Operating System Key.

R22 CS 303PC

II. B.Tech I-Sem

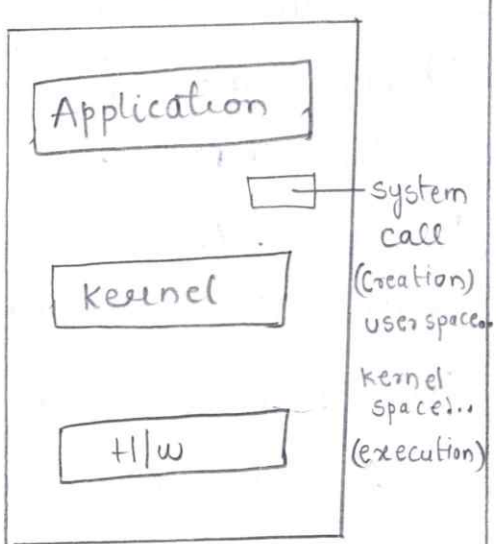
PART-A

(1) a) What is the main purpose of an operating system?

- Operating system act as an interface between the user and Hardware.
- OS act as a system software.
- A system software is nothing but which manages the hardware. In order to communicate with all the things we require one software that is operating system.

b) Define a system call.

- System call will act as an interface between user application and Kernel.
- System call provide services to OS through API (Application Programming interface) when the process is being executed. If it is required any resource then a process create a system call and send to the Kernel.



c) What is turnaround time in cpu scheduling?

- Turnaround Time is the total amount of time taken by a process from the moment it enters the ready queue until it completes its execution.

Turnaround Time = Completion Time - Arrival Time

d) List any two conditions required for a deadlock to occur?

1) Mutual Execution

It means a process can use one resource at a time i.e remaining all resources in non sharable method / state.

2) Hold and wait

A process is holding some resource and waiting for another resources but those resources are acquired by other processes because of hold & wait the deadlock may occur in the system.

3) Non preemption.

In this cpu, cannot stop the execution of Process. A process cannot release the resource until its creation.

4) Circular wait

A set of process leads to be deadlock condition because they are waiting for the resource which are occupied by some other process the entire process will happen in Circular way.

e) Define Critical section?

- Critical section is a part of program where the shared resources are accessed by variable processes.
- Here various process are nothing but coorperative process.

f) What is interprocess communication?

A process can coordinate and interact with with another process using a method called Interprocess communication.

There are two types

1) Independent process
2) cooperative process

g) Define paging and segmentation?

Paging: It is a non-contiguous technique where the process is divided into no.of partitions. Those partitions are called pages. In this method whenever the free space is available At that place the process is going to allocate memory.

Segmentation: It is another type of non contigious memory management technique. It is a method in which the process is divided into some parts & those parts are called segments.

h) What is demand paging?

Demand paging is a memory management technique in which pages of a process are loaded into main memory only when they are required for execution.

It follows 'load on demand' concept.

i) What is a directory structure?

A directory structure is the way an operating system organizes and manages files on storage devices. It defines how directories and sub-directories are arranged so users can store, access, search, and manage files efficiently.

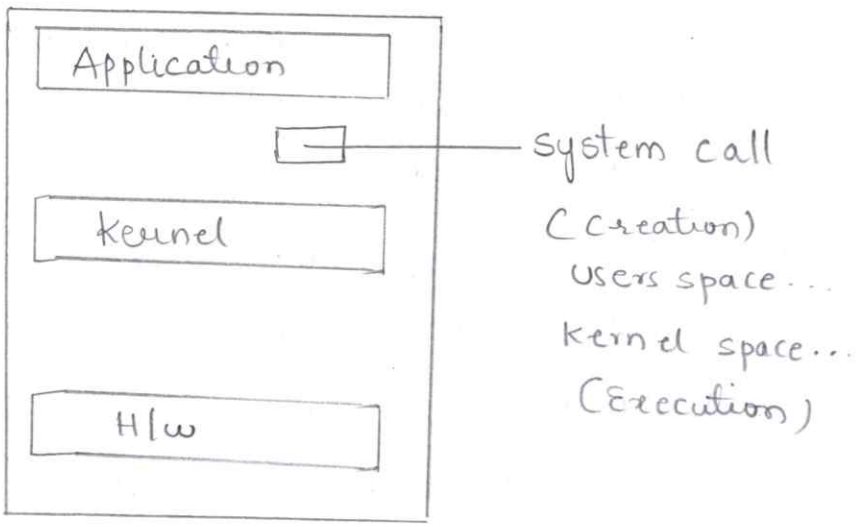
j) Define file protection?

File protection is the mechanism used by an operating system to control improper access to file and to protect them from accidental or intentional damage.

PART-B

(2) Explain the various types of system calls and their purposes in operating system.

→ System call will act as an interface between user applications and kernel.



→ System call provides services to the OS through API (Application programming interface) when the process is being executed. If it is required any resources then a process create a system call and send to the kernel.

→ Generally, memory is divided into user space and kernel space.

User space (creation of sc)
Kernel space (execution of sc)

→ All the system calls are executed internel space

and created in User space.

→ Whenever, Kernel space execute the system call once it is executed the control will be back to the user space because the process execution will be in user space.

→ System calls are only the entry point to the Kernel space.

→ Based on the priority the system calls, will be executed and stored in the system call table.

Types of system call

There are 5 types of system call

1) process control system call
2) File management System call
3) Device management System call
4) Information maintanance system call
5) Communication System call.

1. Process control system call

It is mainly responsible for process management Example, Creation of process, termination of process, resource allocation of process.

2. File Management System call

These are responsible for file manupulation like file creation, reading, writing and closing a file.

3. Device Management System call

These system calls are responsible for device

manipulation like creating the data to buffer and reading the data from buffer.

4. Information maintanance system call.

These system calls are responsible for information being shared between the process.

5. Communication system call

These system calls are respon sible for Ipc.

Acquire Input filename
write prompt to screen
Accept Input
} I/p src

Acquire output filename
write prompt to screen
Accept I/p
} o/p

Open Input File

If the file doesnot exist Abort

create o/p File

If the file exist, Abort

loop until read fails [
Read from I/p, file
write o/p File

close o/p File

Write completion Message to terminate normally.

System calls allow user programs to request services from the operating system. Their main purpose is to provide safe, controlled access to OS resources such as files, devices, memory and processes. They act as an interface between user mode and kernel mode, enabling tasks like file operations, process creation, device I/O, memory management, and communication between processes.

(3) Discuss the components and services of an operating system with suitable example.

An operating system is a system software that manages hardware resources and provides essential services to users and application programs. Its functions are grouped into components and services.

Components of operating system: (internal parts of OS)

1) Kernel
   - The core part of OS
   - Handles CPU scheduling, memory management, device control
   - Runs in kernel with full access to hardware

2) process Management
   - Creates, schedules, and terminates processes.
   - Manages CPU sharing, context switching, and inter-process communication.

3) Memory Management
   - Tracks with parts of Memory are in use.
   - Allocates and deallocates memory.

- Supports paging, Segmentation, and Virtual Memory.

4) File System Management
- Organizes data in files and directories
- provides system calls: Create, delete, open, read, write
- Maintains directory structures and free space list.

5) I/O System Management
- Control and Manages all input/output devices.
- Uses device drivers to communicate with hardware
- Ensures smooth data transfer between devices and CPU

6) Secondary storage Management
- Manages hard disk and SSDs
- Maintain Free space disk and disk scheduling.

7) Security and protection component
- provides protection aganist unauthorized access.
- Uses passwords, permissions, ACLs etc.:

8) System call Interface
- Acts as a bridge between user programs and os kernel.
- Allows applications to request OS service safely.

Services of operating system: (facilities offered by os)

OS services are classified into User and System Services

A. User Services

1) User Interface

Provides an interface for users to interact with the computer.

Types: GUI (Graphical user Interface)
CLI (command Line Interface)

Example: windows desktop, Linux terminal.

2) File Management

Allows users to:
- Create open, read, write, delete file.
- organize data in directories
- Maintain attributes like size, type location

Example: Saving a document in "Documents" folder.

3) I/O operations

Programs cannot access hardware directly
OS provides system calls for I/O: read(), write()

Example: Reading Input from keyboard, writing to screen

4) Program Execution

Loads a program into memory and runs it.

Handles:
Program loading, Execution, Error Handling

Example: Opening a browser

5) communication

Supports data exchange between process via:
- pipes
- shared Memory, Message queues

Example: chat applications using background processes that communicate with each other

B. System Services

1) Resource Allocation

- Allocates resources
- Ensure fair usage and avoids conflicts

Example: CPU scheduling selects next process to run.

2) Accounting

- Keeps Records of CPU usage, I/O usage etc..
- Used for Billing system analysis

Example: Task Manager usage statistics

3) security and privacy

- protects data and system Integrity
- Ensures only authorized user can access resources

Example: A normal user cannot delete system file in Linux

4) Error Detection

OS constantly checks for errors

Example: "Disk read error", "Access Violation"

(4) Compare and contrast preemptive and Non preemptive scheduling algorithms.

Preemptive Scheduling	Non-preemptive scheduling
1. CPU is taken away forcibly from running process.	1. CPU cannot be taken away; process run until completion or I/O.

2. Based on priority or time slice.	2. Based on arrival and burst time without interruption
3. Provides better response time.	3. Response time may be high for long jobs.
4. More suitable for interactive and real-time systems.	4. Suitable for batch system.
5. More overhead due to context switching.	5. Little to no context switching overhead.
6. can cause race conditions if not handled correctly.	6. No race condition caused by forced preemption.
7. Improves CPU utilization and responsiveness	7. CPU utilization may be low for mixed workloads.
8. Requires complex hardware like	8. Hardware requirement is simple; no timer interrupt needed.
9. Starvation is more likely	9. Less starvation because processes finish once they get CPU.
10. Scheduling is more complex in implementation	10. Scheduling is simple to implement.
11. Better suited for modern OS.	11. Used in very simple OS or older system.
12. Lower average waiting time for short processes	12. Average waiting time may be higher if a long process arrives first.

13. Fairness improves when time quantum is used.	13. Fairness may suffer if long jobs block short jobs.
14. Higher overhead of maintaining ready queue with priorities.	14. Less overhead in maintaining queues.

(5) Explain the conditions that lead to deadlock and describe the resource allocation graph method for detection?

In operating system Deadlock is a situation where set of process is blocked because of each Processes is holding a resource and wait for another resource which are acquired by another Process.

There are 4 necessary conditions for deadlock.

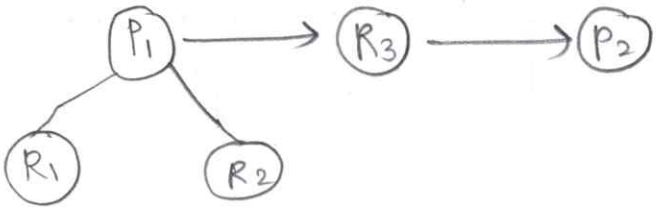
1) Mutual execution

It Means a process can use one resource at a time i.e remaining all resources in non shorable method / static.

Example, Assume that we are sharing the Printer for two processes p1, p2 then it will give the wrong result. So, we have to share one resource at one time.

2) Hold and wait

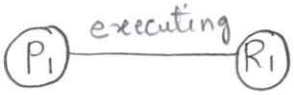
Consider two process P_1, P_2. Where P_1 will hold R_1 & R_2 and wait for R_3 which is holded by P_2 Processes.



A process is holding some resources and waiting for another resources but those resources are acquired by other processes because of hold & wait the deadlock may occur in the system.
If we eliminate this condition then there is no deadlock.

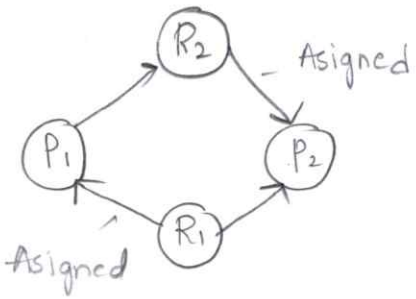
3) Non preemption

In this CPU, cannot stop the execution of Process. A process cannot release, the resource until the execution. If we eliminate this condition then we can say that there is not deadlock.



4) Circular wait

A set of process leads to be deadlock condition because they are waiting for resources which are occupied by some other processes the entire process will happen in circular way.



If these 4 conditions occured simultaneously in the system then deadlock may occur.

Resource Allocation Graph

→ Generally a graph is represented with vertices and edges G*(V,E). In this resource allocation graph the resource type contain instances. Here, vertices is nothing but process or resource type.

→ A process is represented by circle 'O'.

→ Resource type is represented with the help of instances.

→ where each dot represented as instance means of a process contain u instance

Edges:

There are mainly 3 types of edges

1) claim edge
2) request edge
3) Assignment edge

1) claim edge:

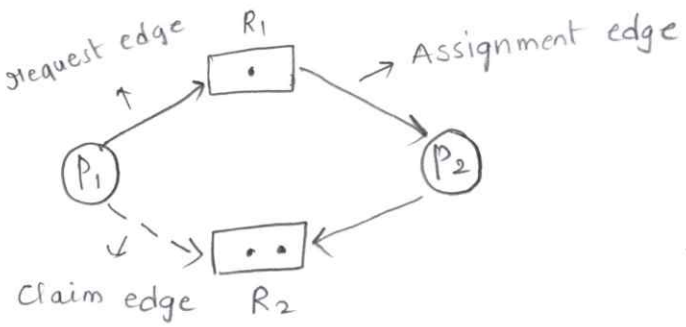
whenever a process want the resource in future then it will use claim edge. It is denoted as (- - -->).

2) Request edge

whenever a process want to use the resource then it put a request to a particular resource It is denoted as ($\longrightarrow$)

3) Assignment edge

whenever the resource is allocated to particular Process then we will use assignment edge. It is denoted by ($P_2 \longleftarrow R_2$)



If a graph contain a cycle then there is possibility deadlocks.

If there is no cycle in the graph then there is no deadlock.

Note:

If any resource type contain single instance in the group then there is a chance of deadlock.

⑥ Explain the producer - consumer problem and demonstrate its solution using semaphores?

Producer - consumer problem

→ Here, we have two processes p1, p2 the p1 will act as a producer and p2 will act as a Consumer.

→ Here, we have shared buffer and count Variable.

→ The shared buffer is nothing but memory (or) storage area and the count variable is nothing but an integer variable which is accessed by both produce and consumer.

→ The producer will produce the item and that should be placed in shared buffer as well as the consumer will consume the item from shared Buffer.

→ Here, the size of the buffer is 5 and initial value is 0.

→ when ever a producer produces an item the count is incremented by 1.

Parallely, whenever Consumer consumes the item the count is decremented by 1.

Solution using Semaphore

- This problem consist of n slots and each slot is capable of storing unit amount of data.

We will use 3 semaphore

1) m(mutex): It is a binary semaphore which is used to acaure and release the lock. Then it

is also called mutual exclusion.

2) Full: It is also a counting semaphore whose initial value is 0

3) empty: It is a name of a semaphore and it is also a counting semaphore. whose initial value is equal the no of slots in the buffer i.e initially all the slots in buffer are empty.

Algorithm:

Producer:- (When Buffer full, producer cannot Produce item)

```
do
{
wait(Empty); // wait until empty > 0 and then
                decrement empty
wait(mutex); // acquire lock (when producer
        acquire lock, consumer will not do any
            changes)
Signal(mutex); // release lock
```

Consumer:-

```
do
{
wait(full); // full>0
wait(mutex); // acquire lock (consumer got lock)
// remove data from Buffer
Signal(mutex); // unlock by signal
Signal(empty); // increment the empty (consumer
```

remove item from Buffer)
} while (true);

⑦ Explain inter process communication (IPC) and describe how it is implemented in a single computer system.

→ Interprocess communication is a mechanism that allows processes to exchange data and coordinate their activities.

→ Processes may be running concurrently, and IPC ensures that they can share information, synchronize actions, and communicate effectively.

→ IPC is essential because processes normally run in separate memory space, so they cannot directly access each other's data.

Purpose of IPC:

- Data sharing
- computation speedup
- Modularity
- Convenience
- process + synchronization

Two Types of IPC:

1) Message passing
- communication happens by sending Messages.
- processes do not share memory.
- useful in distributed systems or when sharing memory is unsafe.

2) Shared Memory
- A region of memory is shared by processes.
- Both process read/write data in that common area
- fastest form of IPC.

IPC Implementation in a single computer system:

on a single machine, Ipc is implemented using various os mechanisms.

The most commonly used are:

1) pipes
- pipes allow unidirectional communication between processes
- Data flow in a producer → consumer fashion.
- common in UNIX/Linux.

2) Named pipes
- similar to pipes, but they have a name in file system
- Allow communication between unrelated processes.

3) shared memory
- os creates a memory segment that multiple processes can attach to.
- Fastest Ipc method since no os intervention is need after creation.
- Requires Synchronization using semaphores or mutex locks.

4) Message Queues
- Kernel-managed linked list of messages
- processes send and receive messages asynchronously
- Suitable when processes need structured communication.

5) Semaphores
- Used for synchronization, not for data transfer
- Prevent race condition when processes access shared data.

6) Signals
- Used to notify a process that an event has occured
- Signals interrupt a process to call a predefined signal handler.

7) Sockets
- Even on a single machine, process can communicate using local sockets.
- useful for client-server applications.

Example:
Web server and browsers communicating locally.

(8) Explain the concept of Virtual Memory and discuss different page replacement algorithms?

Virtual Memory is a memory management Technique that allows a computer to execute programs that are larger than the available physical RAM.

It gives each process the illusion of having a large, continuous memory, while the OS actually stores parts of the process in RAM and the rest in secondary storage.

How virtual Memory works

- The OS divides the logical memory of each process into fixed size blocks called pages.
- physical memory is divided into frames.
- only the required pages are loaded into RAM, the remaining pages stay on disk.
- If a required page is not in RAM, a page fault occurs.
- The OS brings the required page from disk to RAM
- If RAM is full, a page replacement algorithm selects which page to Remove.

Advantages

- Allows execution of large programs
- Increases Multiprogramming.
- Efficient use of physical RAM.

Page Replacement Algorithm

When a page fault occurs and the memory is full, the OS must select a page to replace. Different page replacement algorithms decides which page to remove.

1) FIFO (First-in-First-Out) page replacement algorithm:

→ The oldest page in memory is replaced first.
→ Maintains pages in a queue.
→ simple to implement

Advantages:
- Easy and fast to implement

Disadvantage:
- may remove frequently used pages
- Suffers from Belady's Anomaly

2) Optimal page Replacement

→ Replaces the page that will not be used for the longest time in the future.
→ Gives the minimum possible page faults.

Advantages
- Best performance; theoretical benchmark

Disadvantages
- Cannot be implement in real systems
- Used for comparison/study.

3) LRU (Least Recently used) page Replacement

→ Replaces the page that has not been used for longest time

→ uses a stack or timestamp to track recent use.

Advantages:

- Approximates OPT.
- Good performance in practice.

Disadvantages

- High overhead to keep track of usage order

(9) Describe paging and segmentation schemes with neat diagrams?

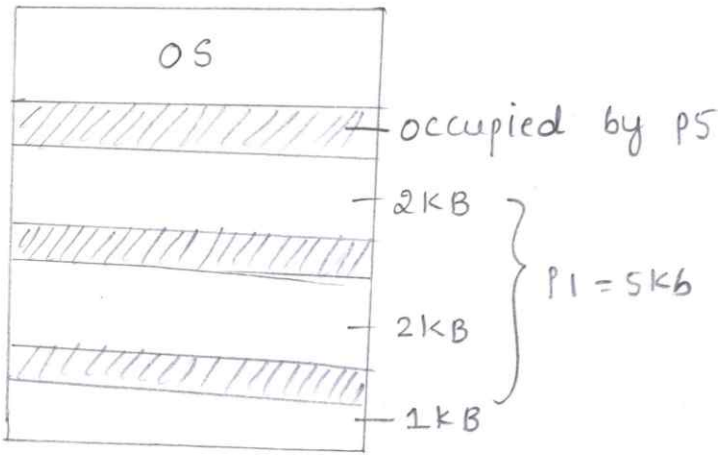
Paging:

Paging is a non-contiguous memory allocation technique in which logical memory of a process is divided into fixed size blocks called pages, and physical memory is divided into frames of the same size. A page can be loaded into frames, removing external fragmentation.

How paging works

- Logical Address space → divided into pages
- physical memory → divided into frames
- A page table maps each page to a corresponding frame.
- Logical address = page number + page offset.

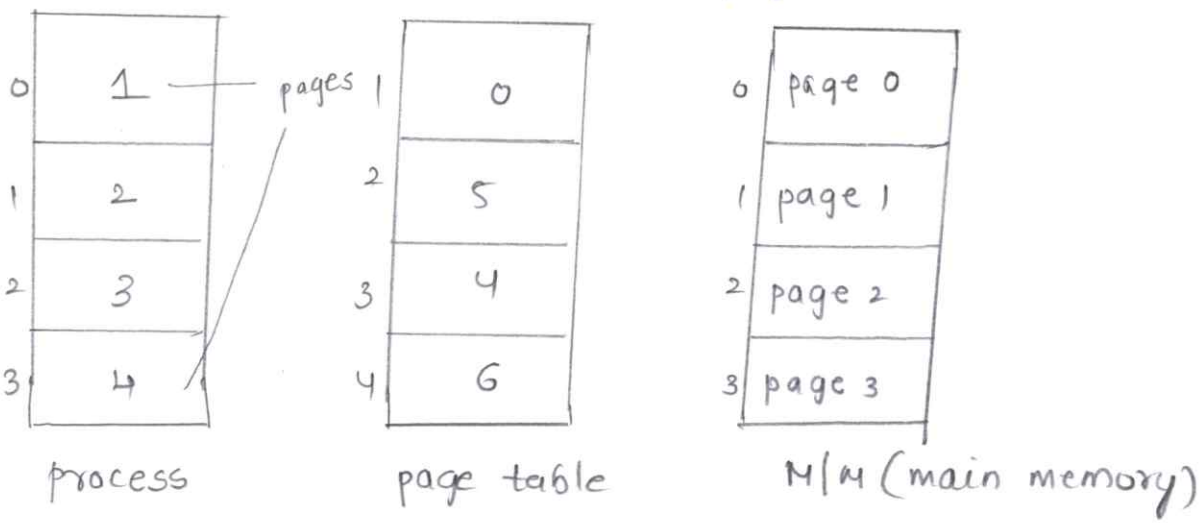
Eg: P_1 request for 5kb of memory at that time. Now P1 will allocate memory in 3 different places where memory is available



Paging Implementation:

→ In paging main memory is divided into no. of Partitions called frames.

→ Here the process is divided into no. of partitions those partitions are called pages.



Note:

→ The page size and frame size must be equal

→ The page table consists of page numbers which

are stored in main memory.

→ whenever CPU execute the process the pages which are in the secondary memory are loaded into the main memory.

Paging Hardware

→ when ever CPU executes page instructions at that time it generates a logical address.

→ Generally the logic address is divided into 2 parts

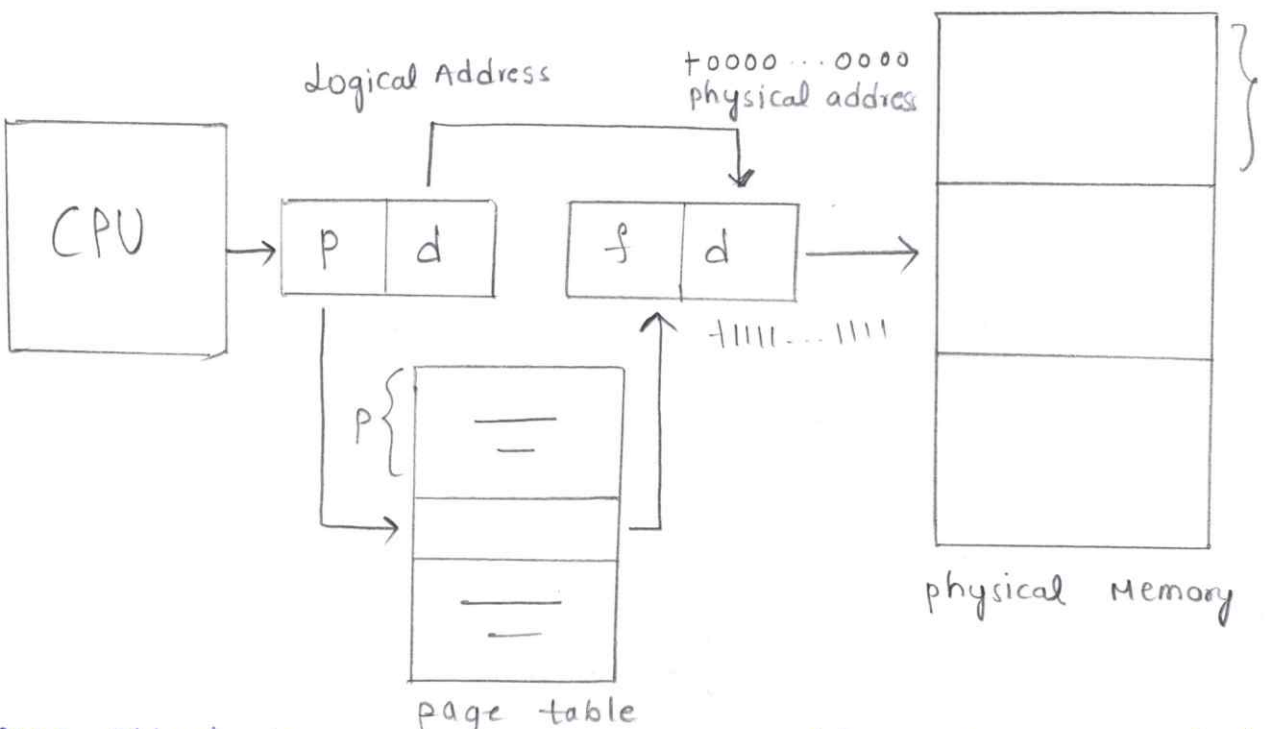
1) page no

2) page displacement / page offset [D]

1) page no :- consider of pages which are stored in main memory.

2) page displacement:

which tells which pages are accessed frequently



Page offset is nothing but the location of page which is available in physical Memory.

Segmentation

→ It is another type of non contiguous memory management technique.

→ Segmentation is a method in which the process is divided into some parts & those parts are called segments.

→ In paging, there is no modular structure. It means same method may reside with multiple process. But in segmentation the modules are divided into group of segments.

Eg:
```
Void main()
{ }
Void add()
{ }
Void sub()
{ }
```

→ The above program is divided into segments according to user.

→ Here each segment size is not same it means the segment can be various size

→ whenever CPU generate a logical address. It consists of 2 parts.

1) Segment no: It is a number of bits required to represent a segment. It is used as a index.

2) Segment offset: It is the no. of bits required to access the particular segment.

→ In segmentation there is one table which is called segment table. segment table stores the information about all the segments of process of 2 parts.

1) Base
2) segment size

1) Base address: It refers to the starting address

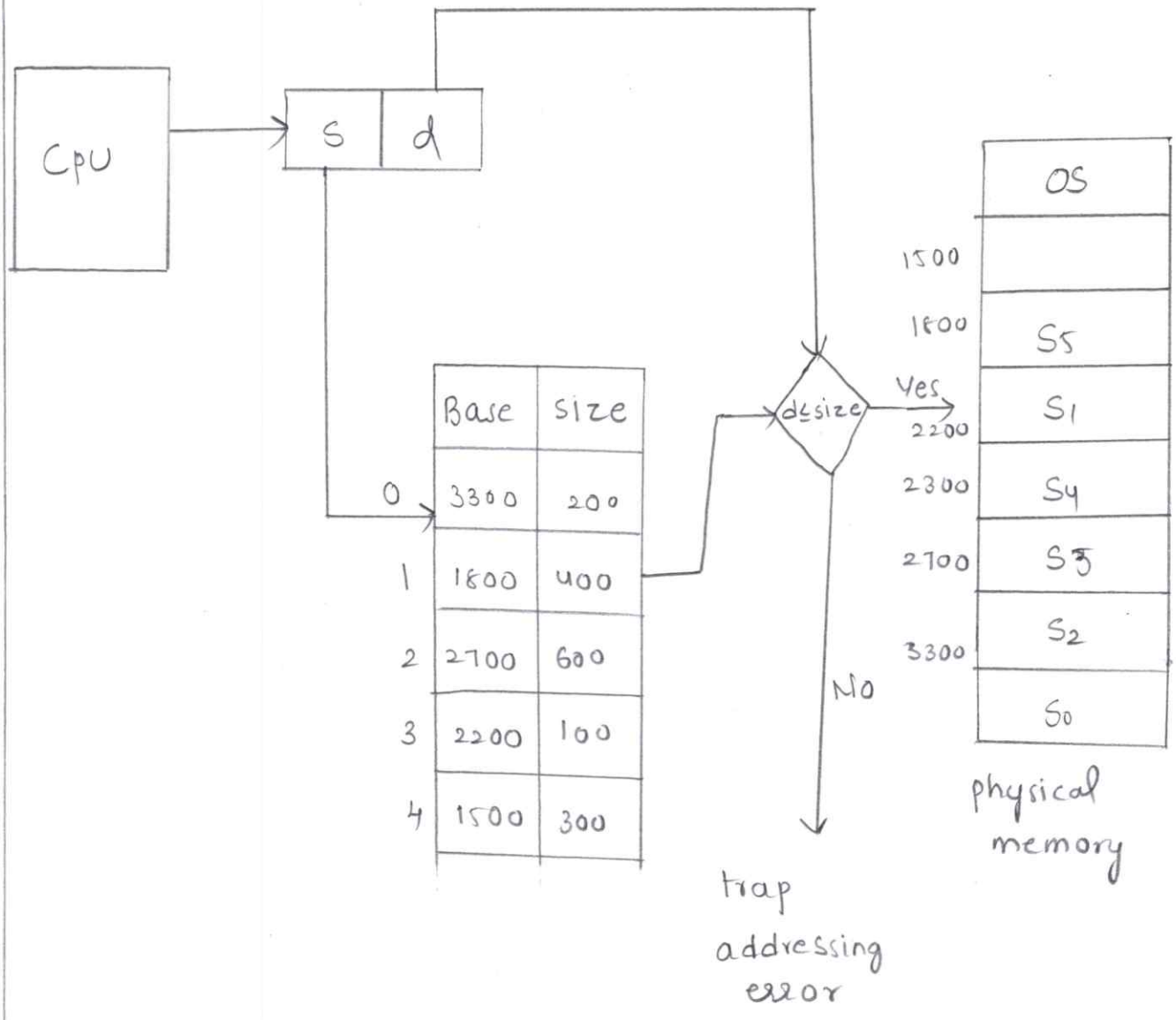
2) segment size: It refers to the segment size.

→ Whenever CPU dividing logical address, what ever the displacement we will get that should be less than segment size then only the particular segment is going to be allocated to memory

→ If it is (d) not ≤ segment size [d ≤ size] then it is going to be interrupted.

Advantages:

→ There is no internal fragmentation in segmentation.


CPU
S
d
Base
size
0
3300
200
1
1800
400
2
2700
600
3
2200
100
4
1500
300
d≤size
Yes
No
trap
addressing
error
OS
1500
1800
S5
S1
2200
2300
S4
2700
S3
S2
3300
S0
physical
memory

(10) Explain file operations and directory structure in an operating system.

File Operations:

- A File is a named collection of related information stored on secondary storage.
- OS provides several system calls to manage files.

1) Create()
→ Used to Create a new file
→ The OS allocates space in the file system and adds an entry in the directory.

2) Open()
→ Before using a file, it must be opened.
→ OS places the file entry in the open file table

3) Read()
→ Reads data from a file
→ The system call specifies the file name and a read pointer indicating from where to read next.
→ After each read, read pointer is automatically updated.

4) Write()
→ Used to write data into file
→ Uses write pointer and it is also updated same as read.
→ OS searches the directory to get location of file.

5) close()
→ Ends the file access
→ Decreases the open-count

6) delete()
→ Remove a file permanently
→ Directory is searched, space allocated to the file is released, and directory entry is deleted

7) Truncate()
→ Erases the contents of a file without deleting its attributes

8) seek()
→ Changes the current file pointer to a given position.

Directory structures

A directory stores information like file name, type, size and location.

Directory operations include
- Search for a file
- Create a file
- Delete a file
- Lists files
- Rename
- Traverse file system

1) Single Level Directory

- Simple structure

- All files are stored in one directory.
- Easy to understand and implement.

Limitations:

- All files must have unique names
- Difficult for a single user to manage many files.

2) Two-Level Directory:

- separates directory for each user
- Contains:
  - Master File Directory
  - User File Directory
  - Actual Files

Working:

- MFD indexed by username or account number.
- Each MFD entry point to user's UFD.
- User searches only his own UFD.

3) Tree-structure Directory

- Most common directory structure
- One rood directory with subdirectories and files
- Each file as unique path name
  - Absolute path → starts from root
  - Relative path → starts from current directory

Deleting a directory

Two approaches

1. Allow deletion only if empty

2. provide option to recursively delete all contents

4) Acyclic Graph Directory

- Allows sharing of files or subdirectories.
- No cycles allowed.

Implementation

1. Link → direct entry pointing to another file
2. Duplication → copies of the same directory entry

(11) Discuss file access methods and their uses?

File access methods define how data inside a file can be read or write.

1) Sequential Access

→ simplest access method.
→ Data is processed record by record in order.
→ Based on the tape model

Operations:

- ReadNext()
- writeNext()
- Some systems allow forward/backward skipping by n records.

Uses:

- Most suitable for:
  - Editors

- Compilers
- Text files
- Log files

- Ideal when data must be processed in sequence.

2) Direct Access

- File is divided into numbered blocks of fixed size.
- Allows reading/writing any block directly.
- Based on disk model, supports random access.

Operations:

- read(n)
- write(n)
- position file(n)

Uses:

- Used in:
  - Databases
  - Airlines reservation system
  - Large data systems needing fast random access
- Helps retrieve data quickly using computation.

3) Indexed Access

- An Index is created for the file, like a book index.
- Index contains pointers to file blocks
- searching is done on index first, then the block is accessed.

Uses:

- used for large data files where random access is required
- common in:
  - Price lists
  - Product catalogues
  - Databases
  - system using ISAM

Handling Large files

- If Index becomes big:
  - Use secondary indexes
  - Use master index pointing to sub-index.