Chapter 1

Date: 7/7

An OS a program that acts as an interface between the users and the hardware. controls the computer program.

[bridge between user and hardware.]

Computer system structure

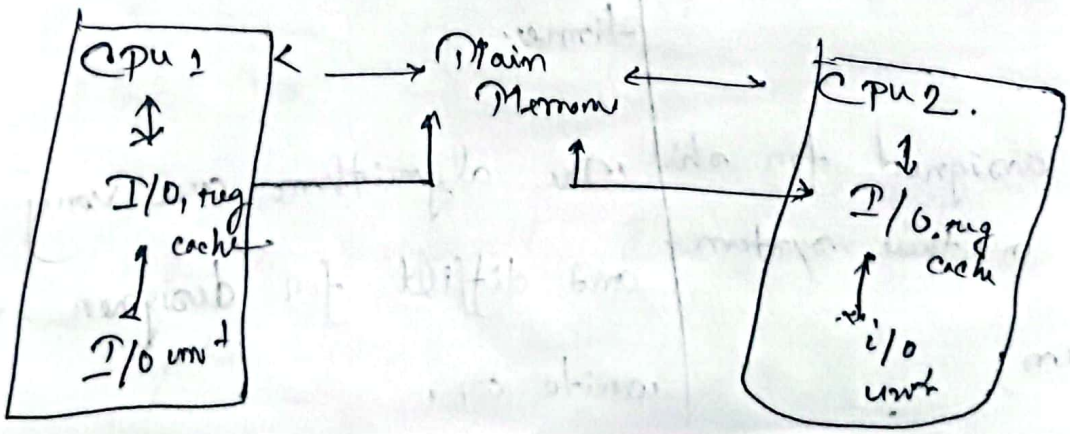
[USER ⟷ Application Program ⟷ OS ⟷ Computer hardware.]

Types of operating system.

- Multiprocessing OS: Parallel computing is achieved.

It is more than one cpu present in the system and execute more than one process at the same time.

Example: UNIX OS.



Advantage

Advantage	Dis Advantage
• Increase throughput of the ~~system~~ system • It has several processor, so if one is fails, we can ~~produced~~ proceed with another.	• Due to multiple cpu it can be complex. • Some how Difficult to understand.

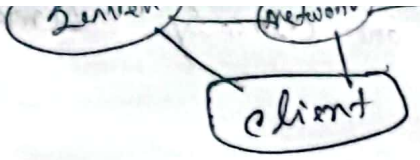
Real Time OS:

each job carries a certain ~~ded~~ deadline within which the job is supposed to be completed.

(time কবি ২০-২৫ বাইশে under)

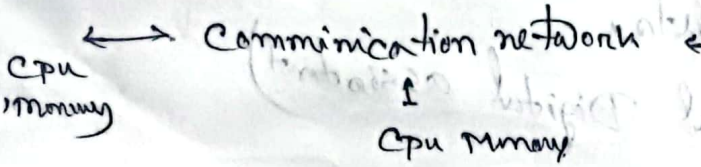
Example : Missile, traffic control, robots.

ADvantage	Dis Advantage
• Maximum utilization of system • The time assigned for shifting tasks in these system is very less.	• very few tasks at the same times. • The algorithm are very complex and diffilt for designer to write on.



Distributed OS
(LAN, MAN, WAN)

Communicate with each other using a shared communication network. (Independent system - Group).



Advantage	Disadvantage
• Failure of one will not affect the other network communication.	• Failure of network stop the entire communication.
• Many system can be easily added to the network.	• It is very expensive.

Distributed OS 2 type → Client server (Unix / Linux) / Peer to peer.

Client server → large number of computer communicate with shared server one local area network.

Peer to peer → Contains large number of processors that do not share memory or clocks.

Clustered OS:

. It is a like as parallel system because both system use multiple CPU.

(It is composed 2 or more individual computer system joined together.

share storage and are closely linked through local area network.

Handheld OS: Its present in all handheld devices like smartphone and tablets.
(Personal Digital assistant)

Goals of OS:

- Execute User programs and solving user problems easier.
- Make the Computer system convenient to use.
- Use the Computer hardware in an efficient manner

Storage device hierarchy

register
cache
Main memory } Primary Storage → access time faster → capacity smaller

optical disk
Magnetic tapes } access time slower → Storage Capacity larger → tertiary storage

Process

. Process একটা job.

a process is a program in Execution. (Sequential ভাবে)

Process management

The activities involved in managing the execution of multiple processes in an OS.

Layout of process (Program → process হয়ে যখন executable file memory তে load হয়)

max

int *value → Stack. - contain temporary data.
int i; return address, local variable.

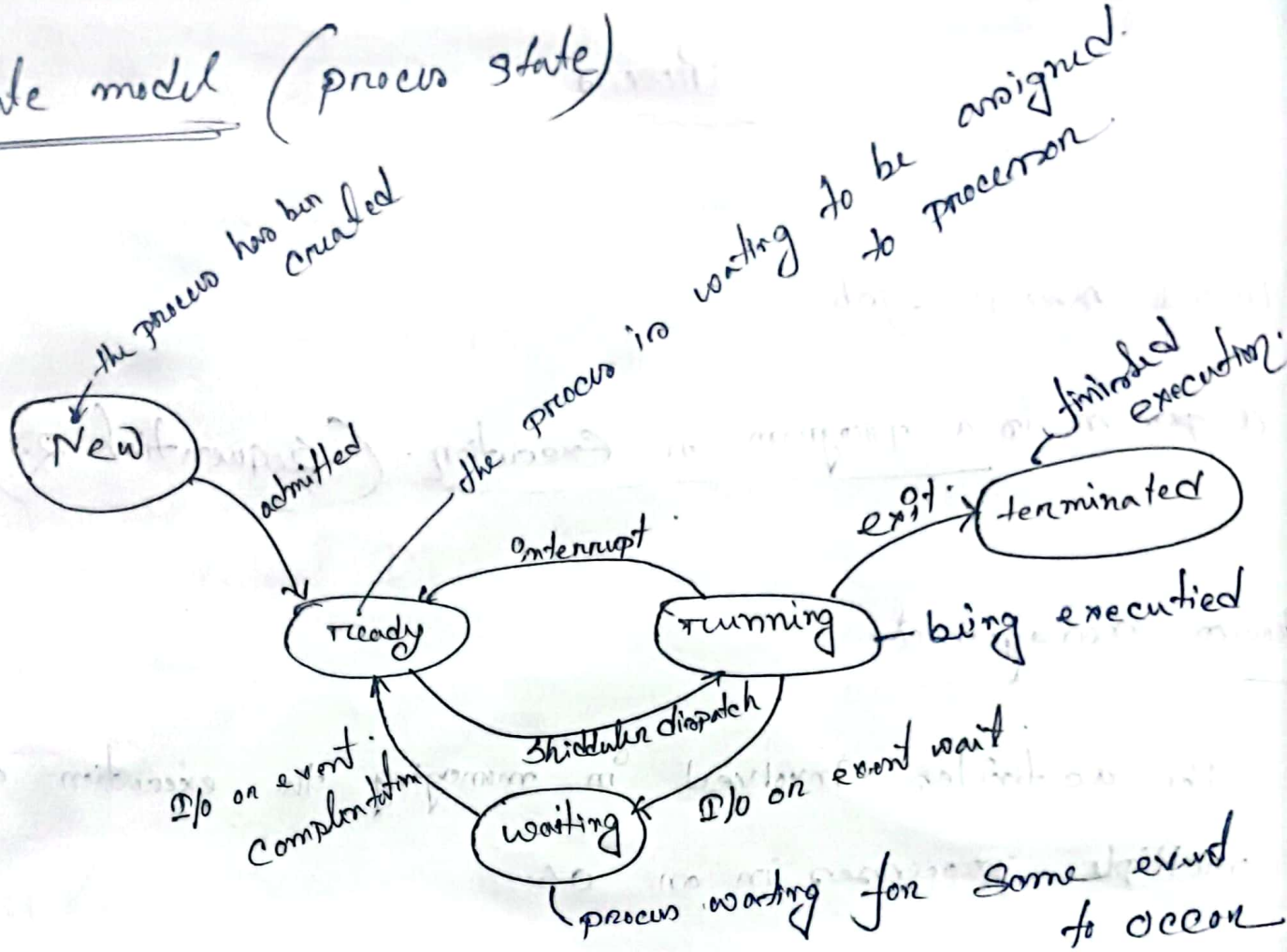
value = int * 5 → heap — dinamically memory allocated during run time

int x, y → data — contain the global value.
int x = 15

text — program code.

0

Five state model (process state)



Process Scheduling : It is scheduler process of different states like ready, waiting and running.

why should process be scheduled ?

- Ensure that process execute efficiency and have reduce wait time. The goal of process shedulling policies is to use cpu resources wisely, increase throughput reduce wait time, increase

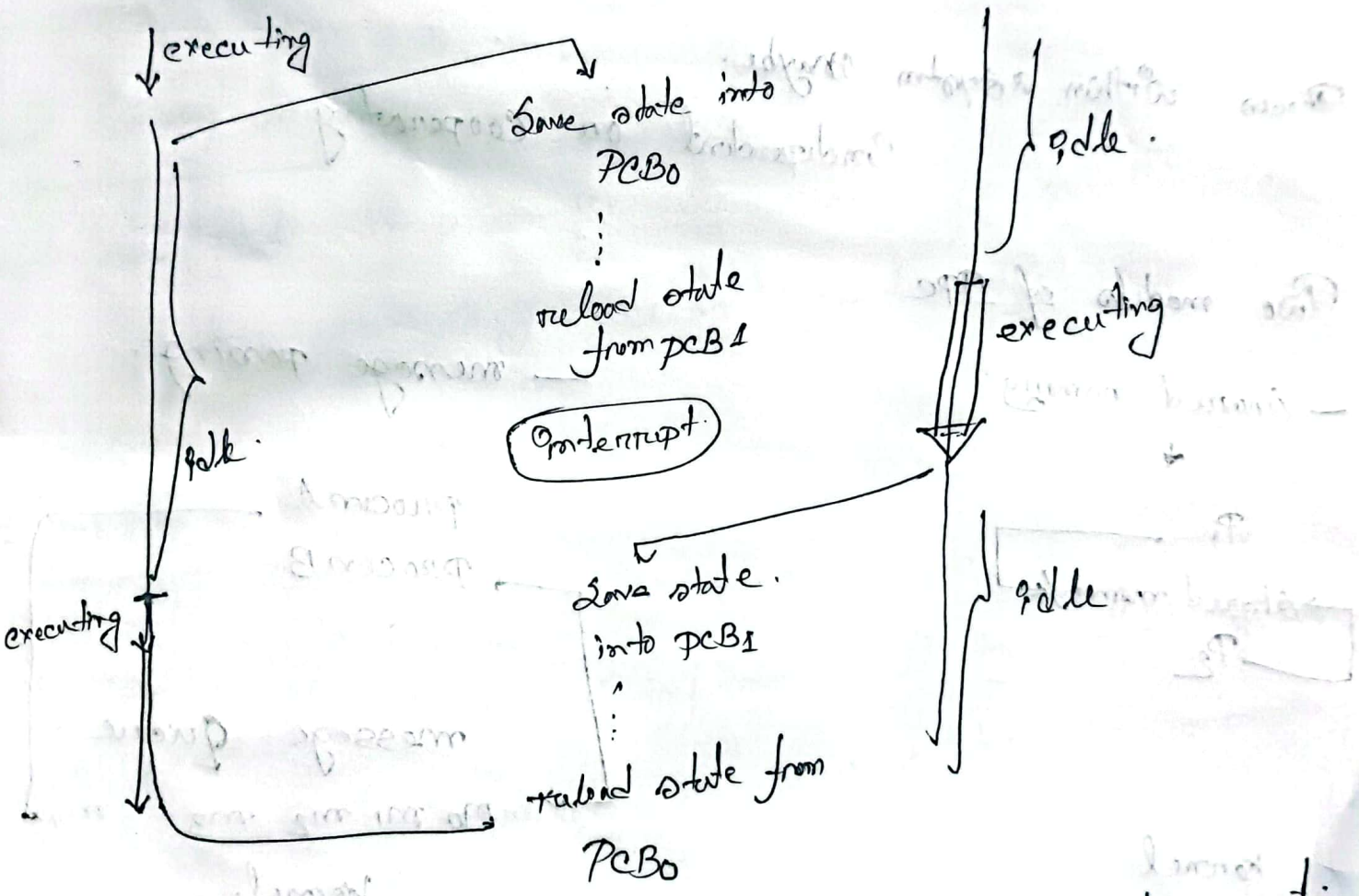
Context switching

CPU switch process to process

- Store off the registers

Process P_0

Process P_1



when CPU switches to another process, the system must save the state of the old process and load the saved state for the new process via a context switch.

PCB — program control Block

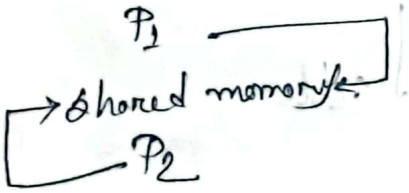
Interprocess Communication (IPC)

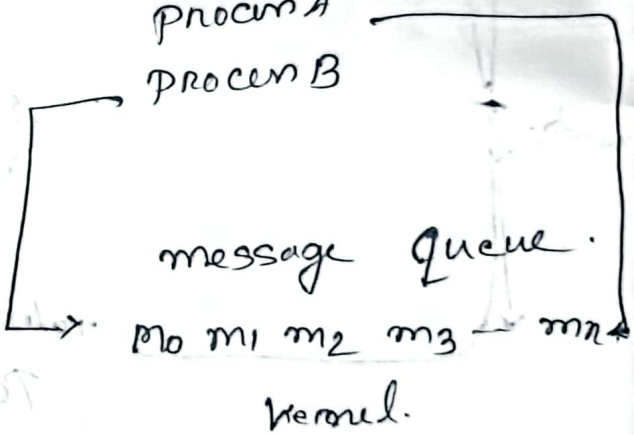
IPC is the mechanism provided by the os that allows processes to communicate with each other.

Process within a system may be independent or cooperating.

Two models of IPC

- shared memory
- message passing.





Why Ipc occur : IPC helps exchange data between multiple threads in one or more process or program. Doesn't matter weather process is running on single or multiple Computer

IPC - Message passing

Mechanism for processes to communicate and to synchronize their actions.

two operation (send, receive)

Shared memory (IPC)

Mechanism that will allow the user procen to synchronize their action when they access shared memory

Synchronization :

It refers to a case where the data used to communicate between processors is control information. It is either given by the IPC mechanism or handled by the communicating processors

Synchronization is used to ensure that procen operate in a coordinated and mutualy agreed - upon manner. to prevent error or conflict.

Buffering

The act of storing data temporarily in the buffer is called buffering. It is used when moving data between processes within computer. It can be implemented in a fixed memory location in hardware or by using a virtual data buffer in software, pointing location in physical memory.

options include

Zero-queue has a maximum length of zero.

Bounded capacity - atmost n message can reside in it.

unBounded - length is potentially infinite.

Sender never delayed.

operation on process

System must provide mechanisms for

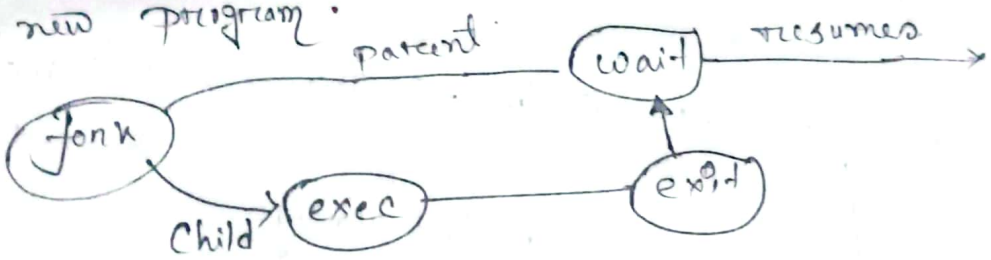
Process creation - Parent process create children processes, which in turn create other processes forming a tree.

process termination

* resource sharing
- parent and children share all resources.
- children share subset of parent's resources.

Execution >> parent and children execute concurrently.
parent waits until children terminate

* fork() system call creates process.
* exec() used after a fork() to replace process memory space with new program.



Process termination

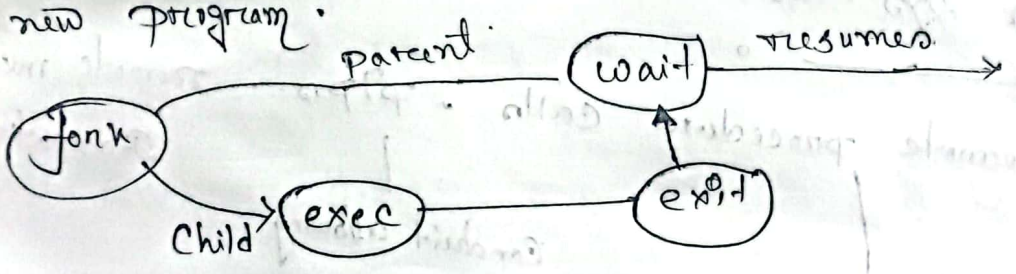
Process executes last statement and then asks the OS to delete it using exit() system call.
- returns status data from child to parent via wait()
- parent may terminate the execution of children process using the abort() system call.
- Child has exceeded allocated resources.
- Task assigned to child is no longer required.

parent terminated
- so child continue
- so on

* resource sharing
- parent and children share all resources.
- children share subset of parent's resources.

Execution >> parent and children execute concurrently.
parent waits until children terminate

* fork() system call creates process.
* exec() used after a fork() to replace process memory space with new program.



Process termination

Process executes last statement and then asks the OS to delete it using exit() system call.

- return status data from child to parent via wait().
- parent may terminate the execution of children process using the abort() system call.
  - Child has exceeded allocated resources.
  - Task assigned to child is no longer required.
  - parent terminated
  - OS does not allow child continue

* The parent process may wait for termination of a child process by using wait. The call return info. and the pid of terminated process

pid = wait (&status)

- If no parent waiting process is zombie.
- If parent terminated without invoking wait then process is orphan

Client Server Systems

- Socket → webserver
- remote procedure calls
- pipes
- remote method Invocation.

host → webserver

Client → server

conduit allowing 2 processes to Communicate

process Control block: Information associated each process.

Process state → program counter, CPU registers, CPU scheduling, memory management, accounting information, I/O status info.

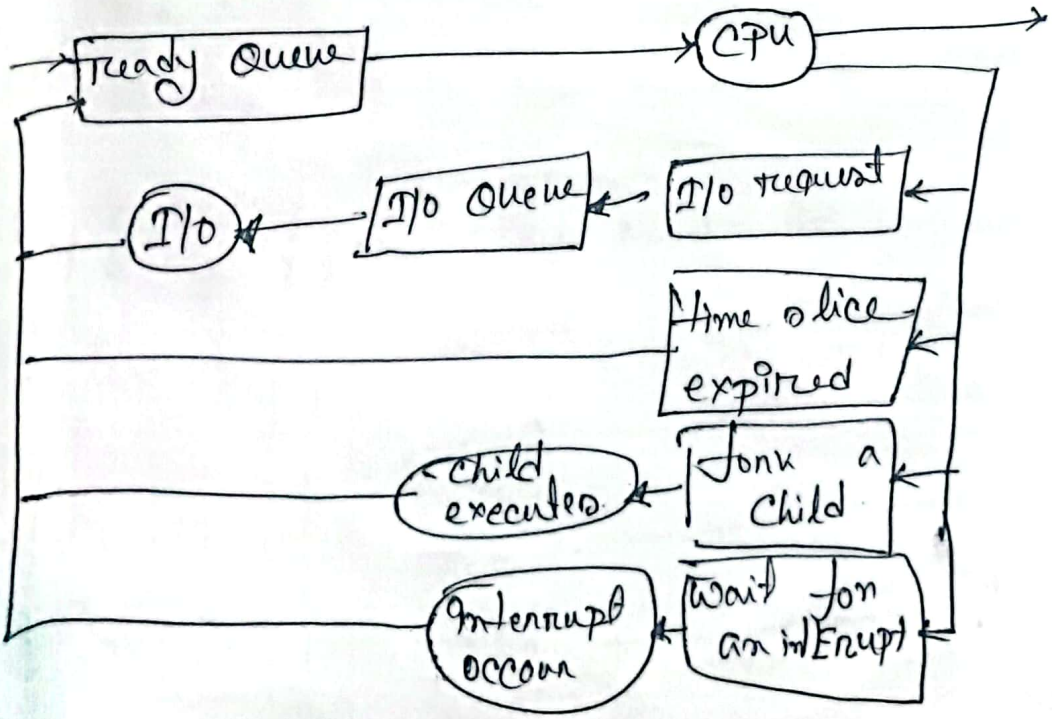
Process Scheduling

- We quickly switch process onto CPU for time sharing
- Process scheduler selects among available process for next execution on CPU.

Maintains scheduling queues of processes

- Job queue – set all process in system
- ready queue – " " residing in main memory, ready and waiting to execute.
- Device queues – set of process waiting for an I/O device

representation of processes scheduling.



Schedulers

Short-time (cpu scheduler)

Selects which process should be executed next and allocated cpu.

- Sometimes the only scheduler in a system.
- short time scheduler invoked frequently → must be fast

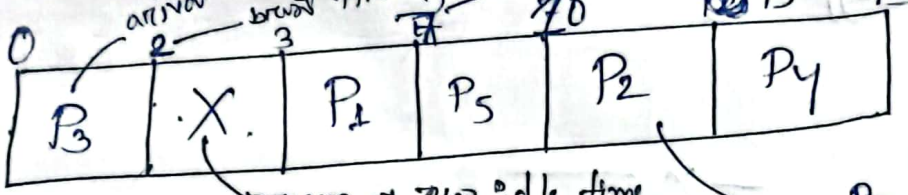
Long term scheduler (job scheduler) - selects which process should be brought in the ready queue.

- It is invoked (ورد) infrequently (sec, min) → slow
- It controll degree of multiprogramming.

First Come, First Served Scheduling

Process id	Arrival time	Burst time
P_1	3	4
P_2	5	3
P_3	0	2
P_4	5	1
P_5	4	3

Queue:



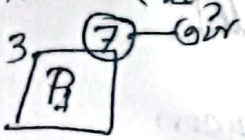
arrival time এবং burst time এর যোগফল (4+3)

process না থাকা idle time

2-3 মধ্যে process নেই

P_2 আসার আগেই process আসা

Exit time হলো শেষ এর



Turn around time = Exit time − Arrival time

Waiting time = Turn Around time − Burst time

Exit time
$P_1 = 7$
$P_2 = 13$
$P_3 = 2$
$P_4 = 14$
$P_5 = 10$

Process	Turn Around time	waiting time
P_1	4	0
P_2	8	5
P_3	2	0
P_4	9	8
P_5	6	3

Advantage

- simple easy to understand.
- easily implemented using queue ds
- It doesn't lead to starvation.

Disadvantage

- doesn't consider the priority on burst time of the processes.
- It suffers from convoy effect.

—x—

Preemptive

Process ready queue (RAM)
থেকে running queue (CPU)
execute হওয়ার সময়
যদি CPU থেকে বাহির
হয়ে RAM এ যায়
পুনরায় ফিরে এসে আবার
আসলো.

SRTF

non preemptive

burst time full complete
হলে then new process
আসবে.

ready queue থেকে
running queue (CPU)

FCFS

SJF

Shortest job first : (SJF) – used both preemptive and non-preemptive mode.

(or SRTF)

of shortest job first called as shortest remaining time first (SRTF)

for n processes time complexity = n log(n)

(Burst time 亡4(01)

(1 unit time execute 2010).

Process	Arrival time	Burst time
✓ P_1	3	~~1~~ 0
✓ P_2	1	~~4~~ ~~8~~ ~~2~~ ~~1~~ 0
✓ P_3	4	~~2~~ ~~1~~ 0
P_4	0	~~6~~ ~~5~~ ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0
P_5	2	~~3~~ ~~2~~ ~~1~~ 0

0–1	1–2	2–3	3–4	4–5	5–6	6–7	7–8	8–9	9–10	10–11	11–12	12–13	13–14	14–16	16–16
P_4	P_2	P_2	P_1	P_2	P_2	P_3	P_3	P_5	P_5	P_5	P_4	P_4	P_4	P_4	P_4

0–1	1–3	3–4	4–6	6–8	8–?	?–16
P_4	P_2	P_1	P_2	P_3	P_5	P_4

Round Robin Scheduling

Pro	Arrival	Burst
P_1	0	4 2 0
P_2	1	5 3 1 0
P_3	2	2 0
P_4	3	1 0
P_5	4	6 4 2 0
P_6	6	3 1 0

fixed amount of time

Time Quantum = 2

ready Queue:

$\Rightarrow P_1, P_2, P_3, P_1, P_4, P_5, P_2, P_6, P_5, P_2, P_6, P_5$

0–2	2–4	4–6	6–8	8–9	9–11	11–13	13–15	15–17	17–18	18–19	19–21
P_1	P_2	P_3	P_1	P_4	P_5	P_2	P_6	P_5	P_2	P_6	P_5

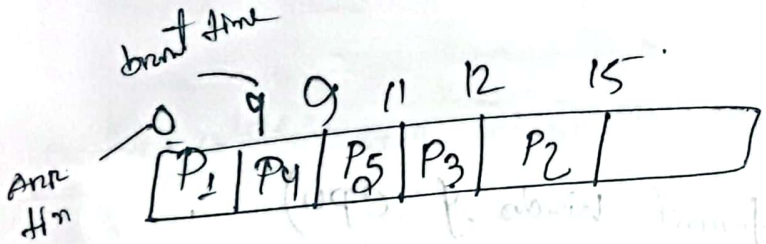
যেহেতু Arr 0 তাই P_1 আসলো Qunt 2, তাই SO 4/2 হয়েছে
2 টো P_2, P_3 আছে so ওগুলো নিয়েছে ready queue তে আর
P_1 কেউ Main Queue তে আসেনি তাও 2 বাকি so আবার
ready queue তে P_2 P_3 এর পিছে দিলাম ready queue

Priority Scheduling: (non preemptive) higher Num represent higher priority.

Process Id	Arr	Bt	Priority
P_1	0	4	2
P_2	1	3	3
P_3	2	1	4
P_4	3	5	5
P_5	4	2	5

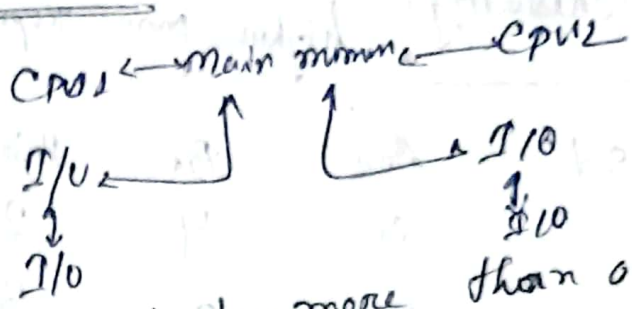
প্রথম 0 থেকে দেখতে হবে.

priority যদি একই same থাকে তাহলে কি করবে.



3rd chapter

Mul-ti procesn scheduling



A multi-processor is a system that has more than one processor but shares the same memory, bus, and j/o devic

It is complex as compared to single processor sc- heduling.

Multiprocesnor system may be.
heterogeneous (different kinds of CPU)
homogenous (the same cpu)

* 2 apptroches in multiple processor scheduling in os

Symmetric / Asymmetric.

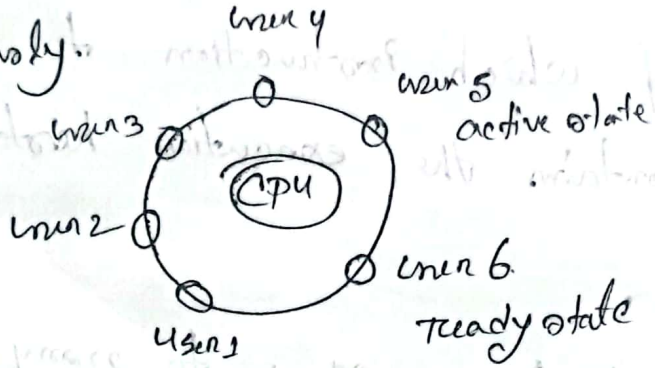
Symmetric
↓
where each processor is self scheduling each procesor have privte que for rady procss.
↓
Execute rq

Asymmetric
↓
used when all the scheduling decisions. and I/O processing are handled by a single processor called the master server

- Simple, reduces the need for data sharing.

Time sharing

A time shared OS allows multiple users to share computers simultaneously.



Here 5 active state, 6 ready stat 1, 2, 3, 4, waiting.

active state: user program is under the control of cpu. only one program in this state.

ready state: user program is ready to execute wait for turn to get cpu

waiting state: user program is waiting for some I/O operation.

Advantage

- Each task get equal opportunity
- CPU Idle time can be reduced

disAdvantage

- reliability problem
- Data communication problem.

Thread

(Code এর কোন অংশ নিয়ে unit unit করে কাজ করা)

It is a flow of execution through the process code that keeps track of which instruction to execute next and stack which contains the execution history.

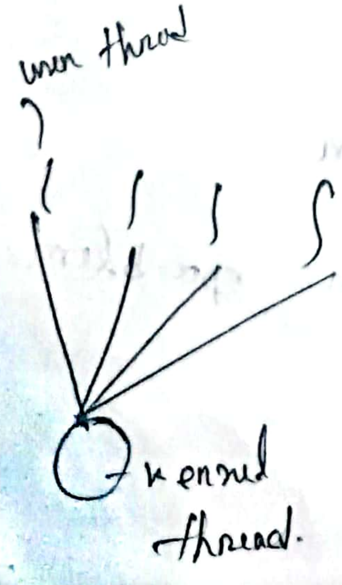
Multithreading Models

Many to one, one to One, Many to many.

Many to one

- We have multiple user threads mapped to one kernel thread when user thread makes a blocking system call entire process blocks.

It is more efficient

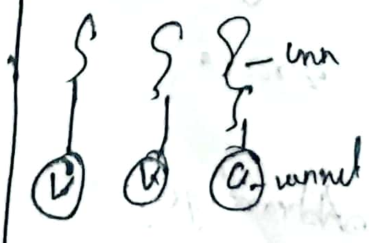


One to One

Each user level thread maps to kernel thread.

একটার block হলে অন্যটার block হবে না)

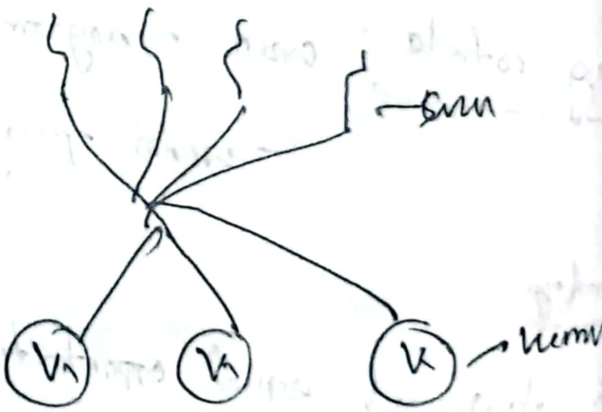
Exam - Windows. OS/2



Many to many

- Allows many user level threads to be mapped to many kernel threads
- Allows the OS to create sufficient number of kernel threads

Ex - Solaris 2, windows NT/2



~~multithreading~~

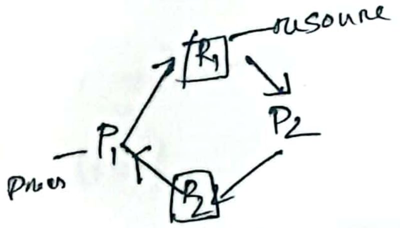
resource allocation: It is the process by which a computing system aims to meet the hardware requirement of an application run by it.

when multiple user are logged on the system or multiple jobs are running at the same time, resources must be allocated to each of them

OS live as manager decide করে কাকে দিবে.

2 types of resource allocation in OS.

resource allocation with deadlock



circle create হলে deadlock.

একটা wait করে অন্যটার. একটা অন্য করে so circle

resource allocation without deadlock.

If there are multiple instance of the resource, there are chances that deadloc will not occur

resource and process মধ্যে circle হলে deadlock occur হবে না হলে হবে না.

Dispatcher

It is done after the scheduler.

After selecting process → dispatcher given to it CPU.

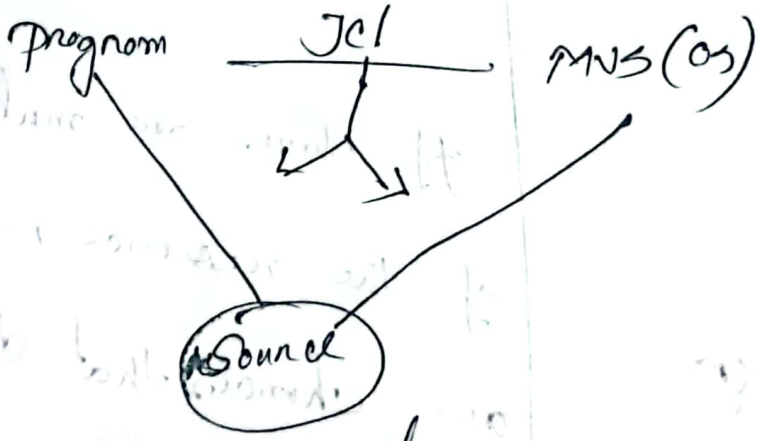
ready state → short term → Dispatcher → running state.

(function ⇒ switching context
switching to user mode)

Job control language:

It is instruct the system for the ~~bath~~ batch job.

Code to tell the OS about
the task you want to perform.



Job statement in jcl:

Job statement gives the job identity to the OS in the spool and the scheduler. It is 1st control statement of jcl.

It help the OS in allocating rs rescheduling. also useful for analyzing the required CPU and issuing notification to the user

Syntax

Job statement (jcl)

//Job name Job positional-param, keyword-param.

(11) Symbol of JCL

(11) is the symbol of jcl statements. Each jcl statements must begin with this symbol. otherwise jcl statements error.

(11) It to abovid runtime exceptions. (11) biggining me hi

Condition checking in jcl.

Condition checking is possible at both job and code level It is done through the COND keyword with a return code and operand as predefined in jcl.

hienachy levels in jcl

Every statement of jcl consist of the following keywords.

- Name. Field. operation. operand. parameters. positional.
- keyword. comments if any.

Shortest job first (Non preemptive)

full Execute.

Proc	Arriv	Burst
P1	1	3
P2	2	4
P3	1	(2)
P4	4	4.

1st arrive (যে আগে আসবে)

(Burst time কম যে same হলে)

P1, P3 এর মধ্যে P3 burst 2 কম

(burst time same হলে arrival দেখতে হবে)

Grand chart

//////	P3	P1	P2	P4

0 1 3 6 10 14.

Shortest job first (Non preemptive)

full Execute.

Proc	Arriv	Burst
P_1	1	3
P_2	2	4
P_3	1	2
P_4	4	4

P_1, P_3 এর — P_3 burst 2 কম

1st arrive (যে আগে এসেছে)

(Burst time কম যে Same হলে)

(burst time same হলে arrival দেখতে হবে)

Grand chart

	P_3	P_1	P_2	P_4
0–1	1–3	3–6	6–10	10–14

0 1 3 6 10 14.

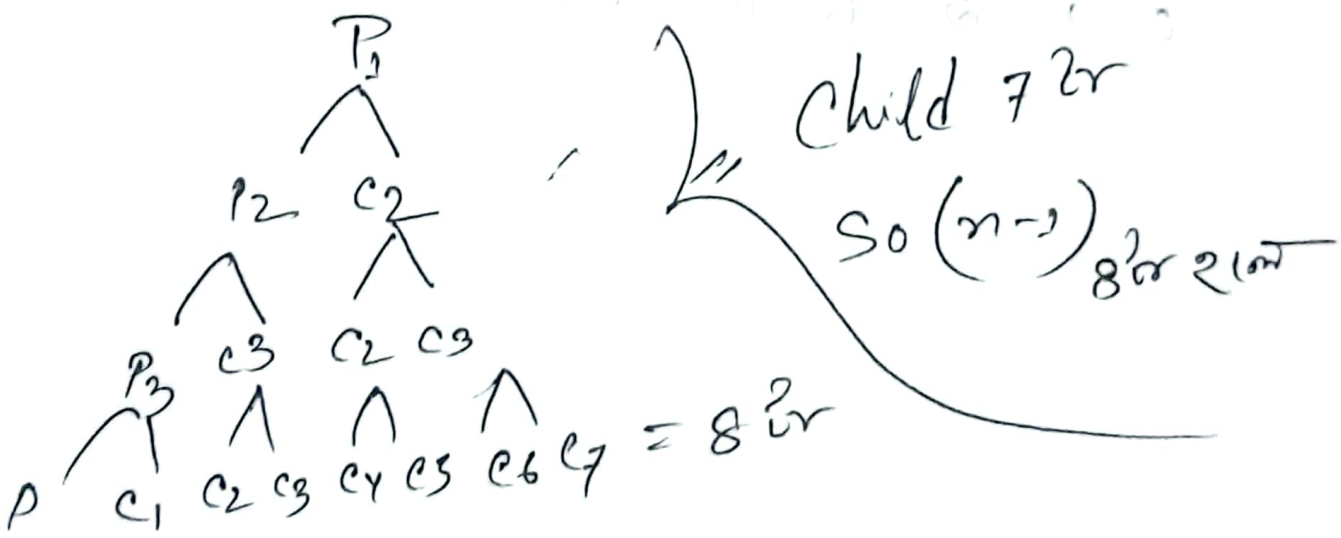
fork code

int main

```
fork
fork
fork
```

} 3 বার ⇒ 2^n

- output $2^3 = 8$ বার



যদি

i = 5

হবে $2^5 = 32$ বার হবে

child 31 বার

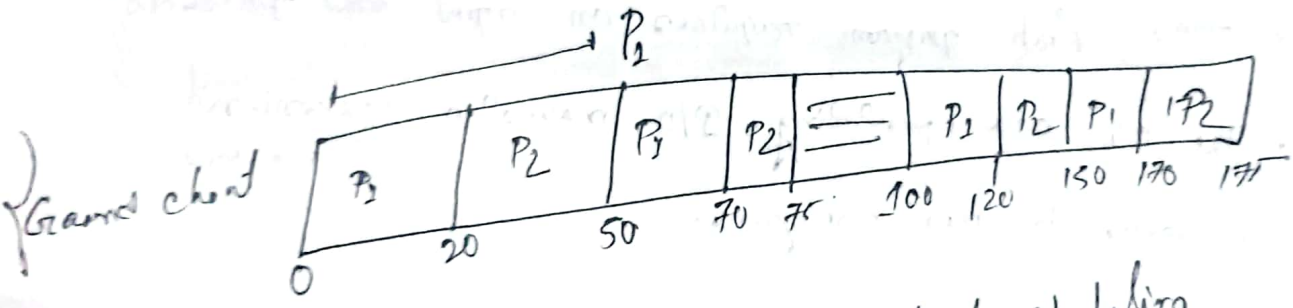
Rate monotonic scheduling

Lower priority gets higher priority.

high " " lower "

$P_1 = 50$, $t_1 = 20$ — higher priority

$P_2 = 100$, $t_2 = 35$

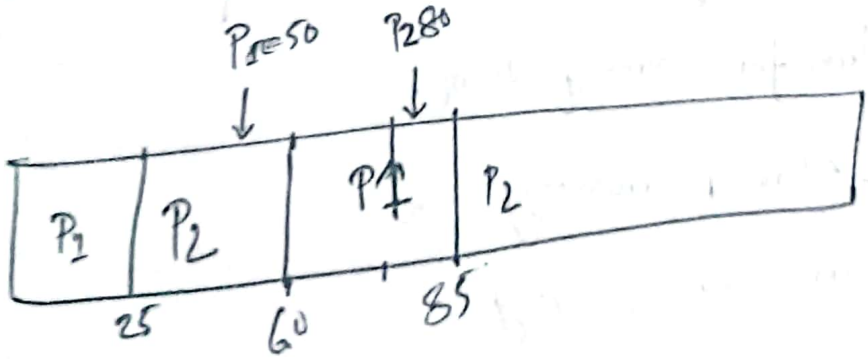


Earliest deadline first scheduling

high = high priority, low = low priority

$P_1 = 50$: $t_1 = 25$

$P_2 = 80$ $t_2 = 35$



Chap 4

OS Services **

Program execution: The OS provides an environment where the user can easily run programs. Main purpose is allow the user to execute programs.

I/O operations: Each program requires an input and produces output. The OS are providing I/O makes it convenient (ব্যবহার করে) for the users to run programs.

File management: OS also helps manage files. If a program needs access to a file, it's in the OS that grants access. These permission include read, write etc.

Communication with process: The process need to communicate with each other to exchange information during execution. Its include data transfer among them.

Communication via - Shared memory or
- massage passing

একটি Computer আরেক Comnd বা আরেকটা Network Conect থাকলে info exchng হয়।

Error handaling: OS also handles the error occouring in the cpu, in I/O devices. It ensure an error doesn't occour frequently (CPU) and fixes errors. It also looks for any type of error or bug that occour in the task.

Security and privicy: OS keeps our computer safe from an unathorized user by adding security layer.

OS tasprects our secrets and provide us facility to keep safe.

User interface: User either interface with the OS through

- Command line (CMD)
- graphical user interface.

OS Advantage vs Disadvantage

Advantage	Disadvantage
① providing user interface	① It's complex and diffcult to use
② Manages files and folder	② Expensive to purchase
3) Managing security, privicy	③ difficult to maintain.
④ Allowing Different Application to run concurrently (Together)	④ It can be poor to attacks from malicious users.

* (function / features) of OS
  - Controlling Computer hardware
  - organising of files
  - providing user interface
  - security
  - protecting data files
  - reporting & logging
  - provides platform for running Application
  - handle memory management & CPU scheduling

* The OS has 2 components
  - shell
  - kernel

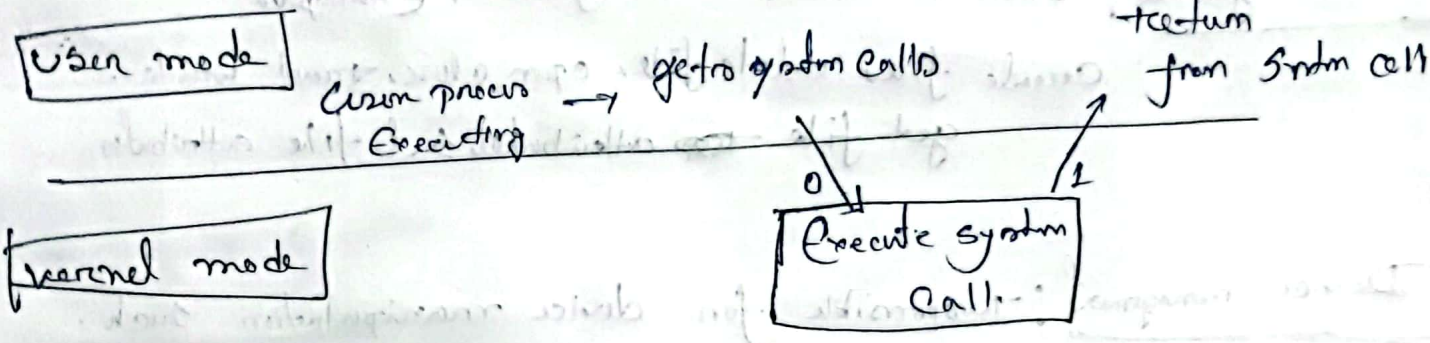
Shell: Shell handle user ~~instruction~~ interaction. It's outermost layer of the OS and manages the interaction between user and operating system.
(interpreting input and output from OS)

Kernel: Central component of an OS. that manages operation of computer and hardware. basically manage operation of memory and CPU time. It acts bridge between application and data processing performed at hardware level using IPC and system calls.

System Calls

- A system call is a way for a user program to interface with OS. (Provide service via API)

Computer program makes System call when it makes a request to OS kernel.



Some system calls are:

open, read write, close.

wait () = a process must wait for another process finish

fork () = To make a copies of themselve.

exec () = Executable file replaces an earlier executable file within an active process.

kill () = processes to terminate by sending them a signal.

Exit: when programme must be Stopped

Types of System calls:

process control: It is used to direct the processes.

- Create process, terminate process, load, execute and exit allocate and free memory.

File management: used to handle the files : Examples
Create file, delete file, open, close, read write.
get file attributes, set file attributes

Device management: responsible for device manipulation such as → request device, release device, logically attach or detach devices

Information management: responsible for device manipulation. Such as
get date and time, set date or time
get " " set data.
get process file, set process file

Communication: used for communication. Such as
Create delete connection.
Send, receive message.
transfer Status.
attach detach remote devices

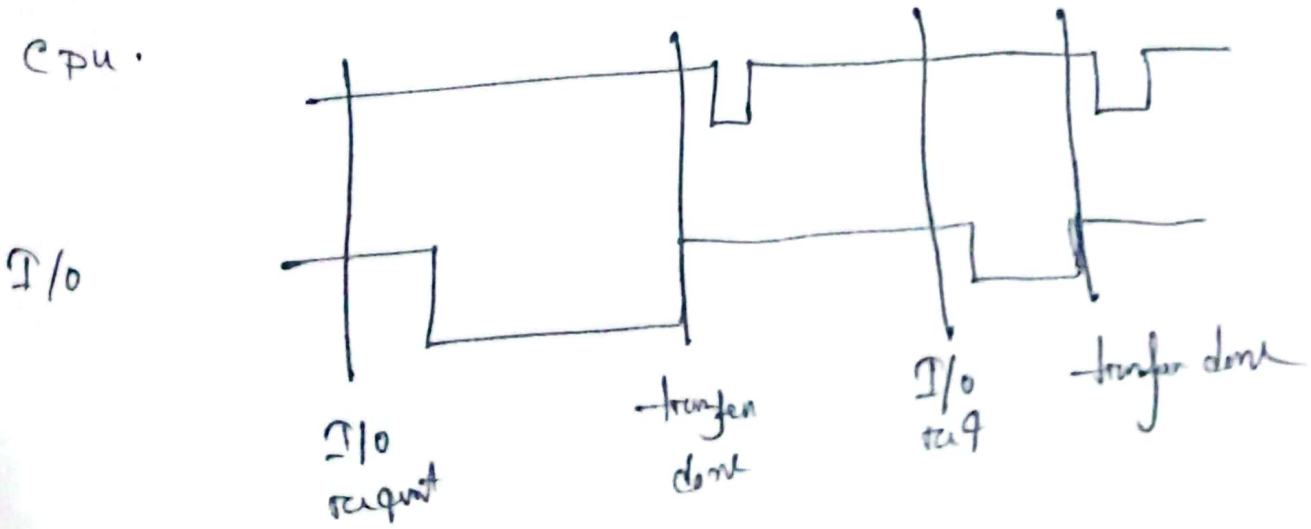
Interrupt

device controller inform CPU that it has finished its operation by causing an interrupt.

An OS is interrupt driven.

Interrupt handling

- An OS preserve the state of the CPU by storing registers and program counter.
- Determine which type of interrupt has occurred
  - polling → vectored interrupt system.
- Separate seg. of code determine what action should be taken for each type of interrupt.



Interrupt OS handling significant for

- Efficient multitasking
- Device communication.
- Error management
- Scheduling.
- responsiveness, concurrency.

Multiprogramming

- where multiple programs consists in the main memory

objective - maximize Cpu utilization
: - reduce cpu idle time
- used time sharing mode

Multitasking

Logical extension of multiprogramming. It allows OS to execute multiple tasks simultaneously. on a single Cpu.

- increase Cpu utilization.
- enhances system responsiveness
- shorten response time

It refers to thread execution when one process divide sub tasks.

Multiprogramming

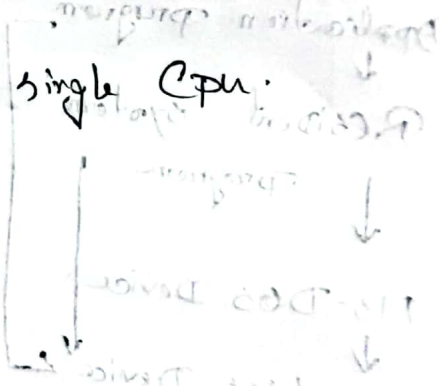
- where multiple programs consists in the main memory.

objective - Maximize Cpu utilization.

- reduce cpu idle time.
- used: time sharing mode

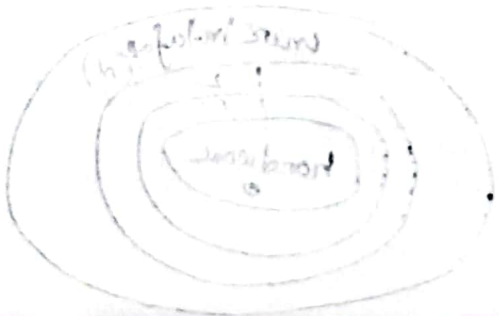
Multitasking

Logical extension of multiprogramming. It allows OS to execute multiple tasks simultaneously. on a single Cpu.



- increase Cpu utilization.
- enhances system responsiveness.
- shorten response time

It to refer to thread execution when one Process divide sub tasks.

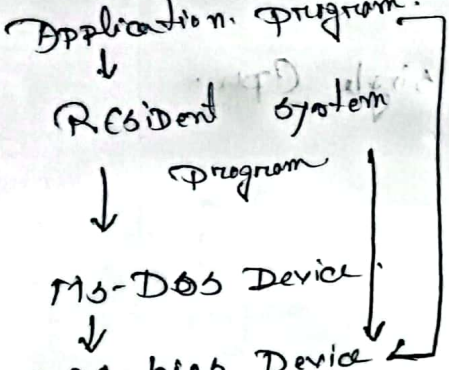


Operating System Structure (interconnection of components)

- Simple
- Layered approach
- Monolithic approach
- Micro kernels
- virtual machine

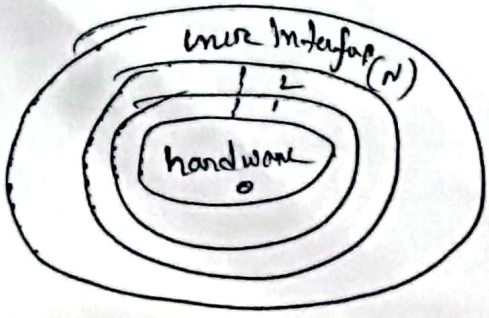
Simple Structure: OS is simple, limited, with no well defined structure. The MS-DOS is best example of OS.

user program fails then full system crashes.



Layered approach: OS can broken into pieces and retain much more control on system.

OS broken into layer. bottom is hardware topmost is user interface.



← UNIX

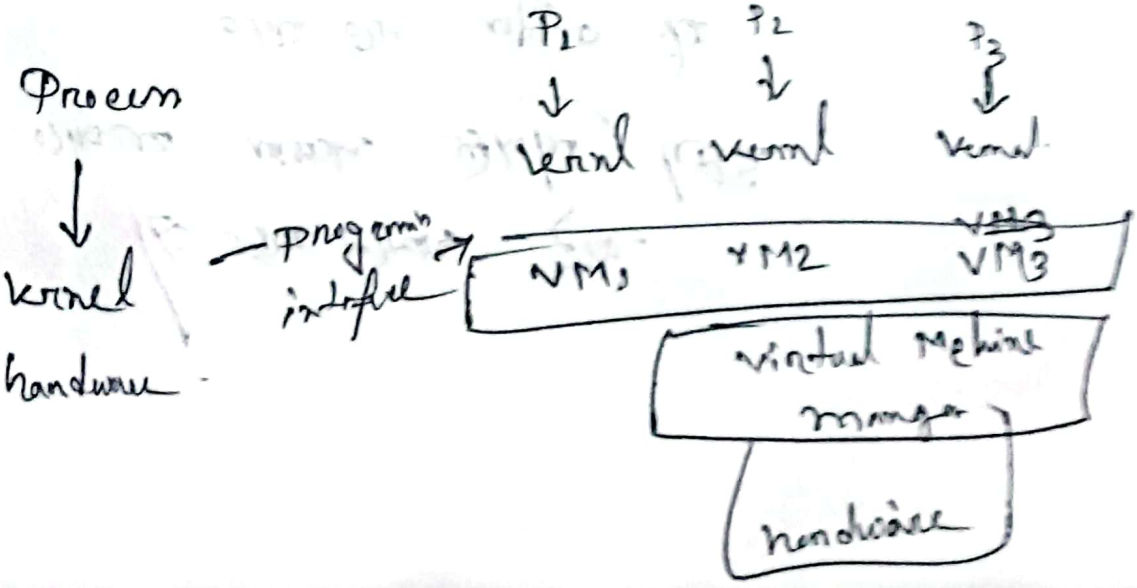
monolatic approach : where the entire os is a large binary program in a kernel mode.

Microkernel mode : In this Design for removing all non essential components from the kernel and implementing them as system and user program.

virtual machine : virtualization is a technology that allows us to abstract the hardware of a single computer into several different execution environments.

(Personal Computer each execution)

Goal = multiple operating sys can operate on one same hardware, data restore easy, operational feasibly basis



Load balancing (server farm)

- It refers to efficiently distributing incoming network traffic across a group of backend servers.

Multicore processor: In os can tell the processor to do multiple things at once (single cpu)

OS review class

- Interrupts and OS
- Multiprogramming and multitasking
- Computer system architecture – single processor/ multiprocessor/ clustered system, real time system
- Dual mode
- Operating-System Services
- System call
- OS structures
- Process, process state, PCB, Operations on Processes, thread
- Interposes communication : shared memory approach, producer –consumer algorithm
- CPU Burst cycle
- Dispatcher
- Context switching
- Scheduling criteria
- Scheduling algorithms ; FIFO, SJF, SRTF, RR , PRIORITY
- Real time scheduling algorithms ; Monotonic and EDF
- Multicore processor
- Multithreading
- Load Balancing
- Deterministic approach to evaluate the performance of scheduling algorithms