Oprating System

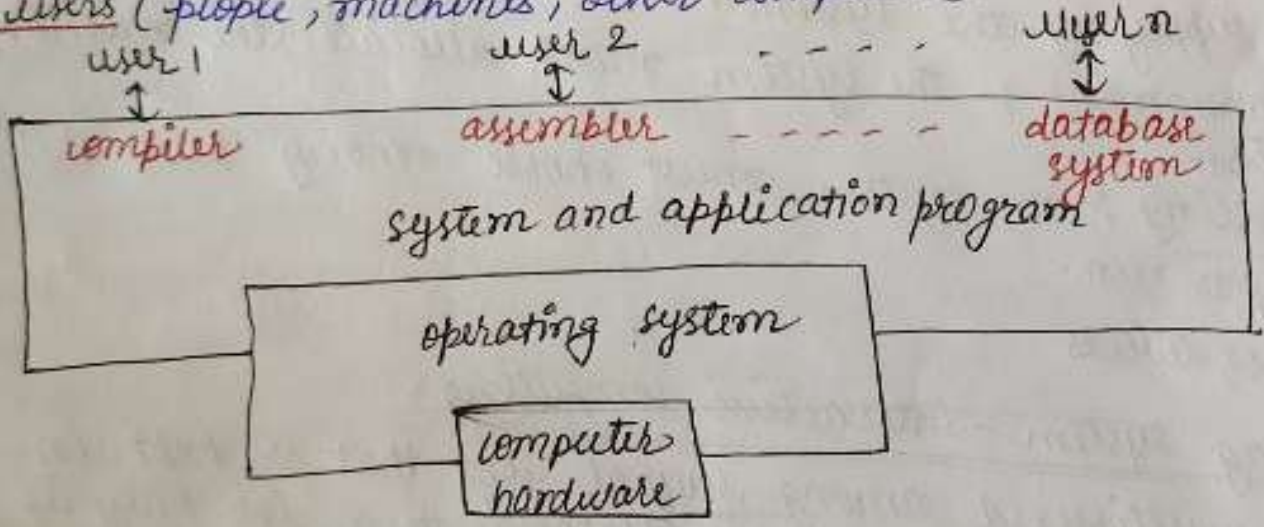
Introduction

→ A program that acts as an intermediate b/w a user of a compu and the computer Hardware. (user & computer hardware)

- Goals of OS
  - → Execute user programs and make solving user problem easier.
  - → Make computer System more convenient to use.
  - → use the computer hardware in efficient manner.

computer system components

1. Hardware :- provides basic computing resources (CPU, memory, I/O devices).
2. operating system :- controls and coordinates the use of the hardware among the various applications programs for the various users.
3. Application programs :- defines the ways in which the system resources are used to solve the computing problems of the users.
4. users (people, machines, other computers)



⭐ Operating system defination

i) Resource allocator :- manages and allocates resources.
ii) control program :- controls the execution of user program and operations of I/O devices
iii) Kernel :- the one program running at all times (all else being application programs)

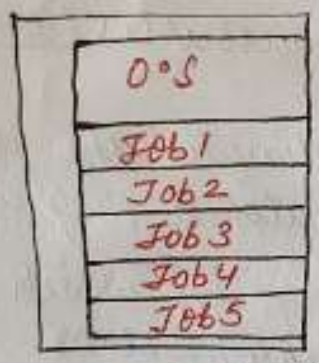
⭐ mainframe Systems

→ Reduce setup time by batching similar jobs.
→ Automatic job sequencing :- automatically tranfers control from one job to another. first rudimentary operating system.
→ Resident monitor
  - → intial control in monitor.
  - → control tranfer to job.
  - → when job completes control transfer back to monitor.

Memory layout for a simple Batch system.

⊛ Multiprogramed Batch System.

several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.



⊛ OS features needed for multiprogramming.

i) I/O routine supply by the system.
ii) Memory managment :- The system must allocate the memory to several jobs.
iii) CPU scheduling :- the system must choose among several jobs ready to run.
iv) Allocation of devices

⊛ Time sharing system - interactive computing.

→ The CPU is multiplexed among several jobs that are kept in memory and on disk (the CPU is allocated to a job only if the job is in memory)
→ A job swapped in and out of the memory to the disk
→ online - communication between user and computer system when the operating system finishes the execution of one command, it seeks for next "control statement" from the user's keyboard.
→ on-line system must be avilable for users to access data and code.

⭐ Desktop Systems

i) personal computer:- computer system dedicated to a single user.

ii) I/O devices :- keyboard, mice, display screen (monitor) etc.

iii) user convenience and responsiveness.

→ can adopt technology developed for larger operating system' often individuals have sole use of computer and do not need advanced CPU utilization of protection features.

→ may run several different types of operating systems (windows, UNIX, Linux).

⭐ Parallel System

→ multiprocessor system with more than one CPU in close communi-cation.

→ Tightly coupled system:- processors share memory and clock; communication usually takes places through the shared memory.

→ Advantages of parallel system.

→ Increased throughout

→ Economical

→ Increased reliability
- i) graceful degradation
- ii) fail-soft system.

⭐ Parallel System (cont.)

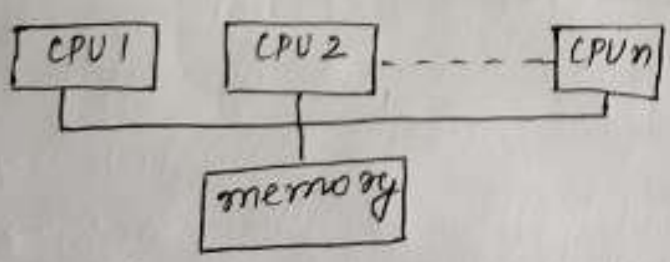
Symmetric multiprocessing (SMP)
- each processor runs and identical to each other, copy of the O.S.
- runs without performance deterioration.
- most morden O.S supports SMP.

Asymetric multiprocessing (AMP)

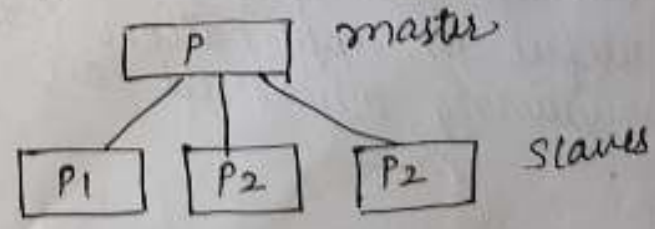
→ Each processor is assigned a specific task; master processor schedulles and allocated work to slave processor

→ more common in extremely large system.

Symmetric multiprocess-



Asymetric multiprocessing



Distributed system

→ Distributes the computations among several physical processors.
→ **loosely coupled system** :- each processor has its own local memory ; processors communicates with one another through various communication lines, such as high-speed buses or telephone lines.
→ Advantags of distributd system
  → Resource sharing
  → computation spud up - load sharing
  → Reliability
  → communications.

Distributed systems (cont)

→ Requires networking infrastructure
→ local area nework (LAN) or wide are Network (WAN)
→ may be either client - server or peer-to-peer system.

Real time system

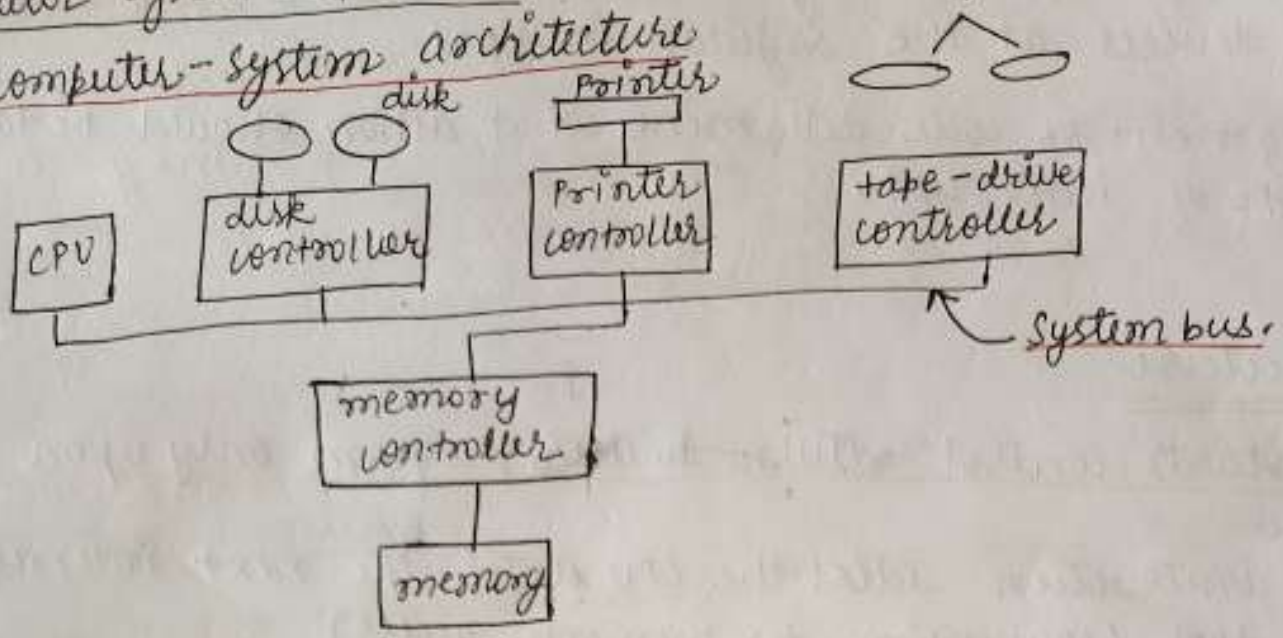
→ often used as control device in a dedicated application such as controlling scientific experiment, medical imaging system, industrial control system, and some display system.
→ well defined fixed-time constaints
→ Real-time system may be either hard or soft real-time.

Real time system (cont.)

→ Hard time / real :- secondary storage limited or absent, data stored in short term or ROM
  → conflict with time-sharing systems, not supported by general purpose O.S
→ Soft real-time
  → limited utility in industrial control of robotics.
  → useful in application (multimedia, virtual reality) requiring advanced O.S features.

computer-system structures

ⓐ computer-system architecture



★ computer-system operations.

ⓐ I/O devices and the CPU can execute concurrently.
ⓑ Each device controller is in charge of a particular device type.
ⓒ Each device controller has a local buffer.
ⓓ CPU moves data from/to main memory to/from local buffers.
ⓔ I/O is from the device to local buffer of controller.
ⓕ Device controller informs CPU that it has finished its operation by causing an interrupt.

→ functions of Interrupts

ⓐ Interrupt transfers control to the interrupt services routine generally, through the interrupt vector, which contains the addresses of all the service routines.
ⓑ Interrupt architecture must save the address of the interrupt instruction.
ⓒ Incoming interrupt are disabled while another interrupt is being processed to prevent a lost interrupt.
ⓓ A trap is a software-generated interrupt caused either by an error or a user request.
ⓔ Interrupt Handling
i) The O.S preserves the state of the CPU by storing registers and the program counter.
ii) → Determines which type of interrupt has occured.
i) Polling :- in Polling CPU waste lots of CPU cycles by repeatedly checking the command ready bit of every command.
ii) vectored interrupt system :- an alternative to a polled interrupt, which requires that the interept handler to send a signal to each device in turn in order to find out which one send the interrupt requests.

interrupt can generate at any instant of the time, CPU keeps polling the devices at the regular interval.

→ Separate segment of code determine what action should be taken for each type of interrupt.

✱ I/O structure

→ After I/O starts, control return to user program only upon I/O completion.

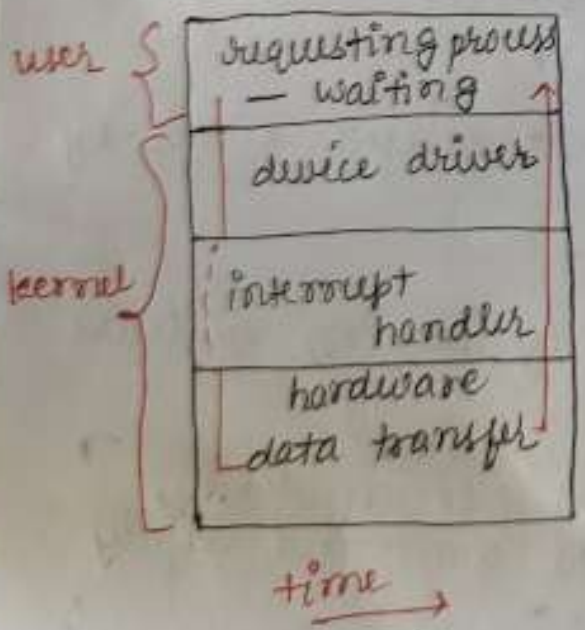
- → wait instruction idles the CPU until the next interrupt.
- → wait loop (contention for memory access)
- → At most one I/O request is outstanding at a time, no simultaneous I/O processing.

→ After I/O starts, control returns to user program without waiting for I/O completion.

- → System call :- request to the O.S to allow user to wait for I/O completion.
- → Device-status table contains entry for each I/O device indicating its type, address and state.
- → O.S indexes I/O devices table to determine device status and to modify table entry to include interrupt.

Two I/O methods

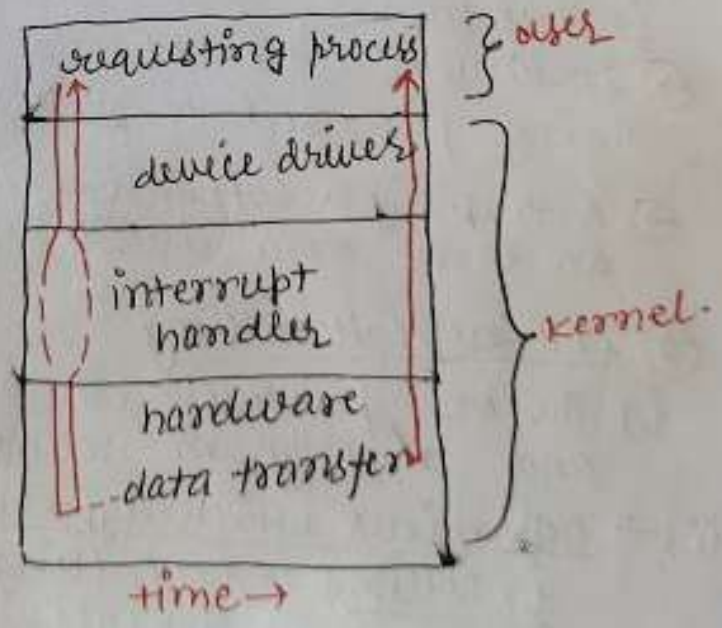
Synchronous



→ fast transmission

→

Asynchronous



slower transmition due to extra bit gap.

Direct memory access structure

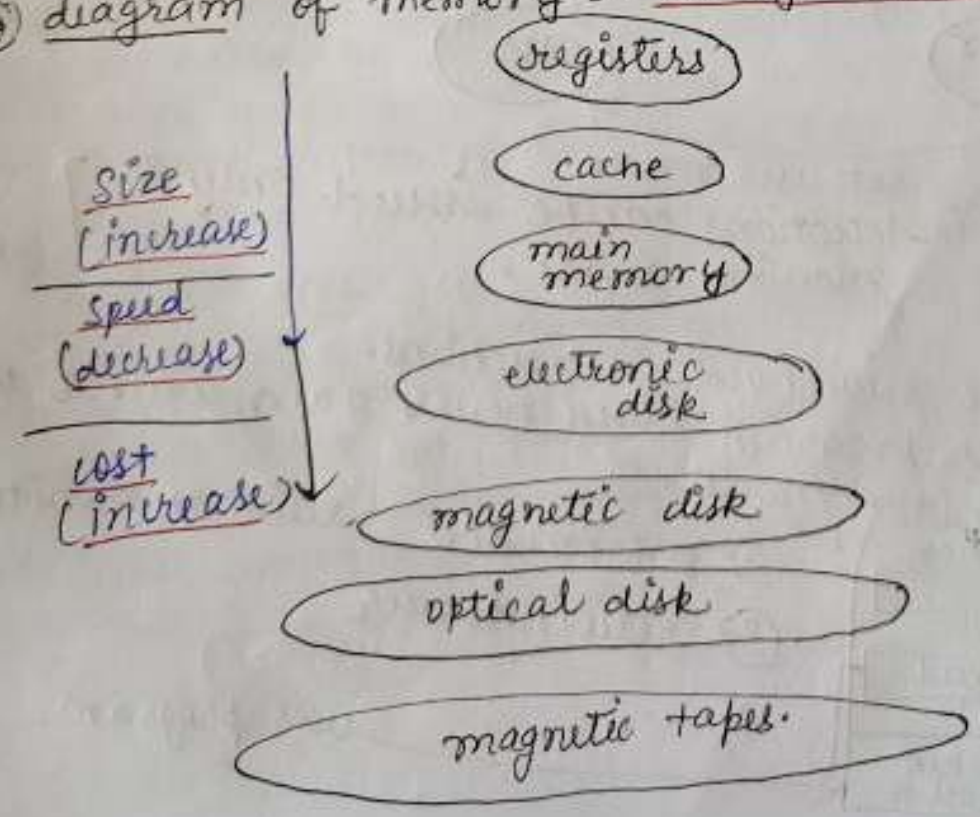
- used for high-speed I/O devices able to transmit info. at close to memory speeds.
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention.
- only on interrupt is generated per block, rather than the one interrupt per byte.

Storage structure

- **main memory** :- only large storage media that the CPU can access directly.
- **secondary memory** :- extension of main memory that provides large non volatile storage capacity.
- **magnetic disk** :- Rigid metal or glass platters covered with magnetic recording material
  - → Disk surface is logically divided into tracks, which are subdivided into sectors.
  - → The disk controller determines the logical interaction b/w the device and the computer.

Storage Hierarchy

- Storage system organized in hierachy.
  - → Speed
  - → cost
  - → volatility
- **Caching** :- coping info. into faster storage system main memory can be viewed as a last cache for secondary storage.
- diagram of memory or storage-device Hierarchy



Caching

→ use of high speed memory to hold recently-accessed data.
→ Requires a cache managment policy.
→ Caching introduces another level in storage hierarchy. This requires data that is simultaneously stored in more than one level to be consistent.

Hardware Protection

→ i) Dual-mode Operation
→ I/O protection
→ memory protection
→ CPU protection.

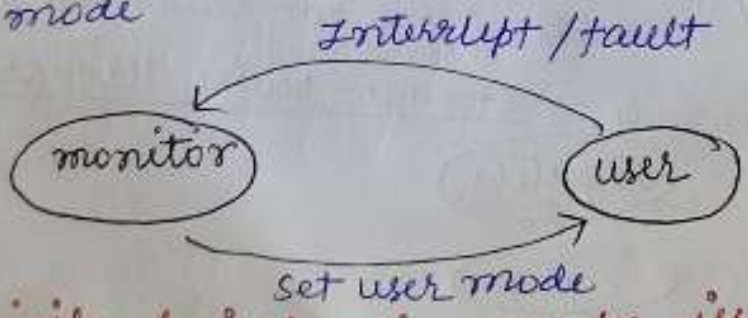
Dual-mode Operation.

→ Sharing system resources requires O.S to ensure that a incorrect program cannot cause other programs to execute incorrectly.
→ Provide hardware support to differentiate b/w at least two modes of operation.

1. User mode :- execution done on behalf of user.
2. monitor mode :- (also kernel mode or system mode) - execution done on behalf of O.S.

Dual-mode Operation (cont.)

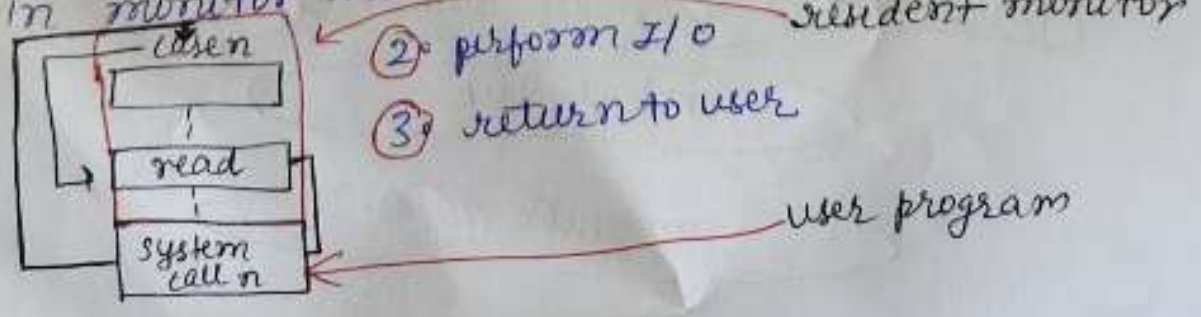
→ mode bit added to computer hardware to indicate the current mode :- monitor (0) or user (1)
→ when an interrupt or fault occurs hardware switches to monitor mode



[Privileged instructions can be issued only in monitor mode.]

I/O Protection

→ All I/O instructions are privilged instructions.
→ must ensure that user program could never gain control of the computer in monitor mode.



⭐ memory Protection

→ Must provide memory protection at least for the interrupt vector and the interrupt services routines.

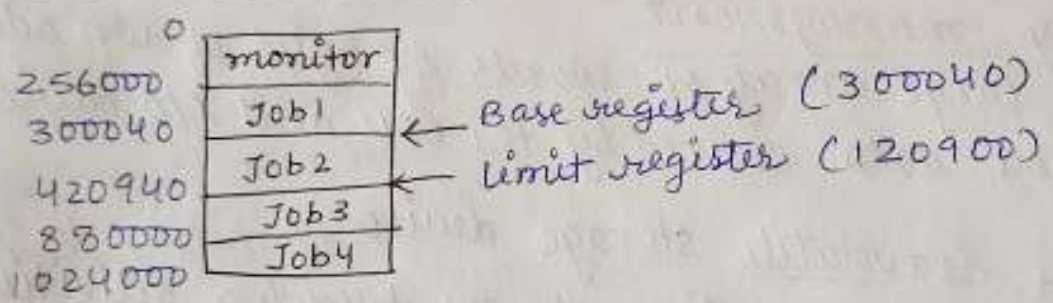
→ In order to have memory protection, add two registers that determines the range of legal addresses a program may access

→ Base register :- holds the smallest legal physical memory address.

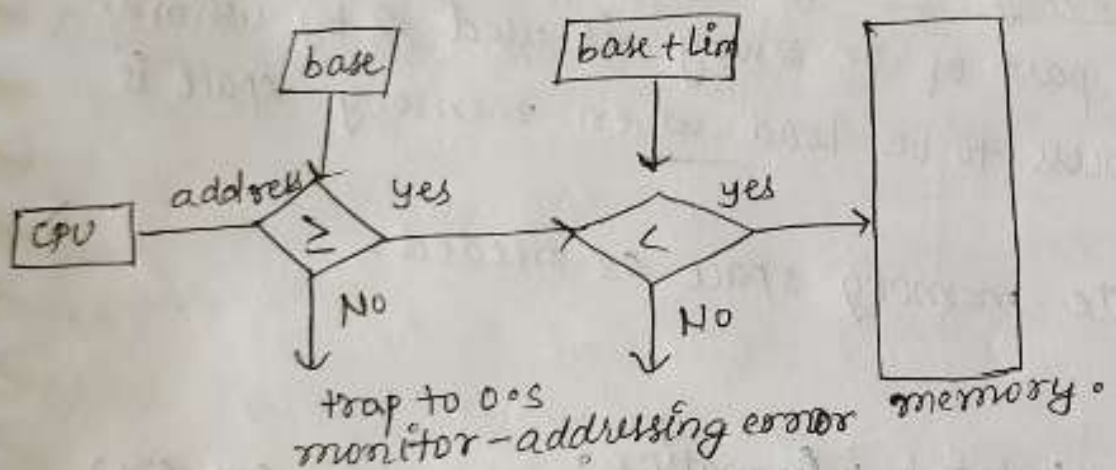
→ Limit register :- contains the size of the range.

→ memory outside the defined range is protected.

use of a Base & Limit Register



⭐ Hardware address Protection



⭐ Hardware protection.

→ When executing in monitor mode, the O.S has unrestricted access to both monitor and user's memory.

→ The load instructions for the base and limit registers are privileged instructions.

⭐ CPU protection

→ Timer – interrupt computer after specified period to ensure O.S maintain control.

→ Timer is decremented every clock tick.

→ When timer reaches the value 0, an interrupt occures.

→ Timer commonly used to implement time sharing.

→ Timer also used to compute the current time

→ load-timer is a privileged instruction.

★ Process management

→ A process is an program in execution.
→ process need resources, including CPU time, memory, files & I/O devices, to accomplish its task.

★ Operating system's work in process managment.

i) Process creation and deletion.
ii) process suspension and resumption.
iii) Provision of mechanisms for
   a) process syncronization.
   b) process communication.

★ main-memory managment

→ memory is a large array of words or bytes of own address.
→ quickly accessible data shared by the CPU and I/O devices.

→ main memory is a volatile storage device.
→ operating system is responsible for the following activities in connections with memory managment.
   i) Keep track which part of memory is used & by whom.
   ii) Decide which process to be load when memory space is available in.
   iii) Allocate & deallocate memory space as needed.

★ file management

→ collection of relative/related information.
→ commonly, file represent program (both source & object file) and data.
→ operating system's work.
   → file creation and deletion.
   → Directory creation and deletion.
   → Support of primitive for manipulating files and directories
   → mapping file onto secondary storage.
   → file backup on stable (non-valatile) storage media.

☆ I/O system management

→ It consits of :-
a) A buffer-caching system
b) A general device interface. b/w device-driver.
c) Drivers for specific hardware devices.

⊛ Secondary-storage management

→ Non-volatile
→ Large size
→ morden computer system use disks.
→ work of ~~Operation~~ operating system :-
i) Free space managment.
ii) storage allocation
iii) Disk scheduling

⊛ Networking (Distributed system)

i) collection of processors that do not share memory or clock. each processor has its own memory.
ii) Processors are connected through a communication Network.
iii) communication takes place using a protocol.
iv) provides access to various system resources.
↓
benefits of shared resources.
↓
a) computational speed up
b) Increased data availability
c) Enhanced reliability

⊛ Protectiation System

→ Protection refers to a mechanism for controlling access by programs, processors, or users to both system and user resources.

works of it :- a) distinguish b/w authorized and unauthorized usage
b) Specify the controls to be imposed.
c) provide the means of enforcement.

(★) Command - Interpreter System

→ many command give to the OS by control statements which deal with:

a) process creation and managment.
b) I/O handling
c) secondary - storage managment.
d) main - memory management.
e) file - system accss.
f) protection
g) networking.

(→) Operating System services

- Program execution :- system capability to load a program into memory and to run it.
- I/O Operations :- provide ways for I/O.
- file system menupulation :- programs capability to create, read, write and delete files.
- communications :- exchange of info. b/w processors on same computer or on different system tied together by network. Implemented via shared memory or message passing.
- Error detections :- ensure correct computing by detection of errors in memory hardware, in I/O system etc.

* Addition works of OS to ensure efficient system operations.

(a) Resource allocations :- allocating resources to multiple users or multiple jobs running at the same time.

(b) Accounting :- keep records of which user used how much and what resources. (for usage statistics.

(c) Protection :- ensuring that all access to system resources is controlled.

(★) system calls

(→) Provides the interface b/w running program & the operating system.

→ available as assembly - language instructions.

Ex C, C++ to replace assembly - language inst.

⊛ Three general methods for system calls.
a) Pass parameters in register.
b) store the parameter in a table in memory, and the table address is passed as a parameter in registers.
c) Push(store) the parameters onto the stack by the programs, and pop off the stack by OS.

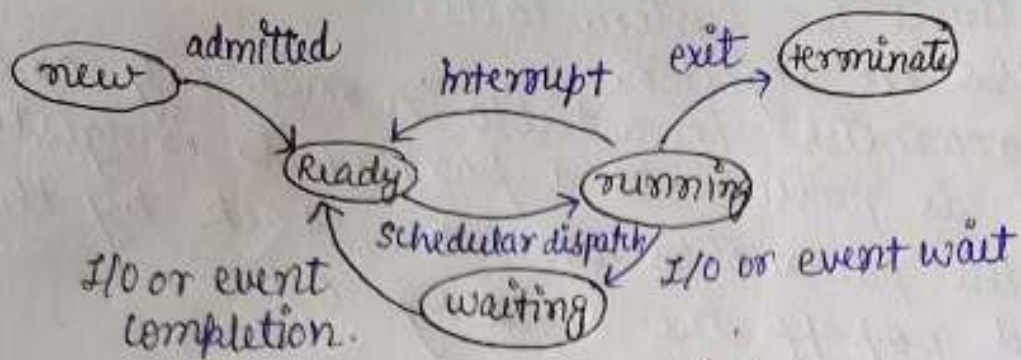
★ Types of system calls
- Process control.
- file managment.
- Device managment.
- Information maintainance.
- communication.

⊛ System Programs
→ it provides convenient environment for program development and execution
divided into :-
a) file manipulation
b) status information.
c) file modification.
d) Programming language support.
e) Program loading and execution.
f) communication.
g) Application programs.

Chapter 4 : Processes

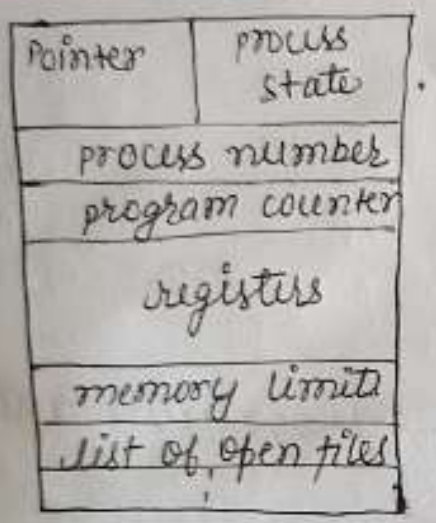
⊛ Process concept.
Process :- a program in execution
- sequential execution

Process State
i) New :- The process is being created.
ii) running :- Instruction is being executed.
iii) waiting :- The process is waiting for event to occur.
iv) Ready :- The process is waiting to be assigned to a process.
v) Terminated :- The process has finished execution.

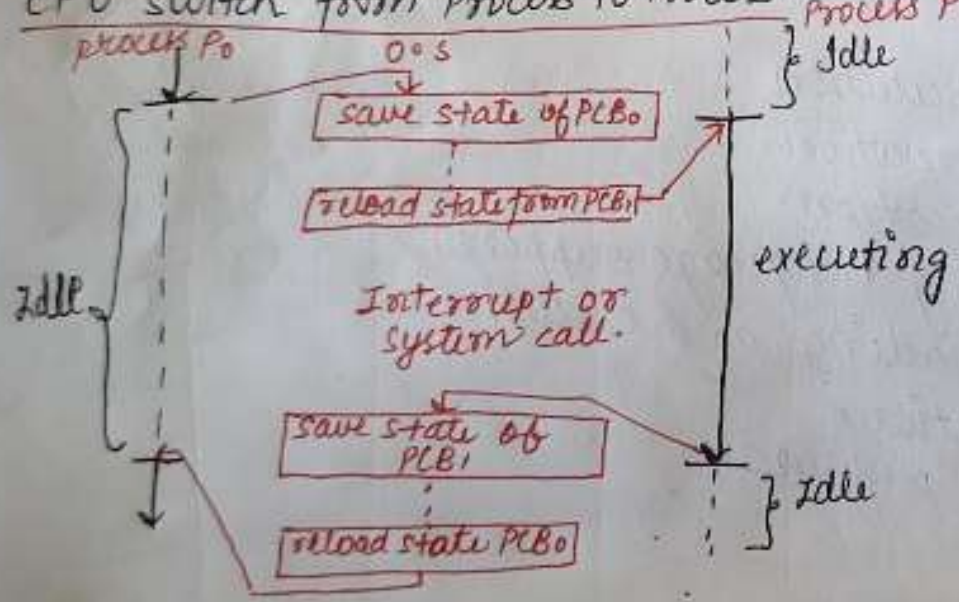


Process state diagram

★ Process control Block (PCB)



CPU switch from Process to Process.



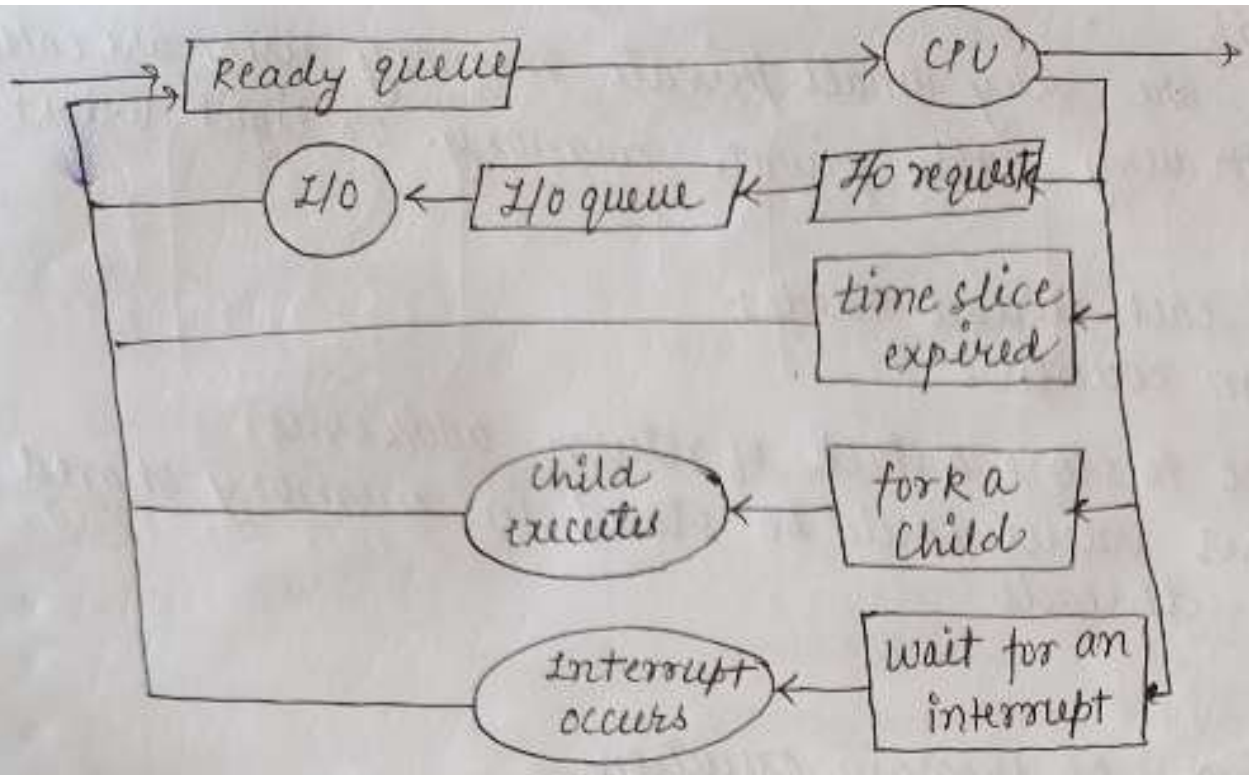
(fx) Process scheduling Queues

i) Job queue :- Set of all the process in the system.

ii) Ready queue :- set of all processes residing in main memory, Ready, waiting to execute.

iii) Device queues :- set of process waiting for I/O devices

Note: processes migrate from one queue to another.

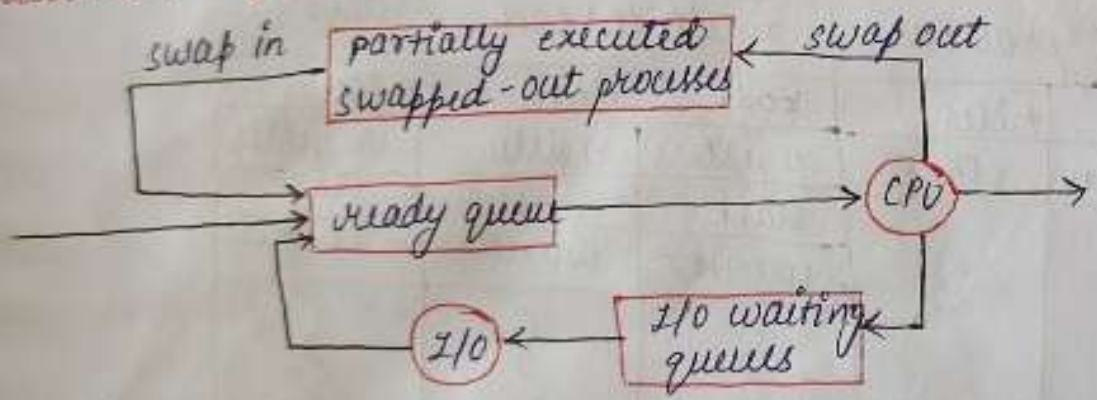


★ Schedulers

★ long-term scheduler :- selects which process should be brought to ready queue.

★ short-term schedular :- selects which process should be executed next and allocates CPU.

Addition of medium term scheduling



Shedulers (cont.)

milliseconds

→ short-term scheduler is invoked very frequently (ms) (fast)

→ long-term schedular is invoked very infrequently (seconds, min.) (may be slow)

→ long Term scheduler controls the degree of multiprogramming.

Processes can be described as either

a) I/O bound process :- spends more time doing I/O than computations, many short CPU bursts.

b) CPU-bound process :- spend more time on computation, few long CPU burst

Ⓐ context-switch :- when process switch it store its current state.

→ time dependent hardware.

Chapter 5 :- Threads

→ Processes which are very small private memory user are called thread or with very little private memory. or light weight proclsus.

→ At a minimum, each thread needs :-

→ a) program counter.

b) a place to store a stack of return addresses.

c) all other values could be stored in memory shared by all threads.

⊛ Threads

→ alternative model of program execution

→ A process creates threads through a system call.

→ threads oprates within process contents.

Process split into two parts

a) Resource state remains with process.

b) CPU state is associated with thread.

Switching b/w threads is less overhead than processes.

single thread processes		
code	data	files
registers	counters	stack

multithread processes.		
code	data	files
registers	register	register
stack	Stack	stack
counter	counter	counter

Process

a) It has heavy weight or resource intensive

b) process switching need interaction b/w OS & process

c) has its own memory & file resources

d) one process blocked then no other execution

e) use more resources

~~Threads~~ Threads

a) It is light weight taking user resources than a process.

b) no need intraction with OS.

c) share same set of open files child processes.

d) if one blocked than other execute.

e) use less resources.

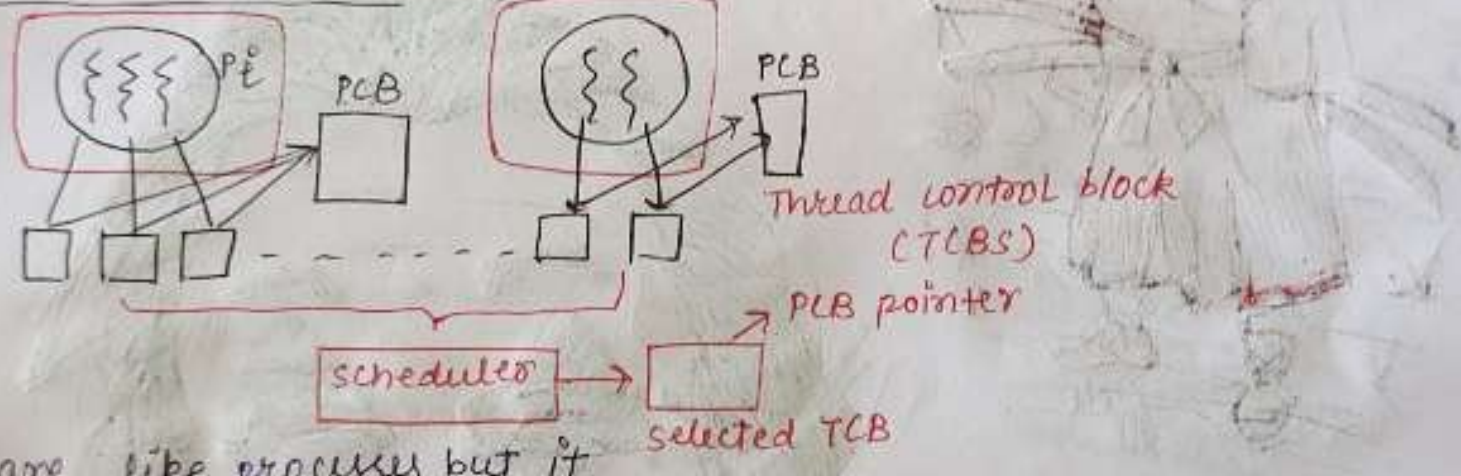
Benefits

i) more efficient communication
ii) simplification of design
iii) Responsiveness
iv) Resource sharing
v) Economy
vi) utilization of MP architecture

→ level of threads

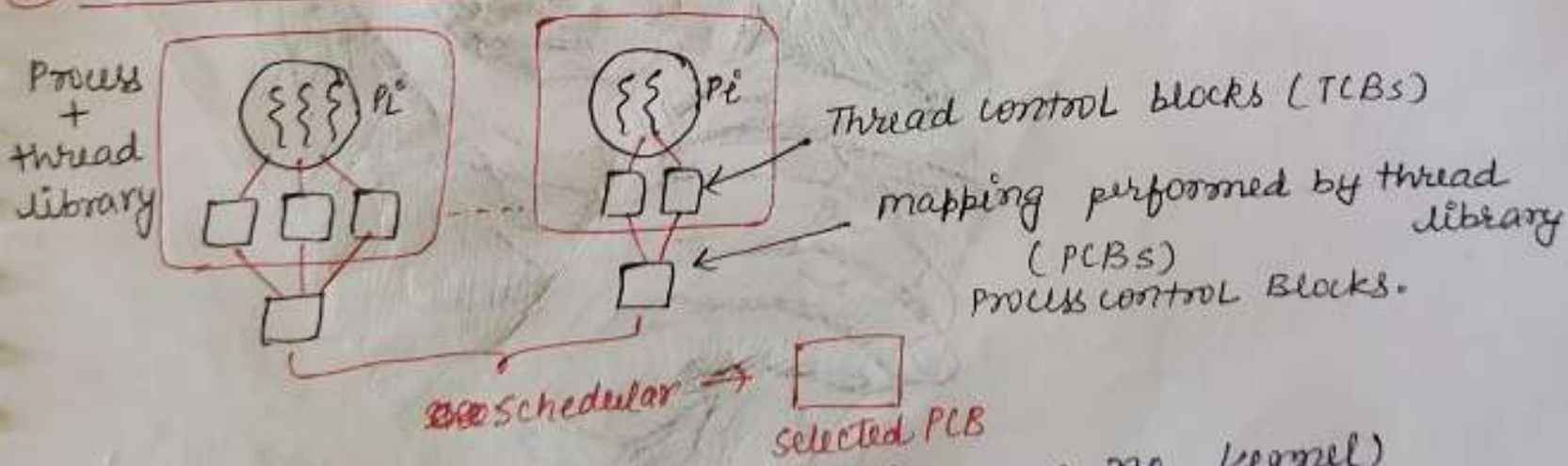
i) kernel-level Threads :- managed by kernel.
ii) user-level thread :- managed by thread library.
iii) Hybrid Threads :- combind of both kernel & user-level threads

★ kernel-level thread



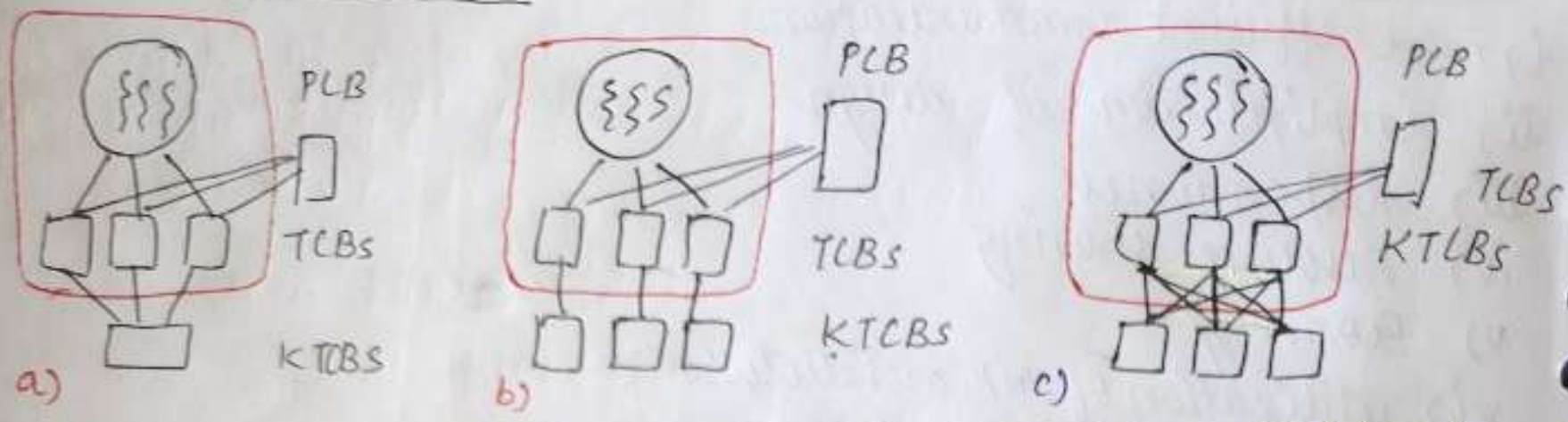
they are like processes but it has smaller amount of state information.

★ User-level threads



→ fast thread switch b/w threads (because no kernel)
→ one Block all threads block
→ NO concurrency or parallelism.

Hybrid - Thread models



- can provide a combination of parallelism and low overhead.