Operating System

Windows

Linux

Ubuntu

Mac OS X
iOS

Android

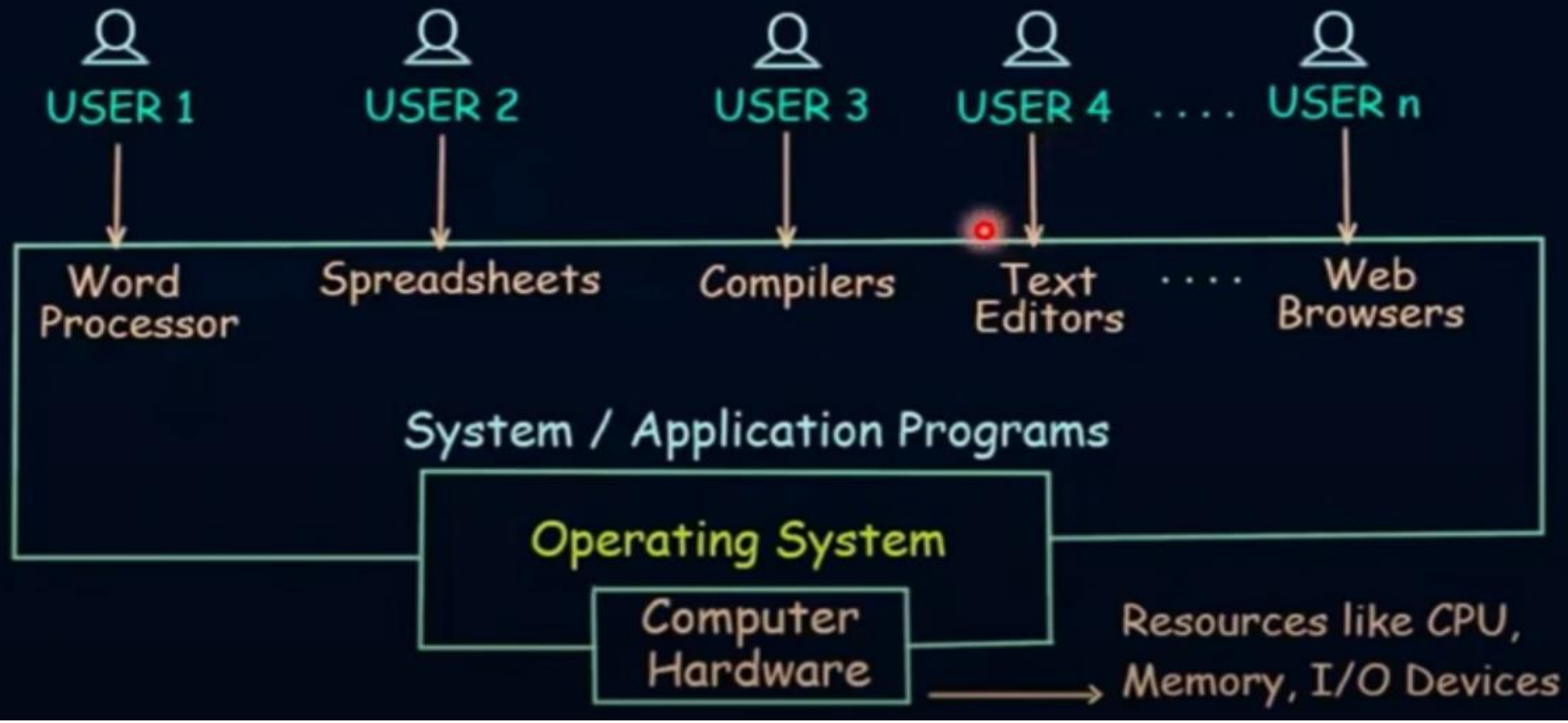
* An Operating System (OS) is a program that manages the computer hardware.

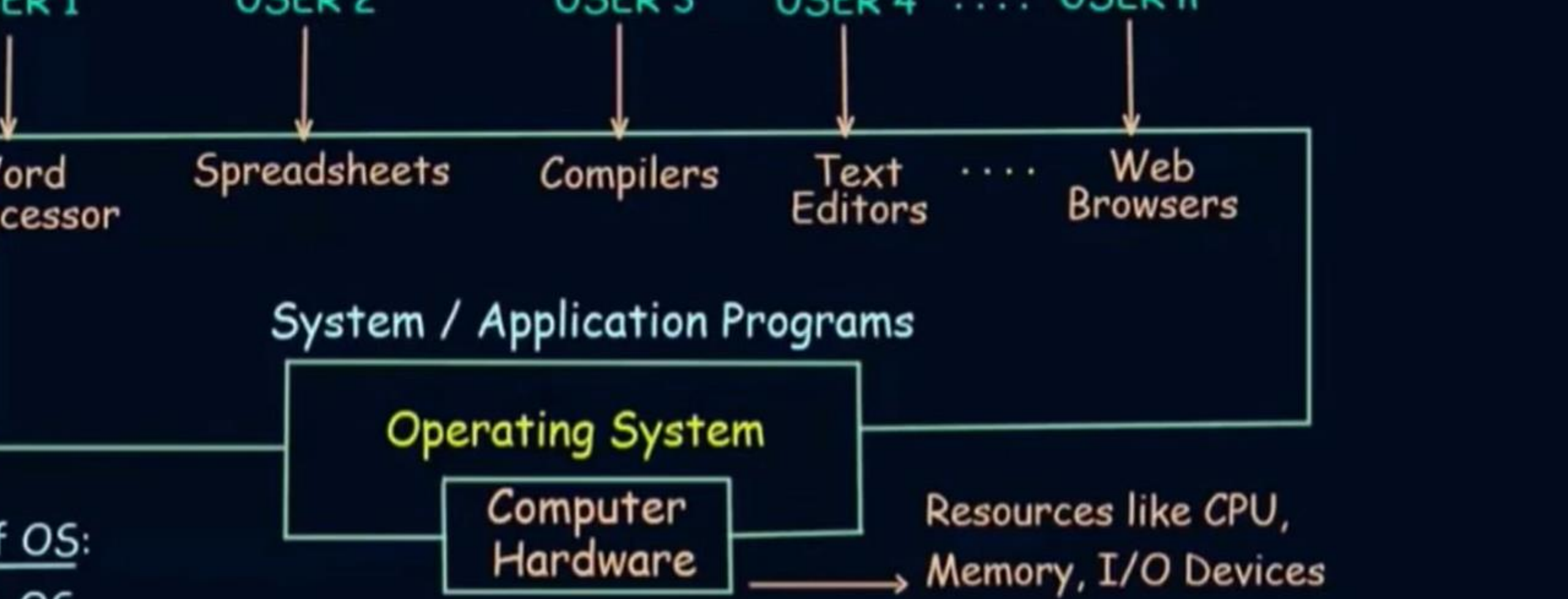
*It also provides a basis for Application Programs and acts as an intermediary between computer User and compter Hardware.

Windows Linux Ubuntu Mac OS X iOS Android

→ An Operating System (OS) is a program that manages the computer hardware.

→ It also provides a basis for Application Programs and acts as an intermediary between computer User and compter Hardware.





f OS:
OS
sharing OS
ibuted OS
ork OS
Time OS

Goals of OS:

i) Convenience
ii) Efficiency
iii) Both

Functions of OS:

→ It is an interface between User & Har
→ Allocation of Resources
→ Management of Memory, Security, etc

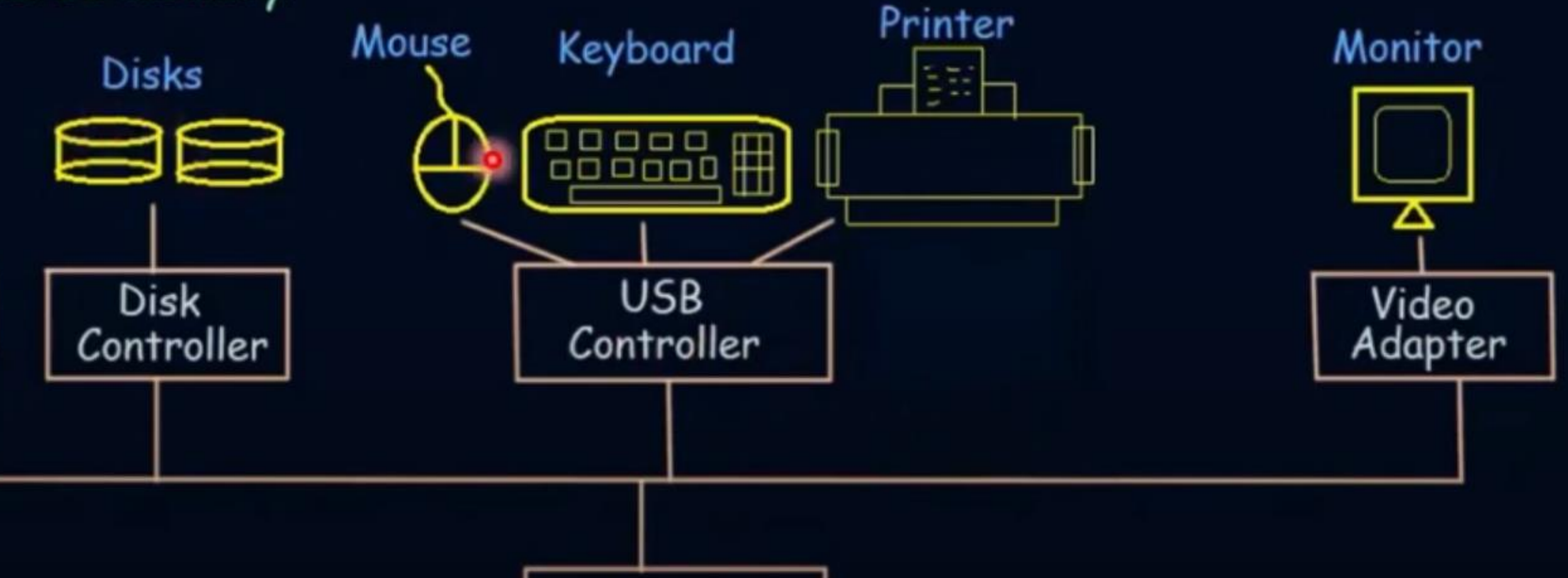
Basics of Operating System (Computer System Operation)

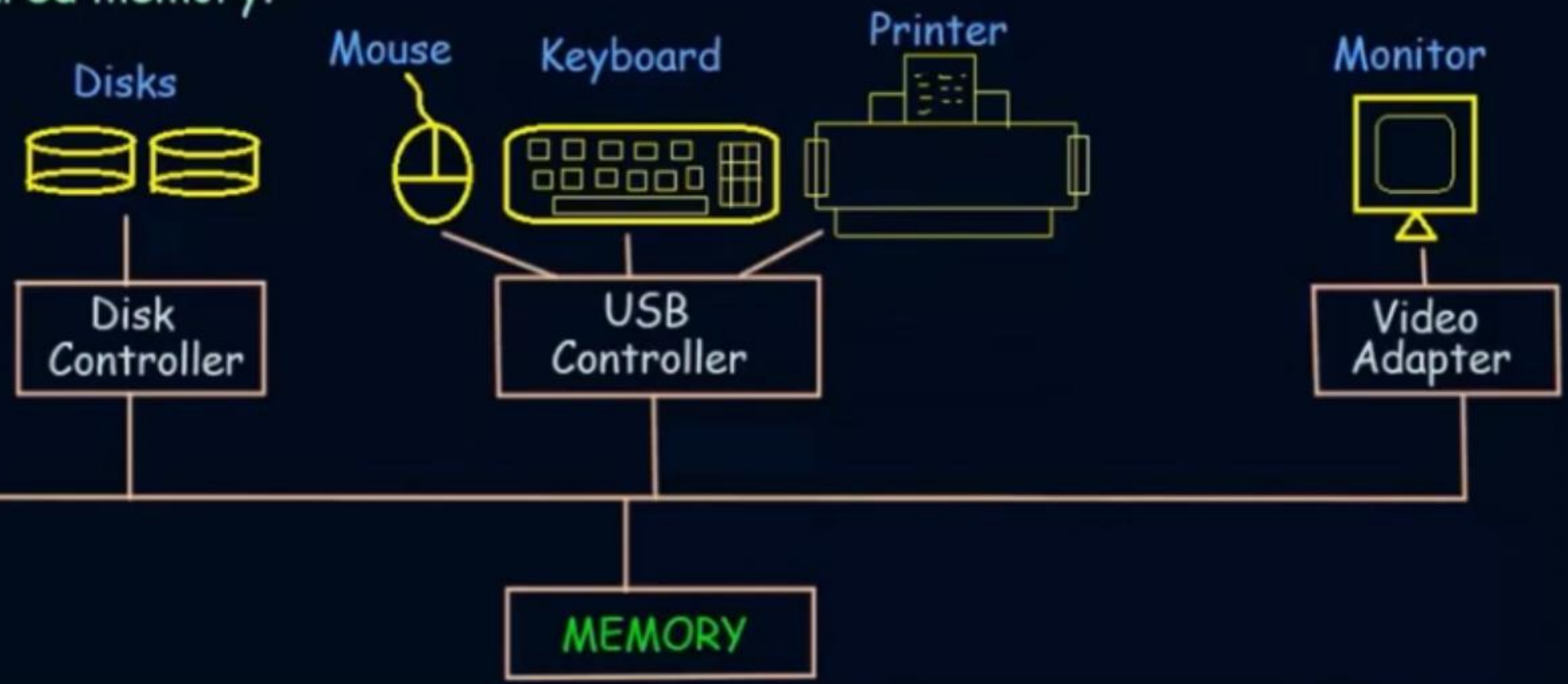
asic knowledge of the structure of Computer System is required to underst
erating Systems work.

dern general-purpose computer system consists of one or more CPUs and
er of device controllers connected through a common bus that provides acce
ed memory.

e basic knowledge of the structure of Computer System is required to unde
Operating Systems work.

modern general-purpose computer system consists of one or more CPUs and
nber of device controllers connected through a common bus that provides a
nared memory.





h device controller is in charge of a specific type of device

CPU and the device controllers can execute concurrently, competing for
nory cycles

ensure orderly access to the shared memory, a memory controller is provided w

portant terms:

trap Program:→The initial program that runs when a computer is powered up or rebooted.

→It is stored in the ROM.

→It must know how to load the OS and start executing that syste

→It must locate and load into memory the OS Kernel.

rupt: → The occurence of an event is usually signalled by an Interrupt from Hardware or Software.

→ Hardware may trigger an interrupt at any time by sending a signal to th CPU, usually by the way of the system bus.

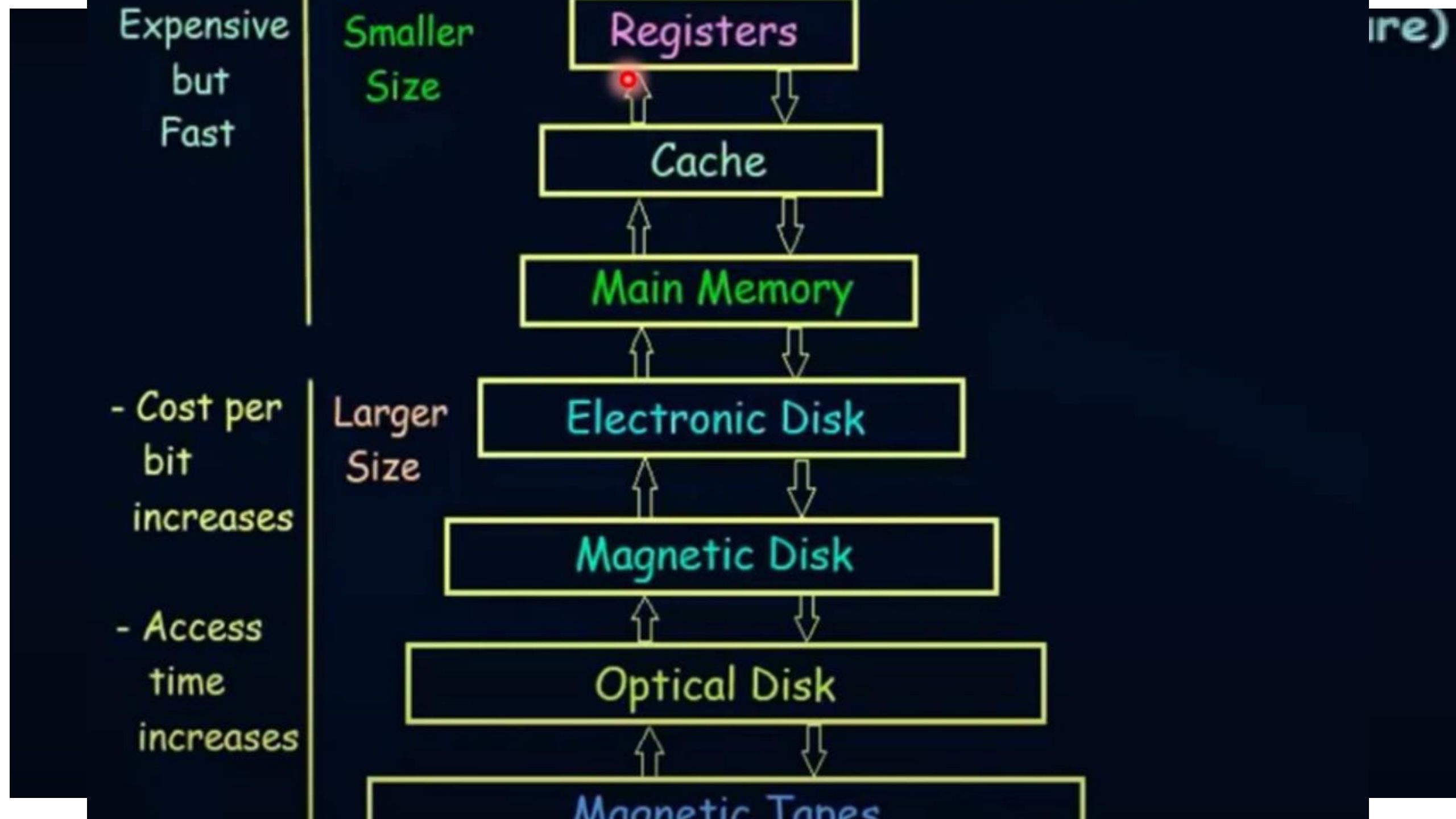
m Call (Monitor call): → Software may trigger an interrupt by executing a speci operation called System Call.

ne CPU is interrupted, it stops what it is doing and immediately transfers executi
ed location.

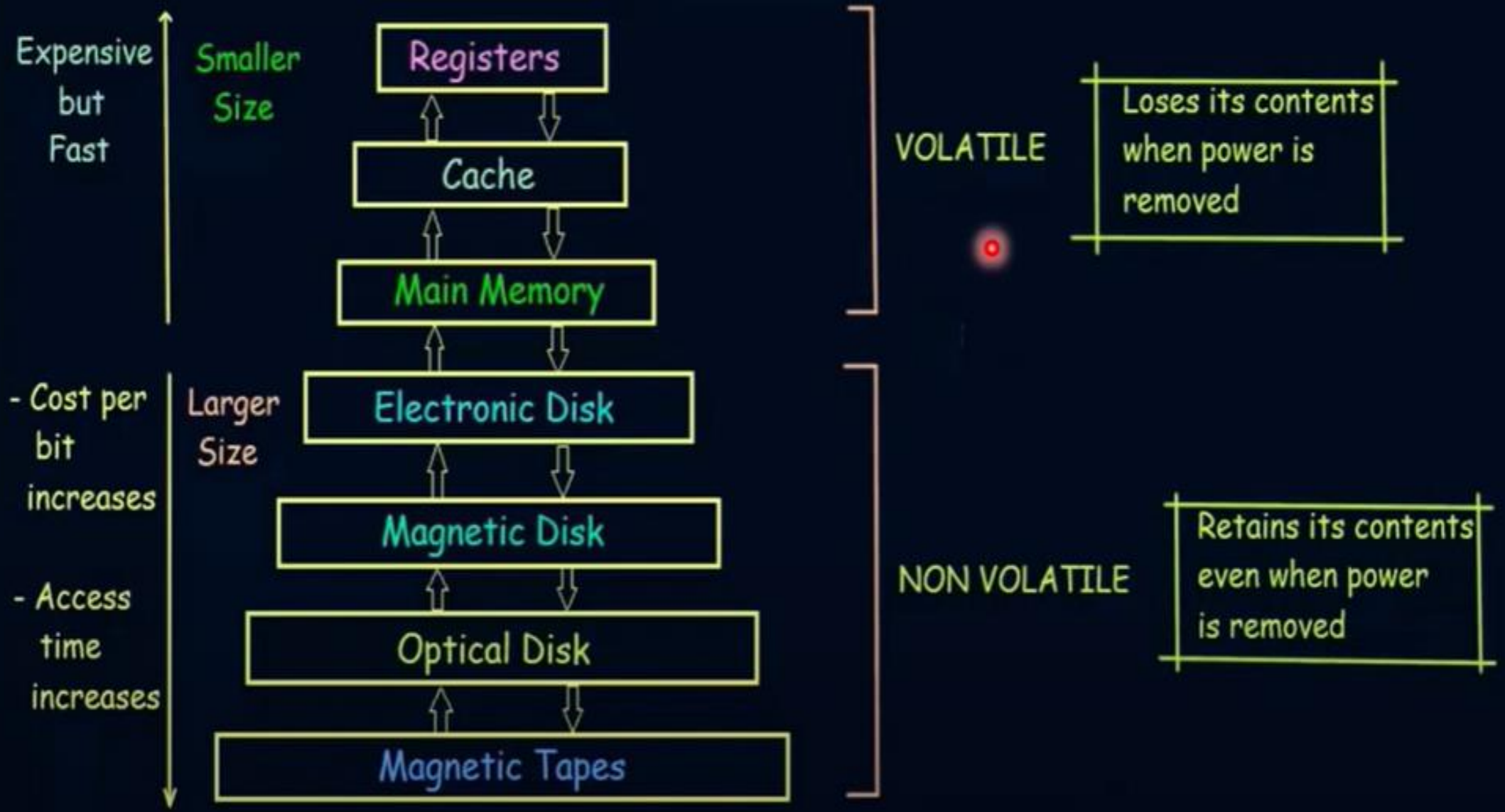
→ The fixed location usually contains the starting address where the Service Routine of the interrupt is located.

errupt Service Routine executes.

pletion, the CPU resumes the interrupted computation.


Expensive
but
Fast
Smaller
Size
Registers
Cache
Main Memory
- Cost per
bit
increases
Larger
Size
Electronic Disk
Magnetic Disk
- Access
time
increases
Optical Disk
Magnetic Tapes

Basics of Operating System (Storage Structure)



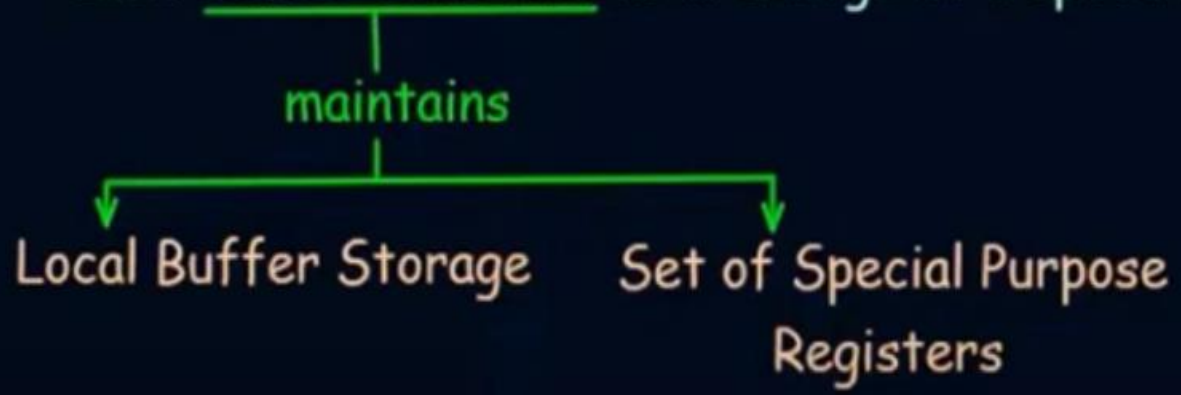
Basics of Operating System (I/O Structure)

-> Storage is only one of many types of I/O devices within a computer

-> A large portion of operating system code is dedicated to managing I/O, both because of its importance to the reliability and performance of a system and because of the varying nature of the devices

-> A general-purpose computer system consists of CPUs and multiple device controllers that are connected through a common bus

-> Each device controller is in charge of a specific type of device



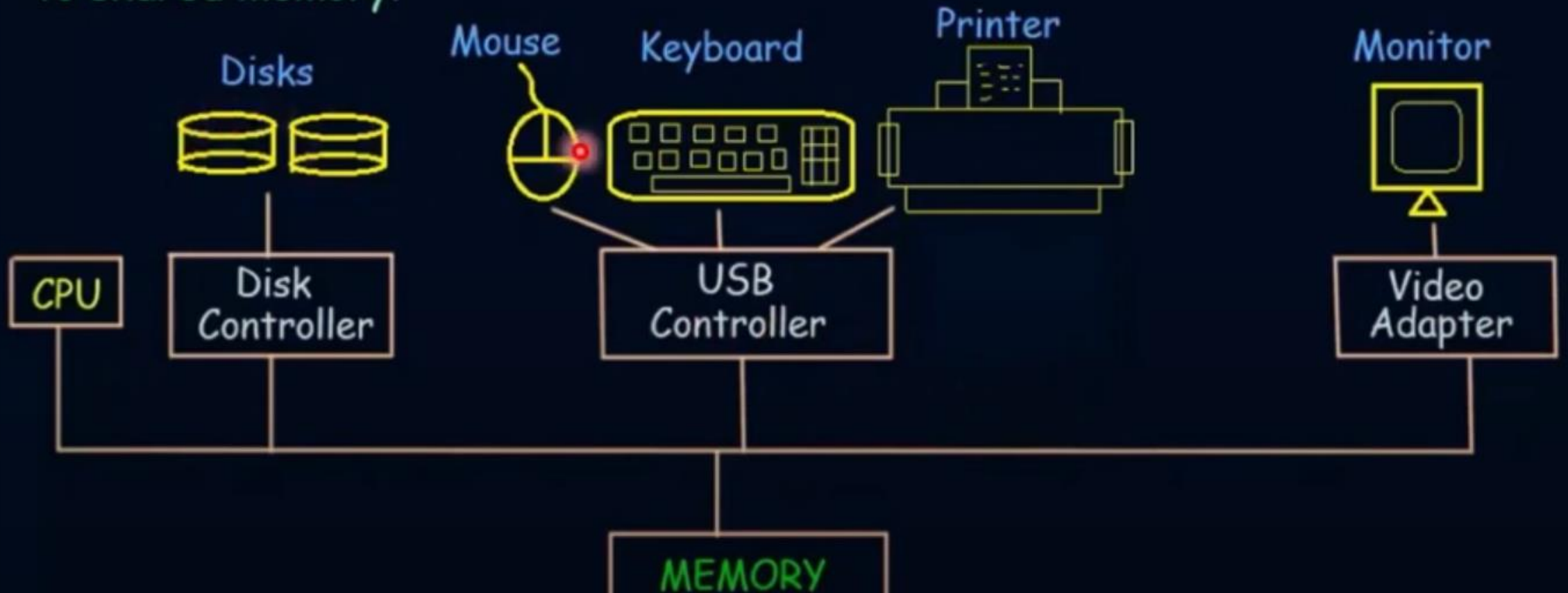
-> Typically, operating systems have a device driver for each device controller

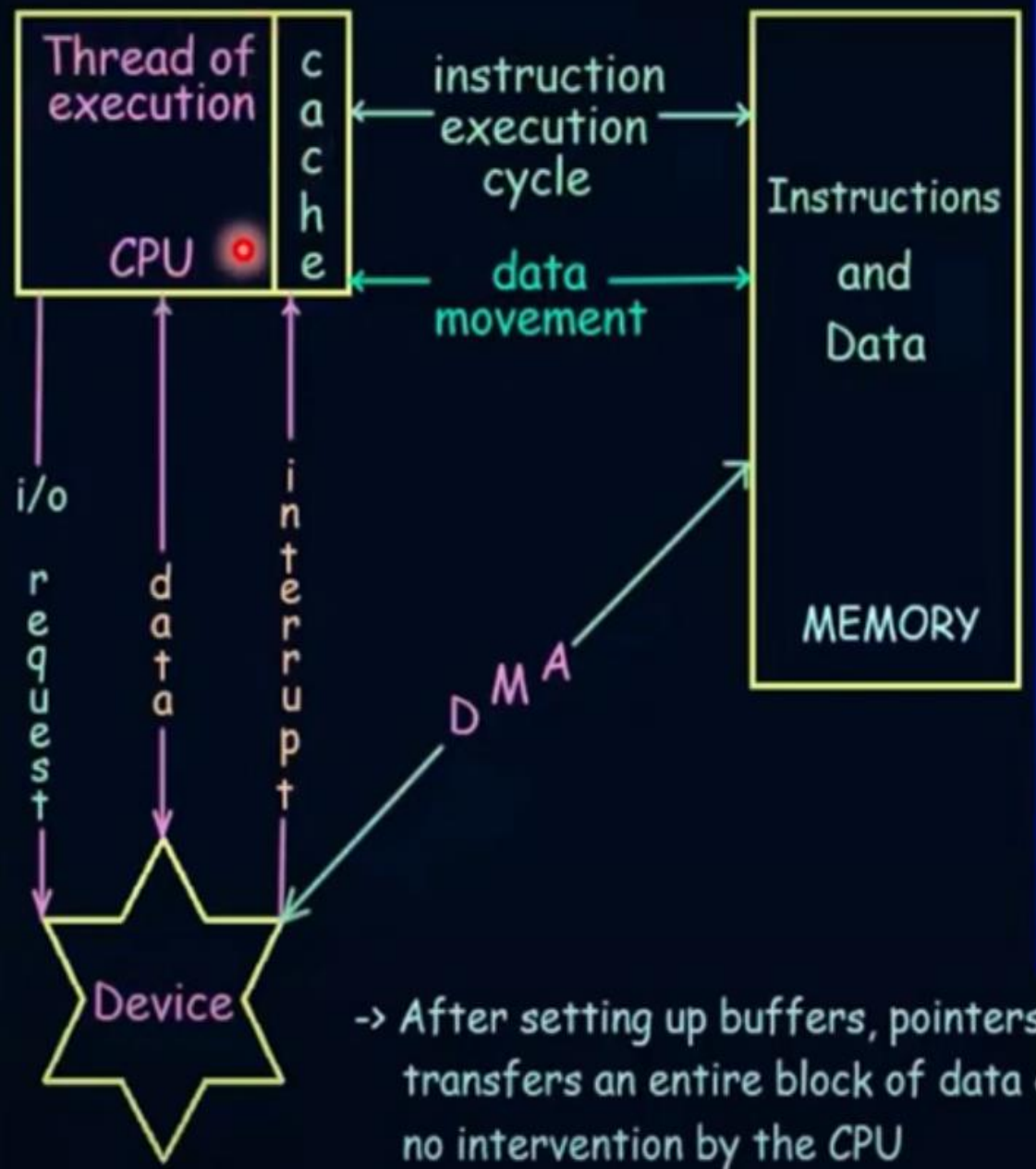
-> This device driver understands the device controller and presents a uniform interface to the device to the rest of the operating system

Basics of Operating System (Computer System Operation)

Some basic knowledge of the structure of Computer System is required to understand how Operating Systems work.

→ A modern general-purpose computer system consists of one or more CPUs and a number of device controllers connected through a common bus that provides access to shared memory.





-> To start an I/O operation, the device driver loads the appropriate registers within the device controller

-> The device controller, in turn, examines the contents of these registers to determine what action to take

-> The controller starts the transfer of data from the device to its local buffer

-> Once the transfer of data is complete, the device controller informs the device driver via an interrupt that it has finished its operation

-> The device driver then returns control to the operating system

This form of interrupt-driven I/O is fine for moving small amounts of data but can produce high overhead when used for bulk data movement

To solve this problem, Direct Memory Access (DMA) is used

-> After setting up buffers, pointers, and counters for the I/O device, the device controller transfers an entire block of data directly to or from its own buffer storage to memory, with no intervention by the CPU

-> Only one interrupt is generated per block, to tell the device driver that the operation

COMPUTER SYSTEM ARCHITECTURE

s of Computer Systems based on number of General Purpose Processors

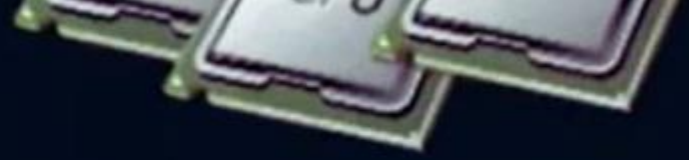
1. Single Processor Systems 

2. Multiprocessor Systems 

3. Clustered Systems 

1. Single Processor Systems:

- One main CPU capable of executing a general purpose instruction set including instructions from user processes.
- Other special purpose processors are also present which perform device specific tasks



Also known as parallel systems or tightly coupled systems.

Has two or more processors in close communication, sharing the comp
bus and sometimes the clock, memory, and peripheral devices

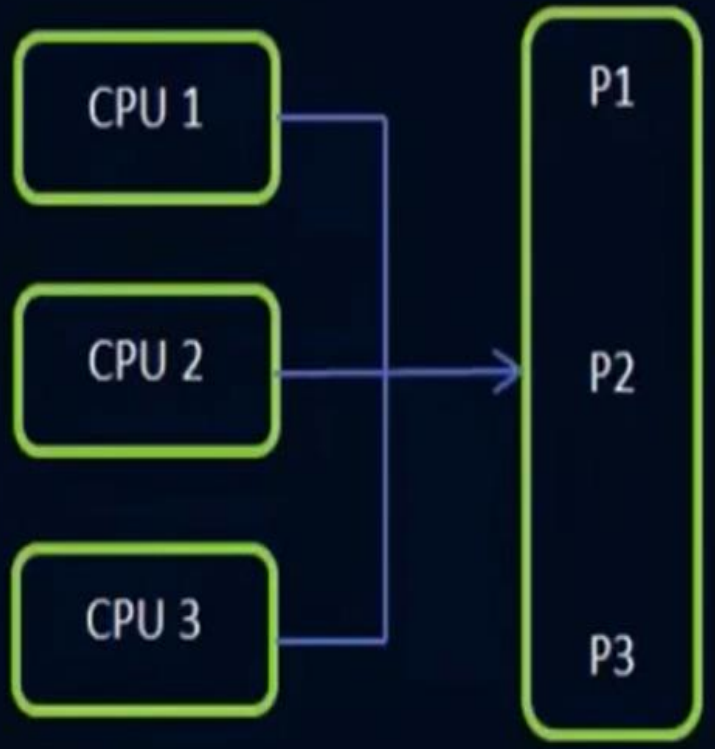
dvantages:

👍 Increased throughput

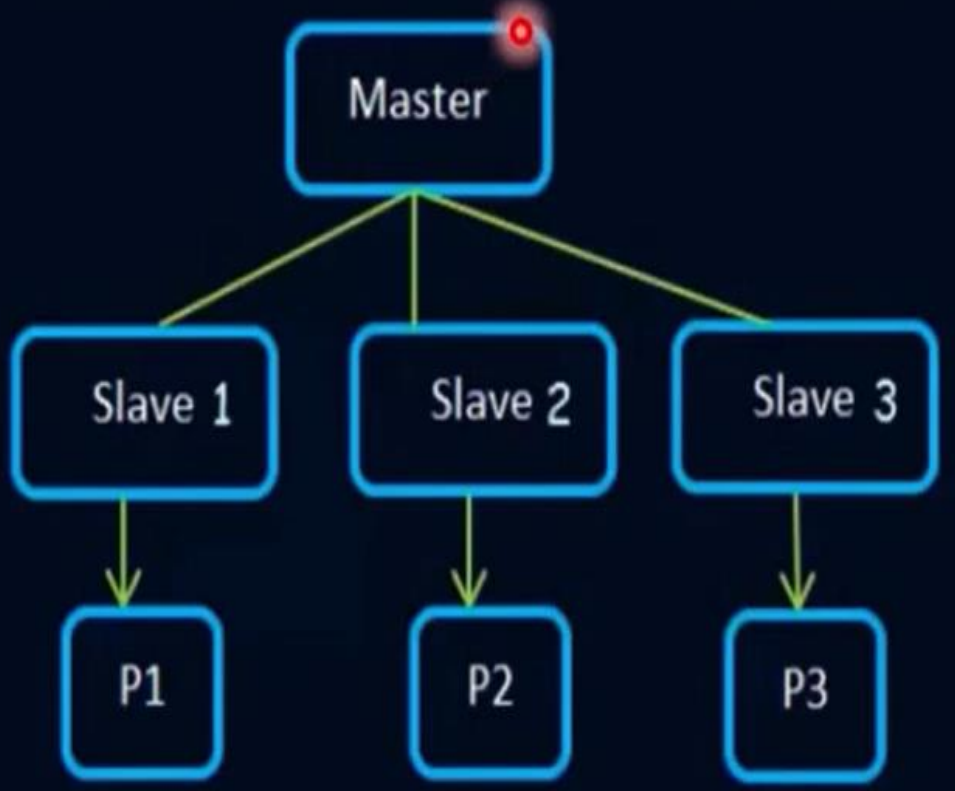
👍 Economy of scale

Types of Multiprocessor Systems:

Symmetric Multiprocessing



Asymmetric Multiprocessing



3. Clustered Systems

- Like multiprocessor systems, clustered systems gather together multiple CPUs to accomplish computational work.
- They are composed of two or more individual systems coupled together.
- Provides high availability
- Can be structured asymmetrically or symmetrically

asymmetrically

- One machine in Hot-Standby mode
- Others run applications

symmetrically

- Two or more hosts run applications
- Monitors each other

Operating System Structure

Operating System Structure

(Multiprogramming & Multitasking)

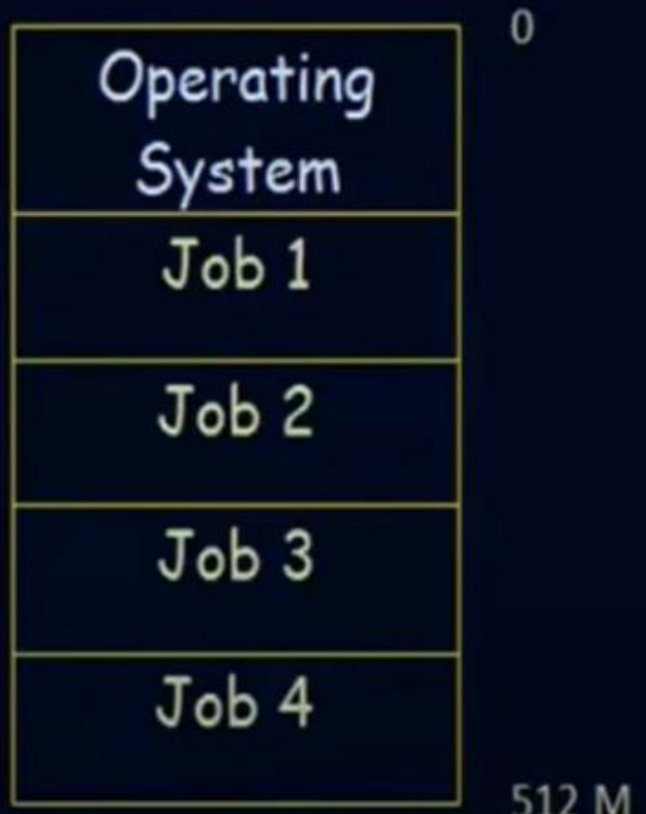
- Operating Systems vary greatly in their makeup internally
- COMMONALITIES:

  (i) Multiprogramming

  (ii) Time Sharing (Multitasking)

(i) Multiprogramming

- A single user cannot, in general, keep either the CPU or the I/O devices busy at all times
- Multiprogramming increases CPU utilization by organizing jobs (code and data) so that the CPU always has one to execute.

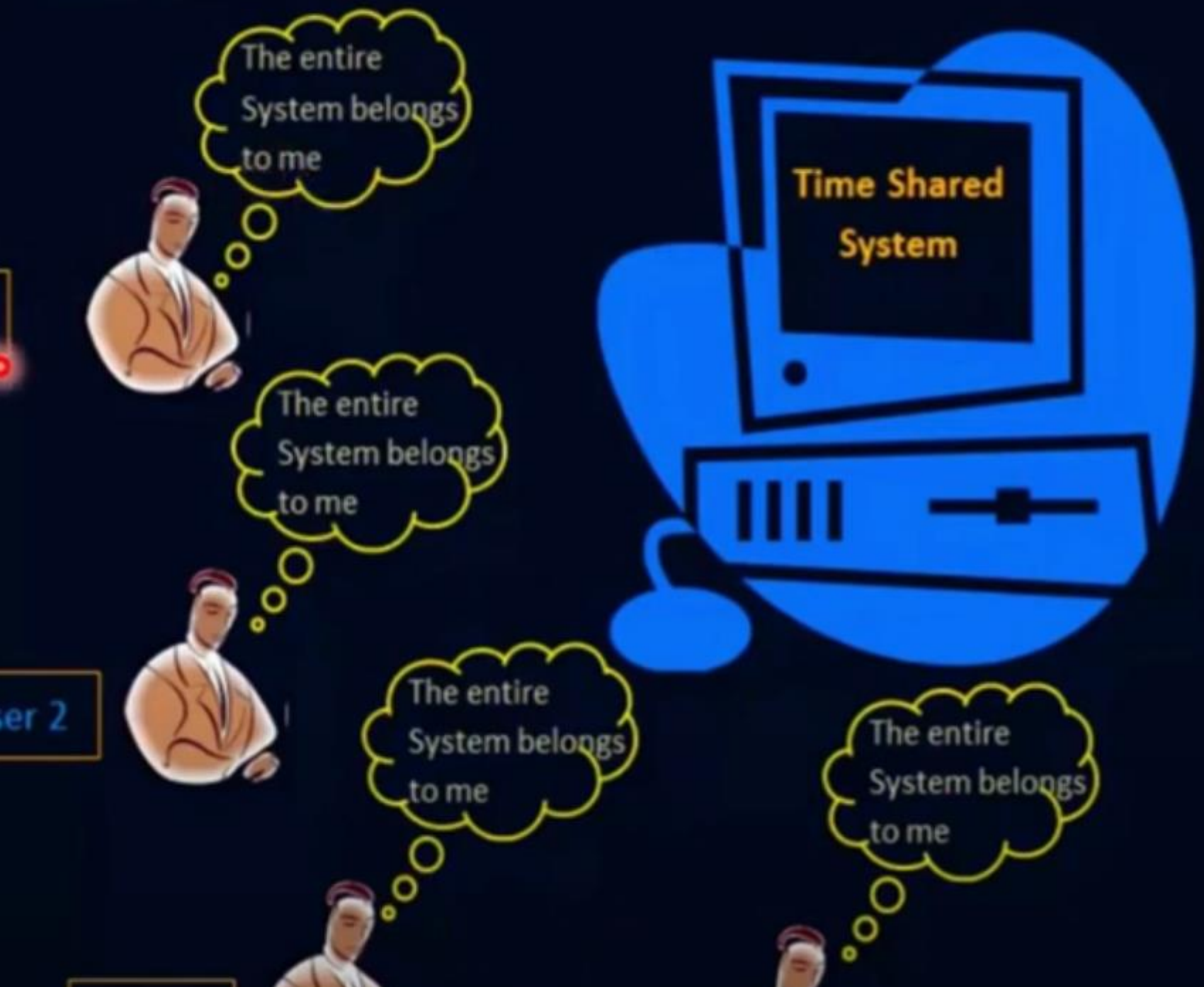


Multiprogrammed syste
provide an environmen
which the various
system resources (fo
example, CPU, memory,
peripheral devices) a
utilized effectively, but
do not provide for use
interaction with the

(ii) Time Sharing (Multitasking)

- CPU executes multiple jobs by switching among them
- Switches occur so frequently that the users can interact with each program while it is running
- Time sharing requires an interactive (or hands-on) computer system, which provides direct communication between the user and the system.
- A time-shared operating system allows many users to share the computer simultaneously.

simultaneously?



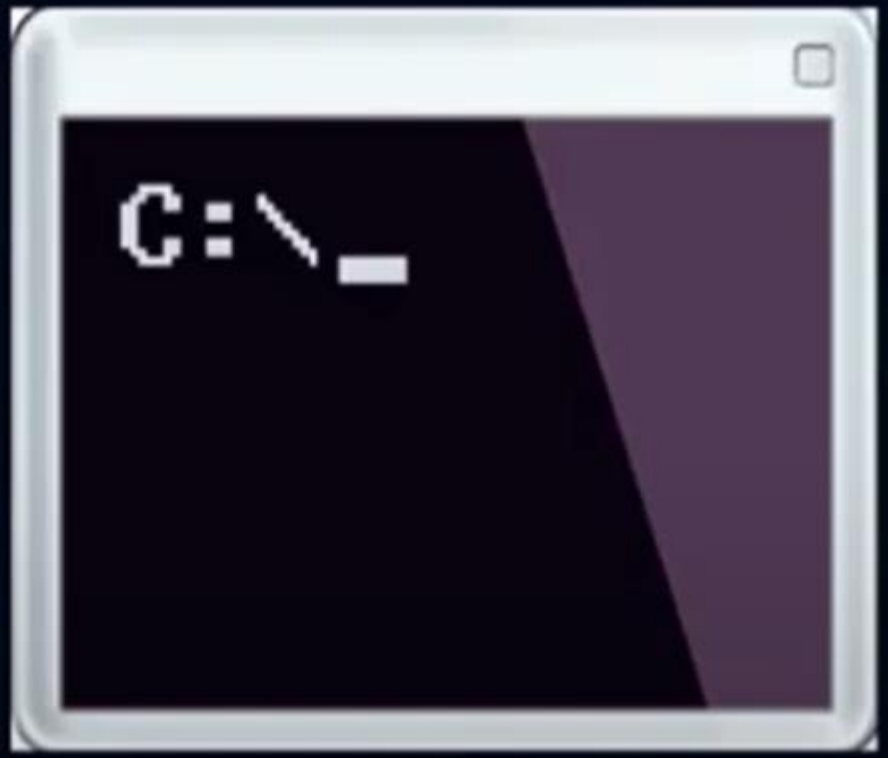
- Uses CPU sched
  and multiprogra
  to provide each
  with a small por
  a time-shared
  computer.
- Each user has at
  one separate pr
  in memory
- A program loade
  memory and ex
  is called a "PRO

Operating System Services

OS provides an environment for the execution of programs

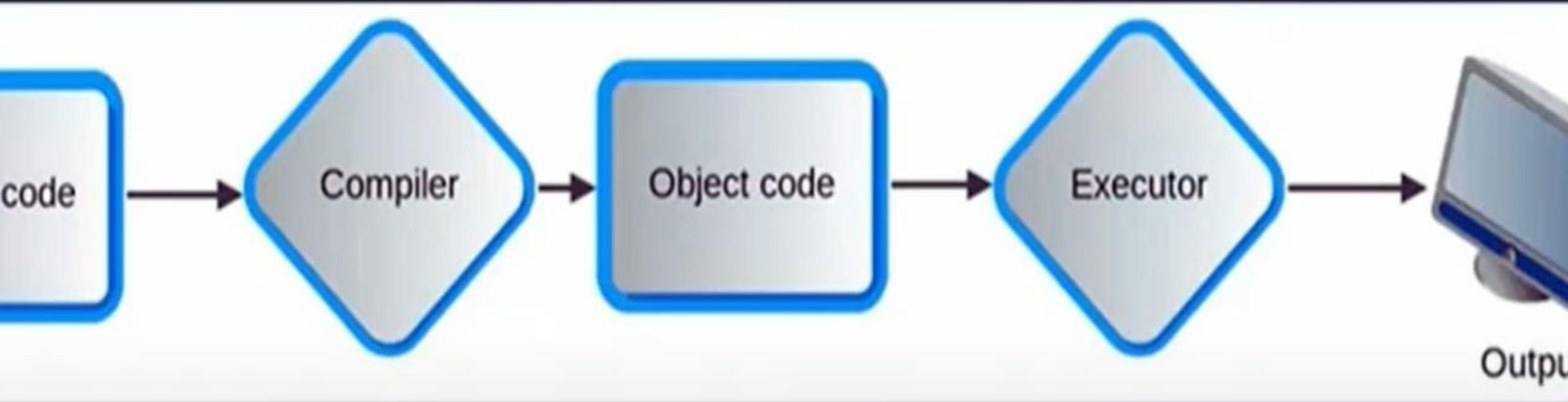
provides certain services to programs and to users of those programs

1) User Interface

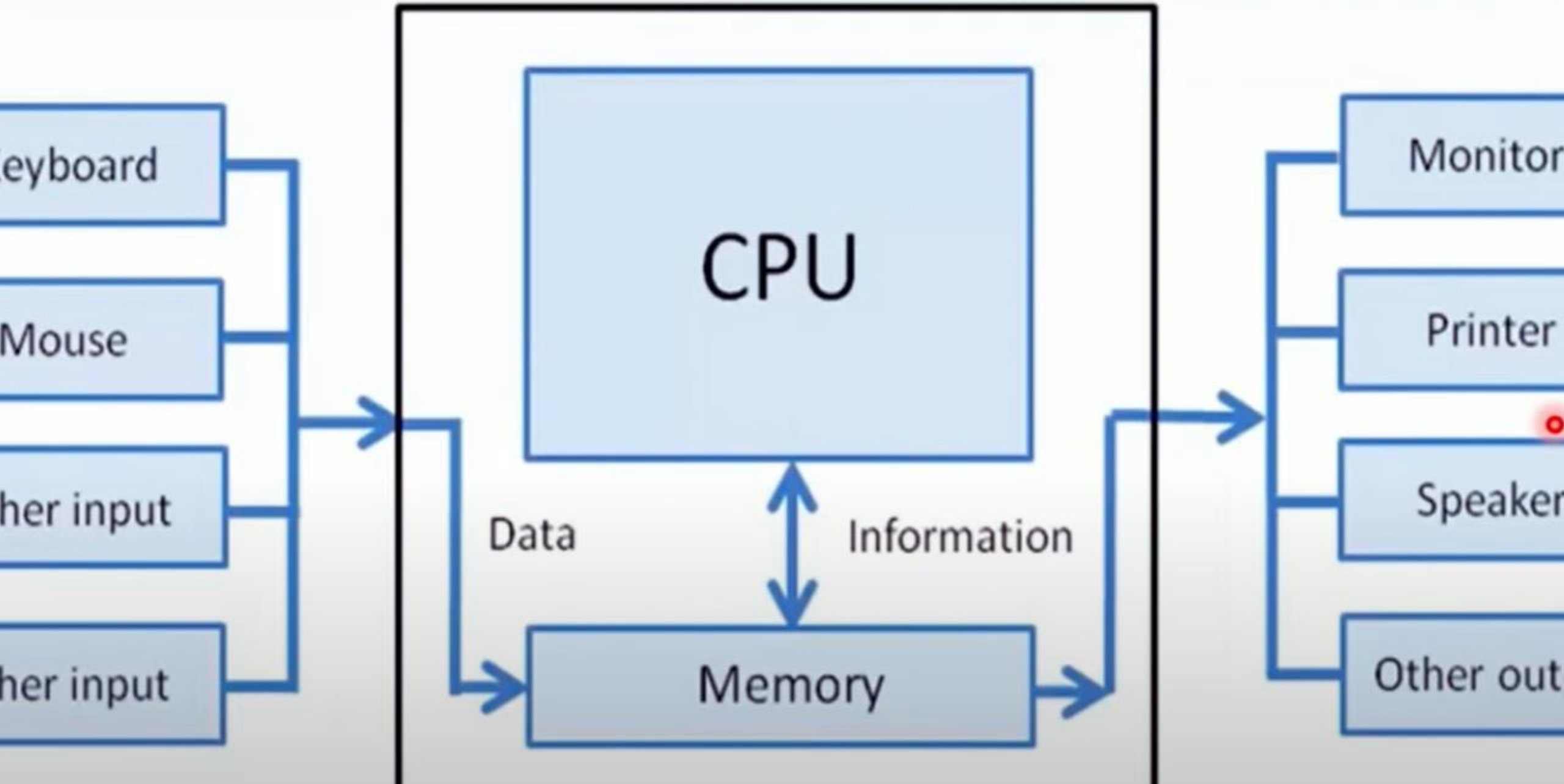


Command Line Interface (CLI)

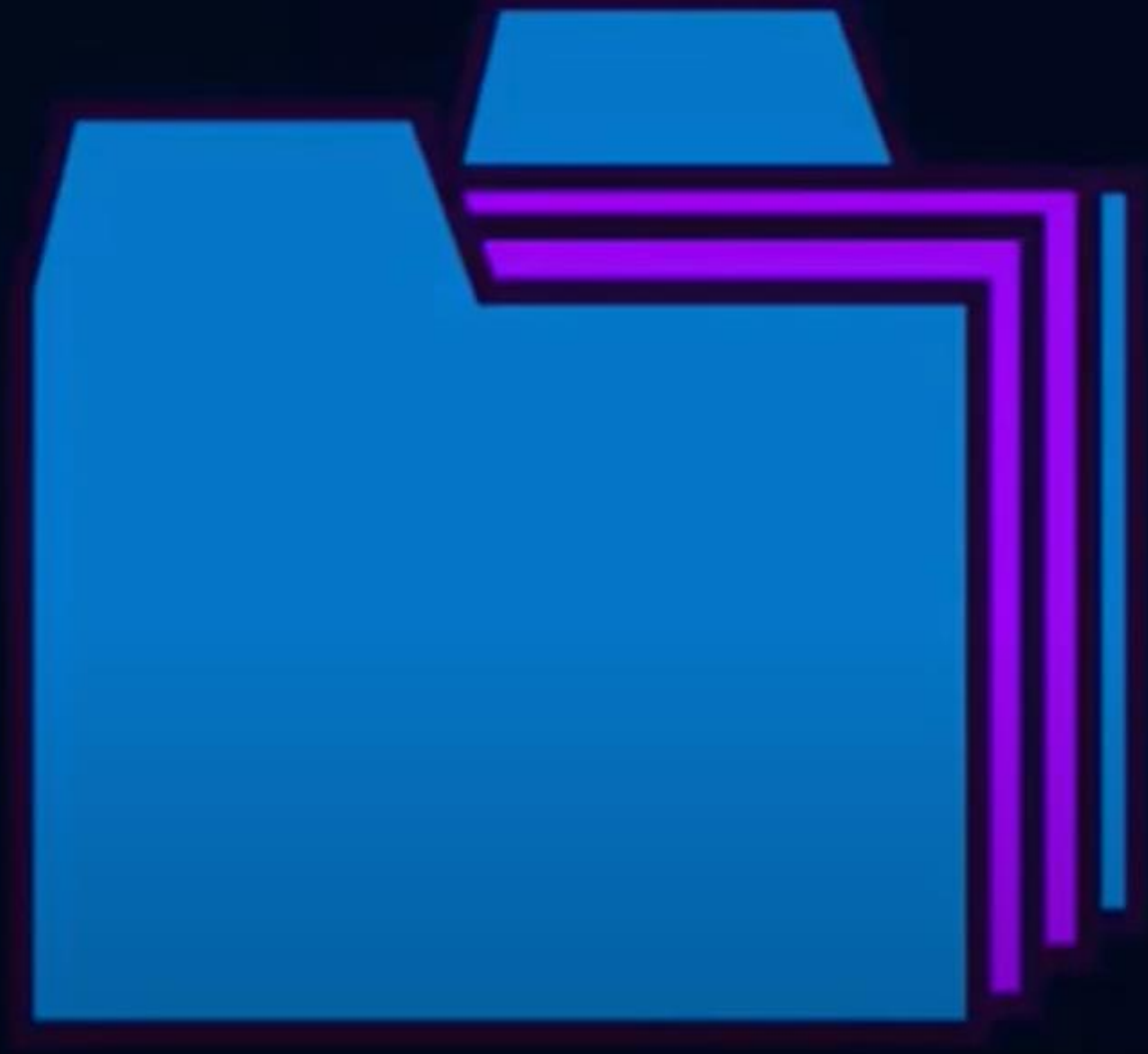
2) Program Execution



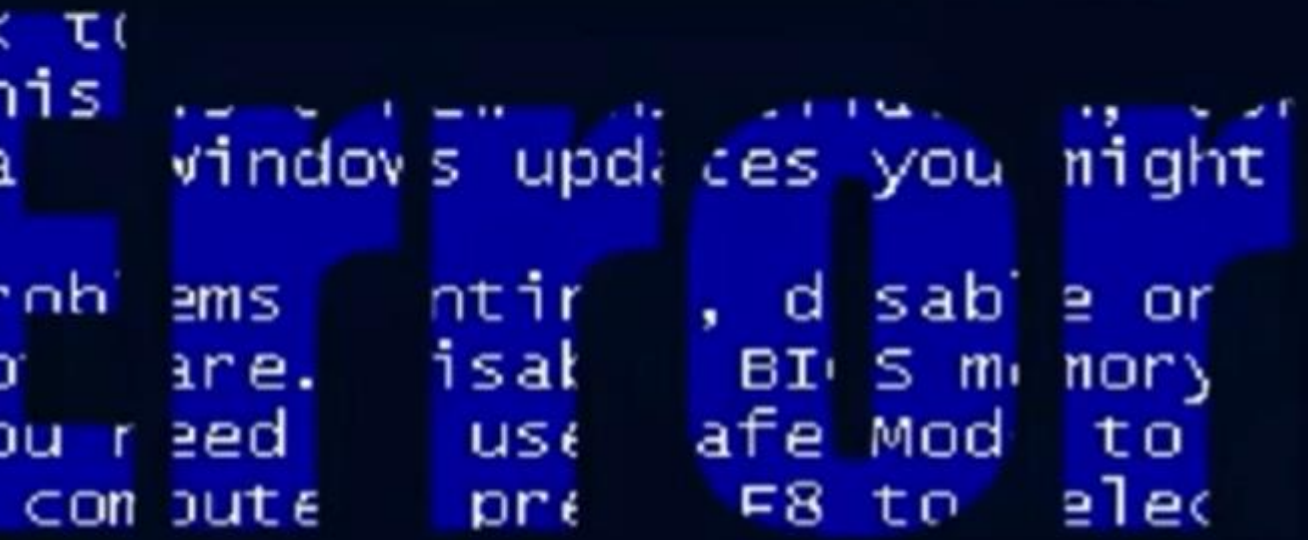
3) I/O Operations



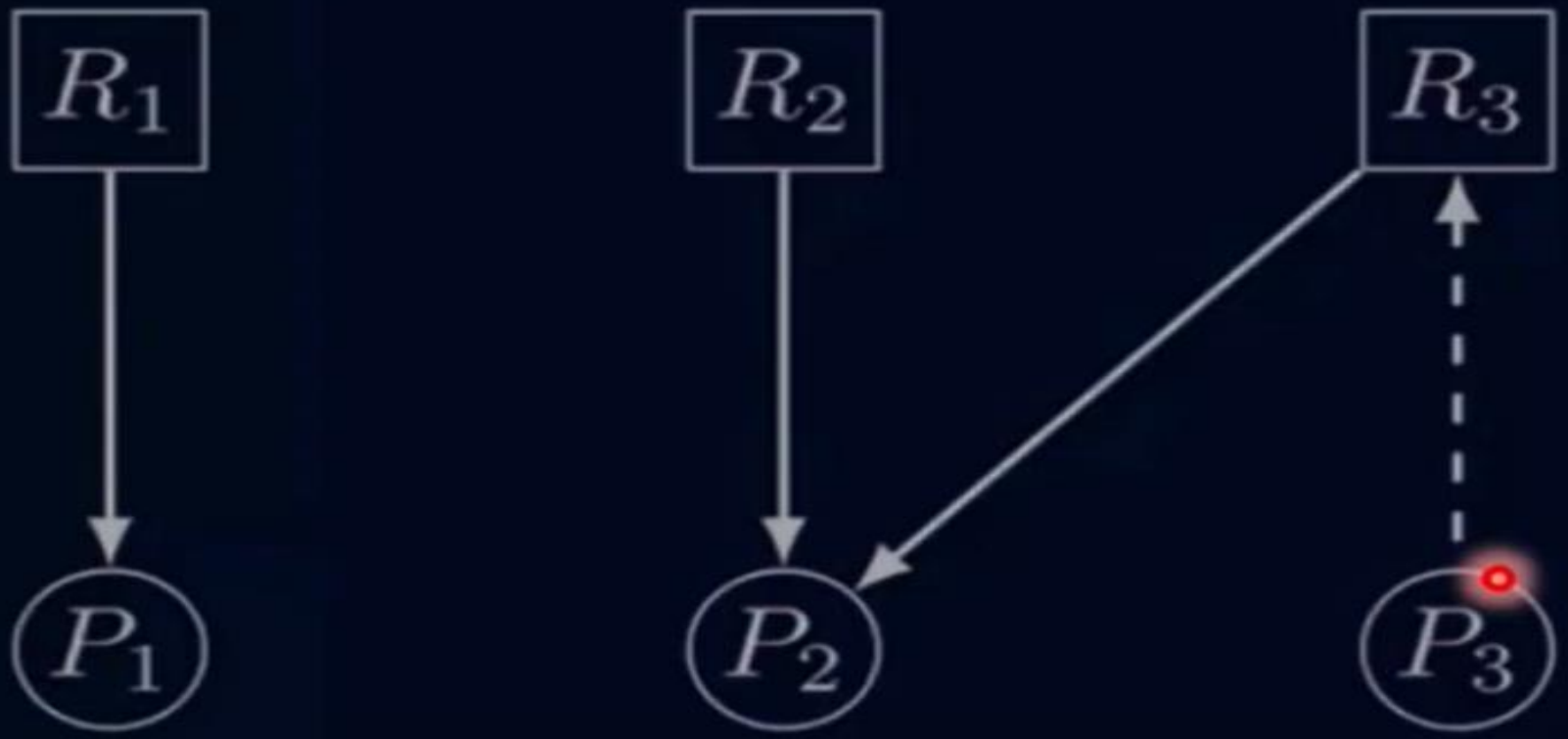
4) File System Manipulation



6) Error detection



7) Resource Allocation



8) Accounting

1. User Interface
2. Program Execution
3. I/O Operations
4. File System manipulation
5. Communications
6. Error Detection
7. Resource Allocation
8. Accounting
9. Protection and Security

Operating System Services

User Operating System Interface

User Operating System Interface

ere are two fundamental approaches for users to interface with the operating system:

rovide a Command-Line Interface (CLI) or Command Interpreter that
llows users to directly enter commands that are to be performed by
perating system.

User Operating System Interface

ere are two fundamental approaches for users to interface with the operating system:

rovide a Command-Line Interface (CLI) or Command Interpreter tha llows users to directly enter commands that are to be performed by perating system.

llows the user to interface with the operating system via a Graphical ser Interface or GUI.

Command Interpreter

- Some operating systems include the command interpreter in the kernel.
- Others, such as Windows XP and UNIX, treat the command interpreter as a special program.

Command Interpreter

- Some operating systems include the command interpreter in the kernel.
- Others, such as Windows XP and UNIX, treat the command interpreter as a special program.
- On systems with multiple command interpreters to choose from, the interpreters are known as shells.
- E.g.
  - Bourne shell,
  - C shell
  - Bourne-Again shell (BASH)
  - Korn shell, etc.

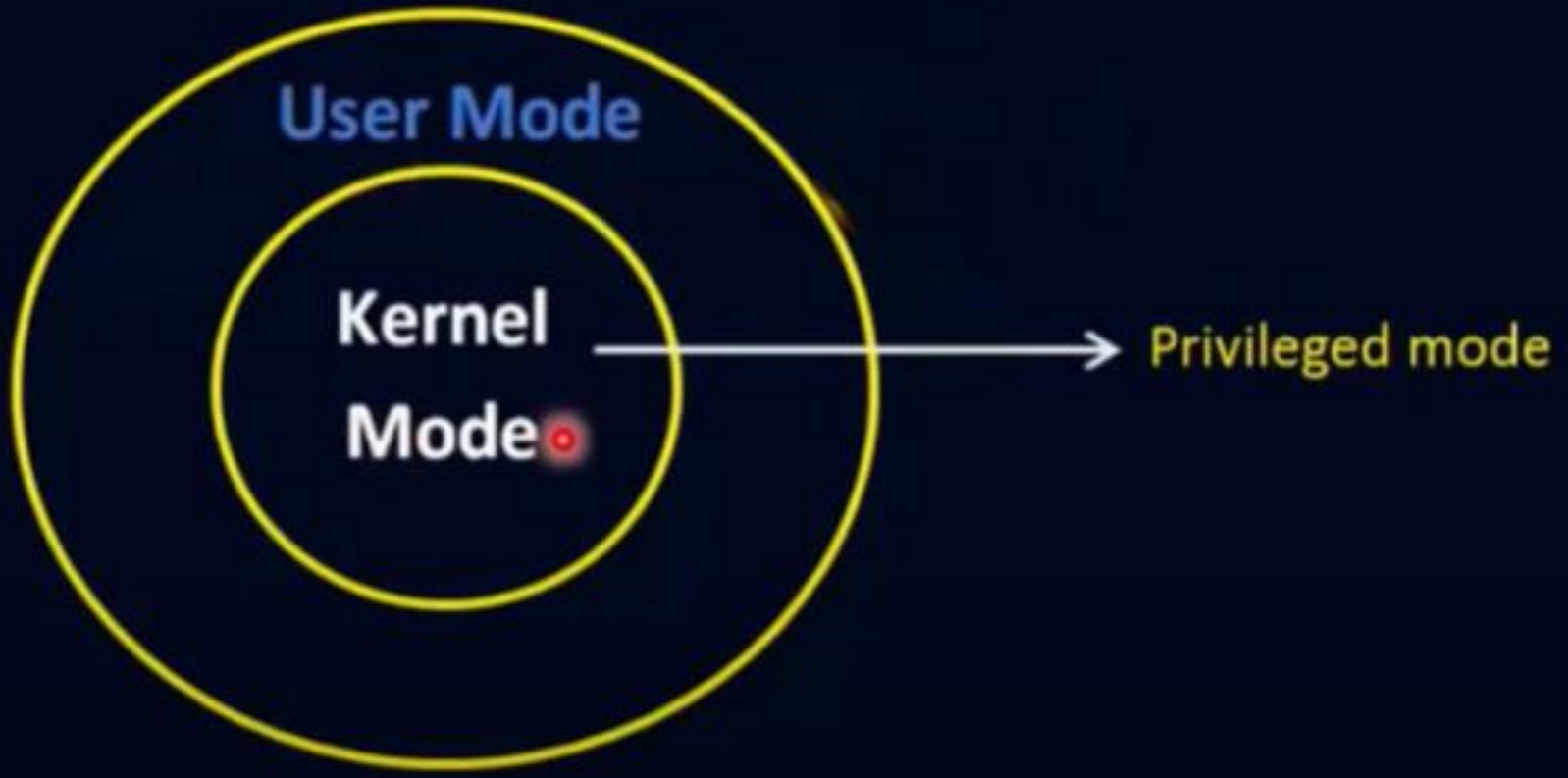
What is System Call

System Calls

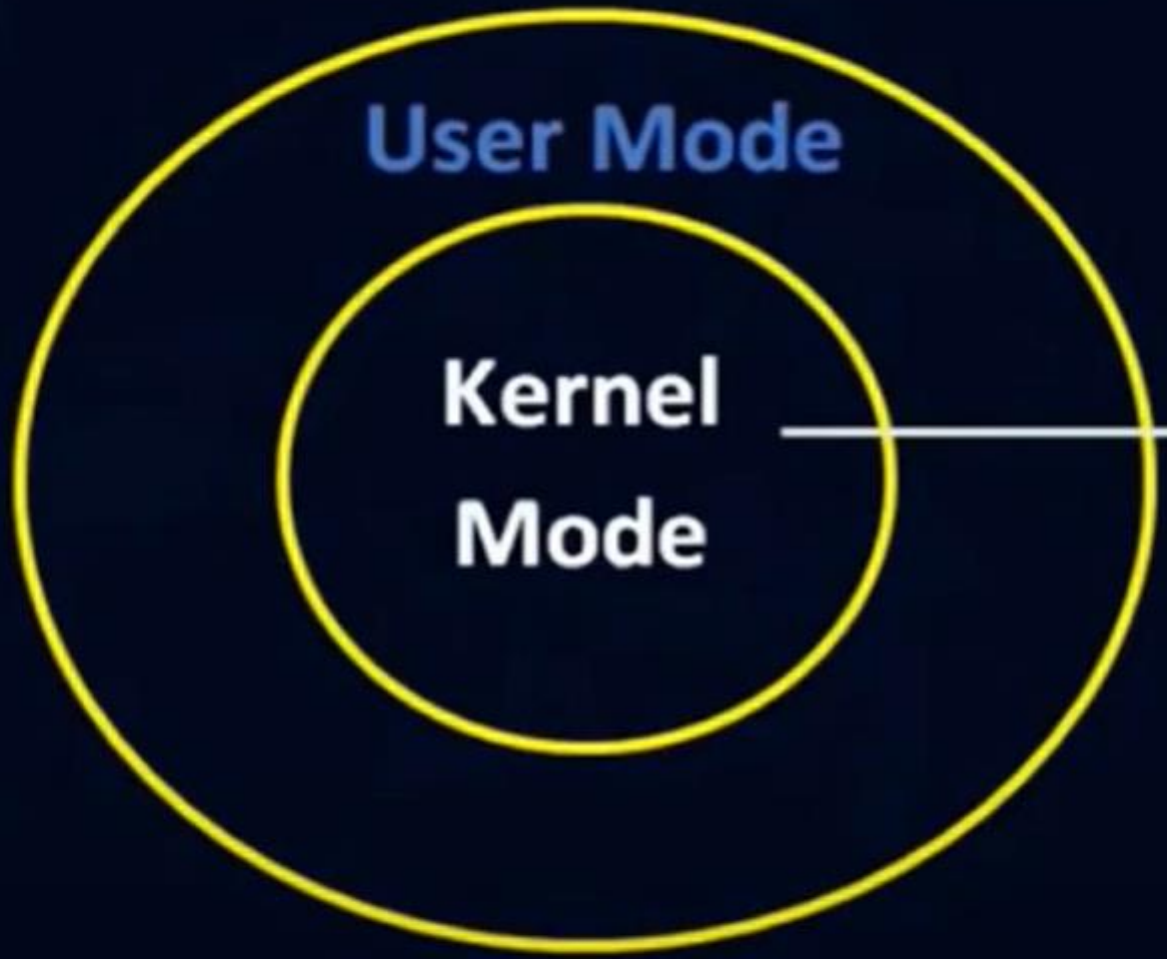
System calls provide an interface to the services made available by an Operating System.

System Calls

System calls provide an interface to the services made available by an Operating System.



perating System.



Privileged mo

System call is the programmatic way in which a computer program requests a service from the kernel of the operating system.


Source File
Destination File
Acquire Input Filename
Write prompt to screen
Accept input
Acquire Output Filename
Write prompt to screen
Accept Input
Open Input File

Types of System Call

Types of System Calls

System calls can be grouped roughly into five major categories:

1. Process Control
2. File Manipulation
3. Device Management
4. Information Maintenance
5. Communications

ss Control

l, abort

l, execute

ate process, terminate process

process attributes, set process attributes

for time

event, signal event

cate and free memory

2. File Manipulation

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

3. Device Manipulation

- request device, release device
- read, write, reposition
- get device attributes, set device attributes
- logically attach or detach devices

4. Information Maintenance

- get time or date, set time or date
- get system data, set system data
- get process, file, or device attributes
- set process, file, or device attributes

5. Communications

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

System Programs

MCQ :

1. Which of the following is the fastest type of storage in a computer system?

A. Hard Disk Drive (HDD)
B. Cache Memory
C. Solid State Drive (SSD)
D. Main Memory (RAM)

MCQ

- **Answer:** B. Cache Memory

- **2. Which storage type is most suitable for long-term, large-scale data storage?**
- A. Registers
  B. Cache Memory
  C. Hard Disk Drive (HDD)
  D. Main Memory (RAM)

- **Answer:** C. Hard Disk Drive (HDD)

Process and Threads

Program and Process difference

- **Program**
- **Definition**: A program is a static set of instructions written in a programming language to perform a specific task or solve a particular problem. It is essentially a file or a collection of files that contains executable code, along with data and other resources required to execute that code.
- **State**: A program is not active; it's just a set of instructions stored on disk (or another storage medium). It remains inert until it is executed.
- **Example**: A word processor application or a web browser installed on your computer.

Program and Process difference

Process

•**Definition**: A process is a dynamic entity that represents an instance of a program in execution. It is an active state that includes not only the program code but also the current activity, including the program counter, registers, and variables. A process is essentially the execution of a program.

•**State**: A process is active and exists in memory while the program is running. It includes the code, data, and the execution context (e.g., the CPU registers, memory allocation, and execution status).

•**Example**: When you open a word processor, an instance of the word processor program is loaded into memory, and the operating system manages this as a process. If you open multiple documents simultaneously, each document might be handled by a separate process or by threads within a process.

Process:

A process can be thought of as a program in execution.

Thread:

A thread is the unit of execution within a process. A process can have anywhere from just one thread to many threads.

Process State

- As a process executes, it changes state.
- The state of a process is defined in part by the current activity of that process.

Process State

- As a process executes, it changes state.
- The state of a process is defined in part by the current activity of that process.

Each process may be in one of the following states:

NEW — The process is being created.

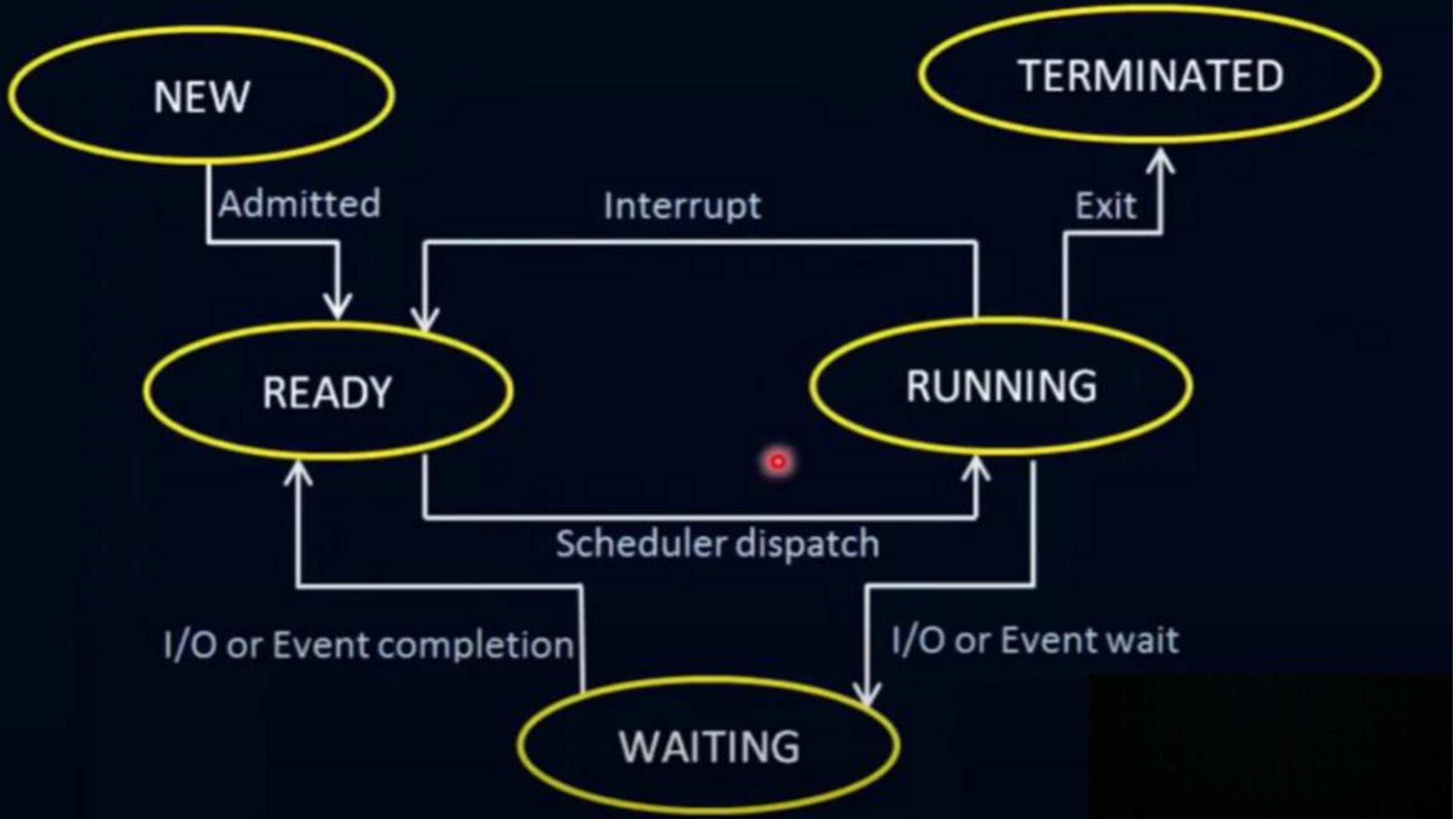
RUNNING — Instructions are being executed.

Process State

- As a process executes, it changes state.
- The state of a process is defined in part by the current activity of that process.

Each process may be in one of the following states:

NEW	The process is being created.
RUNNING	Instructions are being executed.
WAITING	The process is waiting for some event to (Such as an I/O completion or reception
READY	The process is waiting to be assigned to a processor.
TERMINATED	The process has finished execution.


NEW
TERMINATED
Admitted
Interrupt
Exit
READY
RUNNING
Scheduler dispatch
I/O or Event completion
I/O or Event wait
WAITING

Process Control Block (PCB)

- Every process is represented in the operating system by a process control block, which is also called a task control block.
- Here, are important components of PCB

PCB


Process state
Program Counter
CPU registers
CPU scheduling Information
Accounting & Business information
Memory-management information
I/O status information

Process in OS

- **Process state:** A process can be new, ready, running, waiting, etc.
- **Program counter:** The program counter lets you know the address of the next instruction, which should be executed for that process.
- **CPU registers:** This component includes accumulators, index and general-purpose registers, and information of condition code.
- **CPU scheduling information:** This component includes a process priority, pointers for scheduling queues, and various other scheduling parameters.
- **Accounting and business information:** It includes the amount of CPU and time utilities like real time used, job or process numbers, etc.
- **Memory-management information:** This information includes the value of the base and limit registers, the page, or segment tables. This depends on the memory system, which is used by the operating system.
- **I/O status information:** This block includes a list of open files, the list of I/O devices that are allocated to the process, etc.

Process in OS

- **Summary**
- A process is defined as the execution of a program that performs the actions specified in that program.
- Process management involves various tasks like creation, scheduling, termination of processes, and a dead lock.
- The important elements of Process architecture are 1)Stack 2) Heap 3) Data, and 4) Text
- The PCB is a full form of Process Control Block. It is a data structure that is maintained by the Operating System for every process
- A process state is a condition of the process at a specific instant of time.
- Every process is represented in the operating system by a process control block, which is also called a task control block.