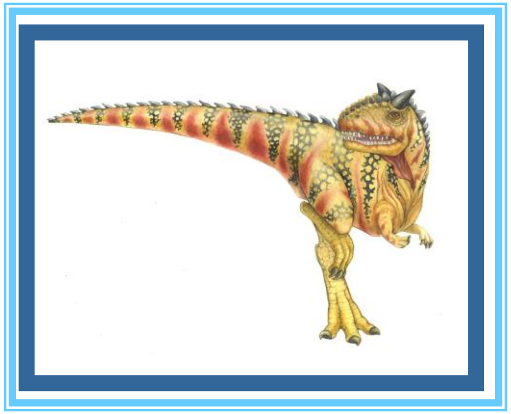
# Introduction to Operating System
# Day1: March 2022

**Kiran Waghmare**



# Learning and understanding

Top down

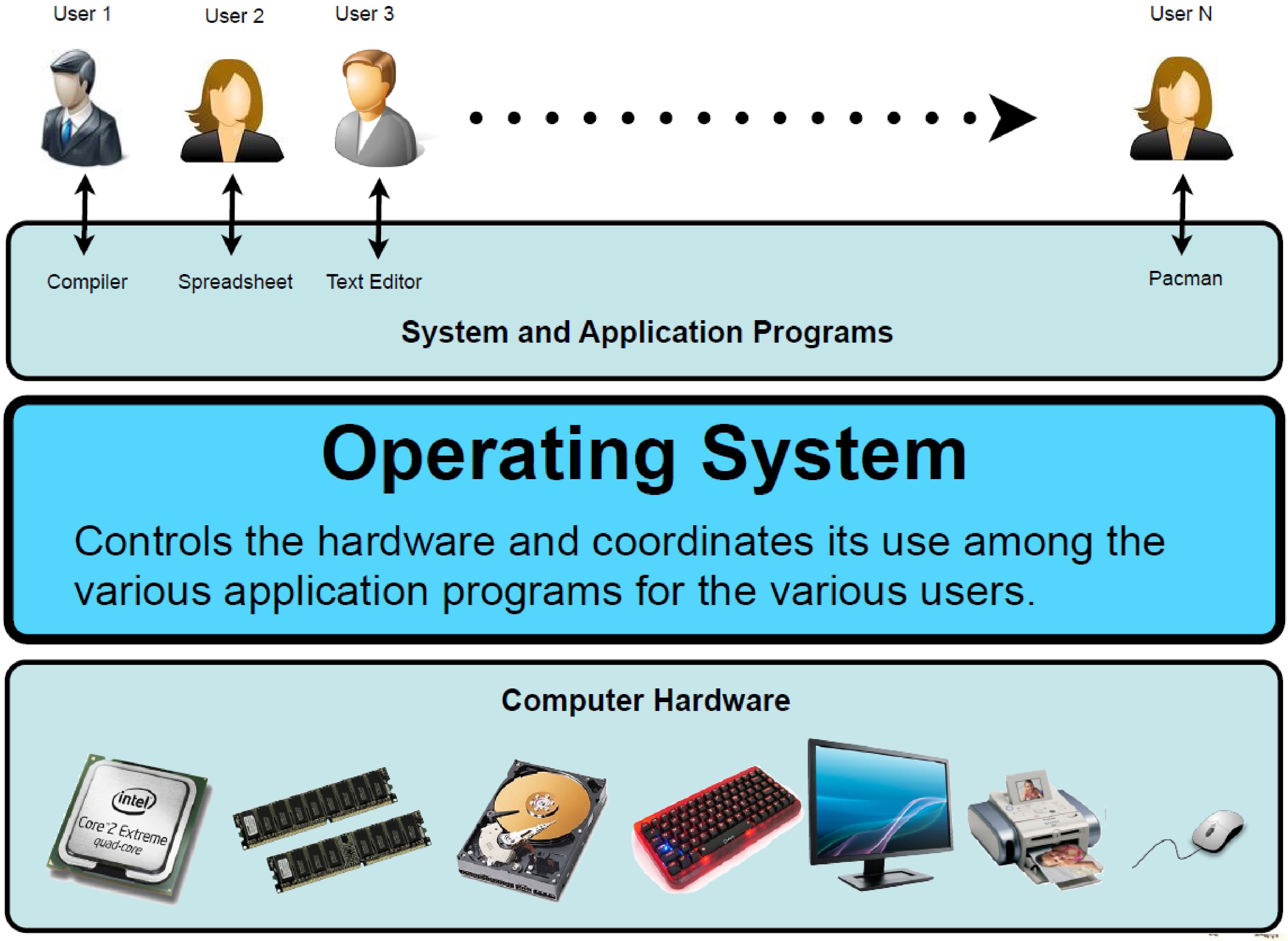
Bottom up

# Agenda

- **Introduction to OS**
  - OS
  - Application Software
  - Hardware dependent
  - Components of OS
  - Difference between :
    - Mobile OS, Embedded system OS,
    - Real Time OS,
    - desktop OS server machine os
  - Functions of OS
  - User and Kernel space & model
  - Interrupts & system calls

# 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**
  - Make the **computer system convenient** to use
  - Use the computer hardware in an efficient manner



User 1

User 2

User 3

User N

Compiler

Spreadsheet

Text Editor

Pacman

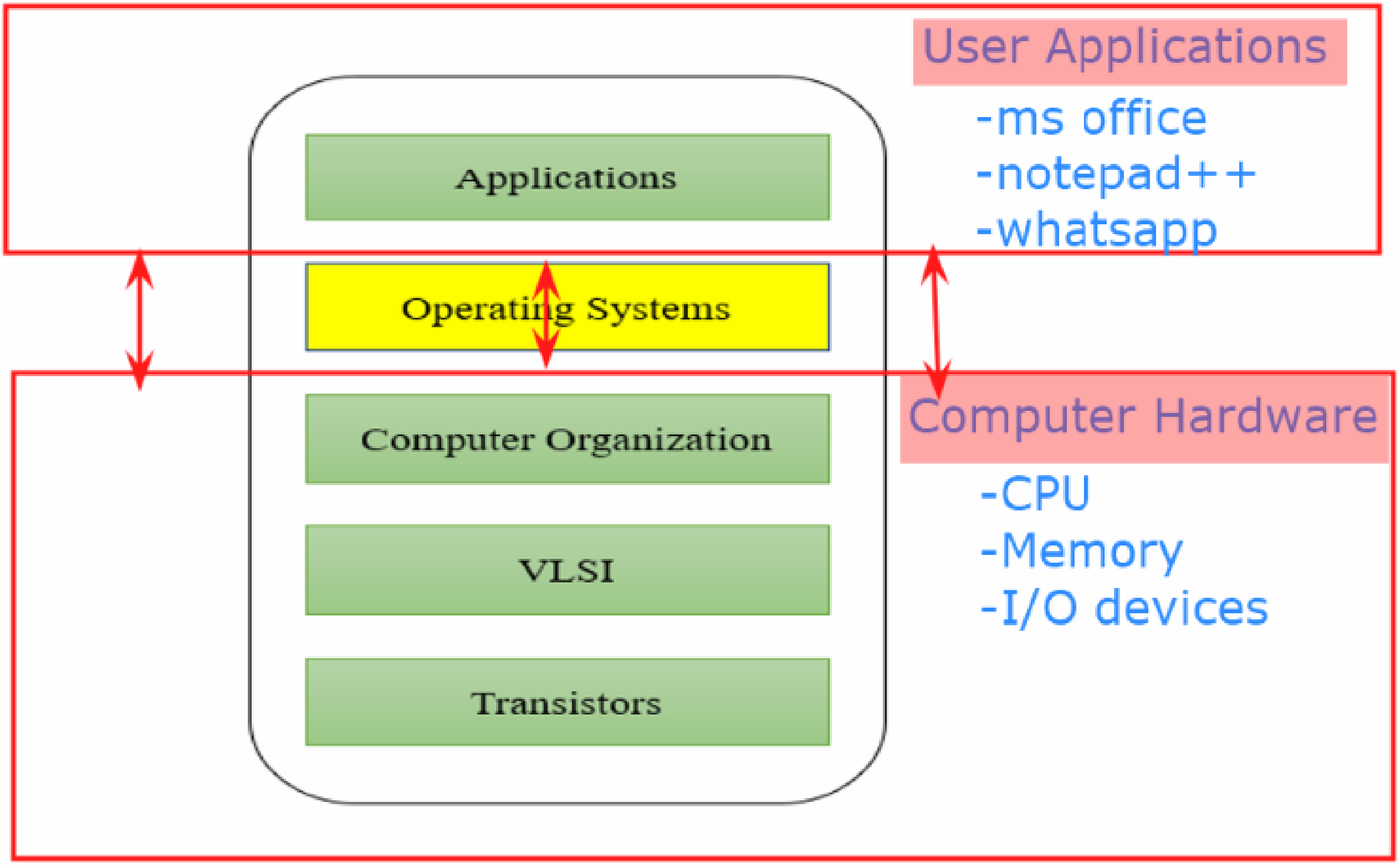
**System and Application Programs**

# Operating System

Controls the hardware and coordinates its use among the various application programs for the various users.

**Computer Hardware**

# The Layers in Systems



User Applications

-ms office
-notepad++
-whatsapp

Computer Hardware

-CPU
-Memory
-I/O devices

1. Hardware Abstraction
2. Resource Management

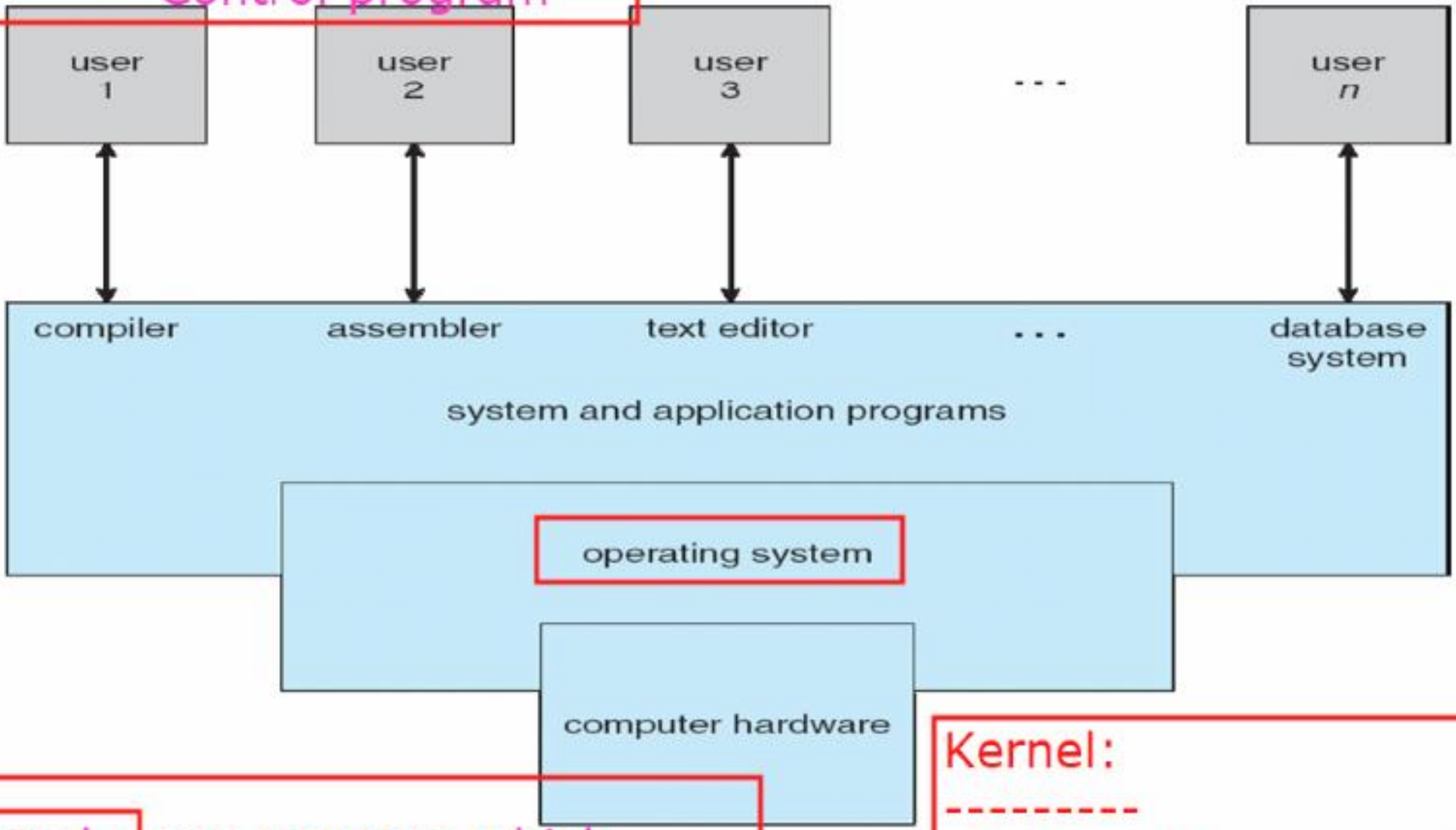
# Computer System Structure

- 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 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

You are screen sh Who can see what you share here? R

# Four Components of a Computer System

OS = Resource allocator
= Control program

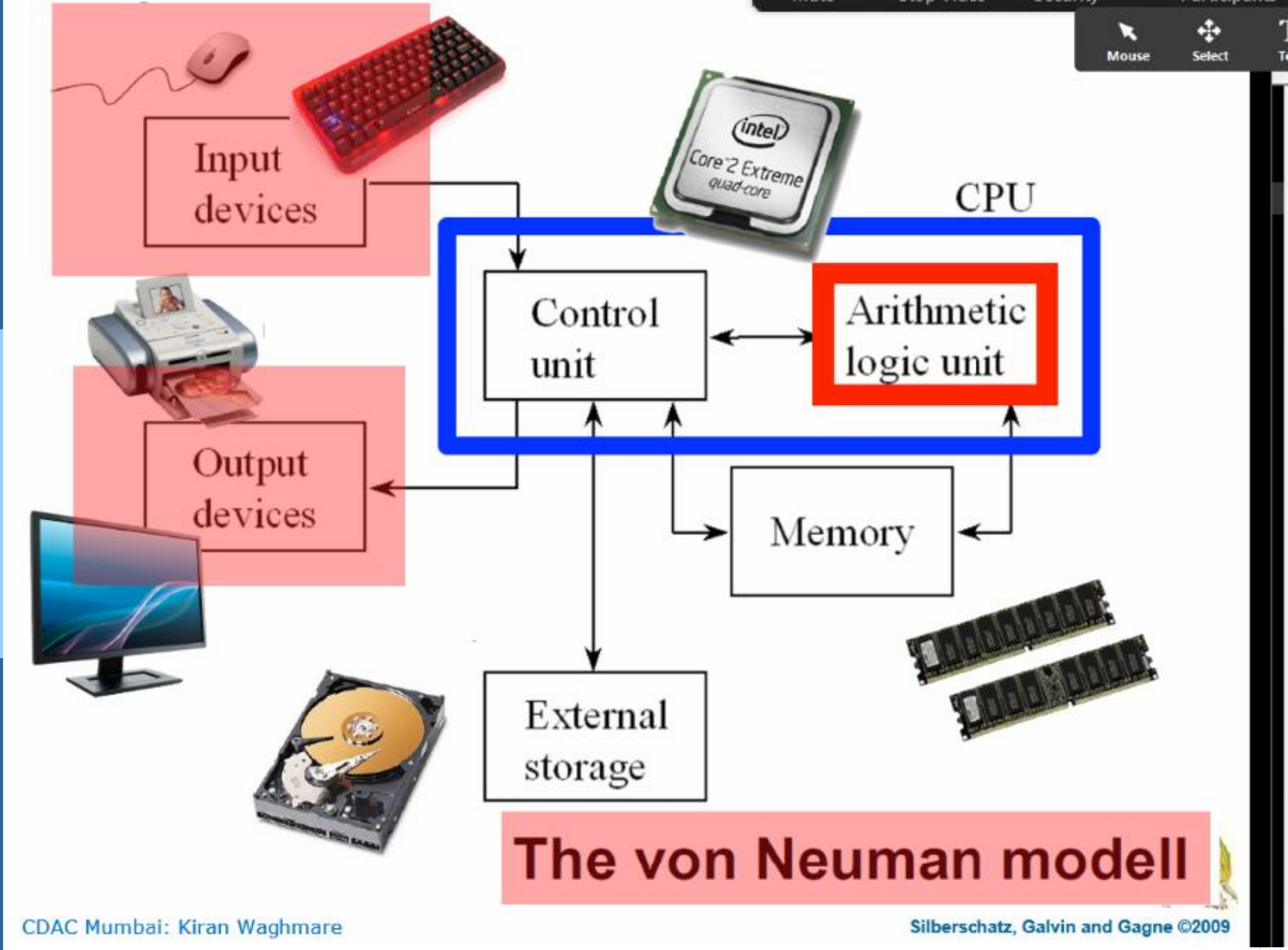


Kernel : one program which runs all the time in computer

Kernel:
---------
-System Programs
-Application Programs

# System Boot

- Operating system must be made available to hardware so hardware can start it
  - Small piece of code – **bootstrap loader**, locates the kernel, loads it into memory, and starts it
  - Sometimes two-step process where **boot block** at fixed location loads bootstrap loader
  - When power initialized on system, execution starts at a fixed memory location
    - Firmware used to hold initial boot code

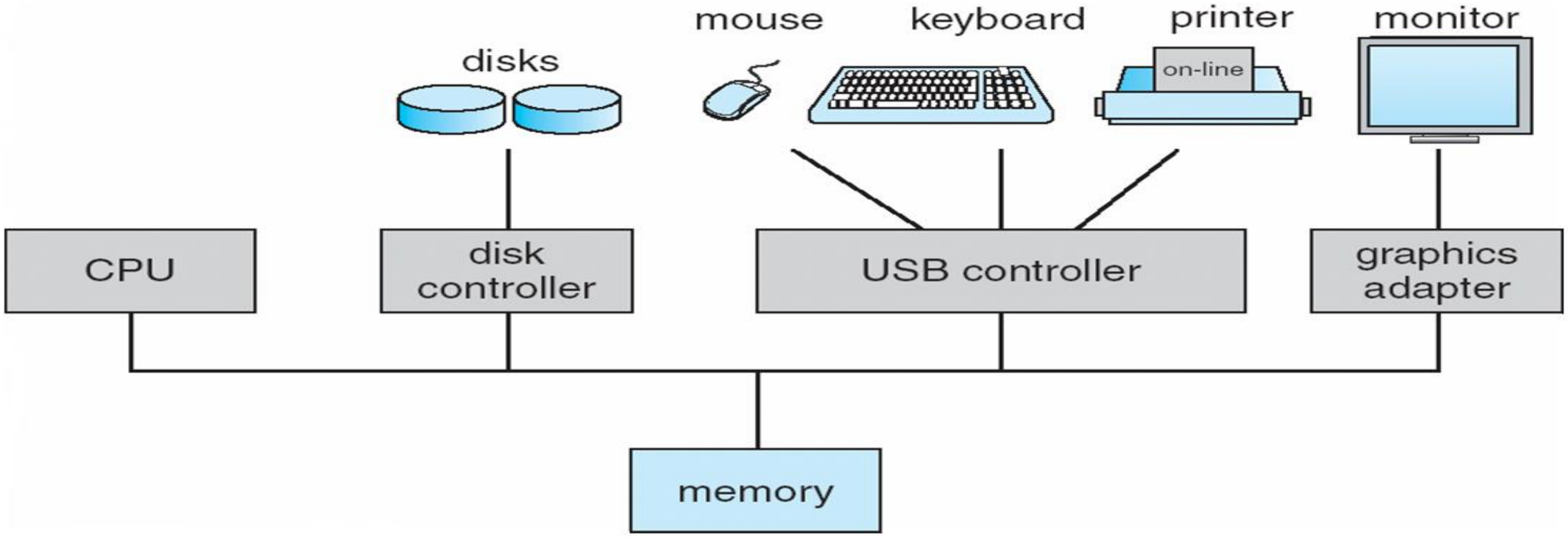


# The von Neuman modell

CDAC Mumbai: Kiran Waghmare
Silberschatz, Galvin and Gagne ©2009

# 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



## System and Application Programs

## Operating System

Controls the hardware and coordinates its use among the various application programs for the various users.

### Bootstrap program

### Kernel

## Computer Hardware

# Operating System Management Tasks

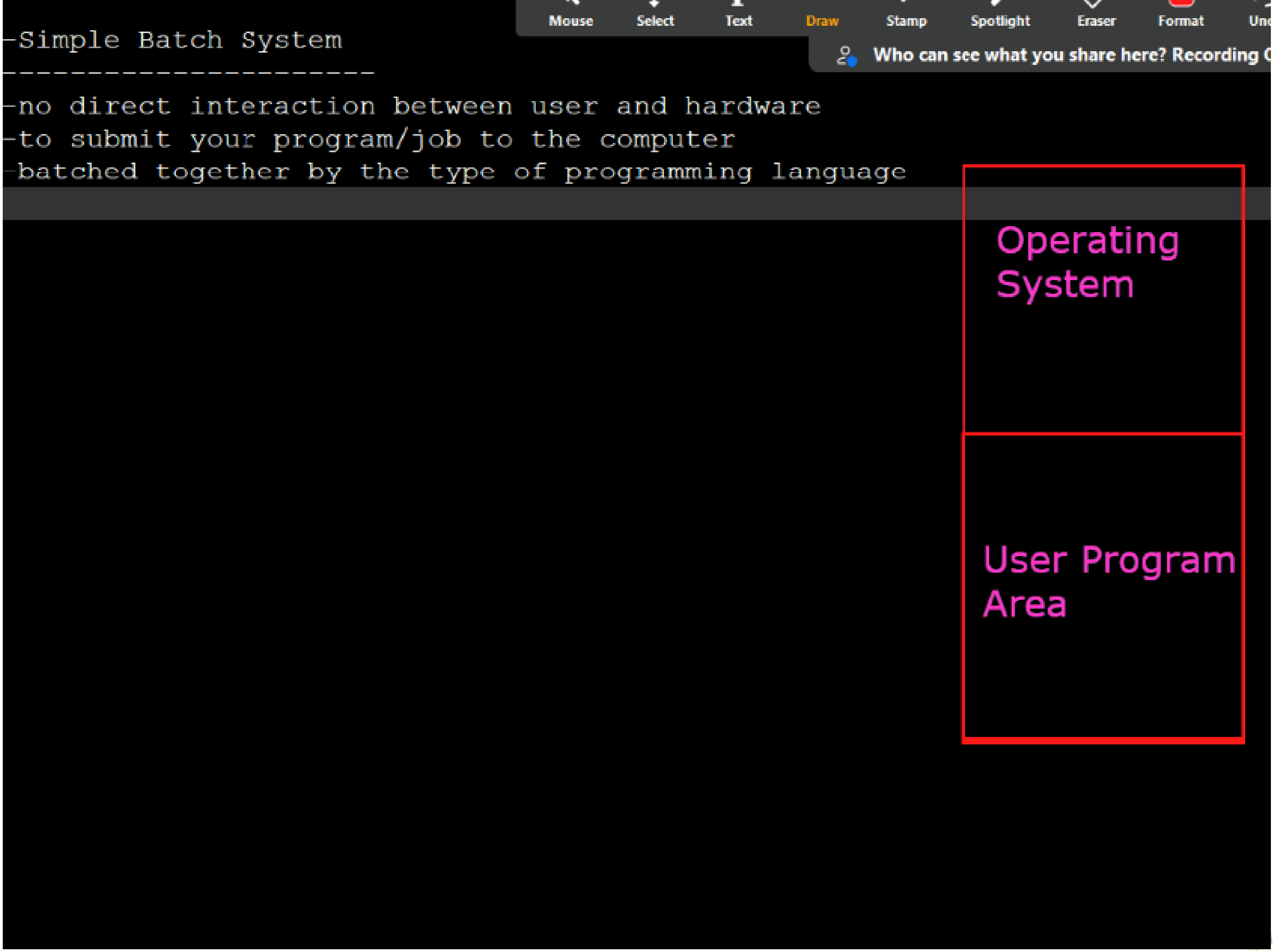
1. **Process management** which involves putting the tasks into order and pairing them into manageable size before they go to the CPU.
2. **Memory management** which coordinates data to and from RAM (random-access memory) and determines the necessity for virtual memory.
3. **Device management** provides an interface between connected devices.
4. **Storage management** which directs permanent data storage.
5. **An application** that allows standard communication between software and your computer.
6. **The user interface** allows you to communicate with your computer.

# Examples of Operating System

- Windows
- Android
- iOS
- Mac OS
- Linux
- Window Phone OS
- Chrome OS

# Types of Operating Systems

- Following are some of the most widely used types of Operating system.
  - Simple Batch System
  - Multiprogramming Batch System
  - Multiprocessor System
  - Desktop System
  - Distributed Operating System
  - Clustered System
  - Realtime Operating System
  - Handheld System

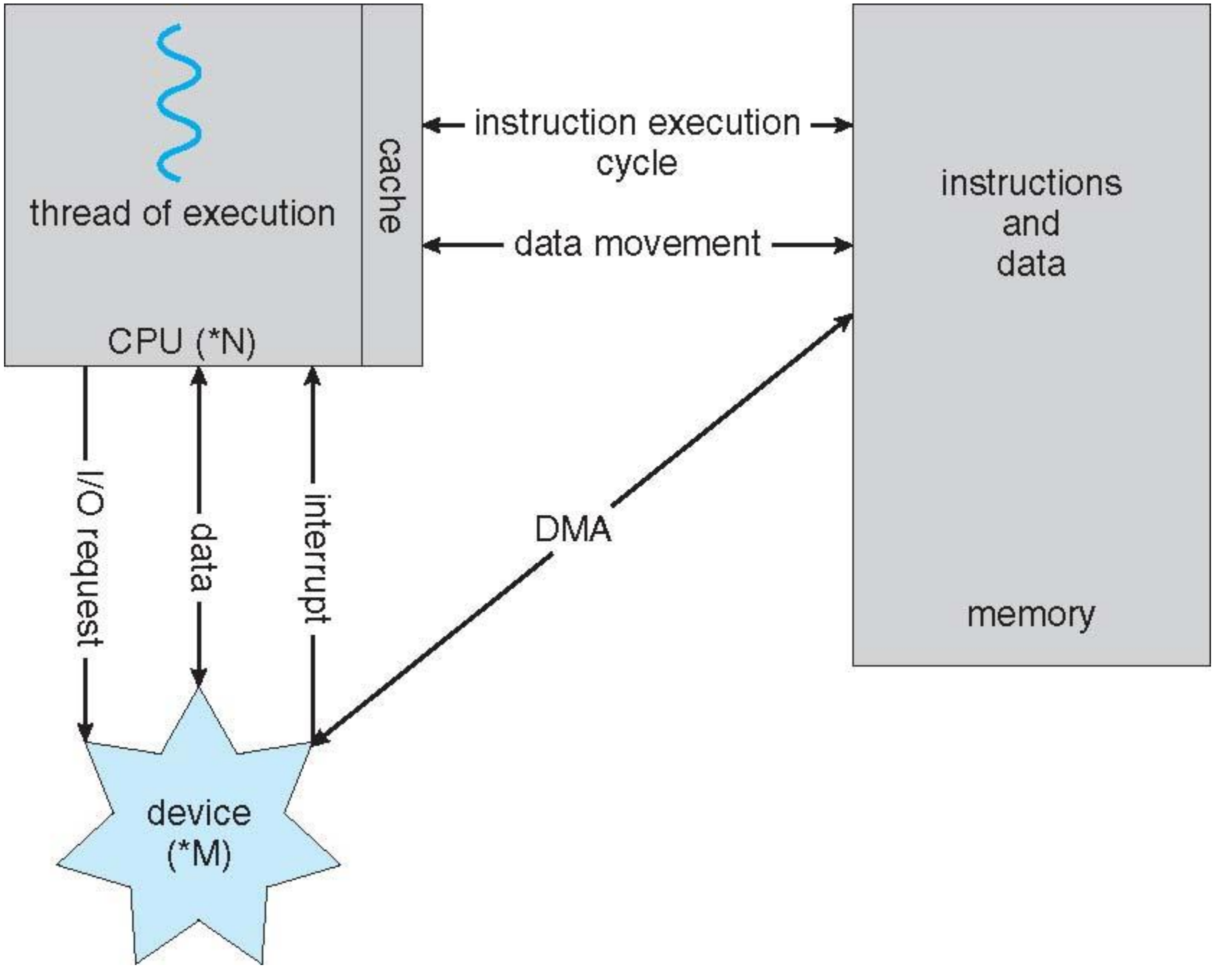


```
-Simple Batch System
-----------------------
-no direct interaction between user and hardware
-to submit your program/job to the computer
-batched together by the type of programming language
```

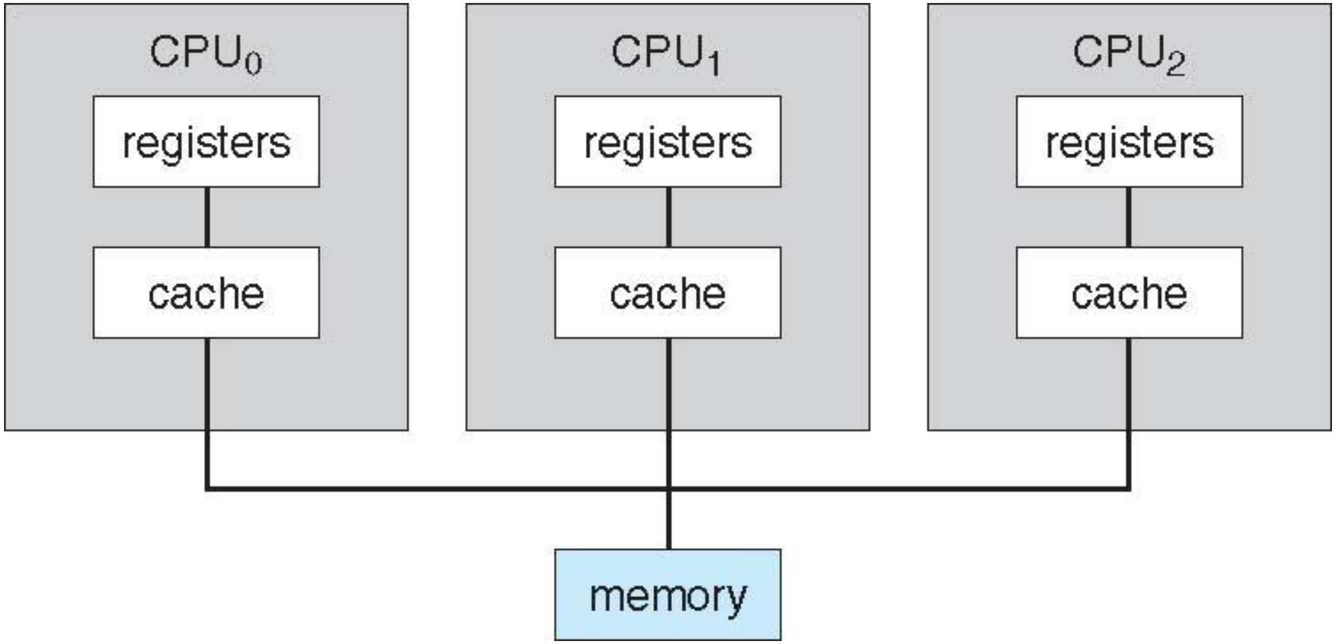
# Computer-System Architecture

- Most systems use a single general-purpose processor (PDAs through mainframes)
  - Most systems have special-purpose processors as well
- Multiprocessors systems growing in use and importance
  - Also known as parallel systems, tightly-coupled systems
  - Advantages include
    1. Increased throughput
    2. Economy of scale
    3. Increased reliability – graceful degradation or fault tolerance
  - Two types
    1. Asymmetric Multiprocessing
    2. Symmetric Multiprocessing

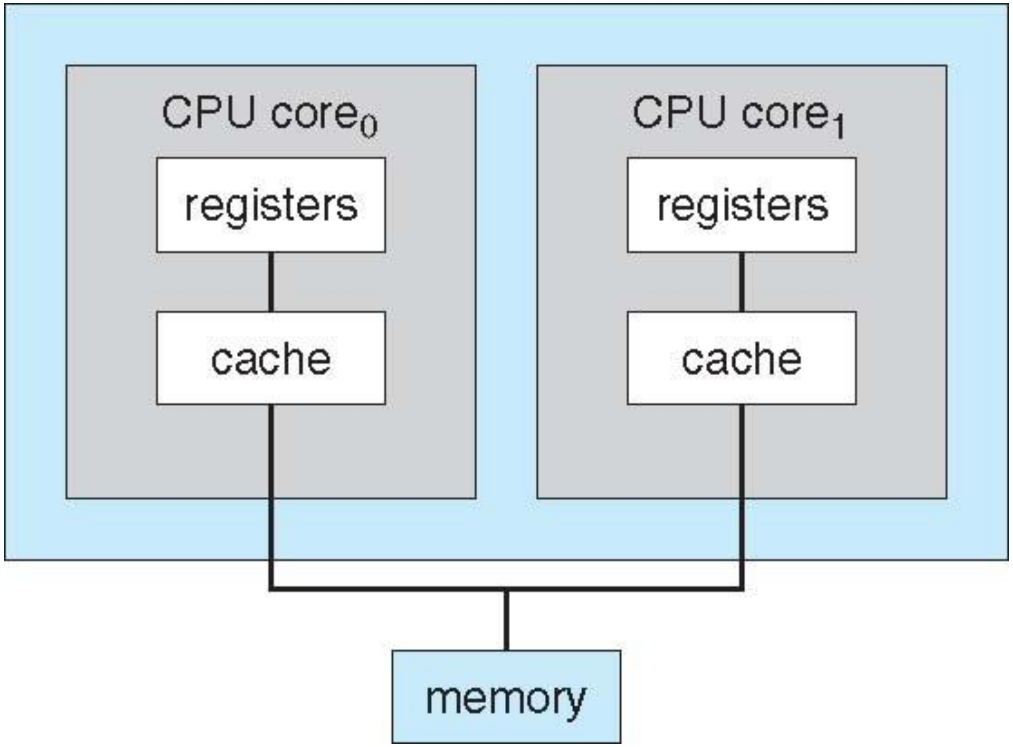
# How a Modern Computer Works



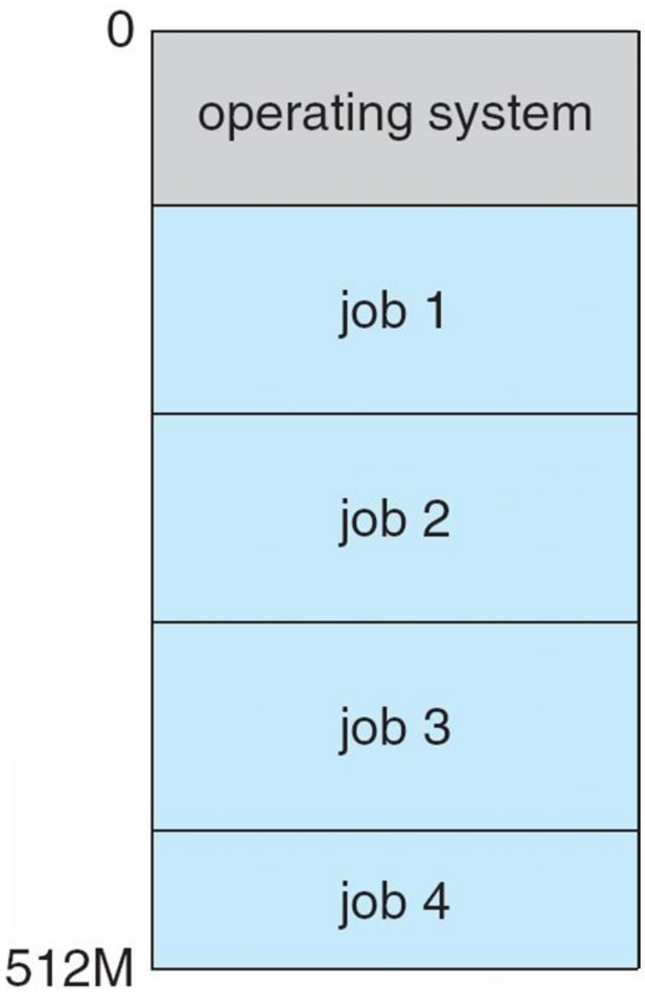
# Symmetric Multiprocessing Architecture



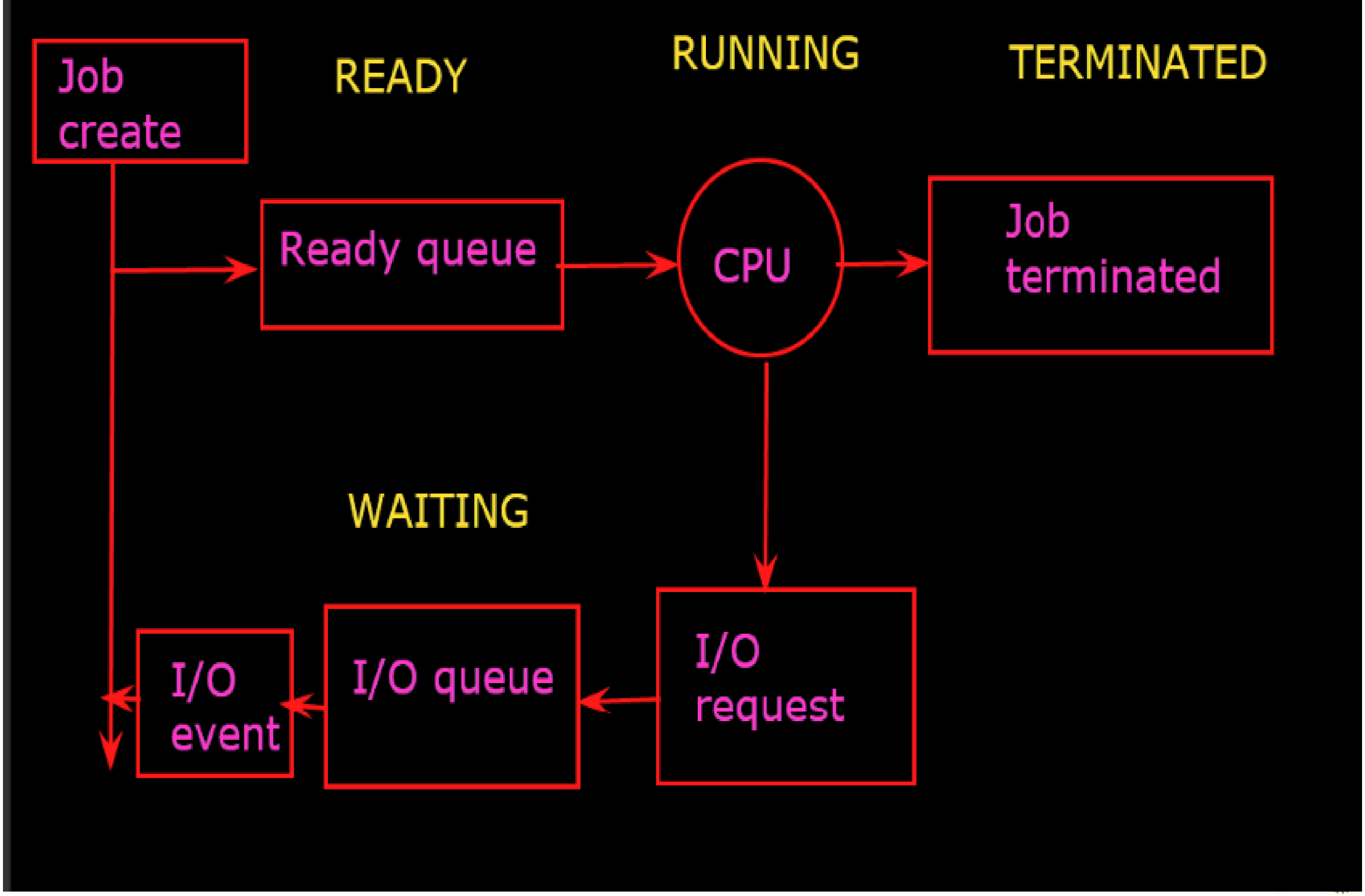
# A Dual-Core Design



# Memory Layout for Multiprogrammed System



## Multiprogramming Batch System

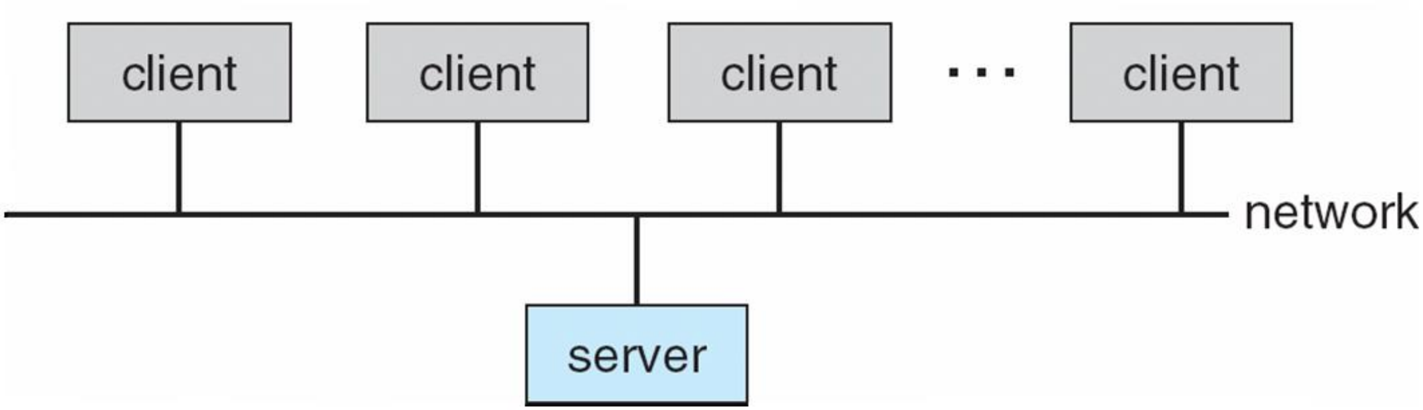


# Operating-System Operations

- Interrupt driven by hardware
- Software error or request creates **exception** or **trap**
  - Division by zero, request for operating system service
- Other process problems include infinite loop, processes modifying each other or the operating system
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user

# Computing Environments (Cont)

- Client-Server Computing
  - Dumb terminals supplanted by smart PCs
  - Many systems now **servers**, responding to requests generated by **clients**
    - **Compute-server** provides an interface to client to request services (i.e. database)
    - **File-server** provides interface for clients to store and retrieve files

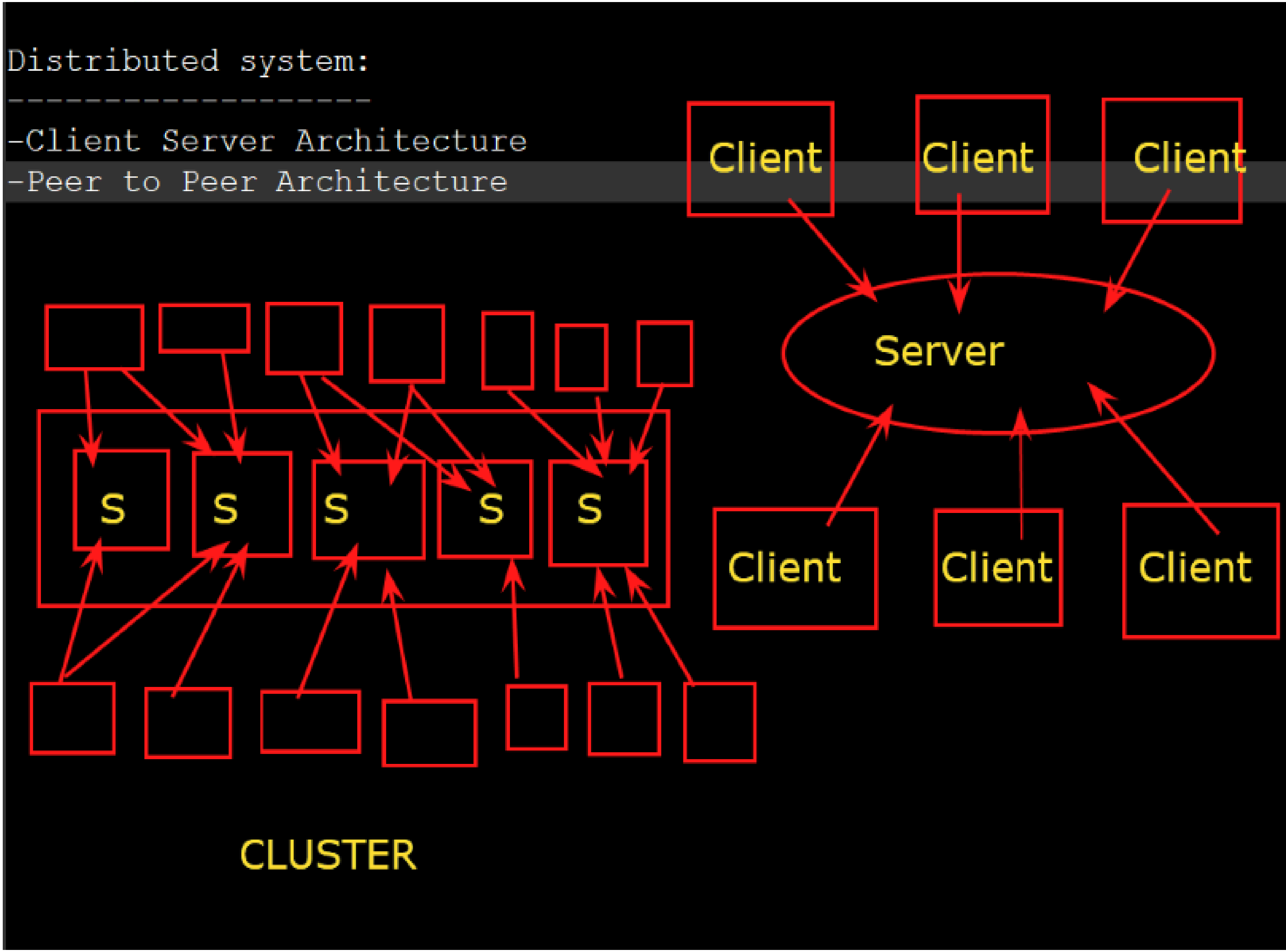


# Types of Distributed Operating Systems

- Following are the two types of distributed operating systems used:
  - Client-Server Systems
  - Peer-to-Peer Systems
- **Client-Server Systems**
- Centralized systems today act as server systems to satisfy requests generated by client systems. The general structure of a client-server system is depicted in the figure below:
- Server Systems can be broadly categorized as: Compute Servers and File Servers.
- Compute Server systems, provide an interface to which clients can send requests to perform an action, in response to which they execute the action and send back results to the client.
- File Server systems, provide a file-system interface where clients can create, update, read, and delete files.

# Peer-to-Peer Computing

- Another model of distributed system
- P2P does not distinguish clients and servers
  - Instead all nodes are considered peers
  - May each act as client, server or both
  - Node must join P2P network
    - Registers its service with central lookup service on network, or
    - Broadcast request for service and respond to requests for service via **discovery protocol**
  - Examples include *Napster* and *Gnutella*



```
Distributed system:
--------------------
-Client Server Architecture
-Peer to Peer Architecture
```

# Web-Based Computing

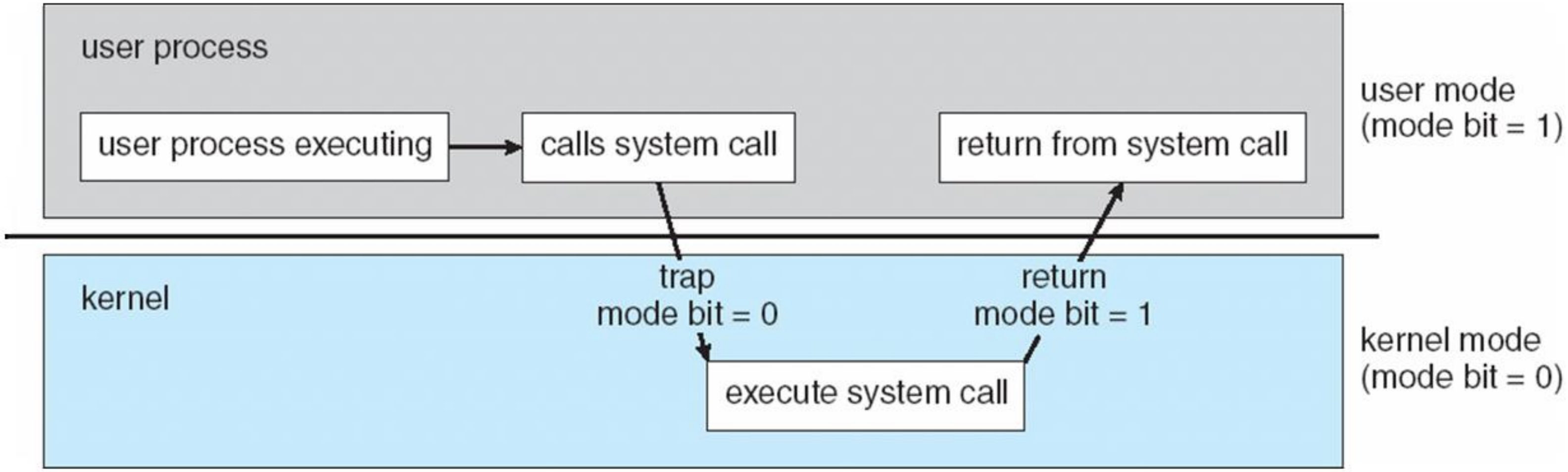
- Web has become ubiquitous
- PCs most prevalent devices
- More devices becoming networked to allow web access
- New category of devices to manage web traffic among similar servers: **load balancers**
- Use of operating systems like Windows 95, client-side, have evolved into Linux and Windows XP, which can be clients and servers

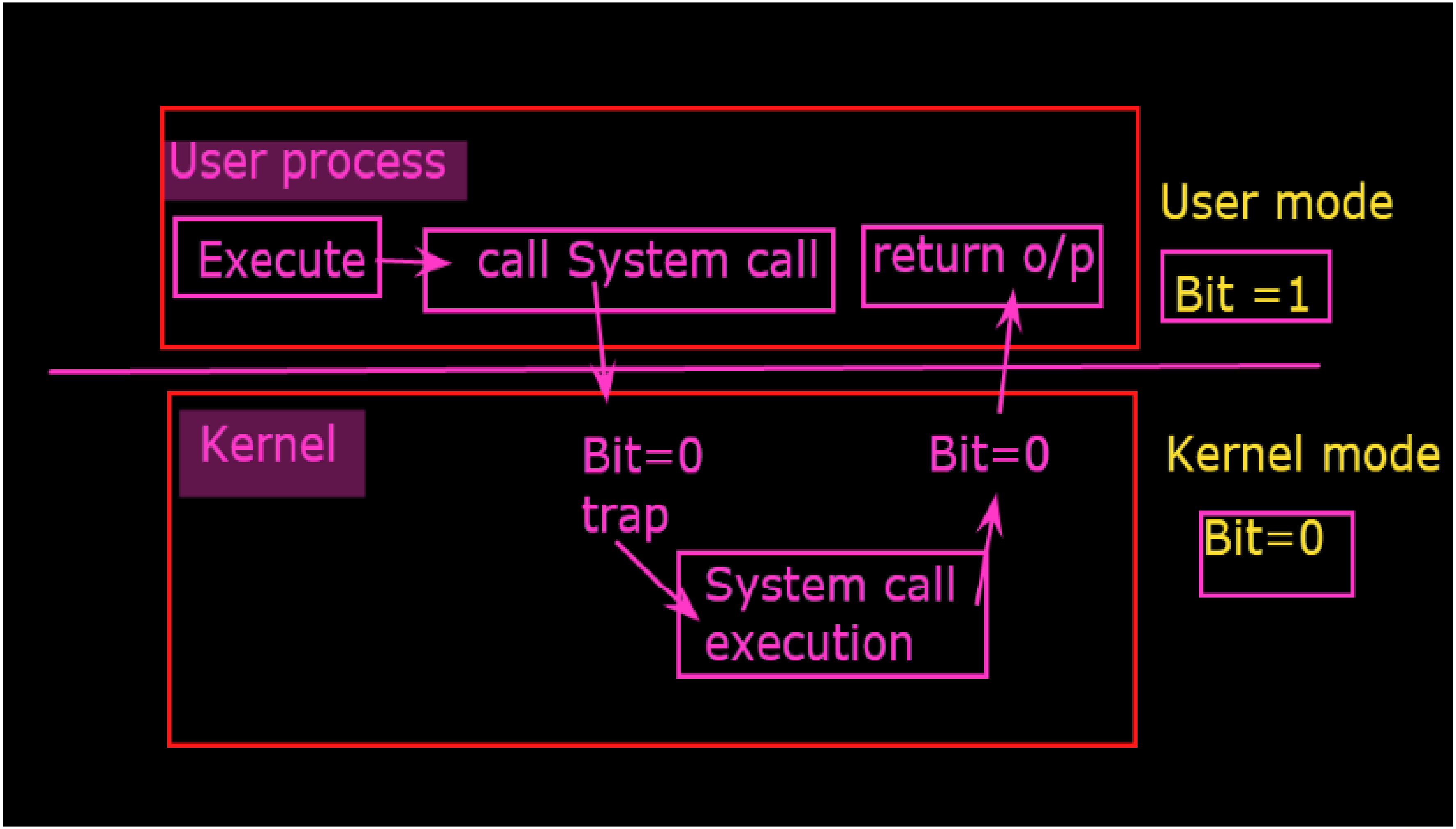
# Clustered Systems

- Like multiprocessor systems, but multiple systems working together
  - Usually sharing storage via a storage-area network (SAN)
  - Provides a high-availability service which survives failures
    - Asymmetric clustering has one machine in hot-standby mode
    - Symmetric clustering has multiple nodes running applications, monitoring each other
  - Some clusters are for high-performance computing (HPC)
    - Applications must be written to use parallelization

# Transition from User to Kernel Mode

- Timer to prevent infinite loop / process hogging resources
  - Set interrupt after specific period
  - Operating system decrements counter
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time

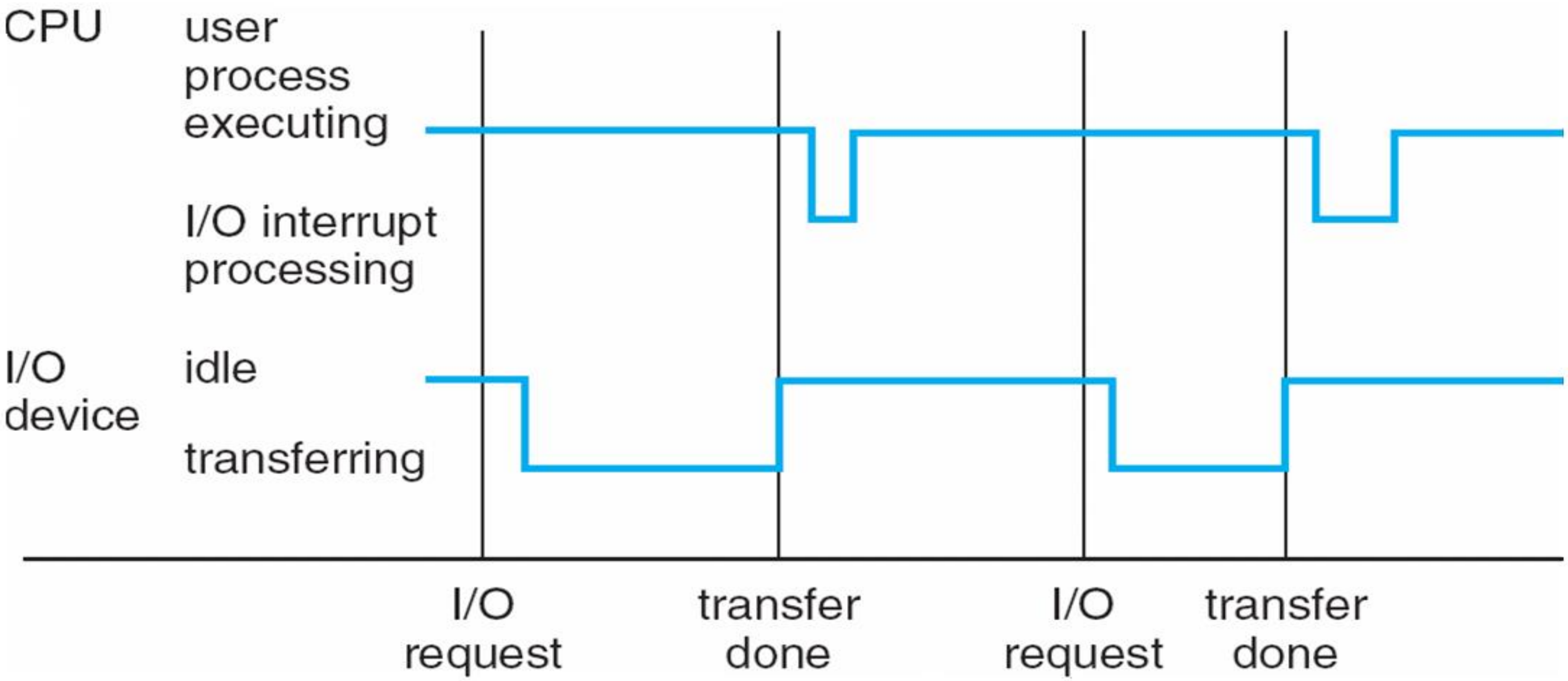


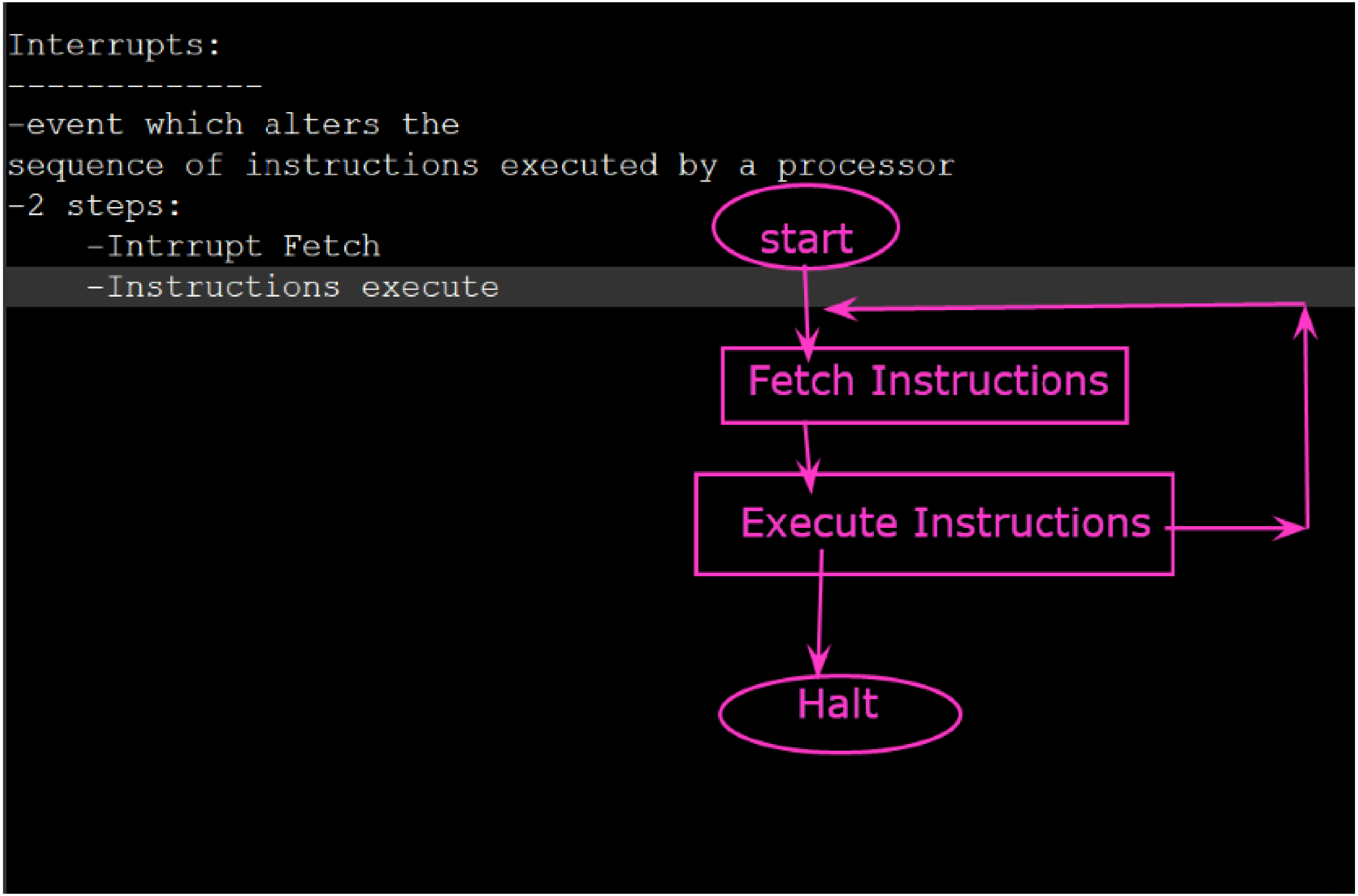

User process
Execute
call System call
return o/p
User mode
Bit =1
Kernel
Bit=0
trap
Bit=0
System call execution
Kernel mode
Bit=0

# Interrupt Handling

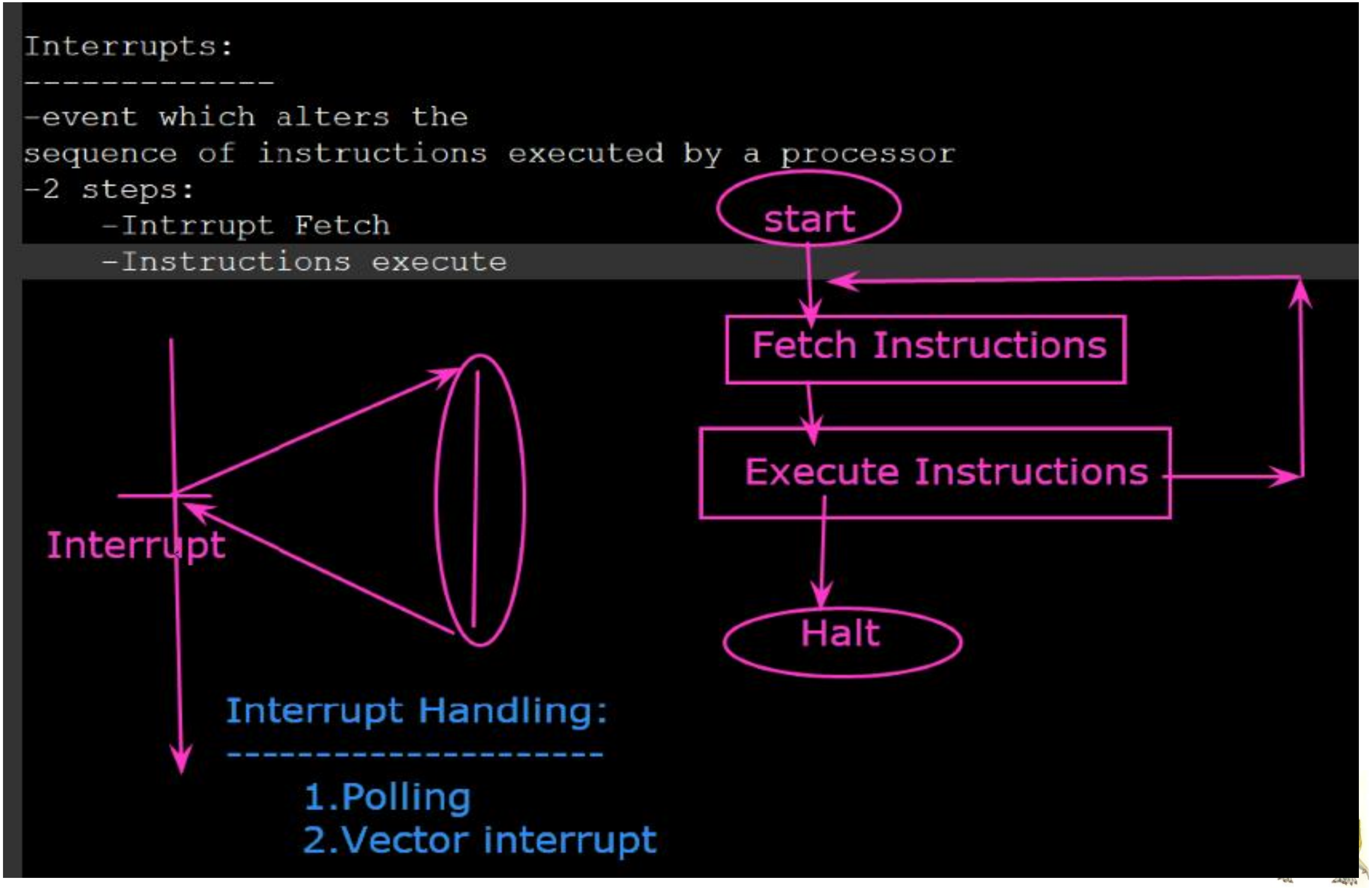
- The operating system preserves the state of the CPU by storing registers and the program counter
- Determines which type of interrupt has occurred:
  - **polling**
  - **vectored** interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt

# Interrupt Timeline





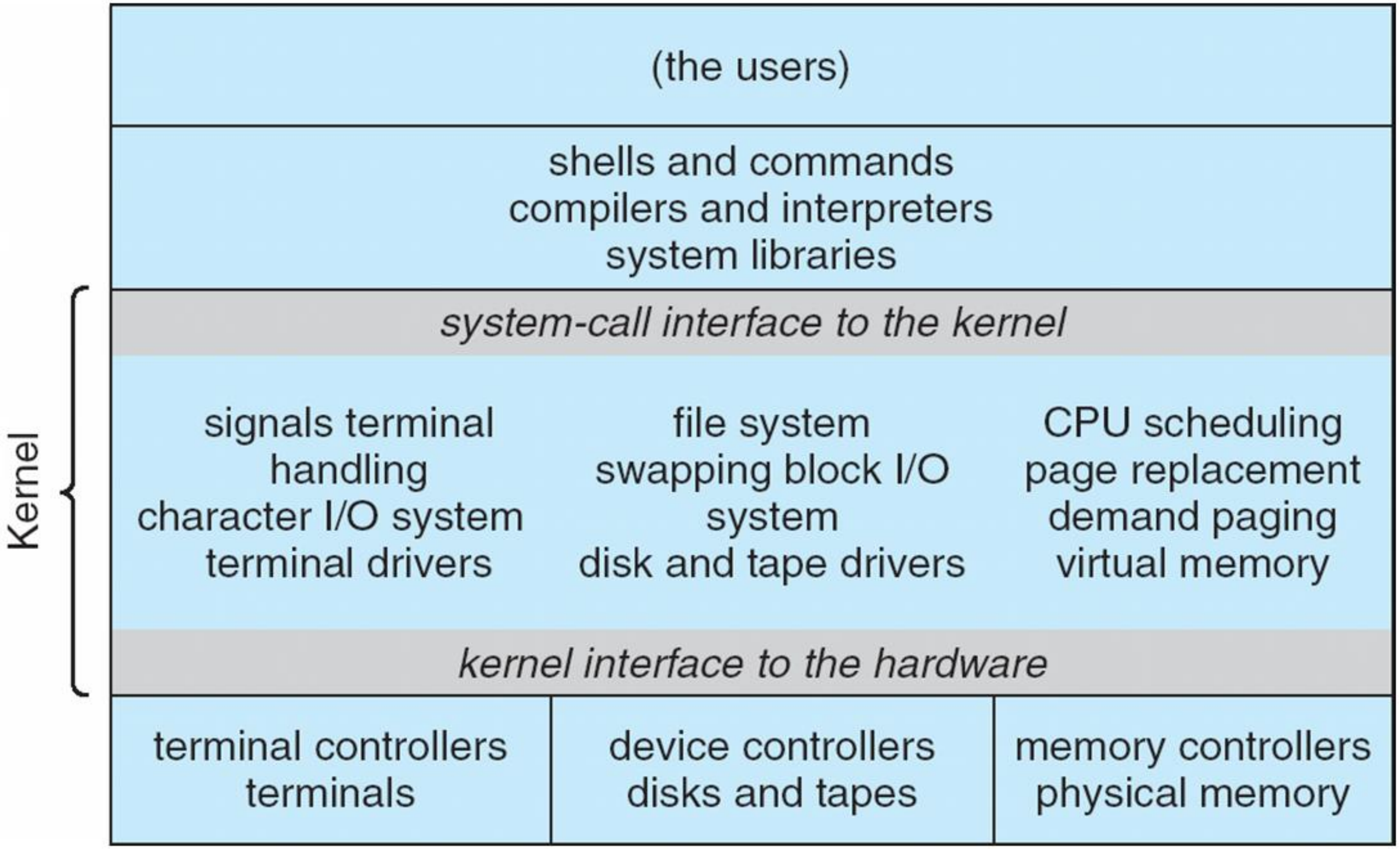
```
Interrupts:
-------------
-event which alters the
sequence of instructions executed by a processor
-2 steps:
    -Intrrupt Fetch
    -Instructions execute
```



```
Interrupts:
-------------
-event which alters the
sequence of instructions executed by a processor
-2 steps:
    -Intrrupt Fetch
    -Instructions execute
```

Interrupt Handling:
---------------------
1.Polling
2.Vector interrupt

# Traditional UNIX System Structure



(the users)

shells and commands
compilers and interpreters
system libraries

*system-call interface to the kernel*

signals terminal handling
character I/O system
terminal drivers

file system
swapping block I/O system
disk and tape drivers

CPU scheduling
page replacement
demand paging
virtual memory

*kernel interface to the hardware*

terminal controllers
terminals

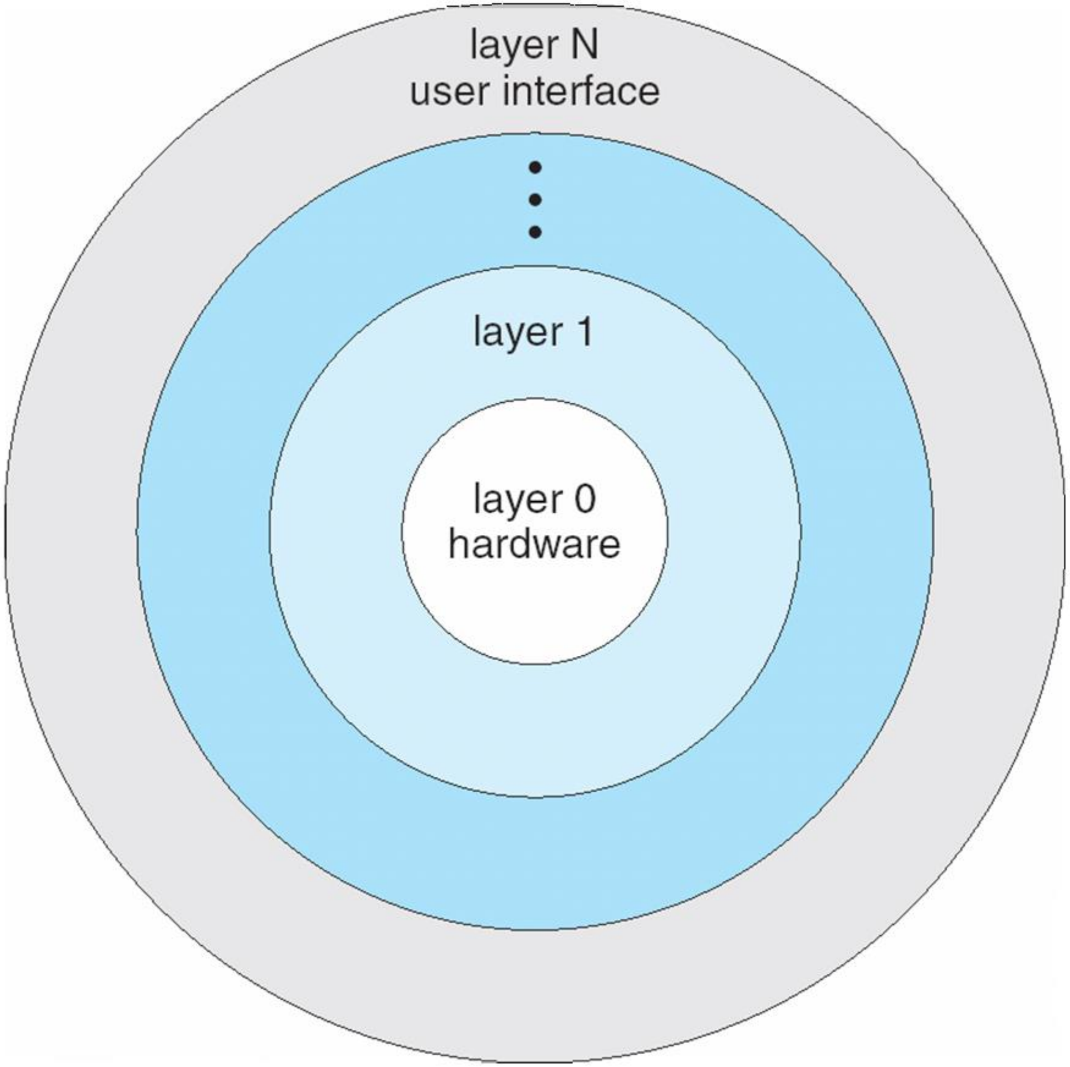
device controllers
disks and tapes

memory controllers
physical memory

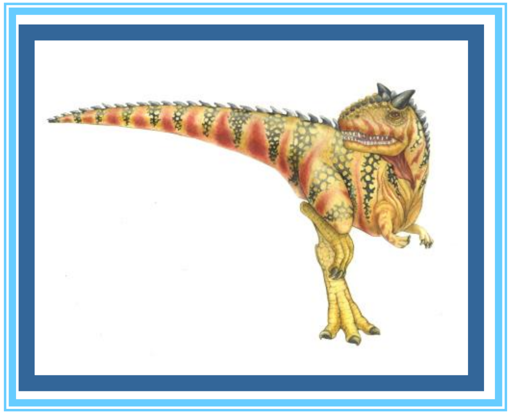
# UNIX

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts
  - Systems 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

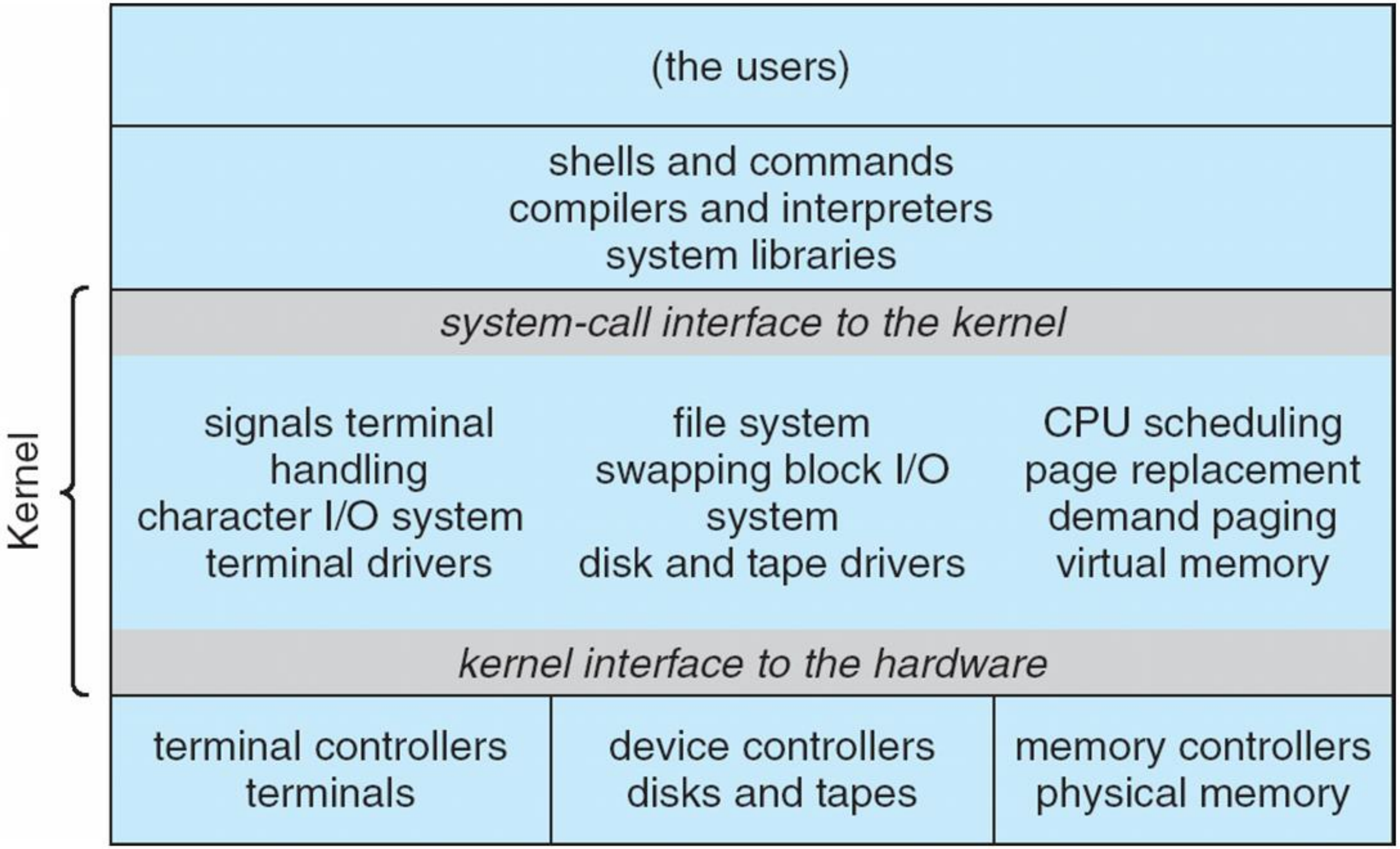
# Layered Operating System



# Introduction to Linux



# Traditional UNIX System Structure



(the users)

shells and commands
compilers and interpreters
system libraries

*system-call interface to the kernel*

signals terminal handling
character I/O system
terminal drivers

file system
swapping block I/O system
disk and tape drivers

CPU scheduling
page replacement
demand paging
virtual memory

*kernel interface to the hardware*

terminal controllers
terminals

device controllers
disks and tapes

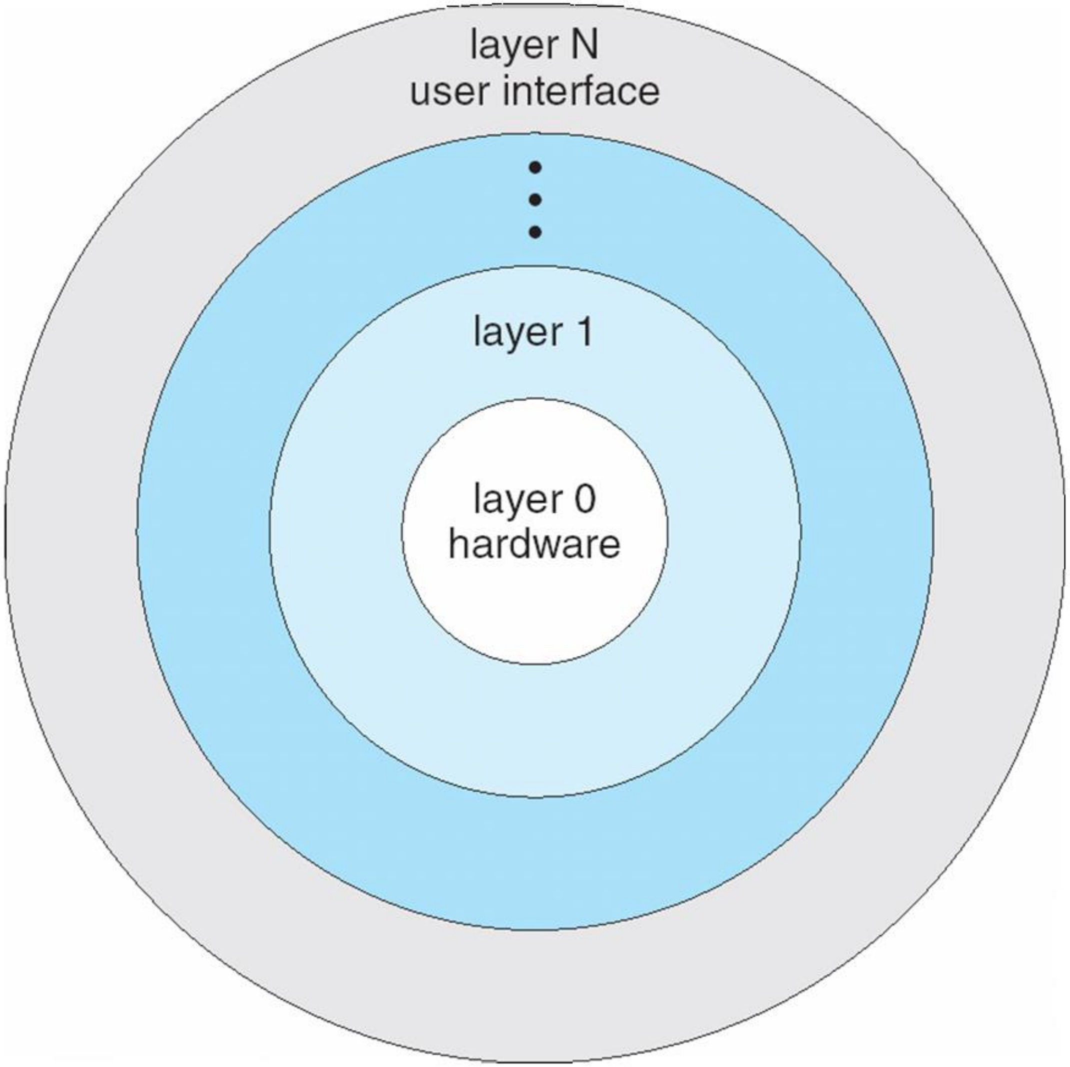
memory controllers
physical memory

Kernel

# UNIX

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring.  The UNIX OS consists of two separable parts
  - Systems 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

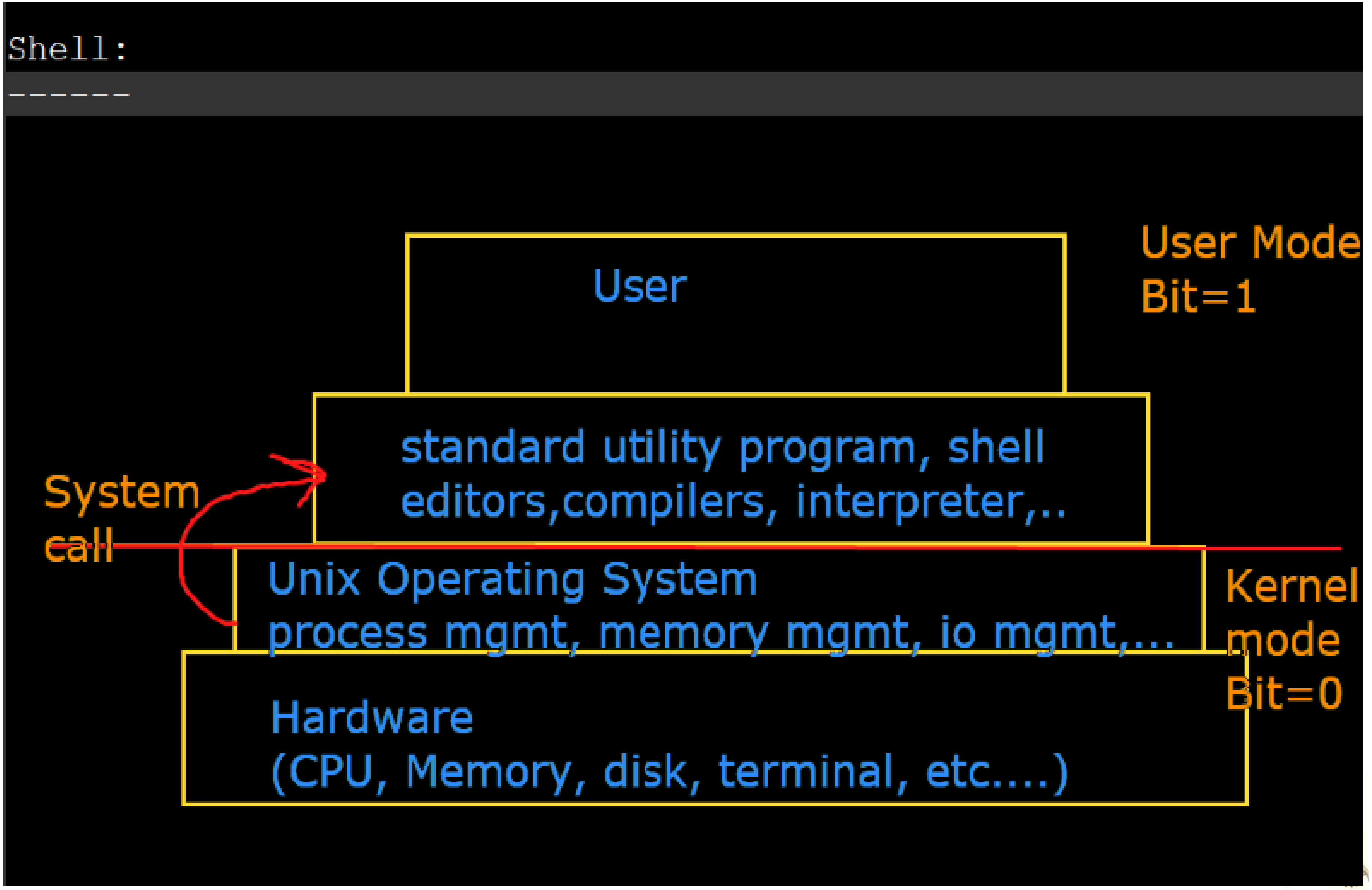
# Layered Operating System



# Linux Bash

- The Linux Bash is also known as **'Bourne-again Shell**.’
- It is a **command language interpreter** for the Linux based system.
- It is a replacement of Bourne shell (sh).
- It was developed under the GNU Project and written by **Brian Fox**.
- Nowadays, Bash is the default user shell of most of the Linux distributions.
- The Bash is a **command language interpreter** as well as a **programming language**.
- It supports **variables, functions, and flow control**, like other programming languages.
- It can also read and execute the commands from a file, which is called a **shell script.**

# Shell



```
Shell:
------
```

User
standard utility program, shell
editors,compilers, interpreter,..
Unix Operating System
process mgmt, memory mgmt, io mgmt,...
Hardware
(CPU, Memory, disk, terminal, etc....)
User Mode
Bit=1
Kernel
mode
Bit=0
System
call