Unit I
Introduction

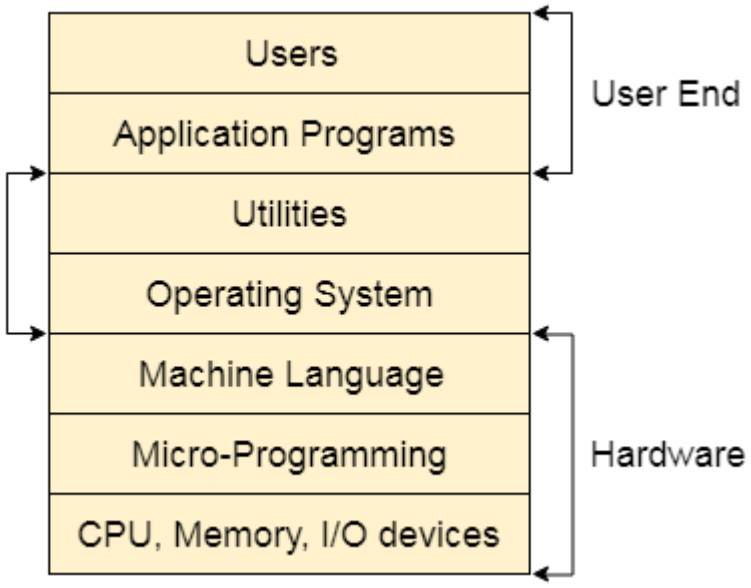
- Operating System Operations
- Functions,
- Multiprogramming
- Multiprocessing System

BY

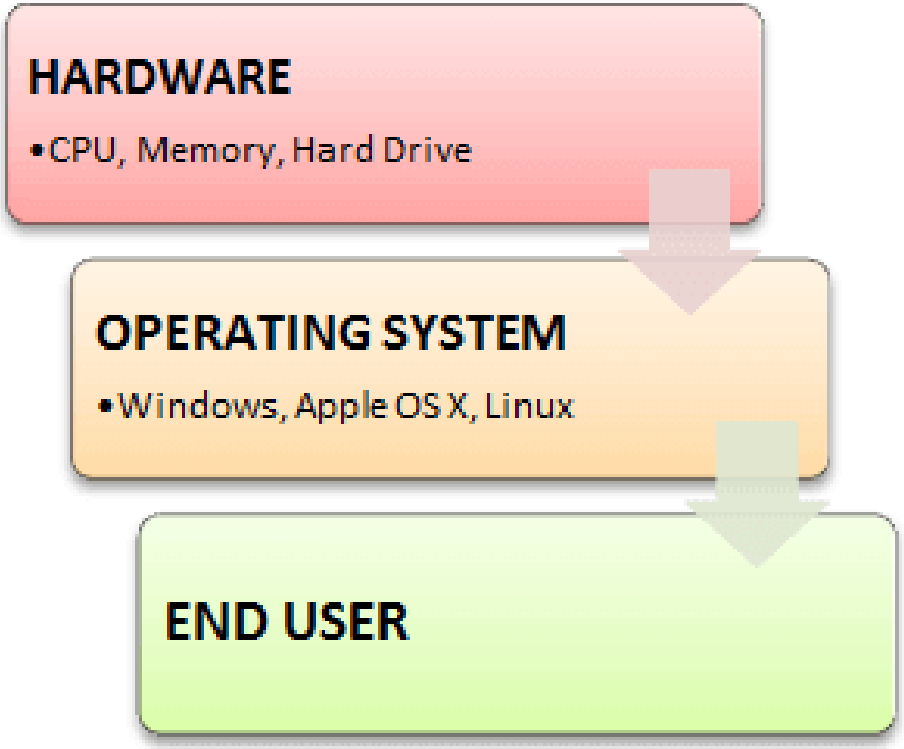
KARTHICK P M Tech., NET ., (Ph D)

A Computer System consists of:

- Users (people who are using the computer)
- Application Programs (Compilers, Databases, Games, Video player, Browsers, etc.)
- System Programs (Shells, Editors, Compilers, etc.)
- Operating System (A special program which acts as an interface between user and hardware)
- Hardware (CPU, Disks, Memory, etc)

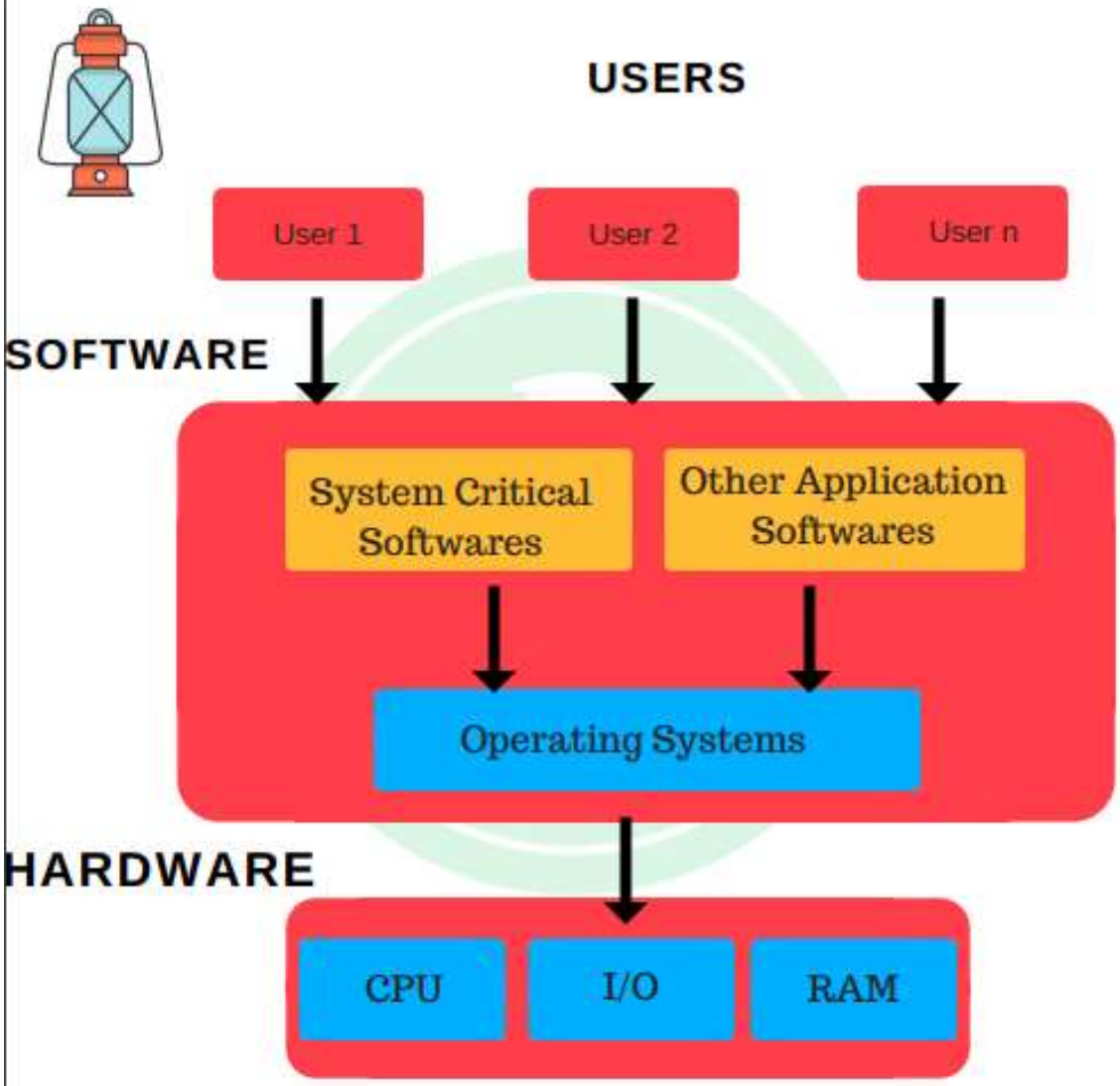


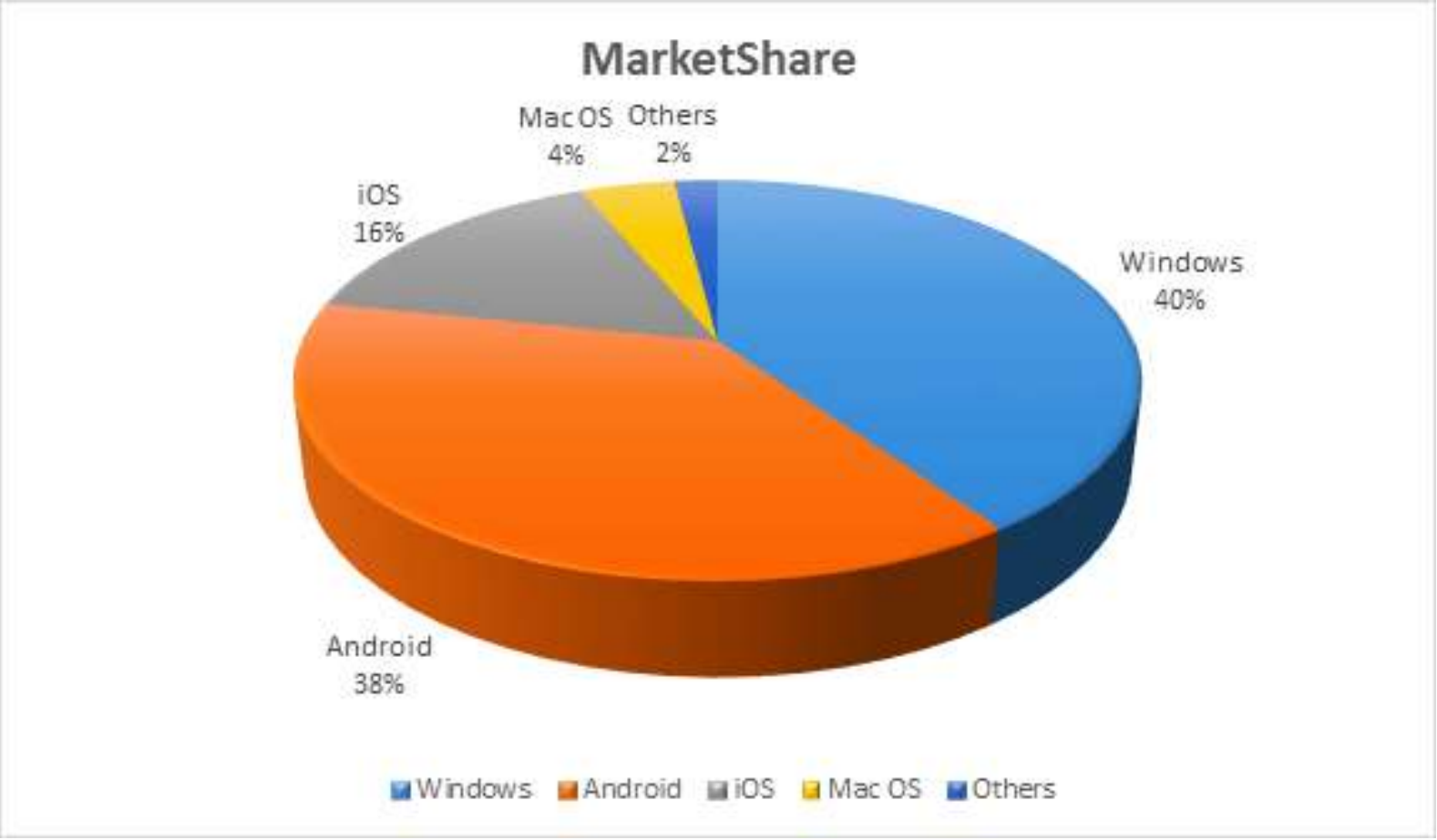
- An **Operating System (OS)** is a software that acts as an **interface between computer hardware components and the user**.
- Every computer system must have at least **one operating system** to run other programs.
- Applications like **Browsers, MS Office, Notepad Games,** etc., need some environment to run and perform its tasks.
- The OS helps you to **communicate with the computer** without knowing how to speak the computer's language.
- It is not possible for the user to use any computer or mobile device without having an operating system



Operating system goals:

- Execute user programs and make **problem solving easier**.
- Make the computer system **convenient to use**
- Efficiently use **available resources**


USERS
User 1
User 2
User n
SOFTWARE
System Critical Softwares
Other Application Softwares
Operating Systems
HARDWARE
CPU
I/O
RAM


MarketShare
Mac OS
4%
Others
2%
iOS
16%
Windows
40%
Android
38%
Windows
Android
iOS
Mac OS
Others

History Of OS

•Operating systems were first developed in the late 1950s to manage tape storage

•The General Motors Research Lab implemented the first OS in the early 1950s for their IBM 701

•In the mid-1960s, operating systems started to use disks

•In the late 1960s, the first version of the Unix OS was developed

•The first OS built by Microsoft was DOS. It was built in 1981 by purchasing the 86-DOS software from a Seattle company

•The present-day popular OS Windows first came to existence in 1985 when a GUI was created and paired with MS-DOS.

What is an operating system?

a) interface between the hardware and application programs

b) collection of programs that manages hardware resources

c) system service provider to the application programs

d) all of the mentioned

Which of the following is not an operating system?

1.Windows

2.Linux

3.Oracle

4.DOS

Oracle is an RDBMS (Relational Database Management System). It is known as Oracle Database, Oracle DB, or Oracle Only. The first database for enterprise grid computing is the Oracle database

BIOS

The **Basic Input Output System**, or BIOS, is a very small piece of code contained on a **chip on your system board**.

When you start your computer, BIOS is the first software that runs. It identifies your computer's hardware, configures it, tests it, and connects it to the operating system for further instruction.

Computer BIOS

ComputerHope.com

What are the four main functions of a PC BIOS?

•**POST** - Test the computer hardware and make sure no errors exist before loading the operating system. Additional information on the POST is available on our POST and beep codes page.

•**Bootstrap Loader** - Locate the operating system. If a capable operating system is located, the BIOS will pass control to it.

•**BIOS drivers** - Low-level drivers that give the computer basic operational control over your computer's hardware.

•**BIOS setup** or **CMOS setup** - Configuration program that allows you to configure hardware settings including system settings, such as date, time, and computer passwords.

BIOS is used?

1.By operating system

2.By compiler

3.By interpreter

4.By application software

Types of Operating System (OS)

- Batch Operating System
- Multitasking/Time Sharing OS
- Multiprocessing OS
- Real Time OS
- Distributed OS
- Network OS
- Mobile OS

Batch Operating System

Some computer processes are very lengthy and time-consuming. To speed the process, a job with a similar type of needs are batched together and run as a group.

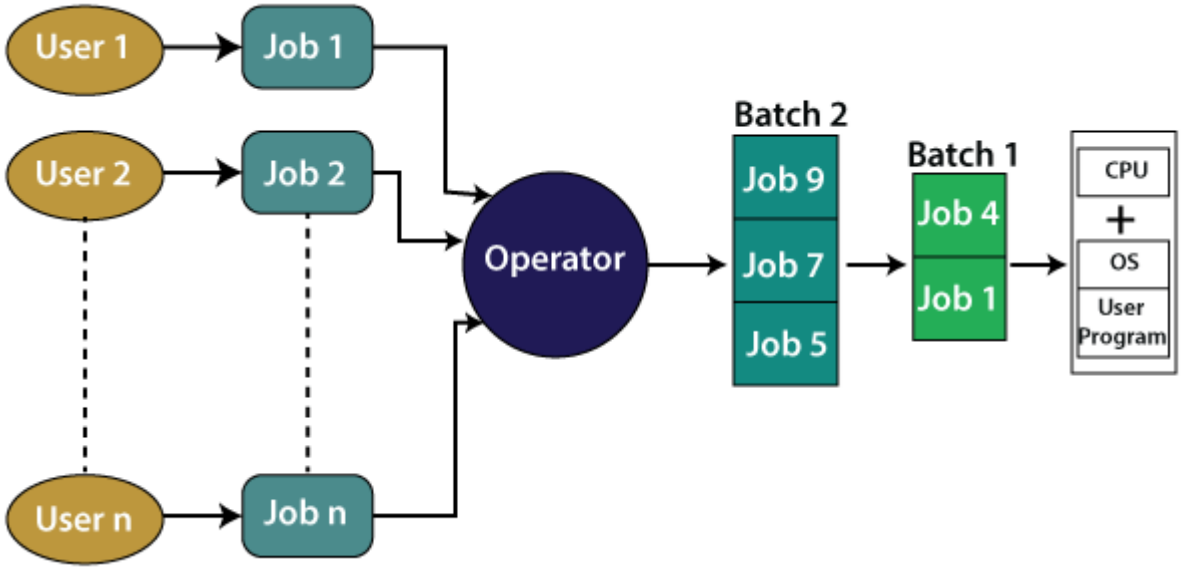
The user of a batch operating system **never directly interacts** with the computer. Every user prepares his job on an offline device like a punch card and submit it to the computer operator.

The problems with Batch Systems are as follows −

•Lack of interaction between the user and the job.

•CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.

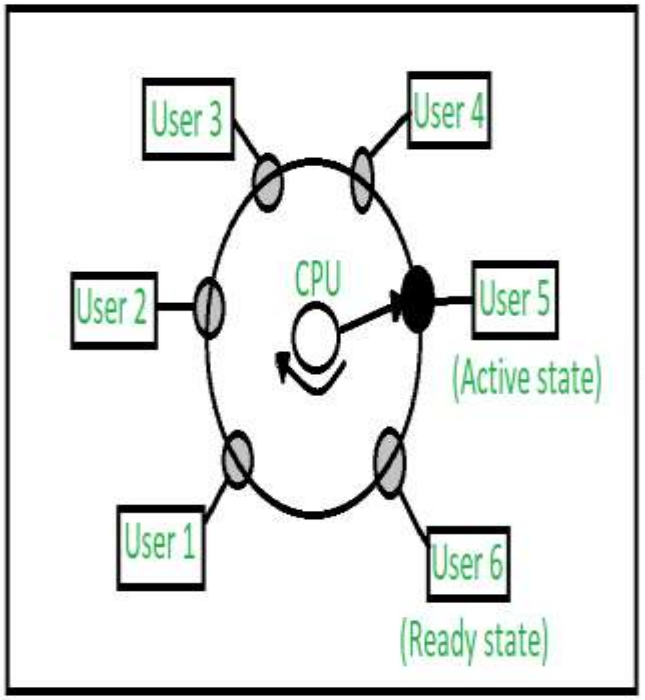
•Difficult to provide the desired priority.



Multi-Tasking/Time-sharing Operating systems

Time-sharing operating system enables people located at a different terminal(shell) to use a single computer system at the same time. It is logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing.

Multiprogrammed batch systems, the objective is to **maximize processor** use, whereas in Time-Sharing Systems, the objective is to **minimize response time.**



Real time OS

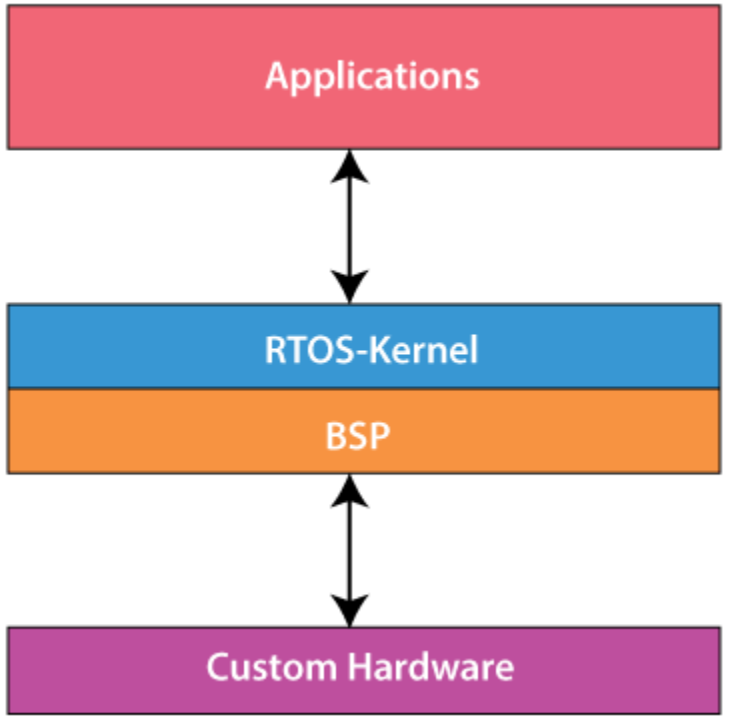
- A real time operating system time interval to process and respond to inputs is very small.
- The time taken by the system to respond to an input and display of required updated information is termed as the **response time.**
- A real-time operating system must have well-defined, fixed time constraints, otherwise the system will fail**. There are two types of real-time operating systems.**

Hard real-time systems

Soft real-time systems

- data should be processed in a fixed, small duration of time

e.g., in a petroleum refinery



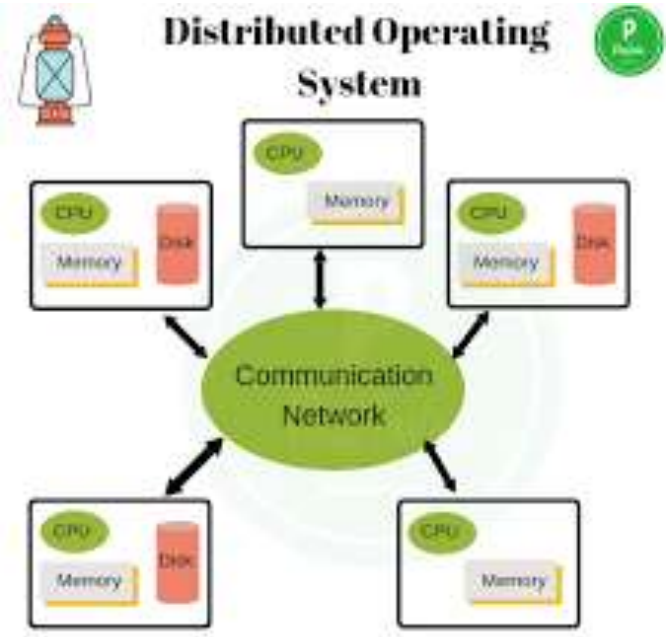
This system is used to control scientific instruments, missile launch systems, traffic lights control systems, air traffic control systems, etc.

Distributed Operating System

- Distributed systems use multiple central processors to serve multiple real-time applications and multiple users.
- Data processing jobs are distributed among the processors.

<u>Examples of Distributed Operating Systems</u>

- Ubuntu
- Windows Server 2012
- Linux (Apache Server)

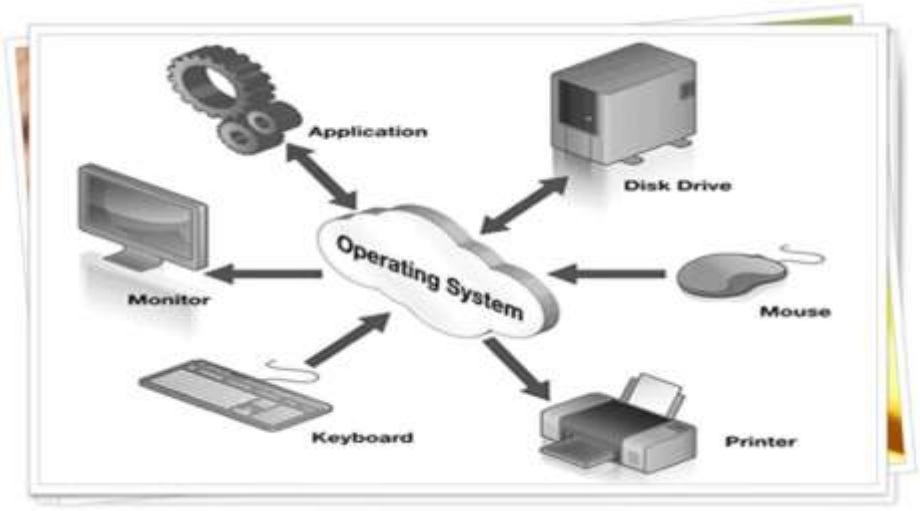


Network Operating System

Network Operating System runs on a server. It provides the capability to serve to manage data, user, groups, security, application, and other networking functions.

The purpose of this allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), a private network or to other networks.

- Microsoft Windows
- Server 2003,
- Microsoft Windows Server 2008,
- UNIX,
- Linux,
- Mac OS X,
- Novell NetWare,
- BSD.



Mobile OS

- Mobile operating systems are those OS which is especially that are designed to power smartphones, tablets, and wearables devices.
- Some most famous mobile operating systems are Android and iOS, but others include BlackBerry, Web, and watchOS.

Where is the operating system placed in the memory?

a) either low or high memory (depending on the location of interrupt vector)

b) in the low memory

c) in the high memory

d) none of the mentioned

Which one of the following is not a real time operating system?

a) RTLinux

b) Palm OS

c) QNX

d) VxWorks

In a timeshare operating system, when the time slot assigned to a process is completed, the process switches from the current state to?

a) Suspended state

b) Terminated state

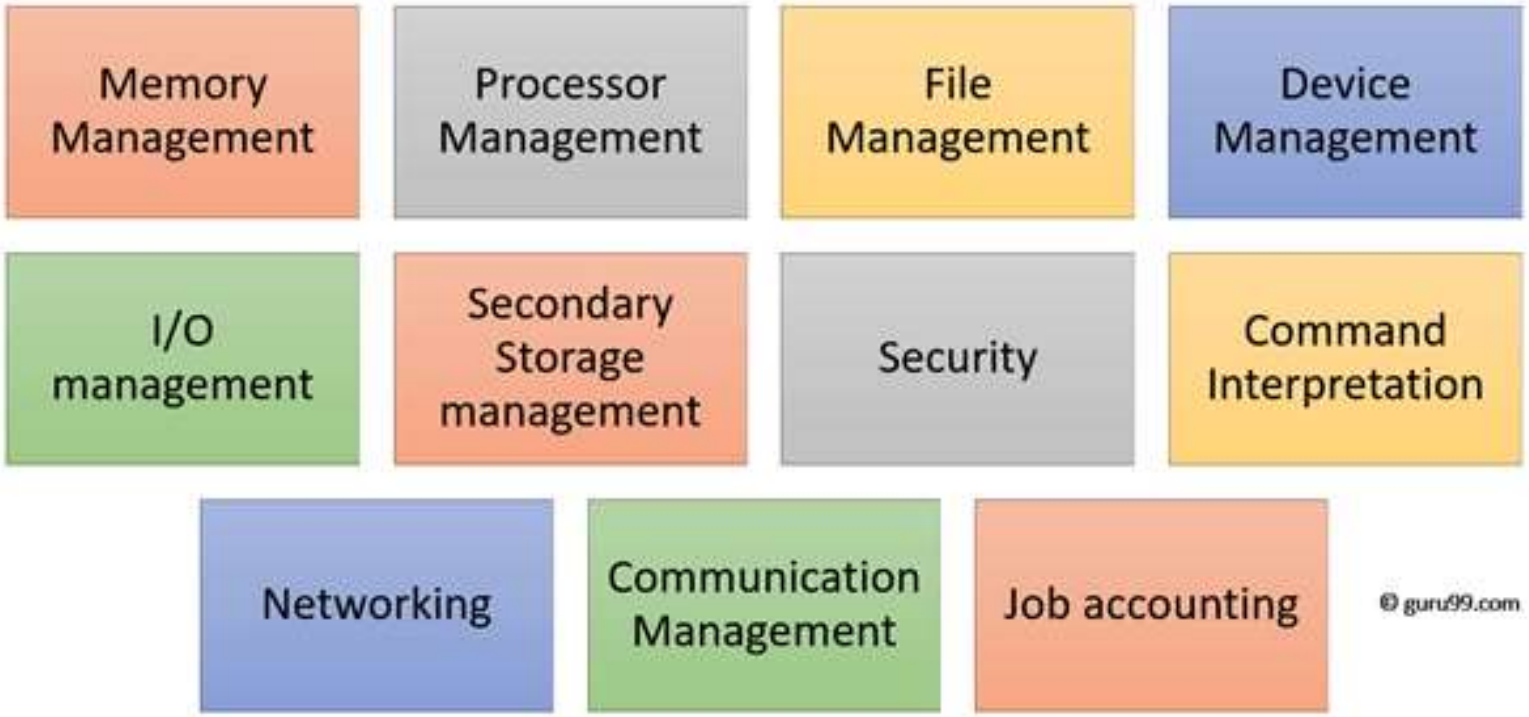
c) Ready state

d) Blocked state

Functions of Operating System

operating system functions may include managing memory, files, processes, I/O system & devices, security, etc.

Below are the main functions of Operating System:



1.Process management: Process management helps OS to **create and delete** processes. It also provides mechanisms for synchronization and communication among processes.

2.OS decides **which process gets the processor** when and for how much time. This function is called **process scheduling.**

3.Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.

Memory management:

- Memory Management refer Management of Primary Memory or Main Memory.
- Memory management module performs the task of allocation and de-allocation of memory space to programs in need of this resources.
- Main memory is a large array of words or bytes where each word or byte has its own address.
- Main memory provides a fast storage that can be accessed directly by the CPU

File management:

- A file system is normally **organized into directories** for easy navigation and usage
- It manages all the file-related activities such as **organization storage, retrieval, naming, sharing, and protection of files.**

4.Device Management: Device management keeps tracks of all devices.

- This module also responsible for this task is known as the I/O controller.
- It also performs the task of allocation and de-allocation of the devices.
- System manages device communication via their respective drivers.

5.I/O System Management: One of the main objects of any OS is to hide the peculiarities of that hardware devices from the user.

6.Secondary-Storage Management: Systems have several levels of storage which includes primary storage, secondary storage, and cache storage. Instructions and data must be stored in primary storage or cache so that a running program can reference it.

7.Security: Security module protects the data and information of a computer system against malware threat and authorized access.

8.Command interpretation: This module is interpreting commands given by the and acting system resources to process that commands.

9.Networking: A distributed system is a group of processors which do not share memory, hardware devices, or a clock. The processors communicate with one another through the network.

10.Job accounting: Keeping track of time & resource used by various job and users.

11.Communication management: Coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.

Operating system operations

- Program Execution
- I/O Operations
- File system manipulation
- Communication
- Error Detection
- Resource allocation
- Protection & Security

A multiprogramming operating system may run many programs on a single processor computer. If one program must wait for an input/output transfer in a multiprogramming operating system, the other programs are ready to use the CPU. As a result, various jobs may share CPU time.

The primary goal of multiprogramming is to manage the entire system's resources.

Types of the Multiprogramming Operating System

There are mainly two types of multiprogramming operating systems. These are as follows:

1.Multitasking Operating System

In operating systems, to improve the performance of more than one CPU can be used within one computer system called Multiprocessor operating system. Multiple CPUs are interconnected so that a job can be divided among them for faster execution. When a job finishes, results from all CPUs are collected and compiled to give the final output.

multiprocessing operating system effectively, the computer system must have the following things:

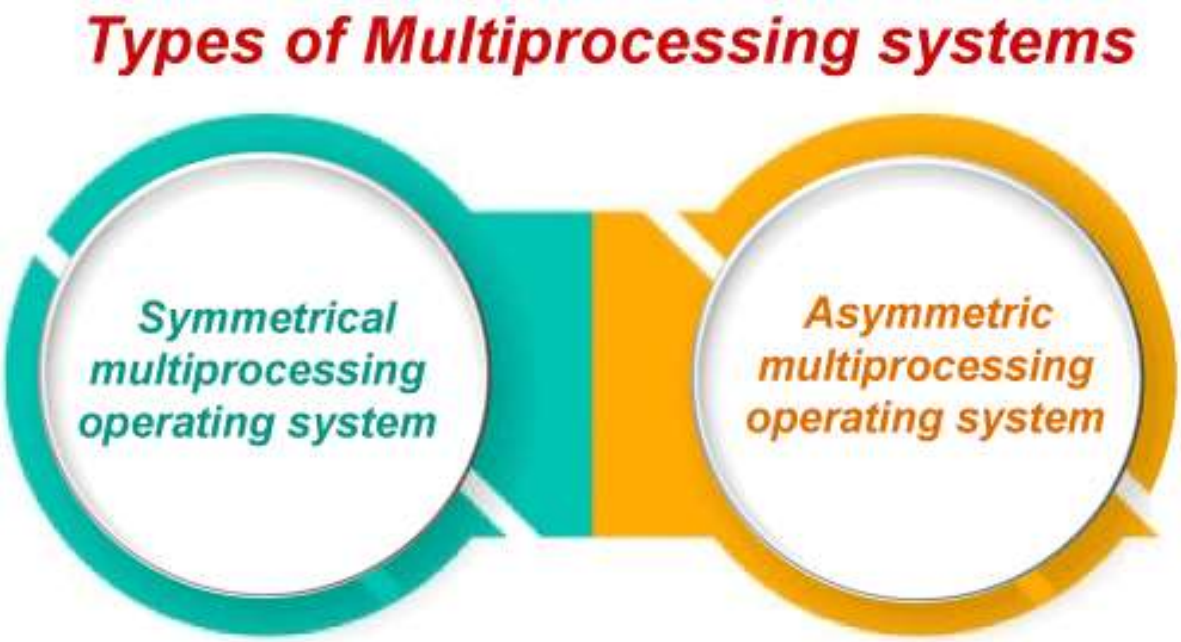
•A motherboard is capable of handling multiple processors in a multiprocessing operating system.

•Processors are also capable of being used in a multiprocessing system.

https://youtu.be/IZfWjg3U3mA

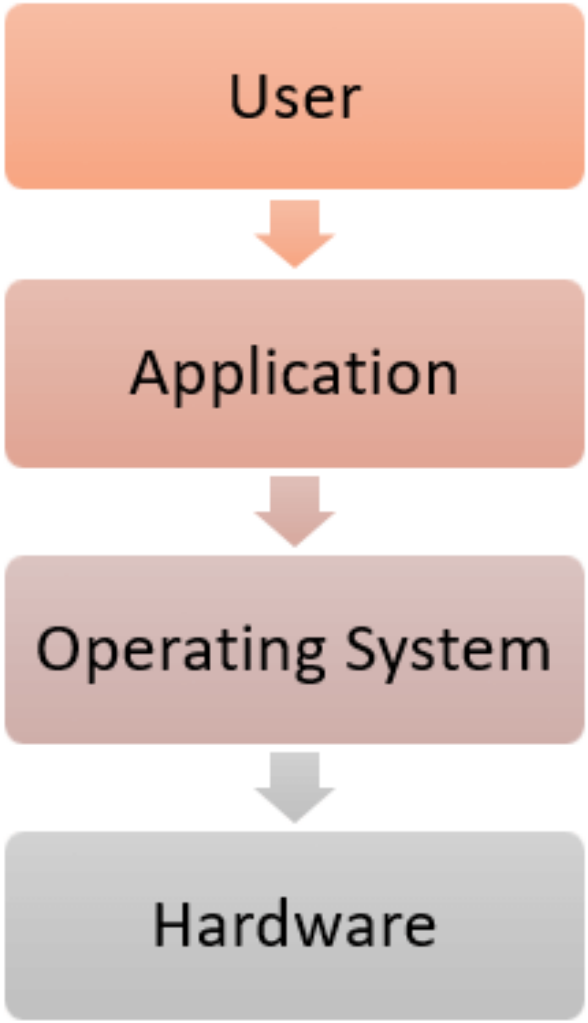
Types of multiprocessing systems

- Symmetrical multiprocessing operating system
- Asymmetric multiprocessing operating system



Features of Operating System (OS)

- Protected and supervisor mode
- Allows disk access and file systems Device drivers Networking Security
- Program Execution
- Memory management Virtual Memory Multitasking
- Handling I/O operations
- Manipulation of the file system
- Error Detection and handling
- Resource allocation
- Information and Resource Protection


User
Application
Operating System
Hardware

Advantage of Operating System

- Allows you to hide details of hardware by creating an abstraction
- Easy to use with a GUI
- Offers an environment in which a user may execute programs/applications
- The operating system must make sure that the computer system convenient to use
- Operating System acts as an intermediary among applications and the hardware components
- It provides the computer system resources with easy to use format
- Acts as an intermediator between all hardware's and software's of the system

Disadvantages of Operating System

- If any issue occurs in OS, you may lose all the contents which have been stored in your system
- Operating system's software is quite expensive for small size organization which adds burden on them. Example Windows
- It is never entirely secure as a threat can occur at any time

What is the main function of the command interpreter?

a) to provide the interface between the API and application program

b) to handle the files in the operating system

c) to get and execute the next user-specified command

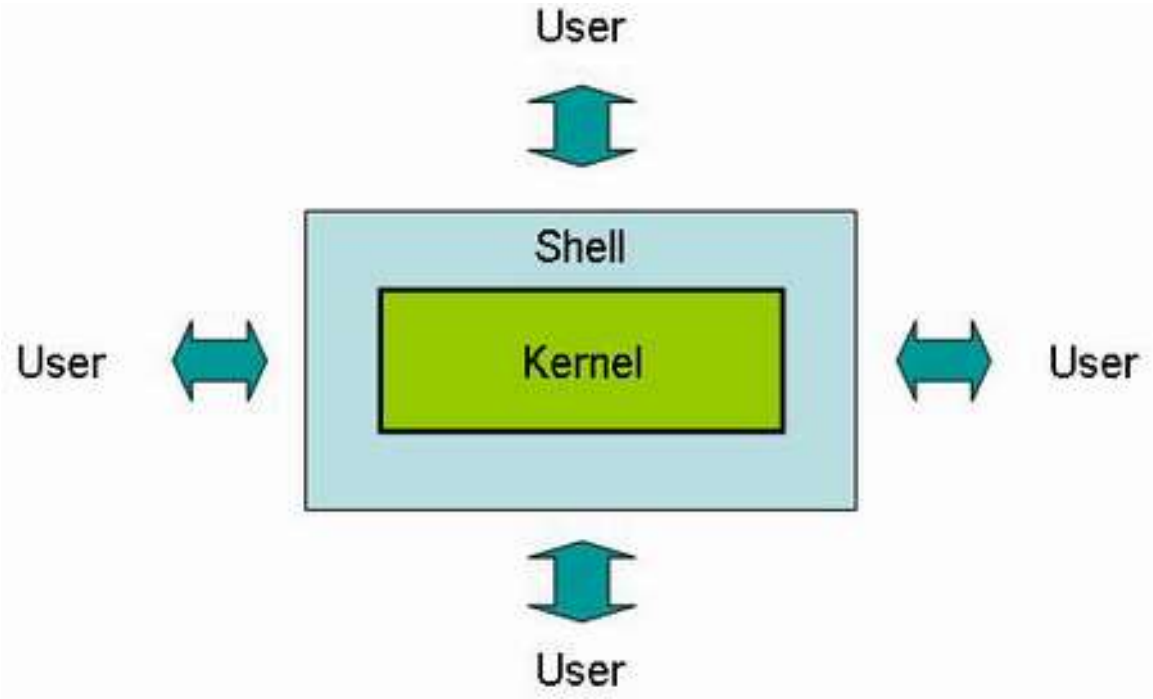
d) none of the mentioned

Components of Operating System

Now to perform the functions mentioned above, the operating system has two components:

•Shell

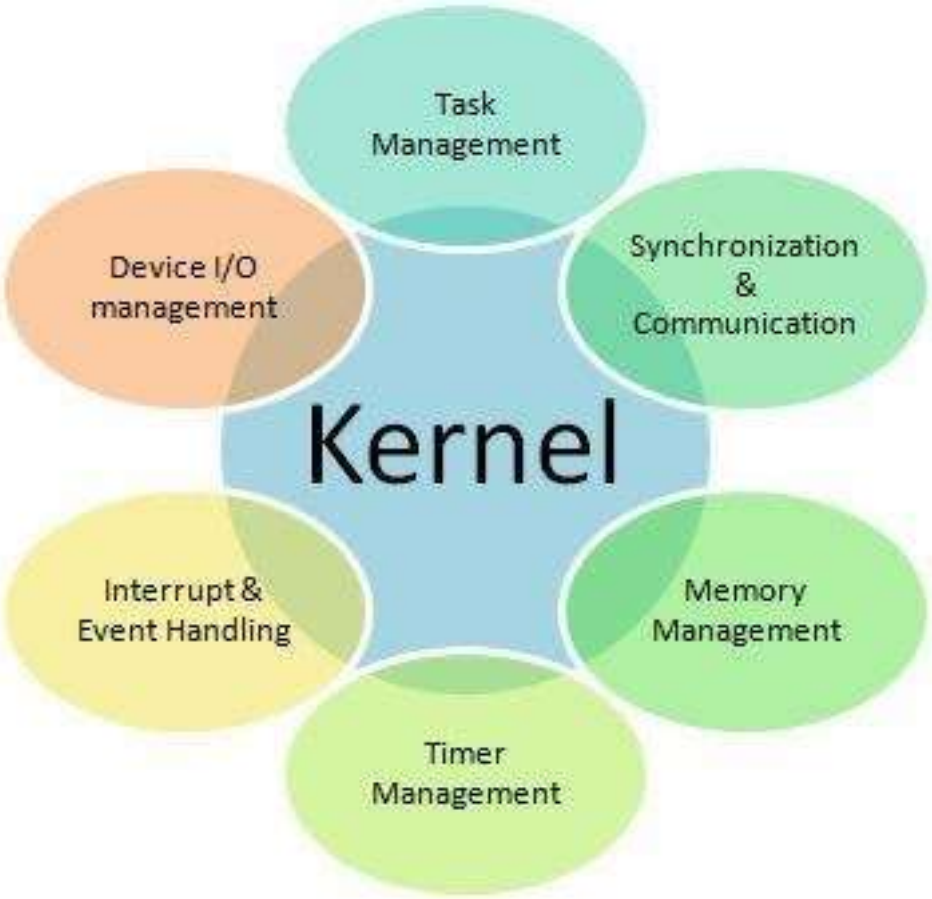
•Kernel



Shell handles user interactions. It is the outermost layer of the operating system and manages the interaction between user and operating system by:

- Prompting the user to give input
- Interpreting the input for the operating system
- Handling the output from the operating system.

Shell provides a way to communicate with the operating system by either taking the input from the user or the shell script. A shell script is a sequence of system commands that are stored in a file.


Task Management
Synchronization & Communication
Memory Management
Timer Management
Interrupt & Event Handling
Device I/O management
Kernel

Function of kernel

The kernel is the core component of an operating system which acts as an interface between applications, and the data is processed at the hardware level.

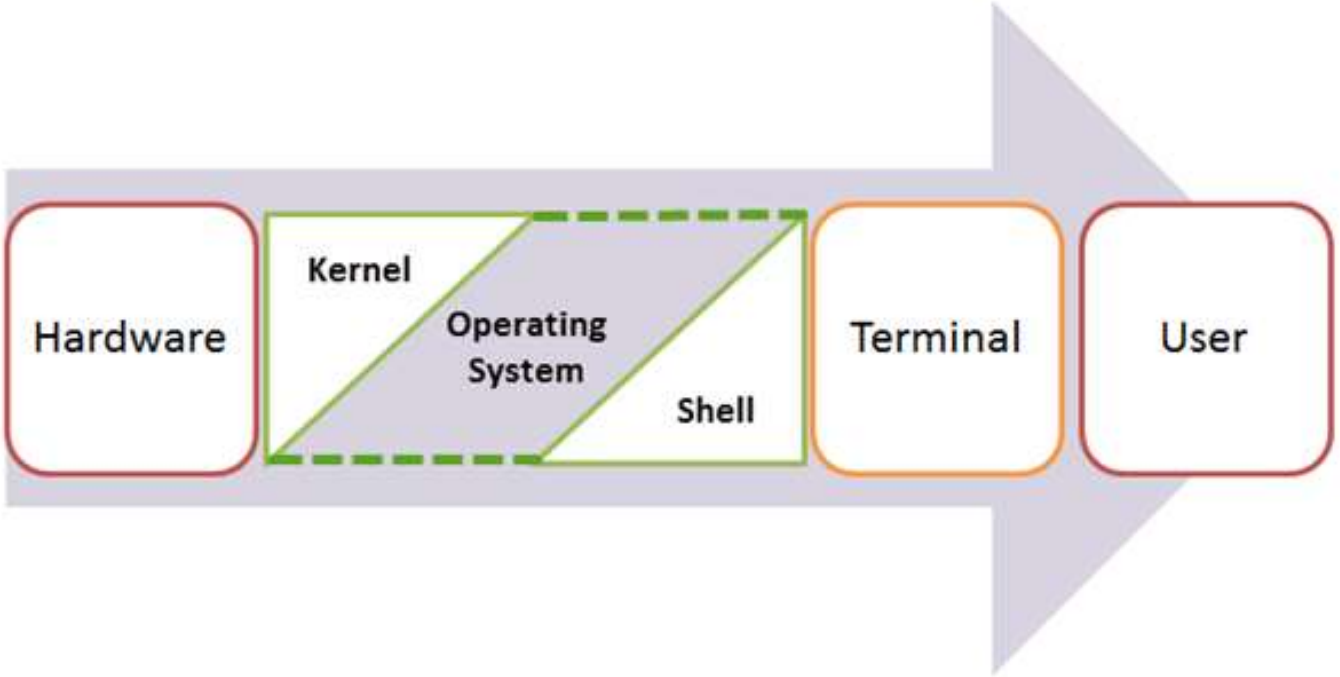
When an OS is loaded into memory, the kernel is loaded first and remains in memory until the OS is shut down. After that, the kernel provides and manages the computer resources and allows other programs to run and use these resources. The kernel also sets up the memory address space for applications, loads the files with application code into memory, and sets up the execution stack for programs

The kernel is responsible for performing the following tasks:

- Input-Output management
- Memory Management
- Process Management for application execution.
- Device Management
- System calls control

What is Kernel in Operating System?

- The kernel is the central component of a computer operating systems.
- The only job performed by the kernel is to the manage the communication between the software and the hardware.
- A Kernel is at the nucleus of a computer.
- It makes the communication between the hardware and software possible. While the Kernel is the innermost part of an operating system, a shell is the outermost one.


Hardware
Kernel
Operating System
Shell
Terminal
User

Features of Kernel

- Low-level scheduling of processes
- Inter-process communication
- Process synchronization
- Context switching

Types of Kernel

There are many types of kernels that exists, but among them, the two most popular kernels are:

1. Monolithic

A monolithic kernel is a single code or block of the program. It provides all the required services offered by the operating system. It is a simplistic design which creates a distinct communication layer between the hardware and software.

2. Microkernels

Microkernel manages all system resources. In this type of kernel, services are implemented in different address space. The user services are stored in user address space, and kernel services are stored under kernel address space. So, it helps to reduce the size of both the kernel and operating system.

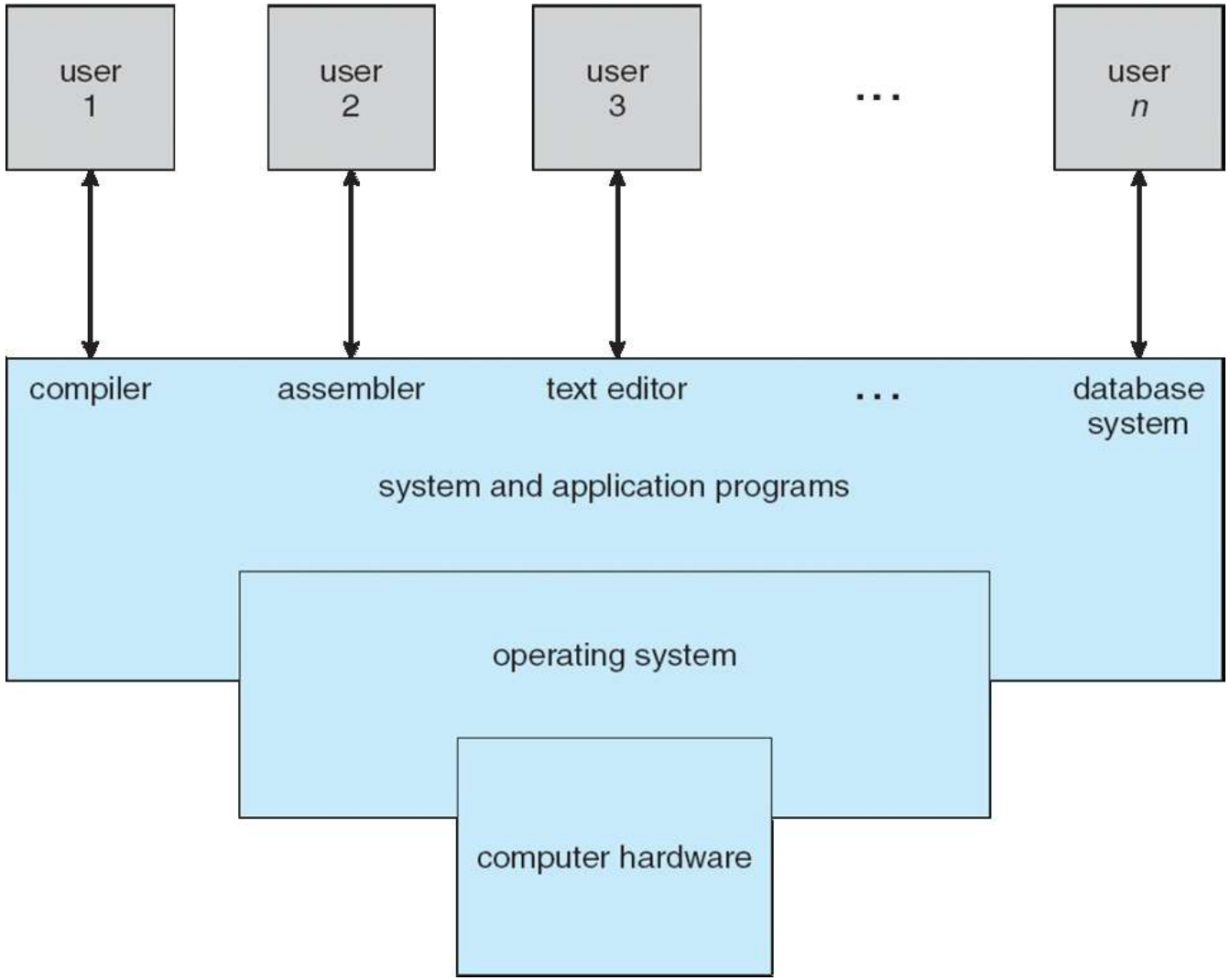
Difference between 32-Bit and 64-Bit Operating System

Parameters	32. Bit	64. Bit
Architecture and Software	Allow 32 bit of data processing simultaneously	Allow 64 bit of data processing simultaneously
Compatibility	32-bit applications require 32-bit OS and CPUs.	64-bit applications require a 64-bit OS and CPU.
Systems Available	All versions of Windows 8, Windows 7, Windows Vista, and Windows XP, Linux, etc.	Windows XP Professional, Vista, 7, Mac OS X and Linux.
Memory Limits	32-bit systems are limited to 3.2 GB of RAM.	64-bit systems allow a maximum 17 Billion GB of RAM.

Computer System Structure

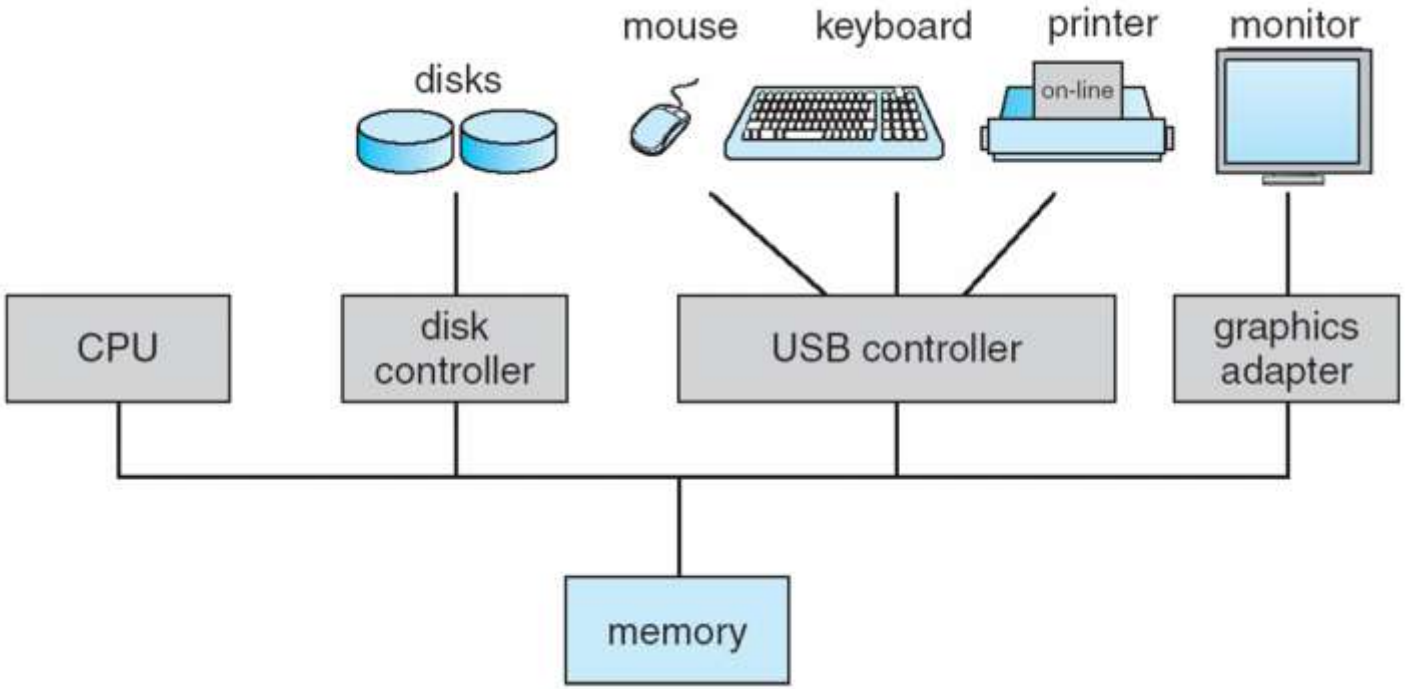
- **Computer system can be divided into four components:**
  - **Hardware** – provides **basic computing resources**
    - CPU, memory, I/O devices
  - **Operating system**
    - **Controls and coordinates use of resources** among various applications and users
  - **System/Application programs** – define the ways in which the system resources are used to solving user problems
    - Word processors, compilers, web browsers, database systems, video games
  - **Users**
    - People, machines, other computers

Four Components of a 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



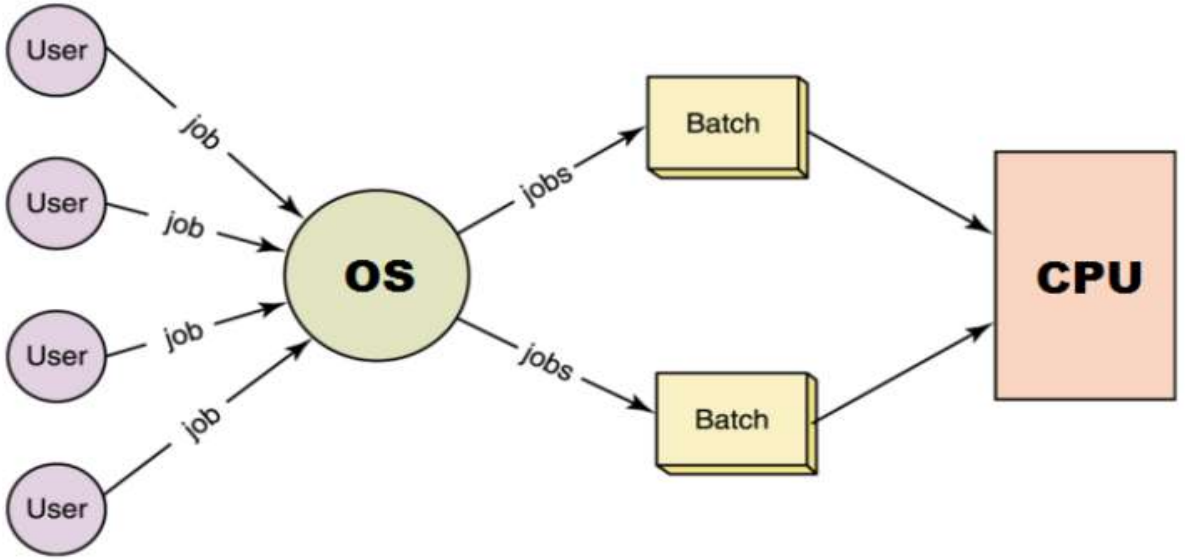
TYPES OF OS

Batch Systems

“Batch operating system. The users of a batch operating system do not interact with the computer directly.

Each 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”.



m]

TYPES OF OS: Batch Systems

- There is no direct interaction between user and the computer.
- The user has to submit a job (written on cards or tape) to a computer operator.
- Then computer operator places a batch of several jobs on an input device.
- Jobs are batched together by type of languages and requirements.

Disadvantages:

- No interaction between user and computer.
- No mechanism to prioritize the processes

TYPES OF OS: Batch Systems

- Early computers were machines run from a console
- The common **input devices** were **card readers and tape drives**.
- The common **output** devices were **line printers, tape drives, and card punches.**
- To speed up processing, operators batched jobs with similar needs together and ran them through the computer as a group.

Multiprogrammed OS

Multiprogramming: When 2 or more processes reside in memory at the same time

- Single user processes cannot keep CPU and I/O devices busy at all times
- **Multiprogramming organizes jobs (code and data) so CPU always has one to execute**
- Multiprogramming assumes a **single shared processor.** One job selected and run via **job scheduling**
- Multiprogramming increases CPU utilization.
- **It is mixture of I/O bound and CPU bound processes**

Multiprogrammed OS

- In this the operating system picks up and begins to execute one of the jobs from memory.
- Once this job needs an I/O operation operating system switches to another job (CPU and OS always busy).
- Jobs in the memory are always less than the number of jobs on disk(Job Pool).

Multiprogrammed OS

- If several jobs are ready to run at the same time, then the system chooses which one to run through the process of **CPU Scheduling**.

- In Non-multiprogrammed system, there are moments when CPU sits idle and does not do any work.

- In Multiprogramming system, CPU will never be idle and keeps on processing.

Multitasking/Timesharing OS

- **Timesharing (multitasking)** when multiple jobs are executed by the CPU simultaneously by switching between them.
  - There is at least one program is 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
  - **Only one CPU is involved**, but **it switches** from one process to another **so quickly** that it gives the appearance of **executing all of the processes at the same time.**

Multiprocessing OS

- A multiprocessor system consists of several processors that share a common physical memory.
- Multiprocessor system provides higher computing power and speed.
- In multiprocessor system all processors operate under single operating system.

Multiprocessing OS

- Multi-processor systems; that is, they **have multiple CPU.**

- **Also known as parallel systems** or **tightly coupled systems**

- **Such systems have more than one processor** in close communication, sharing the computer bus, the clock, and sometimes memory and peripheral devices.

Distributed Systems

- A network is a communication path between two or more systems.
- Each system over the network keeps copy of the data, and this leads to Reliability (Because if one system crashes , data is not lost).

- CLIENT SERVER SYSTEMS
- PEER TO PEER SYSTEMS

Real Time Systems

- **Time bound systems**
- Real time systems are of 2 types:

- **1. Soft Real time Systems:** Process should complete in specific time but May have some delay (Positive delay) and will not harm the system.

Exp: Session expires but can be re-logged in.

- **2. Hard Real Time Systems:** Each process is assigned a specific time instance, and Process must complete in that time otherwise system will crash.

Real Time Embedded Systems

- is a computing environment that **reacts to input within a specific time period.**
- Time Driven
- Task specific
- Exp: Microwave, Washing Machine…

Operating System Views

OS can be explored from 2 view points:

1. User view:

- The goal of the Operating System is to **maximize the work and minimize the effort of the user**.
- Operating System gives an effect to the **user as if the processor is dealing only with the current task**, but in background processor is dealing with several processes.

Operating System Views

OS can be explored from 2 view points:

2. System View:

Operating System is a program involved with the hardware.

- OS is a **resource allocator**
  - Allocates and Manages all resources and their sharing.
  - 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**
  - **It prevents** improper usage, error and handle **deadlock** conditions.

Operating-System Operations

- **OS's are Interrupt driven**. If no process, no I/o devices, No users then OS will sit quietly waiting for some event to occur.

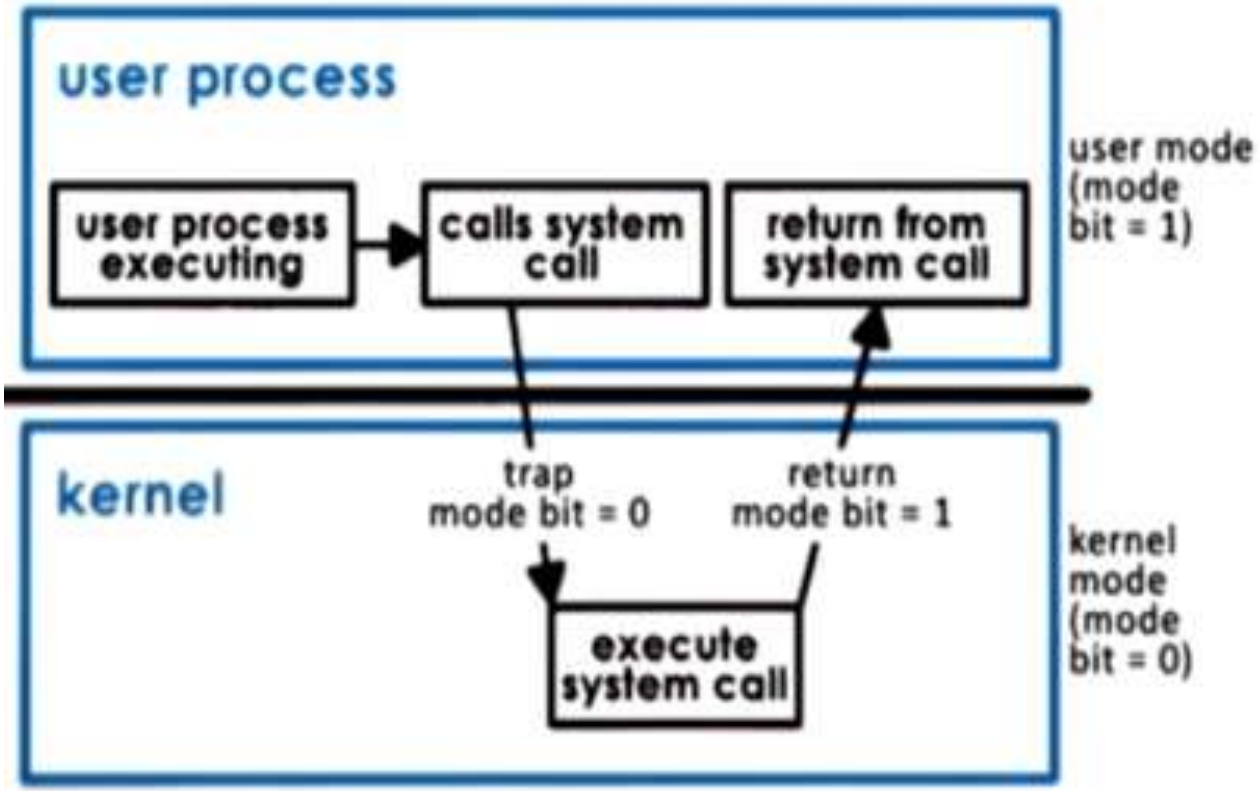
- Program or **software send, Generate Events by using system calls**. Error or request by a software creates **exception** or **trap**
  - E.g. Division by zero

- **To ensure proper execution of OS, we must distinguish between execution of OS code and user defined code.**

Operating-System Operations

- **To protect OS, Dual-mode** operations exist:
  - **User mode (1)** and **kernel mode (0)**
  - A **Mode bit is added to** hardware to indicate mode
    - Provides ability to distinguish when system is running user mode or kernel mode
    - System call changes mode to kernel, return from call resets it to user

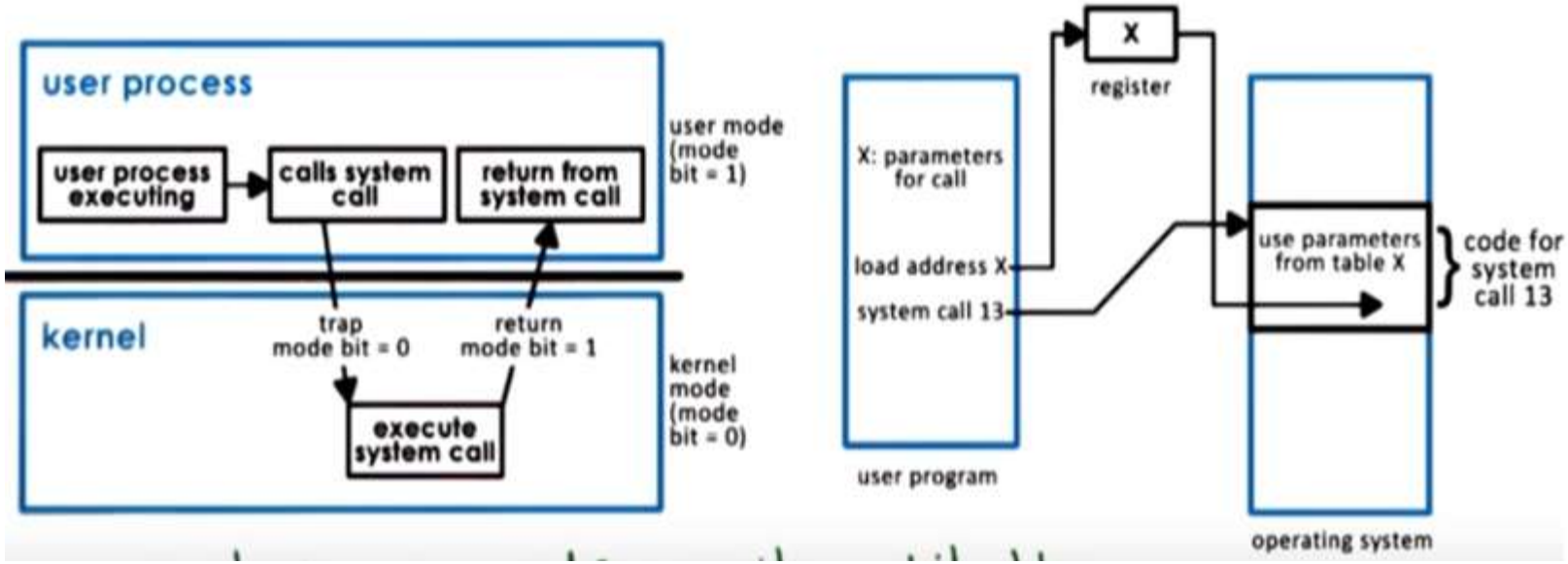
Transition from User to Kernel Mode



To make a system call an application must

- write argoments
- save relevant data at well-defined location
- make system call

Transition from User to Kernel Mode

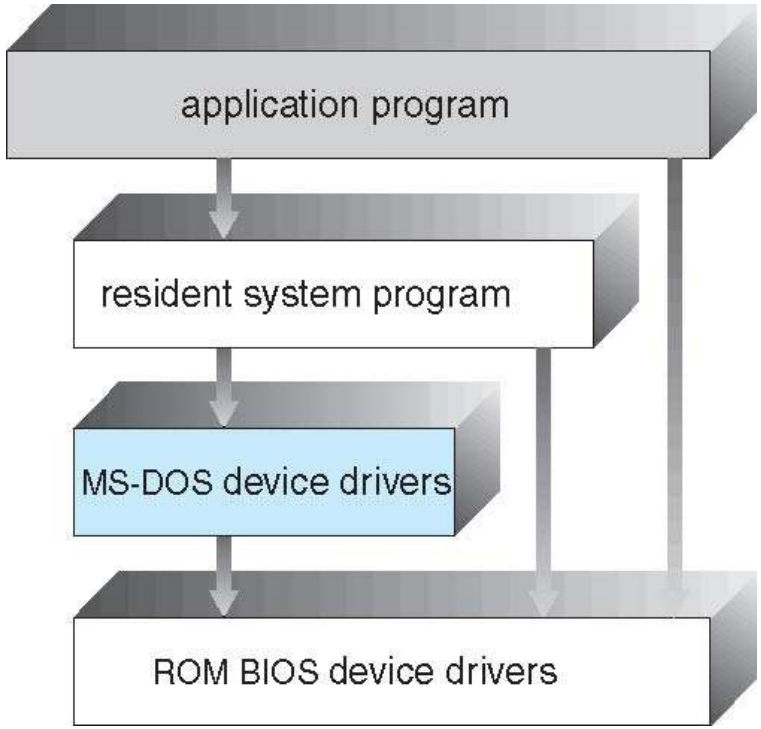


Operating System Structure

- Various ways to structure a system

Simple Structure -- MS-DOS

- MS-DOS – written to provide the most functionality in the least space
  - Not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated

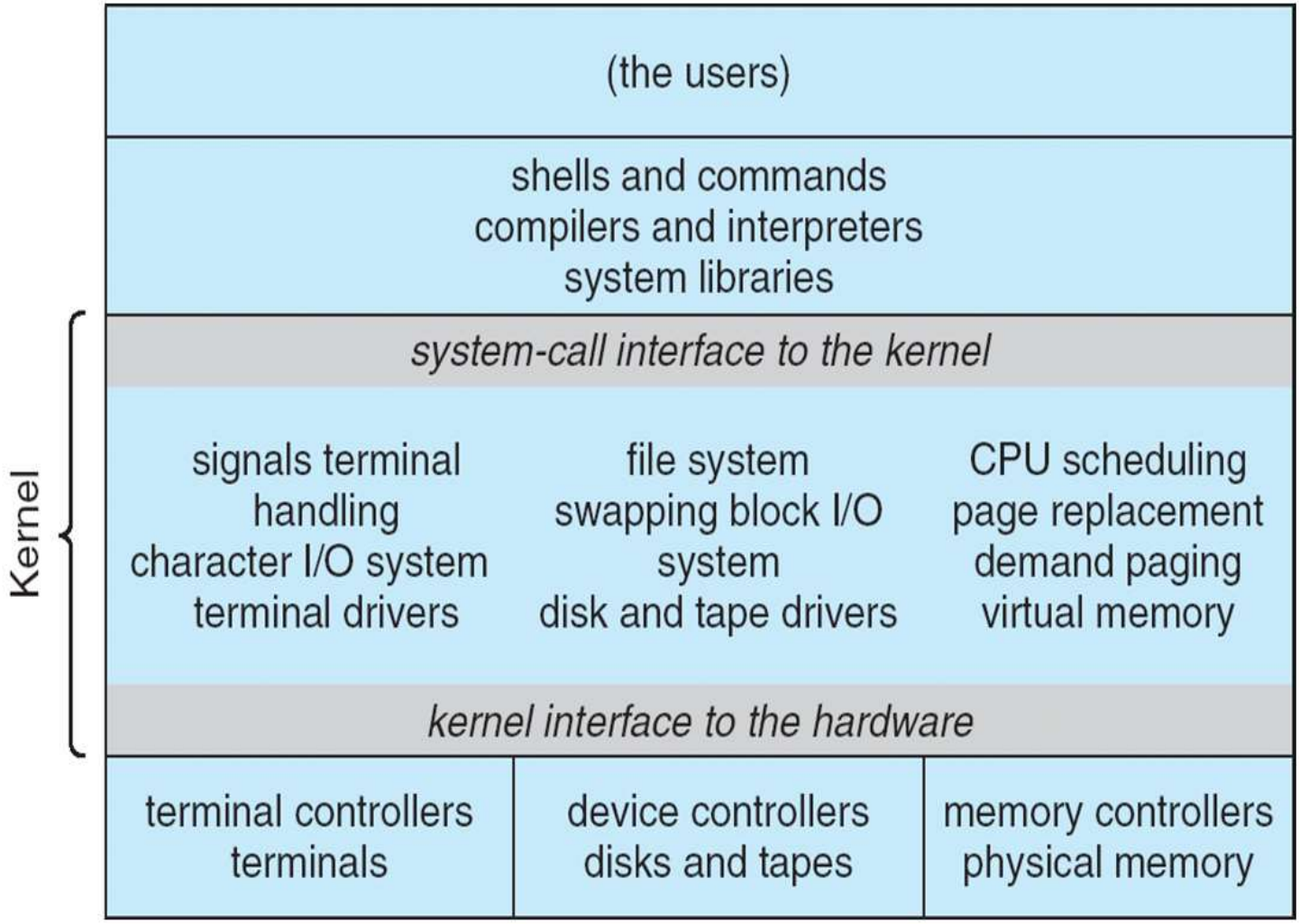


Non Simple Structure -- UNIX

The UNIX OS consists of two parts:

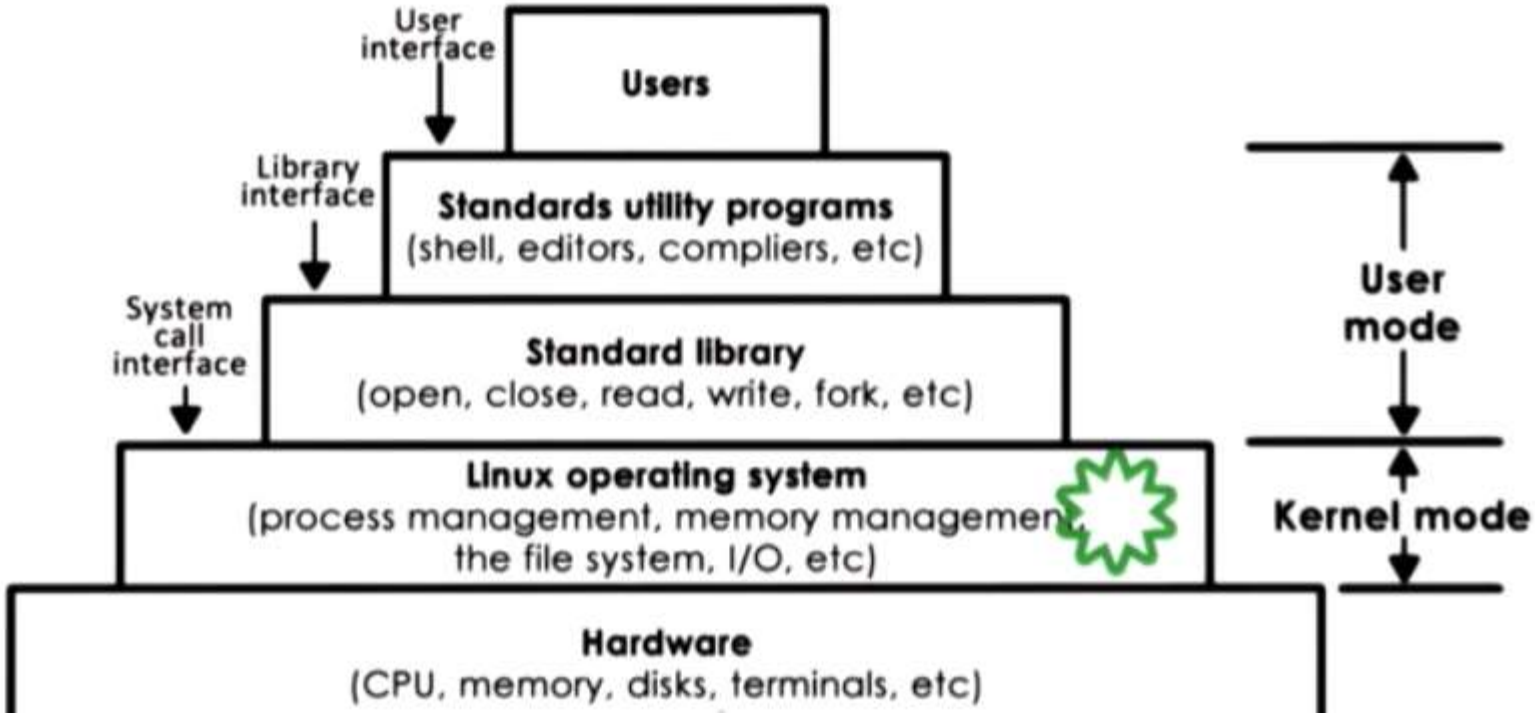
- System programs
- The kernel
  - Consists of everything below the system-call interface and above the physical hardware
  - Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level

Traditional UNIX System Structure

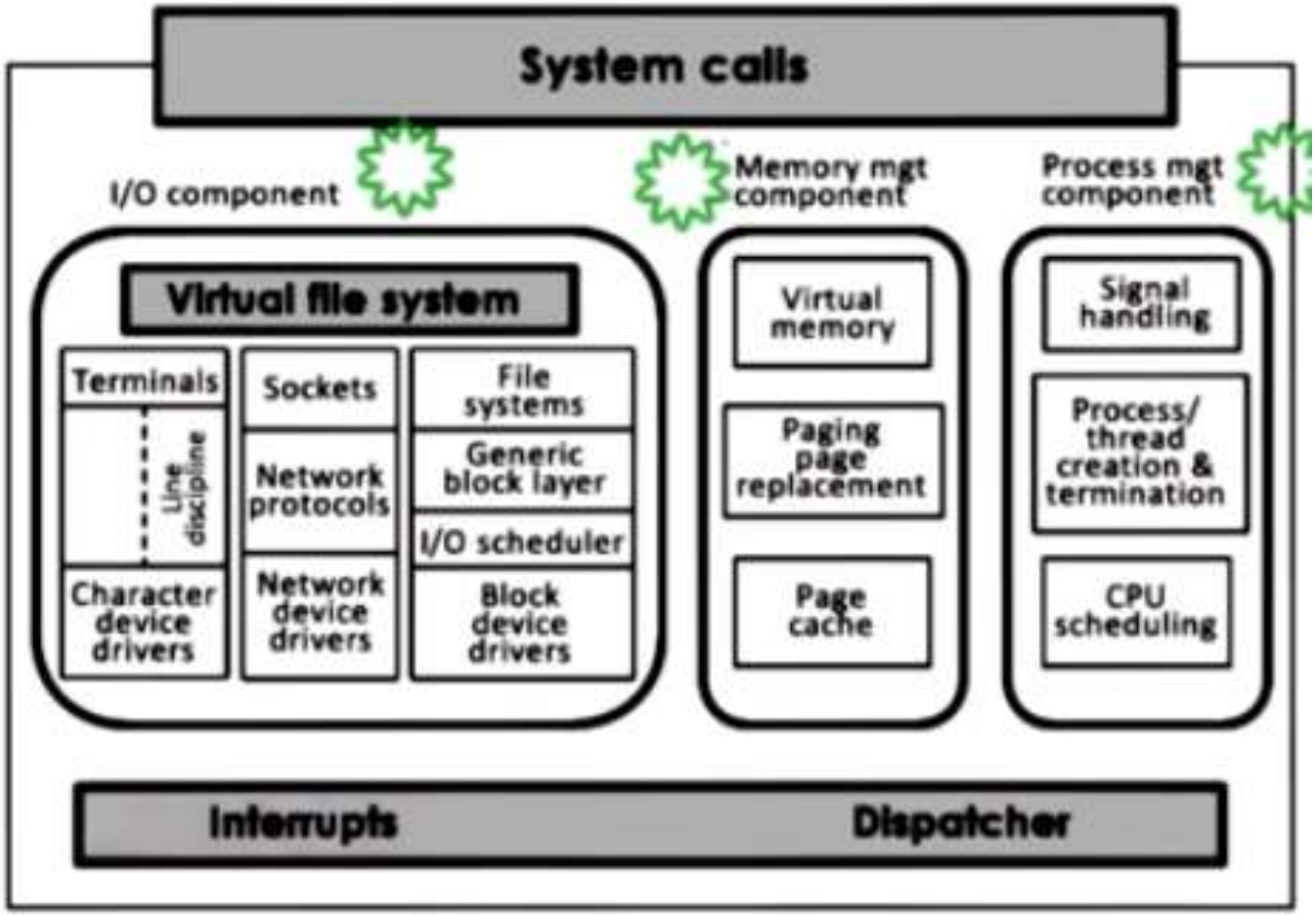


Traditional LINUX System Structure

Linux Architecture

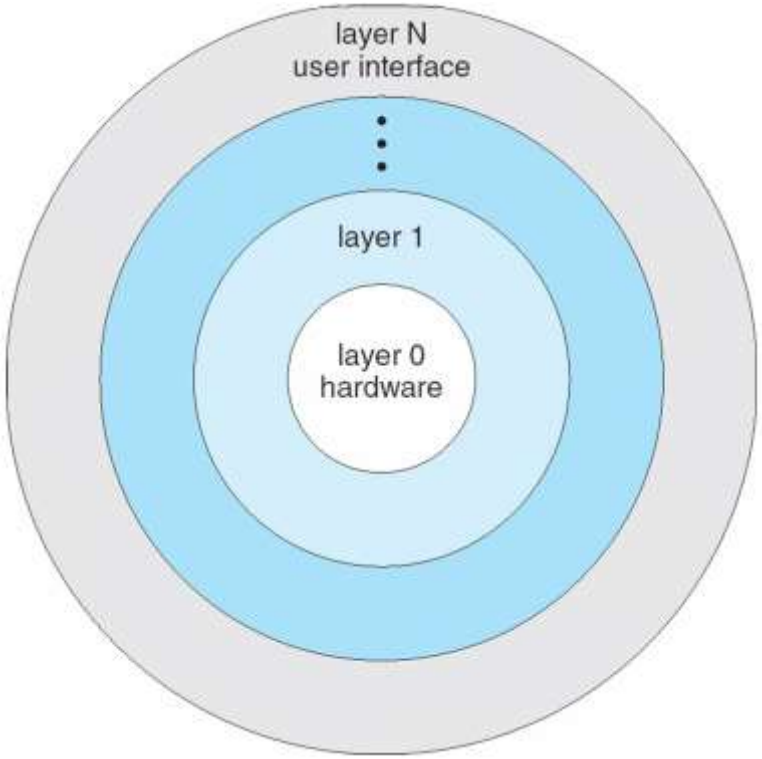


Kernel has many inbuilt modules



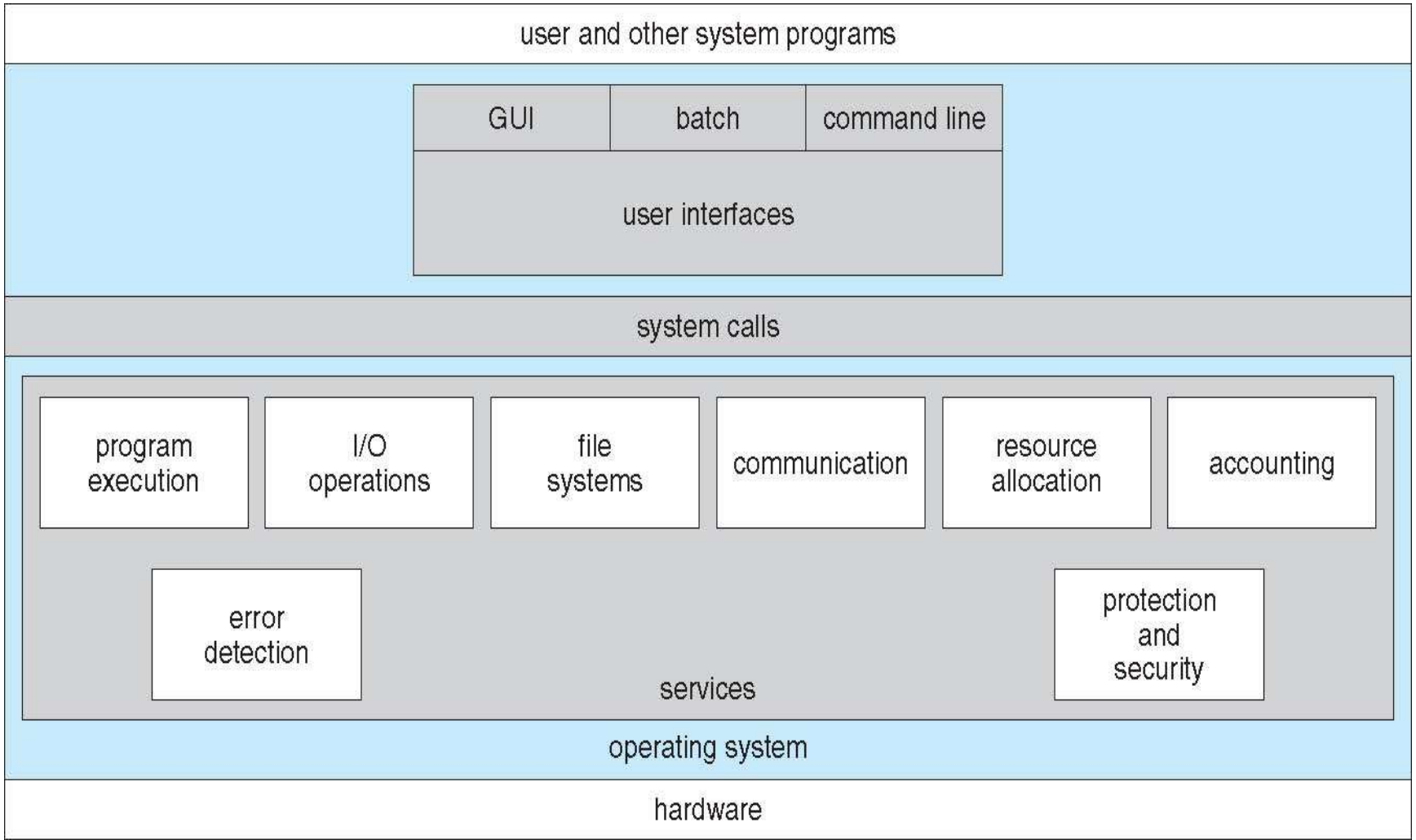
Layered Approach

- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers



Operating System Services

- An operating system provides an **environment for the programs to run.**
- It provides certain services to programs



Operating System Services

- 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).
  - **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.
  - **File-system manipulation** - read and write files and directories, create and delete them, search them, list file Information, permission management.

Operating System Services

- **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.

- **Resource allocation –** OS must ensure allocation of resources to all programs running.
  - **Many types of resources** - such as **CPU cycle time, main memory, and file storage**, **I/O devices**

Operating System Services

- **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
  - If a system is to be protected and secure, precautions must be instituted throughout it.
  - **A chain is only as strong as its weakest link.**

Kernel

- A kernel is a central component of an operating system.
- It acts as an interface between the user applications and the hardware.
- The sole aim of the kernel is to manage the communication between the software (user level applications) and the hardware (CPU, disk memory etc).
- The main tasks of the kernel are :
  - Process management
  - Device management
  - Memory management
  - Interrupt handling
  - I/O communication
  - File system...etc..

Kernel

- A kernel is the lowest level of software that interfaces with the hardware in your computer.

- It is responsible for interfacing all applications that are running in “user mode” down to the physical hardware, and allowing processes, to get information from each other using inter-process communication (IPC).

Kernel Types

kernels fall into one of three types:

- Monolithic
- Microkernel
- Hybrid

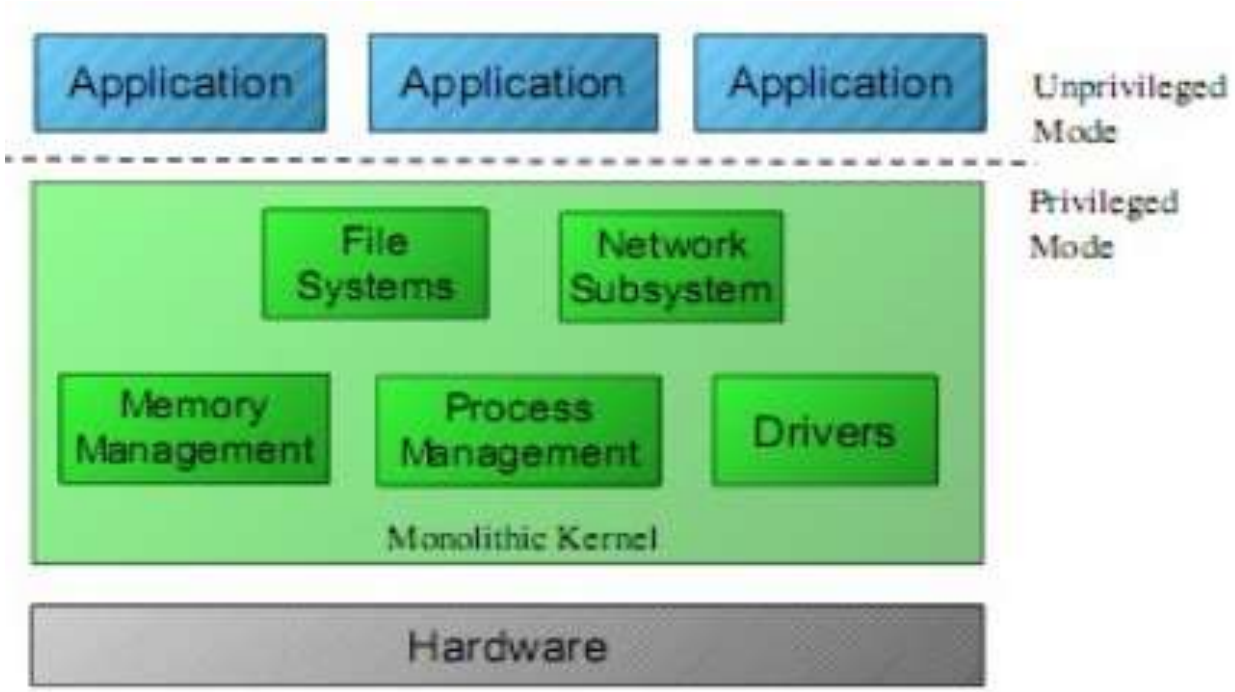
Monolithic Kernel

- A **monolithic kernel** is an operating system architecture where the entire operating system (which includes the device drivers, file system, and the application IPC etc.) is working in kernel space, in supervisor mode.

- Monolithic kernels are able to dynamically load (and unload) executable modules at runtime.

- Examples of operating systems that use a monolithic kernel are - Linux, Solaris, OS-9, DOS, Microsoft Windows (95,98,Me) etc.

Monolithic Kernel



Monolithic Kernel

Pros:

- More direct access to hardware for programs
- Easier for processes to communicate between each other
- If your device is supported, it should work with no additional installations
- Processes react faster because there isn't a queue for processor time.

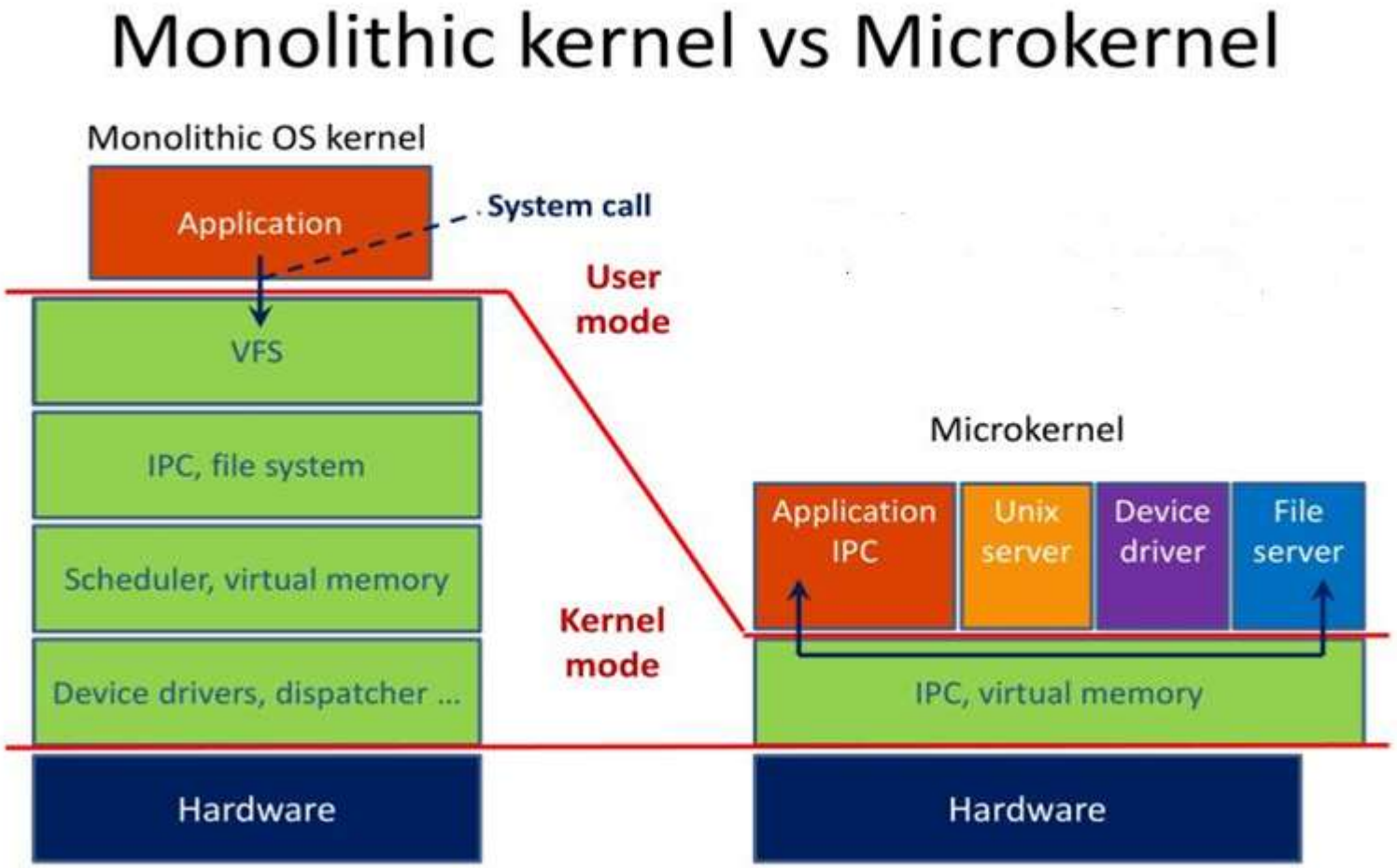
Cons:

- Large install footprint
- Large memory is needed
- Less secure because everything runs in supervisor mode

MicroKernel

- In a Microkernel architecture, the core functionality is isolated from system services and device drivers.
- This architecture allows some basic services like device driver management, file system etc. to run in user space.
- This reduces the kernel code size and also increases the security and stability of OS as we have minimum code running in kernel.
- Examples of operating systems that use a microkernel are - QNX, Integrity, PikeOS, Symbian, L4Linux, Singularity, K42, Mac OS X, HURD, Minix, and Coyotos.

Types of Kernel



MicroKernel

Pros

- Portability
- Small install footprint
- Small memory
- Security

Cons

- Hardware is more abstracted through drivers
- Hardware may react slower because drivers are in user mode
- Processes have to wait in a queue to get information
- Processes can’t get access to other processes without waiting

HybridKernel

- Hybrid kernels have the ability to pick and choose what they want to run in user mode and what they want to run in supervisor mode.
- Device drivers and file system I/O will be run in user mode while IPC and server calls will be kept in the supervisor mode.
- This require more work of the hardware manufacturer because all of the driver responsibility is up to them.

Hybrid Kernel

Pros

- Developer can pick and choose what runs in user mode and what runs in supervisor mode
- Smaller install than monolithic kernel
- More flexible than other models

Cons

- Processes have to wait in a queue to get information
- Processes can’t get access to other processes without waiting
- Device drivers need to be managed by user

Interrupts

- An interrupt is a signal from a device attached to a computer or from a program within the computer that causes the main program that operates the computer (the operating system) to stop and figure out what to do next.

- Interrupts can be of following type:

  - Generated by Hardware (Hardware Interrupt)

  - Generated by Software (Software Interrupt)

Hardware Interrupt

1. Hardware interrupts are used by **devices** to communicate that they **require attention from the operating system**.
2. Hardware interrupts by sending signal to CPU **via system bus.**
3. Hardware interrupts are referenced by an ***interrupt number*.**
4. These numbers are mapped with hardware that created the interrupt. This enables the system to monitor/understand **which device created the interrupt and when it occurred**.

Software Interrupt/ Trap

- Interrupt generated by executing a instruction.
- Software interrupts by a special operation called a System Call or Monitor Call.

Exp: 1. cout in C++ is a kind of interrupt because it would make a system to print something.

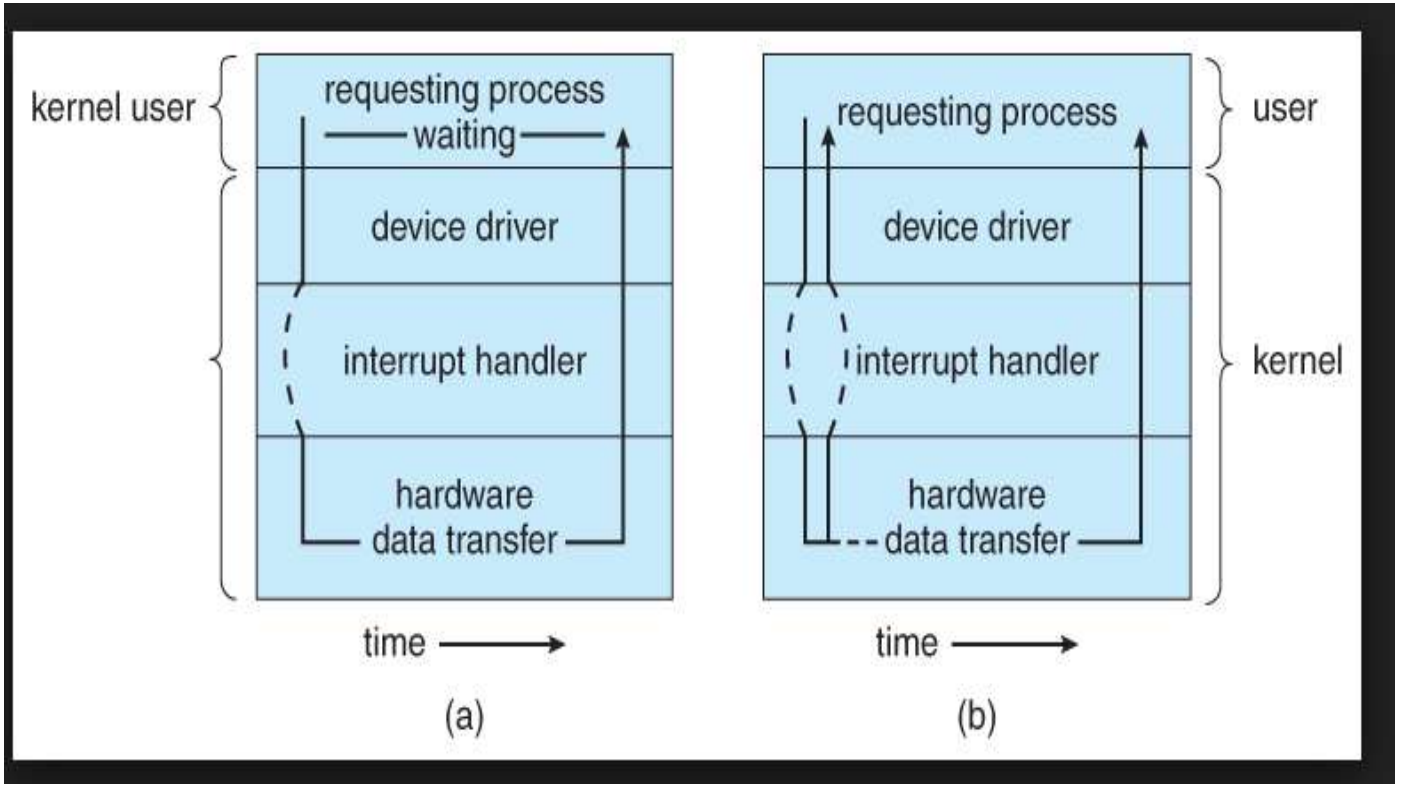
2. division by zero

Basic steps when interrupt occurs:

1. Interrupt Occurred?
2. H/w Transfers control to OS
3. OS preserves current state of process by using Registers and Program counter
4. Determine which kind of interrupt has occurred and Provides resources
5. When interrupt is executed, address is loaded to program counter and interrupted services are resumed.

I/O Interrupt

- Once the I/O is started, 2 kind of interrupts may arise:
- **Synchronous interrupt (I/O):** The control is transferred to the user process when I/O completes.
- **Asynchronous interrupt (I/O):** Returns control to user process without waiting for I/O to complete.



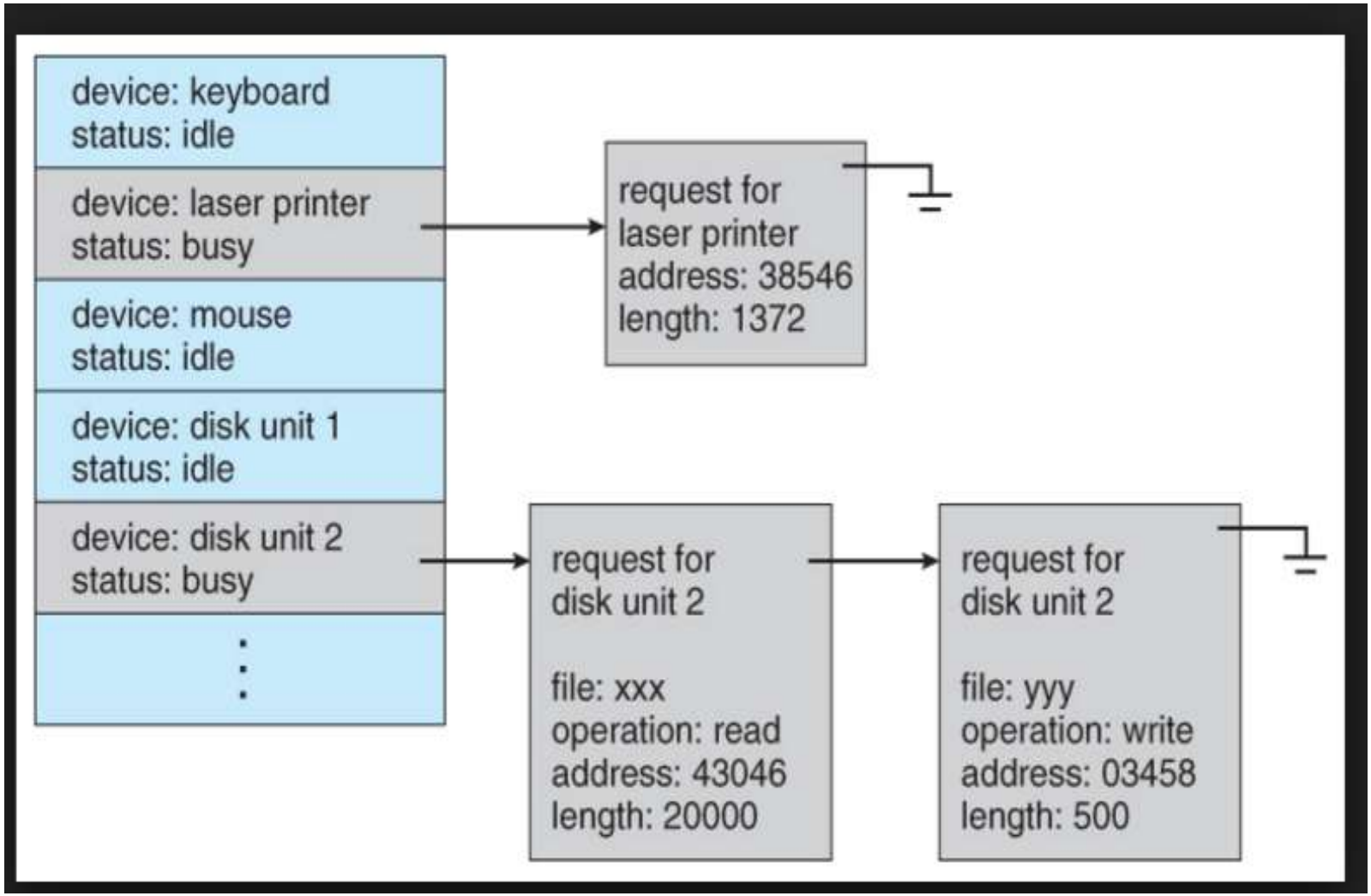
(a) Synchronous (b) Asynchronous

Waiting of I/O may be accomplished in 2 ways:

- Wait: it let the CPU wait till the next interrupt.
- Loop: Loop continues until interrupt is active.

Device Status Table

- To keep the track of how many I/O requests are pending
- Device Status Table is maintained.
  - Device type
  - Address
  - State (idle,busy)


device: keyboard
status: idle
device: laser printer
status: busy
device: mouse
status: idle
device: disk unit 1
status: idle
device: disk unit 2
status: busy
request for
laser printer
address: 38546
length: 1372
request for
disk unit 2
file: xxx
operation: read
address: 43046
length: 20000
request for
disk unit 2
file: yyy
operation: write
address: 03458
length: 500

System Calls

- Allow user-level processes to request services of the operating system.
- It provides a way in which program talks to the operating system.

Why system calls are required?

- **It is a request to the operating system to perform some activity**.
- It is a **call to the kernel** in order to **execute a specific function** that controls a device or executes a instruction.
- A system call looks like a procedure call

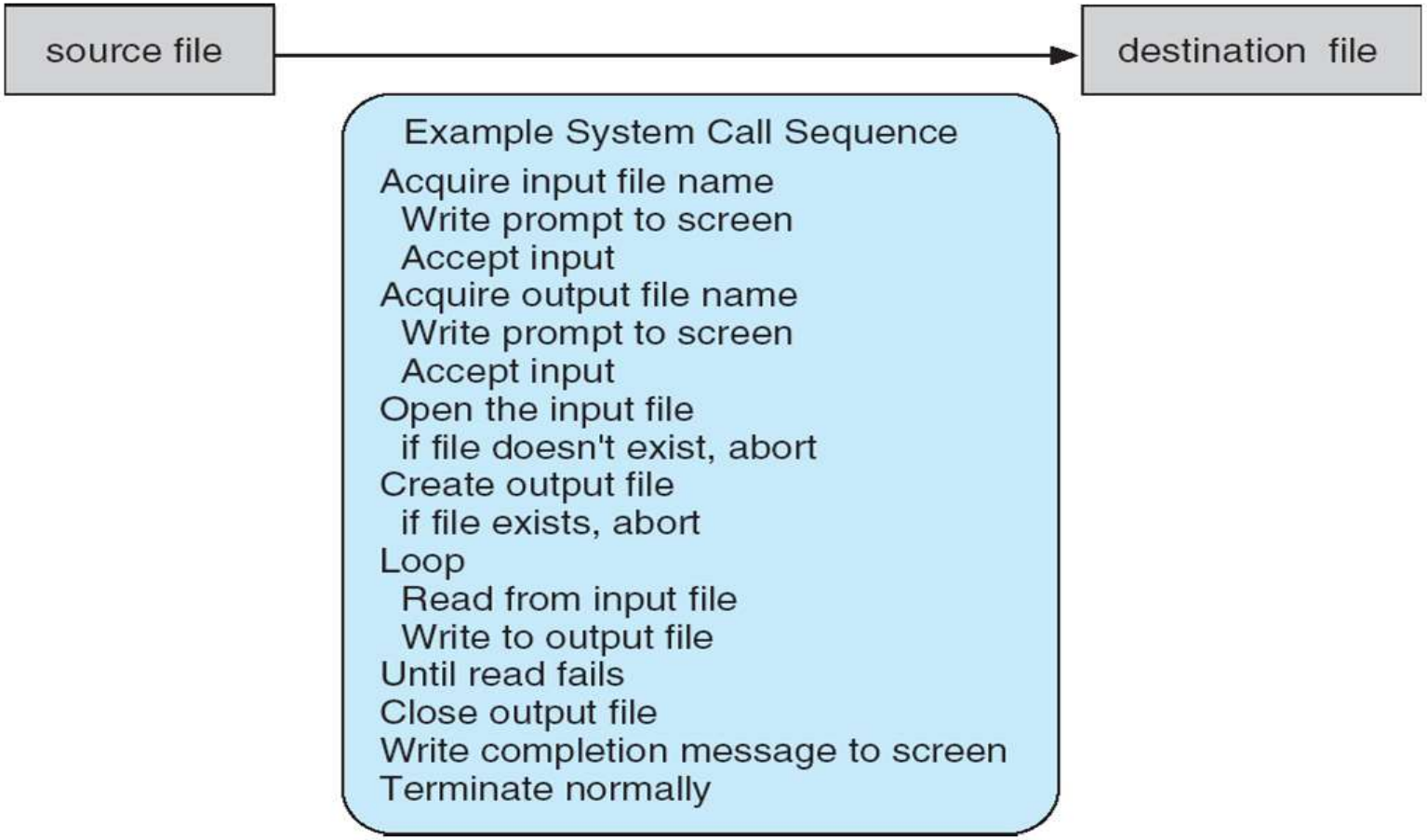
Accessing System Calls

- System calls are accessed by programs via a high-level **Application Program Interface (API)** (provides run time environment)

- System calls are implemented via API, API's interact with kernel of OS.

Example of System Calls

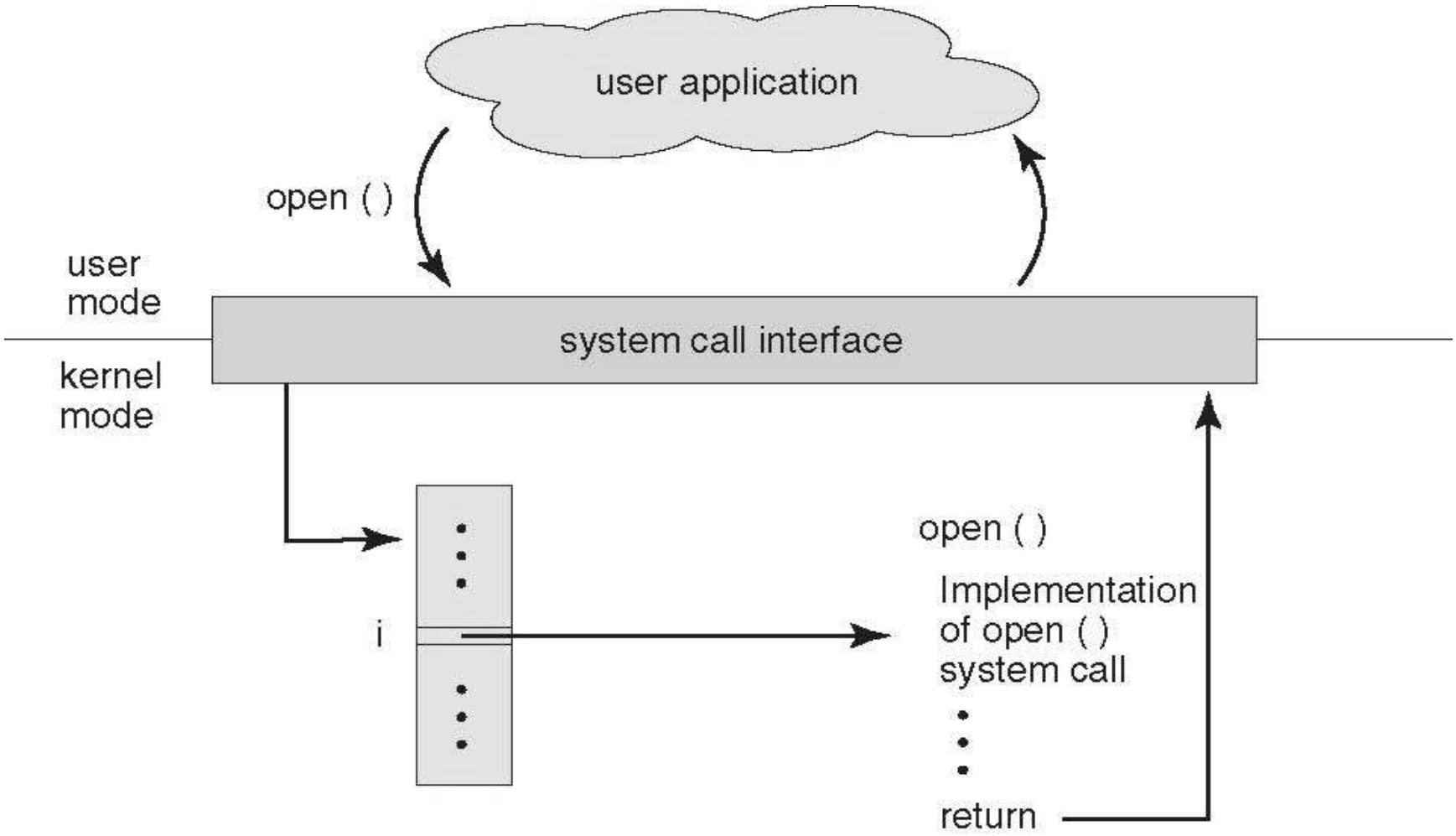
- System call sequence to copy the contents of one file to another file



System Call Implementation

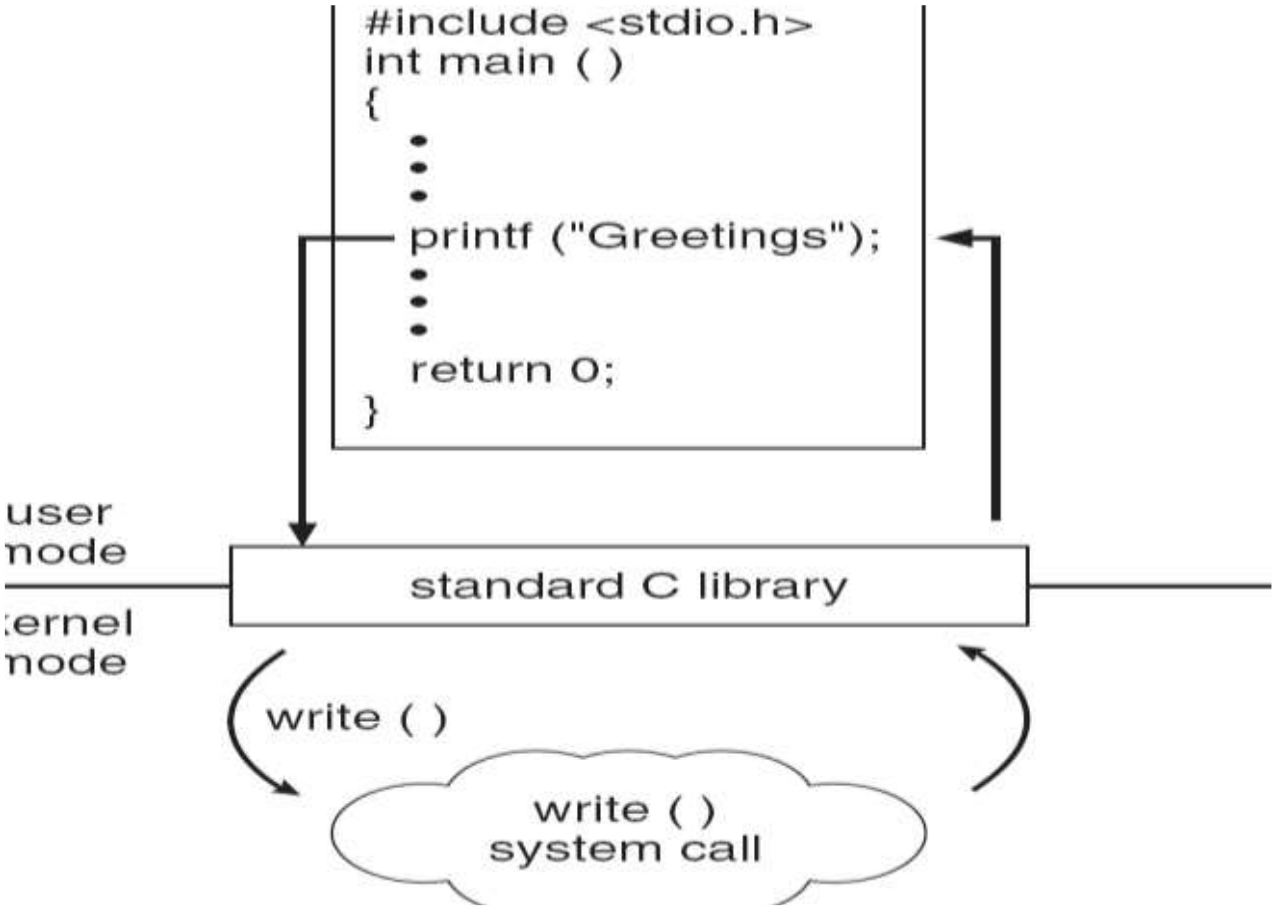
- A number is associated with each system call
  - System-call interface maintains a table indexed according to these numbers
- The system call interface invokes intended system call in OS kernel and returns status of the system call with a return value.
- The caller needs to know nothing about how the system call is implemented
  - Just needs to obey API and understand what OS will do as a result call
  - Most details of OS interface hidden from programmer by API

API – System Call – OS Relationship



Standard C Library Example

- C program invoking printf() library call, which calls write() system call



System Call Parameter Passing

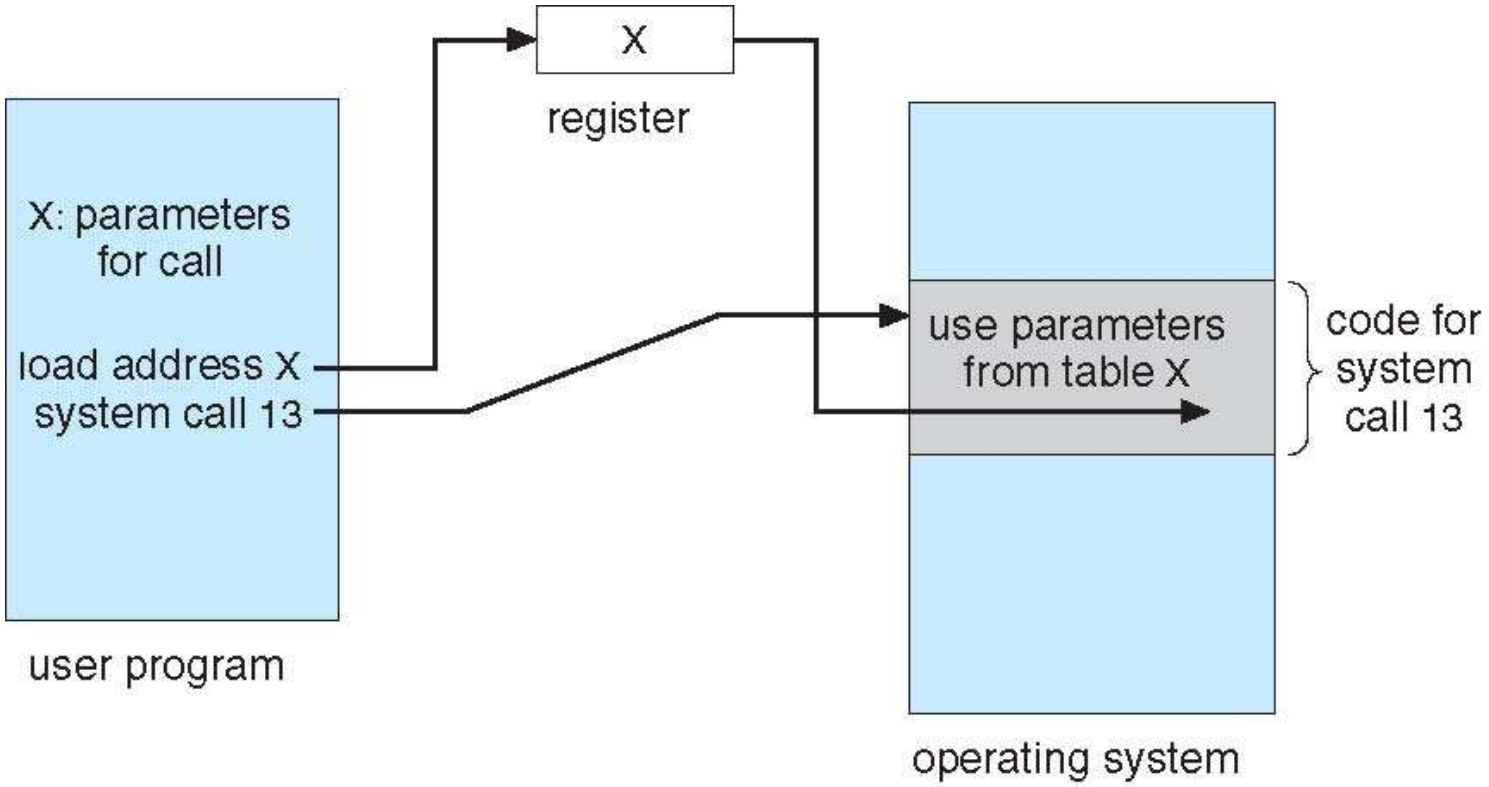
Passing Parameters to System Calls:

- Information required for a system call vary according to OS and call.

- **Three general methods used to pass parameters to the OS**

  1. **Pass the parameters in *registers***
     - When parameters are < 6.
  2. **Parameters stored in a *block,* or table**, in memory, and address of block passed as a parameter in a register. (6 or more)
     - This approach taken by Linux and Solaris
  3. **Parameters placed**, or *pushed,* **onto the *stack*** by the program and *popped* off the stack by the operating system.

Parameter Passing via Table



Types of System Calls

5 Categories

- Process Control
  - end, abort
  - load, execute
  - create process, terminate process
  - get process attributes, set process attributes
  - wait for time
  - wait event, signal event
  - allocate and free memory

- File Management
  - create file, delete file
  - open, close file
  - read, write, reposition
  - get and set file attributes

Types of System Calls (Cont.)

- Device Management
  - request device, release device
  - read, write, reposition
  - get device attributes, set device attributes
  - logically attach or detach devices
- Information Maintenance
  - get time or date, set time or date
  - get system data, set system data
  - get and set process, file, or device attributes

- Communications
  - create, delete communication connection
  - send, receive messages
  - transfer status information
  - attach and detach remote devices

Storage Structure and Hierarchy