~: Introduction To Operating Systems :~

1.1 what operating systems do?

Operating System is a Program that manages the computer H/w. It acts as a bridge between user & computer H/w.

operating System Provides an environment within which other Program can do useful wor

The main goal of operating system i to Provide the efficient

1) Process management
2) memory management
3) Resource allocation
4) Provides protection & Security
5) File management

From user view the O.S is designed for ease of use.

From system view, the O.S ~~is the~~ as resource allocator. O.S has to manage CPU time, memory space, file - storage space, I/O devices, O.S acts as manager of these ~~[illegible]~~ resources.

Imp (M

1.2 Computer-System Architecture :-

The computer system architecture can be categorized according to the number of processors used.

1) single-processor systems
2) multiprocessor systems
3) clustered systems.

Single Processor Systems :-

on a single processor system there is one main cpu capable of executing a general-purpose instruction set, including instructions from the user processes.

All most all system have special-purpose processors, they are device specific processors, such as disk, keyboard, & graphics controllers.

special - purpose processors run a limited instruction set and do not run user processes. special-purpose processors are monitored & managed by operating system.

Ex:- ① PC contain microprocessor in the keyboard to convert the keystrokes into codes to be sent to the cpu.

Ex:② A Disk controller microprocessor recieves a sequence of requests from the main cpu & implements its own disk queue & scheduling algorithm..

Multiprocessor Systems :-

Multiprocessor systems have two or more processors, sharing computer bus ~~&~~, clock, memory, and peripheral devices

** V.V. Imp. (MSE)
Advantages of Multiprocessor Systems

1) Increased throughput :-

By increasing number of processors work will be done in less time. (The speed-up ratio with 'N' processors is not 'N', it is less than 'N'.

2) Economy of scale :-

multi processor systems can cost less than ~~expand~~ equivalent multiple single processor systems, because they share peripherals, mass storage, & power supplies

3) Increased reliability :-
If functions can be distributed properly among several processors, then the failure of one processor will not halt the system, only slow it down. If one processor fails out of ten, then remaining nine processors share the work of failed processor.

There are two types of multiple processor system ① Asymmetric multiprocessing.
② Symmetric multiprocessing.

Asymmetric multiprocessing :- In which each processor is assigned a specific task. A master processor controls the system, other processors looks for master instruction.

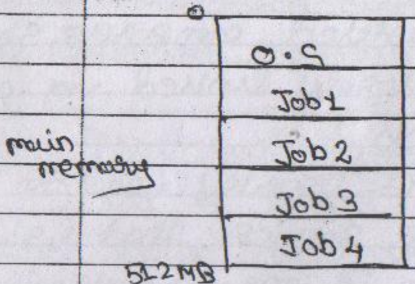
Symmetric multiprocessing :- Here each processor performs all tasks within the O.S. All processors are independent no master-slave relationship exists between processors.

MSP
Clustered Systems :-
Clustered systems gather together multiple CPU'S to accomplish computational work.
Defination :- ~~the [illegible]~~
Clustered computers share storage & are closely linked via a local-area network.
Clustering is usually used to provide high-availability service; that is, service will continue even if one or more systems in the cluster fails. Cluster S/w runs on each clustered system (clustered nodes) Each node monitors one or more nodes. If monitored machine fails, then monitoring machine can take ownership & restart the applications that were running on failed machine

1.3 Operating System structure :-

A single user program cannot keep CPU or I/O devices busy at all times. Multiprogramming increases CPU utilization by organizing Jobs, So that CPU always has one to execute.

In multiprogramming the OS keeps several jobs in memory



The operating system picks Jobs & begins to execute one of the jobs in memory. The job may have to wait for some task, such as an I/O operation, to complete. In non-multiprogrammed system, the CPU would remain idle. But in multiprogrammed system, the O.S simply switches to and executes another job. When that job needs to wait the CPU is switched to another job & so on. As Job1 finishes waiting & gets the CPU back. In multiprogramming the CPU is never idle.

Time sharing (or multitasking) :-

Timed Shared System is extension of multiprogramming. In time sharing systems, the CPU executes multiple jobs by switching among them, but the switching occurs so frequently that the users can interact with each program while it is running.

As the time switches rapidly from one user to the next, each user is given the impression that the entire computer is dedicated to his use, even though it is shared among many users.

1.4 Operating System Operations :-

The O.S offers 2 modes of operations for the given computer H/w.

1) user mode
2) kernel mode.

Therefore O.S is called as Dual-mode operation. The computer system can be in user mode of execution or kernel mode of execution. A mode bit helps computer H/w to distinguish between the modes.

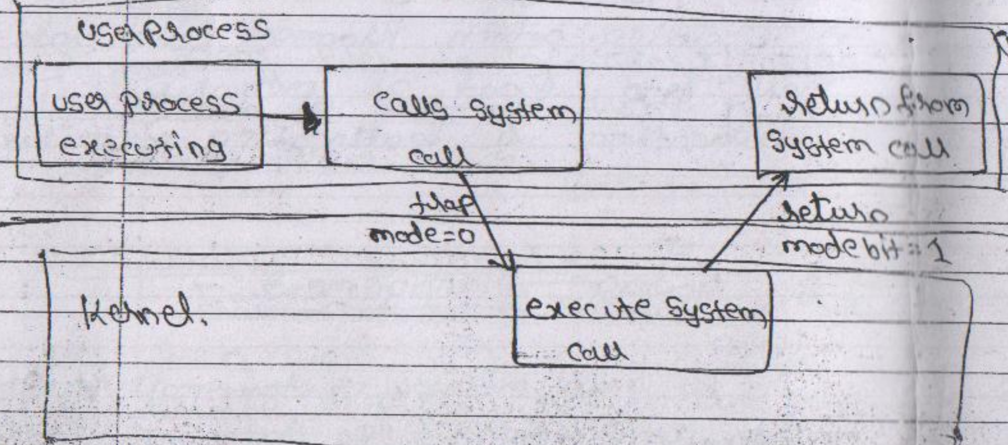
If mode bit = 0, the computer is in kernel mode. If mode bit = 1 the computer is in user mode.

When booting takes place the system is in kernel mode.

when the computer is in user mode, the CPU is executing user applications.

while executing user program may request some services of the O.S via system call. A system call can be treated as S/W generated interrupt.

Now the transition or switching takes place from user mode of execution to (supervisor) kernel mode by resetting mode bit to zero. At the end of execution system call again switching occurs from kernel mode of execution to user mode execution via setting the mode bit = 1.



Transition from user to kernel mode

Advantages of Dual modes operations:-

1) It protects the O.S from accidental damage if any from user processes.
2) Dual mode operation offers (privileged) instructions to access operating system via system call.

Imp

(1.5) Process management :-

Program under execution is defined a Process.

Process management refers to processor management. Process management is to allocate processes to a processor

Process is program under execution, which needs certain resources - like CPU time, memory, files, & I/O devices. to get work done.

The process can be in one the following states ① Ready
② Running
③ Blocked.

when process ~~get CPU (for execution)~~ is waiting for CPU, then it will be in Ready state.

once the process gets CPU it is in Running state.

A system consists of collection of processes. all these processes can execute concurrently by switching CPU among the processes.

The O.S is responsible for the following activities with process management.

1) creation & deletion of user & system process
2) suspending & resuming process
3) providing mechanism for ① process synchronization
② process communication
③ deadlock handling.

1.6. Memory mangement :~

main memory is repositary (storage area) of quickly accessible data.
Memory is large array of words or bytes ranging in size from hundreds of thousands to billions. Each word or byte has its own address.

The processors reads instruction from memory during the instruction fetch cycle & reads & writes data from main memory during data fetch cycles.

For the program to executed, it must be loaded onto memory, as prog executes it access program instruction & data from memory. as execution completes the main memory will be freed. So that next program can be loaded & executed

The O.S is responsible for the following activities in connection with main memory management.
1) Keeping track of which parts of memory are currently being used & by whom,
2) Deciding which processes and data to move into & out of memory
3) Allocating & deallocating memory space as needed.

1.8 Storage Management :-

As main memory is too small to accomodate all data & programs. So data is stored on files & storage medias like disks, tape etc

File-~~System~~ system management :-

File is a collection of related information defined by creator. Files may contain Programs & data. Data files may be numeric, alphabetic, alphanumeric or binary.

Files are organized into directories. when a Particular file is used by multiple users then ~~a~~ File accessing must be controlled like read operation, write operation or append operation.

The O.S is responsible for following activities with file management.

1) creating & deleting files
2) creating & deleting directories to organize files
3) mapping files to secondary storage
4) Backing up files on stable (non volatile) storage media.

Mass-storage management :-

main memory is volatile. i.e., data are lost when Power is lost. So computer System must Provide secondary storage to backup main memory

System Programs such as compilers Assemblers, Text editors, Loaders, Linker including O.S Programs Permanently reside on disk medium and loaded to main memory while execution of user Programs

The Proper management of Disk storage is of central importance to a computer.

O.S is responsible for the following activities with mass storage management.

1) Free-space management.
2) ~~storage~~ storage allocation.
3) Disk scheduling.

1.9 Protection & Security :-

Protection is mechanism for controlling the access of Processes or users to the resources defined by a computer system.

Protection can improve reliability of the system by Protecting $\overset{\text{computer}}{\uparrow}$ system against unauthorised users

O.S must Provide memory Protection-when Process running in one Portion, must not modify the memory locations of other Portion

O.S must provide the following security mechanisms.

1) Provide a controlled access to resources of a computer system

2) Provide mechanism for detecting hidden or dormant errors of the interfaces of component subsystems.

3) Provide protection schemes to distinguish between authorised & unauthorised ~~accessing~~ accesses to the resources of computer system.

Security system should safeguard the computer system both from external & internal attacks. Attacks may be due to viruses, worms & denial of services or refusing access to authorised users.

O.S maintains list of user names and its user identifiers (user ID). These ID's are unique to each user. when a user logs in to the system, the authentication stage determines the appropriate user ID for the user.

1.4 Distributed system:-

It is a model in which components located on networked computers communicate & coordinat their actions by passing messages. The components interact with each other in order to achieve a comm on goal.

chapter-2

Process management.

Process concept:-

(2.1) Process is defined as Program under execution. Process can be referred to a Task. Process represents user job.

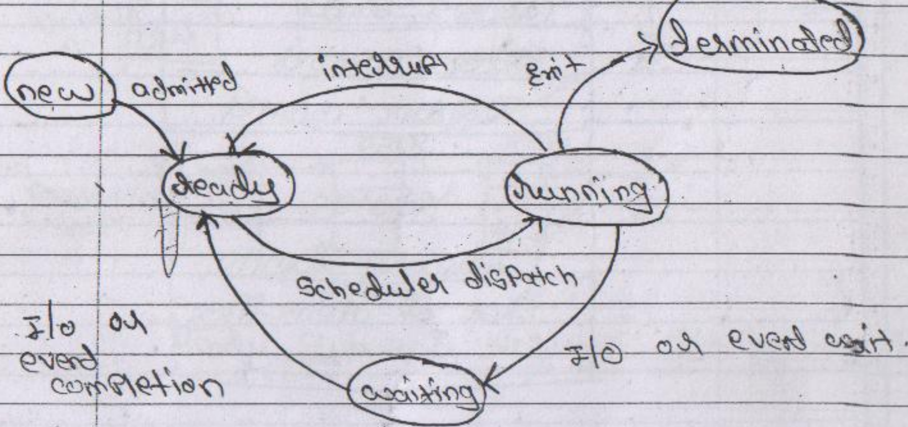
Difference Between Process and Program

Process	Program
1) An active entity residing in main memory	(1) A passive entity residing on harddisk
2) Finite sequence of instructions already in execution.	(2) Finite sequence of instructions to be called for execution
3) It can be in any one of the ready, running or blocked states	(3) It is always in dormant state.

A Process consists of Program code, which is called as text section. It also includes the current activity, represented as Program counter. The Process includes, Stack which contains temporary data. and data Section which contains global variables. A Process also contains a heap which is memory that is dynamically allocated during run time.

Process State :-

As Process executes it changes its State. The State of a Process is defined in Part by current activity of that Process. Each Process may be in one of the following states:



* New: The Process is being created
* Running : Instruction are being executed
* waiting : The Process is waiting for some event to occur (such as an I/o completion or reception signal).
* ready :- The Process is waiting to be assigned to a processor
* terminated : The Process has finished execution.

Process control Block:-

Each Process is represented in the O.S by the Process control Block (PCB) -also called as (task) control block.

Process State
Process number
Program counter
registers
memory limits
list of open files
......

1) Process State:- The state may be in new, ready, running, waiting, halted, and so on.

2) Program counter:- The counter indicates the address of (next instruction) to be executed.

3) CPU registers:- The registers vary in number & type, depending on computer architecture. They include accumulators, index registers, stack pointers, & general-purpose registers.

4) CPU-Scheduling information:-

This information includes a process priority, pointers to scheduling queues, & other scheduling parameters.

5) memory management Information:-

This information may include ~~the~~ the value of the base and limit registers, the page table or segment tables depending on the memory system used by operating system.

6) Accounting information:-

This information includes the amount of CPU and real time used, time limits, accounts numbers, job or process number, and so on.

7) I/O Status Information:-

This information includes the list of I/O devices allocated to the Process, a list of open files, and so on.

⟨2.2⟩ Process Scheduling:-

The objective of multi Programming is to have some Process running at all times, to maximize CPU utilization. The objective of time sharing is to switch the CPU among Processes so frequently the user can interact with each Program while it is running.

~~To meet these objectives,~~ the Process scheduler selects an available Process, If there are more then ~~one~~ ~~running~~ processes have to wait until the CPU is free and can be rescheduled.

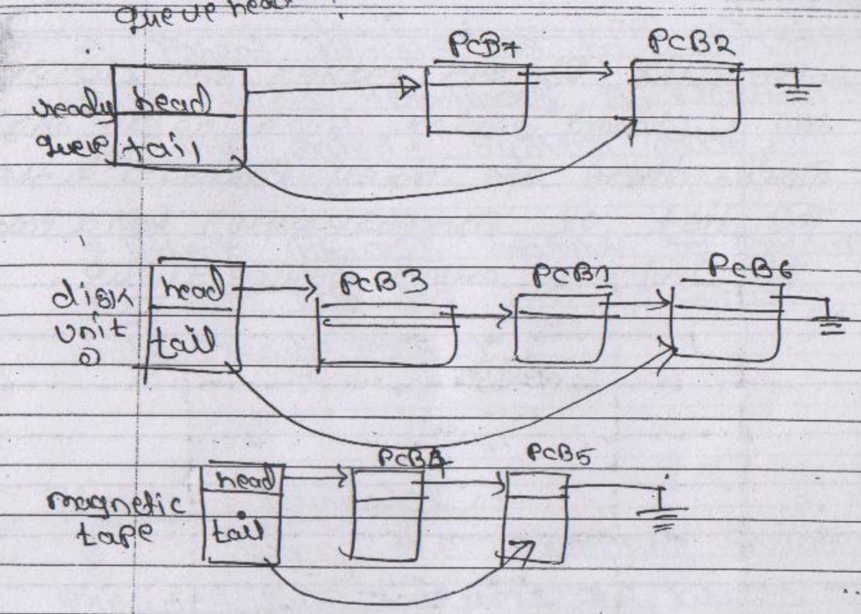
Scheduling Queues :~

→ As the process enter the System they are put into a job Queue which contains all the processes in the system.

→ The Processes that are residing in main memory and are ready & waiting to execute are kept on a list called ready Queue

⇒ The ready queue is generally stored as linked list. The ready queue header contains Pointer to the first & final PCB in the list.

⇒ Each PCB includes a pointer that Points to next PCB in the ready Queue.



"Ready Queue & various I/O devices Queue"

The System also includes other queues. when Process is allocates the CPU, it executes for while & eventually quits, as interrupted or waits for the occurrence of a Particular event, such as the completion of an I/O request.

suppose the Process makes an I/O request to a shared device, such as an disk since there are many Processes in the system The list of Process waiting for a Particular I/O device is called device Queue.

context switch :- is switching the CPU one process to other

Dispatchers :-

The dispatcher is module that gives the control of CPU to the Process selected by the short-term scheduler. The function involves the following:

1) context switching,
2) switching to user mode from supervisor mode.
3) jumping to the Proper location in the user Program to restart that Program

2.3 operations on Processes

O.S must Provide for a "Process creation and Process termination & they may be created or deleted dynamically.

Process creation

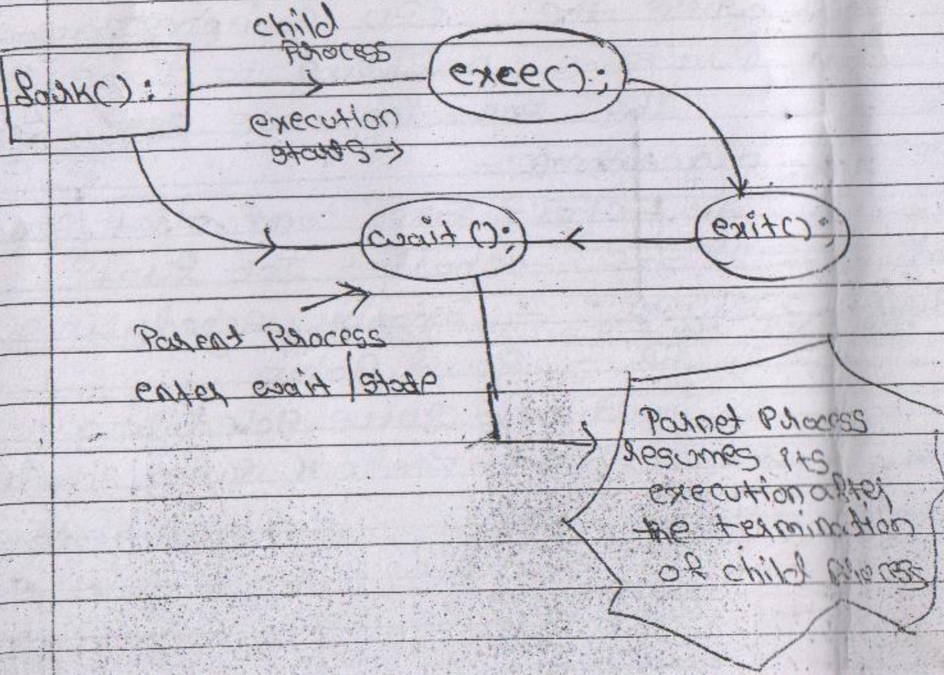
when the Processes are being created ~~by~~, O.S assigns the unique integer number called Pid (Proces identifier)

A Process can create many new Processes with an OS System call invocation know as "create- Proces" fork() → is system call used to create a child Process.

The newly created Processes are termed as children or child Processes of the Parent.

There can be two types of Process execution as and when a Process creates a new Process:

1) A Parent Process continue to execute along with its child Processes simultaneously.
2) A Parent Process has to wait until a few or all of its child Process complete their executions.


fork();
child Process
exec();
execution starts →
wait();
exit();
Parent Process enter wait state
Parnet Process resumes its execution after the termination of child Process

=> Parent Process has to wait until its child Process terminates its execution by invoking exit(); System call

=> Parent Process continue to execute furth by with drawing call to wait(); if the child execute exit();

Process Terminated:-

when the executing Process executes last statement, the Process Terminat

This is informed to O.S to remove the Process via an exit(); System call

The child Process need not continue to exist if its Parent Process terminated well before.

(2.5) (~~Process~~ Scheduling & ~~Basic concepts~~

when the Process terminates All the resources of the Process - including Physical memory, virtual memory, open files, I/O Bu are deallocated by O.S.

Imp

Scheduling criteria :-

The following are criteria for Process scheduling :

1) CPU Utilization :- we want to keep CPU as busy as possible. CPU Utilization can range from 0 to 100 %. In real system CPU utilization should range from 40 %.

2) Throughput :- The number of processes that are completed per time unit, called throughput.

3) Turnaround Time :- The interval from the time of submission of a process to the time of completion is the turnaround time.
Turnaround time is the sum of the periods spent waiting to get into memory, ready queue, executing on CPU & doing I/O.

4) Waiting time :- Waiting time is the sum of periods spend waiting in the ready queue.

5) Response time :-
Defines the time it takes to commence responding to a request or comm
Response time defines the length of time between entering a command & beginning to recieve a response from the system.

~~See~~ Scheduling algorithms :-

CPU Scheduling or Process Schedul
where the CPU switches among various processes according to some policies.
The following are ~~set~~ scheduling algorithms :-
1) FCFS - First come, First serve
2) SJF - shortest Job first
3) PR - Priority scheduling
4) RR - Round Robin
5) multilevel Queue scheduling
6) Multilevel Feedback Queue scheduling.

come First serve Scheduling.

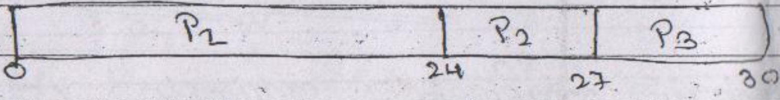
The Process which comes first or which is the first at the head of ready queue will be allocated the CPU.

The code of FCFS is easy to understand & write.

consider the following set of Processes

Process	Burst time
P_1	24
P_2	3
P_3	3

Gantt chart :-



The	waiting	time	for	P_1	is	zero milliseconds
"	"	"	"	P_2	"	24 "
"	"	"	"	P_3	"	27 "

average wait time = $(0+24+27)/3 = 17$ millisec.

Turn around time = $(24+27+30)/3$.

= 27 milliseconds.

FCFS scheduling algorithm is non-preemptive, once the CPU is assigned to process, that process keeps the CPU until it releases the CPU, either by terminating or by requesting I/O.

Shortest - Job - First scheduling :- (SJF)

This algorithm compares each process the length of the process's next CPU Burst. when the CPU is available, it is assigned to the process that has the smallest next CPU Burst. If the CPU Bursts of two Process is same the FCFS algorithm is used to break the tie.

**
Imp

Scheduling criteria :-

The following are criteria for Process scheduling :

1) CPU utilization :- we want to keep CPU as busy as possible. CPU utilization can range from 0 to 100%. In real system CPU utilization should range from 40%.

2) Throughput :- The number of processes that are completed per time unit, called throughput.

3) Turnaround Time :- The interval from the time of submission of a process to the time of completion is the turnaround time.
Turnaround time is the sum of the periods spent waiting to get into memory, ready queue executing on CPU & doing I/O.

4) Waiting time :- Waiting time is the sum of periods spent waiting in the ready queue.

5) Response time :-
Defines the time it takes to commence responding to a request or command. Response time defines the length of time between entering a command & beginning to recieve a response from the system.

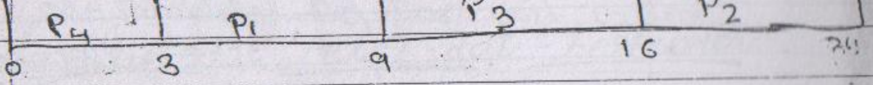
~~Sc~~ Scheduling algorithms :-

CPU Scheduling or Process Scheduling where the CPU switches among various processes according to some polices.
The following are ~~are~~ scheduling algorithms :-
1) FCFS - First come, First serve
2) SJF - Shortest Job first
3) PR - Priority scheduling
4) RR - Round Robin
5) multilevel Queue scheduling
6) Multilevel Feedback Queue scheduling.

consider the following set of Process.

Process	Burst Time
P1	6
P2	8
P3	7
P4	3

Gantt Chart :-



average time = $(3+16+9+0)/4$
= 7 milliseconds

Turnaroundtime = $(9+24+16+3)/4$
= 13 milliseconds

Advantages :-
1) Suitable for long-term scheduling
2) can be preemptive or non-preemptive
3) waiting time is less compared to FCFS
4) average ~~time~~ wait time is minimum.

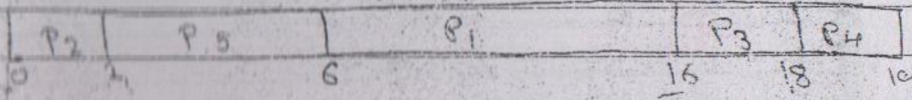
Disadvantage:-
1) Implementing SJF scheduling is difficult
2) It suffers from ~~[illegible]~~ Starvation Problem.

Priority scheduling :-
A Priority is associated with each Process, and CPU is allocated to the Process with the highest Priority.
Equal-Priority Processes are scheduled in FCFS order.

consider the following set of Processes

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

Gantt chart



average wait time = $(6+0+16+18+1)/5$
= 8.2 millisec.

turn around time = $(16+1+18+19+6)/5$
= 12 millisec.

Disadvantage :-
1) Priority scheduling algorithms suffers indefinite block or starvation.
2) Difficult to implement.
3) Determining & assigning the Priority is tedious job.

Round-Robin Scheduling :-

RR algorithm is designed especially for time sharing systems. A small unit of time called time quantum or time slice is defined.

The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum.

Time Quantum is the amount of time share given to each process interrupting the job if it is not completed by then

Disadvantages of RR :-
-> Average waiting time increases with increased size of time quantum.
-> small time quantums introduce scheduling overheads in terms of excessive context-switch times.

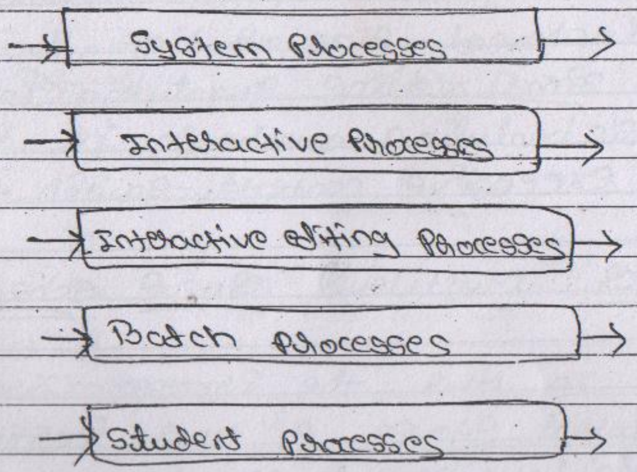
MLQ - Multilevel Queue scheduling.

In MLQ the processes are classified into different groups. A MLQ Partitions the ready queue into several seperate queues.
MLQ algorithms uses FCFS and Priority algorithm.

Ex:- The MLQ scheduling algorithm with five queues, listed below in order of priority
1) System processes
2) Interactive processes
3) Interactive editing processes
4) Batch processes
5) student processes.

Time duration is the unit of time given to each process interval the job if it is not completed

highest Priority



lowest Priority.

~~Ex~~

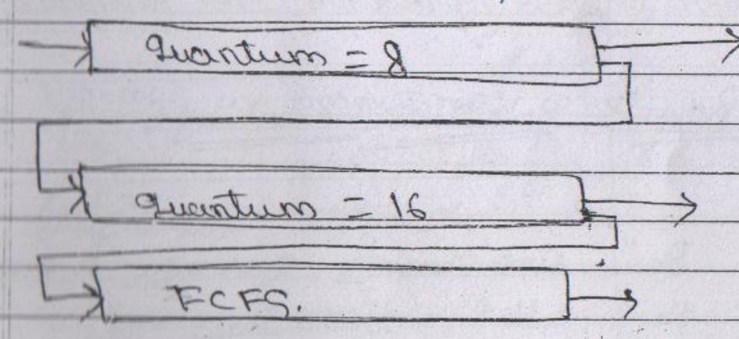
No Processe in the batch queue could run unless the queue for system processes. Interactive Processes & Interactive editing processes were all empty.

If Interactive editing processes entered ready queue while batch processes running, the Batch process would be preempted.

Multilevel Feedback-Queue Scheduling:-

In multilevel Feedback Queue Scheduling algorithm, allows a process to move between queues. The queues are seperated according to their CPU Burst If a process uses too much of CPU time it will be moved to lower priority queue.

In addition, a process that waits too long in lower priority queue is moved to a higher priority queue. This is called aging an solution to prevent starvation.



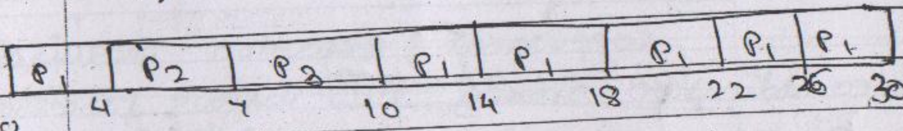
Ex:- consider a multilevel feedback-queue with 3 queues numbered from 0 to 2. The Scheduler first executes all processes in queue 0. only when queue 0 is empty it will execute processes in queue 1.

Similarly Processes is Queue 2 will be execute
if queue 'o' and ~~[illegible]~~ Queue '1' are empt
A Process that arrives for queue '1'
will Preempt a Process in Queue 2. A
Process in Queue 1 will in turn be
Preempted by a process arriving for queu
'O'.

RR algorithm uses FCFS algorithm
consider the following set of Processes.

Process	Burst time
P_1	24
P_2	3
P_3	3.

Time quantum = 4



average wait time $= ((0+(10-4))+4+7)/3$

$= (6+4+7)/3 = 17/3 =$

$= 5.66$ milliseconds

turn around time $= (30+7+10)/3$

$= 15.6$ milliseconds.

advantages

1) offers good response time & turn around time.
2) Process wait time is not longer

chapter - 3

Synchronization

5 marks

cooperating Process is one that can affect or be affected by other Processes executing in the System.

In general any Process sharing data with other Processes is a co-operating Processes

concurrent execution requires co-ordination among the various Processes, needs a mechanism that Permit Processes to communicated each other via IPC. (Inter Process communication) & synchronise their actions.

Imp

* concurrent access to shared data can introduce the Problems of data inconsistency.

To overcome the Problems of data inconsistency and ensure orderly execution of Co-operating Process we are going to study critical section Problem.

classical Problems of synchronisation.

Following are the 3 synchronisation Problem:

1) Bounded - Buffer Problem
2) Readers - writers Problem
3) Dining - Philosophers Problem.

Bounded - Buffer / Producer -consumer Problem:

1) Producer Process generates useful data or information. Producer - Process has to Produce full buffers of Information for its consumer.

2) The consumer Process makes use of generated data. The consumer Process has to consume the necessary information (full buffer data) such that it should Produce the empty buffers for its Producer

3) concurrent execution of both Producer Process and consumer Process takes Place by sharing a common buffer, that can be filled by Producer & ~~consumed~~ emptied by consumer Process

9449636633

4) The Bounded-Buffer/Producer-consumer Problem make use of a fixed size buffer. Therefore, Producer process must wait if the buffer is full. Similarly, consumer process must wait if the buffer is empty.

Both Producer & consumer Process must be Synchronised:
Synchronisation is achieved by employing mechanism as "Semaphore".

v.v.v.v Imp.
Critical Section Problem:-

Consider a System consisting of 'n' Processes waiting to be assigned to CPU, {P_0, P_2, ... P_{n-1}}.
All the processes share data and are co-operating processes. Each Processes has the following section
1) Entry Section
2) Critical Section
3) Exit Section
4) Remainder Section

```
do{
        entry section

            critical section

            exit section

            Remainder section

    }while (TRUE);
```

Each Process must request Permission to enter its critical section. This Section of code is the entry section.

Critical section Performs the critical activities accessing &, modifying, shared global variables, ~~[illegible]~~ updating, file, or modifying files etc.

The critical section is followed by exit Section.
The remaining code is Remainder Section

Imp

The critical-section Problem suggest that the critical section code of Process must be in used and executing by one & only running process.

when one Process is executing ~~remaining~~ ~~Processes must not be~~ its critical section, other remaing Processes must not be allowed to execute their critical section.

Imp

The Solution to Critical-Section Problem must Satisfy the following 3 Requirements.
1) mutual exclusion
2) Progress
3) Bounded waiting.

Mutual exclusion :-

If Process "Pi" is executing in its critical section, then no other Processes can execute their critical section

Progress :-

If no Process is executing its critical section and some Process wish to execute the critical section then only those Processes that are not executing in their remainder sections can participate in the decision on which will enter its critical section.

Bounded Waiting :-

There ~~exists~~ a is a bound or limit on the number of times the Processes are allowed to enter their critical section after process has made a request to enter its critical section.

Synchronisation Hardware :-

There are special Instructions with which mutual exclusion can be implemented as a solution to critical-section problem.

* If the system support test-and-set instruction then it permits the verification & modifying ~~the~~ Process critical section.

* A Boolean variable "Lock" is defined as synchronising variable.

~~Chapter~~.

Synch is mech of ensuring that only One process at time access a shard resource

"Lock" is initialized to false (0) value.

The following code illustrate the structure of process Pi for implementing mutual exclusion via test-and-set instruction

```
repeat
  {
    while (test and set (lock))
    do {nothing ;}

        | critical section |

    lock; =false;

        | remainder section |

  } until (false);
```

Disadvantages :-

1) Processes have to wait to get an entry to their critical sections, thus consuming processor time.

2) Processes may come across starvation

Semaphores :-

A general solution to synchronisation & critical section is a synchronisation tool called as semaphores.

* A semaphore 'S' is a synchronising variable that can hold an integer value
* A semaphore variable 'S' is accessed via two functions ; "wait" and "signal" ' Each function accepts a single argument 'S'

wait (S) :→ The function is defined with following program segment

```
wait (S)
{
    while (S < =0)
        ;   // no operation
    S--;
}
```

Signal(S) :- The function is defined with following 'C' code

```
signal (s)
  {
    s++;
  }
```

Semaphores does not permit the simultaneous modification of semaphore value by more than one process at a time.

The solution to n-process critical section can be provided by implementing mutual exclusion with semaphore variables. All processes share common variable called 'mutex'

```
Repeat {
    wait (mutex);
    critical section
    signal (mutex);
    remainder section
}until(false);
```

Semaphores Types.

1) counting semaphores :- There is no restriction on the integer value of semaphore variable

2) Binary semaphores :- can store the binary values either '0' or '1'

3) spinlock semaphore :- make the process to spin/loop around while waiting for the lock.

1JSRDV5E10137

chapter - 7. 19

FILE SYSTEM

File provide a way to store information on the secondary storage the disk structures and read it back in the same way at later stages. File can be defined as container for a collection of information and it can be for future use, for longer periods of time.

File attributes :-

The following are the attributes of files:-

1) Name :- The symbolic name, kept in human readable form.

2) Identifier :- This is unique tage, usually a number, identifies the file within the filesystem, it is non-human-readable name for file

3) - Type :- This information is needed for systems that support different types of files.

4) Location :- This information is ~~needed for~~ a pointer to a device & to the location of the file on that device.

5) Size :- The current size of the file in bytes, words or blocks, & possible maximum allowed size are included in this attribute.

6) Protection :- Access-control information determines who can do reading, writing, executing.

7) Time, data, & user Identification :- This information may be kept for creation, last modification & last use. These data can be useful for protection, security, & usage monitoring.

File operations :-

The following are the operations performed on file

1) Creating file → Two steps are necessary to create a file. First, space in the file system must be found for the file. Second, an entry for the new file must be made in the directory.

2) Writing a file :- The write system call writes data to the given file which is opened in write mode. The writing takes

place again at current write pointer.
The system maintains a write pointer to the location in the file as to where exactly the next write to take place.

3) Reading file :-
The read system call reads data from the given open file at current file position pointer. The caller must specify the file pointer, name of the file & the how much data to be read from file.

4) Repositioning within file (seek to file):-
The seek system call performing file seek operation & there by repositions the current file pointer to a specific value in the file

5) Deleting a file :-
The delete system call erases the specified file & releases the file space to disk.

6) Truncating file :-
whenever the structure of file is to be preserved & only its contents are to be erased the truncate system call is used.

File types :-

The following are the types of the files :

1) Directories :- are the system files that maintain the structure of the file system. It maintains groups of files or other nested- sub directories.

2) Regular Files :- are either ASCII files or binary files. ASCII files contains lines of text data.
Binary files exists in two forms i.e, with .com and .exe extensions. Both forms are executable files.

A Technique for implementing file types is to include the type as part of the file name. The name is split into two parts - name and an extension, usually separated by period character.

File type	usual extension
1) executable	exe, com, bin
2) object	obj, o
3) source code	c, cc, java, asm
4) Batch	bat, sh
5) text	txt, doc
6) word processor	doc, wp, rtf, tex
7) library	lib, dll
8) print	ps, pdf
9) archive	arc, zip, tar, rar
10) multimedia	mpeg, mp3, avi

File structures:-

File types are used to suggest the required internal structure of the file. Also files must be designed, created as per the file structure that can be identifie and accepted by underlying operating system.

The common file structures are

1) Byte streams:-

File is an unstructured collection o bytes. O.S looks at file as a sequence o 8-bit bytes.

byte 0	byte 1	byte 2	- - - -	byte n

2) Record streams:-

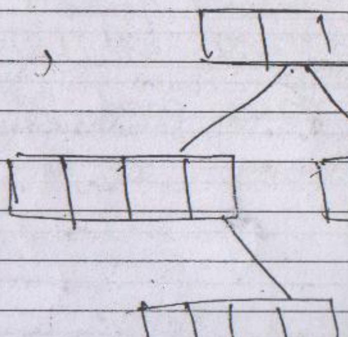
A high level of file system provides record-streams. that is file is organized into a sequence of fixed length records. Accordingly a read operation returns one records, & write operations writes or appends one record.

Record 1	Record 2	- - - -	Record n

3) Tree structured :-

Here file is organized into a tree of records, records can be both varying length & same length. The tree structure is sorted on the specified key field, thus searching a record is faster.

~~Internal file stru~~



Internal file structure :-

Internall all disk I/O is performed interms of disk blocks. ie. one block as Physical record at a time and all blocks are of same size.

Logical record & Physical record may vary in length

Logical Records are Packed into a Physical block before writing it to the disk. There can be 512 bytes per block.

After reading a disk block into main memory, unpacking procedure is employed for converting secondary disk blocks into a byte stream logical records.

A file may be considered as a sequence of blocks. All the basic I/O functions operates in terms of blocks.

Disk space is always allocated in blocks, Some Portion of the last block of each file is generally wasted. If each block were 512 bytes, For eg:- If the file is of 490 bytes then 22 bytes of a block are wasted. The waste incurred to keep everything in units of blocks is internal fragmentation.

Block 1 2

Access Methods:-

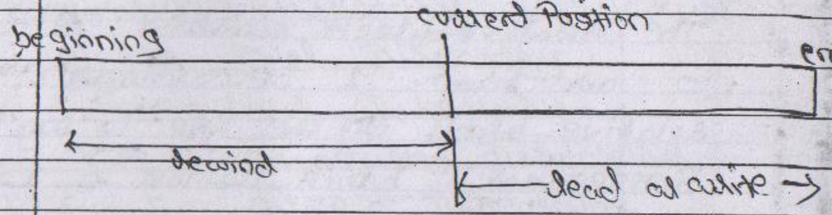
Files store information. This information must be accessed and read into computer memory. The following are file access methods
1) Sequential Access
2) Direct or Relative Access
3) Indexed Sequential Access

Sequential Access:-

In Sequential Access, the information in the file is processed in order, one record after ~~the~~ other.

Sequential Access to the file records is defined by current file position and the current position pointer indexes the records sequence by successive addition to locate the next record in sequence. The system application program must be able to jump or advance forward or backward "n" records.

The sequential access is illustrated in figure below:



Direct Access:-

In direct access method allows the prog to read & write records rapidly in no particular order. The direct access method is based on Disk model of a file, sinc disks allow random access to any file bloc In direct access method, files allow arbitrary blocks to be read or written. for eg: application may read 8th record & then writes 15th record.

For the direct access, the file operatio must ~~been~~ contain a new parameter calle block number. This block number serve a an index relative to the beginning of file

The block number provided by the user to the O.S is normally a relative block number. The relative block numbers begin from either '0' or from '1'. Accordingly, the first relative block of the file is say '0' & the subsequent block number is '1' & it continues.

Indexed Sequential Access :-

This method involves the construction of an index for the file. ~~The~~ Now to locate particular entry, in the given file, first the index table is searched for a given search key value. If it is found then the corresponding pointer to the record is used to access the file directly and desired record is read or written.

When an index file is created, an index record or index entry for each block in the ordered file, contains search key field & it points to one or more records.

Directory Structure And Disk Structu

The directory is structure in whi the files are organised. Directory is set of logically files & other sub directories of files.

Storage Structure :-

File system must contain inform about the files in the system. This informat is kept in entries in an device directory or volume table of contents. The device directory records information - such as name location, size, and type for all files on the volume.

Imp M.Q.P 6m.

The following are the operations performed on the directory.

1) Search for file :- The directories can be searched for specific file entries we should be able to find all files whose name match a particular pattern.

2) Create file :- new files need to be created & added to directory.

3) List a directory:- we should be able to list the files in a directory & the contents of directory entry for ~~each~~ each file in the list.

4) Rename a file:-
we must be able to ~~change~~ rename the file. Renaming the file allows its position within the directory structure to be changed

5) Delete a file:- when the file is no longer needed, we should be able to remove it from directory.

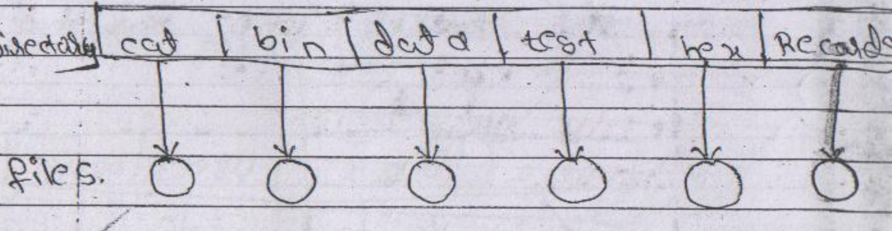
6) Traverse the file system:-
we should be able to access every directory & every file within a directory structure.

v.v.v.v.v. Imp

Directory structures:-

The following are ways of representing directory structure.
1) single-level Directory
2) Two-level Directory
3) Tree-structured Directory
4) Acylic graph Directory.

Single-level Directory:-
It is simple directory structu
all the files are contained in the same directory.



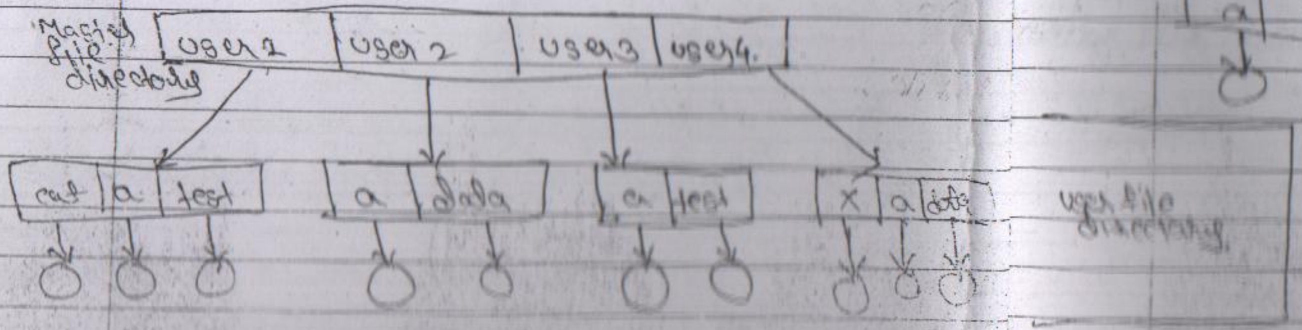
Limitations of single level Directory:-

1) Not suitable for a large number of files & more than one user
2) Because of single ~~file~~ directory, fil require unique name.

3) It is difficult to remember the names of all the files as the number of files increases.

Two level directory:

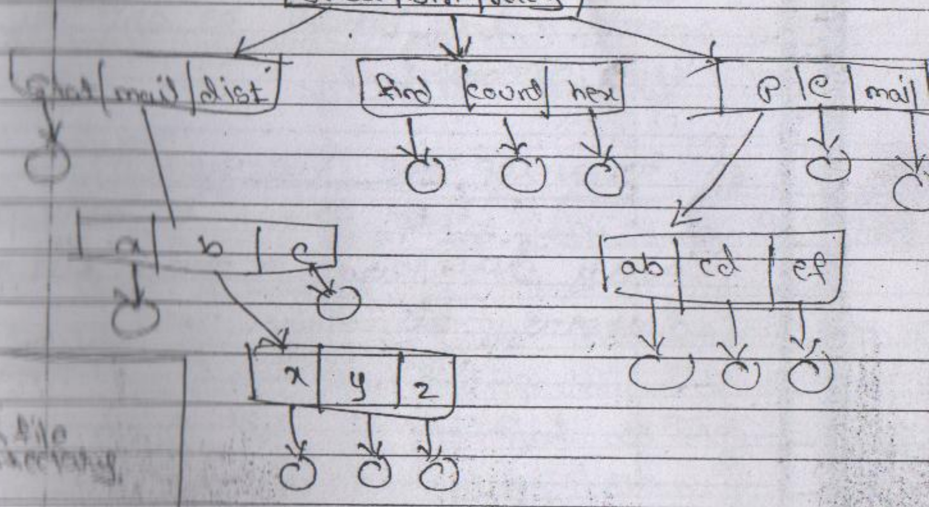
In the two-level directory structure each user has his own user file directory (UFD). The UFD's have similar structure but each lists only the file of a single user. when a user job starts or user logs in, the system's master file directory (MFD) is searched. The MFD is indexed by user name or ~~according to~~ account number each entry points to UFD for that user



v.v.v. Imp

Tree structured Directories

Tree structured directories are th extension of two-level directories. This generalization allows user to create their own subdirectories and to organiz their files accordingly. A tree is the most common directory structure. The tree h a root directory, and every file in the system has a unique Path name.



All the directories have the same internal format. one bit in each directory entry defines the entry as a file (0) or as a subdirectly (1). special system calls are used to create and delete directories.

Each user has a current directory. current directory should contain most of the files that are of current interest to the user. when a reference is made to a file, the current directory is searched. Pathname is used to search any operation on the file with another directory.

Path names can be of two types
1) Absolute Path name.
2) Relative Path name.

Absolute Path name begins at the root and follows a Path down to specified file giving the directory names on the Path

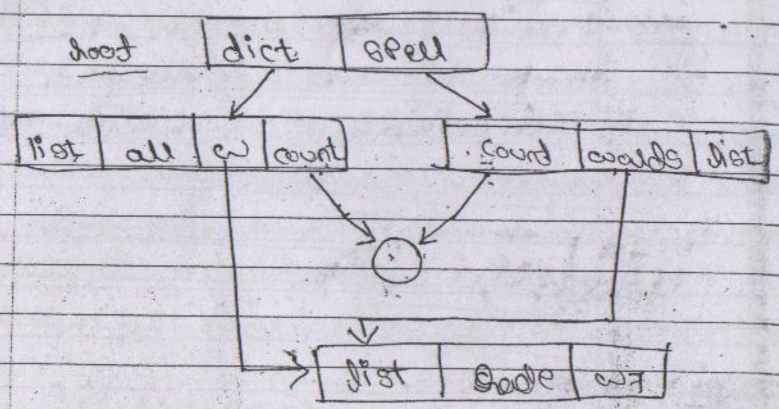
Relative Path name defines a Path from the current directory.

Acyclic Graph Directories:-

Acyclic graph with no cycles allows directories to share subdirectories and files. same file or directory ma in two different directory.

Shared files and subdirectories can implemented by using links.
A link is effectively a Pointer to another file or subdirectary.

link is implemented as an absol or relative Path name.



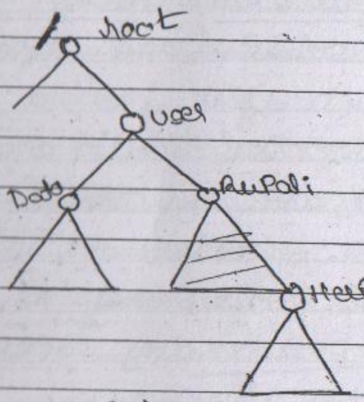
File system ~~mounting~~ mounting

File must be opened before it is used, a file system must be mounted before it can be available to processes on the system.

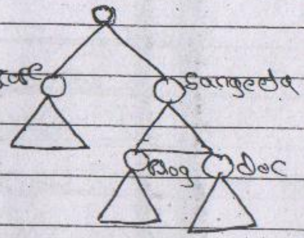
The mount procedure is straightforward. The o.s is given the name of the device & the mount Point — the location within the file structure where the file system is to be attached.

-> Mount Point is empty directory at which the mounted file system will be attached
-> Name of the device and location within the file structure at which to attach the file system is required.
-> O.S ~~[illegible]~~ verifies that device contains a valid file system.
-> Device ~~driver~~ driver is used by O.S for these verifications.
=> finally o.s mounts the file system at a specified mount Point.

To illustrate file mounting, consider the file system where the triangle represents subtrees of directories.



(a) Existing



(b) unmounted Partition.

In the figure (a), existing file system is shown & figure (b) an unmounted Partition residing on /device/dsk is shown. At this Point only files on the files on the existing file system can be accessed.

File sharing :—

Multiple users :-

1) Given a directory structure that allows files to be shared by users, the O.S must mediate the file sharing. The system either can allow a user to access the files of other users by default or it may require that a user specifically grant access to the files

2) To implement sharing & protection, the system must maintain more file and directory attributes than on a single user system. Most of the system uses the concept of file directory owner & group. The owner is the user who may change the attributes, grant access. The group attribute of the file is used to define a subset of users who may share access to the file.

3) Most of the system implement owner attributes by managing a list of user names & associated user identifiers. The owner & group IDs of a given file or directory are stored with the other file attributes. when the user requests an operation on a file, the user ID can be compared to the owner attribute to determine if the requesting user is the owner of the file. Same way group IDs can be compared. The result indicates which permissions are applicable.

Imp

Remote file systems :-

Communication among remote computers is possible. Networking allows the sharing of resources spread across a campus or even around the world.

The first implemented method involves manually transferring files between the machines via programs like ftp (file transfer protocol). The second method uses a distributed file system (DFS) in which remote directories are visible from the local machine. The third method is WWW (world wide web). A browser is needed to gain access to the remote files.

i) Client-server ~~method~~ model:-

machine containing the files is the server and the machine seeking access to the files is a client. Server declares that a resource is available to clients & specifies exactly which resource and exactly to which clients. Files are usually specified on a partition or subdirectory level. A server can serve multiple clients. & a client can use multiple servers.

ii) Distributed information systems:

To make client-server systems easier to manage distributed information systems also known as distributed naming services, provide unified access to the information needed for remote computing. Distributed information system provide username / password / user ID / group ID space for a distributed facility.

Imp.

Protection :-

Information must be protected from a physical damage and improper access i.e reliability & protection.

Types of Access :-

Protection mechanisms provide controlled access by limiting the type of file access that can be made. Access is permitted or denied depending on several factors, one of which is type of access required.

Different types of operations may be controlled in access type These are.

1) Read - Reads from the file
2) write - write the file
3) Execute - Load the file into memory & execute it.
4) Append - write new information at the end of the file.
5) Delete - Delete the file & free its ~~[illegible]~~ space for possible reuse
6) List - List the name & attributes of the file. eg ls (Linux command)

Access control.

Various users may need different types of access to a file or directory. when a user requests a particular file, operating system checks the access list associated with that file. If that user is listed for the requested access, the access is allowed. otherwise a Protection violation occurs, & the user job is denied access to the file.

Many system recognize three classifications of users in connection with each files for access control.

1) owner.
2) Group.
3) universe.

owner — user who created file.

Group — set of users who are sharing the file & need similar access is a group

universe — All other users in the system constitute the universe.

(Linux) ahmad → Linux OS

Chapter - 8

: File System Implementation :

Disks are secondary storage device which are used to store large amount da
Disks have two characteristics:

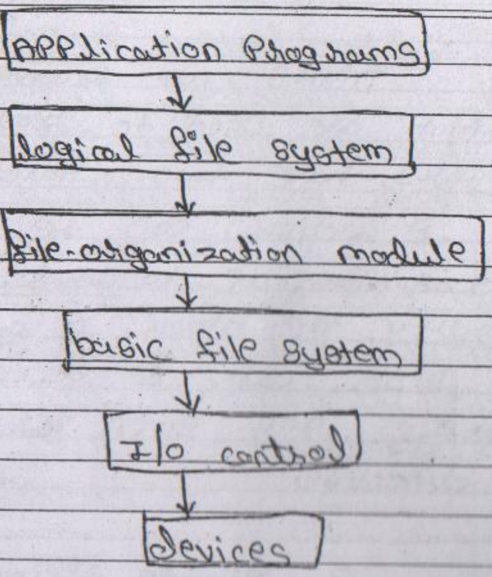
1) A disk can be rewritten in Place, it is Possible to read a block from disk, modify the block, & write block of data.
2) Disk can be accessed directly. i.e., any given block can be accessed directly

To Provide efficient and convenient access to the disk, the operating system imposes files systems to allow the data to be stored, located, & retrieved easily.

Imp. V.V

File system is composed of many different levels. Each level in design uses the features of lower levels to create new features for use by higher levels.

Layered file system:-



I/O control:- It is lowest level, consists of device drivers and interrupt handlers. to transfer information between the main memory and disk system. A device driver can be thought as a translator.

Basic file system:- It needs generic commands to appropriate device driver to read and write physical blocks on the disk. Each physical block is identified by its numeric disk address

Eg - drive 1, cylinder 73, track 2, sector 10.

File organization module:- It knows about files and their logical blocks, as well as physical blocks. file organization module can translate logical block addresses to physical block addresses for the basic file system to transfer. File organization module also includes the free-space manager.

Logical file system:-

It manages metadata information. Metadata includes all of the file-system structure except the actual data. It maintains file structure via file-control blocks. A file-control block (FCB) contains information about the file, including ownership, permissions, and location of the file contents. The logical file system is also responsible for protection & security.

M.O.S [9m]

File system Implementation :-

whenever the file system implementation is on-disk, the secondary storage device that provides bulk storage may contain one or more ~~information~~ of the following structures such as

→ the Boot Block or boot record
→ total no.r of blocks on the disk
total " " free blocks and their address.
→ File allocation table (FAT)
→ Directory structure ~~[illegible]~~ and so on

① Boot control Block :-

It may contain bootstrap loading information as how to locate and load resident O.S from the disk volume to Physical main memory of the given computer.

If the disk does not contain an O.S this block will be empty. It is first block of Disk which contains O.S

This Process is reffered is Bootstrapping.

② Volume control Block :-

It contains volume details, su as the number of blocks in the Partiti size of the blocks, free block count a free block pointer, and free FCB & FCB Pointers.

V.V. Imp

Creation of new file :-

The following are the steps

① To create new file application Progr calls the logical file system.

② Now to create new file logical file system allocates a new FCB. otherwise FC is allocated from the set of free FCB's that have been already used.

③ Further the Logical file system updates the new file name & ~~[illegible]~~ FCB.

④ Lastly the logical file system re-write the updated or modified directory from ~~Physical~~ Primary memory back to disk.

The typical FCB is as shown in th figure.

(P.T.O)

File control Block

file Permissions
file dates (create, access, write)
file owner, group, universe
file size
file data blocks or Pointers to the file data blocks.

~~Properties~~

Implementing "Open a file"

① Application Program ~~contends~~ contains (Eg. fopen()) the "open()" System call which Passes the file name to logical file system.

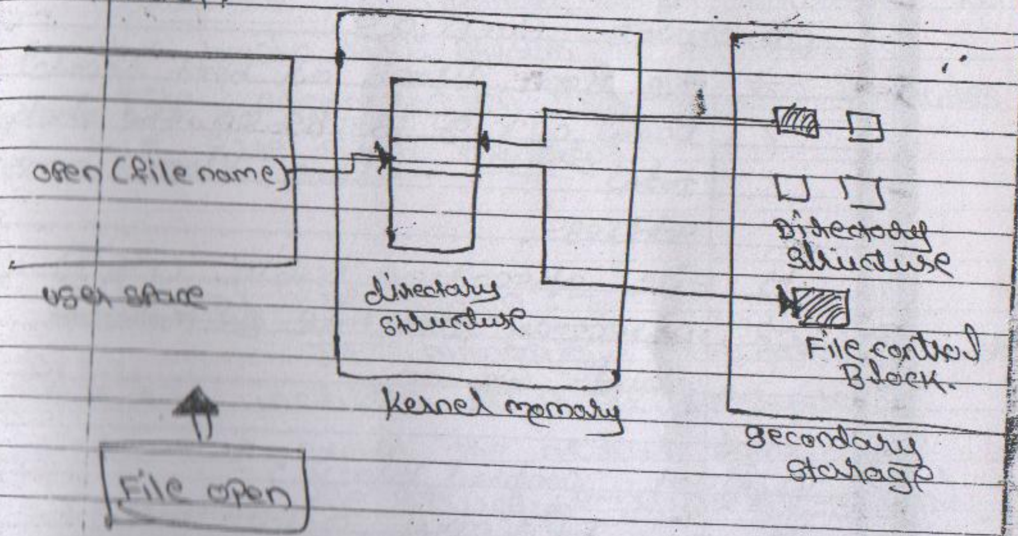
② First open() call searches the system to verify whether specified file is already in use by some other process.

If it is so per-process open file table entry will be created that points to the existing system wide open-file table.

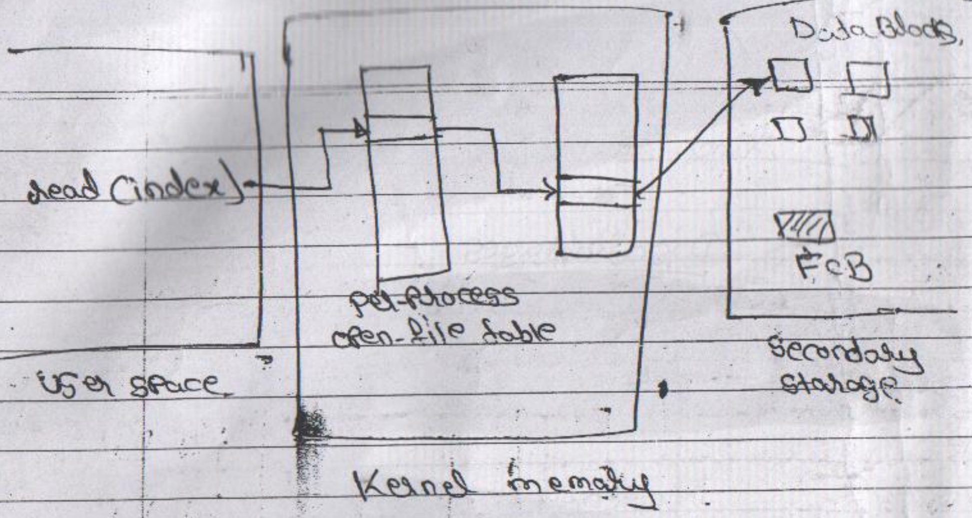
If the file specified in open() system call is not yet opened, then the directory structure will be searched to locate the specified file name

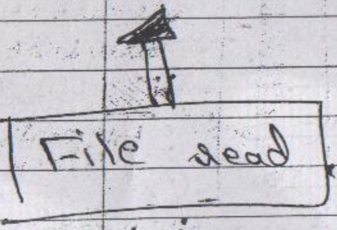
③ Once the file is found, its FCB will be loaded into a system-wide open-file table in Physical memory

④ The open() system call returns a Pointer.



(P.T.O)





Virtual File System:-

The O.S have to support & accomodate mutiple file-systeme types concurrently. To do so file management offers architecture consists of three distinct layers.

It is possible to accomodate different file-system types within the same directory structure, including

"Network File System (NFS)".

The three distinct layers of File-system Implementation can be listed out as

1) ~~The~~ File-system Interface Layer:

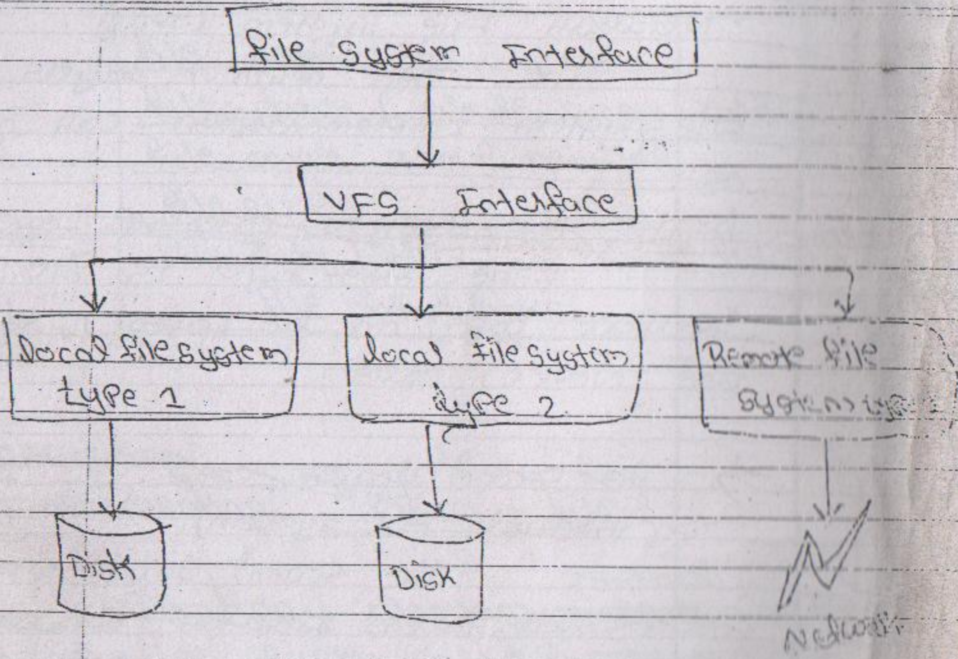
The first layer is file system interface. based on the open(), read(), write() and close() calls.

2) ~~The second layer con~~ VFS Interface. (virtual File system).

The second layer VFS layer defines file access ~~Interface between~~ Interfaces ~~[illegible]~~ and distinguistes the individual files ~~from~~ one local file-system ~~to~~ type to another on a local hard disk and remote file-system on a network.

3) The file-system storage layer:-

This layer implements storage-structure for different file-system including remote-file system types. It has to define the physical properties of storage device ~~for~~ the implementation of mapping to physical storage structures.



Directory Implementation :-

Directory is implemented in two ways
1) Linear List
2) Hash table.

Linear List :-

Linear list is a simplest method. Linear list consists of file names with pointers to data blocks.

Linear list uses a linear search to find ~~a~~ a particular entry.

Disadvantages :-

As it is linear searching of direct entries is time consuming. user would notice a slow implementation of access to

whenever a new file entry is to be ~~[illegible]~~ created, first linear searching has to be conducted to ensure that no other existing file contains the same file-name. Then new file name is to be appended as the last entry in that directory.

File deletion also requires linear searching of file entries across the directory for the name file to be deleted. The deallocate the file if found by releasing the disk space allocated to it.

PCB Problem grade 14K.

14K.