OPERATING SYSTEM

OBJECTIVE

The main objective of OS

1) Convenience : An OS makes a computer more convenient to u
2) Efficiency :- An OS allows computer system resources to be used in efficient manner
3) Ability To Evolve :- It is constructed in such a way as to permit the effective development, testing & introduction of new system function

Functions Of Operating System.

1) Memory Management
2) Processor Management
3) Device Management
4) File Management
5) Security
6) Control over system performance
7) Job Accounting
8) Error Detection & Response
9) Booting the computer
10) Coordination b/w other software & user

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Outcome Of OS

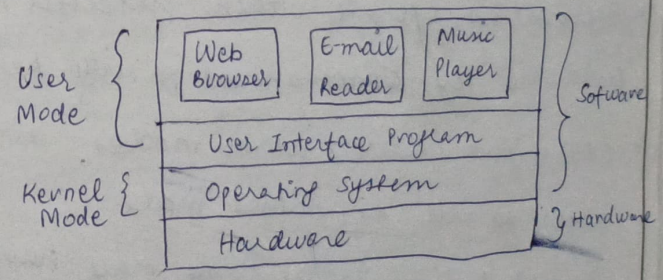
1) To learn the fundamentals of OS.
2) To learn the mechanisms of OS to handle process & threads & their communication.
3) To learn programmatically to implement simple OS mechanisms
4) To know the components & management aspects of concurrency management.

The job of operating system is to provide user programs with a better, simpler, cleaner model of computer to handle & manage all resources.

Definition:-

Operating system is a system software it works as an interface b/w user & hardware.

COMPONENTS



Main Ideas, Questions & Summary:

Library / Website Ref.:-

Modes Of Operation

1) Kernel Mode (Supervisor Mode)
2) User Mode (Protected Mode)

Kernel Mode

- This mode is used by operating systems kernel for low level tasks that need unrestricted accen to hardware, such as controlling how memory is written or erased & communicating with devices like Graphics card.
- When a computer starts it automatically runs in supervisor mode.
- When operating system passes control to another program it can place the CPU into protected mode.

Protected Mode

- Applications operate within protected mode & can use hardware by communicating with kernel.
- CPU's might have other modes similar to protected modes as well as virtual mode.
- In protected mode programs may have accen to more limited set of CPU's instructions.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

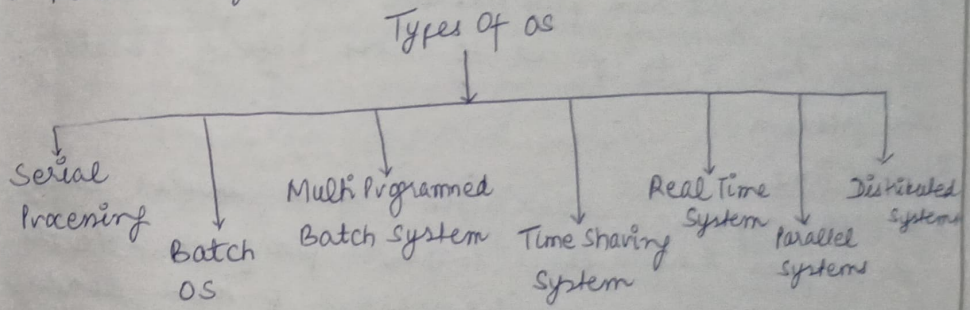
Operating System Structure

Categories of OS

Monolithic Systems | Layered Systems | Micro Kernels | client Server System | Virtual Machines | Exokernels

Types Of Operating System

- Processor Scheduling
- Memory Management
- I/O Management
- File Management



Main Ideas, Questions & Summary:

Library / Website Ref.:-

Serial Processing

- Programs in machine code were loaded via the input device (Card Reader). If an error halted the program the error condition was indicated by signals.
- If the program proceeded to normal completion, the output appeared on the Printer.
- These early systems presented two main problems

1) Scheduling
- Most installation used a hardcopy signup sheet to reserve computer time. A user may sign up for an hour but finishes his job in around 45 minutes. This would result in waster computer processing time.
- Also user might run into problem not finished in allotted time & be forced to stop before resolving the problem.

2) Setup Time
- A single program called a job could involve loading the compiler & high level language program into the memory saving the compiled program & then loading & linking together with common functions.
- Each of these steps involves tapes of card decks If error occured, user had to go back to beginning of the setup sequence
- Thus, lots of time was spent just in setting it

POORNIMA

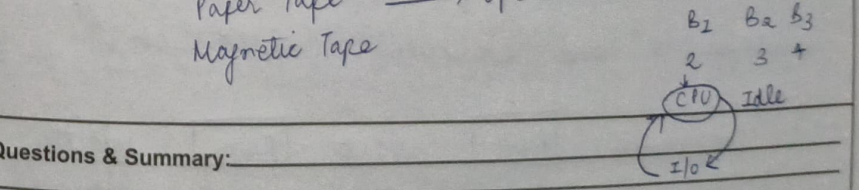
Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

program to run.

2) BATCH OPERATING SYSTEM

A single user operating system that can execute various types of jobs in batches but one after another. In batch operating system environment users submit their programs & data to operator & the operator groups the similar jobs & loads them simultaneously.

- When the execution of one program for performing similar kind of jobs is over, a new program is loaded for execution by operating system.
- Batch operating system is suitable for such applications that require long computation without user intervention.
  Eg:- Forecasting, statistical analysis.
- The main drawback of batch system is lack of interaction b/w the user & the job while the job is executing.



Main Ideas, Questions & Summary:

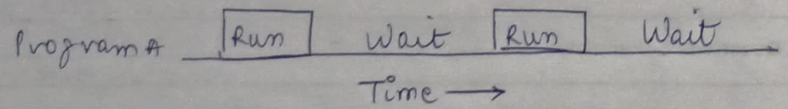
Library / Website Ref.:-

3) Single User Single Tasking Operating System.

- An operating system that allows a single user to work on a computer at a time & can execute a single job at a time is known as single user / single tasking operating system
Eg: MS-DOS is single user single tasking operating system because you can open and run only one application at a time in MS DOS.

4) MULTIPROGRAMMING BATCHED SYSTEM

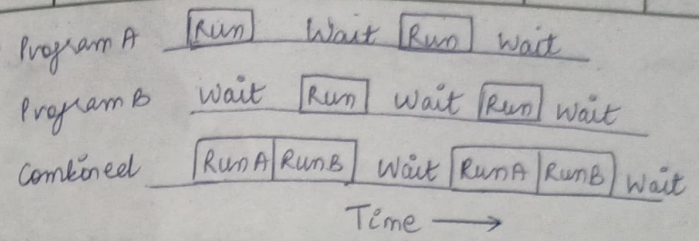
- In multiprogramming batched system, the operating system keeps several jobs in memory at a time. This set of jobs is subset of jobs kept is job pool
- The operating system picks & begin to execute one of jobs from job pool in the memory. The job may have to wait for some task such as I/O operation to complete etc.
- In non multiprogramed system, the CPU would sit idle. However, In multiprogramed system the OS simply switches to and executes another job.



1) Uni Programming

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-



b) Multiprogrammed with two programs

5) MULTITASKING / TIME SHARING

- Time sharing or multi tasking is a logical extension of multiprogramming. In this technique processor time is shared among multiple users.
- Multiple jobs are executed by CPU switching b/w them but the switches occurs frequently that users may interacts with each program while it is running.
- Time sharing systems were developed to provide interactive use of computer system at reasonable cost.

	Batch Multiprogramming	Time Sharing
1) Principal Objective	• Max^m Processor Use	• Minimize Response Time
2) Source of Directives to OS	• Job control language commands provided with job.	• Command entered at the terminal.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

REAL TIME OPERATING SYSTEM

A real time operating system is multitasking operating system intended for real time application. Such application include embedded systems industries, robots,

1) Hard RTOS
2) Soft RTOS

- A RTOS facilities the creation of real time system but does not guarantee the final result will be real time - this require correct development of software.
- The primary objective of RTOS is to provide quick response time.

PARALLEL OPERATING SYSTEM

- Most of the computers used was single processor systems, that is they have only one main CPU. However nowadays multiprocessor systems are used.
- Parallel operating system are primarily concerned with managing the resources of parallel machines.
- Advantage of building these system is to increase the throughput
- Another reason for multiprocessor system is that they increase Reliability, failure of one processor will not halt the system.

DISTRIBUTED SYSTEMS

- In modern computer system, computation is distributed among several processors. Since in this each processor has its own local memory, so the processor do not share memory or clock.
- They may include small microprocessors, workstations, minicomputers, etc. these processor are reffered by different names such as nodes, sites & so on.

Difference Between Distributed & Parallel Processing OS

Distributed OS	Parallel OS
1) These are loosly coupled system	1) These are Tightly coupled systems
2) The Processor can be placed far away from each other to cover wider geographic area	2) Not occur in parallel systems
3) In distributed operating system larger number of processor can be displayed	3) The No of processor that can be deployed is very small.
4) There is unpredictable communication delays with processor	5) In parallel they share over an interconnection network.

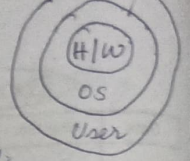
POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Operating System As Resource Manager



OS as an intermediary

- A computer is a set of resource (memory) for the movement, storage & procesing of data & for control of these functions.
- The operating system acts as manager of these resources & allocates them to specific program & users as necessary for tasks
- The OS functions in the same way as ordinary computer. Like other computer programs, the os provides instructions for processor.
- The remainder of ~~maing~~ main memory contains user program & data.
- The allocation of this resource (main memory) is controlled jointly by OS & memory management hardware.
- The OS decides when an I/O devices can be used by program in execution & controls access to and use of files.

POORNIMA

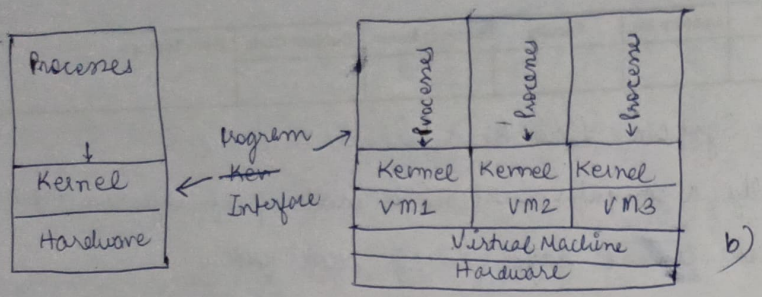
Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Operating System View As A Virtual Machine

- Conceptually, a computer system is made up of layers. The hardware is the lowest layer in all such system.
- The Kernel running at the next level uses the hardware instruction to create a set of system calls for use by outer layers.
- The system programs above the kernel are therefore able to use either system calls or hardware instructions.
- By using CPU scheduling and virtual memory techniques an operating system can create the illusion of multiple process, each executing on its own processor with its own virtual memory.
- The virtual machine concept has several advantages, like there is a complete protection of various system resources. Each virtual machine is completely isolated from all other virtual machines, so there is no security problems.

Main Ideas, Questions & Summary:

Library / Website Ref.:-



a) System Models

a) Non Virtual Machines

b) Virtual Machine

Functions of Operating system

1) Memory Management
2) Processor Management
3) Device Management
4) File Management

Process Management

- Early computer systems allowed only one program to be executed at a time & current day computer system allow multiple programs to be loaded into memory & to be executed concurrently.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Process

- Process is a program in execution.
- Process is a part of a program.
- Process is the part where logic of that particular program exists.
- Process is program in memory
- Process is an operation which takes the given instructions & performs & manipulates as per the code called execution of instructions
- ~~Program can be a~~ Process can be program.
- Process is module that execute modules concurrently. They are separate loadable modules.

Program

- Program is series of instructions to perform a particular task.
- Program is given set of process.
- In some case we may divide a problem into number of parts. At these times we write a separate logic for each part is known as process.
- Program is only set of instruction

Main Ideas, Questions & Summary:

Library / Website Ref.:-

*A program is set of instruction that are to perform designated task.

Process States

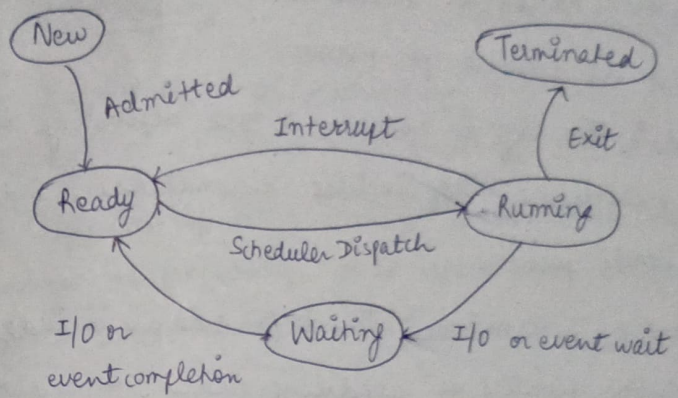
When a process comes in execution its change its state. The state of process is defined in part by current activity of that process.

1) New :- The process is being created.
2) Running :- Instructions are being executed.
3) Waiting/Blocked :- The process is waiting for some event occur. (such as an I/O completion)
4) Ready :- In this state, the process is waiting to be assigned to processor through a short term scheduler.
5) Terminated :- In this state, the process has finished execution.

These states names are arbitrary & they vary across operating system.

POORNIMA

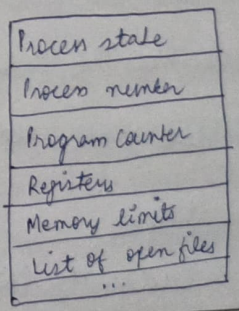
Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-



Process State Transition Diagram

Process Control Block

Each process is represented in operating system by a Process Control Block (PCB) also called as task control block.



(PCB)

Main Ideas, Questions & Summary:

Library / Website Ref.:-

- Process state: The state may be new, ready, running, waiting, halted & so on.
- Program counter: The counter indicates the address of next instruction to be executed for process.
- CPU Registers: The vary in number & type depending on computer architecture. They include accumulator, index registers, stack pointers.
- CPU Scheduling Information: This information includes a process priority, pointers to scheduling queues, & any other scheduling parameters.
- Memory Management Information: This information includes such information as value of base & limit registers, the page tables or segment tables.
- Accounting Information: The information includes the account of CPU & real time used, account numbers, job no.
- I/O status Information: This information includes the list of input & output devices allocated to process, a list of open files & so on.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Process Scheduling

- The objective of multiprogramming is to have some process running at all times to maximize CPU utilization.
- The objective of time sharing is to switch the CPU among process so frequently that users can interact with each program while it is running.

→ Process scheduling refers to the process by which process determines which job (task) should be run on the computer at which time.

- Long Term Scheduling: It determines which programs are admitted to the system for execution & when, and which ones should be exited.
- Medium Term Scheduling: It determines when process are to be suspended & resumed.
- Short Term Scheduling: It determines which of the ready process can be CPU resources & for how long.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Context Switch

- Interrupts cause the operating system to change a CPU from its current task & to run a kernel routine
- ~~Soo~~ Such operations happen frequently on general-purpose systems
- When an interrupt occurs, the system needs to save the current context of the process currently running on the CPU so that it can restore the context when its processing is done, essentially suspending the process & then resuming it.

Interprocess Communication

Processes ~~are~~ executing concurrently in operating system may be either independent processes.

- Independent Process:- They can't affect or be affected by other processes executing in the system.
- Cooperating Process:- They can affect or be affected by the other processes executing the system.
- Any process that shares data with other processes is cooperating process.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

In cooperating process we need interprocess communication

Reasons For Providing An Environment that allow Process Cooperation

1) Information Sharing
2) Computation Speedup
3) Modularity
4) Convenience

Cooperating processes require an interprocess communication mechanism that will allow them to exchange data & information

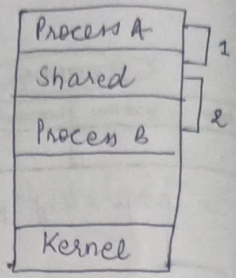
There are two fundamental models of interprocess communication

1) Shared Memory
2) Message Passing

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Shared Memory Model



- Interprocess communication using shared memory requires communicating processes to establish a region of shared memory
- Processes use shared memory creates & shared memory attaches system calls to create & gain access to regions of memory owned by other processes.
- Two or more processes can exchange information by reading and writing data in shared areas.

Strengths

- Shared memory communication is faster than the message passing model when the processes are on the same machine.

Weakness

- Different processes need to ensure that they are not writing to same location simultaneously.
- Processes that communicate using shared memory need to address problems of memory protection & synchronization.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Message Passing Model

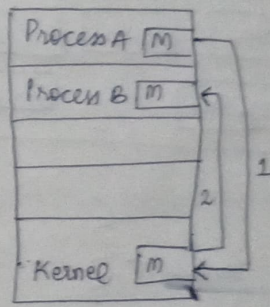
- In message passing model, the communicating processes exchange messages with one another to transfer information.
- Messages can be exchanged b/w the process either directly or indirectly through a common mailbox.
- Message passing is useful for exchanging smaller amounts of data, because no conflicts need to be avoided.

Strengths

- Easier to implement
- Best suited for smaller amount of data

Weakness

- Only suitable for exchange of small amount of data.
- Communication using message passing is slower than shared memory.



Message Passing

Main Ideas, Questions & Summary:

Library / Website Ref.:-

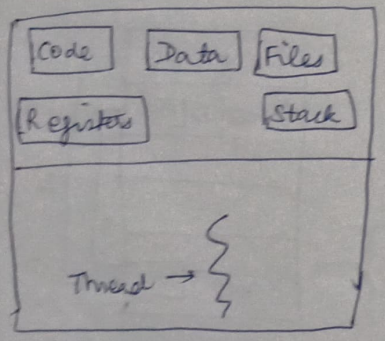
Program under execution : Process
Basic unit of execution : Thread

Threads

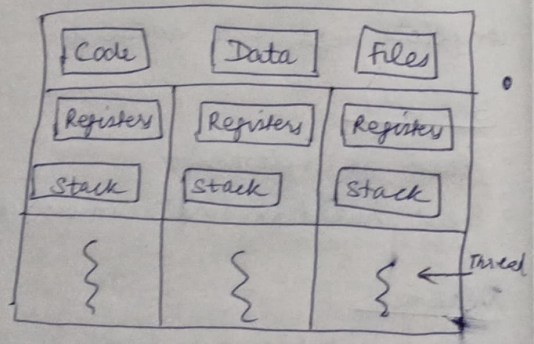
Thread is basic unit of execution or basic unit of CPU utilization

It comprises

- A Thread ID
- A program counter
- A register set
- A stack



Single Threaded Process



Multi Thread Process

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Benefits of Multi Threaded Programming

1) Responsiveness
2) Resource Sharing
3) Economy
4) Utilization of multiprocessor architectures

Types of Threads

1) User Threads :- Supported above the kernel and are managed without kernel support

2) Kernel Threads :- Supported & Managed directly by operating system

Ultimately there must exist a relationship b/w user & kernel threads

Three ways of establishing this relationship

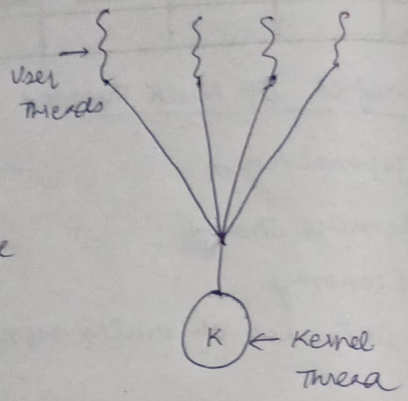
1) Many to One Model
2) One to One Model
3) Many to Many Model

Main Ideas, Questions & Summary:

Library / Website Ref.:-

1) Many to One Model

- Maps many user level threads to one kernel thread
- Thread management is done by thread library in user space so it is efficient

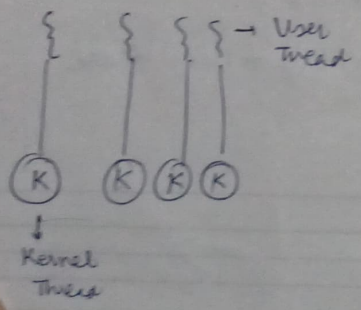


~~Library~~

Limitation

- The entire process will block if a thread makes a blocking system call.
- Because only one thread can access the kernel at a time multiple threads are unable to run in parallel on multiprocessors.

2) One to One Model



- Maps each user thread to kernel thread
- Provides more concurrency than many to one model by allowing another thread to run when a thread makes a blocking system call

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

- Also allows multiple threads to run in parallel on multiprocessors

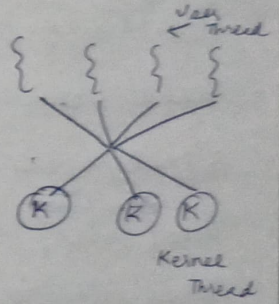
Limitation

- Creating a user thread requires creating the corresponding kernel thread.

! Creating kernel threads can burden performance of an application

3) Many To Many Model

- Multiplexes many user level threads to a smaller or equal number of kernel threads
- The number of kernel threads may be specific to either a particular application or a particular machine



- Developers can create as many user threads as necessary & the corresponding kernel threads can run in parallel on a multiprocessor.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Fork System Call

fork (): The fork() system call is used to create a separate, duplicate process.

exec(): When an exec() system call is invoked, the program specified in parameter to exec() will replace the entire process including all threads

Threading Issues

- Semantics of fork() & exec() system calls
- Thread cancellation.
- Signal handling
- Thread pools
- Thread specific data

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

- Process v/s Threads

Process	Threads
1) System call involved in process.	1) There is no system call involved
2) OS treats different processes differently	2) All user level threads treated as single task for OS.
3) Different process have different copies of data, files, code.	3) Threads share same copy of code and data.
4) Context switching is slower	4) Context switching is faster
5) Blocking a process will not block another	5) Blocking a thread will block entire process
6) Independent	6) Interdependent.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

· User Level Thread and Kernel Level Thread

1) User thread are implement by users	1) Kernel threads are implem-ented by OS.
2) OS doesn't recognized user level thread	2) Kernel thread are recognized by OS.
3) Implementation of user threads is easy	3) Implementation of Kernel thread is complicated.
4) Context switch time is less	4) Context switch time is more.
5) Context switch requires no hardware support	5) Hardware support is needed.
Example :- Java thread, POSIX thread	Example: Window Solaris.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

CPU Scheduling

- The CPU scheduler is an important component of operating system Process must be properly scheduled, or else the system will make inefficient use of its resources.
- Different operating systems have different scheduling requirements.
  For Example:- A supercomputer aims to finish as many as many jobs as it can in minimum amount of time but an interactive multi user system such as Windows terminal server aims to rapidly switch CPU between each user.

→ Difference b/w Premptive Scheduling & Non Premptive Scheduling

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Preemptive Scheduling

1) It allows process to be interrupted in middle of its execution taking the CPU way & allocating to another process

2) Complex to implement

3) Costly

4) High overhead

5) Process switches form running state to ready & waiting state to ready state.

Non Preemptive Scheduling

1) It ensures that a process relinquishes control of the CPU only when its finish its current CPU burst.

2) Simple to implement

3) Cost is less

4) Low overhead

5) Process switches from running state to waiting state and process terminates.

CRITICAL SECTION PROBLEM

Solution to critical section problem must meet these three condition

1) Mutual exclusion :- If process Pi is executing in its critical section no other process is executing in its critical section.

2) Progress :- If no process is executing in its critical section & there exists some processes that wish to enter critical section ~~problem~~. then only those processes that are not executing in remainder section

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

can participate in decision of which will enter its critical section next.

1) If no process is in critical section, can decide quickly who enters.

2) Only one process can enter critical section so in practice others are put in queue.

3) Bounded Waiting

There must exist a bound on number of times, that other processes are allowed to enter their critical section after a process has made a request to enter its critical section & before that request is granted.

RACE CONDITION

In some operating systems, processes that are working together may share some common storage that each one can read and write. The shared storage may be in main memory or it may be a shared file; the location of shared memory does not change the nature of communication or

Main Ideas, Questions & Summary:

Library / Website Ref.:-

the problems that arise

Situation where two or more process are reading or writing some shared data & final result depends on who runs precisely when, are called race condition.

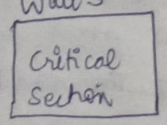
• Debugging programs contains race conditions is no fun at all. The results of most test runs are fine, but once in a rare while something weird & unexplained happens

Fair Share Scheduling

It is scheduling strategy for computer operating system in which CPU usage is equally distributed among systems users or groups as opposed to equal distributions among process.

Critical section

The key to preventing trouble involving shared storage is find some way to prohibit more than one proces from reading & writing the shared data simultaneously. The part of program where shared memory is accessed is called Critical Sectio



← enter

← exit

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Semaphore

A semaphore is the most basic form is a protected integer variable that can facilitate & restrict access to shared resources in multi processing environment.

The two most basic kinds of semaphores are counting semaphores & binary semaphores.

1) Counting semaphores represent multiple resources.

2) Binary semaphore as the name implies, represent two states (generally 0 or 1; locked or unlocked)

• A semaphore can only be accessed using

- wait() & signal operations

→ wait is called when a proces wants access to a resource.

If semaphore is greater than 0 then proces can take resource

If semaphore is 0 then the resource is not available

→ signal() is called when a proces is done using a resource

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Semaphores can be used to solve reader writer problem.

- Semphore is signaling mechanism

Mutex

Mutex is program object that allows multiple program threads to share the same resource. such as file access. but not simultaneously.

- When a program is started a mutex is created with a unique name After this stage any thread that needs the resource must lock the mutex to set to unlock when the data is no longer needed or routine is finished.
- Mutex is locking mechanism used to synchronize access to resource. Only one task can acquire the mutex. Only owner can release the lock.

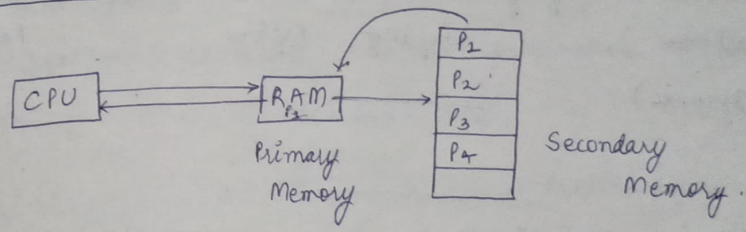
POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

MEMORY MANAGEMENT

Chapter - 2

Goal - Efficient Utilization Of Memory



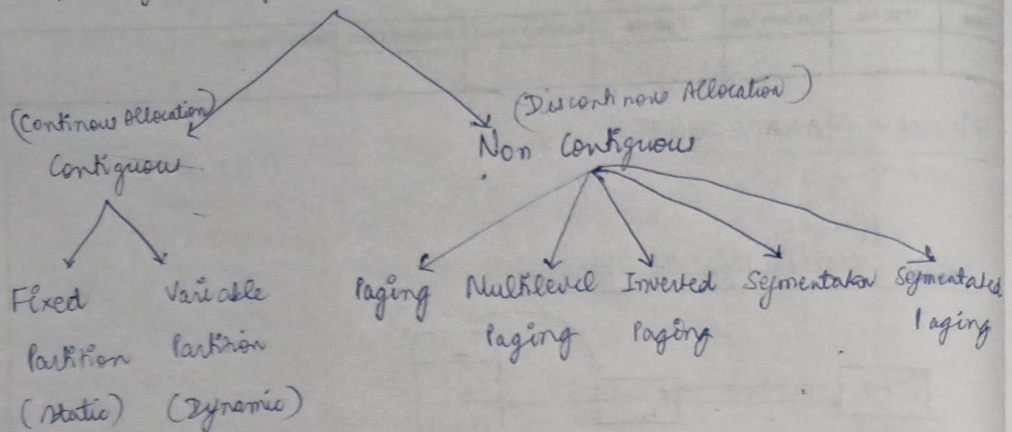
Degree of Multiprogramming.

Whenever ~~we're~~ we are keeping programs or process in secondary memory. load multi programs in ram so can CPU can execute all programs in efficient manner. so the utilization is increasing

Main Ideas, Questions & Summary:

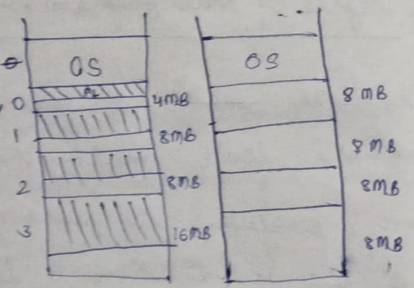
Library / Website Ref.:-

Memory Management Techniques



Fixed Partitioning (static Partion)

- Number of partition are fixed
- Size of each partition may or may not same.
- Contigous allocation or spanning is not allowed



$P_2 = 2MB$

1) Internal Fragmentation

In this example it is shown P_2 is given 2MB is allocated in memory 2MB memory is waste ~~is~~ the wastage of memory is known as Internal fragmentation

Limit in Process size

Spanning : Atleast part of partition atleast part of partition not available next to

POORNIMA 20

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

3) Limitation on degree of multiprogramming.

4) External Fragmentation

Although we are having availability of memory in different-different slots & combination of all the space is equivalent to available size.

Variable Partitioning (Dynamic Partitioning)

Whenever the process is coming into RAM only then we are allocating to space.

1) There is no chance of internal fragmentation
2) No limitation of number of process
3) No limitation of process size
4) Variable partioning suffer external fragmentation
5) Allocation and Deallocation is complex.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

→ Advantage :- Simple / Fast

- First Bit
  Allocate the first hole that is big enough.

- Next Fit :- Same as first fit but start search always from last allocated hole.
  Advantage :- Fast
  → Advantage: Internal Fragmentation is less → Disadvantage :- Very slow

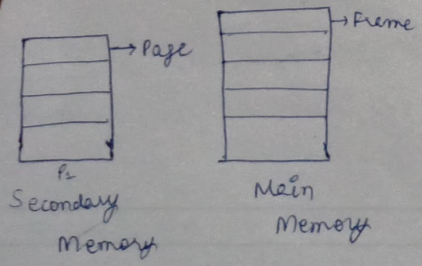
- Best fit :- Allocate the smallest hole that is big enough

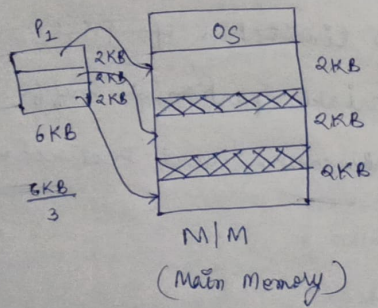
- Worst fit :- Allocate the largest hole.
  Disadvantage : slow

Non Contiguous Memory Allocation

- We're allocating memory in ~~non continous~~ to different different processes in non consecutive manner

- ~~It~~ Here process can be spann.



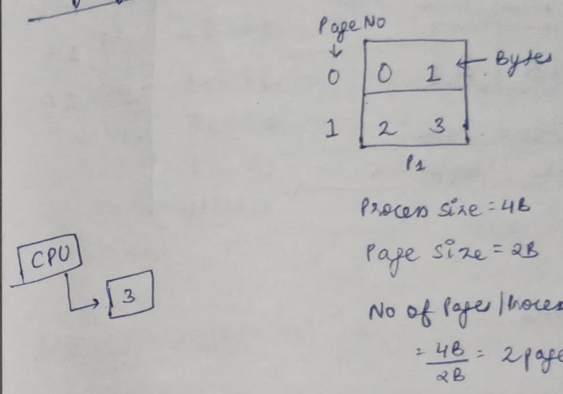


** Page Size = frame size

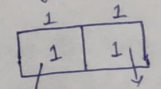
POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Paging

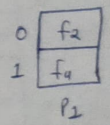


Logical Address

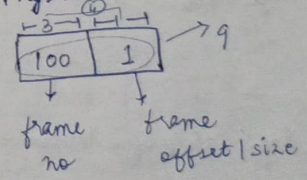


LA → PA

Page Table



Physical Address



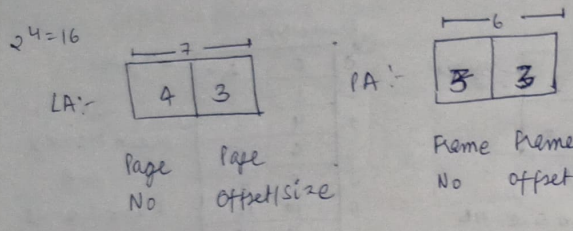
Main Ideas, Questions & Summary:

Library / Website Ref.:-

Consider a system which has

LA = 7 bits, PA = 6 bits, page size = 8 words / byte

then calculate no of pages & no of frames -



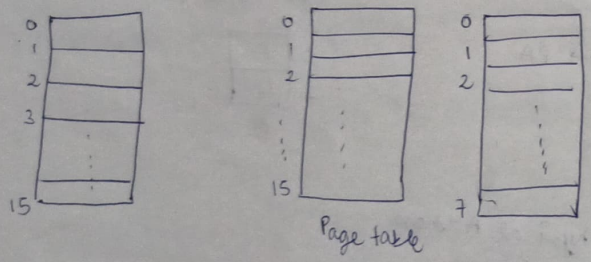
$2^1 = 2$	$2^8 = 256$
$2^2 = 4$	$2^9 = 512$
$2^3 = 8$	$2^{10} = 1K$
$2^4 = 16$	$2^{20} = 1M$
$2^5 = 32$	$2^{30} = 1G$
$2^6 = 64$	$2^{40} = 1T$
$2^7 = 128$	
0 - 127	
1111111	

No of pages

$2^4 = 16$

No of frames

$2^3 = 8$

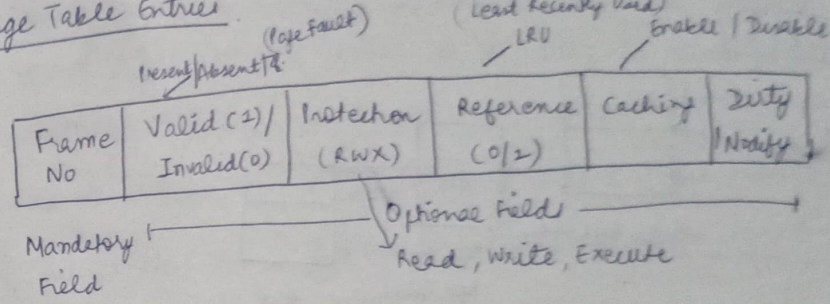


No of enteries in a page Table of a Process = No of Pages in Process

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Page Table Entries



MULTILEVEL PAGING OR

Two Level Page Table Scheme

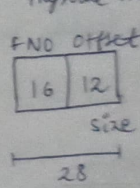
Example :-

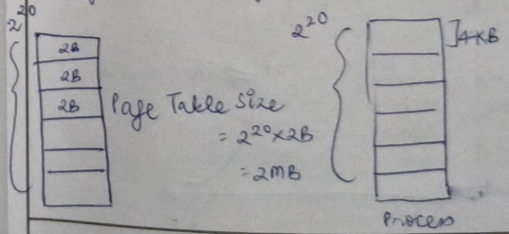
Physical Address Space = 256MB $\rightarrow 2^{28}$B

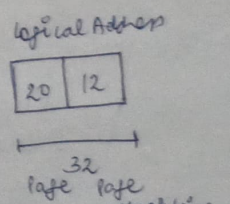
Logical Address Space = 4GB

Frame Size = 4KB = 12

Page Table Entry = 2B







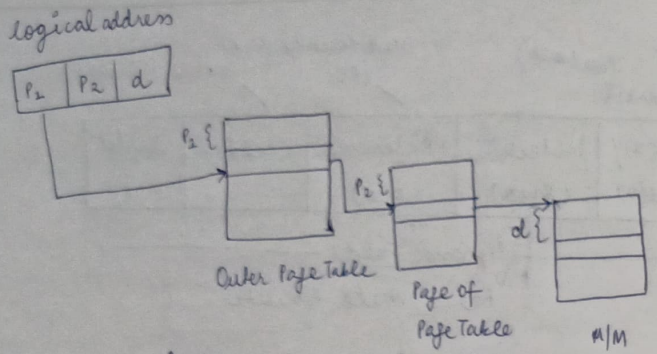
$\frac{2MB}{4KB} = \frac{2^{21}}{2^{12}}$

$= 2^9$

$= 512$

Main Ideas, Questions & Summary:

Library / Website Ref.:-



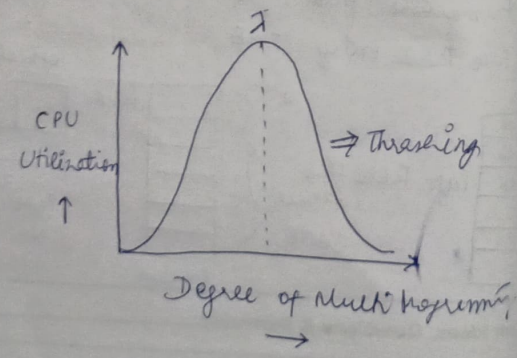
Address Translation Scheme

Inverted Paging

1) Each process has its own page table.

2) Page Table will be in main memory.

Thrashing



POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

Thrashing is a condition or situation when the system is spending a major portion of its time in servicing the page faults. and Process is allocated too few frames, then there will be too many and too frequent page faults.

- This causes the performance of computer to degrade or collapse

Causes Of Thrashing

- It affects the performance of execution
- Process is spending more time in paging or swapping activities rather than its execution
- CPU is so much busy in swapping it can't respond to user program as much as it required

Overcome Thrashing

- Working set model prevents thrashing in OS while keeping the degree of multiprogramming as high as possible
- In this way it optimizes CPU utilization

Main Ideas, Questions & Summary:

Library / Website Ref.:-

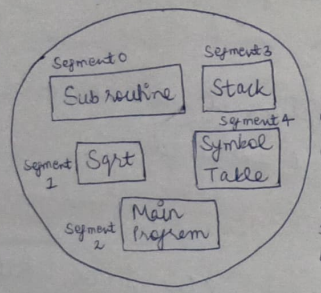
Segmentation

Segmentation is a technique to break memory into logical pieces where each piece represents a group of related information.

For Eg:- Data segments or code segments for each process.
Data segments for operating system and so on.

→ Segmentation can be implemented using or without using paging. Unlike paging segment are having varying size & thus eliminates internal fragmentation.

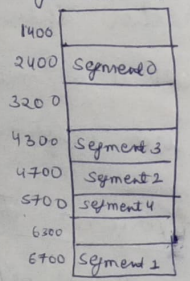
• External fragmentation still exists but to lesser extent.



Segment Table

	Limit	Base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

Physical memory



Example of Segmentation

Segmentation with Paging

Segmentation can be combined with paging to provide the efficiency of paging with protection and sharing

POORNIMA

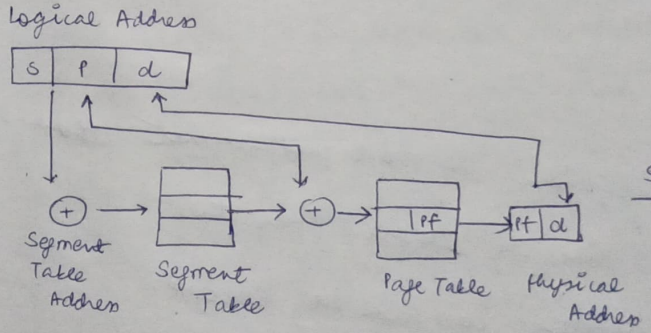
Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-

capabilities of segmentation

As with simple segmentation the logical address specifies the segment number and the offset within the segment However when paging is added, the segment offset is further divided into page number and page offset.

~~segment offset is further divided into page number and page offset~~

The segment table entry contains the address of the segment page table. The hardware adds the logical address' page number bits to page table address to locate page table entry. The physical address is formed by appending the page offset to page frame number specified in page table entry.



Segmentation with Paging

Main Ideas, Questions & Summary:

Library / Website Ref.:-

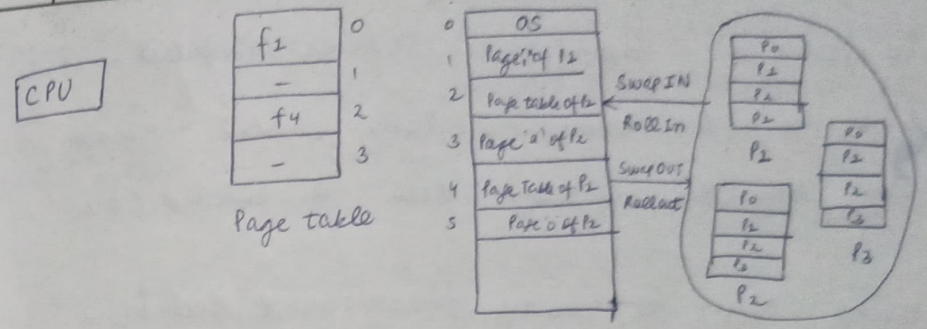
Segmentation	Paging
	Paging is hidden
1) Programmer is aware of segmentation	
	Paging maintain one address space
2) Segmentation maintains multiple address spaces per proces	
	This is hard with paging
3) Segmentation allows procedures and data to be separately protected	
	Pages are of fixed size
4) Segmentation easily permits table whose size varies	
	This is hard with paging
5) Segmentation facilities sharing of procedures b/w process	

Virtual Memory

It provides an illusion to the programmer that a proces whose size is larger than the size of main memory can also be execute.

POORNIMA

Date	Unit No.	Lecture No.	Faculty	Subject Name	Subject Code	Main Topics:-



- As its name suggests doesn't physically exist on a memory chip.
- It is an optimization technique and is implemented by operating system in order to give an application program that impression that it has more memory than actually exists.
- Virtual memory is implemented by various virtual memory such as Windows, Mac os x and Linux
- It is a technique that allows the execution of process which are not completely available in that programs can be larger than physical memory.

Main Ideas, Questions & Summary:

Library / Website Ref.:-

Use of Virtual Memory

- It is an old concept before computers had cache they had virtual memory. For a long time virtual memory only appears on mainframes. Personal computers in 1980s did not use virtual memory. Initially virtual memory mean the idea of using disk to extend RAM

→ Paging :- Paging is memory management technique in which the memory is divided into fixed size pages. Paging is used for faster access to data. When a program needs a page it is available in main memory as the OS copies a certain number of page from your storage device to main memory

Demand Paging

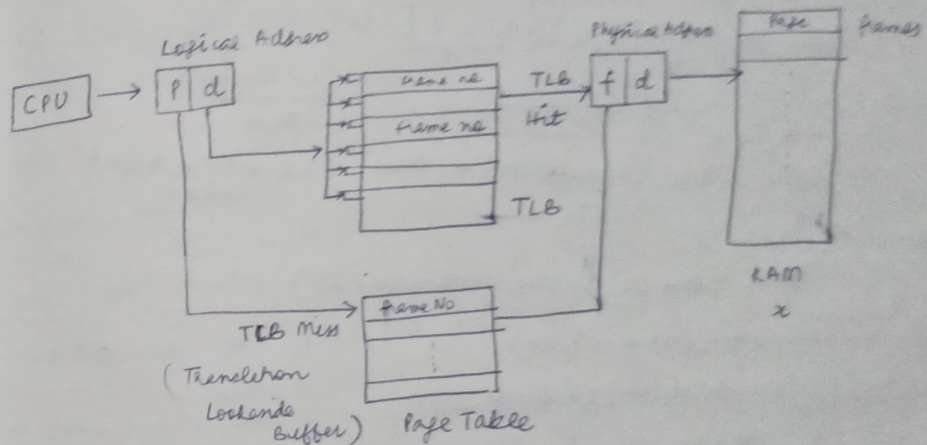
→ It commonly used in virtual memory systems. With demand paged virtual memory, pages are only loaded when they are demanded during program execution.

→ A demand paging system is similar to paging system with swapping where processes reside in

CPU → (PT) frame → (Main)

secondary memory. when we want to execute a process we swap it into memory. Rather than swapping the entire process into memory.

Translation Lookaside Buffer (TLB)



EMAT = (TLB + x) + Miss (TLB + x + x)

(Effective Memory Access Time)

Main Ideas, Questions & Summary:

Library / Website Ref.:-