1. a) Why is dual mode operation in OS required? Explain (4M)

- Sharing system resources requires operating system to ensure that an incorrect program cannot cause other programs to execute incorrectly.
- Provide hardware support to differentiate between at least two modes of operations.

  1. User mode – execution done on behalf of a user.

  2. Monitor mode (also kernel mode or system mode) – execution done on behalf of operating system.
- 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.

b) Explain the salient features of

i) Distributed OS 3M

- A distributed system is one in which H/w or S/w components located at the networked computers communicate & co ordinate their actions only by passing messages.
- A distributed systems looks to its user like an ordinary OS but runs on multiple, Independent CPU's.
- Distributed systems depends on networking for their functionality which allows for communication so that distributed systems are able to share computational tasks and provides rich set of features to users.
- N/w may vary by the protocols used, distance between nodes & transport media.

  Protocols->TCP/IP, ATM etc.

  Network-> LAN, MAN, WAN etc.

  Transport Media-> copper wires, optical fibers & wireless transmissions

ii) Real time OS 3M

- Real time system is one which were originally used to control autonomous systems like satellites, robots, hydroelectric dams etc.
- Real time system is one that must react to I/p & responds to them quickly.
- A real time system should not be late in response to one event.
- A real time should have well defined time constraints.
- Real time systems are of two types

a. Hard Real Time Systems

b. Soft Real Time Systems

1. A hard real time system guarantees that the critical tasks to be completed on time. This goal requires that all delays in the system be bounded from the retrieval of stored data to time that it takes the OS to finish the request.
2. In soft real time system is a less restrictive one where a critical real time task gets priority over other tasks & retains the property until it completes. Soft real time system is achievable goal that can be mixed with other type of systems. They have limited utility than hard real time systems.
3. Soft real time systems are used in area of multimedia, virtual reality & advanced scientific projects. It cannot be used in robotics or industrial controls due to lack of deadline support.
4. Real time OS uses priority scheduling algorithm to meet the response requirement of a real time application.
5. Soft real time requires two conditions to implement, CPU scheduling must be priority based & dispatch latency should be small.
6. The primary objective of file management in real time systems is usually speed of access, rather than efficient utilization of secondary storage.

2. a) Define an OS? Discuss its role with respect to user and system view points. (2+2+2)M

- ➢ An OS is an intermediary between the user of the computer & the computer hardware.
- ➢ It provides a basis for application program & acts as an intermediary between user of computer & computer hardware.
- ➢ The purpose of an OS is to provide a environment in which the user can execute the program in a convenient & efficient manner.

<u>Views OF OS</u>

1. **<u>User Views:-</u>** The user view of the computer depends on the interface used.

i. Some users may use PC’s. In this the system is designed so that only one user can utilize the resources and mostly for ease of use where the attention is mailnly on performances and not on the resource utilization.

ii. Some users may use a terminal connected to a mainframe or minicomputers.

iii. Other users may access the same computer through other terminals. These users may share resources and exchange information. In this case the OS is designed to maximize resource utilization- so that all available CPU time, memory & I/O are used efficiently.

iv. Other users may sit at workstations, connected to the networks of other workstation and servers. In this case OS is designed to compromise between individual visibility & resource utilization.

2. **<u>System Views</u>:-**

i. We can view system as resource allocator i.e. a computer system has many resources that may be used to solve a problem. The OS acts as a manager of these resources. The OS must decide how to allocate these resources to programs and the users so that it can operate the computer system efficiently and fairly.

ii. A different view of an OS is that it need to control various I/O devices & user programs i.e. an OS is a control program used to manage the execution of user program to prevent errors and improper use of the computer.

iii. Resources can be either CPU Time, memory space, file storage space, I/O devices and so on.

The OS must support the following tasks

a. Provide the facility to create, modification of programs & data files using on editors.
b. Access to compilers for translating the user program from high level language to machine language.
c. Provide a loader program to move the compiled program code to computers memory for execution.
d. Provides routines that handle the details of I/O programming.

b) Explain any two types of system calls? (2+2)M

- ➢ System calls may be grouped roughly into 5 categories

1. Process control.
2. File management.
3. Device management.
4. Information maintenance.
5. Communication.

<u>FILE MANAGEMENT</u>

- ➢ System calls can be used to create & deleting of files. System calls may require the name of the files with attributes for creating & deleting of files.
- ➢ Other operation may involve the reading of the file, write & reposition the file after it is opened.
- ➢ Finally we need to close the file.
- ➢ For directories some set of operation are to be performed. Sometimes we require to reset some of the attributes on files & directories. The system call get file attribute & set file attribute are used for this type of operation.

<u>DEVICE MANAGEMENT</u>:-

- The system calls are also used for accessing devices.
- Many of the system calls used for files are also used for devices.
- In multi user environment the requirement are made to use the device. After using the device must be released using release system call the device is free to be used by another user. These function are similar to open & close system calls of files.
- Read, write & reposition system calls may be used with devices.
- MS-DOS & UNIX merge the I/O devices & the files to form file services structure. In file device structure I/O devices are identified by file names.

INFORMATION MAINTAINANCE:-

- Many system calls are used to transfer information between user program & OS.

Example:- Most systems have the system calls to return the current time & date, number of current users, version number of OS, amount of free m/y or disk space & so on.

- In addition the OS keeps information about all its processes & there are system calls to access this information.

COMMUNICATION:-

There are two modes of communication,

1. Message Passing Models:-

- In this information is exchanged using inter-process communication facility provided by OS.
- Before communication the connection should be opened.
- The name of the other communicating party should be known, it ca be on the same computer or it can be on another computer connected by a computer network.
- Each computer in a network may have a host name like IP name similarly each process can have a process name which can be translated into equivalent identifier by OS.
- The get host id & process id system call do this translation. These identifiers are then passed to the open & close connection system calls.
- The recipient process must give its permission for communication to take place with an accept connection call.
- Most processes receive the connection through special purpose system program dedicated for that purpose called daemons. The daemon on the server side is called server daemon & the daemon on the client side is called client daemon.

2. Shared Memory:-

- In this the processes uses the map m/y system calls to gain access to m/y owned by another process.
- The OS tries to prevent one process from accessing another process m/y.
- In shared m/y this restriction is eliminated and they exchange information by reading and writing data in shared areas. These areas are located by these processes and not under OS control.
- They should ensure that they are not writing to same m/y area.

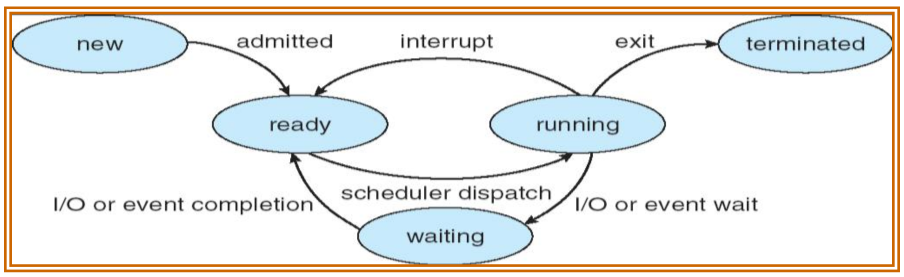
7. Both these types are commonly used in OS and some even implement both.
8. Message passing is useful when small number of data need to be exchanged since no conflicts are to be avoided and it is easier to implement than in shared m/y. Shared m/y allows maximum speed and convenience of communication as it is done at m/y speed when within a computer.

3. a) What is PCB? What are the different states in which a process can be in its life cycle, discuss with the help of state transition diagram. (1+4)M

A process in an operating system is represented by a data structure known as a process control block (PCB) or process descriptor. The PCB contains important information about the specific process including

- The current state of the process i.e., whether it is ready, running, waiting, or whatever.
- Unique identification of the process in order to track "which is which" information.
- A pointer to parent process.
- Similarly, a pointer to child process (if it exists).
- The priority of process (a part of CPU scheduling information).
- Pointers to locate memory of processes.
- A register save area.
- The processor it is running on.

A process goes through a series of discrete process states.



- New State :   The process being created.
- Terminated State: The process has finished execution.
- Blocked (waiting) State: When a process blocks, it does so because logically it cannot continue, typically because it is waiting for input that is not yet available. Formally, a process is said to be blocked if it is waiting for some event to happen (such as an I/O completion) before it can proceed. In this state a process is unable to run until some external event happens.
- Running State: A process is said t be running if it currently has the CPU, that is, actually using the CPU at that particular instant.
- Ready State: A process is said to be ready if it use a CPU if one were available. It is runable but temporarily stopped to let another process run.

b) Is CPU scheduling necessary? Discuss the five different scheduling criteria used in comparing scheduling mechanisms? (5M)

Fairness is important under all circumstances. A scheduler makes sure that each process gets its fair share of the CPU and no process can suffer indefinite postponement.

Scheduling Criteria

1. CPU utilization – keep the CPU as busy as possible
2. Throughput – # of processes that complete their execution per time unit
3. Turnaround time – amount of time to execute a particular process
4. Waiting time – amount of time a process has been waiting in the ready queue
5. Response time – amount of time it takes from when a request was submitted until the first response is produced, not output (for time-sharing environment)

4. a) What are virtual machines? Explain the benefits of creating virtual machines? (5M)

- ➢ A virtual machine takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware
- ➢ A virtual machine provides an interface identical to the underlying bare hardware.
- ➢ The operating system host creates the illusion that a process has its own processor and virtual memory
- ➢ Each guest is provided with a (virtual) copy of underlying computer

Benefits are:

- ➢ Fundamentally, multiple execution environments (different operating systems) can share the same hardware
- ➢ Protect from each other

- Some sharing of file can be permitted, controlled
- Commutate with each other, other physical systems via networking
- Useful for development, testing
- Consolidation of many low-resource use systems onto fewer busier systems.

b. What are schedulers and different types of schedulers? (4M)

The following are the different type of schedulers
1. **Long-term scheduler (or job scheduler) –** selects which processes should be brought into the ready queue.
2. **Short-term scheduler (or CPU scheduler) –** selects which process should be executed next and allocates CPU.
3. Medium-term schedulers
-> Short-term scheduler is invoked very frequently (milliseconds) (must be fast)
-> Long-term scheduler is invoked very infrequently (seconds, minutes) (may be slow)
-> The long-term scheduler controls the *degree of multiprogramming*
->Processes can be described as either:
 I/O-bound process – spends more time doing I/O than computations, many short CPU bursts

 CPU-bound process – spends more time doing computations; few very long CPU bursts

5. a. Define IPC. What are the different methods used for logical implementations of message passing systems?

1. Mechanism for processes to communicate and to synchronize their actions
2. Message system – processes communicate with each other without resorting to shared variables
3. IPC facility provides two operations:
send(*message*) – message size fixed or variable
receive(*message*)
4. If *P* and *Q* wish to communicate, they need to:
establish a *communication link* between them
exchange messages via send/receive
5. Implementation of communication link
physical (e.g., shared memory, hardware bus)

logical (e.g., logical properties)

Direct Communication

1. Processes must name each other explicitly:

send (P, message) – send a message to process P

receive(Q, message) – receive a message from process Q

2. Properties of communication link

Links are established automatically

A link is associated with exactly one pair of communicating processes

Between each pair there exists exactly one link

The link may be unidirectional, but is usually bi-directional

Indirect Communication

1. Messages are directed and received from mailboxes (also referred to as ports)

Each mailbox has a unique id

Processes can communicate only if they share a mailbox

2. Properties of communication link

Link established only if processes share a common mailbox

A link may be associated with many processes

Each pair of processes may share several communication links

Link may be unidirectional or bi-directional

3. Operations

create a new mailbox

send and receive messages through mailbox

destroy a mailbox

4.Primitives are defined as:

send(A, message) – send a message to mailbox A

receive(A, message) – receive a message from mailbox A

5.Mailbox sharing

P1, P2, and P3 share mailbox A

P1, sends; P2 and P3 receive

Who gets the message?

6. Solutions

Allow a link to be associated with at most two processes

Allow only one process at a time to execute a receive operation

Allow the system to select arbitrarily the receiver. Sender is notified who the receiver was.

Synchronization

1. Message passing may be either blocking or non-blocking

2. Blocking is considered synchronous

->Blocking send has the sender block until the message is received.

->Blocking receive has the receiver block until a message is available.

3. Non-blocking is considered asynchronous

->Non-blocking send has the sender send the message and continue.

->Non-blocking receive has the receiver receive a valid message or null.

Buffering

->Queue of messages attached to the link; implemented in one of three ways

1. Zero capacity – 0 messages sender must wait for receiver (rendezvous)

2. Bounded capacity – finite length of n messages Sender must wait if link full

3. Unbounded capacity – infinite length sender never waits

b. What is producer consumer problem? Give a solution to the problem using shared memory.

6. What is a thread and explain its benefits?

A thread is a single sequence stream within in a process. Because threads have some of the properties of processes, they are sometimes called lightweight processes. In a process, threads allow multiple executions of streams. In many respect, threads are popular way to improve application through parallelism.

Following are some reasons why we use threads in designing operating systems.
1. A process with multiple threads make a great server for example printer server.

2. Because threads can share common data, they do not need to use interprocess communication.

3. Because of the very nature, threads can take advantage of multiprocessors.

4. Responsiveness

5. Resource Sharing

6. Economy

7. Utilization of MP Architectures

Threads are cheap in the sense that
1. They only need a stack and storage for registers therefore, threads are cheap to create.

2. Threads use very little resources of an operating system in which they are working. That is, threads do not need new address space, global data, program code or operating system resources.

3. Context switching are fast when working with threads. The reason is that we only have to save and/or restore PC, SP and registers.

But this cheapness does not come free - the biggest drawback is that there is no protection between threads.

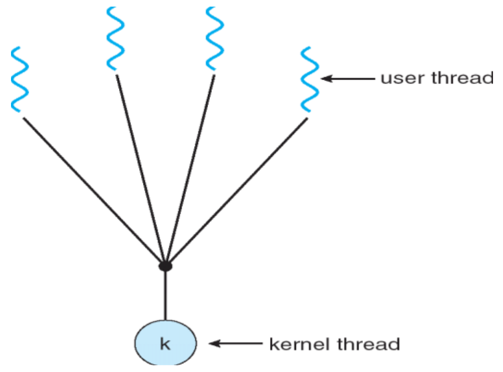
b. Explain different threading models used for establishing a relationship between user and kernel thread along with advantages and disadvantages of each model.

Multithreading Models

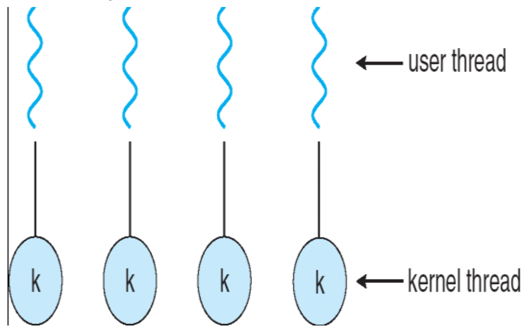
- ❖ Many-to-One
- ❖ One-to-One
- ❖ Many-to-Many

Many-to-One

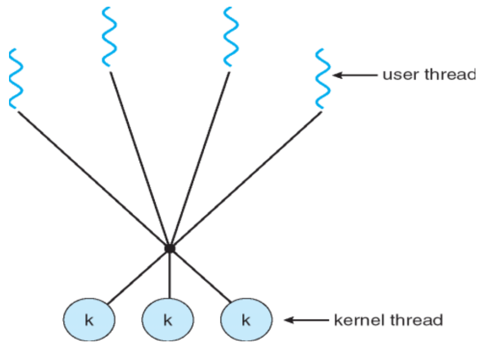
Many user-level threads mapped to single kernel thread
->Examples:
->Solaris Green Threads
->GNU Portable Threads



One-to-One
1. Each user-level thread maps to kernel thread
2. Examples
- ❖ Windows NT/XP/2000
- ❖ Linux



Many-to-Many Model
1. Allows many user level threads to be mapped to many kernel threads.
2. Allows the operating system to create a sufficient number of kernel threads.
3. Solaris prior to version 9.
4. Windows NT/2000 with the *ThreadFiber* package.



7. Explain the following OS structures with a neat diagram:

SYSTEM STRUCTURES
- ➢ Modern OS is large & complex.
- ➢ OS consists of different types of components.
- ➢ These components are interconnected & melded into kernel.
- ➢ For designing the system different types of structures are used. They are,

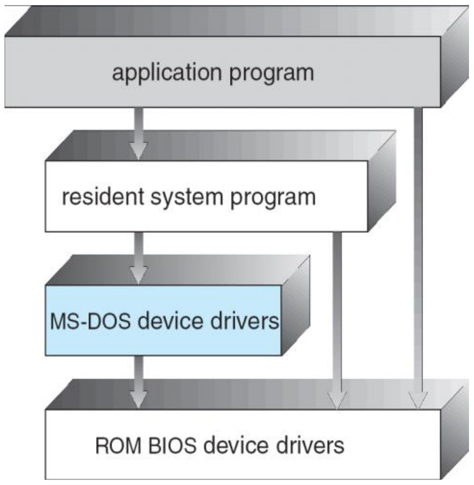
a. Simple structures.

b. Layered structured.

c. Micro kernels.

Simple Structures
- ➢ Simple structure OS are small, simple & limited systems.

- The structure is not well defined
- MS-DOS is an example of simple structure OS.
- MS-DOS layer structure is shown below

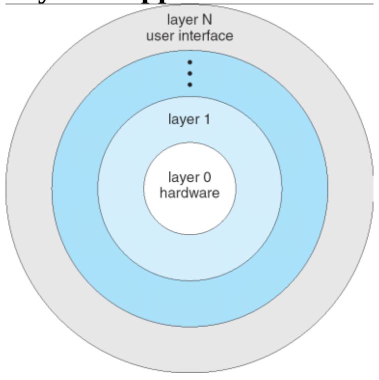


UNIX consisted of two separate modules
a. Kernel

b. The system programs.

- Kernel is further separated into series of interfaces & device drivers which were added & expanded as the UNIX evolved over years.
- The kernel also provides the CPU scheduling, file system, m/y management & other OS function through system calls.
- System calls define API to UNIX and system programs commonly available defines the user interface. The programmer and the user interface determines the context that the kernel must support.
- New versions of UNIX are designed to support more advanced H/w. the OS can be broken down into large number of smaller components which are more appropriate than the original MS-DOS.

Layered Approach



In this OS is divided into number of layers, where one layer is built on the top of another layer. The bottom layer is hardware and higher layer is the user interface.

- An OS is an implementation of abstract object i.e. the encapsulation of data & operation to manipulate these data.
- The main advantage of layered approach is the modularity i.e. each layer uses the services & functions provided by the lower layer. This approach simplifies the debugging & verification. Once first layer is debugged the correct functionality is guaranteed while debugging the second layer. If an error is identified then it is a problem in that layer because the layer below it is already debugged.
- Each layer is designed with only the operations provided by the lower level layers.
- Each layer tries to hide some data structures, operations & hardware from the higher level layers.
- A problem with layered implementation is that they are less efficient then the other types.

Micro Kernels:-

- Micro kernel is a small Os which provides the foundation for modular extensions.
- The main function of the micro kernels is to provide communication facilities between the current program and various services that are running in user space.
- This approach was supposed to provide a high degree of flexibility and modularity.
- This benefits of this approach includes the ease of extending OS. All the new services are added to the user space & do not need the modification of kernel.
- This approach also provides more security & reliability.
- Most of the services will be running as user process rather than the kernel process.
- This was popularized by use in Mach OS.
- Micro kernels in Windows NT provides portability and modularity. Kernel is surrounded by a number of compact sub systems so that task of implementing NT on variety of platform is easy.
- Micro kernel architecture assign only a few essential functions to the kernel including address space, IPC & basic scheduling.
- QNX is the RTOS i.e. also based on micro kernel design.

8. Consider the following set of processes with length of the CPU burst time given in ms: (10M)

Processes	Arrival Time	Burst Time	Priority
P1	0	10	3
P2	0	1	1
P3	3	2	3
P4	5	1	4
P5	10	5	2

a) Draw the Gantt Charts illustrating the execution of these processes using SRTF, Priority and RR (Quantum=2ms) scheduling.

b) What is the turnaround time of each processes for each of the scheduling algorithms in (a).

c) What is the waiting time of each processes for each of the scheduling algorithms in (a).

IAT1-Problem-Solutio
n