

23/02/2021
1st hour

Operating System

Computer is softwares + hardware

CPU, RAM, ROM, I/O devices

Set of tested programs & documents which are used to work on application

A computer is a combination of both hardware and software

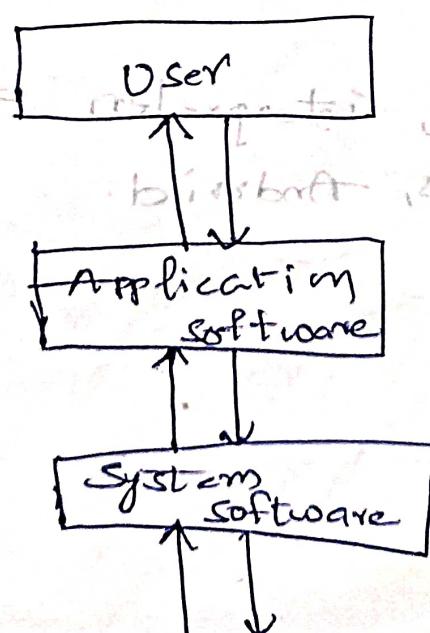
Hardware :- The electronic devices RAM, ROM, I/O devices

are assembled with a help of cables. These are called hardware.

Software :- The set of well tested programs and the document that says how to maintain and how to operate is called software.

These are two types of softwares as

1. Application software
2. System software



Application Software

The set of programs written satisfying the specific tasks called application software.

Eg:- Editplus, Note pad, media player, windows etc.

To install the application software computer must contain an operating system.

System Software - The programs that are written for

operating the hardware components is called as system software.

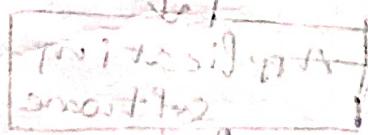
These programs written for general purpose that should be efficient and consistent.

to all the users.

Without system software we can't operate computer.

Compiler, interpreter, debugger etc.

Eg:- windows, Android.



- Functions of operating systems
1. Resource Manager
 2. Process Manager
 3. Memory Manager (Main)
 4. I/O Manager
 5. Storage Manager
 6. Security and protection

GOT is an interface which performs system calls, and internally calls well defined methods.

Types of operating system

1. Batch operating system / Non-multiprogramming system	2. Multiprogramming system	3. Multitasking system	4. Multiprocessing system	5. Real time OS
with respect to generations of the computer	changes in designing the OS from 1st generation	1st generation	2nd generation	3rd generation
there are so many OS. when we see to today. The types can be	the OS can be	the OS can be	the OS can be	the OS can be

a. Batch OS

1. Multi programming OS

2. Multi tasking OS

3. Multi processing OS

4. Real time OS

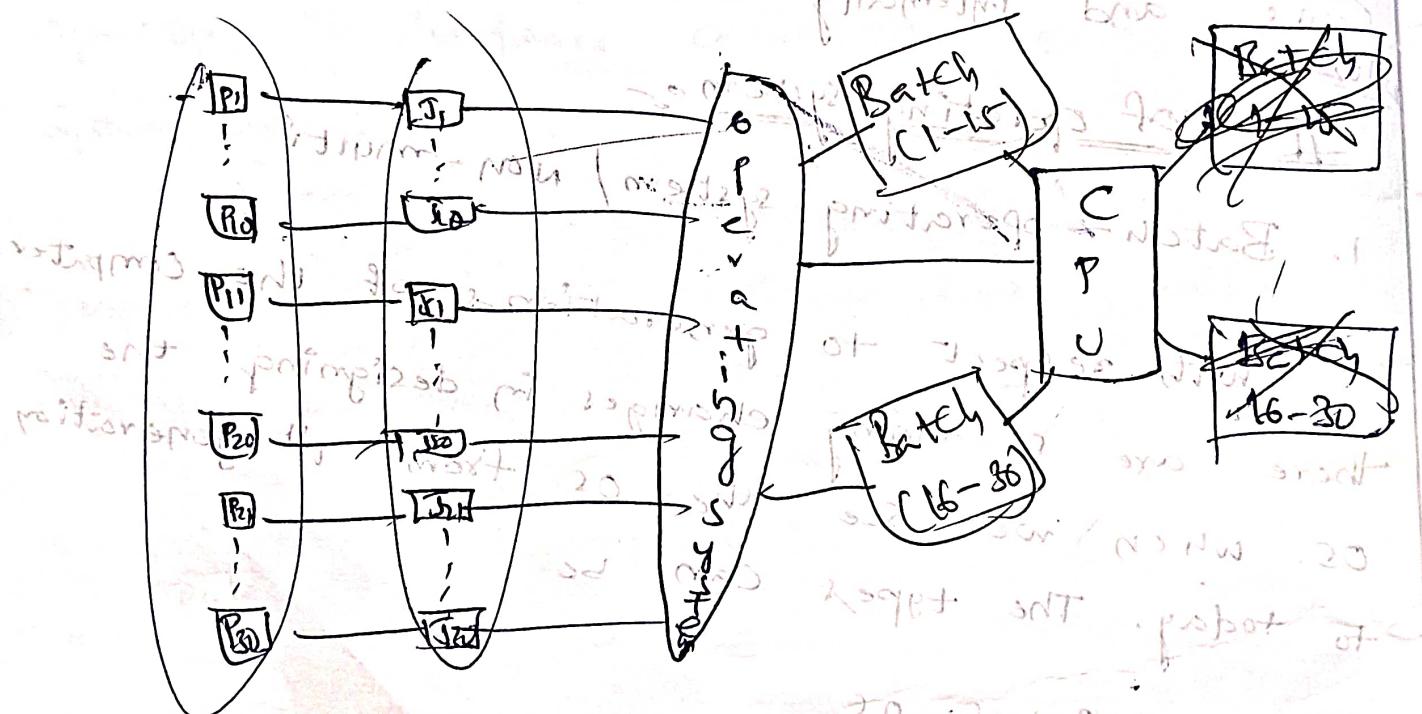
Batch OS \leftarrow operating systems that waits for multiple users to use it at same times without direct communication between them.

This is done by having the users submit their jobs to the operating system, which then processes them one at a time.

Ex:- Payroll, Bank statements

Let us consider 20 users doing job on OS

then it takes 1 hour CPU at a time of allocated



Advantage :-

Easy to estimate the time to complete

a given job, easy to manage large repeated work, idle time is less, sharing by multiple users is possible

Disadvantage :- 1. if a job fails, the other

will take unknown time.

Costly

2. Multiprogramming \Leftrightarrow OS is able to execute more than one program at the same time. It allows multiple task are going to happen on a OS with direct interaction / communication.

Here, let us consider three task.

task 1 takes 2sec of time, task 2 takes 1hr, task 3 takes 3hrs. If task 2 executes first then we have to wait 1hr to execute another but here time space of task one is 2sec. but these has to wait.



advantage :-

Direct interaction between task and OS

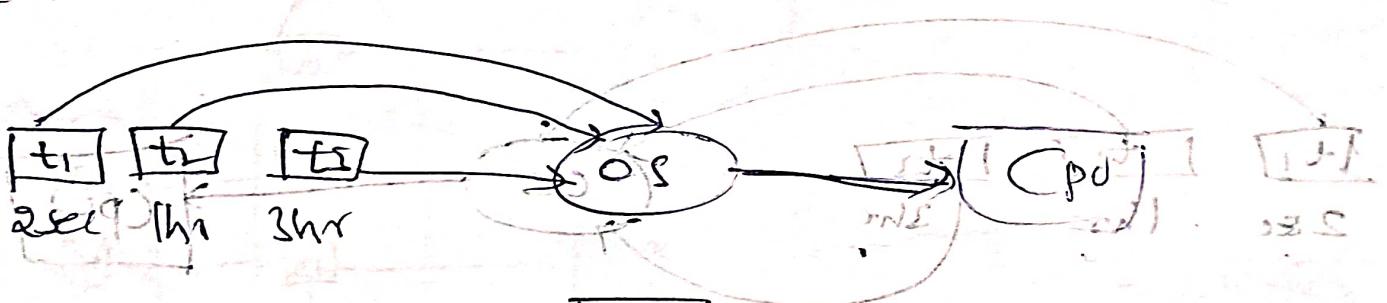
disadvantage :-

* it takes time complexity.

* waiting time.

10 overcomes 10 programming
of multi-tasking os, The multitasking os is
discovered. prop and smart algorithm 2001/02
background information 1996 1997
Here also there is a direct
interaction between task and os. There
is time convention for each task
there is time convention for each task
for certain time. After time is excluded the
next task goes until then pre-rids take
and waits for queue.

Ex:-



Permin

time

convention

→ first arrived → first executed

for

* if t_2 goes first to executes

$$t_2 = 1\text{hr} - 15\text{min}(\text{OS}) \rightarrow 14.5\text{min}$$

Queue:-



script position

* if t_1 executes second,

$$t_1 = 2\text{sec} - 15\text{min} = -14.5\text{sec}$$

These 14.5 sec can't wait

time is used to complete next task to reduce waiting time

* first executes then

$$t_3 = 3\text{hr} - 15\text{min} = 2\text{hr } 45\text{min}$$

Queue :-

t ₁	t ₂	t ₃
----------------	----------------	----------------

These will repeat until the program task completed

Advantage :-

* waiting time reduced

* time consumption is less a disadvantage

complete small time programs

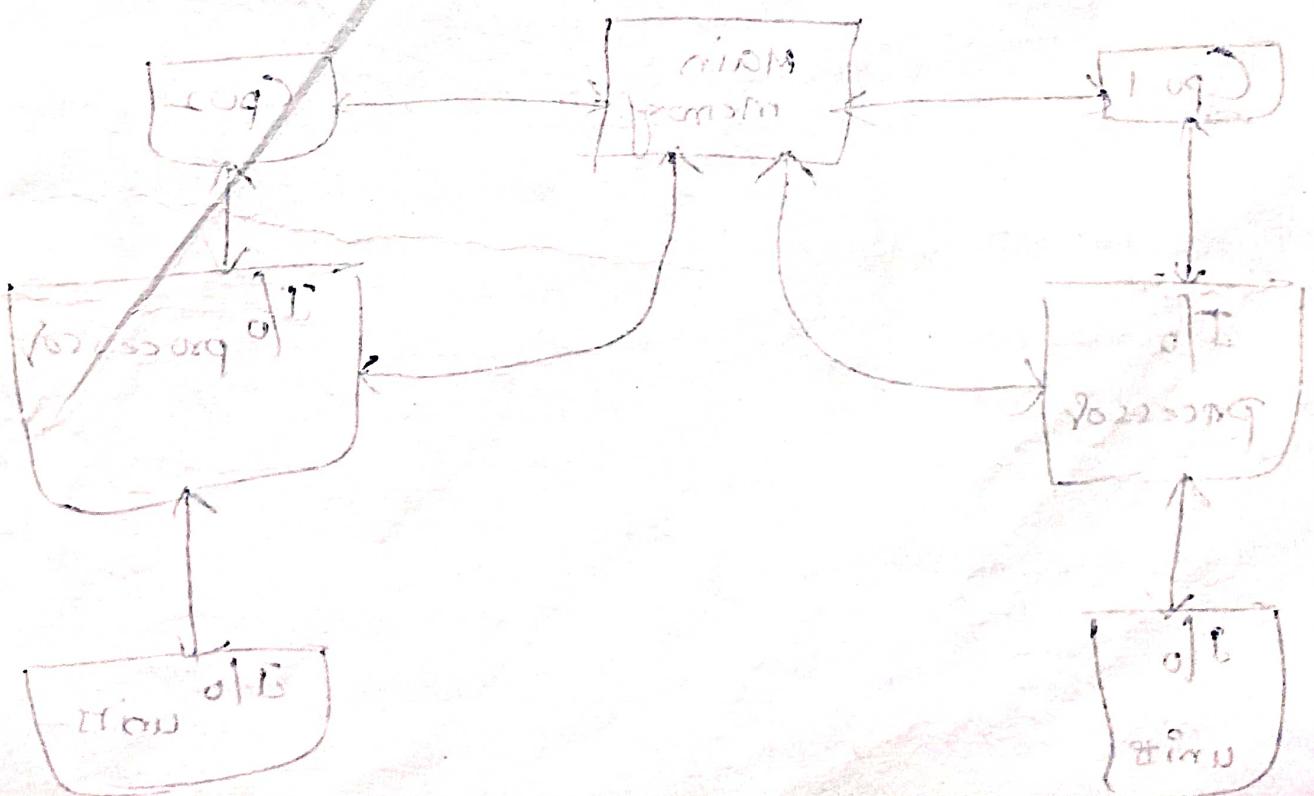
task is completed

disadvantage :-

* greater risk of errors

* lower efficiency

* processor slows very quickly



Multi-processing OS

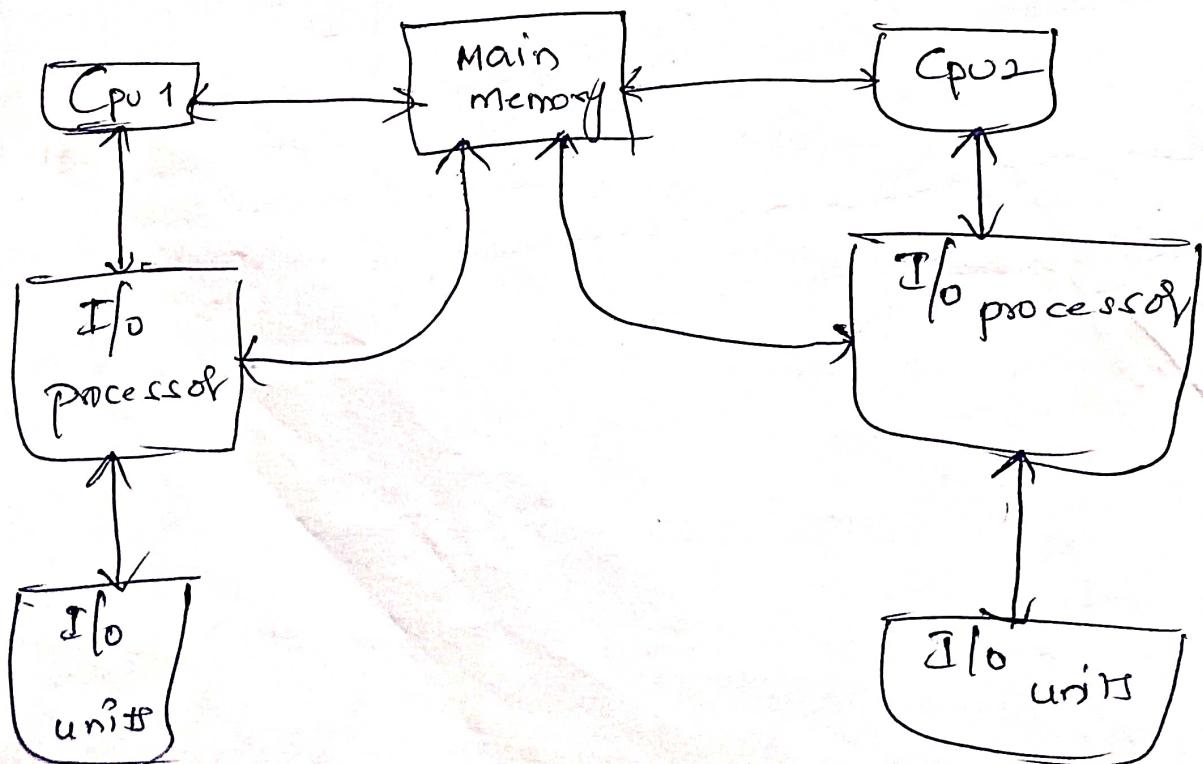
Multiprocessor OS are used in OS to boost the performance of multiple CPUs within a single computer system.

multiple CPUs [are] linked together so that a job can be divided and executed more easily.

Quickly

- parallel computing is performed by multiprocessors
- there are several processors in a system
- and each of them can run multiple processes

Simultaneously



Types of multiprocessing OS:-

1. Symmetrical :- It is a computer processing done by multiple processors that share a common OS and memory unit.
2. Asymmetrical :- An asymmetric multiprocessing system is a multi processor computer system where not all of the multiple processor connected to central processing units are treated equally.

Advantages :-

- * Helps in parallel computing
- * Reliable
- * Increased throughput

Economical

Disadvantages :-

- * More complex architecture
- * Required large memory
- * The system crashes if the processor in charge of a particular task fails.

Real-time OS :-
It is an OS that guarantees that tasks will be performed in real-time by computers that are connected to those systems in real-time. It is mostly used for real-time computations like control systems, traffic control systems, etc. It is employed mostly in those systems where the results of the computation while it is executing influence a process to influence a process.

Advantage :-
→ easy to layout and execute real-time applications under the real-time OS.
→ memory allocation is best managed.

Time scheduling



Task 1

processor

wifi

Bluetooth

time

Resource Management

file

operating systems & operations 2 -

Post
ROM
BIOS
Setup

In every operating system, there are privileged programs and secured programs. This acts as a layer which is called Kernel. The Kernel program is directly interacting with the hardware component. In Kernel there is a mode-bit which is a hardware component having value either 0 (0) or 1.

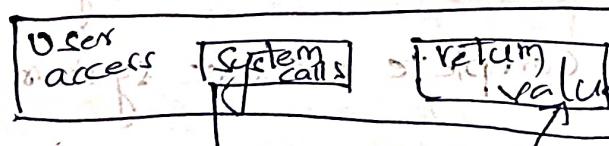
Dual-mode

The user mode and Kernel mode together is called dual-mode. It communicates with the CPU to complete the task.

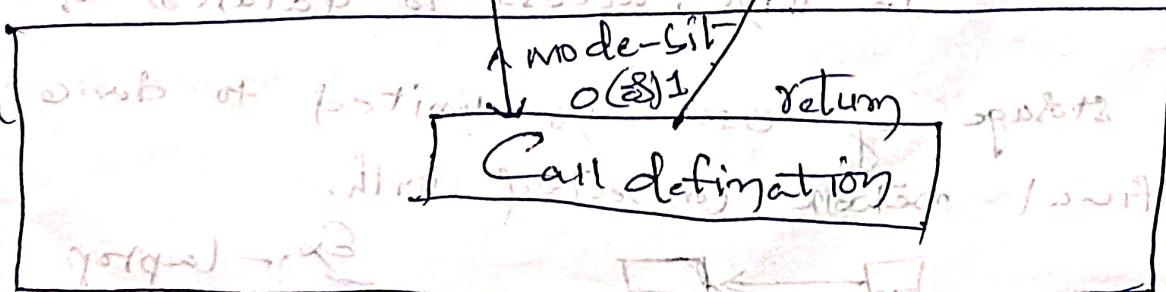
When user mode is going on, the mode-bit value is 0 and 1 for Kernel mode.

Sometimes without Kernel OS can

interact with the CPU

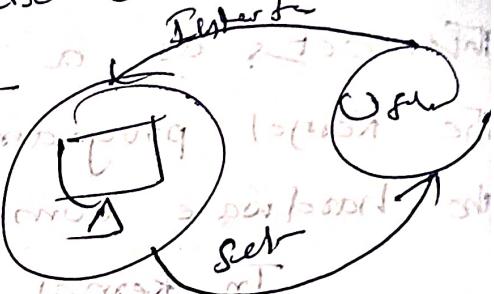


Kernel



Timer mode msg

Another operation of the OS is timer. It starts the counter value and reaches to the set value with interrupt the user with the error msg (or) alert msg.



Computing environments

There are 8 computing environments:

1. Traditional / Wired

2. Mobile / wire

3. distributed / peer - staff - or - clu

4. Client - server / peer - staff - or - clu

5. peer - to - peer computing / peer - staff - or - clu

6. virtualization / images

7. cloud / platform, more, fr

8. Real time Embedded

Traditional

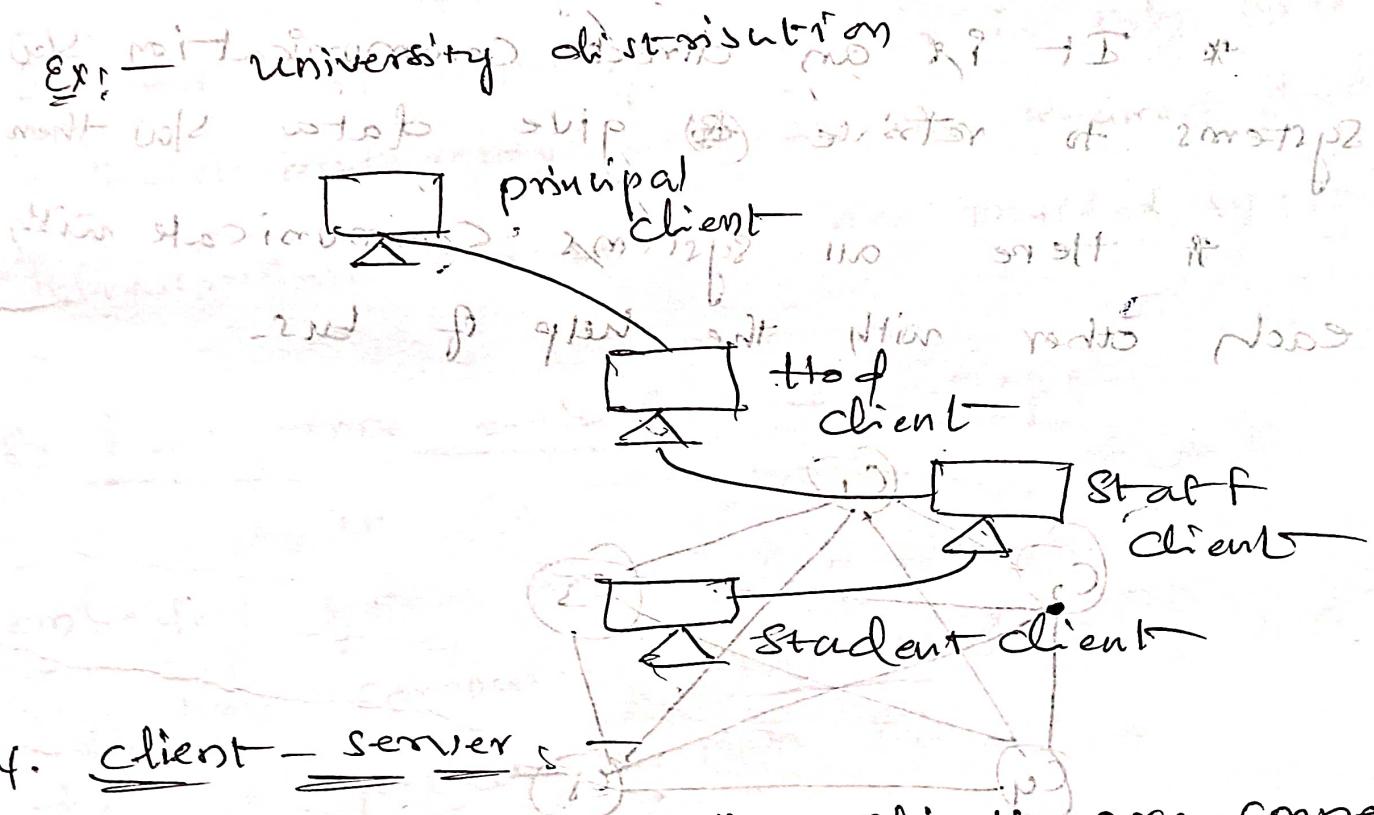
It is a process of using physical data centres for storing digital assets and running complete network systems for daily operations.

In this, access to data, (or) software,

(or) storage system is limited to device (or) official network connected with.

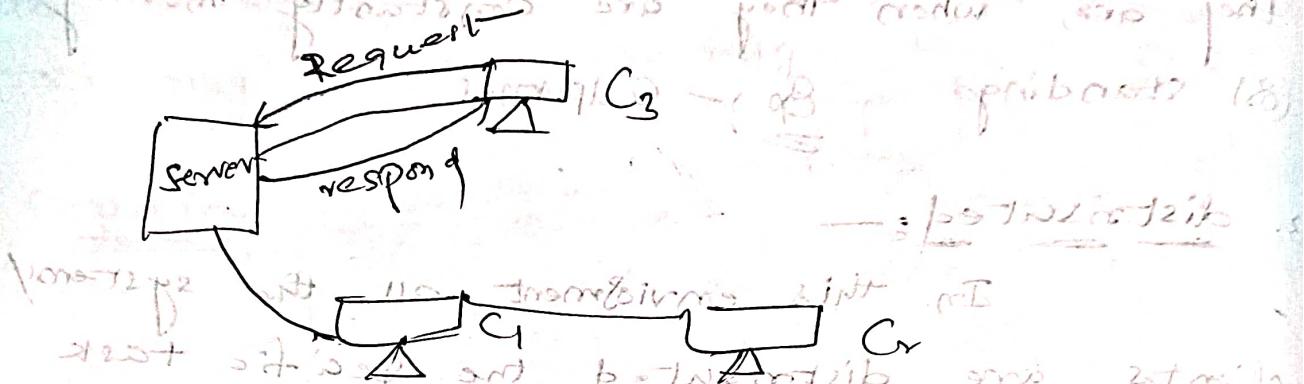
2. Mobile -
- * It is using wireless communication
 - * It refers to various devices allow people to access data and information from where they are, when they are constantly moving
 - (8) standing Ex) - Cellphones

3. Distributed -
- In this environment, all the systems/clients are distributed the specific task to make the task easier and reducing time.



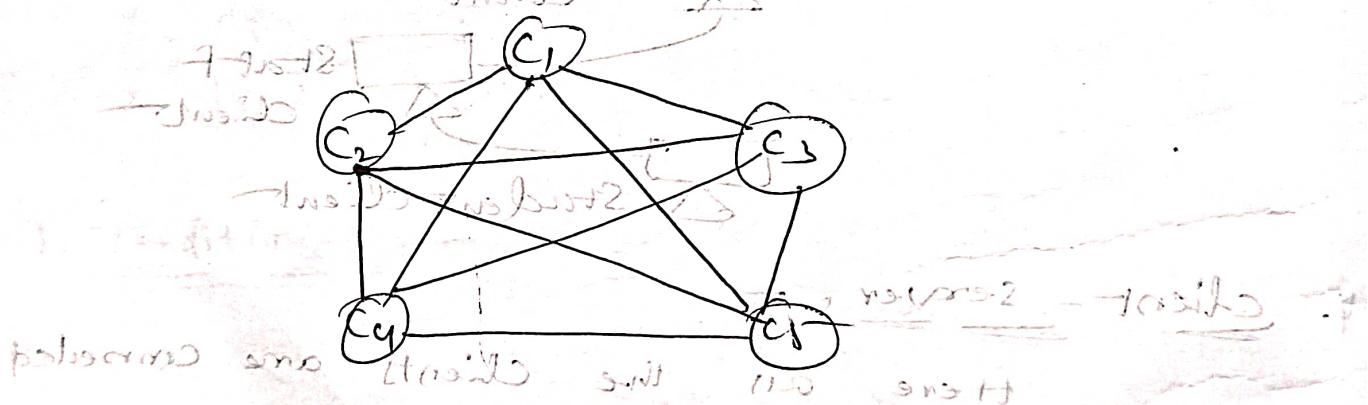
4. Client-Server -
- Here all the clients are connected to a server with topologies. If any client requires some information/data then it request to server and server checks the request and respond.

Ex:- There are 3 computer systems and connected to server. C₃ requires file from them. C₃ request to server and server takes the information and respond to C₃.



Peer-to-peer :-

- * It is an direct communication between systems to retrieve give data & then communicate with each other with the help of bus.



6. Virtualization

Virtualization is the creation of virtual version of something such as server, a desktop, a storage device, an operating system, a network resource such as maintaining called as virtualization.

gathering of the participants of a meeting, without having to travel to a central location or wait for people to arrive. This makes it easier for people to attend the meeting, saving time and money.

Cloud computing is another example of how technology has changed the way we work. Instead of having to buy and maintain our own hardware, we can rent it from a cloud provider and access it from anywhere in the world.

F. Cloud :-

cloud computing consists of

* PaaS

* IaaS

* SaaS

In this cloud computing, the users only have to connect and relax and all the required platform, infrastructure and software are provided by cloud.

G. Real-time embedded Computing :-

It is a particular version of an embedded system that works on the basis of real-time computing represented by a dedicated type of operating system.

If working principle follows:

follows: Quick response to external factors.

factors: an embedded system must work within strict time constraints.

2. Operating System structure :-

→ At the time of designing the operating system, the developers may have a clear idea on the services that should provide to the users/ programs.

→ The operating systems categorized in 2 :-

1. System Services

2. User Services

User Services

Whenever user interacting with OS it facilitates the following services:

- 1) User Interface
- 2) File creation & Manipulation
- 3) Program execution
- 4) Communication

I/O Controller

Error detection

System Services

1) Resource allocation

2) Accounting

3) Security & protection

1. User Interface :-

* User and OS are connected with each other help of Interface.

* There are diff. types of Interface.

Connection.

a. command line execution:-

- * When user needs different commands regarding to I/O and output & to perform tasks like create, delete, print, copy, etc.
- * Basic operation performed by kernel of OS.
- * Responsible for memory management.

b. Graphical User Interface:-

- * playing games, videos done with GUI.
- * need for desktop environment.
- * all need graphics.
- * Taskbar showing running programs.

c. program execution:-

- * program always execute by processor with help of disk as primary storage device.
- * C drive is secondary storage device convert to secondary.
- * program statements sent to processor.
- * code - processor is responsible for which the program is executed and then terminated.
- * OS responsible for execution of programs.
- * Coding, linking, allocation, execution overall it is complex process that involves many steps.

Communication :-
→ process of communicating on another system
is called communication.

→ os refers to exchange data between different processes with system.

4) I/O Controller :-

→ os acts as device controller whatever Input output is attached to os is I/O controller.

→ I/O is a hardware component that interface various input & output devices like keyboard, mouse, printer, networks etc.

→ os communicate with I/O through device drivers acts as interface b/w os and I/O Controller.

5) Error detection :-

→ process of identifying & correcting errors (if any) mistakes in a system.

→ This techniques are used to ensure the accuracy & integrity of data as it is transmitted (if any) started.

6) File Management :-

→ os search the data to store.

→ Computer → process of → process, store & retrieve the data.

→ P-for to process a file.

→ security measures to protect files

systems services: → responsible for maintaining data consistency, providing shared resources, and protecting the system from external threats.

1.1 Resource Allocation:

- A Resource has many processes when P, allocate resource, then positive waiting system calculate by operator
- The process of assigning available resources to processes completing process on system.
 - All process gets a fair share of resources
- Round Robin scheduling among set of processes with priority based on processes to

FCFS

short job first

2. Accounting:

- It maintains all data that how many user are there, when they login & logout & what they do.
- It enables system administrators to make informed decisions about resource allocation, system optimization and system security by tracking the usages of system resources by processes of each user over time.

(B) accounting service set priorities

→ Security measures to protect

systems services: → Help user to maintain security
from point point of view of system.

1. Resource Allocation:

- A Resource has many processes when P, allocate
one resource, then P will wait in system calculate
by operator or programmer to assign resources
- The process of assigning available resources
such as CPU time, memory, I/O devices &
methodic to completing process on system.
- All process get fair share of resources
- using Round Robin priority memory set to
priority of process to make process ID

1. FCF

short job first

2. Accounting

- It maintains all data that
how many user are there, when they login & logout
& what they do.
- It enables system administrators to make
informed decisions about resource allocation, system
optimization and system security by tracking usage of system resource by
processor & user operation with help of
process & system resource.

(b) maintained storage system performance

3. Security & protection :-

- unauthorized access system, protected by password.
- It have many folders which has security.
- It store data & retrieve data. performs Read/ write/ append for protection of file.
- write/ append for protection of file.
- They require a combination of technical measures & user awareness to ensure the safety & security of system resources.

System Calls :-
System calls are the interfaces between operating system and its services.

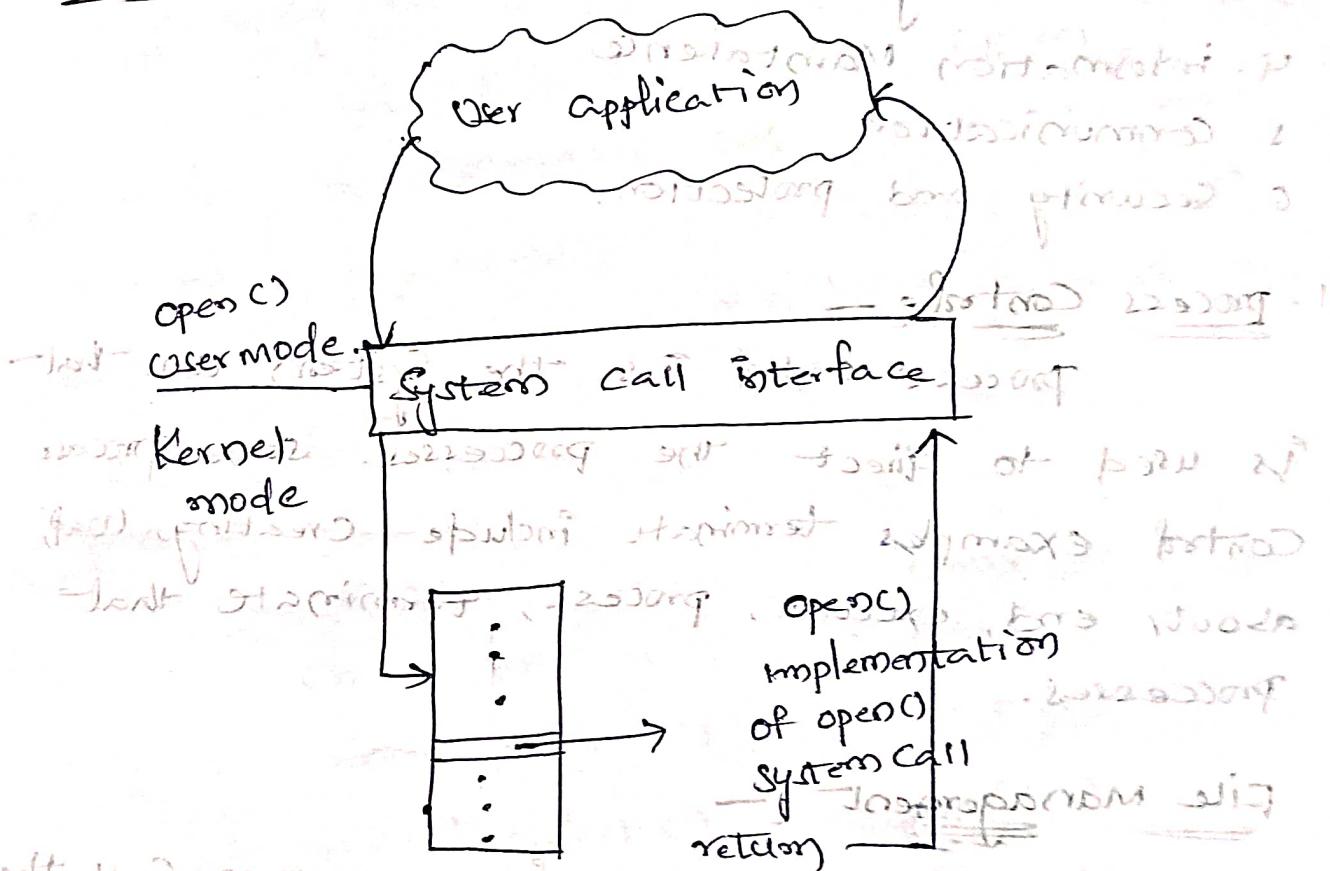
- The program under execution called process at every process system calls are executed.
- The system calls execution starts at, changing the mode bit from 0 to 1 and involving the kernel code.

After completion bit value changes from 1 to 0. This is called context switch.

- The following are the four cases where the operating system make system calls.

1. Creating, closing, deleting the files
2. At every process creation
3. At every new hardware component is attached

- OS has its own API to execute any source language programs.
 - Win 32 API for windows application.
 - JVM API for Java programs.
 - POSIX API for Macintosh.
- API System Calls - OS Relationship



- System calls have a unique identifier that has been indexed in the table.
- Based on the priority of index.

System methods

→ The priority can be 13, 12, 11, etc.

→ For every system method there is a well-defined logic for producing the response to the defined code.

User processing → Error detection

- Sometimes the system calls can take parameters in terms of registers, tables or stack.
- Parameters from task (OS) stack has no limit.

Types of System Calls

1. process Control.

2. File Management

3. Device Management

4. information Maintenance

5. Communication

6. Security and protection.

1. process Control :-

process control is the system call that

is used to direct the processes. Some process control examples terminate include creating, load, about, end, execute, process, terminate that processes.

File Management :-

file management is a system call that

is used to handle files. Some file management examples include creating files, delete files, open, close, read, write etc.

Device Management :-

device management is a system call

that is used to deal with devices. Some examples of device management include

read, device write, get-device-attributes,

release device etc.,

Maintenance

Information :-

Information Maintenance is a system call that is used to maximise information. There are some examples of information maintenance including system data, set time of date, get time of date, set system data etc.,

Communication :-

Communication is a system call that is used for communication. There are some examples of communication including create, delete, communication connections, send receive message etc.,

processes :-

Windows	Linux
Create process()	Fork()
Exit process()	Exit()
Wait for single object()	Wait()
Create file()	Read()
Read file()	Write()
	Open()
Close handle	Close()

Security and protection :-

Protection and security requires that computer resources such as CPU, software, memory etc. are protected. This extends to the OS as well as the data in system.

System Programs :-

* It provides environment to process the user request or services by executing well defined code.

* We know there are two types of software.

1. Application :- program directly interact

with user for its functioning.

2. System :- program that gives service

to another software.

* The system software programs are :-

1. Status information

2. Communication

3. File manipulation

4. File modification

5. Program loading & executing

6. Programming language support

7. Application programs.

1. Status Information :-

It gives the status information

of a drive. That means it gives an

overall idea about the task bar.

e.g. - To check the space availability

in a drive

Let's open drive bar and select the drive status you want and go to properties. Inside of properties we get the total status of the drive we get the information.

2. Communication :-

Communication is used to provide access to system and exchange files and message in text, audio and video format between computers (users).

The communication consists of set of instructions and information.

User to User
User to processor
processor to user

3. File manipulation :-

Firstly we required a file to create already created file. This file we want to perform specific operations on them called as file manipulation. The operations we are going to perform.

1. Create() :- Create is used to create a file.

2. Copy() :- Copy is used to duplicate the same file with some features and properties in another location / same location.

3. Moving() :- Moving is nothing but cut and

Paste that means same file is moved.

from one place to another.

4. File modification :-
The created file is going to perform specific operation inside / internally called as file modification.

The specific operations are :-

1. add :- adding the required data as an internally.

2. open :- To open the file.

3. remove :- to remove unwanted data.

5. program loading + executing -

Placing a program into main storage is called loading the program.

To load and execute a program, use either the LOADGO or CALL command.

* LOADGO command loads object and load modules produced by a compiler or assembler.

and program objects and load modules produced by the linker or linkage editor.

6. program language support -

systems programmers design and write system software.

For ex, they might develop a

Computer OS, such as Mac OS X.

- * Although Java and Python are the great languages for system programming we have to install each language we have to externally and we have to fix path.

7. Application programs :-

A type of programming that is to develop applications that interact with system software (OS) or computer hardware.

Ex:- MS Word, Excel, PowerPoint etc.

OS - Debugging :-

Debugging is the process of finding the problems in a computer system and solving them. This is cost-effective because it has to be solved after deployment.

The operating system performs debugging by using log files.

Log files :-
These record all the events occurred in OS. This is done by writing messages in a file.

There are different types of log files

1. Event logs
2. Transaction logs

3. Message logs

- permanent log and if not responded to log

Event logs :-

These store all the records of events that hit at the time of execution of the events at the system.

— : Logon activity of a system.

Transaction logs :-

These log stores changes of the data so that system can recover from crash and other errors.

Message logs :-

These are public and private, they are mostly plain text files, and some are binary files.

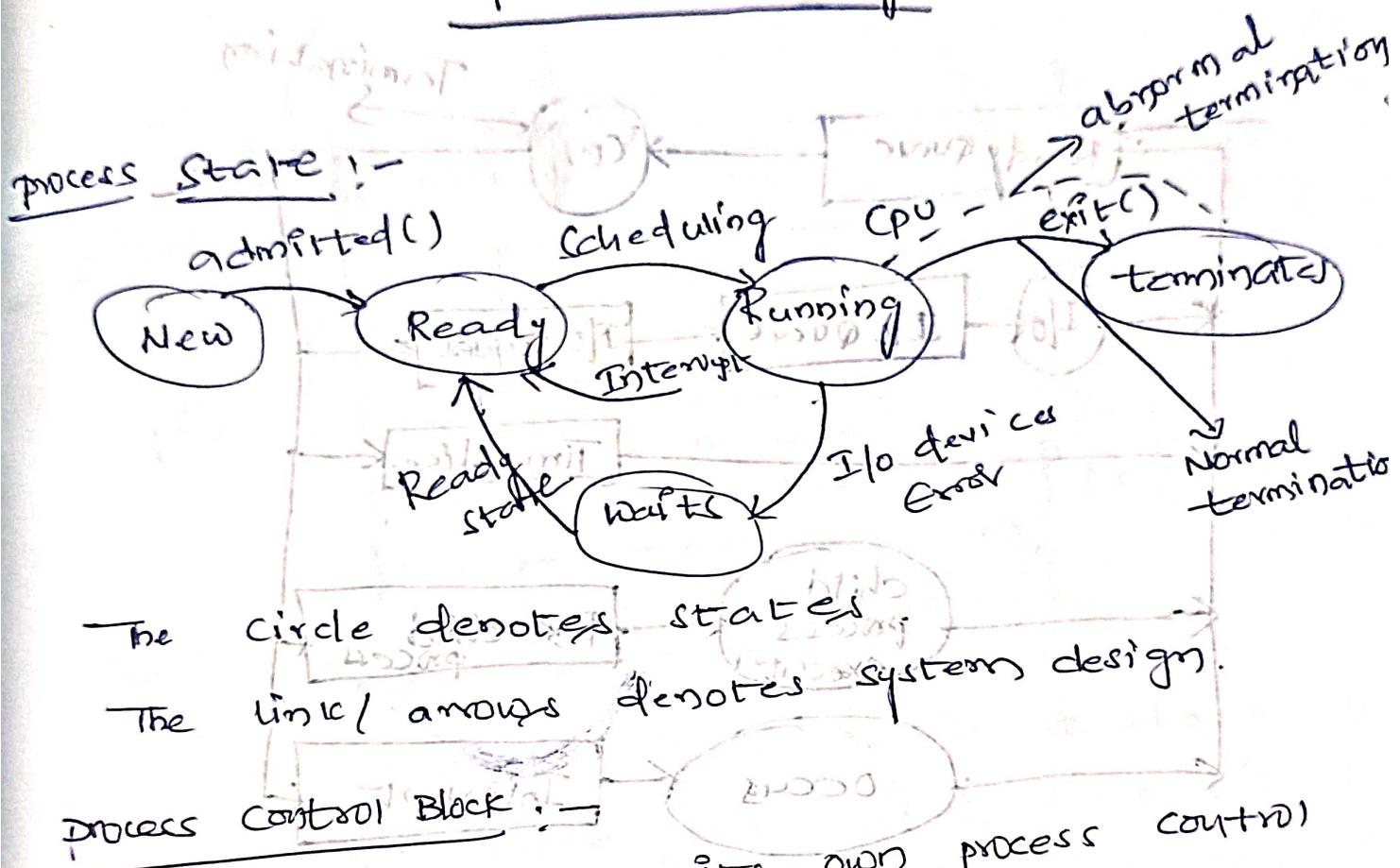
— : Logon activity of a system.

10/03/2025

Unit - II

Process Scheduling

process state :-

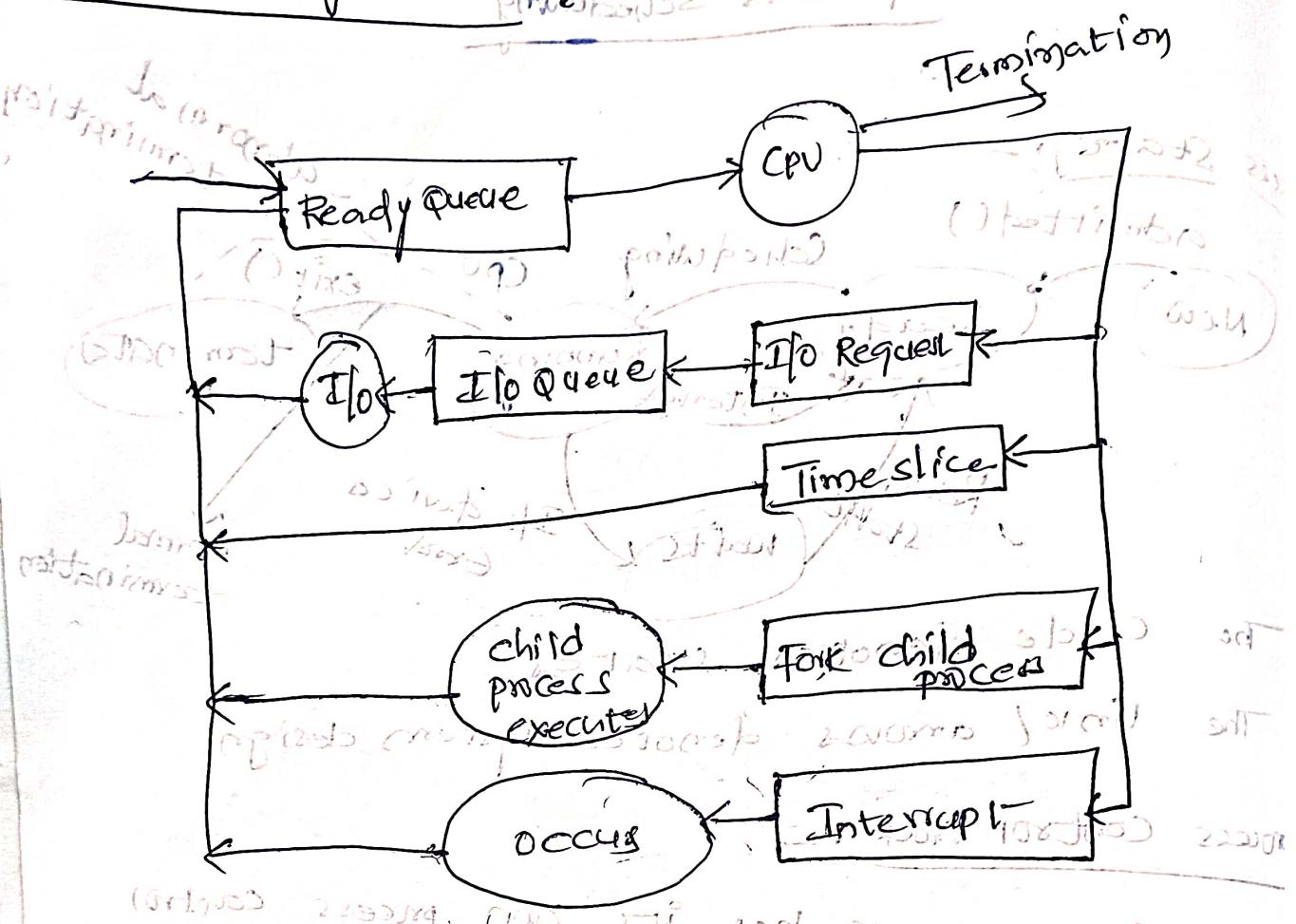


The circle denotes states.
The line/ arrow denotes system design.

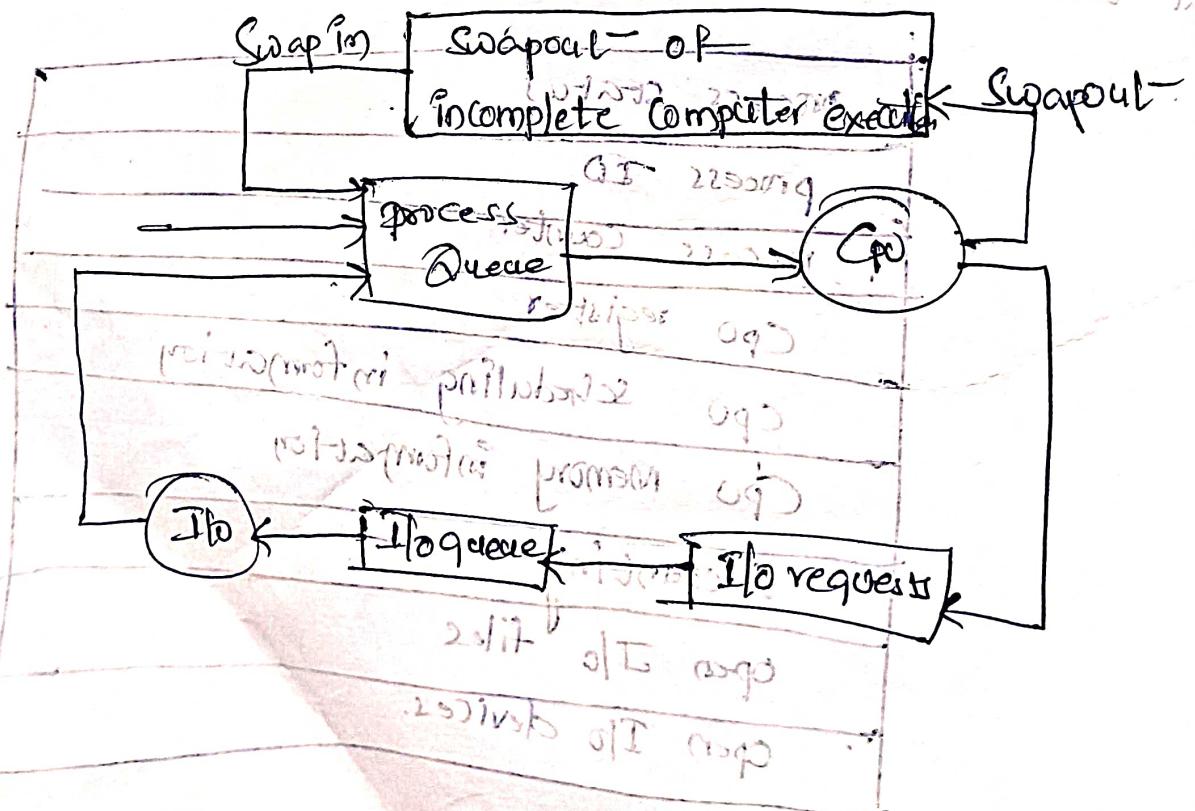
Process Control Block :-
Every process has its own process control block. This block has different types of data structures. This block is related to particular process that all are related to particular process.

process status
process ID
process Counter
CPU register
CPU scheduling information
CPU memory information
Accounting
open I/O files
open I/O devices.

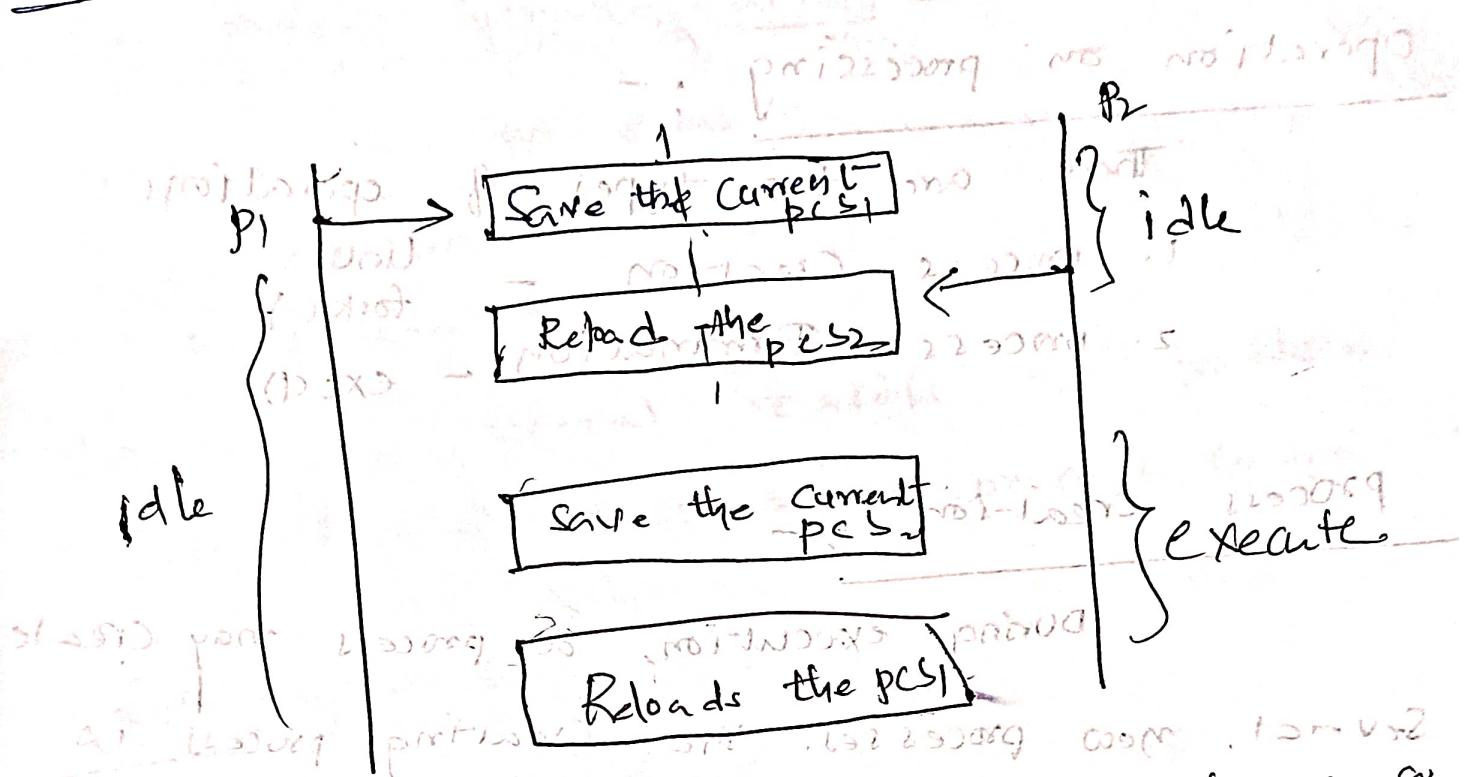
Scheduling, Queues



Schedulers



Context switching :-

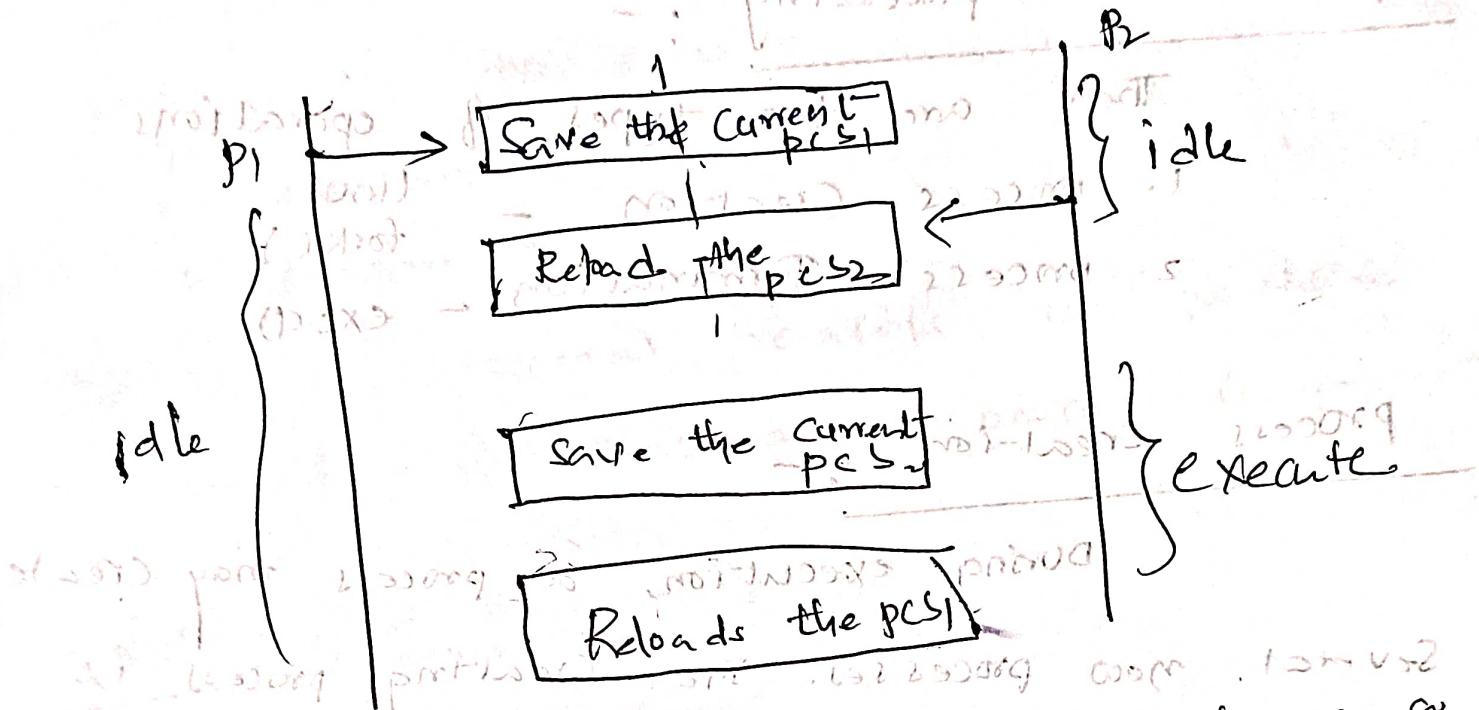


Interruptions cause the of to change a user interrupt and run a kernel. CPU from its current task and run a new task. When an interrupt occurs, the system needs to save the current context of the process running on the CPU so that it can restore that context when its processing is done, basically suspending the process and then resuming it.

The Context (PCB) represented using the PCB of the process which includes the value of CPU register, the process state etc.

Generally we perform a state save and restore operation. User mode and they state restore to resume operations.

Context switching :-



Interruptions cause the OS to change the context of the running task and run a kernel routine - when an interrupt occurs, the system needs to save the current context of the process running on the CPU so that it can restore that context when its processing is done, basically suspending the process and then resuming it. The context (PCB) represented using the PCB of the process which includes the value of CPU register, the process state etc.

Generally we perform a state save of current state of CPU, being Kernel (8) User Mode and they state restore to resume operations.

Process Operation :-

Operations on processing :-

There are two types of operations

1. process Creation -
2. process Termination -

Linux
fork()
exec()

process Creation

During execution, a process may create several new processes. The creating process is

Called parent process and the new process are

Called as child process. If each of the pro-

cesses may in turn create process, forming a tree of processes.

pid < 0 (no child process)

pid = 0 (one child)

pid > 0 (> one child)

process Termination

Process executing concurrently in the system may be either independent processes or co-operating processes. A process

is independent if it cannot affect (d) another process executing

be effected by other processes the system.

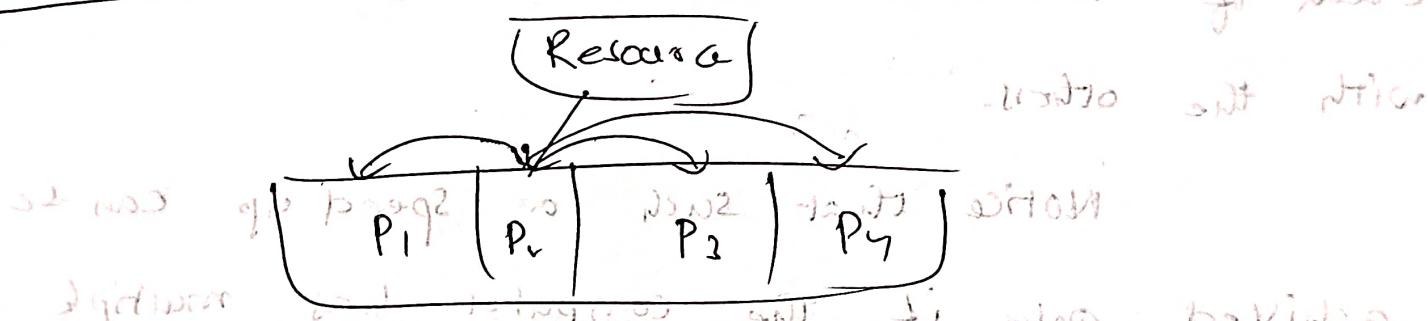
Inter Process Communication :-

Processes are of two types:-

1. Individual user or environment sharing.
2. Co-operation

There are several reasons for providing an environment that allows process co-operation

Information sharing :-



Since several users may be interested in the same piece of information, we must

provide an environment to allow concurrent access to such information.

When the process is cooperative, effective

communication is required between parent & child processes.

Modularity :-

We may want to construct the system in a modular fashion, dividing

the system functions into separate processes (as threads). It decomposes into small addressable sub-processes. whenever modularity occurs, inter-process communication is required.

Computational Speed up

If we want to do a particular task more faster, we must break it into sub-tasks, each of which will be executing in parallel with the others.

Notice that such a speed up can be achieved only if the computer has multiple processing cores.

Speed up process one wants to communicate others.

Convenience

Each individual user may work on many tasks. for ex:- listening mp3, compiling Java programs at the same time.

The ipc convenience is dependent on two factors:- share of memory, message

Communication in client - server environment

In the client - server communication the process may go through the following programs

1. Socket :-

If is the program having the address of both Client & server which is identified by a unique number with IP address & port address there are 1024 standard port address are exists. we have to give more than 1024 as port address.

2. Rpc (Remote procedure calls) :- The OS of

client & server calls the system functions/ routines. When client & server communicating, in between there can be another system may execute its own OS calls.

The following six steps will exists.

Step 1 :- At client side stub (S) proxy call is executed. which means the parameters will be converted into a message called mail

Step 2 :- client send request by calling send system call

Step 3 :- client can call receive methods / function periodically to obtain the response from server

Step 4 :- At server side (a) Server proxy finds it

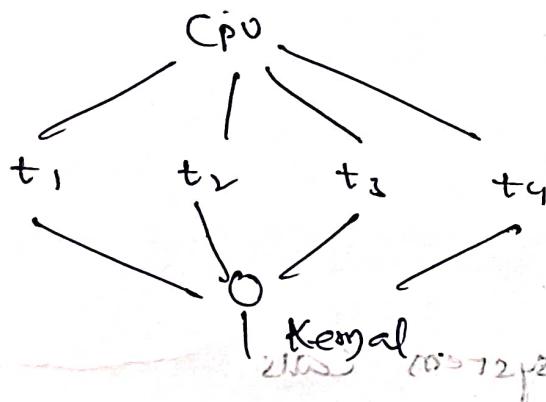
Called → it gathers client information

Step 5 :- Server OS calls a system call to divide the information into parameters & message.

Step 6 :- Server OS calls exec() method to execute the process.

Thread models :-

Many - to - many :-



-: many to many

(OS) (OS) (OS) (OS)

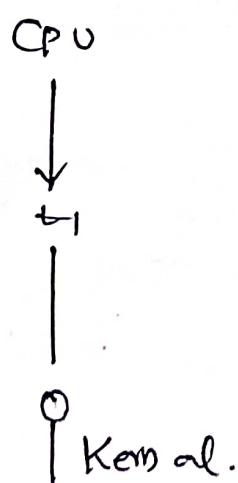
parallel app 1

(parallel). bind 1

parallel local bind 1

parallel process 1

one - to - one :-



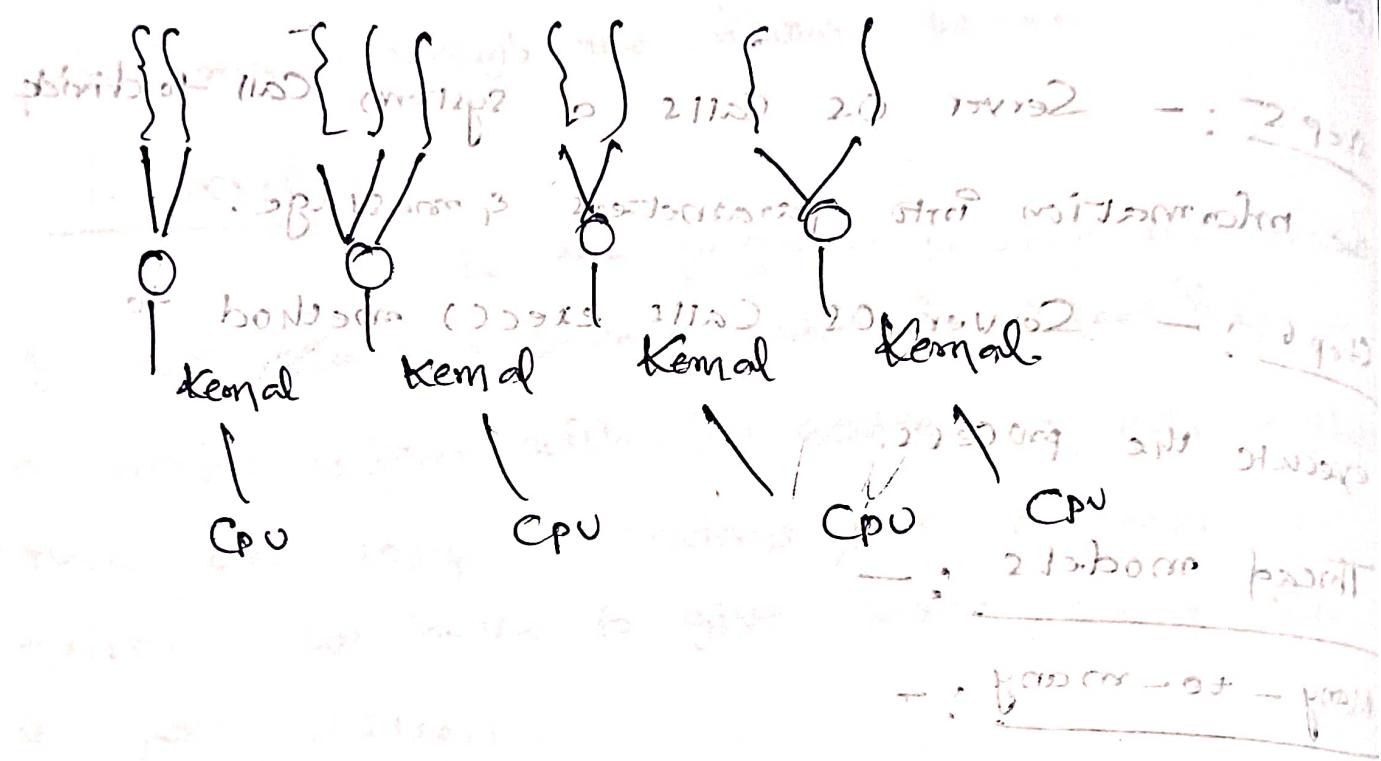
parallel app 2

(parallel). bind 2

parallel local bind 2

parallel process 2

Moving from many to just two threads



1. Pre-emption detection

Kernel to kernel communication

Thread issues

1. fork() 2. exec() system calls

3. Signal handling

4. Thread cancellation

5. Thread local storage

6. Thread cancellations

7. Preemptible threads

8. Thread migration

9. Thread cancellation

10. Thread local storage

11. Thread cancellations

12. Preemptible threads

13. Thread migration

14. Thread cancellations

15. Preemptible threads

16. Thread migration

17. Thread cancellations

18. Preemptible threads

19. Thread migration

20. Thread cancellations

21. Preemptible threads

22. Thread migration

23. Thread cancellations

24. Preemptible threads

25. Thread migration

26. Thread cancellations

27. Preemptible threads

28. Thread migration

29. Thread cancellations

30. Preemptible threads

31. Thread migration

32. Thread cancellations

33. Preemptible threads

34. Thread migration

35. Thread cancellations

36. Preemptible threads

37. Thread migration

38. Thread cancellations

39. Preemptible threads

40. Thread migration

41. Thread cancellations

42. Preemptible threads

43. Thread migration

44. Thread cancellations

45. Preemptible threads

46. Thread migration

47. Thread cancellations

48. Preemptible threads

49. Thread migration

50. Thread cancellations

51. Preemptible threads

52. Thread migration

53. Thread cancellations

54. Preemptible threads

55. Thread migration

56. Thread cancellations

57. Preemptible threads

58. Thread migration

59. Thread cancellations

60. Preemptible threads

61. Thread migration

62. Thread cancellations

63. Preemptible threads

64. Thread migration

65. Thread cancellations

66. Preemptible threads

67. Thread migration

68. Thread cancellations

69. Preemptible threads

70. Thread migration

71. Thread cancellations

72. Preemptible threads

73. Thread migration

74. Thread cancellations

75. Preemptible threads

76. Thread migration

77. Thread cancellations

78. Preemptible threads

79. Thread migration

80. Thread cancellations

81. Preemptible threads

82. Thread migration

83. Thread cancellations

84. Preemptible threads

85. Thread migration

86. Thread cancellations

87. Preemptible threads

88. Thread migration

89. Thread cancellations

90. Preemptible threads

91. Thread migration

92. Thread cancellations

93. Preemptible threads

94. Thread migration

95. Thread cancellations

96. Preemptible threads

97. Thread migration

98. Thread cancellations

99. Preemptible threads

100. Thread migration

101. Thread cancellations

102. Preemptible threads

103. Thread migration

104. Thread cancellations

105. Preemptible threads

106. Thread migration

107. Thread cancellations

108. Preemptible threads

109. Thread migration

110. Thread cancellations

111. Preemptible threads

112. Thread migration

113. Thread cancellations

114. Preemptible threads

115. Thread migration

116. Thread cancellations

117. Preemptible threads

118. Thread migration

119. Thread cancellations

120. Preemptible threads

121. Thread migration

122. Thread cancellations

123. Preemptible threads

124. Thread migration

125. Thread cancellations

126. Preemptible threads

127. Thread migration

128. Thread cancellations

129. Preemptible threads

130. Thread migration

131. Thread cancellations

132. Preemptible threads

133. Thread migration

134. Thread cancellations

135. Preemptible threads

136. Thread migration

137. Thread cancellations

138. Preemptible threads

139. Thread migration

140. Thread cancellations

141. Preemptible threads

142. Thread migration

143. Thread cancellations

144. Preemptible threads

145. Thread migration

146. Thread cancellations

147. Preemptible threads

148. Thread migration

149. Thread cancellations

150. Preemptible threads

151. Thread migration

152. Thread cancellations

153. Preemptible threads

154. Thread migration

155. Thread cancellations

156. Preemptible threads

157. Thread migration

158. Thread cancellations

159. Preemptible threads

160. Thread migration

161. Thread cancellations

162. Preemptible threads

163. Thread migration

164. Thread cancellations

165. Preemptible threads

166. Thread migration

167. Thread cancellations

168. Preemptible threads

169. Thread migration

170. Thread cancellations

171. Preemptible threads

172. Thread migration

173. Thread cancellations

174. Preemptible threads

175. Thread migration

176. Thread cancellations

177. Preemptible threads

178. Thread migration

179. Thread cancellations

180. Preemptible threads

181. Thread migration

182. Thread cancellations

183. Preemptible threads

184. Thread migration

185. Thread cancellations

186. Preemptible threads

187. Thread migration

188. Thread cancellations

189. Preemptible threads

190. Thread migration

191. Thread cancellations

192. Preemptible threads

193. Thread migration

194. Thread cancellations

195. Preemptible threads

196. Thread migration

197. Thread cancellations

198. Preemptible threads

199. Thread migration

200. Thread cancellations

201. Preemptible threads

202. Thread migration

203. Thread cancellations

204. Preemptible threads

205. Thread migration

206. Thread cancellations

207. Preemptible threads

208. Thread migration

209. Thread cancellations

210. Preemptible threads

211. Thread migration

212. Thread cancellations

213. Preemptible threads

214. Thread migration

215. Thread cancellations

216. Preemptible threads

217. Thread migration

218. Thread cancellations

219. Preemptible threads

220. Thread migration

221. Thread cancellations

222. Preemptible threads

223. Thread migration

224. Thread cancellations

225. Preemptible threads

226. Thread migration

227. Thread cancellations

228. Preemptible threads

229. Thread migration

230. Thread cancellations

231. Preemptible threads

232. Thread migration

233. Thread cancellations

234. Preemptible threads

235. Thread migration

236. Thread cancellations

237. Preemptible threads

238. Thread migration

239. Thread cancellations

240. Preemptible threads

241. Thread migration

242. Thread cancellations

243. Preemptible threads

244. Thread migration

245. Thread cancellations

246. Preemptible threads

247. Thread migration

248. Thread cancellations

249. Preemptible threads

250. Thread migration

251. Thread cancellations

252. Preemptible threads

253. Thread migration

254. Thread cancellations

255. Preemptible threads

256. Thread migration

257. Thread cancellations

258. Preemptible threads

259. Thread migration

260. Thread cancellations

261. Preemptible threads

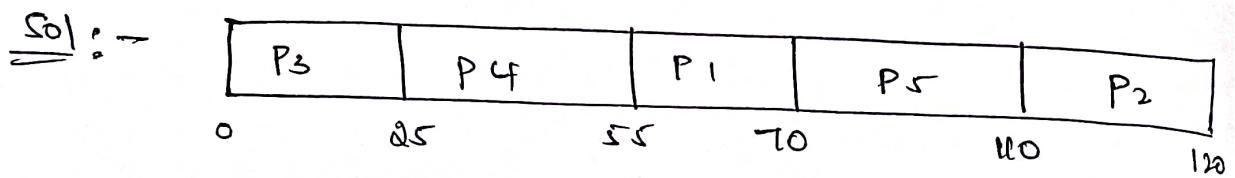
262. Thread migration

263. Thread cancellations

* To calculate Average waiting Time Using Gant chart of following

PID	B.T
P ₁	15
P ₂	10
P ₃	25
P ₄	30
P ₅	40

order :- P₃ P₄ P₁ P₅ P₂
P₂ P₄ P₅ P₁ P₃



Waiting Time of

$$P_2 \rightarrow 0$$

$$P_4 \rightarrow 25$$

$$P_1 \rightarrow 55$$

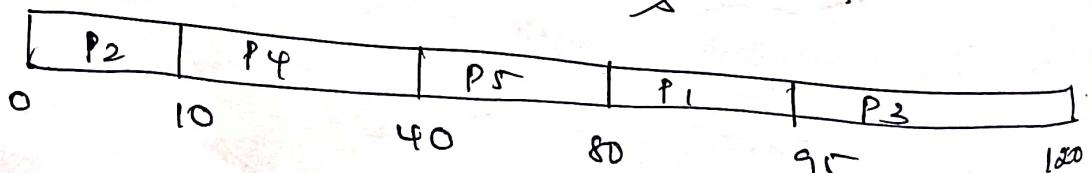
$$P_5 \rightarrow 70$$

$$P_2 \rightarrow 110$$

$$\text{Average Waiting Time} = \frac{0 + 25 + 55 + 70 + 110}{5}$$

$$= \frac{260}{5} = 52.$$

(ii)



Waiting Time of

$$P_2 \rightarrow 0, P_4 \rightarrow 10, P_5 \rightarrow 40, P_1 \rightarrow 80, P_3 \rightarrow 95$$

$$\text{Average Waiting Time} = \frac{0 + 10 + 40 + 80 + 95}{5}$$

$$= \frac{225}{5} = 45$$

Shortest first job scheduling (SJF)

This algorithm allocates the CPU to the process which has the shortest execution time. It will be either preemptive (S) or non-preemptive.

In non-preemptive / cooperative, the process must be completed in its execution so no process will be interrupt the current execution process. But in preemptive the current executing process may go to waiting state and new process will be allocated where the burst time is less than the executing process's burst time with allocate the CPU.

In preemptive, the algorithm works on SJF Non-preemptive:

SJF Non-preemptive :-

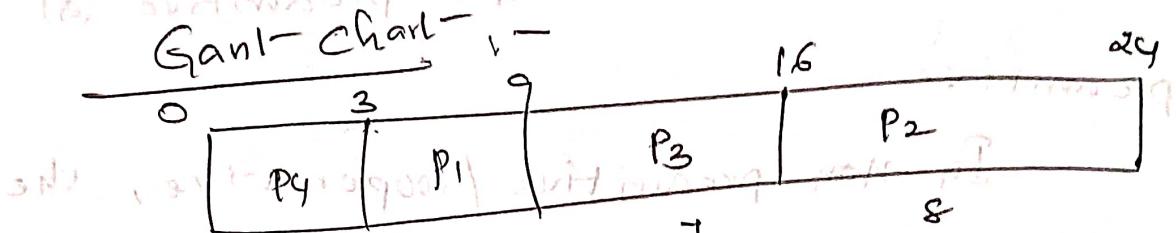
Consider the following 4 processes which are arrived at same time where the burst time is 6, 8, 7, 3 respectively. Calculate the average waiting time.

P ₁	Burst time
P ₁	6
P ₂	8
P ₃	7
P ₄	3

Gant chart

To calculate average waiting time we
should form the Gant chart

(Q) scheduling problem for NBS 10



Waiting time for P₁ = 3 ms

Waiting time for P₂ = 16 ms

Waiting time for P₃ = 9 ms

Waiting time for P₄ = 0 ms

Average waiting time = Sum of individual

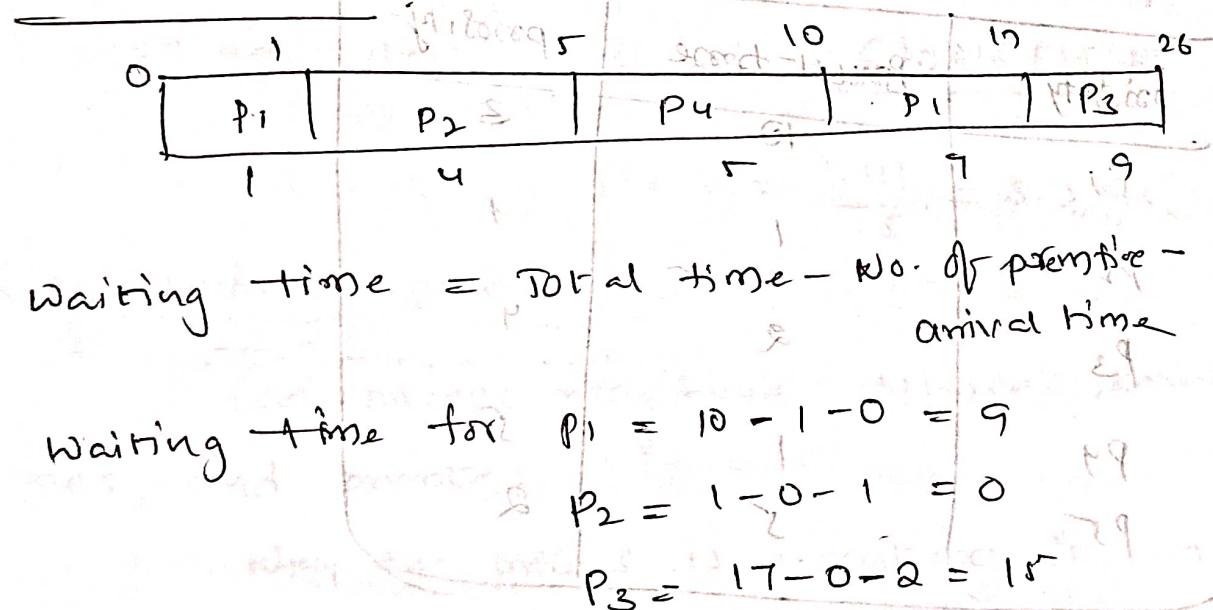
$$= \frac{3+16+9+0}{4} = \frac{28}{4} = 7 \text{ ms}$$

SJF preemtive

PTD	Arrival Time	Burst Time
P ₁	0	8
P ₂	9	7
P ₃	16	4
P ₄	24	5

Sol	PID	Arrival Time	Burst Time
	P ₁	0	8 - 1 = 7
	P ₂	1	4 - 1 = 3
(a)	P ₃	2	9 - 2 = 7
	P ₄	3	5 - 3 = 2

Gant chart:-



$$\text{Waiting time} = \text{Total time} - \text{No. of process} - \text{arrival time}$$

$$\text{Waiting time for } P_1 = 10 - 1 - 0 = 9$$

$$P_2 = 1 - 0 - 1 = 0$$

$$P_3 = 17 - 0 - 2 = 15$$

$$P_4 = 5 - 0 - 3 = 2$$

$$\text{Average waiting time} = \frac{\text{sum of individual w.r.t. processes}}{\text{No. of processes}}$$

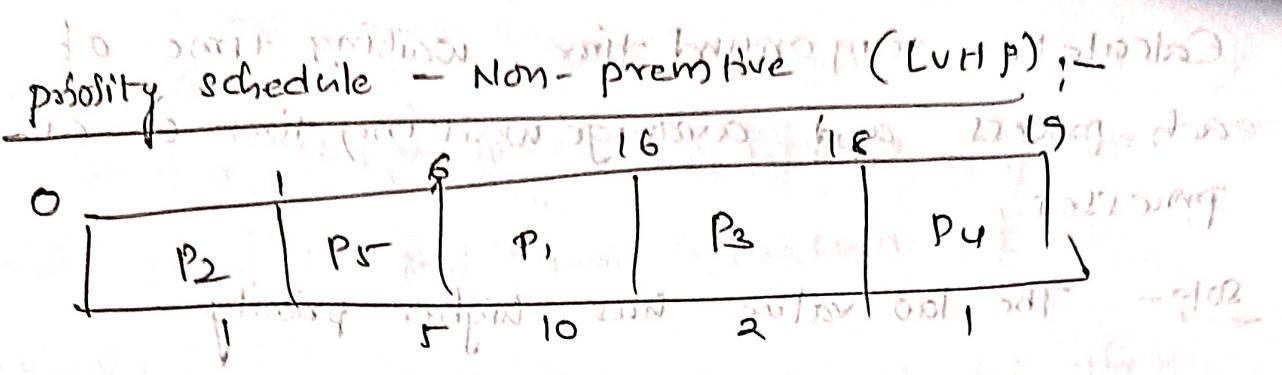
$$\text{Average waiting time} = \frac{9 + 0 + 15 + 2}{4} = \frac{26}{4} = 6.50$$

Priority :-

- Another algorithm where the process processes priority.
- each processes have a ~~have~~ priority value.
- This algorithm may have different approach like the lowest value have high priority & the highest value can be the high priority.

process	burst-time	priority	turn-around
P ₁	10	3	10
P ₂	1	1	1
P ₃	2	4	2
P ₄	1	1	1
P ₅	5	2	5
	20		20

- This may be ~~be~~ pre-emptive or non-pre-emptive.
- In non-pre-emptive the process can't be pre-empted till the process is completed its execution.
- In pre-emptive processes with high-p priority will allocate CPU then the executed process priority



Waiting time for $P_1 = 6$.

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

Average waiting Time = $\frac{6+0+16+18+1}{5}$

$$= 10 \text{ ms}$$

Priority scheduling - primitives :-

Each process may have different arrival times and priorities.

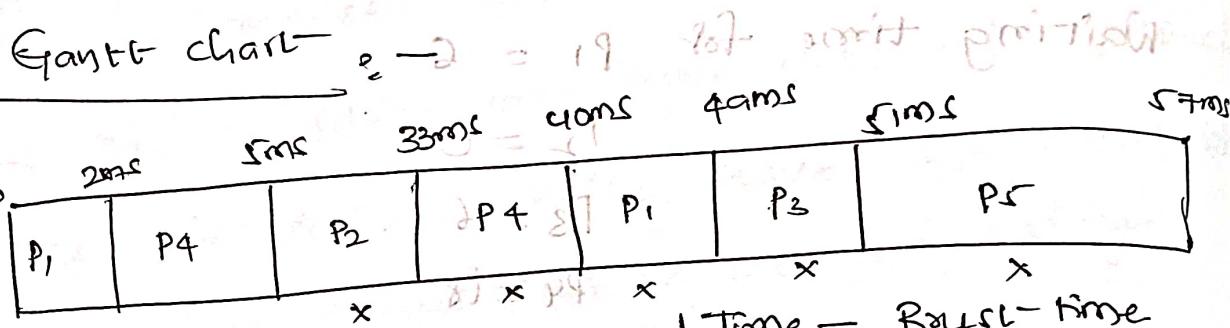
When one process is executing it can be pre-empted by another process for higher priority process. And the current execution process will be in the de-queue.

Consider the following processes with their arrival time, burst time and priority (low value is high priority).

process ID	Arrival Time	Burst Time	Priority
P_1	0	11	2
P_2	5	28	0
P_3	12	2	3
P_4	2	10	1

Calculate turn around time, waiting time of each process and average waiting time of all processes.

Sol: - The low value has higher priority



$$\text{Waiting Time} = \text{Turnaround Time} - \text{Burst Time}$$

$$\text{Turnaround Time} = \text{Complete Time} - \text{AT}$$

$$\text{Turnaround Time of } P_1 = 49 - 0 = 49$$

$$\text{Turnaround Time of } P_2 = 33 - 5 = 28$$

$$\text{Turnaround Time of } P_3 = 51 - 12 = 39$$

$$\text{Turnaround Time of } P_4 = 40 - 2 = 38$$

$$\text{Waiting time for } P_1 = 49 - 41 = 8$$

$$\text{Waiting time for } P_2 = 28 - 28 = 0$$

$$\text{Waiting time for } P_3 = 39 - 2 = 37$$

$$\text{Waiting time for } P_4 = 38 - 10 = 28$$

$$\text{Waiting time for } P_5 = 48 - 6 = 42$$

$$\text{Average Waiting Time} = \frac{28 + 0 + 37 + 28 + 42}{5}$$

$$= \frac{145}{5} = 29$$

Round Robin :- It allocates CPU quantum (equally)

It is similar to FCFS algorithm where preemption and time quantum of CPU is added.

This algorithm is applicable for time sharing operating systems.

In this algorithm CPU gives a time slice (Δ) time quantum for each process.

When burst time < Time Quantum, the process itself terminates and CPU allocation goes to the next process.

When burst time > Time Quantum, the preempted process is added at tail of the ready queue.

When preempted process is adding to the queue at the same time new process is arrived they at tail new process is added.

Added

Process ID	Burst time	TAT = Complete - AT
P ₁	24	Waiting = TAT - BT
P ₂	7	Turnaround time = TAT + waiting
P ₃	10	Completion time = BT + waiting

$$TAT \text{ of } P_1 = 30 - 0 = 30$$

$$P_2 = 7 - 0 = 7$$

$$P_3 = 10 - 0 = 10$$

$$WT \text{ of } P_1 = 30 - 24 = 6$$

$$P_2 = 7 - 3 = 4$$

$$P_3 = 10 - 3 = 7$$

$$\text{avg} = \frac{6+4+7}{3}$$

$$= \frac{17}{3} = 5.6667$$

Waiting time for each process

$$WT = \text{Arrival Time} - \text{Last Started Process Time}$$

(No. of preempted $\times TQ$)

$$WT \text{ of } P_1 = 26 - 0 - (5 \times 4)$$
$$= 26 - 20 = 6$$

$$WT \text{ of } P_2 = 4 - 0 = 4$$

$$WT \text{ of } P_3 = 7 - 0 = 7$$

Round Robin algorithm for different arrival time

Process

In this approach all processes are arranged as circular queue.

For each time quantum CPU makes an alarm, preempts the executing process and allocation will be given to head of the ready queue process.

AT one time (8) at the same time if the queue has new process and preempted process then head position is occupied by new process behind that - preempted process.

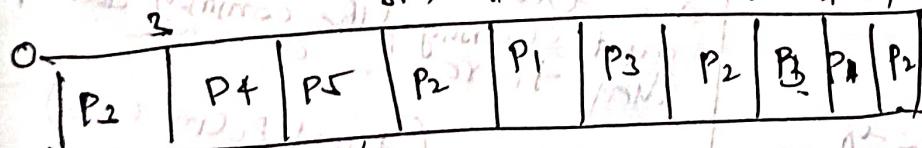
Consider the following process with their arrival times and burst times. The time quantum of the CPU is 2ms. Calculate the average waiting time of all the processes.

Process ID	Arrival Time	Burst Time	Waiting Time	Turnaround Time
P ₁	10	5	0	10
P ₂	11	3	1	14
P ₃	12	2	2	16
P ₄	13	3	3	19
P ₅	14	3	4	22

Round robin scheduling algorithm

Process ID	Arrival Time	Burst Time
P ₁	10	5
P ₂	11	3
P ₃	12	2
P ₄	13	3
P ₅	14	3

Time Quantum = 3 ms



Time 0 = [P₂]

Time 1 = [P₄]

Time 2 = [P₅, P₂]

Time 3 = [P₃, P₄, P₅]

Time 4 = [P₁, P₂, P₅]

Time 5 = [P₂, P₃, P₁, P₂, P₃]

Calculate turnaround time, waiting time and average waiting time

$$\text{turnaround time of } P_1 = CT - AT \\ = 24 - 10 = 14$$

$$P_2 = 26 - 2 = 24$$

$$P_3 = 24 - 0 = 24$$

$$P_4 = 24 - 1 = 23$$

$$P_5 = 24 - 2 = 22$$

Waiting time of process = TAT - BT

Waiting time of process, $P_1 = 15 - 6 = 9$

$$P_2 = 21 - 8 = 13$$

$$P_3 = 23 - 7 = 16$$

$$P_4 = 28 - 3 = 25$$

$$P_5 = 4 - 2 = 2$$

Average waiting time

$$= \frac{9+13+16+1+2}{5}$$

$$\frac{12+15+14+2+4}{5}$$

$$\frac{59}{5} = 11.8$$

process synchronization \rightarrow It's used to waiting all remaining processes in wait state until it completes its execution process.

Individual
main()

int a

$a = a + 10$

co-operation

int a

$a = 10$

$a = a + 10$

func()

decrements

fun()

$a = a - 5$

performs

The change

in main

function

affect the

func() value

Critical Section

If two process are executing parallelly there is no problem. But the process

are executing simultaneously and sharing the common data then there is a race condition.

Condition (Race condition)

In this case inconsistency will occur.