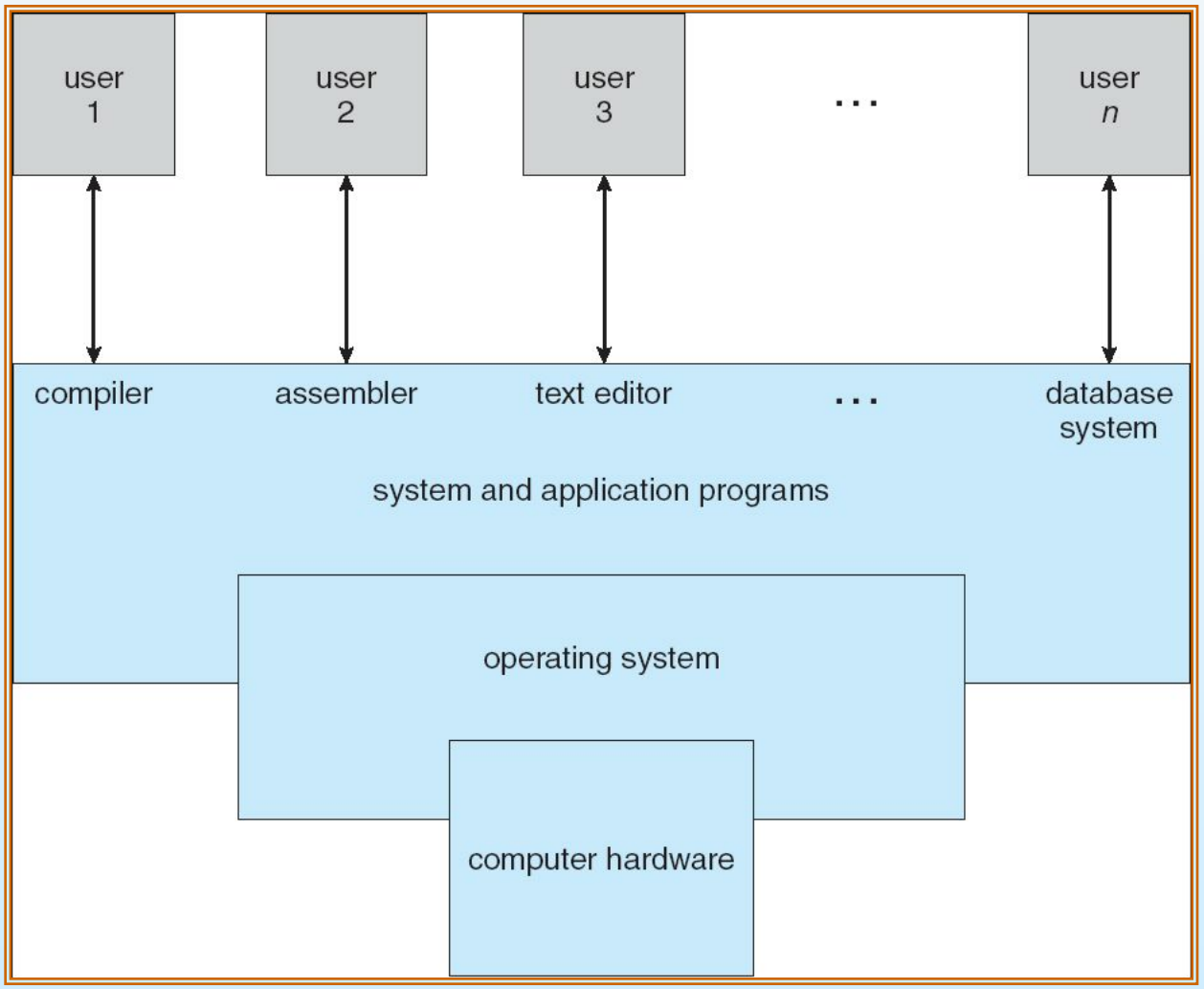
    CPU, memory, I/O devices
  - **Operating system**

    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

Four Components of a Computer System



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

TO UNDERSTAND WHAT OS ARE, WE MUST FIRST UNDERSTAND HOW THEY HAVE DEVELOPED.

History of OS / Types of OS

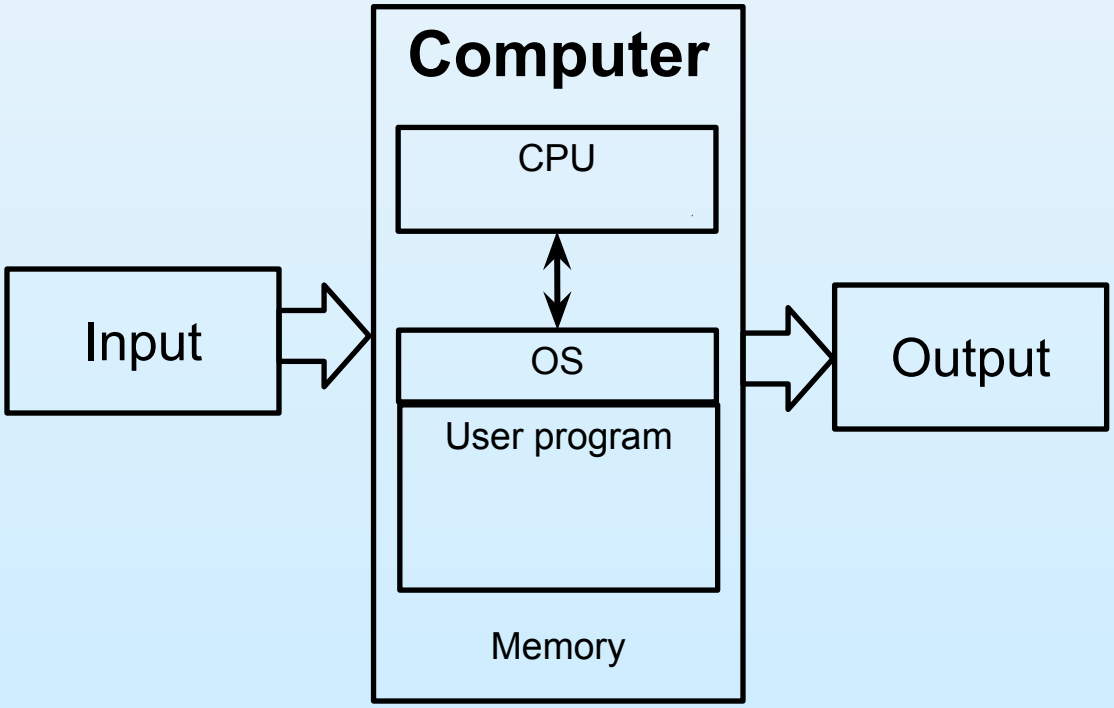
- Simple Batch Systems
- Multiprogramming Systems
- Time-Sharing Systems
- Parallel Systems
- Distributed Systems
- Real -Time Systems

History of OS/ Types of OS

- **Simple Batch Systems:** Automatically transfers control from one job to another.
- **Multi-programmed Batched Systems:** Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.
- **Time Sharing Systems:** Logical extension of multiprogramming.
- **Parallel Systems:** Multiprocessor systems with more than one CPU
- **Distributed Systems:** Distribute the computation among several physical processors.
  - **Loosely coupled system** – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- **Real-time Systems:** Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.

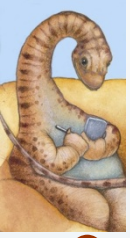
Simple Batch Systems

- Automatically transfers control from one job to another.
- Common input-output device are card readers and tape drivers.
- User prepare a job which consisted of program, input data and control instructions.



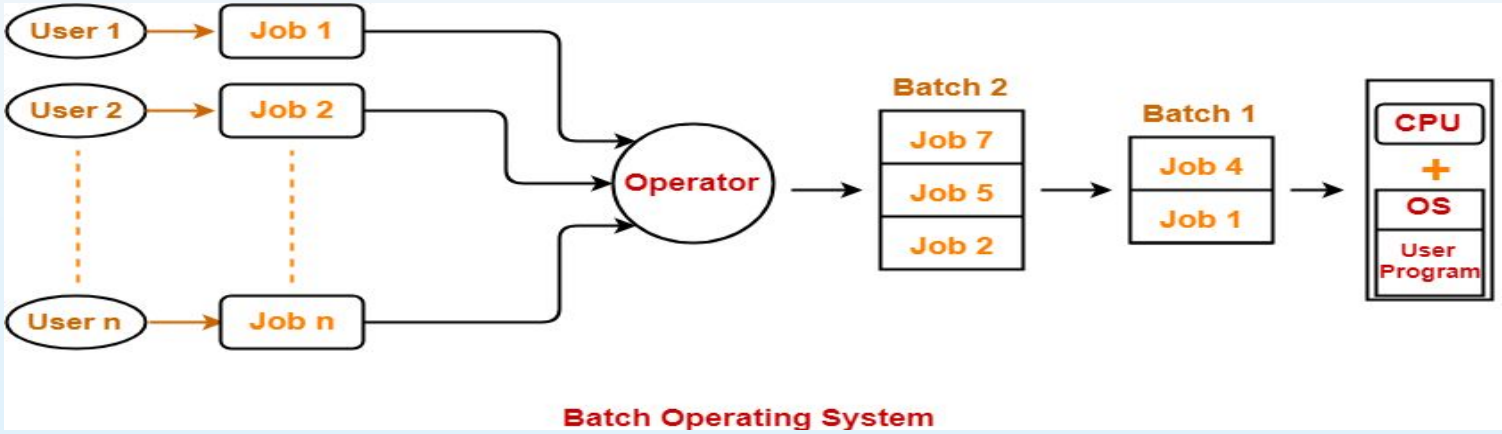
Limitations of Simple Batch Systems

- Not interactive
- Memory is limited
- CPU utilization is poor
- Speed mismatch between I/O device and CPU



Simple Batch Systems

- User prepares his job on an off-line device like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group.



Batch Operating System

Advantages-

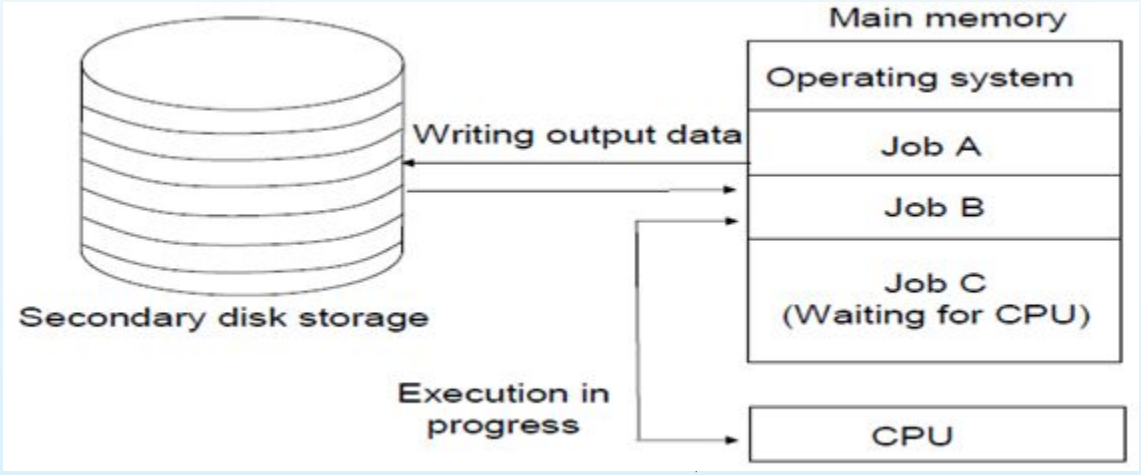
- It saves the time that was being wasted earlier for each individual process in context switching from one environment to another environment.
- No manual intervention is needed.

Disadvantages

- Executing a series of non-interactive jobs all at one time.
- The output is obtained only after all the jobs are executed.
- Thus, priority can not be implemented if a certain job has to be executed on an urgent basis.

Multi-programmed Operating Systems

- Multiprogramming needed for **efficiency** or maximize **CPU utilization**.
- Multiprogramming means more than one process in main memory which are ready to execute.



- 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
- Process generally require **CPU time** and **I/O time**, when CPU has to wait (for I/O for example), OS switches to another job and this idea will continue.

Multi-programmed Operating System

- Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.

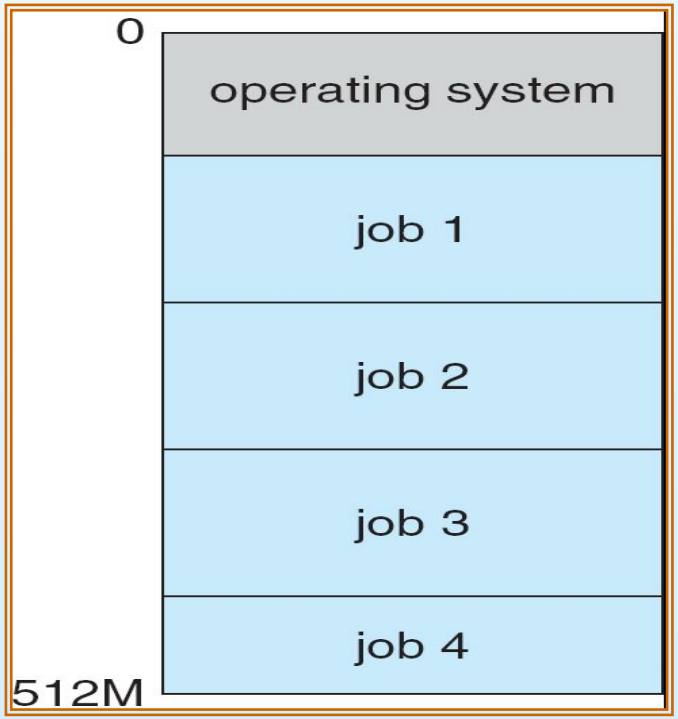
Advantages

- High CPU utilization.
- It appears that many programs are allotted CPU almost concurrently.
- Response time is shorter

Disadvantages

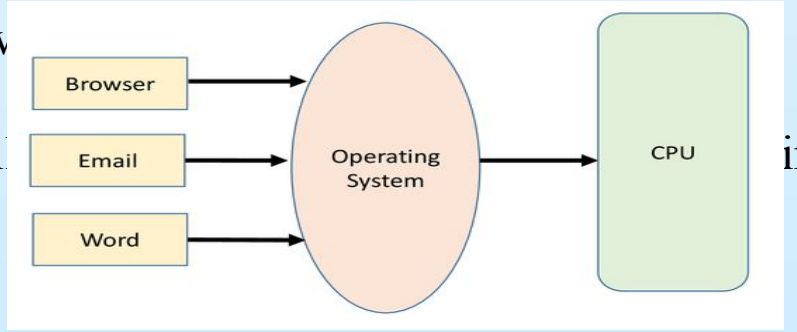
- CPU scheduling is required.
- To accommodate several jobs in memory, memory management is essential.

- **Multi-programmed systems provide an environment in which the various system resources(like CPU, memory and peripheral devices) are utilized effectively but they do not provide for user interaction with the computer system.**



Multitasking/Time-Sharing Systems–Interactive Computing

- Multitasking is multiprogramming with time sharing where CPU executes multiple jobs by switching among them.
- Timesharing (multitasking) is logical extension of multiprogramming, in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing.
- The **operating system** is able to keep track of where you are in these tasks and go from one to the other without losing information.
- Response time should be < 1 second
- Time sharing requires an interactive(or hands-on)computer system, which provides direct communication betw
- A time-shared OS al imultaneously.



Multitasking/Time-Sharing Systems–Interactive Computing

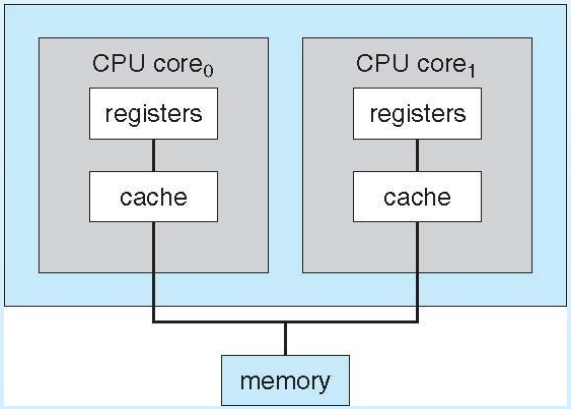
- Each user has at least one separate program for executing in memory.
- A program loaded into memory and executing is called a ⇨process
- ⇨ Job scheduling
- ⇨ CPU scheduling
- If processes don't fit in memory, **swapping** moves them in and out to run from main memory achieving this goal is virtual memory.

Computer-system Architecture

- **Single-processor Systems:** If there is only one general purpose CPU, then the system is a single-processor system.
  - Other special purpose processors are also present which perform device specific task.

Computer-system Architecture

- **Multiprocessor Systems/Parallel System:** Multiprocessor systems with more than one CPU in close communication.
  - **Tightly Coupled System:** processors share memory and a clock; communication usually takes place through the shared memory.
    - HAVE THREE MAIN ADVANTAGES
      - INCREASED THROUGHPUT
      - ECONOMY OF SCALE
      - INCREASED RELIABILITY

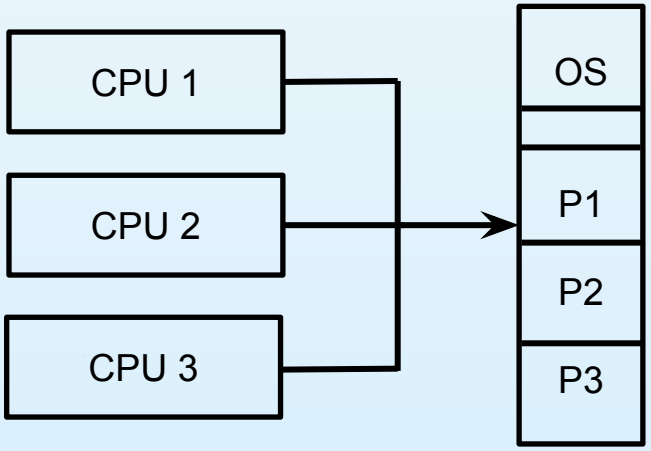


Multiprocessor Systems/Parallel System

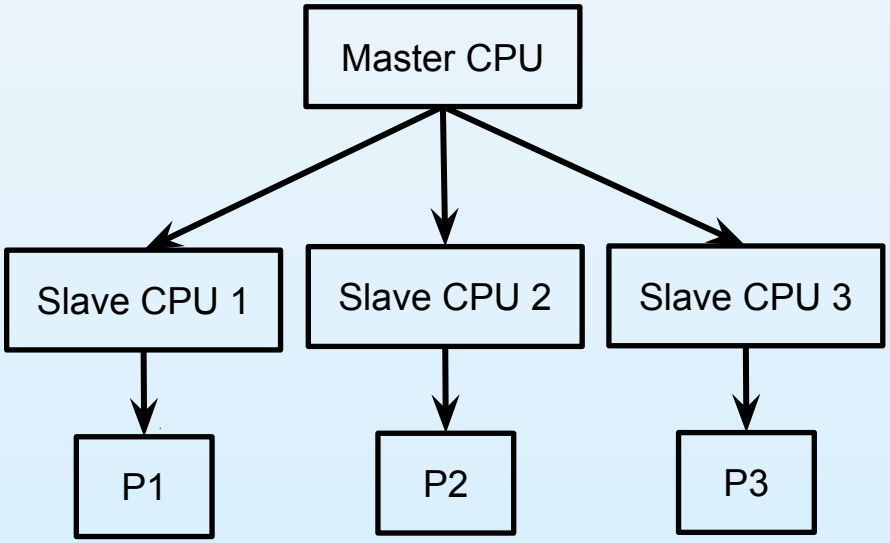
- Two Types:
  - Symmetric Multiprocessing(SMP):
    - Each processor runs an identical copy of the operating system.
    - Most modern operating systems support SMP.
  - Asymmetric Multiprocessing:
    - Each processor is assigned a specific task; master processor schedules and allocates work to slave processors

Two Types

- **Symmetric Multiprocessing Architecture Example**



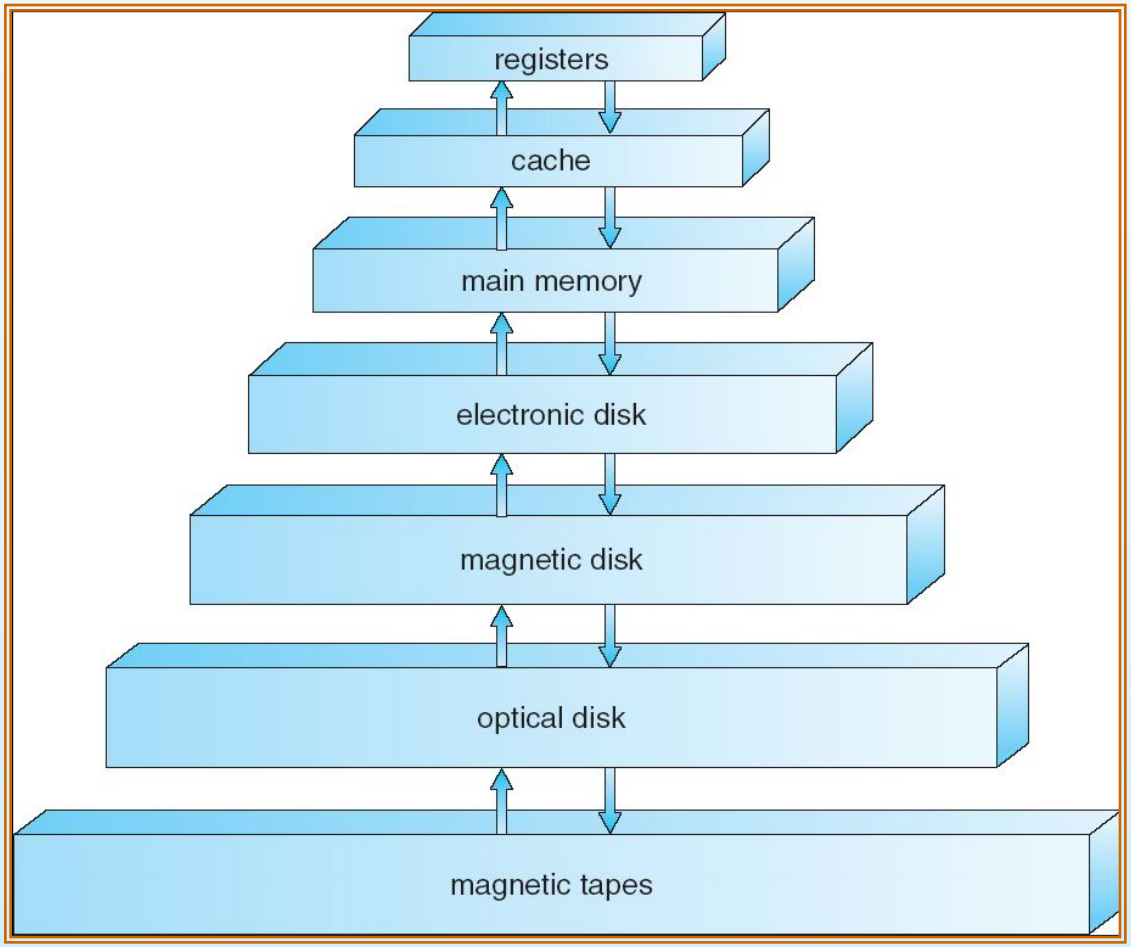
- **Asymmetric Multiprocessing Architecture Example**



Distributed Systems and Real-time Systems

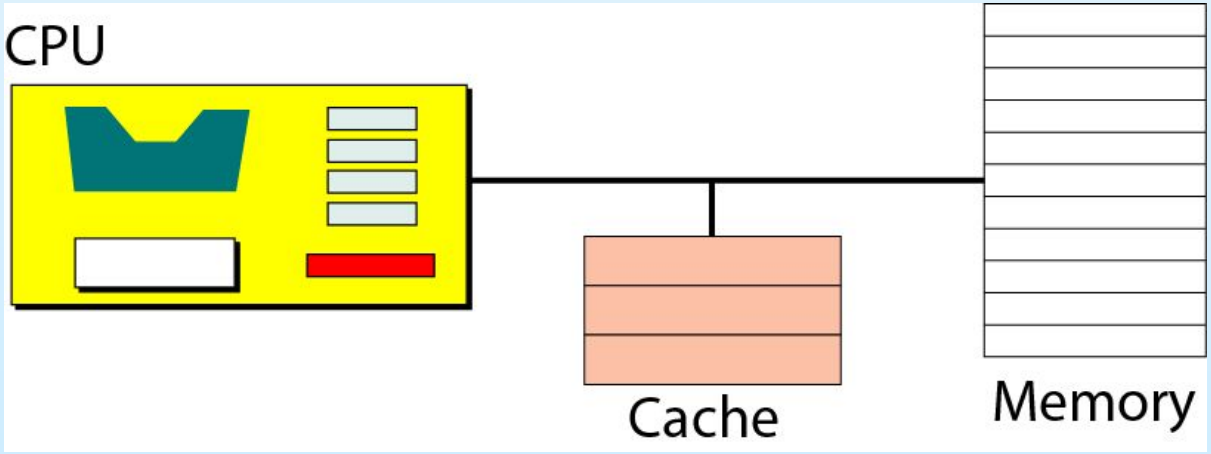
- **Distributed Systems:** Distribute the computation among several physical processors.
  - **Loosely coupled system** – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- **Real-time Systems:** Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.

Storage-Device Hierarchy

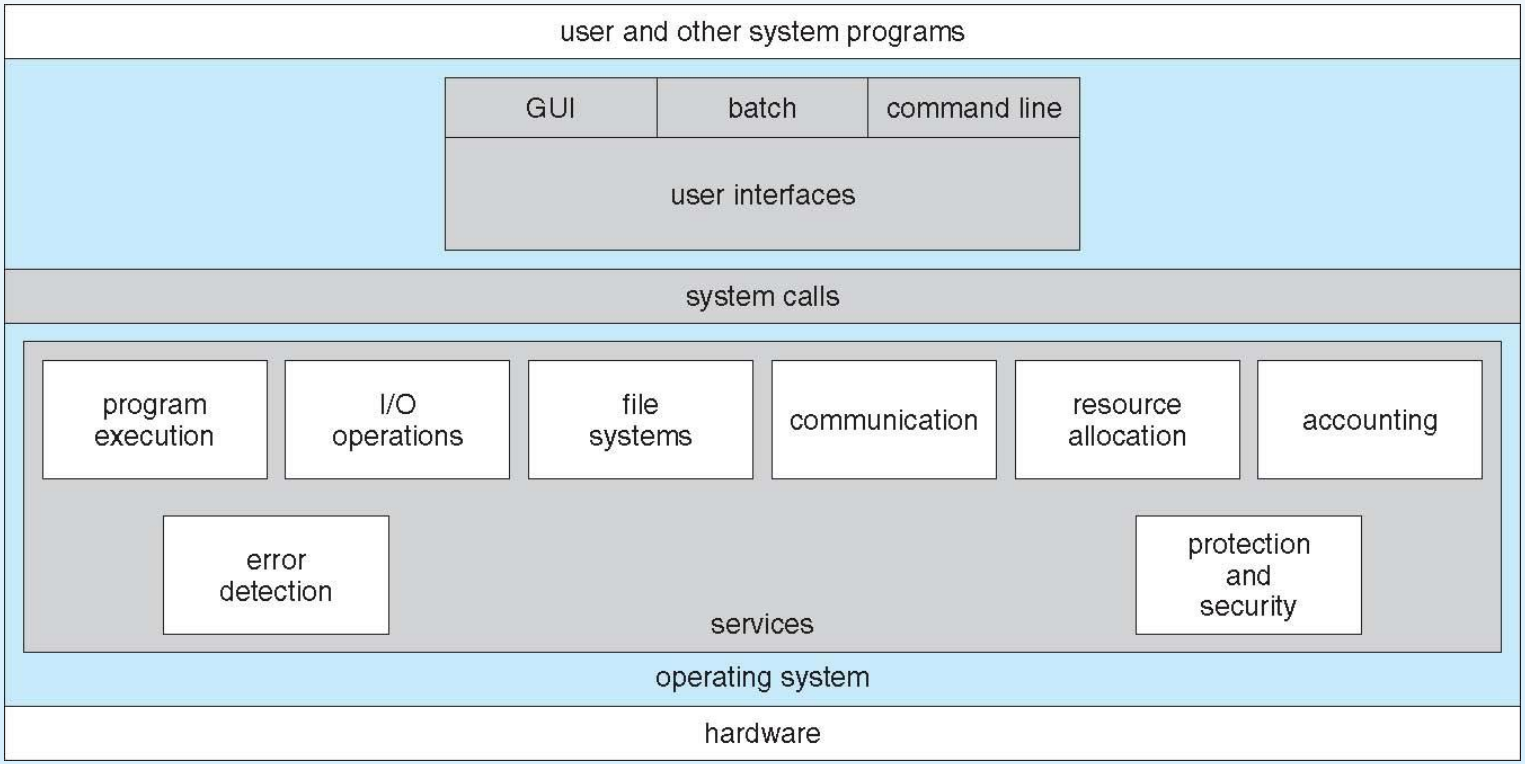


Storage Hierarchy

- Storage systems organized in **hierarchy**.
  - Size
  - Speed (Access time)
  - Cost
  - Volatility
- *Caching* – Information in use copied from slower to faster storage temporarily



A View of Operating System Services



Operating System Services

- Operating systems provide an environment for execution of programs and services to programs and users
- One set of operating-system services provides functions that are helpful to the user:
  - **User interface** - Almost all operating systems have a user interface (**UI**).
    - Varies between **Command-Line** (**CLI**), **Graphics User Interface** (**GUI**), **Batch interface-** in which commands and directives to control those commands are entered into files, and those files are executed.
  - **Program execution** - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
  - **I/O operations** - A running program may require I/O, which may involve a file or an I/O device

Operating System Services (Cont.)

- One set of operating-system services provides functions that are helpful to the user (Cont.):
  - **File-system manipulation** - The file system is of particular interest. Programs need to read and write files and directories, create and delete them, search them, list file Information, permission management.
  - **Communications** – Processes may exchange information, on the same computer or between computers over a network
    - Communications may be via shared memory or through message passing (packets moved by the OS)
  - **Error detection** – OS needs to be constantly aware of possible errors
    - May occur in the CPU and memory hardware, in I/O devices, in user program
    - For each type of error, OS should take the appropriate action to ensure correct and consistent computing
    - Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

Operating System Services (Cont.)

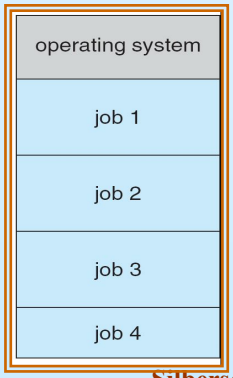
- Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing
  - **Resource allocation -** When multiple users or multiple jobs running concurrently, resources must be allocated to each of them
    - Many types of resources - CPU cycles, main memory, file storage, I/O devices.
  - **Accounting -** To keep track of which users use how much and what kinds of computer resources
  - **Protection and security -** The owners of information stored in a multiuser or networked computer system may want to control use of that information, concurrent processes should not interfere with each other
    - **Protection** involves ensuring that all access to system resources is controlled
    - **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts

Operating-System Operations

- An Operating system is Interrupt driven
  - Hardware interrupt
    - Occurs by one of the devices
  - Software interrupt (exception or trap)
    - Request for operating system service
    - Software error (e.g., division by zero)
    - A *trap* is a software-generated interrupt caused either by an error or a user request.

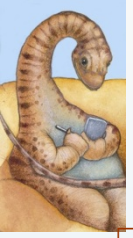
Operating-System operations-protection

- We know modern operating systems are interrupt driven.
  - **Hardware** generates interrupt.
    - Many errors detected by hardware can be handled by OS.
  - **Software** error handled by **exception** or **trap.**
    - A trap (or an exception) is a software-generated interrupt caused either by an error (for example, division by zero or invalid memory access) or by a specific request from a user program that an operating-system service need to be performed.
- With sharing many processes could be adversely affected by a bug in one program. Since the **operating system** and the **users program** share the **hardware** and **software** resources of the computer system, a properly designed OS must ensure that an incorrect program can not run and also can not cause other programs to execute incorrectly.



Operating-System operations-protection

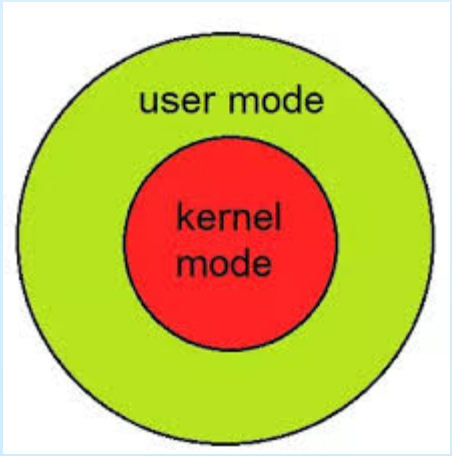
- In order to ensure the proper execution of operating system, we must be able to distinguish between the execution of the operating system code and user defined code.



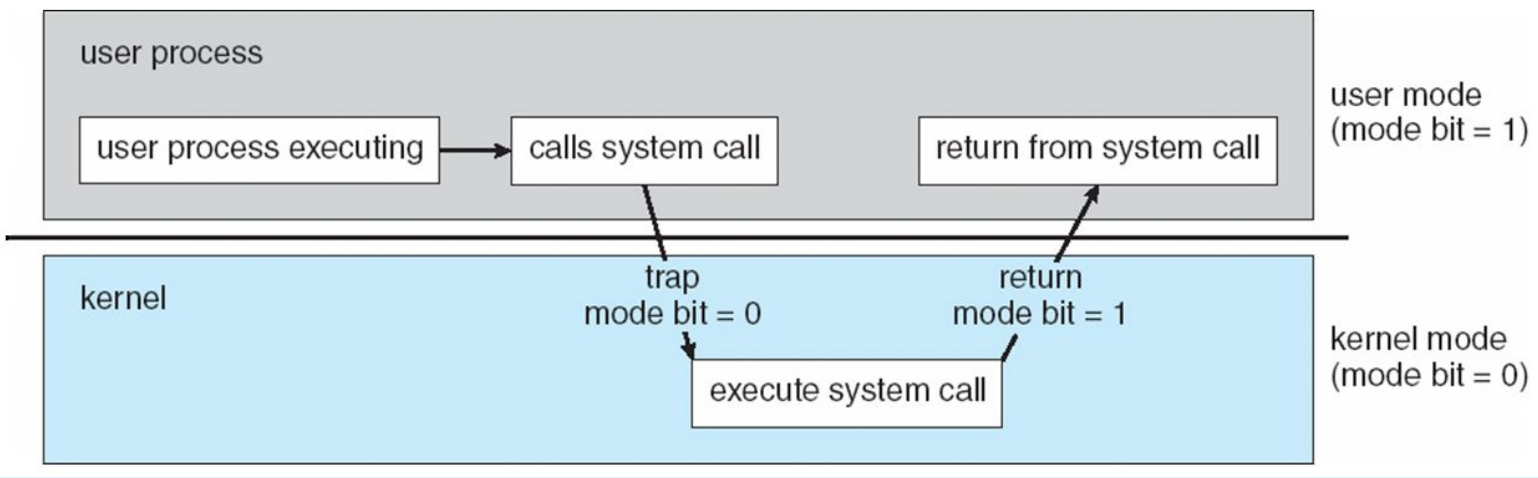
Hardware Protection

- The approach taken by most computer system is to provide hardware support that allows us to differentiate among various modes of execution.

- **Dual-mode operation** allows OS to protect itself and other system components.
  - **User mode (1)** and **Kernel/Monitor/System mode (0)**
  - **Mode bit** provided by hardware.
    - Provides ability to distinguish when system is running user code or system code
    - Some instructions designated as **privileged**, only executable in system mode
    - System call changes mode to kernel, return from call resets it to user



Transition from User to Kernel Mode



- With the mode bit, we can distinguish between a task that is executed on behalf of the operating system that is ‘zero’ and ‘one’ is executed on behalf of the user. When the computer system is executing on behalf of a user application , the system is in user mode .
- when a user application requests a service from the operating system (via a system call), the system must transition from user to kernel mode to fulfill the request. This is shown in his figure.

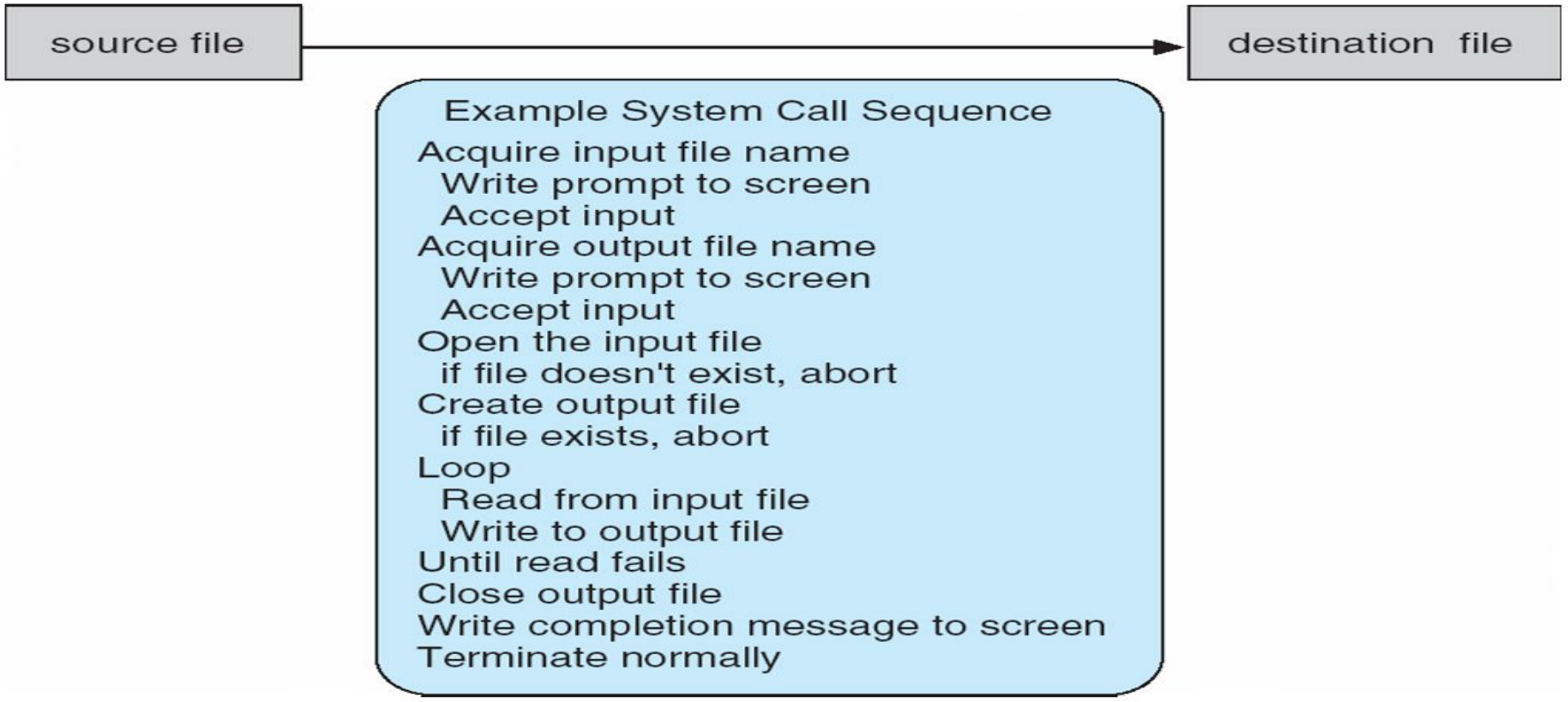
System Calls

- A **system call** is a way for programs to interact with the **operating system**.
- A computer program makes a **system call** when it makes a request to the **operating system's** kernel. **System call** provides the services of the **operating system** to the user programs via Application Program Interface(API).
- Programming interface to the services provided by the OS
- Routines typically written in a high-level language (C or C++)

❖ Note that the system-call names used throughout this text are generic

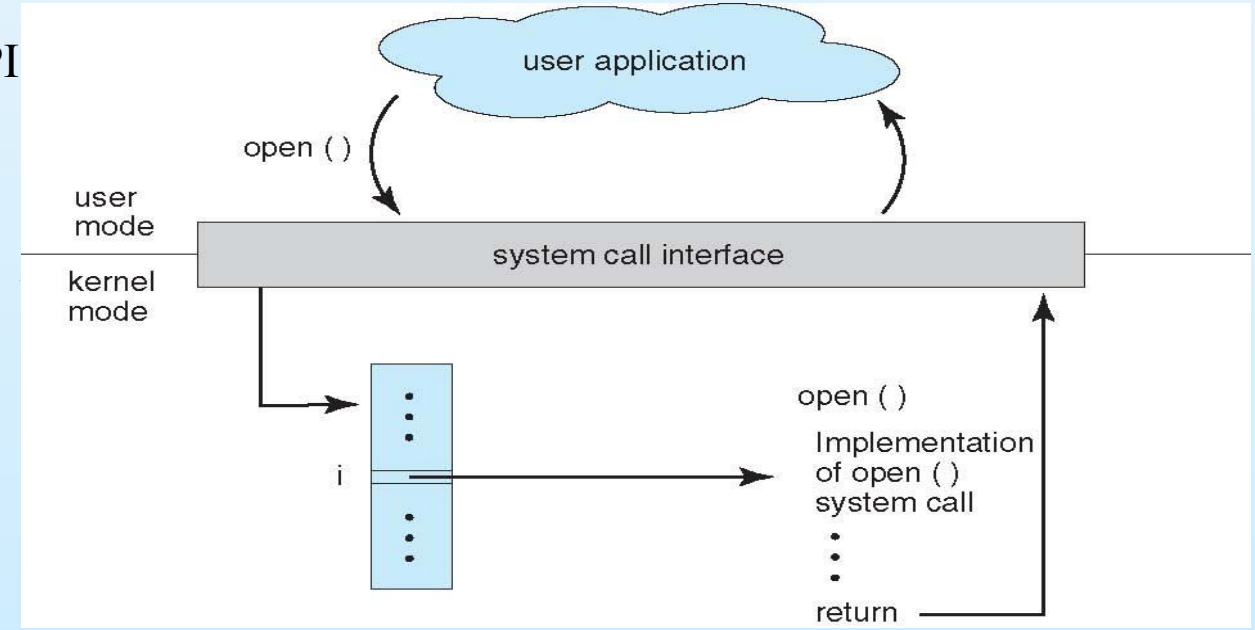
System Calls

- System call sequence to copy the contents of one file to another file



System Call Implementation

- Typically, a number associated with each system call
  - **System-call interface** maintains a table indexed according to these numbers
- The system call interface invokes intended system call in OS kernel and returns status of the system call and any return values
- The caller need know nothing about how the system call is implemented
- API



End