CHAPTER 1

UNIT I

Overview of Operating System

Syllabus

1.1 Operating System - Concept, Components of operating system, Operations of OS : Program Management, Resource management, Security and Protection.

Views of OS : User view, System view

1.2 Different Types of Operating systems - Batch operating system, Multiprogrammed, Time Shared OS, Multiprocessor Systems, Distributed Systems, Real time systems. Mobile OS (Android, iOS)

1.3 Command line based OS : DOS, Unix GUI based OS - WINDOWS, LINUX.

1.1 Operating system : Concept

▶ (MSBTE - S-11, W-12)

Q. 1.1.1 What is an Operating System ? (Ref. Sec. 1.1) **S-11, W-12, 2 Marks**

- Among three major categories of software, system software plays very important role. For each an every machine (computer), operating system is required. You are aware that there are three major categories of the software as follows :
  1. System software
  2. Application software
  3. Software development languages advanced and web tools
- An operating system falls in the category of system software. As mentioned above, it is impossible to start or boot the machine without an operating system. OS is a bridge between computer hardware and user.
- Operating system is also called as platform for the machine. If user would like to use readymade applications or even if tries to use system development tools or languages, an operating system should be present on the machine.
- There are many important functions performed by an operating system like memory management, I/O Management, CPU job scheduling etc. You will find elaboration of these functions in detail in next chapters.
- Major operating systems known worldwide are MS-DOS, Microsoft Windows X, MAC, OS/2, UNIX, Linux, Sun Solaris etc. There are different generations of operating system which we are going to discuss later in the same chapter.

1.1.1 Components of Operating System

▶ (MSBTE - W-08, S-14, W-14, S-15, W-16, S-18)

Q. 1.1.2 What are the parts of computer system? Explain with the help of diagram. (Ref. Sec. 1.1.1) **W-08, 4 Marks**

Q. 1.1.3 What are the components of computer system? Explain with the help of diagram. (Ref. Sec. 1.1.1) **S-14, W-14, S-15, W-16, 6 Marks**

Q. 1.1.4 List and draw a neat labelled diagram of four components of a computer system. (Ref. Sec. 1.1.1) **S-18, 4 Marks**

Computer system can be divided into four components

1. Hardware
2. Operating system
3. Application programs
4. Users

1. **Hardware :** Hardware means the physical parts of computer system. These are physical parts of machine which provides basic computing resources. The major classification of hardware devices is subsequent.
   a) **Input Devices :** Keyboard, Mouse, Microphone, Web Camera, Scanner etc.
   b) **Output Devices :** Printer, Monitor (Screen), Plotter, Speakers etc.
   c) **Central processing unit.**
   d) **Storage Devices :** Hard disk, Magnetic tape, Floppy Disk, CD, DVD, HVD, Pen drives **etc.**
   e) **Networking devices :** LAN card, Router, bridge, Hub, switch, modem etc.
   f) **Other devices like** Motherboard, power supply, different drives like DVD writer etc.
2. **Operating system :** Operating system plays important role here. It works like a coordinator or mediator between the computer hardware and user. Since it is

system software, it can interact with hardware. User use different programs or applications called as software. The harmonization of hardware and software is most important job done by OS.

3. **Application programs :** They define the ways in which the system resources are used to solve the computing problems of the users. e.g. : Word processors, compilers, web browsers, database systems, video games etc. Applications are readymade packages ready to use. They utilize the hardware as well as software resources and fulfills user's computing requirement.

4. **Users :** Users are human beings, machines and other computers. Fig. 1.1.1 shows the four major components of computer system.

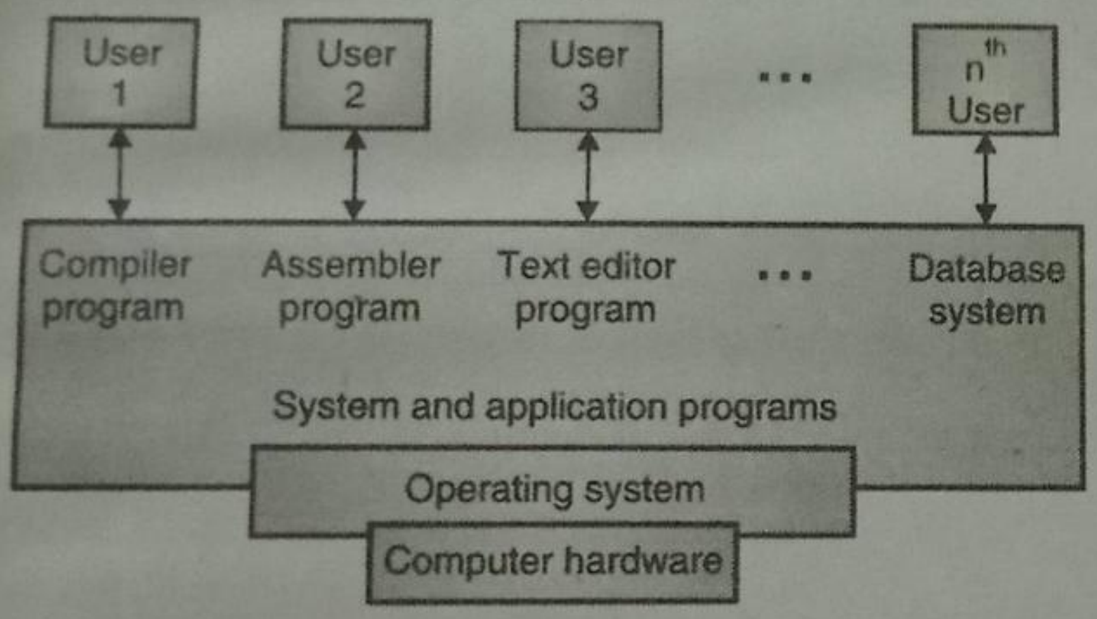


Fig. 1.1.1 : Four components of a computer system

- Computer-system operation involves one or more CPUs, device controllers connected through common bus providing access to shared memory. Concurrent execution of CPUs and devices tries to use memory continuously.

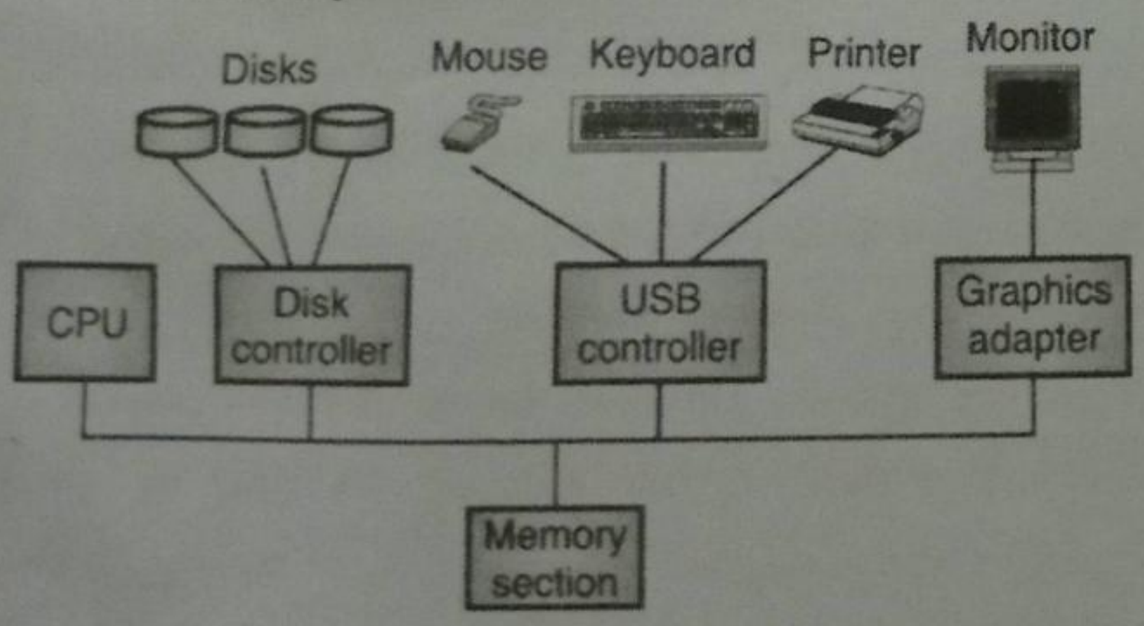


Fig. 1.1.2 : Computer system

- The **software** means set of programs clubbed together to achieve a particular set of tasks or activities. Software is not restricted to only application area but it is used broadly for even system software and other categories of software.
- System software is a set of programs which are used to manage the computer hardware as well as the application programs used by the users. Remaining types of software except system software are used as an application software or computer programming languages or may be front ends, back ends etc.
- System software mainly tries to bridge the gap between computer users and computer hardware. Users do not understand how to operate hardware parts directly without any help of programs. Even computer hardware does not understand what user wish to do.
- An operating system is a major part of system software. The other parts of system software could be compilers, interpreters, translators, some utilities etc. An Operating System (OS) is software that manages computer resources and provides users with an interface used to access those resources.
- Imagine a mobile phone without an operating system, you may not understand how to call someone and even send message.
- An operating system takes care of the files created by the user using different applications. It establishes the communication channel between computer user and hardware
- Operating system processes system data, user input, and responds by allocating and managing tasks. It also manages internal system resources as a service to users and programs of the system. An operating system performs the following basic functions like :

1. Controlling and allocating memory.
2. Prioritizing system requests.
3. Controlling input and output device.
4. Facilitating computer networking and managing files.

- Operating systems is must in almost all devices for communication purposes. Anything made with integrated circuits, such as personal computers, internet servers, cell phones, music players, routers, switches, wireless access points, network storage, game consoles, digital cameras, sewing machines and telescopes.
- When computer starts or boots, the program stored in the firmware i.e. ROM memory gets execute which checks the status of all hardware devices whether they are working properly or not. Read only memory allows to just read the data. Once program written in this memory it is difficult to erase the contents of ROM. Firmware is collection of memories like ROM, EPROM, EEPROM etc.
- After execution of BOOT program, operating system starts executing. The operating system gets loads into main memory i.e. RAM.
- **Kernel** is a key program which is called as heart of an operating system. It is defined as "The one program running at all times on the computer" Everything else is either a system program or an application program. Examples of an operating system are MAC, OS, UNIX, LINUX, SOLARIS, and Windows XP etc.

1.1.2 Functions of an Operating System

▶ **(MSBTE - S-10, S-13, S-14, W-14, S-16, S-18)**

Q. 1.1.5 List any four functions of operating system. (Ref. Sec. 1.1.2) **S-10, S-13, S-14, W-14, S-16, 4 Marks**

Q. 1.1.6 List characteristics of operating system for smooth functioning of a computer system. (Eight points) (Ref. Sec. 1.1.2) **S-18, 4 Marks**

Operating system is responsible for many important functions carried out for smooth functioning of a computer system. Following list demonstrates OS functions.

1. Transferring input from input devices to memory.
2. Transferring input from memory to CPU.
3. Shifting processed data from CPU to output devices.
4. Show messages on the screen in case of input as well as output.
5. Control input and output devices.
6. Loading of application programs from storage devices into main memory.
7. Copy or move data from one storage device to other.
8. Display appropriate error messages by giving the current status of peripherals and processes.
9. Execution of programs and user instructions.
10. Protection of storage devices from overwriting.
11. Maintain details of files or directories with their respective details.
12. To provide security to user's data, programs and other files.

1.1.3 Operating System Evolution and Generations

▶ **(MSBTE - W-10, S-11, S-13, S-13, W-13, W-14, S-15, S-16, W-16, S-17, W-18)**

Q. 1.1.7 Describe evolution and/or generation of operating system. (Ref. Sec. 1.1.3) **W-10, S-11, S-12, S-13, W-13, W-14, S-15, S-16, W-16, W-18, 4 Marks**

Q. 1.1.8 Describe first generation of operating system with its advantages and disadvantages. (Ref. Sec. 1.1.3) **S-17, 4 Marks**

History of an operating system is of a great interest and it is strongly linked with the development and history of different generations of computer system. From this you would be able to find out the development stages in computer system. It had changed drastically in size, capacity, type and speed. It's Cost reduced rapidly till the affordable extent to ordinary persons.

1. Zeroth generations computers : (Mechanical parts)

- The zeroth generation computer was made up of analytical engine which was designed by an English Mathematician Charles Babbage. (Charles Babbage is known as Father of Modern Computer).
- This machine was mechanical design with wheels, cogs, gears etc.
- This machine was able to do certain basic calculations only. Due to slow speed and unreliability, this design was not too popular.
- There was no operating system used with this mechanical hardware.

2. First generation computers

- This generation gave electrical design instead of mechanical one. Due to electrical design the machine speed was fast as compare to mechanical design.
- Some decades later Howard Aiken at Harvard, John Von Neumann at Princeton, J. Eckert and William Mauchely and K. Zuse in Germany succeeded in designing of new electrical machine.
- This machine was a calculating machine with vacuum tubes as a central component. But the vacuum tubes were having the problem of heating (Burning of components due to heating of vacuum tubes) and they were required the large space as compare to other components in the machine.
- Due to the heating problem and space, vacuum tubes were unreliable. In this generation, the programs written were in machine language.
- Again there was no operating system in this generation.

Advantages of first generation of operating system

1. First generation operating system was not full OS. It was just a system program so it was simple.
2. It was easy to use.

Disadvantages

1. This OS was not providing GUI (Graphical User Interface).
2. It was not performing operations like todays operating system.

3. Second generation computers

- Around 1955 the vacuum tubes were replaced by transistors. The problem of space and size was solved. The cost of machine was changed dramatically.
- An assembly language was a second generation language. Second generation computers followed batch system i.e. one by one job were executed in the system.
- The job cards were used to mount the job. The major limitation on computer was it could run just a job at a time.
- Then operator were suppose to unload the job and tape (stored) and then he could have load another job by printing the detailed steps of job done to assure that the full job was done.
- Burroughs was one of the few companies which produced an operating system called Master Control Program (MCP) which had many features of today's operating system such as multiprogramming, Multiprocessing and virtual storage.
- The drawback of transistors was the current leakage. Due to this problem the transistors are replaced in the third generation.

4. Third generation computers

- The transistors were replaced by silicon chip in third generation. In this generation IBM introduced the range of system/360 series of machines, which were known as "Family of Computers". For these range of machines they delivered the following operating systems :

1. CP-67/CMS for the powerful 360/76, using virtual storage.

2. OS/MVT for bigger 360 systems.
3. OS/MFT for the medium 360 systems.
4. DOS/ 360 for the small 360 systems.
5. The systems/360 was based on ICs (Integrated Circuits). These reduced the cost of machine and also it's size. Also the performance was improved.

- Following were the advantages of OS :
  1. Use of ICs in hardware
  2. Portability.
  3. Job control languages.
  4. Multiprogramming.
  5. Spooling.
  6. Time sharing.

5. Fourth generation computers

- In fourth generation computers the technique used was "Large Scale Integration". This technique packed thousands of transistors on small area of silicon chip. From this phase, the fast development of microcomputers started.
- Control Program or Microcomputer (CP/M) was almost the first operating system on the Microcomputer. It was developed on Intel 8080 in 1974 as a first system. PL/M was used instead of assembly language.
- With many developments in microcomputers DOS, UNIX, Linux, GUI i.e. windows operating were invented.
- NOS i.e. Network Operating System development started from the same phase.
- In short the history or evolution of operating system took place as follows :
  1. **The 1st generation (1945-1955) :** Vacuum tubes and plug boards.
  2. **The 2nd generation (1955-1965) :** Transistors and batch systems.
  3. **The 3rd generation (1965-1980) :** ICs and multiprogramming.
  4. **The 4th generation (1980-1990) :** Personal computers.
  5. The latest operating systems (network, distributed, etc.).

1.1.4 Operations of OS

- There are various operations done by OS. They can be called as functions of OS.
- Here we are going to discuss about Program Management, Resource Management and Security and Protection operations.
- These operations are features of contemporary operating systems. They play very important role to make operating system powerful.
- Following is the description of some of the operations of OS.

1. Program Management

- Instructions of the program are executed by the CPU only. Without CPU, program cannot do anything.
- Process management is also called as program management.
- Process is nothing but a program in execution. In a time –shared environment, compiler is a process. Also program for word processing used by single user is known as a process.
- Even sending a document for the printing is also known as a process.
- Every process needs resources to execute it. It includes memory, file, I/O devices and very importantly CPU.
- These resources are allocated to the process at the time of creation or running.
- Along with various physical and logical resources process may need some data as an input.

- A process is collection of threads. Thread is the smallest unit of CPU work.
- Every program is process and it has got a program counter that specifies the next instruction to be executed.
- Single process has got only one counter and generally program is executed sequentially.
- In the process CPU executes instructions one by one till that process is completed.
- Operating system may support single threading or multithreading operating system.
- CPU can process only single thread at a time. Two different processes can be part of a single process.
- A multithreaded process model may contain many programs with many program counters. Every counter will point to its own new instruction to be executed.
- Some threads are user's threads and some are system's thread created by an operating system.
- A process is the unit of work in a system. These processes can be run simultaneously by OS.
- The technique is to run multiple threads is called as multiplexing.
- OS performs program management or process management and is responsible for the following operations:

1. Scheduling of processes as well as threads on the CPUs
2. Creation and deletion of system as well as user's processes.
3. Process suspension and its resuming mechanism.
4. Manage process synchronization
5. Fully support process communication

2. Resource Management

- There are different resources that need to be provided to the different programs or processes.
- Examples of resources could be I/O devices, CPU, memory etc. These resources are physical devices.
- There could be logical resources like data or files. While multiple processes are running, they may demand same resources.
- Operating system uses different algorithms to provide resources to the processes as per their priorities.
- Resource allocation is one of the biggest and critical activities done by operating system.
- For this OS has to continuously keep communication with the different resources, check its availability and allocate it to the process.
- It also does job of de-allocation of the resources. CPU job scheduling is one of the important operations of OS.
- Allocating CPU to the processes for the execution is quite tedious task.

3. Security and Protection

- In multiuser environment data regulation is very important. Security of the data and its protection is an important thing.
- OS needs to take care that resources must be allocated to the processes that are having authorization.
- Processes are executed only within its address space. No process can get the control of CPU unless it is released from other process.
- OS maintains the integrity of the different devices. Protection is a mechanism used for controlling access of users or processes to the resources of the computer system.
- Protection mechanism is used to enforce the control that needs to be imposed and enforced the controls.
- It helps in improving reliability by detection of latent errors at the interface among various computer systems.
- Operating System helps in making the difference between authorization and un-authorization.

- The main function of security is to defend the system from attacks done internally and externally.
- The job of security to defend a system from external and internal attacks. Attacks can be from viruses, worms or it could be denial-of-service.
- Protection and Security mechanism must be able to differentiate among its users along with their rights and privileges.
- These two mechanisms are becoming very challenging day by day because of introduction of networking and Internet.

1.1.5 User's View and System's View

- User's view or system's view are two important views to be discussed to understand concept of view. View is nothing but the way to look at something.

1. User's View

- User's view is decided as per the interface used. In **single standalone system**, user may have his/her own set of devices to be used. They are not shared by anybody else.
- All resources are used by the same machine and its user. Optimization is used for resource management. This is to increase the performance of machine or user program.
- In other cases user can be working on one of the **terminals of mainframe computer or minicomputer.** User has got only his/her I/O devices. So multiple users are working on the same machine by sharing CPU and other required resources.
- One more case can be user working on network systems where he /she may have all dedicated devices at their disposal. But still they share **server and other workstations**.
- Other case may include use of mobile devices like **smart phones or laptops** etc.

2. System's view

- System's view is different as compare to user's view. Operating system is considered to be a closest program to the hardware.
- OS mainly works as a **resource allocator.** It allocates different resources to the processes. They can be logical or physical.
- For example I/O devices, CPU, memory , files , data etc. It decides whether to allocate or de-allocate the devices.
- Another view of system is that OS works as a controller to keep control on various activities happening within the system.

1.2 Types of Operating Systems (Mainframe machines)

▶ **(MSBTE - W-09, S-10, W-15)**

Q. 1.2.1 Define operating system. State the different types of operating system.
(Ref. Sec. 1.2) **W-09, 4 Marks**

Q. 1.2.2 List different types of operating system.
(Ref. Sec. 1.2) **S-10, W-15, 4 Marks**

- As discussed in the previous section, operating system can be defined as a system software without which machine can not boot.
- In another words Operating system can be defined as one of the most important software that drives the hardware as well as software present on the machine or computer.
- There are various types of operating systems available as per the requirements of processing. Among these types let's discuss the major types as follows :

1. Batch processing system.
2. Multiprogramming system.
3. Multitasking.
4. Time sharing system.
5. Desktop systems.

1.2.1 Batch Processing System

▶ **(MSBTE - W-08, S-09, S-13, W-14, S-15, W-18)**

Q. 1.2.3 Explain batch processing operating system with the help of schematic representation. (Ref. Sec. 1.2.1) **S-09, S-13, W-14, S-15, 4 Marks**

Q. 1.2.4 Explain batch monitoring functions. (Ref. Sec. 1.2.1) **W-08, W-18, 4 Marks**

- In early days punch cards were used to record the user's jobs. But this was manual and tedious job which was done by human operator.
- To avoid the continuous human operators devotion the concept of batch processing grown up.
- A batch stands for the sequence of user jobs. In this type of OS the job sequence was decided by an operator.
- The jobs were recognized with the special markers which were indicating the start and the end of those jobs.
- In system the jobs were submitted to the Batch Processing System (BPS) and then those were executed automatically without any operator interruption.
- After execution of a job, it was send to that particular user who owes it. Each job was an independent job in a batch.
- This was achieved by automating transition from execution of job to that of the next job of the same batch.
- Batch monitor or supervisor was the major component which permanently placed in the part of the memory.
- Another part of memory is used to process user's job. i.e. the current job in the batch.

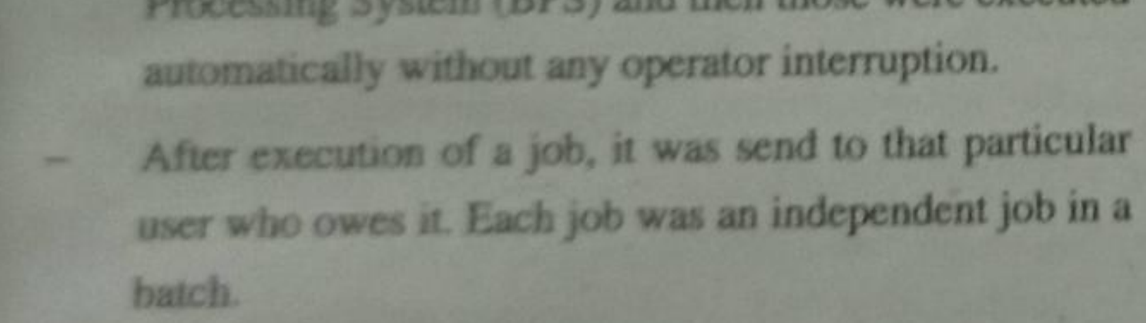


Fig. 1.2.1 : Schematic of batch processing system

- The left side of diagram is memory map in which half portion is occupied by batch monitor known as system area and the remaining memory contains current job of batch for execution.
- The major parts of batch monitor were loader and control interpreter card which does loading of jobs and controls input and output respectively.

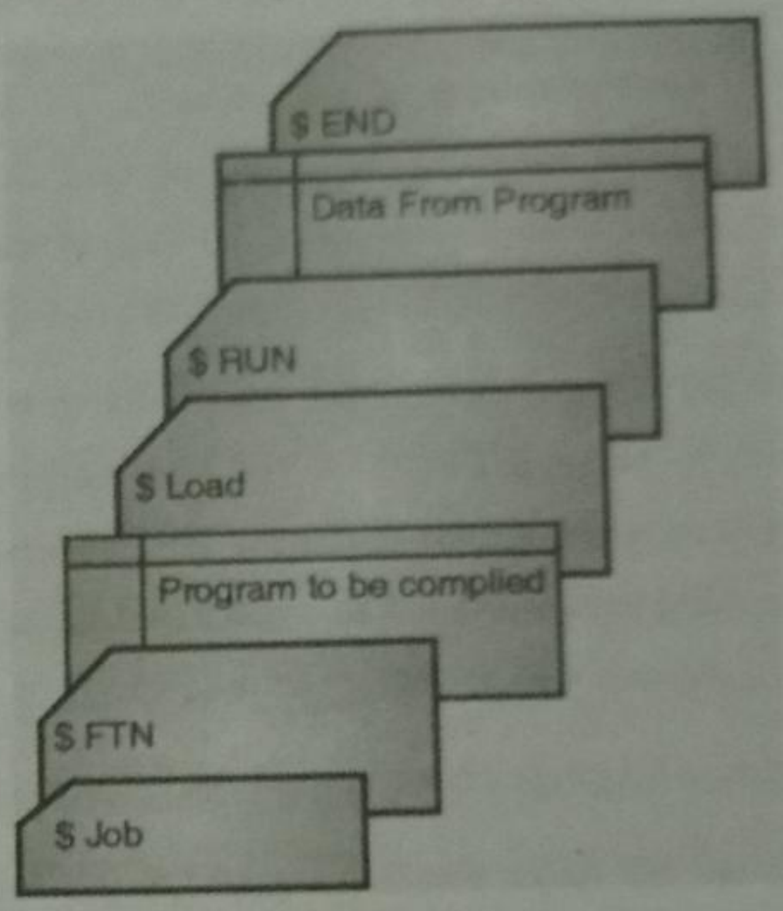


Fig. 1.2.2 : Batch of jobs

Batch monitors functions

- Batch monitor keeps control on processing environment.
- These functions can be categorized into three main tasks ;

1. Scheduling
2. Memory Management
3. Sharing and protections.

- The first two functions were done before the execution of program and third one was done after job was executed.

1. Scheduling

- Scheduling is the activity of determining which service request should be handled next by a server. Here

service request means the user jobs & server is nothing but CPU.

- Batch processing uses FCFS- first come first serve scheduling criterion. i.e. the job which comes first is executed first. i.e. as per the job sequence jobs are executed.

2. Memory management

- During batch processing operations, memory is divided into two parts called System area and user area. An issue of partitioning of memory for system area and user area is handled by an operating system area.
- Some part of the monitor i.e. some part of code of monitor is permanently stored in the memory. At the same time other parts are loaded into the transient (temporary) area. This led to system area as resident area of monitor + transient area of monitor.

3. Sharing and protection

- These are the major function which OS provides for the purpose of multi user system. The sharing of the data can be done by choosing an appropriate operating system. This results in reducing the cost of hardware and processing time.
- During the job processing the resources are allocated or de-allocated sequentially.
- The protection functions are more complicated than sharing.
- In short in Batch systems :
  1. User does not interact with the system.
  2. Jobs with similar needs were grouped into Batches.
  3. Reads a stream of jobs operates on it and generates output.
  4. First come, first served i.e. job which comes first gets execute first.
  5. It can use **Spooling technique.**

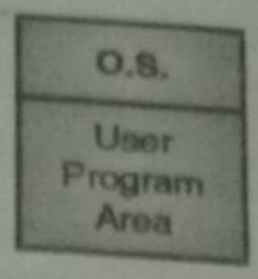


Fig. 1.2.3 : Memory layout of a batch system

☞ **Advantages of batch monitors functions**

1. Jobs are executed in an order.
2. Execution is systematic.
3. Disadvantages:
4. Time consuming
5. Difficult to set the job priorities

1.2.2 Multiprogramming

▶ **(MSBTE - S-09, S-10, W-10, W-15, S-17, W-17)**

Q. 1.2.5 Describe multiprogramming. (Ref. Sec. 1.2.2) **S-09, W-10, W-15, W-17, 5 Marks**

Q. 1.2.6 Explain multiprogrammed O.S. with suitable diagram. (Ref. Sec. 1.2.2) **S-17, 4 Marks**

Q. 1.2.7 Explain spooling. (Ref. Sec. 1.2.2) **S-10, 2 Marks**

- Multiprogramming is the most important feature of an operating systems.
- A single user cannot keep I/O devices as well as CPU busy. Using multiprogramming techniques we can avoid CPU and I/O device idle time and maximum processing can be done.
- From the pool of jobs operating system picks one job and begins executing it. While execution due to some reasons job may wait for some time e.g. may be waiting for storage or for any sort of input/ Output operation.
- In non-multiprogramming system OS may wait or idle. But in multiprogramming system an operating system will switch to next job and will start executing it.
- When this current job waits operating system switches to another job and so on. This way multiprogramming technique continuously keeps CPU and I/O devices in use.

- The other alternatives for keeping CPU and other devices continuously in use, we can use spooling and buffering techniques. But multiprogramming is more useful as compare to spooling and buffering.

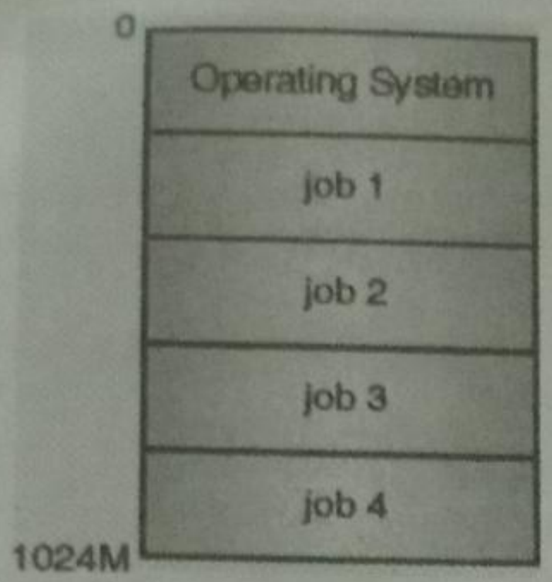


Fig. 1.2.4 : Multi-programming

(a) Buffering

- The solution to the slowness of I/O devices is buffering. In buffering when input devices accept data, it transfers to CPU for process.
- At this time instead of keeping I/O devices idle, input devices are instructed to accept next data. When it finishes accepting new data, CPU gets free and starts processing next data.
- The temporary memory is used for buffering or storing the data. In the same way output devices can also be kept continuously busy.
- In buffering interrupts are used to find out the working status of I/O devices. These interrupt informs whether I/O devices are working or finished their job.
- As soon as I/O devices finish their work, interrupt interrupts the CPU. When CPU is interrupted, the CPU stops its current job and transfers it to the fixed location.
- Interrupt service also checks whether buffer is full or empty and then buffer operation takes place.
- In short buffering can be defined, as "An I/O buffer is a memory area temporarily used to implement on I/O operations."

(b) Spooling

- When the request is made to the printer to output a time, the time is copied into a system buffer and written to disk. When the job is completed the output is actually printed, this form of programming is known as **spooling.**
- The spooling is an acronym for **Simultaneous Peripheral Operation On-Line.** The large buffer used for spooling is disk.
- Due to spooling it is possible for the input devices to accept more data, also it is possible to store the output till O/P devices gets ready for accepting the data.
- Buffering overlaps the I/O devices job with it`s own computations. But spooling overlaps I/O devices' one job with the computation of other jobs.
- In simple spooling system while one job is printed at the same time another job may get inputted. Thus spooling can keep CPU and input /output devices as busy as possible at higher rate as compare to buffering.
- An important data structure provided by spooling is job pool. Spooling arranges some jobs having read operation and some which are waiting to run.
- It helps an operating system to arrange the jobs and let OS select job for execution. It increases the CPU utilization.

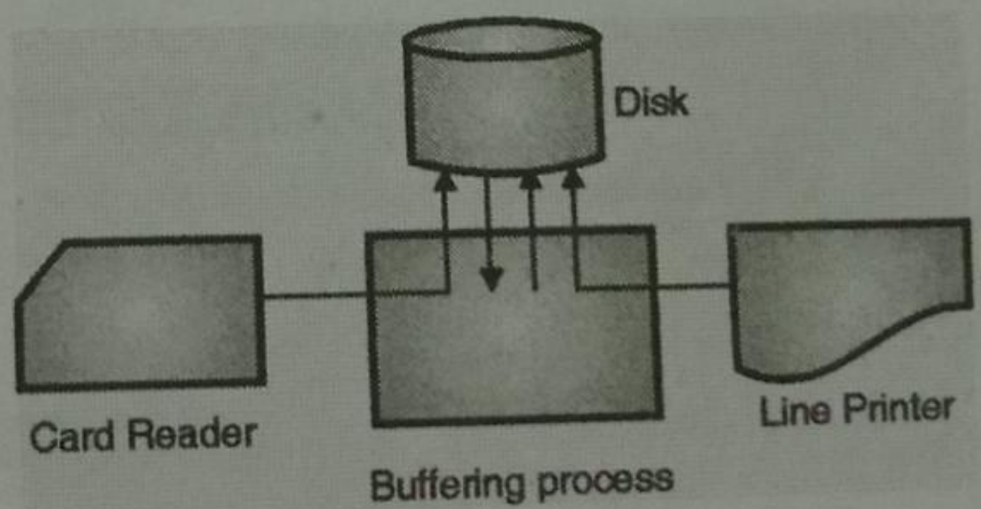


Fig. 1.2.5 : Memory layout of multiprogramming system

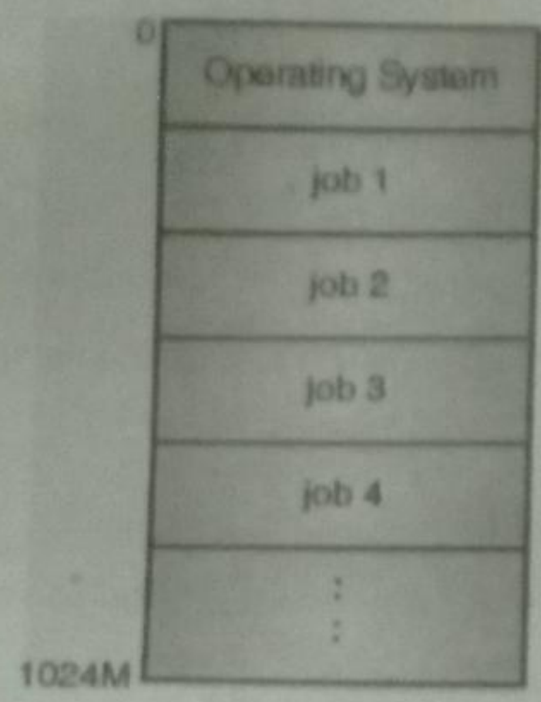


Fig. 1.2.6 : Memory layout of multiprogramming system

- In short in multi-programmed batch systems :
  1. Spooling results in several jobs waiting to get processed.
  2. If several jobs are available on Direct Access device, Job scheduling becomes possible.
  3. An important aspect of Job scheduling is ability to MULTIPROGRAM.
  4. Multiprogramming increases CPU utilization.
  5. When a job needs to wait, CPU switches to another job.
  6. CPU never sits idle.

1.2.3 Multitasking

▶ **(MSBTE - W-08, S-09, W-10, W-11, W-17)**

Q. 1.2.8 What is multitasking operating system? Explain with example. (Ref. Sec. 1.2.3) **W-08, S-09, W-10, W-11, 4 Marks**

Q. 1.2.9 State advantages of multitasking operating system. (Ref. Sec. 1.2.3) **W-08, 4 Marks**

Q. 1.2.10 Describe multitasking. (Ref. Sec. 1.2.3) **W-17, 5 Marks**

- A multitasking operating system is having a unique feature of supporting multiple process executions at a time. These executions are called as concurrent or simultaneous executions.
- In multitasking operating system, the code as well as data of several processes is stored into main memory.
- It also maintains the synchronization between I/O devices and processes. In this way, user can use different application in background and current application in foreground.
- In multitasking the resources are made continuously working.
- The CPU switches from one task to another for reading and processing. Thus idle time of peripherals gets reduced.
- The more programs get reside into main memory as the main memory gets partitioned.
- Many jobs can be handled simultaneously.
- Due to multitasking, many users and user terminal can also be connected to this type of OS.
- The examples of multitasking operating systems are UNIX, Linux and Windows 2000. This operating system allows multiple processes to run at the same time.
- In UNIX and Linux, background process and foreground processes are running at the same time.
- Advantages of Multitasking operating system are as follows :
  (a) Multiple programs can run simultaneously
  (b) It increases the speed of execution by doing resource management.
  (c) It keeps CPU as well as I/O devices continuously busy.
  (d) Multiple programs can reside in the main memory.
  (e) Multiple applications can be opened in background as well as foreground.
  (f) It used time sharing principle to speed up the processing speed.

1.2.3.1 Difference between Multiprogramming and Multitasking Operating Systems

► **(MSBTE - S-10, W-12, S-14, W-18)**

Q. 1.2.11 Differentiate between multiprogramming and multitasking OS. (Ref. Sec. 1.2.3.1)

S-10, W-12, S-14, W-18, 4 Marks

Sr. No.	Multiprogramming operating System	Multitasking operating systems
1.	Multiple jobs are collected and pool of job is formed. Each job is then executed one by one.	Multiple programs are executed at a time.
2.	Programs are executed one by one.	Time sharing principle used while multiple programs are executed
3.	Same program execution takes place in foreground and background by doing resource management.	Multiple applications can be used in foreground and background.
4.	Multiple jobs reside in the main memory.	Multiple programs reside in the main memory.

1.2.4 Time Sharing Operating System

► **(MSBTE - W-10, W-11, S-15, W-17, S-18)**

Q. 1.2.12 Describe time sharing system along with example. (Ref. Sec. 1.2.4)

W-10, W-11, S-15, S-18, 4 Marks

Q. 1.2.13 Explain time sharing operating system and state its advantages and disadvantages. (Ref. Sec. 1.2.4)

W-11, W-17, 4 Marks

- Cards and tapes based systems allowed only sequential access to programs and data. So only one application system could be used at a time.
- In early days the computers were so user friendly that testing and development of programs was very easy for operators or programmers. But this arrangement resulted into more turnaround time.
- Due to time wastage of CPU i.e. CPU's idle time was more which was unreliable. In early days the cost of computer was more.
- In time sharing system the time is managed properly for multiple users in an interactive manner and at a reasonable cost.
- The main principles used by time sharing were CPU scheduling and multiprogramming to provide each user with a small portion of total time shared.
- Each user has a separate program in memory. Normally short execution programs were used to execute in such type of OS.
- In this type of system, I/O operations are also interactive.
- CPU does not wait for any operation of any job, it immediately gets switched to other job.
- Time sharing system allows many users to share computers simultaneously. All actions and commands are short in size in such type of system.

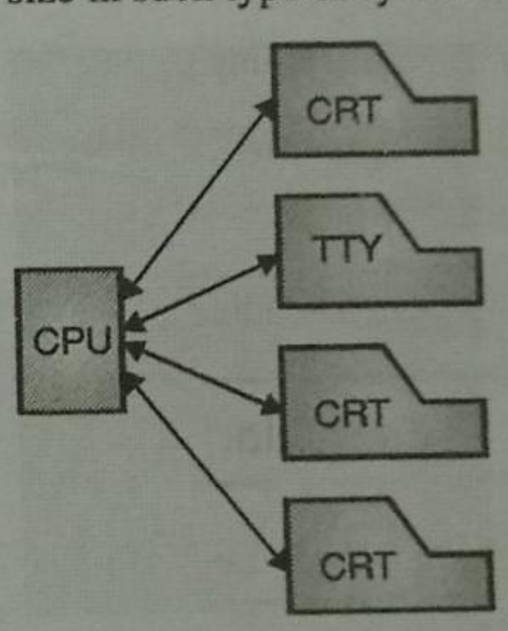


Fig. 1.2.7 : Time sharing OS

- As the user job switching time is very short users are in impression that they are working on their own non shared computer.
- Time shared systems are more difficult and expensive to build.
- In early days due to this till 1970 they were not too popular, but after this period the new operating system was evolved which was combination of the batch system and time shared system.

- The old example of time sharing operating system is "The Dartmouth Time Sharing Operating System".
- Many existing batch systems were modified into batch system and time sharing system. Examples of Time sharing operating system are Multics, Unix etc.

Advantages of time sharing operating system

1. Operating system works on time sharing basis with resource sharing.
2. Time management is properly done with multiple users in an interactive manner.
3. CPU scheduling and multiprogramming enhances the performance of system.
4. I/O operations are interactive.
5. CPU scheduling is fast.

Disadvantages of time sharing operating system

1. Time sharing systems are difficult and complicated.
2. They are expensive to build.
3. Time management involves complicated algorithms.
4. Effective time management may become difficult in case of more users.

1.2.5 Desktop Systems

- Desktop systems are generally a single user system. i.e. it is dedicated to a single user. Desktop systems are also called as personal computers.
- The desktop operating system is user friendly which provides maximum user convenience and responsiveness.
- CPU utilization is NOT very important issue in desktop system :
  1. Personal computers - A computer system dedicated to a single user which has I/O devices like keyboards, mouse, screens, printer, scanner etc.
  2. User convenience and responsiveness.
  3. Desktops can adopt technology developed for larger operating system
  4. Often individuals have sole use of computer and do not need advanced CPU utilization and protection features
  5. Desktops can run several different types of operating systems. e.g. : Windows, MAC. OS/2, UNIX, Linux etc.

1.3 Multiprocessor Systems (Parallel)

▶ **(MSBTE - W-09, S-11, S-14, W-14, W-15, S-16, S-17)**

Q. 1.3.1 Explain the multiprocessor systems concept. (Ref. Sec. 1.3) **W-09. S-11. S-14. W-14. S-16. 4 Marks**

Q. 1.3.2 Give two advantages of multiprocessor system. (Ref. Sec. 1.3) **S-11. W-15. S-16. 4 Marks**

Q. 1.3.3 Explain multiprocessor system and its two types. (Ref. Sec. 1.3) **S-17. 4 Marks**

- The today's normal machines are generally single processor system. i.e. they use just a single CPU for processing.
- The new concept or trend evolving in the market is multiprocessor systems.
- More than one processor can be placed in a single cabinet which share bus, resources clock and might be memory. These systems are called tightly coupled systems.
- The main advantage of parallel systems is to get maximum work done in shorter period. Also this is an economical solution for the users. i.e. Instead of many single CPU systems they can choose a multiprocessor machine system.
- The same copy of the data into main memory can be shared by multiple processors for processing them instead of making its copies.

- These systems provide reliability that if one processor gets failed, the system does not get halt or fail.
- One CPU can run a particular task and other can work as a backup CPU. They can make communication whenever necessary.

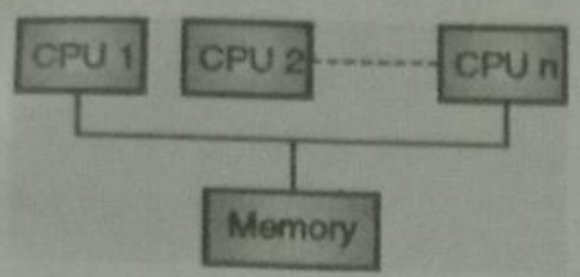


Fig. 1.3.1 : Multiprocessor system

- The most common multiprocessor system uses symmetric multiprocessing or asymmetric multiprocessing.
- In symmetric multiprocessing the same copy of OS is assigned to both the processors.
- In asymmetric multiprocessing the different operating systems are assigned to other processors.
- In short :
  1. More than 1 processor.
  2. Processors share the computer bus, clock, memory and peripheral devices.
  3. Referred to as tightly coupled systems.
  4. Increased throughput.
  5. Speed-up ratio with "n" processors is somewhat less than "n".

Common multiprocessing systems are :

☞ **Symmetric multiprocessing**

- In symmetric processing minimum two or multiple same type of processors are connected to each other by using high-bandwidth link.
- All such processors are managed and handled by single operating system. In this type of system every processor can equally share or access Input/output devices.

☞ **Asymmetric multiprocessing**

- Asymmetric multiprocessing system has got different system. Every processor in this system is given a specific task.
- There is a master processor in the system that controls the system and also delegates the job or work to slave processors. Also some processors may have predefined work or job to execute.

☞ **Advantages of Multiprocessor system**

1. More than one processor can be used - Multiprocessors need to be used for fast processing of complicated operations.
2. Processor shares other resources like bus, clock, memory, peripheral devices etc. - Though the processors are multiple but other devices need to be shared as per the requirement.
3. Tightly coupled system - These systems are not loosely coupled systems for better performance.
4. Maximum work can be done in shorter period - It is obvious when multiple CPU's are used, the processing time required will be very less.

1.4 Distributed Systems

▶ **(MSBTE - S-10, S-11, S-12, S-13, W-16, W-17)**

Q. 1.4.1 Explain distributed system in detail. Also give example. (Ref. Sec. 1.4)

S-10. S-11. S-12. S-13. W-16. W-17. 4 Marks

- Distributed systems are loosely coupled systems. They are exactly opposite to tightly coupled.
- The example of distributed operating system is Amoeba Operating System. In this system, processor does not share memory or devices even clock. Each processor has got its own memory.
- The processor communicates with each other through high-speed buses or telephone lines.

- The two processor in this system may be different i.e. they may be microprocessors, workstations, minicomputers etc.
- Their functions may be different. The main advantages of distributed systems are Resource Sharing. If numbers of sites or machines are connected to each other, user can share the resources of other terminal or site.
- **Speed of computation :** Due to the partition of the computation, the work gets done in short period at the highest rate.
- **Reliability :** If any machine or node gets fail, the other machines can work continuously.
- **Communication :** Communication among the different systems is possible. It may be electronic messaging or file transfer or exchange of programs or data.
- In short :
  1. Processors DO NOT share the memory or the clock.
  2. Each processor has its own memory.
  3. Loosely coupled systems give the impression that there is a single OS controlling the network.
  5. Distribute the computation among several physical locations.
  6. Loosely coupled system - In loosely coupled systems each and every processor has its own local memory. The communication among the processors can take place through different communication lines like telephone lines or high speed buses.

Advantages of distributed systems

- Resources sharing.
- Computation speed up
- Load sharing
- Reliability.
- Communications.

1.5 Clustered System

▶ **(MSBTE - S-12, W-13, S-15, W-15, S-18)**

Q. 1.5.1 What is meant by clustered system ? Explain in detail. (Ref. Sec. 1.5) **S-12, W-13, S-15, W-15, 4 Marks**

Q. 1.5.2 Define clustered systems? List four characteristics of clustered systems. (Ref. Sec. 1.5) **S-18, 4 Marks**

- Cluster architecture interconnects two or more computers using additional network and software technology to make a single virtual or logical server.
- From a technology point of view, cluster architectures provide the opportunity for system architects to link together powerful UNIX systems into even more powerful servers.
- And, since there are multiples of each component in a cluster, it is possible for the virtual server to continue to process information when a components fails or when system operators choose to maintain one component of the cluster.
- Cluster architectures are not new to information system planners. Since the early 1980s, several suppliers have offered cluster systems based on proprietary operating environments.
- Best known is Digital Equipment Corporation's (now Compaq) use of clustering to provide scalability and a uniform application environment for its VAX family of computers.
- The cluster architecture is one of different ways to use parallel processing. It enables use of several processors to work on one or more workloads.
- Other parallel approaches include Symmetric Multiprocessing (SMP), Non Uniform Memory Access (**NUMA**) and Massively Parallel Processing (MPP).

- These are different methods aimed at building more powerful computers with multiple microprocessors Clustering allows two or more systems to share storage. Its advantages are :
  1. Provides high reliability.
  2. Asymmetric clustering: One server runs the application or applications while other server is in a standby mode.
  3. Symmetric clustering: All N number of hosts are running the application.

☞ **Characteristics of cluster system**

1. Very high reliability.
2. Rare failures.
3. Data integrity Assurance.
4. Multiple cluster communication is available.
5. Data sharing and message passing is easy.

1.6 Real Time System

▶ **(MSBTE - S-09, W-09, W-10, W-11, W-12, S-13, S-14, W-14, W-15, S-16, W-16, S-17, W-17, W-18)**

Q. 1.6.1 Describe real time systems. Explain with examples. (Ref. Sec. 1.6)
S-09, W-09, W-10, W-11, W-12, S-13, S-14, W-14, S-16, W-16, W-17, W-18, 4 Marks

Q. 1.6.2 What is real time system ? (Ref. Sec. 1.6) W-15, 2 Marks

Q. 1.6.3 With neat diagram, explain real time system. List its any four application. (Ref. Sec. 1.6) S-17, 4 Marks

- The advent of time sharing provided 'good' response time to computer users. While the actual response times varied with load conditions in time system.
- Due to time slicing, response time was maintained. This satisfied some users for same application.
- However time sharing could not satisfied the requirements of some applications.
- Due to this, real time OS were developed to meet the response requirements of such applications.
- Real time applications: A Real time application is an application which requires a 'timely 'response from the computer system to prevent failures.
- The examples of application or real time operating system are VxWorks, Lynx, MTOS QNX, RTX, pSOS etc.

☞ **Applications of RTOS**

- Preemptive multitasking design paradigm
- Pre-tested and pre-integrated communications stacks and drivers.
- Application portability:
- System-level debug and analysis tools:
- More efficient use of CPU resources

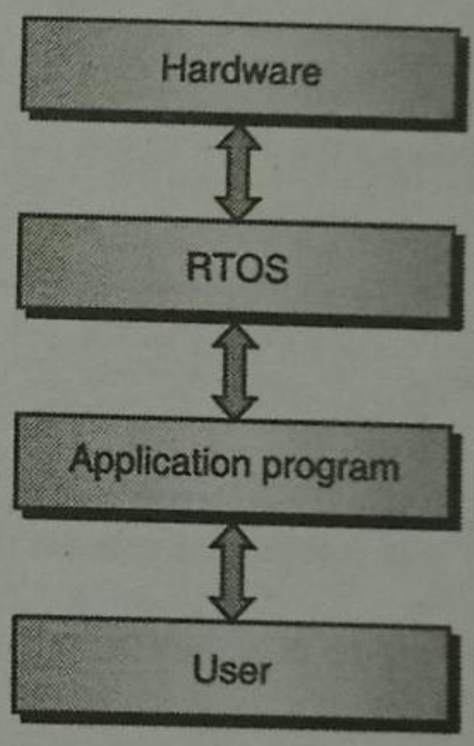


Fig. 1.6.1 : Real Time System

1.6.1 Worst Case Response Time

▶ **(MSBTE - W-15)**

Q. 1.6.4 Explain the types of real time system. (Ref. Sec. 1.6.1) W-15, 2 Marks

- The worst-case response time of an application is the largest value of the response time for which it can still function correctly. This 'timely' system is used where is time is a very important factor.

- The system is also known as online system. In this, system reports or result output in given time constraints.
- An instant output are generated in this type of system, some of the examples of these system are Railway reservation, Airline Reservation, Flight control systems, Banking etc.
- These applications are generally same for multiple users to use.
- They don't have multiple applications at a time for many users.
- Generally users are not allowed to develop or modify programs.
- They can only input the data and can get the results.
- These are dedicated systems for one specific application.
- A real time uses sensors which brings data to computers and analyses the data and sets the proper controls to modify the sensor input.

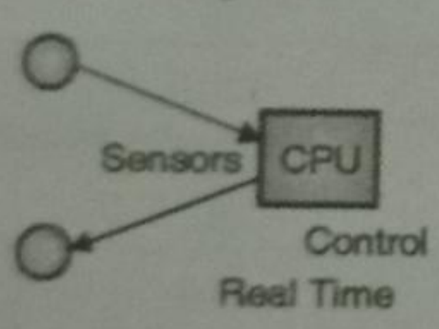


Fig. 1.6.1 : Real time OS

- System which controls scientific experimental, medical systems etc. are real time systems.
- In short in real time system :

1. Real time systems are also called as Multi-tasking systems.
2. Logical extension of Multiprogramming systems.
3. Multiple jobs are executed by CPU switching among them.
4. Switching is very frequent, so that users can interact with programs while they are running.
5. An interactive or Hands-on computer system.
6. Allows many users to share the computer resources simultaneously.
7. **Real time systems are** used when processing must be done within the fixed time constraints.
8. Considered to work correctly, only if it returns correct results with given time constraints.
9. **Examples :** Process control in industrial plants, robotics, air traffic control, telecommunications, military command and control systems, etc. Few other examples are Railway Reservation systems, ATM machines etc.
10. **Hard real-time systems :** Secondary storage limited or absent, data stored in short-term memory, or read-only memory (ROM).Not supported by general-purpose operating systems.
11. **Soft real-time systems :** Limited utility in industrial control or robotics useful in applications (multimedia, virtual reality) requiring advanced operating-system features.

1.6.2 Difference between Time Sharing System and Real-Time System

▶ (MSBTE - S-11)

Q. 1.6.5 Differentiate between time sharing system and real time system. (Ref. Sec. 1.6.2)

S-11. 4 Marks

Sr. No.	Real time system	Time sharing system
1.	In real time system, a job has to be completed within fixed deadline (time allowed).	In time sharing system, fixed time is given to each process and all the processes are arranged in a queue.
2.	If job is not completed within the given time then system may extend time for doing the operations.	If the job is not completed within the given time then it jumps to the next job leaving the previous job unfinished. After processing to each job, it again gives the same time for unfinished job.

Sr. No.	Real time system	Time sharing system
3.	The real time sharing operating system is used where different processes are executed for some time slot but in some process execution, time slot can be extended for process execution but context switching can also take place in same manner.	In time sharing operating system, equal time slots are provided to processor for execution of programs which also leads to the context switching, in which process shifts control from one to another. In case any process does not complete its working within time slot then extra time slot will not be given to it.
4.	For example - you are running a process with a time slot of 5 seconds and in case process does not complete in 5 second and requires 1 extra second then it will be executed in next execution cycle but time slot will not be extended	For example - in case you are executing a process which requires 7 seconds for its completion but process requires 8 seconds then 1 second time period will be extended and after that another time switching take place.

1.7 Desktop Systems

- Personal computers are called as desktop systems. In the period of their first decade, the CPUs were not that efficient to protect an operating system from user programs. In these days PC operating systems were unable to provide the features like multiuser and multitasking.
- The goals of operating systems initially were to maximize CPU and peripheral utilization. Afterwards systems started providing features for maximizing user convenience and responsiveness.
- Microsoft Windows and the Apple Macintosh were the operating systems used. The MS-DOS operating system was overtaken by Microsoft's multiple versions of Microsoft Windows, and IBM changed MS-DOS to the OS/2 multitasking system.
- Apple Macintosh is an operating systems used for advanced technology hardware. The other popular multi user operating systems were UNIX and LINUX.
- Microcomputers immediately adopted some of the technology developed for larger operating systems. For mainframe computers it is difficult to adopt the operating systems available in the market. For mainframes specific operating systems are required.

1.8 Features of Recent Operating Systems including MAC and Android

▶ **(MSBTE - S-12, W-12, W-13, W-16)**

Q. 1.8.1 Explain different characteristics of following operating systems :

(i) Windows XP (ii) Window 7

(Ref. Sec. 1.8) **S-12, W-13. 4 Marks**

Q. 1.8.2 List the features of Mobile Phone Operating System ? (Ref. Sec. 1.8) **W-12. 2 Marks**

Q. 1.8.3 List and explain major features of unix.

(Ref. Sec. 1.8) **W-16. 4 Marks**

- The most recent operating system includes various releases by Microsoft i.e. various versions of Windows operating systems. The other popular category is UNIX and Linux i.e open source software and OS like MAC etc.
- A new trend in the market includes mobile phone operating systems like Android, IOS, Symbian etc. MAC and OS/2 operating systems are also used for specific technology machines.
- Windows versions have got following features :
  - (a) User friendly
  - (b) Multitasking and multiprogramming
  - (c) Online compatibility
  - (d) High GUI Interfaces
  - (e) Plug and play technology

(f) High security

(g) Stand alone and Multiuser features.

(h) Easy updations in programs

- UNIX and Linux come under one category i.e. open source software. It possess following characteristics.

(a) Open source software

(b) Highly secured operating system.

(c) Strong kernel program.

(d) High security in multiuser environment.

(e) Cost effective

(f) Patches of code are added by experts from the different corners of the world.

(g) Highly portable.

(h) Efficient and effective file systems.

- Features of Mobile phone operating system like Android and IOS :

1. Compatible with desktop PCs.
2. Internet Access
3. Strong multimedia features.
4. Portable device management

- Features of Windows XP:

1. Strong Graphical user interface
2. Older version than Windows 7.
3. User friendly.
4. The most popular among Microsoft releases.

- Features of Windows 7:

1. One of the latest version of Microsoft Windows OS.
2. More user friendly than Windows XP
3. Transparent screens/GUIs.
4. Fast Navigation and very easy to use.
5. Has come with lot of new additions.

1.9 Command line based OS : DOS and UNIX

(a) DOS-Disk operating system

1. It uses command line interface to interact with the user.
2. Single user operating system.
3. Everything can be executed with commands only.
4. It has simple directory and file structure.
5. This does not provide GUI facility.
6. Mainly used on IBM machines.
7. Does not need huge amount of memory.
8. Multiprocessing and Multithreading is difficult.
9. Proprietary software

(b) UNIX

1. It is a multiuser and powerful operating system.
2. Interface is command line to interact with users.
3. Most of the UNIX commands are difficult to remember with its options.
4. Security and protection provided by OS is reasonable.
5. Not easily compatible with internet.
6. This is economical OS.
7. It does not provide GUI so not very user friendly.
8. Open source software

1.10 GUI based OS : Windows and Linux

(a) Windows

Windows versions have got following features :

(a) User friendly

(b) Multitasking and multiprogramming

(c) Online compatibility

(d) High GUI Interfaces

(e) Plug and play technology

(f) High security

(g) Stand alone and Multiuser features.

(h) Easy updations in programs

(b) Linux

Open source software

(b) Highly secured operating system.

(c) Strong kernel program.

(d) High security in multiuser environment.

(e) Cost effective

(f) Patches of code are added by experts from the different corners of the world.

(g) Highly portable.

(h) Efficient and effective file systems.

(i) Highly secured and protected OS.

1.11 MSBTE Questions and Answers

Winter 2008 - Total Marks 12

Q. 1 Explain Batch Monitoring functions.
(Ans. : Refer Section 1.2.1) **(4 Marks)**

Q. 2 What is multitasking operating system? State advantages of multitasking operating system.
(Ans. : Refer Section 1.2.3) **(4 Marks)**

Q. 3 What are the parts of computer system? Explain with the help of diagram.
(Ans. : Refer Section 1.1.1) **(4 Marks)**

Summer 2009 - Total Marks 12

Q. 4 Describe multiprogramming and multitasking.
(Ans. : Refer Sections 1.2.2 and 1.2.3) **(4 Marks)**

Q. 5 Describe real time systems. State any two examples of its applications.
(Ans. : Refer Section 1.6) **(4 Marks)**

Q. 6 Explain batch processing operating system.
(Ans. : Refer Section 1.2.1) **(4 Marks)**

Winter 2009 - Total Marks 12

Q. 7 Define operating system. State the different types of operating system.
(Ans. : Refer Section 1.2) **(4 Marks)**

Q. 8 Explain the multiprocessor systems concept.
(Ans. : Refer Section 1.3) **(4 Marks)**

Q. 9 Explain the real time operating system in brief.
(Ans. : Refer Section 1.6) **(4 Marks)**

Summer 2010 - Total Marks 16

Q. 10 List different types of O.S. Explain spooling.
(Ans. : Refer Sections 1.2 and 1.2.2(b)) **(4 Marks)**

Q. 11 Differentiate between multiprogramming and multitasking OS.
(Ans. : Refer Section 1.2.3.1) **(4 Marks)**

Q. 12 Explain distributed system in detail.
(Ans. : Refer Section 1.4) **(4 Marks)**

Q. 13 List any four functions of operating system.
(Ans. : Refer Section 1.1.2) **(4 Marks)**

Winter 2010 - Total Marks 16

Q. 14 Describe time sharing system along with example.
(Ans. : Refer Section 1.2.4) **(4 Marks)**

Q. 15 What is real time operating system? Elaborate with an example. *(Ans. : Refer Section 1.6)* **(4 Marks)**

Q. 16 Describe multiprogramming and multitasking.
(Ans. : Refer Sections 1.2.2 and 1.2.3) **(4 Marks)**

Q. 17 Describe evolution of operating system.
(Ans. : Refer Section 1.1.3) **(4 Marks)**

Summer 2011 - Total Marks 16

Q. 18 What is operating system? Explain the generations of computer system.
(Ans. : Refer Section 1.1 and 1.1.3) **(4 Marks)**

Q. 19 Differentiate between time sharing system and real time system. *(Ans. : Refer Section 1.6.2)* **(4 Marks)**

Q. 20 With examples explain what distributed system is.
(Ans. : Refer Section 1.4) **(4 Marks)**

Q. 21 What is multiprocessor system ? Give two advantages of it.
(Ans. : Refer Section 1.3) **(4 Marks)**

Winter 2011 - Total Marks 12

Q. 22 What is multitasking ? Explain with an example.
(Ans. : Refer Section 1.2.3) **(4 Marks)**

Q. 23 Describe real time systems. State any one example of its applications.
(Ans. : Refer Section 1.6) **(4 Marks)**

Q. 24 Explain time sharing operating system. State its advantages and disadvantages.
(Ans. : Refer Section 1.2.4) **(4 Marks)**

Summer 2012 - Total Marks 12

Q. 25 What is distributed system ?
(Ans. : Refer Section 1.4) **(2 Marks)**

Q. 26 What is meant by clustered system?
(Ans. : Refer Section 1.5) **(2 Marks)**

Q. 27 Explain in detail, various generations of operating system. *(Ans. : Refer Section 1.1.3)* **(4 Marks)**

Q. 28 Explain different characteristics of following operating systems :

(i) Windows XP (ii) Window 7
(Ans. : Refer Section 1.8) **(4 Marks)**

Winter 2012 - Total Marks 16

Q. 29 Differentiate between Multiprogramming and Multitasking OS ?
(Ans. : Refer Section 1.2.3.1) **(4 Marks)**

Q. 30 Explain the real time operating system in brief ?
(Ans. : Refer Section 1.6) **(4 Marks)**

Q. 31 What is an Operating System? List the features of Mobile Phone Operating System ?
(Ans. : Refer Sections 1.1 and 1.8) **(4 Marks)**

Q. 32 What is clustered systems? Write in detail?
(Ans. : Refer Section 1.5) **(4 Marks)**

Summer 2013 - Total Marks 18

Q. 33 Describe functions of an Operating system.
(Ans. : Refer Section 1.1.2) **(4 Marks)**

Q. 34 Describe evolution of operating system.
(Ans. : Refer Section 1.1.3) **(6 Marks)**

Q. 35 What is real time operating system? Explain with example. *(Ans. : Refer Section 1.6)* **(4 Marks)**

Q. 36 Explain Batch processing operating system.
(Ans. : Refer Section 1.2.1) **(4 Marks)**

Winter 2013 - Total Marks 16

Q. 37 What is distributed system?
(Ans. : Refer Section 1.4) **(2 Marks)**

Q. 38 What is meant by clustered system?
(Ans. : Refer Section 1.5) **(2 Marks)**

Q. 39 Explain in detail, various generation of operating System. *(Ans. : Refer Section 1.1.3)* **(8 Marks)**

Q. 40 Explain different characteristics of following operating systems :
(i) Windows XP (ii) Window 7
(Ans. : Refer Section 1.8) **(4 Marks)**

Summer 2014 - Total Marks 20

Q. 41 Describe the multiprocessor systems concepts.
(Ans. : Refer Section 1.3) **(4 Marks)**

Q. 42 What is real time operating system?
(Ans. : Refer Section 1.6) **(4 Marks)**

Q. 43 Differentiate between multiprogramming and multitasking O.S.
(Ans. : Refer Section 1.2.3.1) **(4 Marks)**

Q. 44 Enlist system components? Describe any one in detail. *(Ans. : Refer Section 1.1.1)* **(4 Marks)**

Q. 45 What are function of OS ?
(Ans. : Refer Section 1.1.2) **(4 Marks)**

Winter 2014 - Total Marks 22

Q. 46 State generations of operating system. Describe any one generation.
(Ans. : Refer Section 1.1.3) **(4 Marks)**

Q. 47 Describe following system :
1) Multiprocessor system
(Ans. : Refer Section 1.3) **(3 Marks)**
2) Batch operating system
(Ans. : Refer Section 1.2.1) **(3 Marks)**

Q. 48 Explain Real time OS with the help of diagram. List its type. *(Ans. : Refer Section 1.6)* **(4 Marks)**

Q. 49 Describe functions of OS in detail.
(Ans. : Refer Section 1.1.2) **(4 Marks)**

Q. 50 List components of OS. Explain any one in detail.
(Ans. : Refer Section 1.1.1) **(4 Marks)**

Summer 2015 Total Marks 20

Q. 51 Explain time sharing operating system.
(Ans. : Refer section 1.2.4) **(4 Marks)**

Q. 52 List system component. Explain any two.
(Ans. : Refer section 1.1.1) **(4 Marks)**

Q. 53 Explain batch operating system.
(Ans. : Refer section 1.2.1) **(4 Marks)**

Q. 54 Explain the working of clustered operating system.
(Ans. : Refer section 1.5) **(4 Marks)**

Q. 55 Explain generation of operating system.
(Ans. : Refer section 1.1.3) **(4 Marks)**

Winter 2015 - Total Marks 16

Q. 56 List different types of operating systems. Explain advantages of multiprocessor system (any two).
(Ans. : Refer sections 1.2 and 1.3) **(4 Marks)**

Q. 57 Define multiprogramming system with diagram.
(Ans. : Refer section 1.2.2) **(4 Marks)**

Q. 58 What is clustered system ? Explain it.
(Ans. : Refer section 1.5) **(4 Marks)**

Q. 59 What is real time system ? Explain its types.
(Ans. : Refer sections 1.6 and 1.6.1) **(4 Marks)**

Summer 2016 - Total Marks 16

Q. 60 List any four functions of operating system.
(Ans. : Refer section 1.1.2) **(4 Marks)**

Q. 61 Describe real time operating system in brief.
(Ans. : Refer sections 1.6 and 1.6.1) **(4 Marks)**

Q. 62 Describe evolution of operating system.
(Ans. : Refer section 1.1.3) **(4 Marks)**

Q. 63 What is multiprocessor system? Give two advantages of it.
(Ans. : Refer section 1.3) **(4 Marks)**

Winter 2016 - Total Marks 22

Q. 64 Define real-time operating system. Explain with the help of example.
(Ans. : Refer section 1.6) **(4 Marks)**

Q. 65 List and explain components of operating system. *(Ans. : Refer section 1.1.1)* **(6 Marks)**

Q. 66 List and explain major features of UNIX.
(Ans. : Refer section 1.8) **(4 Marks)**

Q. 67 Describe evolution of operating system.
(Ans. : Refer section 1.1.3) **(4 Marks)**

Q. 68 Describe Distributed Operating System.
(Ans. : Refer section 1.4) **(4 Marks)**

Summer 2017 - Total Marks16

Q. 69 With neat diagram, explain real time system. List its any four application.
(Ans. : Refer section 1.6) **(4 Marks)**

Q. 70 Describe first generation of operating system with its advantages and disadvantages.
(Ans. : Refer section 1.1.3(2)) **(4 Marks)**

Q. 71 Explain multiprocessor system and its two types.
(Ans. : Refer section 1.3) **(4 Marks)**

Q. 72 Explain multi-programmed O.S. with suitable diagram.
(Ans. : Refer section 1.2.2) **(4 Marks)**

Winter 2017 - Total Marks 16

Q. 73 Describe multiprogramming and multitasking.
(Ans. : Refer sections 1.2.2 and 1.2.3) **(4 Marks)**

Q. 74 Explain time sharing operating systems and state its advantages and disadvantages.
(Ans. : Refer section 1.2.4) **(4 Marks)**

Q. 75 Explain distributed system in detail.
(Ans. : Refer section 1.4) **(4 Marks)**

CHAPTER 2

UNIT II

Services & Components of Operating System

Syllabus

2.1 Different Services of Operating System.

2.2 System Calls - Concept, Types of system calls

2.3 OS Components - Process Management, Main memory Management, File Management, I/O System management, Secondary storage management.

2.4 Use of operating system tools, user management, security policy, device management, performance monitor, task scheduler.

2.1 Different Services of Operating System

▶ **(MSBTE - S-09, S-11, S-12, W-12, W-13, S-15, W-15, W-16, S-17, W-17, W-18)**

Q. 2.1.1 Write and explain any four services provided by operating system. (Ref. Sec. 2.1) **S-09, S-11, S-12, W-12, W-13, W-17, 4 Marks**

Q. 2.1.2 Describe any four services provided by an operating system. (Ref. Sec. 2.1) **S-15, W-16, S-17, 4 Marks**

Q. 2.1.3 Explain any six services provided by operating system. Draw diagram showing services. (Ref. Sec. 2.1) **W-15, 6 Marks**

Q. 2.1.4 List any four services provided by operating system. (Ref. Sec. 2.1) **W-18, 2 Marks**

- Operating system sets an environment which is suitable to execute the programs. Operating system provides many services for smooth functioning of overall system.
- Following are the major services provided by operating systems for convenience of the users.

1. User Interface
2. Program Execution
3. Input/output Operations
4. File System Manipulation
5. Communications
6. Error Detection
7. Resource Allocation
8. Accounting
9. Protection and security

2.1.1 User Interface

- User interface is a medium through which user actually interacts with the computer system via operating system. Almost all operating systems have a user interface (UI).
- User interfaces are categorized into three types as follows :

(a) **CLI :** Command line interface, provides an environment where user can write their commands in text format on command prompts.

(b) **Batch interface :** Batch interface allows a command environment which gets execute one by one or sequentially. User creates a batch file which contains multiple executable commands in a sequence. This batch file is executed to execute the set of commands included in it.

(c) **GUI :** Graphical user interface. Most of the operating systems provide graphical interfaces which have tools like pointing devices which directs the input/ output operations with the help of graphical control like menu etc.

2.1.2 Program Execution

- Program execution allows user written programs to get execute or run. Program execution is the most important aspect of the computer system. One of the core functions of operating system is to support program execution. OS takes care of assigning work to the CPU (CPU job scheduling) for the purpose of program execution.
- Every program that runs on computer system needs memory. This memory allocation is done by operating system. Also multitasking, multiprogramming etc features are provided by OS. After executing the program memory de-allocation of memory is done by OS.

- In short, user's program cannot run independently. They need a platform of operating system to run.

2.1.3 Input/output Operations

- Each program requires an input and produces output. This involves the use of I/O device or files. The operating system hides the details of underlying hardware of an I/O from the user. User can run their programs conveniently because they are not aware about the I/O system of OS.
- For better efficiency and protection, the users are not allowed to directly interact with I/O devices. So operating system here becomes the bridge between I/O devices and user.
- OS makes I/O devices available to user in a user friendly manner whenever required by users.

2.1.4 File System Manipulation

▶ (MSBTE - S-18)

Q. 2.1.5 Explain following service of operating system : File system manipulation. (Ref. Sec. 2.1.4) **S-18, 2 Marks**

- The output of a program may need to be written into new files or can be given as input to other files. The operating systems provide this service. The user does not have to worry about secondary storage management.
- User gives a command for reading or writing to a file and can see his task accomplished. Thus operating systems make it easier for user programs to accomplish their task.
- Operating system even handles the issues related to the different permissions like read, write, execute, and even access deny depending on the ownership of the file.
- It also involves creating, deleting or even searching of the files by its name. It is done by operating system.

2.1.5 Communications

- The communication here refers to the communication that takes place among the different processes. It might happen that processes need to communicate with each other for some purpose. Also processes can be present on the same machine (computer) or another machine.
- Operating system supports message passing facility to the process to ensure that processes communicate with each other and process executions get done without any hassles. When processes are present on different computers, communication can be done through the network. User written programs can help in transmitting messages among the different processes.

2.1.6 Error Detection

- An error is one part of the system that may cause malfunctioning of the complete system. To avoid such a situation the operating system constantly monitors the system for detecting the errors.
- OS ensures that the user should not be worried for errors propagating to various part of the system and causing malfunctioning.
- This service cannot be controlled by user programs because it involves monitoring and in cases altering area of memory or de-allocation of memory for a faulty process. It can be even relinquishing the CPU of a process that goes into an infinite loop.
- These tasks are too critical to be handed over to the user programs. A user program may interfere with the normal operation of the operating systems if they are allowed to control errors.

2.1.7 Resource Allocation

▶ (MSBTE - S-18)

Q. 2.1.6 Explain following service of operating systems : Resource Allocation. (Ref. Sec. 2.1.7) **S-18, 2 Marks**

- When multiple users or multiple jobs running concurrently, resources must be allocated to each of them.
- Many types of resources such as CPU cycles, main memory, and file storage etc may have special allocation code, others (such as I/O devices) may have general request and release code.

2.1.8 Accounting

- To keep track of which users use how much and what kinds of computer resources. It also helps in finding out the usage statistics.
- It is even useful for the researchers to find out the usage and the requirements to reconfigure the system.

2.1.9 Protection and Security

- The owners of information stored in a multi-user or networked computer system may want to control use of that information. i.e. access to data and other related privileges like reading, writing , executing etc.
- Even the concurrent processes should not interfere with each other which may cause damage to the information.
- **Protection :** This involves ensuring that all access to system resources is controlled.
- **Security** of the system from unwanted users requires user authentication. The defending of external I/O devices from invalid access is important. It is important to protect the system and keep it secured. The precautions must be instituted throughout the system.

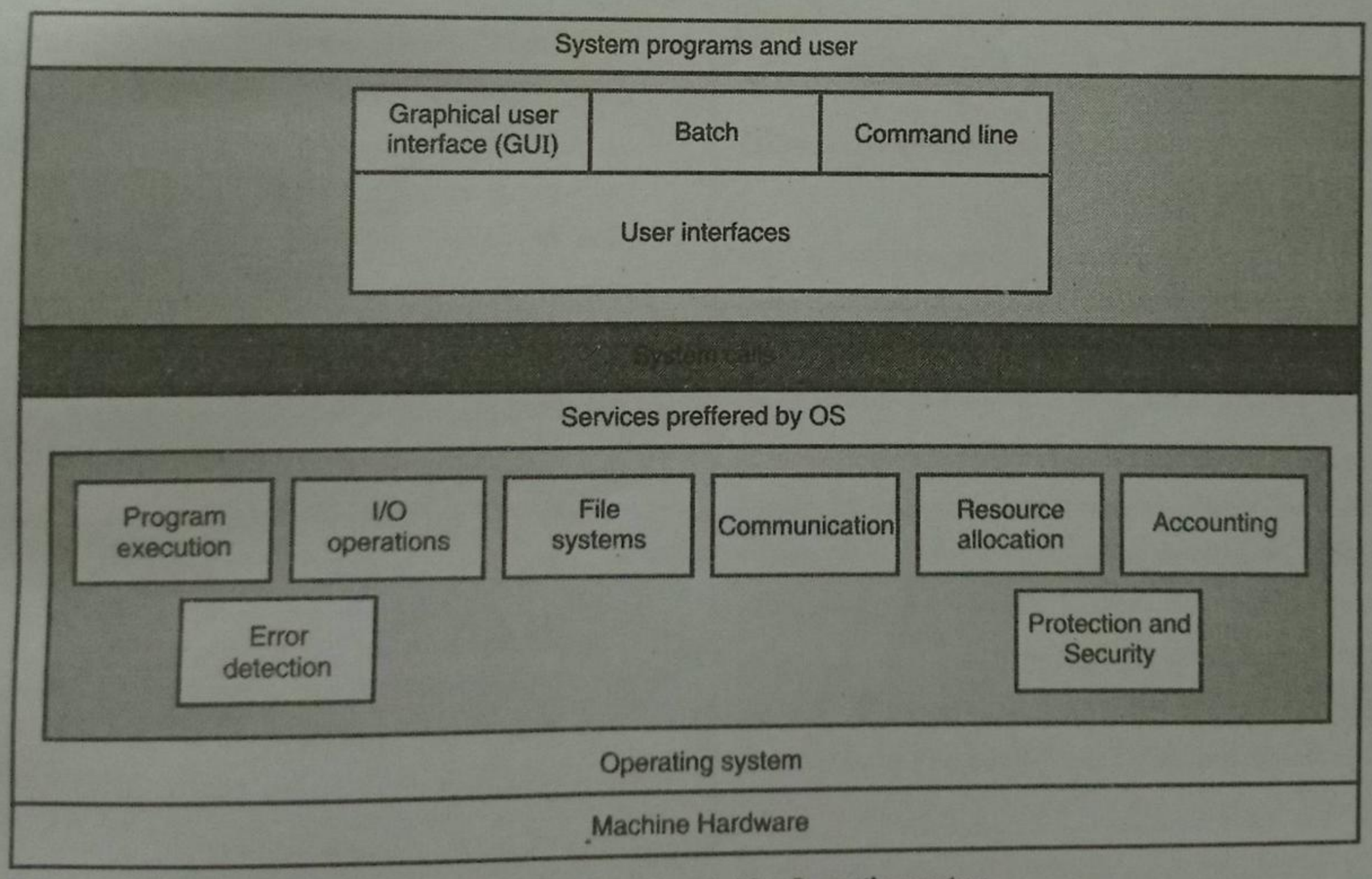


Fig. 2.1.1 : Services provided by Operating system

2.2 System Calls

▶ **(MSBTE - W-08, S-09, W-09, W-10, S-11, W-11, S-12, W-12, W-13, S-14, W-14, S-15, W-15, S-16, W-16, S-17, W-17, S-18, W-18)**

Q. 2.2.1 What is system call ? Explain with example. (Ref. Sec. 2.2)

W-08, S-09, W-09, W-10, S-11, W-11, S-12, W-12, W-13, S-14, 4 Marks

Q. 2.2.2 What is System call ? (Ref. Sec. 2.2)

W-14, S-15, W-15, S-16, S-17, W-17, W-18, 2 Marks

Q. 2.2.3 Describe the purpose of system calls ? State two system calls with its functions. (Ref. Sec. 2.2) W-16, 2 Marks

Q. 2.2.4 Explain the system call-"Information Maintenance" (Ref. Sec. 2.2) S-18, 2 Marks

Q. 2.2.5 Explain open( ) system call and close( ) system call. (Ref. Sec. 2.2) W-15, S-17, 2 Marks

- System call is a communication made by an operating system with the processes.
- System calls provide an interface between the process and the operating system. System calls allow user-level processes to request some services from the operating system which process itself is not allowed to do.
- In handling the trap, the operating system will enter in the kernel mode, where it has access to privileged instructions, and can perform the desired service on the behalf of user-level process. Kernel is a main part or component of an operating system.
- It is because of the critical nature of operations that the operating system itself does them every time they are needed.
- For example, for I/O a process involves a system call telling the operating system to read or write particular area and this request is satisfied by the operating system.
- System programs provide basic functioning to users so that they do not need to write their own environment for program development E.g. editors, compilers etc. and program execution e.g. shells. In short they are bundles of useful system calls.
- Fig. 2.2.1 for system call describes an example where one file is copied on other file. Consider two files one is source and other is destination.
- If data is to be copied from one file to other then the file names should be known. Either it should be initialized in the program or input should be taken interactively from the user when prompted.
- When file names are inputted, system checks whether file exists or do it require access permission, if yes then contents are read from the respective files.
- Just observe the actions that are taken for copying the file, where every time system call is to be made by operating system to do each an every action.
- After confirming details about a file either error messages are prompted on the file or further reading and writing actions are made. In this example system calls are needed for taking input from one file and putting into another file.
- In short system calls are :

1. Programming interface to the services provided by the OS. (System calls are mediator between OS and process).
2. Typically written in a high-level language. E.g. C or C++.
3. Mostly accessed by programs via high-level **Application Program Interface (API)** rather than using direct system call.
4. Three most common APIs are Win32 API for Windows, POSIX API for POSIX-based systems. (It includes virtually all versions of UNIX, Linux, and Mac OS X), and Java API for the Java virtual machine (JVM)

☞ **Open and Close system call**

Open system call ()

Open() system call is basically for opening a file. Open() system call is used by the programs for initializing an access to the file. When this call is used, it returns file descriptor. This descriptor is positive integer value. It indicates the position of file from the table of open files.

Close system call ()

Close () system call is used to close the file that is recently opened. It basically removes all resources allocated to the file that is closed. It clears the memory including buffers. It even updates the metadata related to closed file.

Example of System call

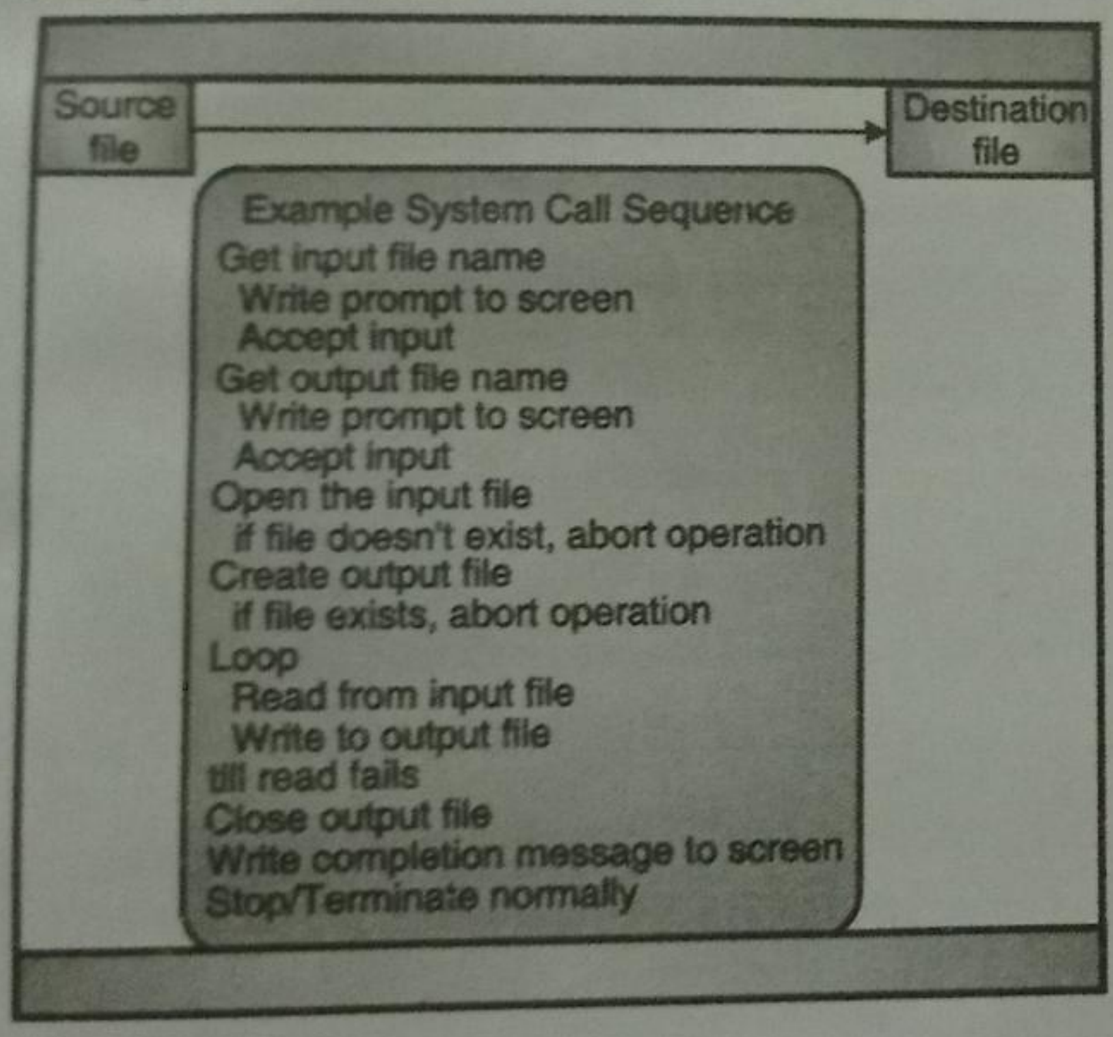


Fig. 2.2.1 : Example of system call

2.2.1 System Call Implementation

▶ **(MSBTE - S-12, W-13, S-16)**

Q. 2.2.6 Explain system call implementation with its working. (Ref. Sec. 2.2.1) **S-12. W-13. 2 Marks**

Q. 2.2.7 Explain how parameter passing is done while implementing system calls.
(Ref. Secs. 2.2.1 and 2.2.2) **S-16. 4 Marks**

- Generally a number is associated with each system call. It is used to number the system calls. System-call interface maintains a table indexed according to these numbers.
- The system call interface invokes intended system call in OS kernel and returns status of the system call and return values.
- The caller needs to know nothing about how the system call is implemented. Just needs to use API (application interface) and understand what OS will do as a result of call.
- Most of the details of OS interface are hidden from programmer by API. It is managed by run-time support library. Compiler includes set of functions built into libraries.

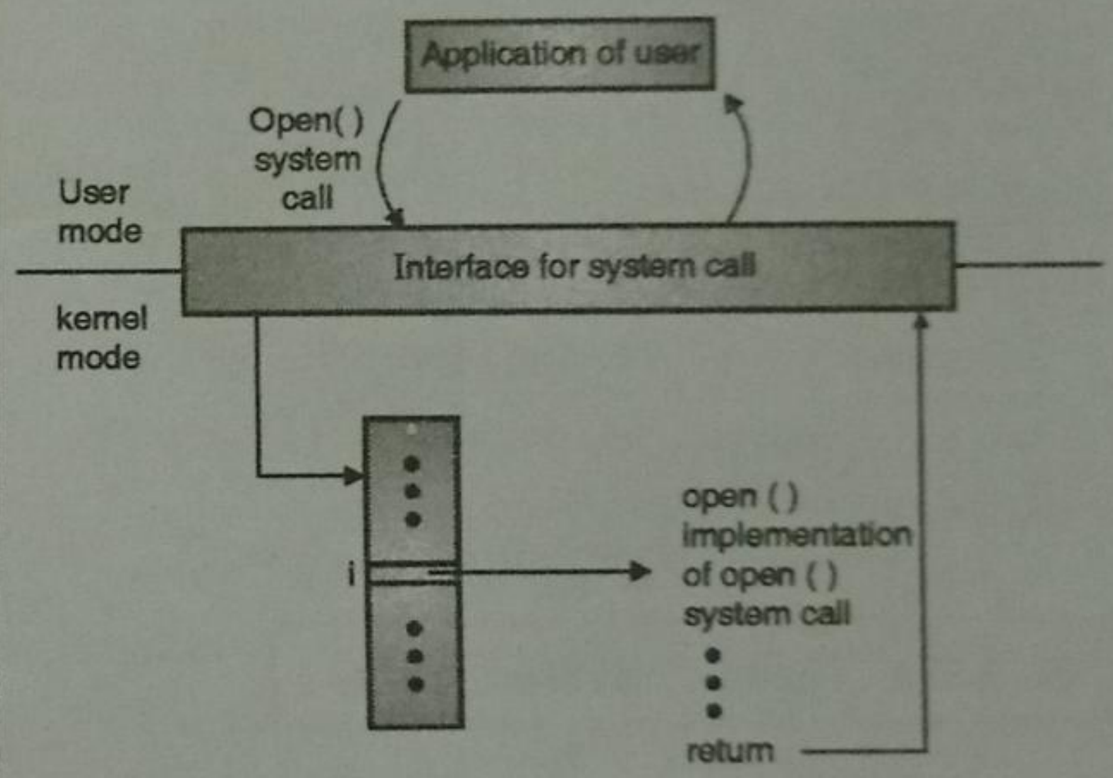


Fig. 2.2.2 : System call implementation

2.2.2 System Call Parameter Passing

▶ **(MSBTE - S-16)**

Q. 2.2.8 Explain how parameter passing is done while implementing system calls.
(Ref. Secs. 2.2.1 and 2.2.2) **S-16. 4 Marks**

- Frequently more information is required than simply identifying of desired system call. Exact type and amount of information vary according to OS and call.

- There are two general methods used to pass parameters to the OS.

1. **Simplest :** Pass the parameters in registers. In some cases, may be more parameters than registers. In such a case parameters are stored in a block, or table, in memory. Then the address of block passed as a parameter in a register. This approach is used by Linux and Solaris.
2. Parameters are placed, or pushed, onto the stack by the program and popped off the stack by the operating system. Block and stack methods do not limit the number or length of parameters being passed.

☞ **Parameter passing via table**

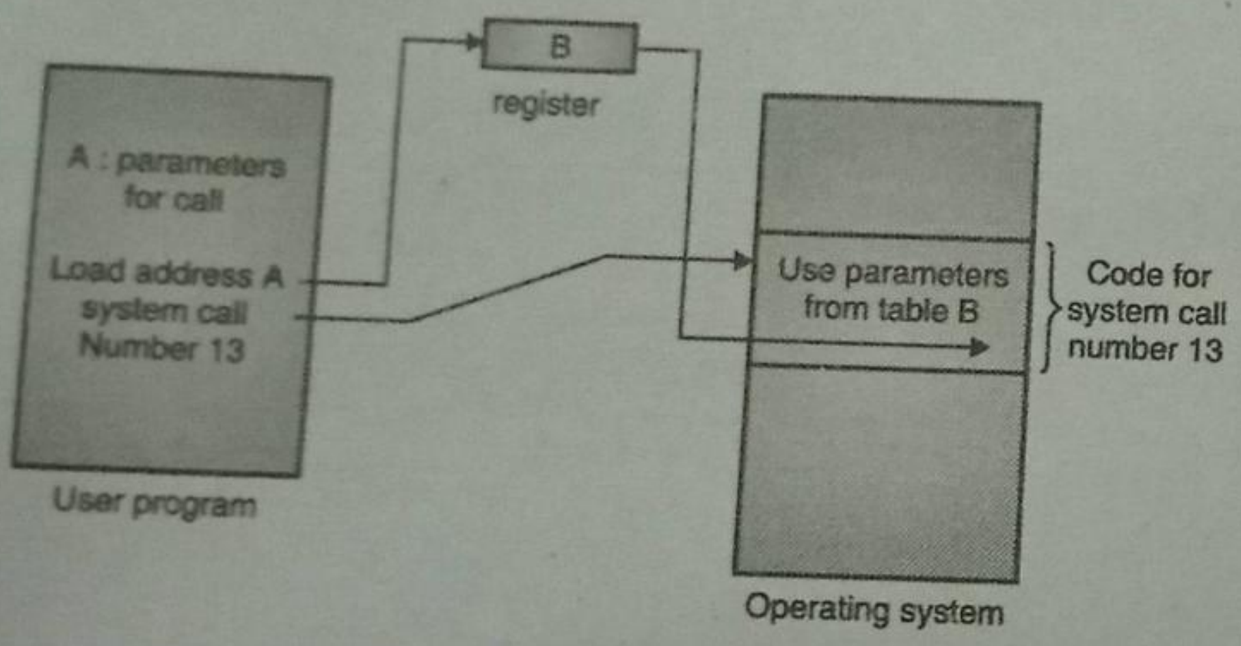


Fig. 2.2.3 : Parameter passing via table

2.2.3 Types of System Calls

▶ **(MSBTE - W-08, S-09, W-09, S-10, W-10, S-11, W-11, W-12, S-14, W-14, S-15, S-16, S-17, W-17, S-18)**

Q. 2.2.9 Explain types of system call. (Ref. Sec. 2.2.3) **W-08, S-09, W-09, S-10, W-10, S-11, W-11**

Q. 2.2.10 Explain use of any two categories of system call. (Ref. Sec. 2.2.3) **W-12, S-14, W-14, 2 Marks**

Q. 2.2.11 Enlist any four system call. (Ref. Sec. 2.2.3) **S-15, S-17, W-17, S-18, 2 Marks**

Q. 2.2.12 List types of system call with one example of system call. (Ref. Sec. 2.2.3) **S-16, 2 Marks**

- There are different system calls made by an operating system. Different types are categorized as per their usage.

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

- Most of the system calls support, or are supported by concepts and functions.

1. Process control (Mainly used for process)

- end , abort
- load, execute
- create process, terminate process
- get process attributes, set process attributes
- wait for time
- wait event, signal event
- allocate and free memory

2. File management (Used for manipulation of files)

▶ **(MSBTE - W-10, W-18)**

Q. 2.2.13 Enlist the system calls for file management. (Ref. Sec. 2.2.3(2)) **W-10, 2 Marks**

Q. 2.2.14 Explain any four file related system calls. (Ref. Sec. 2.2.3(2)) **W-18, 2 Marks**

- create file, delete file
- open, close
- read, write, reposition
- get file attributes, set file attributes.

3. Device management (Used for managing devices)

▶ **(MSBTE - W-15)**

Q. 2.2.15 List any four system calls for device management. (Ref. Sec. 2.2.3(3))

W-15, 2 Marks

- Request device, release device
- Read, write, reposition
- get device attributes, set device attributes
- Logically attach or detach devices.

4. Information maintenance (To do information maintenance)

- get time or date, set time or date
- get system data, set system data
- get process, file, or device attributes
- set process, file, or device attribute

5. Communications (For communicating among calls)

▶ **(MSBTE - W-15)**

Q. 2.2.16 List any four system calls for communication. (Ref. Sec. 2.2.3(5)) **W-15, 2 Marks.**

create, delete communication connection

- send, receive messages
- transfer status information
- Attach or detach remote devices.

6. Memory management related system calls (In UNIX)

▶ **(MSBTE - S-12, W-13)**

Q. 2.2.17 Enlist any four system calls related memory management. (Ref. Sec. 2.2.3(6))

S-12, W-13, 4 Marks

- shmget
- shmat
- shmdt
- shmctl

2.3 Operating System Structure

▶ **(MSBTE - S-10, S-14)**

Q. 2.3.1 Draw neat diagram and explain operating system structure. (Ref. Sec. 2.3)

S-10, S-14, 4 Marks

- Important aspect of an operating system is multiprogramming. Generally a single user cannot keep CPU as well input/output devices busy. This ultimately reduces the CPU as well as I/O device utilization.
- Multiprogramming is a technique used to minimize the CPU utilization where the work is divided into jobs and this way CPU and other devices are kept busy by assigning jobs to them one by one.
- The jobs are kept in the memory and then operating system executes them one by one. Jobs may have to wait for input/output devices and CPU.
- Then according to the availability of devices job waits. During this phase an operating system switches to other job. e.g. if operating system executes job 1 and if I/O devices are free, an operating system executes job 2 which may use I/O devices.
- Till then job 1 makes CPU free and CPU is assigned to job 2. This way operating system manages the resources as well as jobs by switching technique.
- There is another concept called as time sharing which is logical extension of multiprogramming. Time sharing is also called as multitasking.
- In time sharing operating system CPU executes multiple jobs. But the switching time among the different jobs is too less that user do not understand that CPU executes only one job at a time. In short switching speed is very fast in time sharing OS.

- To provide quick access to the users the response time should be reasonable or very short. This is an important feature which operating system should provide to its users. Even the computer system should be more active because it gives the feel that CPU executes many jobs at a time.

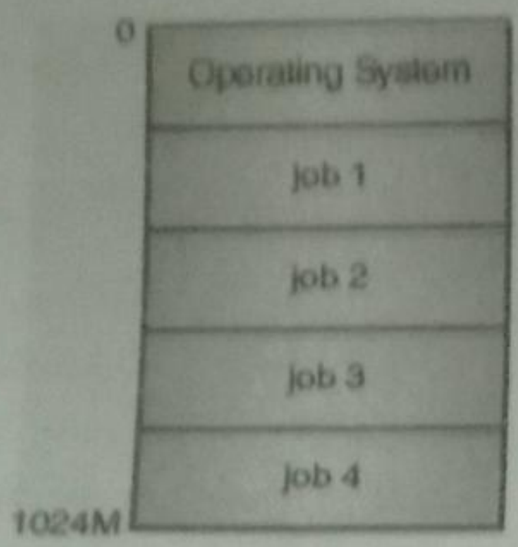


Fig. 2.3.1 : Memory layouts for multi- programmed system

- Time sharing and multiprogramming needs to make the job queue in the main memory. Since the main memory is very less to keep all the jobs, the job pools are created on the disk.
- The job pool is consist of all processes residing on disk awaiting allocation of main memory and if there is not sufficient space for them then main memory chooses the job among them. Then OS does job scheduling using different techniques and after that CPU scheduling is done.
- The time sharing operating system must ensure reasonable response time which is achieved using a technique called as swapping.
- In swapping, processes are swapped in and out of memory to the disk. There is another technique called as "Virtual memory" which allows the execution of process that is not completely in memory.
- The benefit of virtual memory is that this technique allows user to run program whose size is more than physical memory.
- Time sharing system even provides a file system. File system is collection file related utilities present on the disk. File system is present on the disks and needs disk management. Time sharing OS should ensure :
  1. Mechanism for protecting resources from inappropriate use.
  2. Mechanism for job synchronization and communication.
  3. Check that job should not be stuck due to deadlock.
- In short :
  (a) Multiprogramming is mainly needed for efficiency.
  (b) Single user cannot keep CPU and I/O devices busy at all times.
  (c) Multiprogramming organizes jobs (code and data) in such a way that CPU always has at least one job to execute.
  (d) A subset of total jobs in system is kept in memory.
  (e) One job is selected and run via **job scheduling.**
  (f) When it has to wait (for I/O for example), OS switches to another job.
- **Timesharing (multitasking)** is logical extension of multiprogramming in which CPU switches jobs so frequently that users can interact with each job while it is running, It creates interactive computing environment.
  (a) Response time should be < 1 second.
  (b) Each user has at least one program executing in memory.
  (c) If several jobs are ready to run at the same time the CPU scheduling is done.
  (d) If processes don't fit in memory, swapping moves them in and out to run.
  (e) **Virtual memory** allows execution of processes not completely in memory but using the space occupied on the secondary memory.

2.3.1 Types of Operating System Structure

There are different types of operating system structure. Some of them are subsequent :

1. Simple structure
2. Unix Layered
3. MS-DOS Layer Structure
4. Microkernel
5. Monolithic system

2.3.1.1 Simple Structure

- Some operating systems are commercial systems which do not have well-defined structures or architecture. In the beginning these operating systems were small and simple. Even their scope was not known to us.
- But then these operating systems grew more than their scope. The best example of this type of OS is MS-DOS.
- MS-DOS was originally designed and implemented by a few people who were unknown about the popularity of this product.
- They never thought that this product will capture a very big market in the future. The purposes behind writing were to provide the most functionality and use the least space. This is the reason that's why this OS is not divided into different modules.
- MS-DOS has not created interfaces, functional or access levels. This has reduced the complications of OS structure designing. This could be explained with the example. Though any application runs on MS-DOS, it used basic I/O functions to execute its input and output operations.
- This is a type of freedom which makes MS-DOS vulnerable to errant or malicious programs. Due to this entire system crashes when user programs fail.
- Reason to this was restricted by the hardware in those times. In those days Intel 8088 was unable to provide dual mode and no hardware protection. Because of this the designer of MS-DOS had no choice but to leave the base hardware accessible.
- One more example of limited structuring is the original UNIX operating system. UNIX was also limited by hardware functionality. It has mainly two separable parts called the **kernel** and the **system programs**.

2.3.1.2 UNIX Layered Structure

▶ (MSBTE - S-16)

Q. 2.3.2 Draw and explain structure of Unix operating system. (Ref. Sec. 2.3.1.2) **S-16, 4 Marks**

- Kernel in UNIX is like heart in human body. The kernel is further divided into a series of interfaces and drive drivers. These interfaces and device drivers were added on need basis due to which UNIX has evolved over the years. The traditional view of UNIX operating system in different layers is shown in Fig. 2.3.2.
- There is a specific position of kernel. Everything which is below the system call interface and above the physical hardware is the kernel.

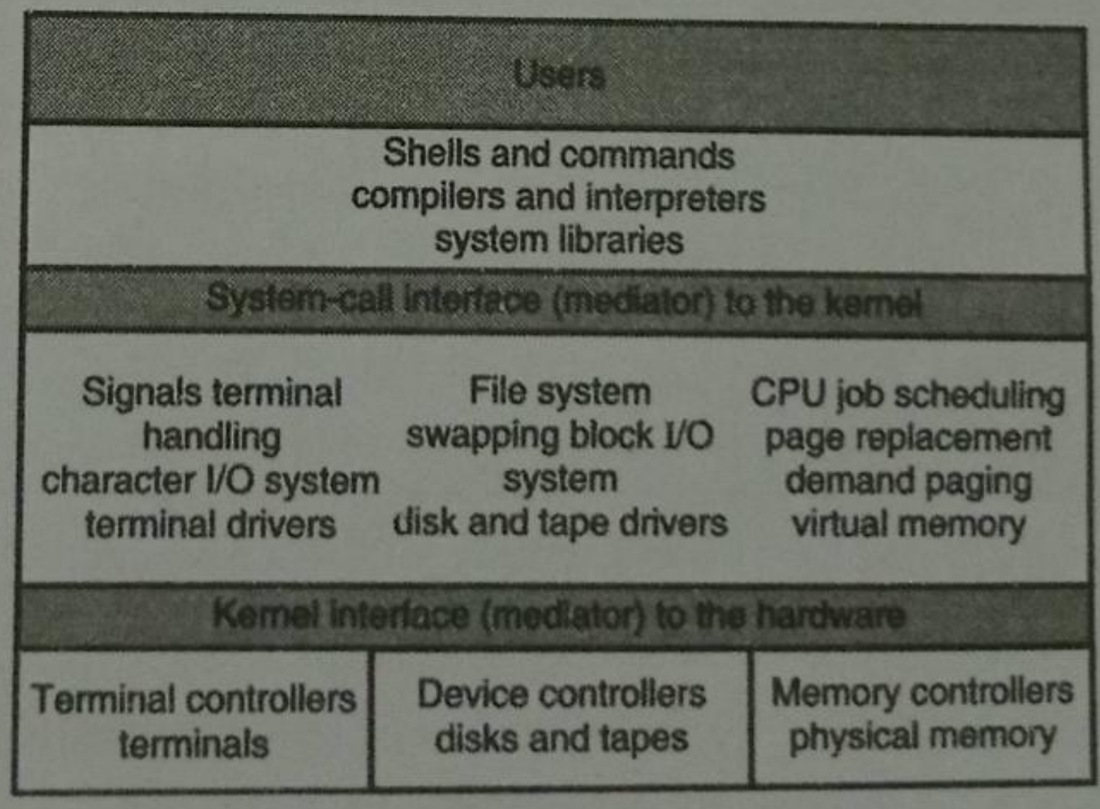


Fig. 2.3.2 : UNIX system structure

- The kernel provides the main functions which are performed by operating system itself. Some of the functions are file system, CPU scheduling, memory management, and other operating system functions through system calls.
- In short :
  1. MS-DOS - written to provide the most functionality in the least space.
  2. It is not divided into modules.
  3. Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated.

2.3.1.3 Comparison between Unix and Linux operating system

▶ (MSBTE - S-16)

Q. 2.3.3 Compare Unix and Linux operating system w.r.t.
(1) User interface
(2) Name of provider
(3) Processing speed
(4) Security
(Ref. Sec. 2.3.1.3) **S-16, 4 Marks**

☞ **Difference between Unix and Linux operating system**

Criteria	LINUX	UNIX
User interface	Linux provides two GUIs, called KDE and Gnome. There are many similar products available in the market.	In the beginning unix was command line, later on GUI was created and was called as Common Desktop Environment.
Name of Provider	Redhat, Ubantu, Fedora etc.	Osx, Solaris, etc.
Processing speed	Processing speed is low because of GUI.	Processing speed is high for it is command line.
Security	There are approximately 60 to 100 viruses. They are no more spreading. Linux has had about 60-100 viruses listed till date. None of them actively is spreading nowadays.	Unix viruses are around 85 to 120. A rough estimate of UNIX viruses is between 85 -120 viruses reported till date.

2.3.1.4 MS-DOS Layer Structure and Layered Approach

▶ (MSBTE - S-09, S-10, S-12, S-13, W-13, W-14,W-16, W-17)

Q. 2.3.4 Describe layered approach to system design. Also write its advantages and disadvantages. (Ref. Sec. 2.3.1.4) **S-09, 4 Marks**

Q. 2.3.5 Explain : Layered OS structure.
(Ref. Sec. 2.3.1.4)
S-10, S-12, S-13, W-13, W-14, W-16, W-17, 4 Marks

- Operating systems can be broken into pieces with the support or help of hardware. These parts will be smaller and more appropriate than those allowed by the original MS-DOS or UNIX systems.

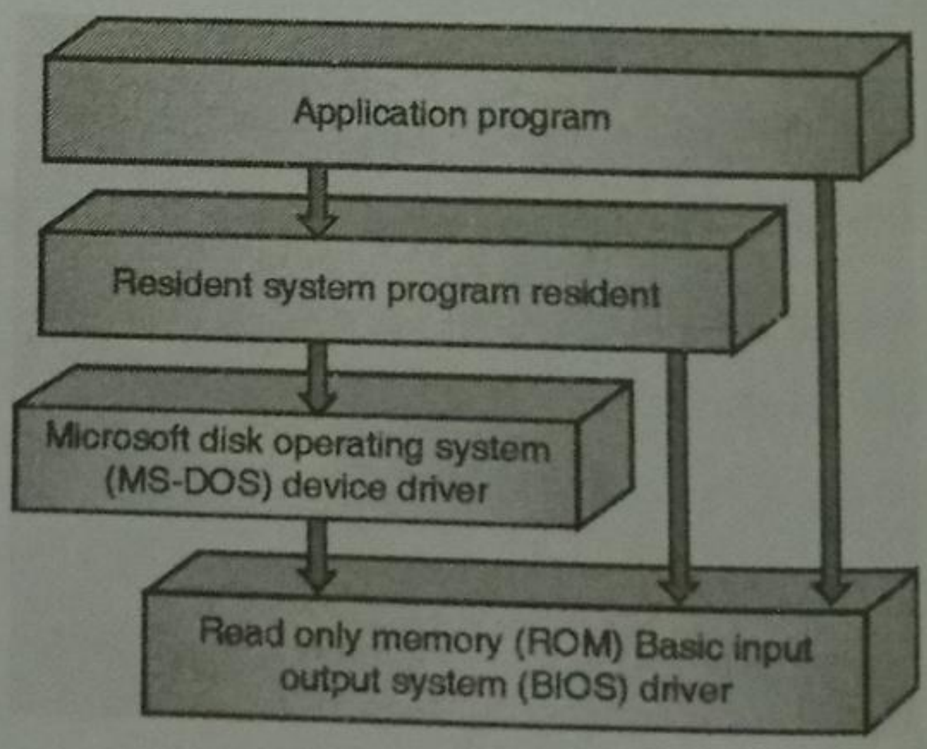


Fig. 2.3.3 : MS-DOS Layered system

- Good operating system can retain great control over the computer and over the applications that make use of that computer.

- Operating system implementers can change the inner workings of the system and create modular operating systems.
- In top to down approach the functions and features are of OS are determined and they are divided into different components.
- One of the methods to make OS modular is a Layered approach. In layered approach the operating system is broken up into a number of layers or levels. It forms a less complicated structure of OS.
- The bottom layer i.e. layer 0 is hardware and the highest (layer N) is the user interface layer. This layering structure is portrayed in Fig. 2.3.4.

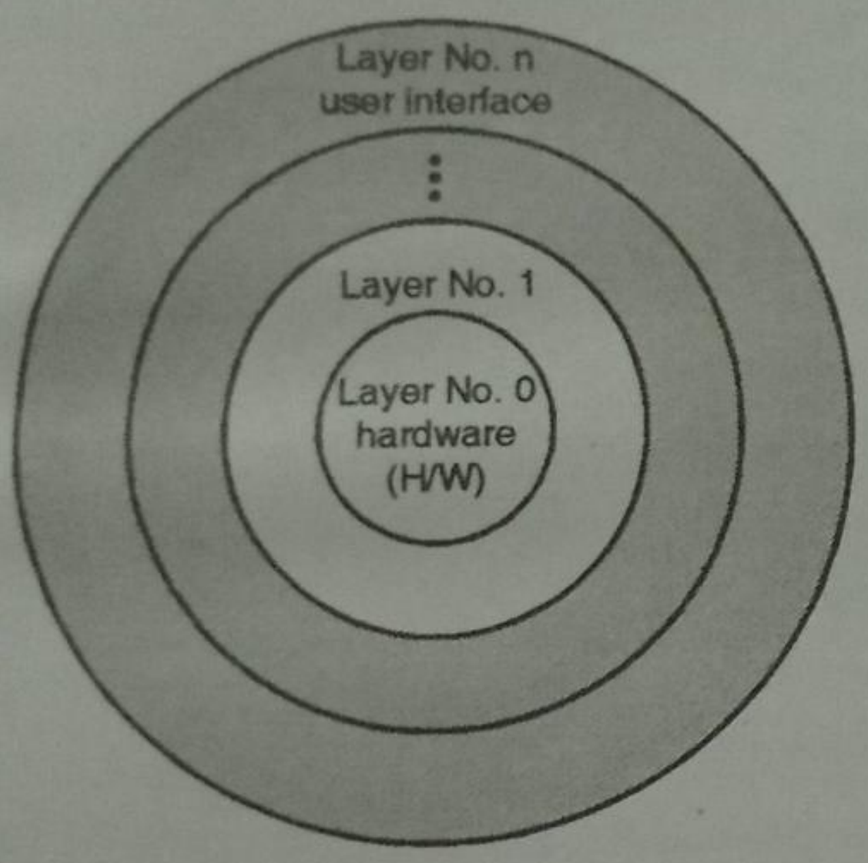


Fig. 2.3.4 : Layered operating system

- An operating-system layer is an implementation of an abstract object consists of data and the operations that can manipulate those data.
- A typical operating system for example consist of M layers of data structures and a set of functions or routines that can be invoked by higher level layers. Also Layer M can invoke operations of lower level layers.
- Important benefit of the layered approach is simplicity of construction and debugging. The layers are selected so that each layer uses function (operations) and services of only lower level layers. This approach makes debugging and system verification simpler.
- It is easy to debug first layer without any worries for the rest of the system, because, by definition, it uses only the basic hardware to implement its function.
- In this way once the first layer is debugged, one by one other level are debugged and so on. If an error is found during the debugging of a particular layer, the error must be on the same layer. In this way the design and implementation of the system are simplified.
- Each layer is implemented with only those operations which are facilitated by lower level layers. A layer need not to bother how these operations are implemented. The layer needs to know only what these operations do exactly.
- Therefore it is essential for each layer to hide the existence of certain data structures, operations and hardware from higher-level layers.
- Defining layer is a main difficulty with the layered approach. It involves appropriately defining various layers. The reason behind this is, a layer can use only lower level layers.
- Layered implementation is inclined to be less efficient than other types. This is a major problem with layered approach of designing OS structure.
- There are different steps, which takes place in layered approach.
- When a user program executes an I/O operation, it executes a system call that is trapped to the I/O layer.
- Then it calls the memory management layer, After that it calls the CPU scheduling layer and then it is passed to the hardware.
- Modifications of the parameters at each layer are difficult as data needs to pass to different layers. It is time consuming process than non layered system.

☞ **Advantages of Layered approach of OS**

1. Layered structure is easy to understand.
2. It allows user to use services of the lower layers
3. Simplicity is most important in the structure.

4. No complicated construction and executions are possible.
5. Debugging is easy in layered approach.

Disadvantages of Layered approach of OS

1. Cannot carry out big tasks because of limitations of layers.
2. For new functions, new layers need to be formed.
3. The structure is old evolved in around 1960s.
4. Modern operating system demands complicated and smart structure.

2.3.1.5 Microkernel

▶ **(MSBTE - W-09, S-11, W-12, W-14, S-16, W-16, W-17, W-18)**

Q. 2.3.6 Draw and explain microkernel operating system structure. (Ref. Sec. 2.3.1.5)

W-09, S-11, W-12, W-14, W-16, W-18, 4 Marks, W-17, 3 Marks

Q. 2.3.7 Describe following operating system structure : Microkernel.
(Ref. Sec. 2.3.1.5) S-16, 3 Marks

- Due to expansion of UNIX the Kernel has became large. It is very difficult to manage the kernel. In the mid-1980s, researchers at Carnegie Melon University developed an operating system called Mach that modularized the kernel using the microkernel approach. This allowed kernel to be separated in different modules.
- Microkernel approach ensures the removal of all unnecessary components from the kernel and they are implemented as system and user level programs.
- The result of this is a smaller kernel. The important thing here is which services should remain in the kernel and which should be implemented in user space. Microkernel provides minimal process and memory management along with the communication function.
- The key function of the microkernel is to facilitate the communication between the client program and the various services that are running in user space. Message passing is the way of communication used.
- In case if client program wish to access a file then it must interact with the file server. There is no direct interaction between client program and service. An interaction happens indirectly by exchanging messages with the microkernel.
- Extending an operating system is one of the benefits of the microkernel approach. In this approach all new services are added to user space which does not need modification of the kernel. IF kernel needs to modify, the changes are fewer, as the microkernel is a smaller kernel. This results into operating system which can be port from one hardware design to another
- Since most of the services are running as a user process rather than kernel processes, the microkernel is able to provide more security and reliability. Incase if any service fails, the rest of the operating system remains untouched.
- Few modern operating systems have used the microkernel approach. Tru64 UNIX provides a UNIX interface to the user with a Mach Kernel.
- There is another example QNX. QNX is real time operating system based on the microkernel design. The QNX microkernel provides services for message passing as well as process scheduling.
- There is a problem which can occur with microkernel and that is reduction in performance due to increased system function overhead. The best example of this is Windows NT first release.
- Windows NT's Microkernel version resulted into low performance compared with that of Windows 95.
- In short Microkernel :

Microkernel moves as much from the kernel into "*user*" space. Communication takes place between user modules using message passing.

Benefits of Microkernel model are :

1. Easier to extend a microkernel
2. Easier to port the operating system to new architectures.
3. More reliable (less code is running in kernel mode)
4. More secured.
5. Very less Detriments.

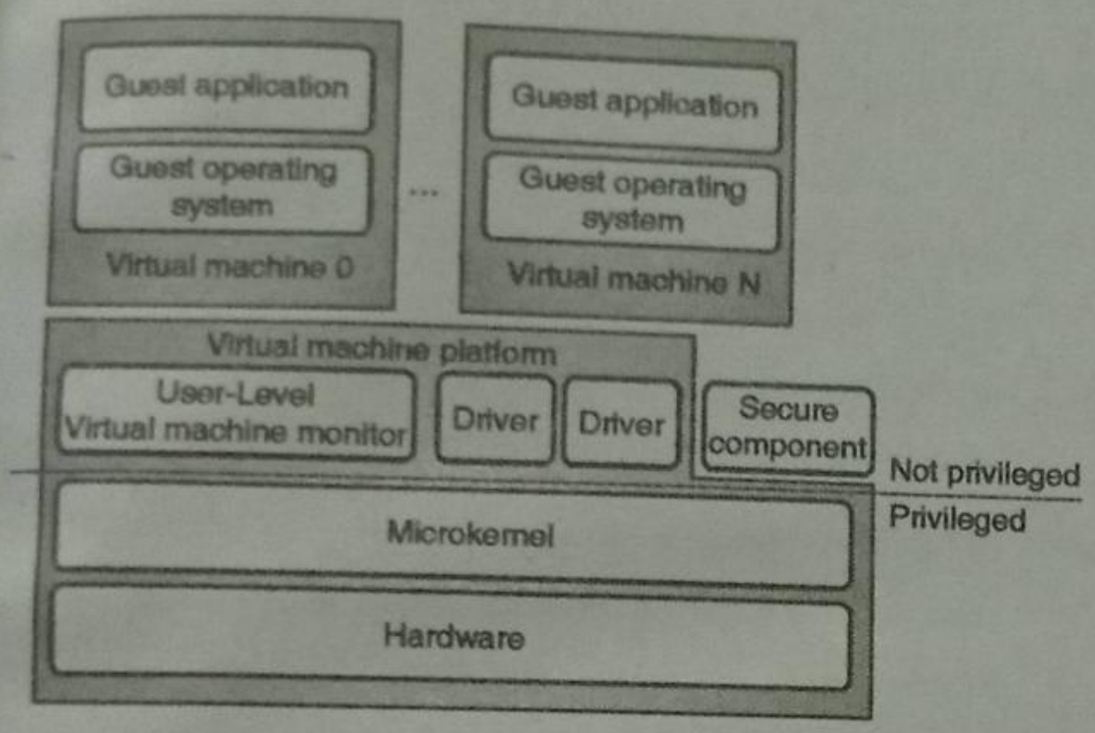


Fig. 2.3.5 : Microkernel

2.3.1.6 Monolithic System

▶ **(MSBTE - W-08, W-09, S-10, W-10, W-11, S-12, S-13, W-13, S-14, W-14, S-15, W-15, S-16, S-17, W-17, W-18)**

Q. 2.3.8 Draw and explain monolithic structure of O.S. (Ref. Sec. 2.3.1.6)

W-08, W-09, S-10, W-10, W-11, S-12, S-13, W-13, S-14, W-14, S-15, W-15, S-17, 4 Marks, W-17, 3 Marks

Q. 2.3.9 Describe following operating system structures : Monolithic (Ref. Sec. 2.3.1.6)

S-16, 3 Marks

Q. 2.3.10 Draw the diagram of monolithic structure of operating system. (Ref. Sec. 2.3.1.6)

W-18, 2 Marks

– This is common approach for an organization. This approach can be referred as "Big mess". The structure can be described, as there is no structure. The operating system is collection of procedures each of which can call any of the other ones whenever it needs to.

– This is a technique which is used where each procedure in the system has a well-defined interface in terms of parameters and results. Each procedure is free to call any other one.

– In the construction of actual object program of the operating system when this approach is used, first compilers compile all the individual procedures, or files containing the procedures. After this it binds them all together into a single object file using the file linker.

– In terms of information hiding, there is essentially none-every procedure is visible to every other procedure or packages, in which much of the information is hidden away inside modules, and only the officially designed entry points can be called from outside the modules.

– Monolithic systems can even have a small structure. The system calls provided by the operating system are requested by arranging parameters in well defined places such as in registers or on the stack and then executing a special trap instruction known as a kernel call or supervisor call.

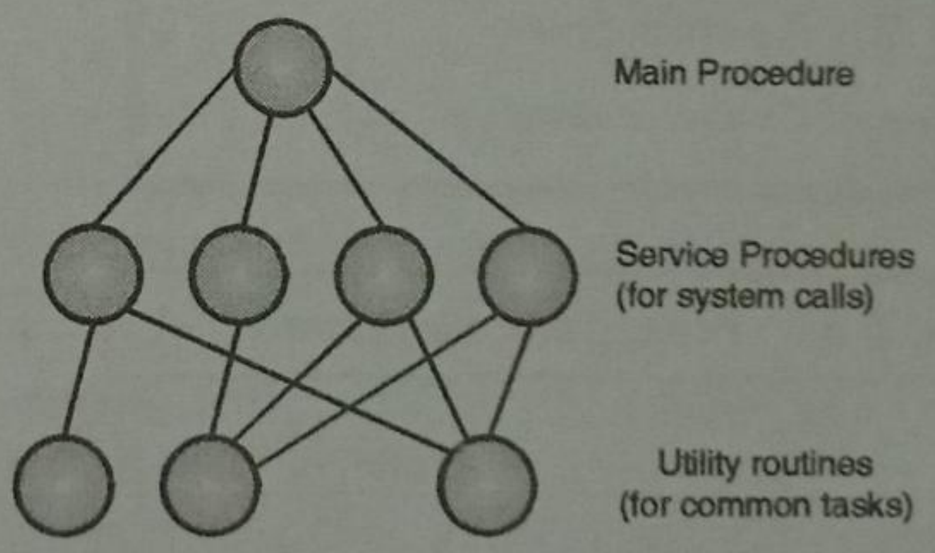


Fig. 2.3.6 : Monolithic structure of operating system

– The monolithic organization propose a basic structure for the operating system:

1. The main program which invokes the requested service procedure.
2. The set of service procedures which carry out the system calls.

3. A set of utility procedures that help the service procedures.

- In this model for each system call there is one service procedure that takes care of it. The utility procedures do things that are needed by several service procedures. E.g. fetching data from user programs. The division of the procedure is shown in the Fig. 2.3.6.

2.4 System Components Activities

▶ **(MSBTE - W-08, S-10, W-10, S-11, S-13, W-17)**

Q. 2.4.1 List and explain system component of system. (Ref. Sec. 2.4)

W-08, S-10, W-10, S-11, S-13, W-17, 4 Marks

1. Process management
2. Main memory management
3. File management
4. I/O system management
5. Secondary storage management.
6. Networking
7. Protection System
8. Command Interpreter

2.4.1 Process Management

▶ **(MSBTE - W-09, W-15, S-16)**

Q. 2.4.2 What is process management ? State four functions to be performed by OS for process management. (Ref. Sec. 2.4.1)

W-09, S-16, 4 Marks

Q. 2.4.3 Describe any four activities of process management. (Ref. Sec. 2.4.1) **W-15, 4 Marks**

- The task executed by the CPU is referred as a process. It could be a program execution which gets execute or it could be internal system tasks executed by CPU.
- The operating system manages many kinds of activities ranging from user programs to system programs like printer spooler, name servers, file server etc. Each of these activities is called as process.
- While executing a program, different resources are required e.g. : input/output devices, data, program, memory etc.
- So obviously these resources becomes the part of process. e.g. : If we execute a program where we need to find out different information related to a file then an input required is name of the file. So process even deals with input and output through programs.
- In short a process requires input that includes the complete execution components like code, data, PC(program counter), registers, OS resources in use etc.
- Important thing to remember is that a process is not a program. A process is only ONE instant or moment of a program in execution. For example a formula execution i.e. calculation in a program could be a process. **Process is unit, which measures the work done by the system.**
- CPU executes the process, so process management is required throughout the execution. Many processes can run on the same program. Process management is an activity which is handled by operating system.
- The five major activities of an operating system related to process management are as follows :

(a) **Creation and deletion of system and user processes** : This activity creates and deletes the user's as well as system's processes.

(b) **Suspension and resumption of processes** : This activity suspends (temporary stops) and continues the process as per situation which occurs in the system.

(c) **A mechanism for process synchronization :** Process synchronization is very important as synchronization ensures the coordination of different process which runs at a time. Synchronization is required when one process may give some input to other process and then execution of second process proceeds.

(d) **A mechanism for process communication :** Processes may need to communicate with each other to ensure its full execution and smooth running of multiple processes by using different resources at a time.

(e) **A mechanism for deadlock handling :** When multiple processes runs at a time and does wait for each other, a deadlock situation occurs. OS handles deadlocks.

2.4.2 Main-Memory Management

► **(MSBTE - S-09, W-12, W-15, W-16, S-18)**

Q. 2.4.4 What are the different responsibilities of memory management ? Explain. (Ref. Sec. 2.4.2) **S-09, W-12, W-16, 4 Marks**

Q. 2.4.5 Describe any four activities of memory management. (Ref. Sec. 2.4.2) **W-15, 4 Marks**

Q. 2.4.6 Explain major activities of memory management component of an operating system. (Ref. Sec. 2.4.2) **S-18, 4 Marks**

- There are two types of memories in computer system. Primary memory which includes RAM, ROM, PROM and EPROM etc. Secondary memory is a mass storage which includes Hard disk, Floppy disk, CD, DVD or even HVD.
- RAM memory is a major part of primary memory or main-memory which is a large array of words or bytes. Each word or byte has its own memory address by which it is referred.
- CPU mainly deals with main memory to execute input/output calls given by it. It's all instructions are executed using main memory. Main-memory provides storage that can be access directly by the CPU. That is to say for a program to be executed, it must be in the main memory.
- The major activities of memory management component of an operating system are :
  1. To keep track of part of memory which are currently being used and by whom? This facilitates to know how much memory is free and occupied by which processes.
  2. Decide which process is to be loaded into main memory when memory space becomes available.
  3. Allocate and de-allocate memory space as per the requirement.

2.4.3 File Management

► **(MSBTE - S-10, S-15)**

Q. 2.4.7 Explain file management in details. (Ref. Sec. 2.4.3) **S-10, S-15, 4 Marks**

- A file is defined as collection of related data. A file could be a program i.e. collection of executable statements. It could be a simple data file which stores data defined by its owner or creator.
- Computer can store files on the disk (secondary storage), which provide long-term storage. Some examples of storage media are magnetic tape, magnetic disk and optical disk.
- Every media from these Medias has its own characteristics like speed, capacity, and data transfer rate and access methods. These devices are basically categorized on their capacity and even size.
- One of the main function of an operating system is to decide where and how to store the data on to the disk. It is assumed that storage devices stores data in the locations called as tracks, sectors and cluster. There is a special system which takes care of files and directories.

- Directories are memory location which stores or organizes files into it. File systems normally organized into directories to ease their use. These directories may contain files and other directions.
- The five key activities of an operating system's file management component are :
  1. The creation and deletion of files.
  2. The creation and deletion of directories.
  3. The support of primitives for manipulating files and directories.
  4. The mapping of files onto secondary storage.
  5. The back up of files on stable storage media i.e. on secondary storages.

2.4.4 Input / Output System Management (Functions)

► (MSBTE - W-09, W-12, S-17)

Q. 2.4.8 Explain the I/O system management component of OS. State different functions required in I/O system management. (Ref. Sec. 2.4.4) W-09, W-12, 6 Marks

Q. 2.4.9 Describe activities of I/O system. (Ref. Sec. 2.4.4) S-17, 2 Marks

- I/O management is a major function of an operating system which mange input output system inclusive of hardware, I/O devices with device drivers and its supporting systems.
- I/O subsystem hides the complexity of hardware devices from the user. Only the device driver knows the peculiarities of the specific device to which it is assigned.
- I/O devices must be utilized properly so they will be utilized maximum. So operating system keeps track of I/O devices to assign jobs to it.
- When I/O devices are busy and job is in process then data are kept in the buffers and even some other techniques called spooling are used.
- In short the IO management function includes:
  1. Input output operations management
  2. Hiding complexity of IO devices from user.
  3. Keep I/O devices continuously busy
  4. Memory management for IO devices

2.4.5 Secondary-Storage Management

► (MSBTE - S-13, W-13, S-15, W-15, S-17, S-18, W-18)

Q. 2.4.10 Explain Secondary storage management. (Ref. Sec. 2.4.5) S-13, W-13, W-18, 4 Marks

Q. 2.4.11 Describe any four secondary storage management activities. (Ref. Sec. 2.4.5) S-15, W-15, 4 Marks, S-17, 2 Marks

Q. 2.4.12 What are the activities involved in secondary storage management? (Ref. Sec. 2.4.5) S-15, 4 Marks

Q. 2.4.13 List three main levels of data storage and explain cache storage. (Ref. Sec. 2.4.5) S-18, 4 Marks

- There are mainly three levels of data storage as follows :
  1. Primary storage
  2. Secondary storage
  3. Cache storage
- While program is running, it is necessary to keep instructions and data in primary memory or cache memory for the purpose of instant access.
- Also main memory is a small and volatile. We cannot store huge amount of data permanently in it. We may lose data if we try to store it in main memory. Because of this, secondary memory is very much required to store data as well as programs. Secondary memory is used to backup the primary memory.
- Secondary storage or secondary memory can be hard disk, floppy disk, magnetic tape, CD, DVD, HVD, Pen

drive, memory card etc. The data present in secondary devices are in the form of bits and bytes.

- Every place or location in the storage device has got its own address where data is present or may not present. The collection or set of all addresses that are available to a program is known as address space.
- The major functions related to secondary storage management are as follows :

(a) Managing free space existing on secondary storage devices.

(b) Allotment of storage space when new files have to be written or create.

(c) Setting up or scheduling the requests for memory access.

2.4.6 Networking

- Networking technology allows multiple computers to get connected to each other. A distributed system can be referred as a collection of processors that do not share the computer resources like memory, peripheral devices, or a clock etc.
- The communication that takes place among processors through communication lines is referred as network. It is important to consider routing strategies, connections, contention and security related data while designing the system.
- There is difference in the functions that are provided by the OS in case of standalone machine and network system. Operating system takes care for data sharing, data sending, data receiving, routing data etc.

2.4.7 Protection System

- In case when multiple users interact with the computer then data security becomes an important issue. Also when concurrent executions are taking place in the machine, internal protection plays important role. Every activity or process should be executed in isolation.
- Protection is a kind of mechanism that controls the access of programs, processes, or users to the resources of computer system.

2.4.8 Command Interpreter System

- A command interpreter is an interface between operating system and the user. The user gives commands which are executed by operating system.
- The main function of a command interpreter is to get and execute the user specified command. It is not the part of the kernel, since multiple command interpreters (E.g. shell, in UNIX) may be supported by an operating system, and they do not really need to run in kernel mode.

2.5 System Boot

▶ **(MSBTE - W-08, S-10, S-11, W-11, W-12, S-13, S-14, W-17, S-18)**

Q. 2.5.1 Explain booting and its procedure in detail with the help of diagram. (Ref. Sec. 2.5)

W-08, S-10, S-11, W-11, W-12, S-13, S-14, W-17, 4 Marks

Q. 2.5.2 With neat labelled diagram, explain the working of booting process. (Ref. Sec. 2.5)

S-18, 4 Marks

- The procedure of starting a computer by loading the kernel is known as **booting the system**. Bootstrap is a smaller piece of code which finds outs where kernel is and locates it in main memory and starts its execution.
- When the system is rebooted or powered up, the instruction registers are loaded in predefined memory locations, and then execution starts. There exists a program bootstrap loader. This program is stored in the read only memory. Read only data exists in ROM memory which is permanent.
- Bootstrap is stored in ROM instead of RAM as RAM memory is unknown in the initial stage when machine

starts or boots. But on the other hand ROM memory does not require initialization and cannot infect by virus problems.

- The bootstrap program is very powerful which performs many activities. One of the activities is to execute the diagnostic test. This diagnostic test checks whether all devices are in proper state to continue the system program or not. It also initializes the CPU registers, device controllers etc. After doing these activities it starts operating system in main memory.
- The operating system software is actually stored in the disk of computer. But when bootstrap program starts the operating system gets load in main memory or RAMS memory. The reason behind this is ROM memory is small size due to which OS cannot be stored in main memory.
- But in case of some derives like Cellular phones, PDA's and game consoles, an operating system is stored in the ROM memory. All forms of ROM memory are called as firmware. Different forms of ROM are PROM, EPROM, EEPROM etc.
- The main problem with ROM is it executes code very slowly. Sometimes solution for this problem could be solved by storing memory in ROM and then by transferring it to the main memory or RAM memory. Some major operating systems like windows, UNIX, MAC, OS x stores the OS into disk and then gets transfer into main memory.
- Boot block is an area in the disk which reads and then transfers to main memory. The disk which has boot partition is called as boot disk or system disk.
- Once the full bootstrap program has been loaded, it transverses the file system to find operating system kernel, loads it into memory and starts its execution. At this point it is said that system running.
- In short :
  1. Small piece of code called **bootstrap loader**, locates the kernel, loads it into memory, and starts it.
  2. Sometimes two-step process where **boot block** at fixed location loads bootstrap loader.
  3. When power initialized on system, execution starts at a fixed memory location.
  4. Firmware used to hold initial boot code.

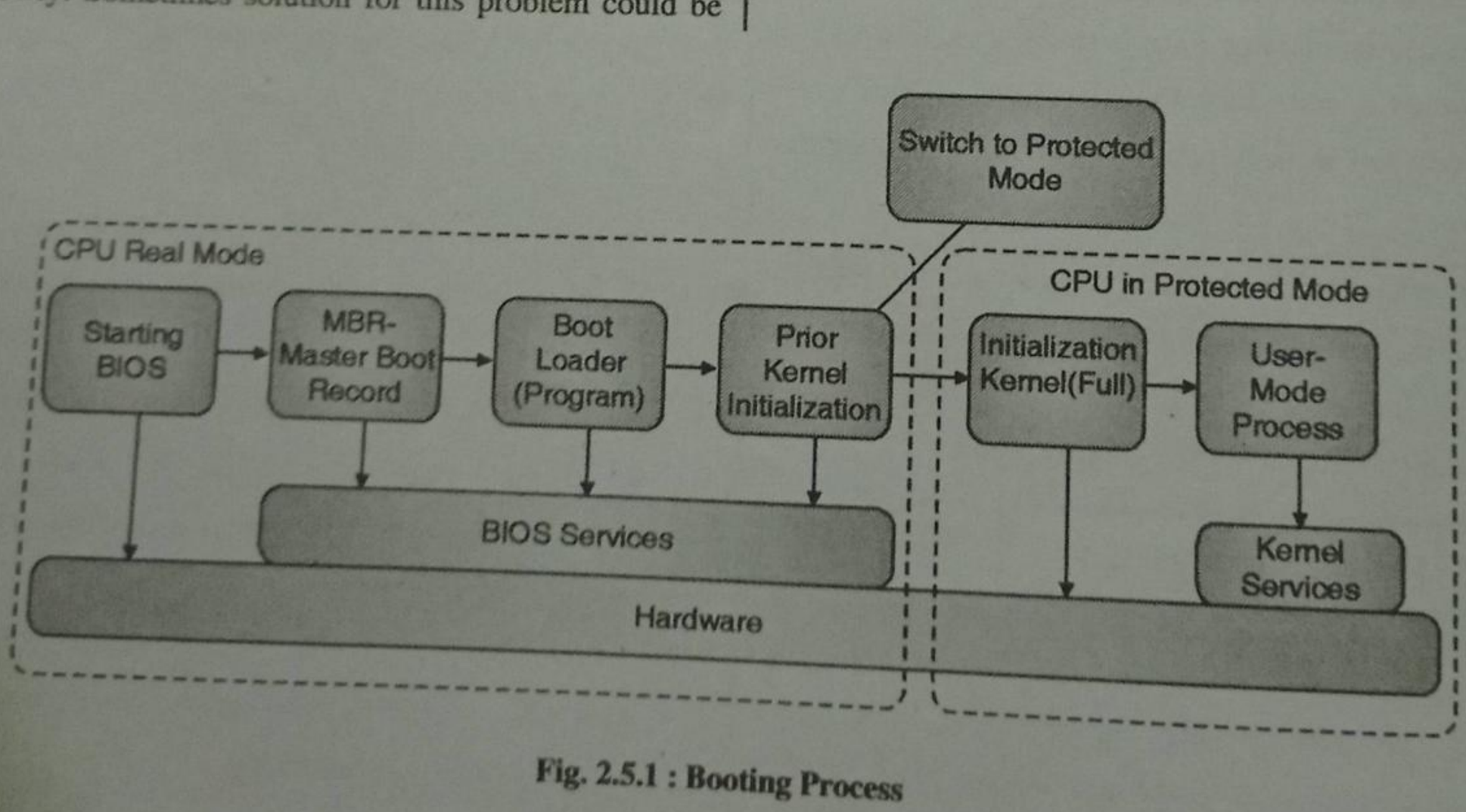


Fig. 2.5.1 : Booting Process

2.6 Use of Operating System Tools

- Operating system tools are special system programs that are used to manage the overall functions of an operating system.
- These tools are used for following purposes:
  (a) Memory Management
  (b) Processor Management
  (c) Device Management
  (d) File Management
  (e) Security
  (f) Control over system performance
  (g) Job accounting
  (h) Error detecting aids
  (i) Coordination between users and other software
- Some of the tools are discussed below :

1. User Management

- User management is one of the important tools that OS has to provide. Operating system can be multiuser or single user.
- In case of multiple users, the levels of rights and privileges can be different. So operating system needs to keep log of everything related to the user. It could be even login and logout times.
- OS provides tools to create users,modify them or even delete them from the system.
- Managing users is nothing but managing different user accounts. In case multiuser system, it is a complicated task. OS provides basic security to the organization with the help of this service.
- It even ensures required and limited access to the system.

2. Security Policy

- There is variety of operating systems available in market. Every OS has standard security policies. Implementation of most of the security policies depends on the organization.
- Security policies are used by OS to ensure that it maintains availability, integrity and confidentiality.
- OS security policy helps in protecting systems from various threats to the system , malware, spyware, ransomware, viruses, worms, malware, backdoor intrusions, and many more.
- Security policies provide all possible preventative measures and techniques to keep operating system safe. It also protects OS from activities like data theft, unwanted modification to data, deletion of data etc.
- There are wide operations performed by OS security policies. Some of the major areas are listed below:
  (a) System updating or patching system regularly.
  (b) Software installation or updation of various antiviruses.
  (c) Firewall installation and management of incoming and outgoing traffic on networks.
  (d) Provide security to users account.
  (e) Deciding what kind of data, users and hardware is vital in the system.

(a) Device Management

- Device management is needed for smooth functioning of computer system. One of the most vital operations of device management is resource allocation. Here resource means devices.
- In multiuser environment input, output and network devices and processors need to be allocated to the required users. And this allocation is not very easy.
- Operating system is capable of device communication. It communicates and does coordination among devices.

Some of the activities of device management are as follows :

(a) Keep track of present devices.

(b) Resource/ device allocation

(c) Choose efficient ways to allocate devices to different programs or processes.

(d) De-allocate the devices.

(b) Performance Monitor

- Performance monitor supervises performance of the system.
- It monitors different activities like memory usage and CPU processing.
- It also monitors performance of the hardware, software and applications.
- It analyses the different problems in the system and can find out by monitoring function.
- The function of performance monitor is much complicated on networks.

(c) Task Scheduler

- Task scheduler can be called as CPU scheduler. Throughout the lifetime of process it migrates or travels to the various scheduling queues.
- An operating system should select processes for scheduling purposes from these queues in some manner or style. The selection procedure is done by **schedulers**.
- In a batch system there can be many processes which can be executed immediately.
- In batch systems if processes are more then they are spooled to the mass storage devices like disks in which they are kept for the later execution.
- The scheduler which picks up job from this pool and loads into main memory for execution is called as **long term scheduler or job scheduler**.
- There is another scheduler which selects the jobs or processes which are ready to execute from this pool and allocates the CPU to one of them is called **short term scheduler or CPU scheduler**.

2.7 MSBTE Questions and Answers

Winter 2008 : Total Marks 20

Q. 1 State and explain six major components of system. *(Ans. : Refer Section 2.4)* **(6 Marks)**

Q. 2 Draw and explain monolithic structure of O.S. *(Ans. : Refer Section 2.3.1.6)* **(4 Marks)**

Q. 3 What is System call? Explain types of system calls. *(Ans. : Refer Sections 2.2 and 2.2.3)* **(6 Marks)**

Q. 4 What are the steps involved in booting ? *(Ans. : Refer Section 2.5)* **(4 Marks)**

Summer 2009 : Total Marks 16

Q. 5 Write any four services provided by operating system. *(Ans. : Refer Section 2.1)* **(4 Marks)**

Q. 6 What is a system call ? Enlist any four system calls related to files. *(Ans. : Refer Sections 2.2 and 2.2.3)* **(4 Marks)**

Q. 7 Describe layered approach to system design. Also write its advantages and disadvantages. *(Ans. : Refer Section 2.3.1.4)* **(4 Marks)**

Q. 8 What are the different responsibilities of memory management ? Explain. *(Ans. : Refer Section 2.4.2)* **(4 Marks)**

Winter 2009 : Total Marks 20

Q. 9 What is process management ? State four functions to be performed by OS for process management. *(Ans. : Refer Section 2.4.1)* **(4 Marks)**

Q. 10 What is the purpose of system calls? State two systems calls with its functions. *(Ans. : Refer Sections 2.2 and 2.2.3)* **(4 Marks)**

Q. 11 Explain following operating system structure in detail.

(I) Monolithic *(Ans. : Refer Section 2.3.1.6)*

(ii) Microkernel
(Ans. : Refer Section 2.3.1.5) **(6 Marks)**

Q. 12 Explain the I/O system management component of OS. State different function required in I/O system management.
(Ans. : Refer Section 2.4.4) **(6 Marks)**

Summer 2010 : Total Marks 24

Q. 13 Draw neat diagram and explain operating system structure. *(Ans. : Refer Section 2.3)* **(4 Marks)**

Q. 14 List different types of system calls mention their uses. *(Ans. : Refer Section 2.2.3)* **(4 Marks)**

Q. 15 List system component. Explain file management in details. *(Ans. : Refer Sections 2.4 and 2.4.3)* **(4 Marks)**

Q. 16 Explain system booting in detail.
(Ans. : Refer Section 2.5) **(6 Marks)**

Q. 17 Explain :

(i) Monolithic OS structure
(Ans. : Refer Section 2.3.1.6)

(ii) Layered OS structure
(Ans. : Refer Section 2.3.1.4) **(6 Marks)**

Winter 2010 : Total Marks 18

Q. 18 Describe monolithic operating system structure.
(Ans. : Refer Section 2.3.1.6) **(4 Marks)**

Q. 19 What is system call? Enlist any four system calls related with process management.
(Ans. : Refer Section 2.2 and 2.2.3) **(4 Marks)**

Q. 20 Describe file management. Enlist the system calls for file management.
(Ans. : Refer Sections 2.4.3 and 2.2.3(2)) **(4 Marks)**

Q. 21 Enlist system components. Describe any two in details. *(Ans. : Refer Section 2.4)* **(6 Marks)**

Summer 2011 : Total Marks 20

Q. 22 Explain any three system components with their activities. *(Ans. : Refer Section 2.4)* **(4 Marks)**

Q. 23 Draw and explain microkernel operating system structure. *(Ans. : Refer Section 2.3.1.5)* **(4 Marks)**

Q. 24 Explain any six operating system services.
(Ans. : Refer Section 2.1) **(4 Marks)**

Q. 25 What is system call? State any four system calls for process control and file management.
(Ans. : Refer Sections 2.2 and 2.2.3) **(4 Marks)**

Q. 26 Explain Booting procedure in detail with the help of diagram. *(Ans. : Refer Section 2.5)* **(4 Marks)**

Winter 2011 : Total Marks 16

Q. 27 What is a system call ? Explain use of any two categories of system call.
(Ans. : Refer Sections 2.2 and 2.2.3) **(4 Marks)**

Q. 28 Explain booting in detail.
(Ans. : Refer Section 2.5) **(4 Marks)**

Q. 29 Draw and explain monolithic structure of operating system. *(Ans. : Refer Section 2.3.1.6)* **(4 Marks)**

Q. 30 State any four types of process related system call ? Explain any one system call in detail.
(Ans. : Refer Section 2.2.3) **(4 Marks)**

Summer 2012 : Total Marks 16

Q. 31 What are system calls? Explain system call implementation with its working.
(Ans. : Refer Sections 2.2 and 2.2.1) **(4 Marks)**

Q. 32 Explain :

(i) Monolithic operating system structure
(Ans. : Refer Section 2.3.1.6)

(ii) Layered OS structure
(Ans. : Refer Section 2.3.1.4) **(4 Marks)**

Q. 33 List services provided by operating system and explain any four of them.
(Ans. : Refer Section 2.1) **(4 Marks)**

Q. 34 Enlist any four system calls related memory management. *(Ans. : Refer Section 2.2.3 (6))* **(4 Marks)**

Winter 2012 : Total Marks 24

Q. 35 List any four services provided by OS? Explain any one of them? *(Ans. : Refer Section 2.1)* **(4 Marks)**

Q.36 Explain the I/O System Management component of OS. State different function required in I/O System Management. *(Ans. : Refer Section 2.4.4)* **(4 Marks)**

Q. 37 Explain Microkernel OS structure? *(Ans. : Refer Section 2.3.1.5)* **(4 Marks)**

Q. 38 What is the purpose of system calls? State two system calls with is function? *(Ans. : Refer Sections 2.2 and 2.2.3)* **(4 Marks)**

Q. 39 Explain system booting in detail? *(Ans. : Refer Section 2.5)* **(4 Marks)**

Q. 40 What are the different responsibilities of memory management? Explain. *(Ans. : Refer Section 2.4.2)* **(4 Marks)**

Summer 2013 : Total Marks 20

Q. 41 Explain secondary storage management. *(Ans. : Refer Section 2.4.5)* **(4 Marks)**

Q. 42 Enlist system components. Describe any two in detail. *(Ans. : Refer Section 2.4)* **(4 Marks)**

Q. 43 Explain layered approach operating system. *(Ans. : Refer Section 2.3.1.4)* **(4 Marks)**

Q. 44 Describe Monolithic Operating system structure. *(Ans. : Refer Section 2.3.1.6)* **(4 Marks)**

Q. 45 Explain booting in detail. *(Ans. : Refer Section 2.5)* **(4 Marks)**

Winter 2013 : Total Marks 22

Q. 46 Enlist any four system calls related memory management. *(Ans. : Refer Section 2.2.3(6))* **(2 Marks)**

Q. 47 Explain :

(i) Monolithic operating sys. structure *(Ans. : Refer Section 2.3.1.6)*

(ii) Layered OS structure *(Ans. : Refer Section 2.3.1.4)* **(8 Marks)**

Q. 48 List services provided by operating system and explain any four of them. *(Ans. : Refer Section 2.1)* **(4 Marks)**

Q. 49 What are system calls? Explain system call implementation with its working. *(Ans. : Refer Sections 2.2 and 2.2.1)* **(4 Marks)**

Q. 50 Explain secondary storage management. *(Ans. : Refer Section 2.4.5)* **(4 Marks)**

Summer 2014 : Total Marks 20

Q. 51 Describe monolithic operating system structure. *(Ans. : Refer Section 2.3.1.6)* **(4 Marks)**

Q. 52 Explain microlevel OS structure. *(Ans. : Refer Section 2.3)* **(4 Marks)**

Q. 53 Explain system booting in detail. *(Ans. : Refer Section 2.5)* **(4 Marks)**

Q. 54 What is the purpose of system calls? State two system calls with its functions. *(Ans. : Refer Sections 2.2 and 2.2.3)* **(4 Marks)**

Q. 55 What are different responsibilities of memory management. *(Ans. : Refer Section 2.4.2)* **(4 Marks)**

Winter 2014 : Total Marks 20

Q. 56 Draw Microkernel OS structure. *(Ans. : Refer Section 2.3.1.5)* **(4 Marks)**

Q. 57 List services provided by OS and explain any four of them. *(Ans. : Refer Section 2.1)* **(4 Marks)**

Q. 58 What is system call ? Explain use of any two categories of system call. *(Ans. : Refer Sections 2.2 and 2.2.3)* **(4 Marks)**

Q. 59 Explain :

(1) Monolithic OS structure *(Ans. : Refer Section 2.3.1.6)* **(4 Marks)**

(2) Layered OS structure *(Ans. : Refer Section 2.3.1.4)* **(4 Marks)**

Summer 2015 : Total Marks 22

Q. 60 Draw and explain monolithic structure of operating system. *(Ans. : Refer Section 2.3.1.6)* **(6 Marks)**

Q. 61 Describe any four secondary storage management activities.
(Ans. : Refer Section 2.4.5) **(4 Marks)**

Q. 62 What is system call ? Enlist any four system call.
(Ans. : Refer Sections 2.2 and 2.2.3) **(4 Marks)**

Q. 63 Describe any four services provided by an operating system.
(Ans. : Refer Section 2.1) **(4 Marks)**

Q. 64 Describe file management. Enlist the system call for file management.
(Ans. : Refer Sections 2.4.3 and 2.2.3(2)) **(4 Marks)**

Winter 2015 : Total Marks 26

Q. 65 Draw and explain monolithic structure of operating system.
(Ans. : Refer Section 2.3.1.6) **(4 Marks)**

Q. 66 Explain any six services provided by operating system. Draw diagram showing services.
(Ans. : Refer Section 2.1) **(6 Marks)**

Q. 67 Describe any four activities of process management and memory management.
(Ans. : Refer Sections 2.4.1 and 2.4.2) **(4 Marks)**

Q. 68 List any four system calls for device management and communication.
(Ans. : Refer Section 2.2.3(3) and (5)) **(4 Marks)**

Q. 69 Describe any four secondary storage management activities.
(Ans. : Refer Section 2.4.5) **(4 Marks)**

Q. 70 What is system call ? Explain open( ) system call and close( ) system call.
(Ans. : Refer Section 2.2) **(4 Marks)**

Summer 2016 : Total Marks 34

Q. 71 What is process management? State four functions to be performed by OS for process management.
(Ans. : Refer Section 2.4.1) **(4 Marks)**

Q. 72 Describe following operating system structures :
(i) Monolithic *(Ans. : Refer Section 2.3.1.6)*
(ii) Microkernel. *(Ans. : Refer Section 2.3.1.5)*
(6 Marks)

Q. 73 Compare Unix and Linux operating system w.r.t.
(1) User interface
(2) Name of provider
(3) Processing speed
(4) Security.
(Ans. : Refer Section 2.3.1.3) **(4 Marks)**

Q. 74 List any four operating system services and describe in one/two sentences.
(Ans. : Refer Section 2.1) **(4 Marks)**

Q. 75 What is system call ? List types of system call with one example of system call.
(Ans. : Refer Sections 2.2 and 2.2.3) **(4 Marks)**

Q. 76 What are the activities involved in secondary storage management?
(Ans. : Refer Section 2.4.5) **(4 Marks)**

Q. 77 Explain how parameter passing is done while implementing system calls.
(Ans. : Refer Sections 2.2.1 and 2.2.2) **(4 Marks)**

Q. 78 Draw and explain structure of Unix operating system. *(Ans. : Refer Section 2.3.1.2)* **(4 Marks)**

Winter 2016 : Total Marks 20

Q. 79 Describe the purpose of system calls ? State two system calls with its functions.
(Ans. : Refer Section 2.2) **(4 Marks)**

Q. 80 List and state any four services provided by an operating system.
(Ans. : Refer Section 2.1) **(4 Marks)**

Q. 81 Explain layered operating system structure.
(Ans. : Refer Section 2.3.1.4) **(4 Marks)**

Q. 82 Explain microkernel operating system structure.
(Ans. : Refer Section 2.3.1.5) **(4 Marks)**

Q. 83 What are the different responsibilities of memory management? Explain.
(Ans. : Refer Section 2.4.2) **(4 Marks)**

Summer 2017 : Total Marks 22

Q. 84 State any four types of system calls provided by an operating system.
(Ans. : Refer Section 2.2.3) **(4 Marks)**

Q. 85 State and describe services provided by an operating system.
(Ans. : Refer Section 2.1) **(6 Marks)**

Q. 86 Describe activities of I/O system and secondary storage management. (four each)
(Ans. : Refer Sections 2.4.4 and 2.4.5) **(4 Marks)**

Q. 87 With neat diagram, Explain Monolithic structure of operating system.
(Ans. : Refer Section 2.3.1.6) **(4 Marks)**

Q. 88 What is system call? With the help of diagram explain open( ) system call.
(Ans. : Refer Section 2.2) **(4 Marks)**

Winter 2017 : Total Marks 26

Q. 89 What is system call? Enlist any four system calls related with process management.
(Ans. : Refer Sections 2.2 and 2.2.3) **(4 Marks)**

Q. 90 Explain following operating system structure in details.
(1) Monolithic
(2) Microkernel
(Ans. : Refer Sections 2.3.1.5 and 2.3.1.6)
(6 Marks)

Q. 91 List any four services provided by OS and explain any two of them.
(Ans. : Refer Section 2.1) **(4 Marks)**

Q. 92 Explain booting procedure in details with the help of diagram.
(Ans. : Refer Section 2.5) **(4 Marks)**

Q. 93 Explain layered approach operating system.
(Ans. : Refer Section 2.3.1.4) **(4 Marks)**

Q. 94 Enlist system components. Describe any two in detail. *(Ans. : Refer Section 2.4)* **(4 Marks)**

Summer 2018 - Total Marks 20

Q. 95 List three main levels of data storage and explain cache storage.
(Ans. : Refer Section 2.4.5) **(4 Marks)**

Q. 96 Explain following two services of operating system. **(4 Marks)**
(i) File system manipulation
(Ans. : Refer Section 2.1.4)
(ii) Resource Allocation
(Ans. : Refer Section 2.1.7)

Q. 97 List types of system call and explain the system call –"Information Maintenance".
(Ans. : Refer Sections 2.2 and 2.2.3) **(4 Marks)**

Q. 98 Explain major activities of memory management component of an operating system.
(Ans. : Refer Section 2.4.2) **(4 Marks)**

Q. 99 With neat labelled diagram explain the working of Booting process.
(Ans. : Refer Section 2.5) **(4 Marks)**

Winter 2018 - Total Marks 18

Q. 100 What is system call?
(Ans. : Refer Section 2.2) **(2 Marks)**

Q. 101 List any four services provided by operating system.
(Ans. : Refer Section 2.1) **(2 Marks)**

Q. 102 Draw the diagram of monolithic structure of operating system.
(Ans. : Refer Section 2.3.1.6) **(2 Marks)**

Q. 103 Explain any four file related system calls. *(Ans. : Refer Section 2.2.3(2))* **(4 Marks)**

Q. 104 Explain secondary storage management.
(Ans. : Refer Section 2.4.5) **(4 Marks)**

Q. 105 Describe microkernel structure of operating system.
(Ans. : Refer Section 2.3.1.5) **(4 Marks)**

Chapter Ends...

❑❑❑

CHAPTER 3

UNIT III

Process Management

Syllabus

3.1 Process - Process states, Process Control Block (PCB).

3.2 Process Scheduling - Scheduling Queues, Schedulers, Context switch.

3.3 Inter-process communication (IPC) - Introduction, shared memory system and message passing system.

3.4 Threads - Benefits, users and kernel threads, Multithreading Models - Many to One, One to One, Many to Many.

3.5 Execute process commands - like ps, wait, sleep, exit, kill.

3.1 Introduction to Process Management

▶ (MSBTE - W-08, S-13, W-18)

Q. 3.1.1 Explain process management in detail. (Ref. Sec. 3.1) **W-08, S-13, W-18, 4 Marks**

Q. 3.1.2 What is process management? State any four functions of process management. (Ref. Sec. 3.1) **W-18, 4 Marks**

- In earlier days computer systems used to run single program at a time. But now a day's computers run multitasking operating systems. The systems run multiple programs at a time. Because of multitasking environment, different processes or different jobs are created for multiple programs.
- When one program or job runs and if it does not require CPU then it may be assigned to other devices like I/O devices. Till that time CPU should not be free so may be another process is assigned to CPU. For this kind of processing the process management is required.
- Process management includes creating, running, terminating and assigning different processes to different devices. Even process scheduling is a part of process management where the sequence and priorities of the processes are defined.
- The term job and process can be used interchangeably. Operating system has to perform many operations at a time. It is not only program related or CPU related jobs but it performs certain operations like memory management and even other input output operations.
- Multiprogramming systems explicitly allow multiple processes to exist at any given time, where only one is using the CPU at any given moment, while the remaining processes are performing I/O or they are waiting.
- The process manager is one of the main parts of the four major parts of the operating system. It implements the process abstraction. Process abstraction hides the complexity of the processes from the users. It does this by creating a model for the way the process uses CPU and any system resources.
- The major reason of complexity of the operating system is the need for multiple processes to share the hardware at the same time. As a consequence of this goal, the process manager implements CPU sharing i.e. CPU scheduling, process synchronization mechanisms, and a deadlock strategy.
- In addition, the process manager implements part of the operating system's protection and security. Process is a sequential program in execution. The components of a process are as follows :

1. The object program to be executed (called the program text in UNIX).
2. The data on which the program will execute (obtained from a file or interactively from the process's user).
3. Resources required by the program (for example, files containing requisite information)
4. The status of the process's execution.

☞ **Functions of process management**

1. Process creation
2. Process termination
3. Process suspension
4. Make process sleep

3.1.1 Process Concept

▶ (MSBTE - S-09, S-10, S-12, W-12, W-14, S-15, W-15, W-16, S-17, W-18)

Q. 3.1.3 What is process? How is it different from a program? (Ref. Sec. 3.1.1) **S-09, S-10, S-12, W-12, W-14, W-15**

Q. 3.1.4 Define process. (Ref. Sec. 3.1.1) **S-15, W-16, W-18, 4 Marks**

Q. 3.1.5 Describe Process in memory with diagram. (Ref. Sec. 3.1.1) **S-17, 4 Marks**

- **Process** is a program in execution. This is a typical definition of a process. A process does not mean only program but it could contain some part called as **text section**. It may contain the current activity, represented by the value of the program counter and the contents of CPU register.
- The main difference between program and process is that program is user written and process is generated by the operating system to run small part of that program.
- The program is converted into different processes to execute by the CPU. Multiple processes represent a program at the time of execution.
- Another part of process is a **stack**. Stack is basically used to store the temporary values. The temporary values can be parameter or return values of functions, local variables, addresses of the return values etc.
- There is another part called data section. Data section stores the global variable. Global variables are accessible throughout the program.
- Sometimes it is necessary to use dynamic memory allocation while executing program. The Heap is a part, which is used for dynamic memory allocation. Fig. 3.1.1 shows the memory state.

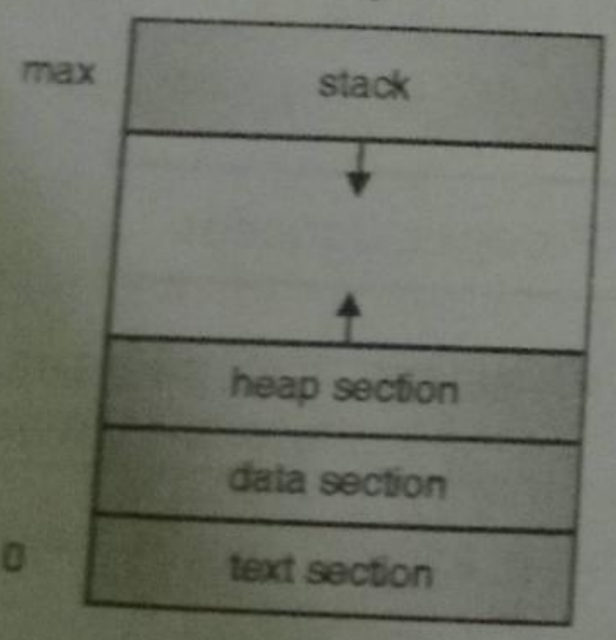


Fig. 3.1.1 : Process in memory

Process is also refereed as an active entity. A program cannot be directly referred as a process. e.g.: A file contains the instructions or commands given to machine. This file is present on the disk. The file is called as executable file. It can not be directly said that this executable file is a process.

- In case when executable file gets loaded into memory and start executing instructions one by one using program counter, it can be called as process.
- Two common techniques to run executable files are double clicking on the file name icon and run it or run it on the command prompt by inputting its name.
- There could be two processes associated with the same program. But remember that these two processes are treated as separate processes.
- This may happen that same user may run multiple copies of a same program at a time. e.g.: One user may load multiple copies of web browser or mail at a time. Though the program is same, its different copies will be created.
- For all these copies different processes will be created in the memory. Processes created are same but there is difference in the sections of processes. The **text** section will be same but other sections like stack, data and heap will be different.
- There is one more case that when a process runs in the memory may generate multiple processes. In short :
  - An operating system executes a variety of programs.
  - The terms job and process are used almost interchangeably.
  - Process - a program in execution; process execution must progress in sequential fashion.
  - A process includes :
    1. Text section (program counter)
    2. Stack
    3. Data section
    4. Heap section

3.1.2 The Process States

▶ **(MSBTE - W-08, S-09, W-09, S-10, W-10, W-11, S-12, W-12, W-13, W-14, S-15, W-15, S-16, W-17, S-18, W-18)**

Q. 3.1.6 Explain the different process states with the help of state diagram. (Ref. Sec. 3.1.2) **W-08, S-09, W-09, S-10, W-10, W-11, S-12, W-12, W-13, W-14, S-15, W-15, 4 Marks**

Q. 3.1.7 Draw and explain process state diagram. (Ref. Sec. 3.1.2) **W-15, 4 Marks**

Q. 3.1.8 Draw the process state diagram and describe each state in one/two sentences. (Ref. Sec. 3.1.2) **S-16, 4 Marks**

Q. 3.1.9 State and explain different process state. (Ref. Sec. 3.1.2) **W-17, 4 Marks**

Q. 3.1.10 List and draw a neat labelled diagram of process state. (Ref. Sec. 3.1.2) **S-18, 4 Marks**

Q. 3.1.11 Draw process state diagram and describe each state. (Ref. Sec. 3.1.2) **W-18, 4 Marks**

- As the process runs, it changes its states. Process states are the different stages through which it passes. The state of a process is defined in part by the current activity of that process.
- During the lifespan of a process, its execution status may be in one of four states. : (associated with each state is usually a queue on which the process resides). Fig. 3.1.2 shows the different states of processes.

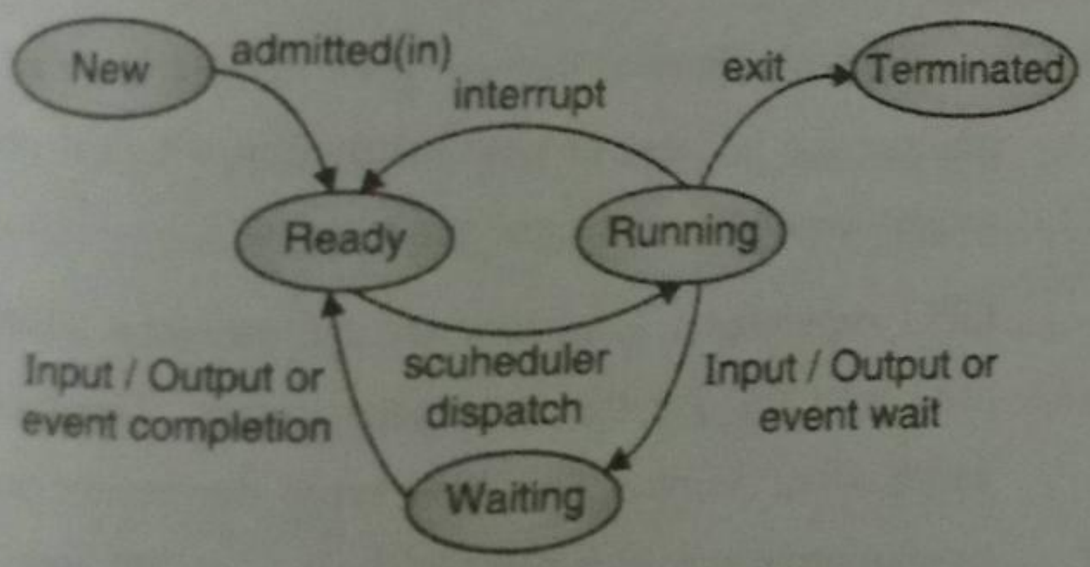


Fig. 3.1.2 : Diagram of process state

- Lets understand these processes :

1. **New :** The process is being created. Here the process is created.
2. **Running :** The process is currently running and has control of a CPU. i.e. Instructions are being executed one after another.
3. **Waiting :** The process is currently able to run, but must wait until a CPU becomes available or may be I/O device available.
4. **Ready :** The process is in memory and will be assigned to the CPU.
5. **Terminated :** The process has finished execution.

- The names of process states are random, and they are subject to change across operating systems. Some operating systems more keenly define the different states.
- The important thing to understand from this is only one process can run at a time and at the same time other processes may be in ready or waiting state.
- The Fig. 3.1.2 shows the sequence of process states when a new process gets create, it is denoted by new. After that it becomes ready to execute, it may wait if CPU is not available. If CPU available, it directly goes to running state i.e. execution and after that it terminates.
- In short when process gets execute, it changes state as follows :

1. **New :** Process creation
2. **Running :** Instructions' execution.
3. **Waiting :** Process waiting for some event to occur.
4. **Ready :** The process is waiting to be assigned to a processor.
5. **Terminated :** Process execution is over.

3.1.3 Process Control Block (PCB)

► **(MSBTE - W-08, S-10, W-10, S-11, W-12, S-13, W-13, W-14, S-14, S-15, S-16, W-16, S-17, W-17, S-18, W-18)**

Q. 3.1.12 Draw and explain process control block in detail. (Ref. Sec. 3.1.3)

W-08, S-10, W-10, S-11, W-12, S-13, W-13, W-14, S-14, S-15, W-16, W-18, 4 Marks

Q. 3.1.13 With neat diagram describe use of Process Control Block (PCB). (Ref. Sec. 3.1.3)

S-16, 4 Marks

Q. 3.1.14 Describe Process Control Block (PCB) with suitable diagram. (Ref. Sec. 3.1.3)

S-17, W-17, 4 Marks

Q. 3.1.15 Explain working of CPU switch from process to process with diagram. (Ref. Sec. 3.1.3)

S-18, 6 Marks

- Operating system executes programs as different processes. As processes runs it is necessary to keep certain details related to processes.
- If the OS supports multiprogramming, then it is necessary to keep track of all the processes. For each process, its process control block PCB is used to track the process's execution status, including the following :

Fig. 3.1.3 shows different parts of process control block. Let's discuss each section one by one. Before that just observe the Fig. 3.1.4, which shows the CPU, switch from one process to another process.

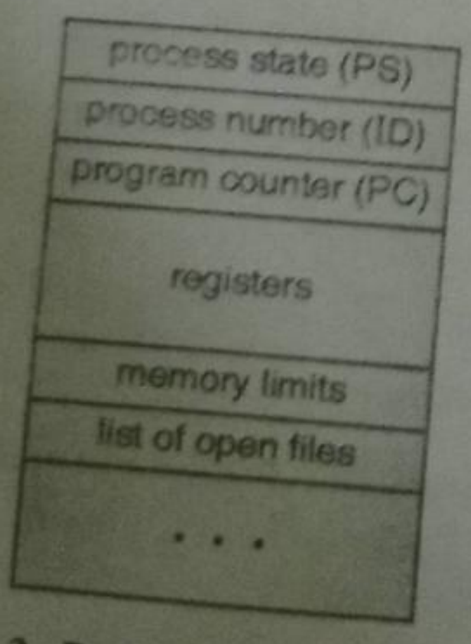


Fig. 3.1.3 : Process control block (PCB)

Fig. 3.1.4 shows two processes as P_1 and P_2. The situation could be described as when P_1 runs at that time P_2 is seems to be idle and when P_2 is running P_1 is idle. This shows how CPU switches from P_1 to P_2.

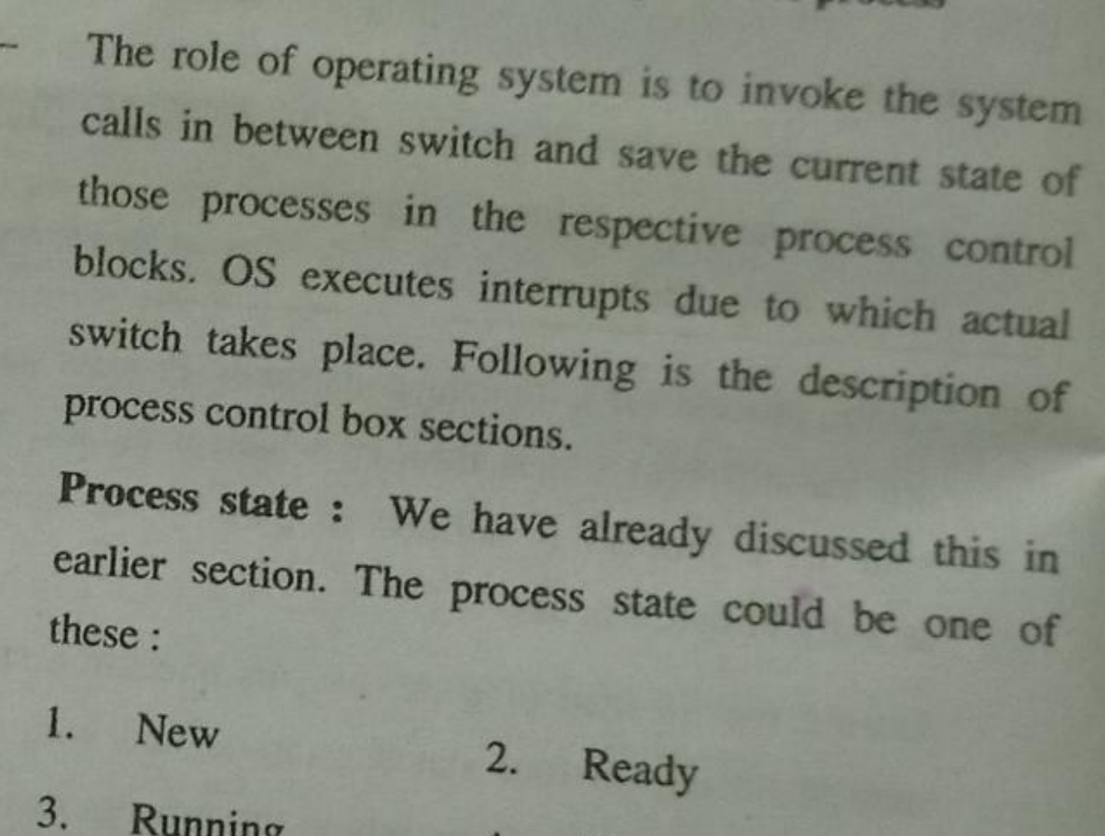


Fig. 3.1.4 : CPU switch from process to process

- The role of operating system is to invoke the system calls in between switch and save the current state of those processes in the respective process control blocks. OS executes interrupts due to which actual switch takes place. Following is the description of process control box sections.
- **Process state :** We have already discussed this in earlier section. The process state could be one of these :

1. New
2. Ready
3. Running
4. Waiting
5. Terminated etc.

- **Program counter :** The program counter is used to indicate the address of next instruction which is going to get execute for the same process.
- **CPU registers :** Registers are the memory locations used by CPU for the temporary usages. The register varies from number and their types depending on the type of architecture it has. The types of CPU registers are as follows :

1. Accumulators
2. Index registers

3. Stack pointers
4. General purpose registers
5. Condition code information registers.

When interrupt occurs it is necessary to store state information with the program counter. It is useful for process to continue what it was suppose to do.

- **CPU scheduling information :** The CPU scheduling is done for smooth and preauthorized execution. This information describes what type of CPU scheduling is done to execute the processes. It even stores the information of schedule pointer for scheduling queues and other scheduling parameters.
- **Memory management information :** Memory management information includes value of base and limit of registers, the page table, segment table, etc. These tables are memory management related information. This information varies from one operating system to other operating system.
- **Accounting information :** Accounting information includes the following information :
  1. Amount of CPU and real time used
  2. Time limits
  3. Accounts number
  4. Job or process number etc.
- **I/O status information :** I/O information includes list of different I/O devices allocated to different processes. It also shows information of open files etc.
- In short process control block is collection of information related to a process, which changes time to time.
- Information associated with each process :
  1. Process state
  2. Program counter
  3. CPU registers
  4. CPU scheduling information
  5. Memory-management information
  6. Accounting information
  7. I/O status information

3.2 Process Scheduling

- The main objective of operating system is multiprogramming. For multiprogramming it is necessary to run multiple programs at all times. This utilizes CPU and I/O device utilization.
- The main aim of time sharing operating system is to keep CPU switching among multiple devices. To meet these both objectives of multiprogramming and time sharing, the processes should scheduled according to their priority.
- The processes are arranged in queue. Each process is associated with its respective process control block. The process scheduler selects an available process from set of available queues for execution.
- For a single processor, there will be only one process running at the given time. If there are more processes, they will have to wait till the CPU gets free and can be rescheduled.

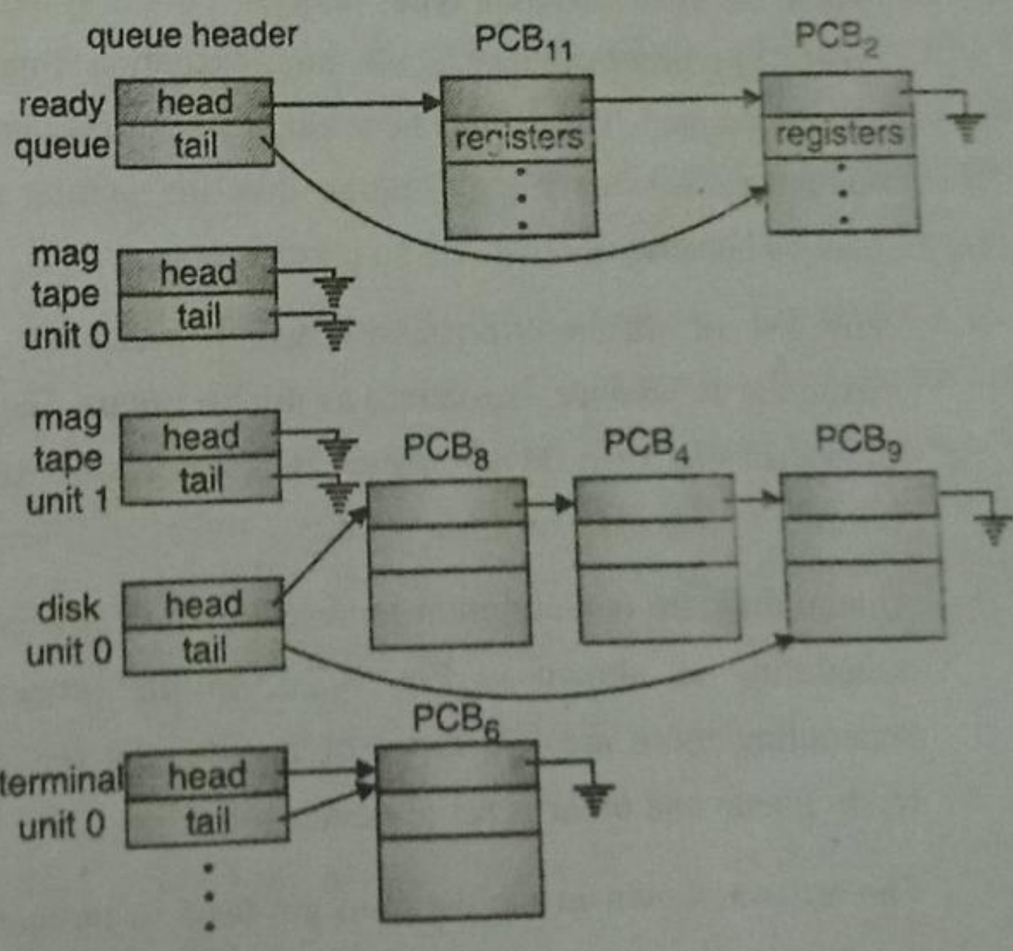


Fig. 3.2.1 : Ready queue and various I/O device queues

3.2.1 Scheduling Queues

▶ **(MSBTE - W-10, W-11, W-12, S-12, W-13, S-14, W-14, S-16)**

Q. 3.2.1 Describe the term : Scheduling Queues (Ref. Sec. 3.2.1)

W-10, W-11, W-12, S-12, S-14 W-14, S-16, 4 Marks

Q. 3.2.2 Explain multilevel queue scheduler with suitable example. (Ref. Sec. 3.2.1) **W-13**

- In recent section we have seen that when a process enters in the system they are included in the job queues. Just observe the queue shown in Fig. 3.2.1.
- The queue is called as **ready queue** when the processes in the queue are residing or stored in the main memory and they are ready and waiting for CPU to get execute. These queues are stored as link lists.
- From the Fig. 3.2.1 we can say that a ready queue header pointer contains pointers to the first and final process block control in the list. Each process block contains a pointer field that points to the next PCB in the ready queue.
- There are again different types of queues in the system. While one process gets execute, after execution it may get interrupted. This is may be because it requires input /output devices or it may require disk for sharing or may be because of other reason it waits.
- The list of different processes, which waits for a particular I/O device, is referred as **device queue.** Each device has its own device queue, which describes the number of processes waiting for it.
- Queue diagram is a common representation of process scheduling as shown in Fig. 3.2.2. In the process scheduling there are two types of queues, the one is ready queue and other is set of device queues.
- The arrows shown in the diagram are used to indicate the direction or flow of the process in the system. The rectangle denotes the queues and circle represents the resources, which serves the queues.

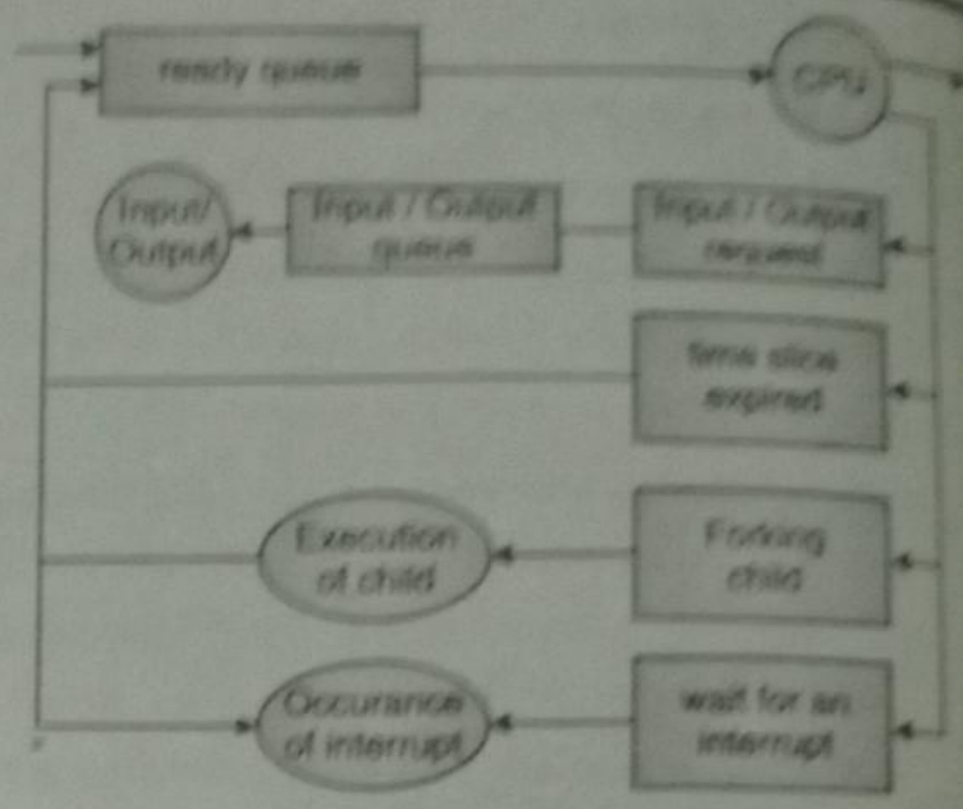


Fig. 3.2.2 : Representation of process scheduling

- Initially a new process is assigned for the execution or it is dispatched. When a process is allocated to the CPU, many situations or conditions could occur at a time :
  1. The process could issue an I/O request and then it can be placed in an I/O queue.
  2. The process could create a new sub process and wait for the sub process's termination.
  3. The process could be removed forcibly from the CPU, as a result of an interrupt, and can be put back in the ready queue.
- In case one and two it is observed that the process eventually switches from the waiting state to the ready state and is then put back in the ready queue.
- A process continues this cycle till it gets terminate. This is the time when process is removed from all queues and has its PCB (Process Block Control) and resources de-allocated.
- **Job queue :** Job queue is collection or set of all processes in the system present at a particular moment.
- **Ready queue :** Ready queue is collection of all processes that are residing in main memory. Also these processes are ready and are waiting for execution
- **Device queues :** Device queue is set or collection of processes waiting for input/output (I/O) device.

3.2.2 Schedulers

▶ **(MSBTE - S-10, W-10, S-11, W-11, S-12, W-13, W-14, S-15, S-16, S-17)**

Q. 3.2.3 State the meaning of the scheduler. Explain any one type of scheduler in detail. (Ref. Sec. 3.2.2) **S-10, W-10, S-12, W-13, W-14, S-15, S-16, 4 Marks**

Q. 3.2.4 Draw types of schedulers. State and explain any one scheduler used in scheduling. (Ref. Sec. 3.2.2) **S-11, W-11, 4 Marks**

Q. 3.2.5 What are I/O bound processes? What are CPU bound processes ? (Ref. Sec. 3.2.2) **W-13, 4 Marks**

Q. 3.2.6 Describe the following : Schedulers. (Ref. Sec. 3.2.2) **S-17, 2 Marks**

- Throughout the lifetime of process it migrates or travels to the various scheduling queues. An operating system should select processes for scheduling purposes from these queues in some manner or style. The selection procedure is done by **schedulers**.
- In a batch system there can be many processes which can be executed immediately. In batch systems if processes are more then they are spooled to the mass storage devices like disks in which they are kept for the later execution. The scheduler which picks up job from this pool and loads into main memory for execution is called as **long term scheduler or job scheduler**.
- There is another scheduler which selects the jobs or processes which are ready to execute from this pool and allocates the CPU to one of them is called **short term scheduler or CPU scheduler**.
- The main difference in long term scheduler and short term scheduler is a frequency of execution. Frequency of execution means the number of time process gets execute. The short term scheduler selects a process which is ready to execute. A process may execute for a very less time, may be in few milliseconds before waiting for an I/O request.
- The short term scheduler generally executes at least once every 100 milliseconds. Due to the short time between executions, short term schedulers are fast. E.g. if it takes 10 milliseconds to decide to execute a process for 100 milliseconds, then we can do following calculation.
- 10 / (100 + 10) = 9. This indicates that 9% of CPU is used only for scheduling work. This time is actually wastage of CPU time.
- The long-term scheduler's frequency is less than shot term one. The reason behind this is that, it takes time in minutes for the long-term scheduler to collect all the processes and load them into main memory. Due to this long scheduler's speed is very less as compare to short-term scheduler.
- The main aspect of long term scheduler is that it controls the degree of multiprogramming. The degree of multiprogramming indicates the number of processes in memory. If the degree of multiprogramming is constant then the average rate of process creation should be equal to the average departure rate of processes leaving the system.
- There is one advantage for long scheduler that due to big interval between executions, this scheduler gets sufficient time to decide which process should be selected for execution. It is necessary that long term scheduler should carefully select process for execution.
- There are two types of processes known as **I/O bound process** and **CPU bound process.** The process which spends more time in I/O operation than computation (time spend with CPU) is **I/O bound process**. The process which spends more time in computations or with CPU and very rarely with the I/O devices is called as **CPU bound process**.
- It is important that long term scheduler selects a good mixture of CPU bound and I/O bound processes. Due to the good combination of CPU bound and I/O bound the system will be balance and no queue will be empty. This is useful for even short term scheduler to select proper process for execution.

In some operating system the long term scheduler may not be present or minimal. The example of this is Windows or UNIX operating system where long term scheduler is not present. In these systems every new process is kept in the main memory for execution. It affects the stability of the system.

In some time-sharing systems an additional scheduling level may get add. This level is nothing but **medium term scheduler**. The main purpose of medium term scheduler is that it removes the process from main memory and again reloads afterwards when required. It is shown in Fig. 3.2.3.

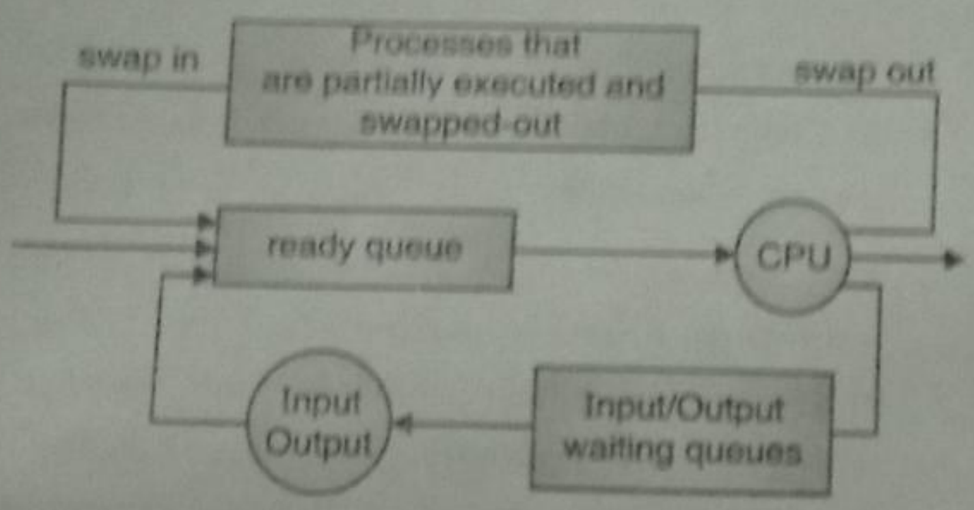


Fig. 3.2.3 : Addition of medium term scheduling

This process is called as swapping. It is memory management technique. Process could be swap in memory and swap out from memory as per the requirement and availability of the memory. This is helpful to improve the performance of the system. It can maintain the multiprogramming degree of operating system.

In short :

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. ***I/O-bound process :*** Spends more time doing I/O than computations, many short CPU bursts.
4. **CPU-bound process :** Spends more time doing computations; few very long CPU bursts.
5. **Medium-term scheduler :** An additional level or queue scheduling used to remove process from memory and reload it again if required.

3.2.2.1 Difference between Short Term, Medium and Long Term Scheduling

► **(MSBTE - S-09, W-12, S-14, S-16, W-16, W-17, S-18)**

Q. 3.2.7 Write difference between short term, medium and long term scheduling. (Ref. Sec. 3.2.2.1) **S-09, W-12, S-14, W-16, S-18, 4 Marks**

Q. 3.2.8 Differentiate between long term scheduler and short term scheduler on basis of :

(i) Selection of job

(ii) Frequency of execution

(iii) Speed

(iv) Accessing which part of system.

(Ref. Sec. 3.2.2.1) **S-16, 4 Marks**

Q. 3.2.9 Differentiate between short term and long term scheduler. (Ref. Sec. 3.2.2.1) **W-17, 4 Marks**

Difference among short term, medium and long term scheduling is :

Sr. No.	Short-term Scheduling	Long-term Scheduling	Medium-term scheduling
1.	Scheduler which selects the jobs or processes which are ready to execute from the ready pool and allocates the CPU to one of them is called short term scheduler or CPU scheduler.	The scheduler which picks up job from pool and loads into main memory for execution is called as long term scheduler or job scheduler	The medium term scheduler is one that removes the process from main memory and again reloads afterwards whenever required.

Sr. No.	Short-term Scheduling	Long-term Scheduling	Medium-term scheduling
2.	Frequency of execution is high. May be in some milliseconds	Frequency of execution in few minutes.	Execution frequency is medium.
3.	Short term schedulers work fast.	Long term schedulers are slow as compare to short term schedulers	Medium term schedulers are called whenever required
4.	Deals with CPU	Deals with main memory for loading process.	Deals with main memory for removing processes and reloading whenever required

☞ ***Difference* between long term scheduler and short term scheduler**

Criteria	long term scheduler	short term scheduler
Selection of job	Job is selected from pool and gets load into memory for getting executed.	Process is selected from ready queue. Processes in the ready queue are ready to execute.
Frequency of execution	Frequency of execution is very much less. Gets executed when queue of ready processes have space for accommodating new process.	Gets executed frequently for selecting a new process for the CPU. Frequency could be 100 milliseconds
Speed	Speed is less as compare to short term scheduler.	Speed is fast
Accessing which part of system	It accesses pool of jobs and ready queue .	This accesses CPU and ready queue.

3.2.3 Context Switch

▶ **(MSBTE - W-10, S-12, W-12, S-13, W-13, S-14, S-15, W-15, S-16, W-16, S-17, W-17, W-18)**

Q. 3.2.10 Describe the term : Context Switch. (Ref. Sec. 3.2.3) **W-10, S-12, W-12, S-13, W-13, S-14, W-16, S-17, 4 Marks**

Q. 3.2.11 Explain context switch with suitable example. (Ref. Sec. 3.2.3) **S-15, 4 Marks**

Q. 3.2.12 Explain the concept of context switching. (Ref. Sec. 3.2.3) **W-15, W-16, 4 Marks**

Q. 3.2.13 Describe how context switch is executed by operating system. (Ref. Sec. 3.2.3) **S-16, 4 Marks**

Q. 3.2.14 Explain context switch with help of diagram. (Ref. Sec. 3.2.3) **W-17, 4 Marks**

Q. 3.2.15 How context switching is done? (Ref. Sec. 3.2.3) **W-18, 4 Marks**

Q. 3.2.16 List any two condition leading to process suspension. (Ref. Sec. 3.2.3) **W-18, 2 Marks**

- The occurrence of the event is marked by an interrupt from either the hardware or software. Hardware may trigger an interrupt at any time by sending signal to the CPU, by the way of system bus. Software can also trigger an interrupt by a special means and that is called as system call.
- Let's discuss with example where we assume CPU is busy serving users process. Interrupts cause an operating system to change the control of CPU from a current running process to kernel mode or even user mode. Now when CPU is switched from the current process to other task it is necessary to store the context i.e. the background information of the running process before really switching it to kernel.
- The context is generally stored in the form of process control blocks. CPU can restore these details when process is suspended and resumed again when the switching task gets over. PCB includes the value of CPU registers, the process state and memory management information.

- Commonly the **state save** is done when CPU switches to kernel and then **state resume** is done to resume the process.
- Switching to the different processes requires two operations i.e. the one is state save for running process and state restore for another process. This task is referred as a **context switch.**
- Kernel stores the context of the old process in its PCB when context switch occurs and loads the saved context of another process, which is scheduled to run.
- The context switching time is an overhead time. During switching time system does not do any useful work. The switching time varies from machine to machine. This time is highly dependant on the type of hardware used. It mainly dependent of memory speed, the number of registers etc. Switching time of CPU is generally in some milliseconds.
- Some processors may provide multiple CPU sets of registers. In this case context switch is required to just change its pointer to the current register set. If there are many active processes, which are more than register sets available, then system resorts to copying register data to and from the memory.
- If the operating system is more complicated, more work has to be done. In that case to put the extra data to and from the memory, different memory management techniques are used.

In short :

1. When CPU switches from one process to another process, the system must save the state of the old process and load the saved state when process has to be executed again after switching.
2. Context-switch time is overhead; the system does no useful work while switching.
3. Switching time is dependent on hardware support.

- Following are two conditions when process can be suspended.
  - When its execution priority is lesser than other system process.
  - Resources allocated are not sufficient or appropriate.

3.3 Operations on Processes

▶ **(S-09, S-11, S-12, S-13, W-13, S-14, W-14, S-15, W-16)**

Q. 3.3.1 Describe various operations of process. (Ref. Sec. 3.3) **S-09, S-11, S-12, S-13, W-13, S-14, W-14, S-15, 4 Marks**

Q. 3.3.2 Define process. Describe process creation and termination. (Ref. Sec. 3.3) **W-16, 8 Marks**

- The processes in the system can execute concurrently and different operations can be performed on the processes. Processes can be created and deleted dynamically.
- The system should provide the mechanisms, which are used for the creation and deletion of the processes. Here we are going to focus on two main operations i.e. creation of process and deletion of processes.

3.3.1 Process Creation

▶ **(MSBTE - S-17, W-18)**

Q. 3.3.3 Write use of following system calls.
(1) fork( ) (2) exec( )
(3) abort( ) (4) end( )
(Ref. Sec. 3.3.1) **S-17, 4 Marks**

Q. 3.3.4 Why is process creation necessary? State the role of fork process in the context. (Ref. Sec. 3.3.1) **W-18, 4 Marks**

- As we have discussed in last sections that processes are created by the operating systems to simplify the job of execution of the program by the CPU. One process which is created may create several new processes when it runs. These processes are created by the original process using create process system call.

- The original or main process is called as **parent** process and the new processes are called as **children** of that process. Each of these children processes may create other processes, which forms **tree of processes**.
- In many operating systems, processes are identified by the unique process identifier which is a unique number given to it. It is an integer value assigned to the process. It is referred as **pid**.
- Observe the Fig. 3.3.1 that shows the process creation of a typical Solaris operating system. In the Fig. 3.3.1 shed is the parent process with pid 0 which creates children process init (pid-1), pageout(pid-2), fsflush(pid-3). After creation of init, it creates further processes inetd and dtlogin and so on.
- A process requires certain resources like CPU time, memory, files, I/O devices to complete its tasks. When parent process creates the process it is called as sub process. Operating system may provide required resources to the sub process or it may have to use the resources available to its parent process.
- The parent process may have to partition its resources with its children processes or may be parent process should be able to share the partition resources of the children processes.
- If child processes are restricted to become subset of parent processes then we can avoid children processes to overload the system by creating many sub processes.
- Apart from the logical and physical resources which process share with its children processes, it even shares the input. i.e. the input received to the parent process could be shared by its children.

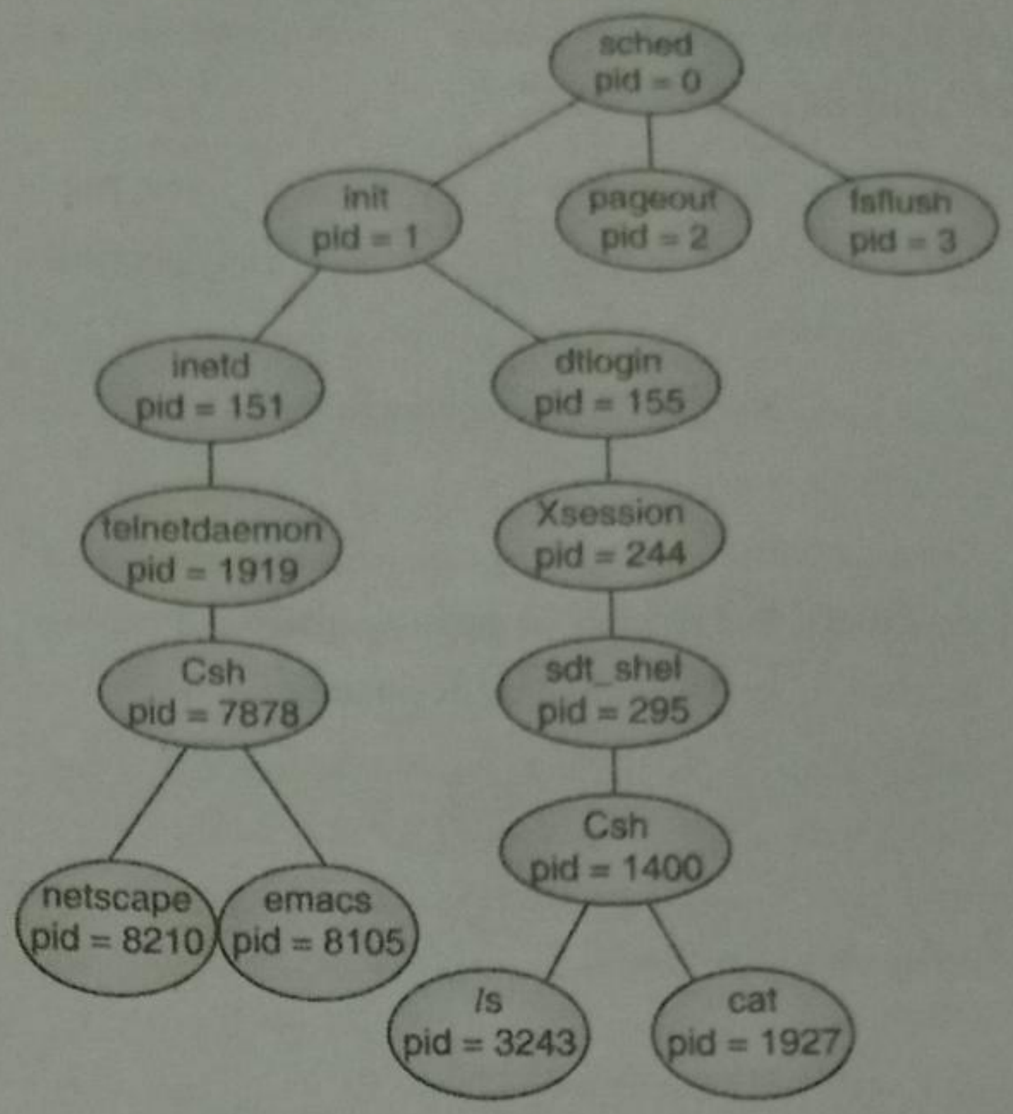


Fig. 3.3.1 : A typical Solaris system

- E.g. considers the process, which is supposed to display the image file on the screen. There are different sub processes or parent processes while doing this operation. The file has to be open first and the contents have to be written. Another thing is the name of display device is required.
- In UNIX both files and devices are treated as files. So the file name which has come as an input to the parent process will be passed on to the child process which will display the actual image on the display device.
- When process creates a new process, two possibilities exists in terms of execution as follows :
  1. The parent continues to execute concurrently with its children.
  2. The parent waits until some or all of its children have terminated.

- Related to address space again there are two possibilities as follows :
  1. The child process is a duplicate of the parent process (It has the same program and data as the parent)
  2. The child process has a new program loaded into it.
- Consider the following program written in the Unix to understand the relation of parent process and children process : C Program Forking Separate Process

```
int main()
{
pid_t pid;
   /* fork another process */
   pid = fork( );
   if (pid < 0) { /* error occurred */
      fprintf(stderr, "Fork Failed");
      exit(-1);
   }
   else if (pid == 0) { /* child process */
      execlp("/bin/ls", "ls", NULL);
   }
   else { /* parent process */
      /* parent will wait for the child to complete */
      wait (NULL);
      printf ("Child Complete");
      exit(0);
   }
}
```

- From the program we come to know that pid is an unique id given to the process. A new process is created by fork( ) system call. A new process is consist of a copy of the address space of the original process. This mechanism is useful which allows parent process to communicate with its children processes.
- Both the parent as well as child processes get executed when fork( ) system call occurs. The main difference is when new process is created as a child the return code is zero but when fork( ) is executed for the parent the return code is nonzero number i.e. child's process code is returned to parent.
- The exec( ) system call is used after fork( ) system call by one or two processes to replace the process's memory space with a new program.
- The exec( ) system call loads a binary file in the memory which destroys the memory space of the process where exec( ) system call is written and then starts executing. In this way two processes can communicate with each other and after doing this they get separated to their ways.
- The parent process can then create more children processes or when children runs and parent is nothing to do then it execute wait( ) system call for itself and gets away from the ready queue of the processes till the child process gets terminated.
- The Fig. 3.3.2 as well as the C program, which we have written, shows that when fork ( ) system call is create a parent and child processes are created.
- Parent process waits till the child process runs and once the child process is terminated using exec( ), it removes the address space of parent process. Then exit( ) system call exits from the child process and parent process will be resumed.

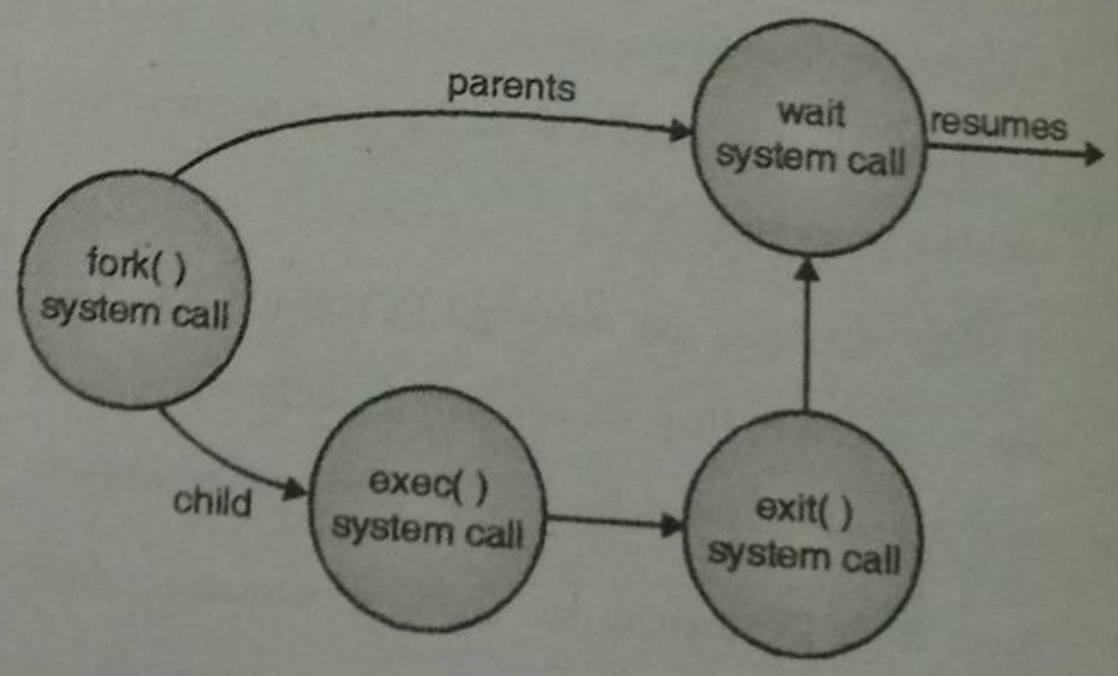


Fig. 3.3.2 : Process creation

- In short :
  - Parent process creates children processes, which, in turn create other processes, forming a tree of processes.

- **Resource sharing**
  1. Parent and children may share all resources.
  2. Children share subset of parent's resources.
  3. Parent and child may not share any resources.
- **Execution**
  1. Parent and children execute concurrently.
  2. Parent waits until children terminate.
- **Address space**
  1. Child is duplicate of parent.
  2. Child may have different programs loaded into it.
- **UNIX examples**
  1. **Fork** system call creates new process.
  2. **Exec** system call used after a **fork** to replace the process' memory space with a new program.

1) fork() - A new process is created by fork() system call
2) exec() - The exec() system call is used after fork() system call by one or two processes to replace the process's memory space with a new program.
3) abort() - Used to abort the process.
4) end() - Used to end the process.

3.3.2 Process Termination

▶ **(MSBTE - S-10, W-17)**

Q. 3.3.5 Explain process termination in detail. (Ref. Sec. 3.3.2) **S-10, 4 Marks**

Q. 3.3.6 Describe how process is terminated. (Ref. Sec. 3.3.2) **W-17, 4 Marks**

- A process is terminated when it finishes by executing an exit( ) system call. When its last statement gets execute the operating system deletes it using exit( ) system call.
- At this point process may return a status value to its parent process via wait( ) system call. This return status value is generally an integer. All the logical and physical resources including open files, I/O buffers, memory etc allocated to the process is de-allocated by an operating system.
- Termination can even occur in different situations. A process can be terminated by another process via appropriate system call. Such a system call can be invoked by only parent of the process that has to be terminated.
- If this rule would not be there, user could have killed job processes of each others. Here parent processes need to know the pid's of its children processes. Thus when a new process is created its identifier is return to its parent process.
- A parent may terminate its children process due to following reasons :
  1. The child has exceeded its usage of some of the resources that it has been allocated.
  2. The task assigned to child process is not required any more.
  3. The parent is exiting, and the operating system does not permit it's a child to continue if its parent terminates.
- Some systems like VMS do not allow child processes exist if parent process is terminated. If parent process terminates, all its children processes should be terminated. This phenomenon is called as **cascading termination.** This is initiated by an operating system.
- In case of UNIX operating systems, the processes are terminated by exit ( ) system call. In this case parent process waits till child process gets execute.
- If parent process terminates then child processes do not terminate. They are given a parent process **init** by operating system. So though parent process terminates children process continues with new parent, which keeps track of children processes.

3.3.3 Operations Performed on Process

Following operations can be performed on the process with the help of system calls.

1. **Ending or aborting process :** This helps in ending or aborting process.
2. **Loading or executing process :** Used for loading and executing process.
3. **Create process or terminate process :** This operation creates or terminates process.
4. **Get process attributes and set process attributes :** To set or get process attributes.

3.4 Inter-Process Communication

▶ **(MSBTE - W-08, W-10, S-11, W-11, S-12, W-12, S-13, W-13, S-14, W-14, S-15, W-15, S-16, W-16, W-17, W-18)**

Q. 3.4.1 Draw and explain inter-process communication model. (Ref. Sec. 3.4)

W-08, W-10, S-11, W-11, S-12, W-12, S-13, W-13, S-14, W-14, S-15, W-15, W-16, W-17, W-18, 8 Marks

Q. 3.4.2 What is inter process communication? (Ref. Sec. 3.4)

S-16, 4 Marks

- Processes executing concurrently in the system may be independent or may be cooperating processes. Independent processes are those, which cannot affect other processes or may not get affected by other processes executing in the system.
- A process, which does not share anything with other process is also called as independent process.
- On other hand the processes, which are cooperating with each other or which gets affect due to other executing processes and affect the other processes are called as cooperating processes. This means the processes, which shares each other's data are called as cooperating processes.
- There are certain reasons due to which cooperative environment is needed :

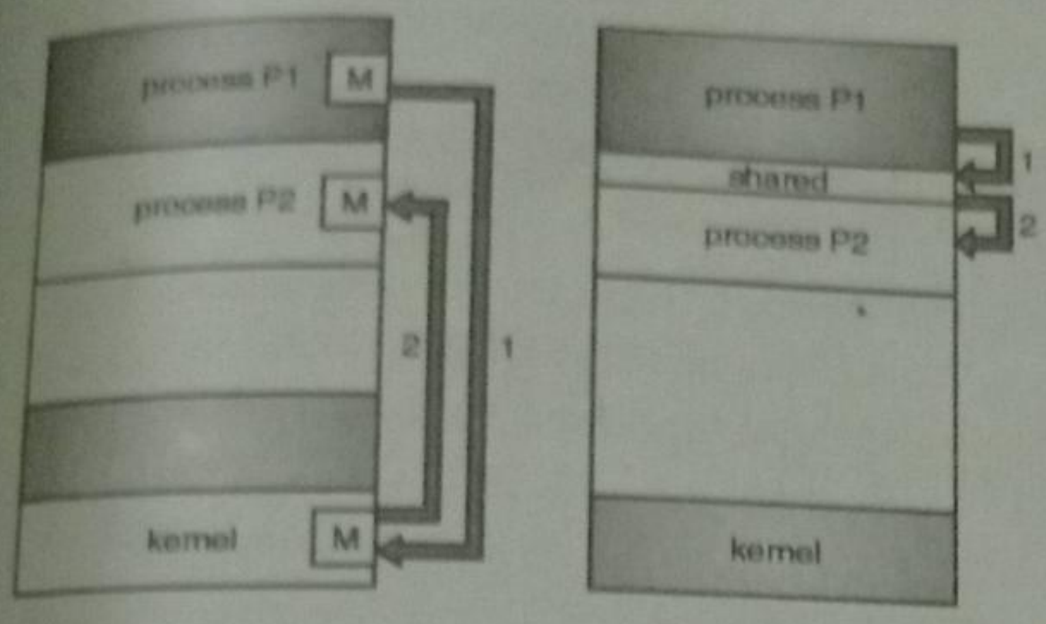
1. **Information sharing :** Many users may be interested in the same piece of data or information e.g. shared files, instances etc. OS should provide an environment to allow concurrent or simultaneous access to such information.
2. **Computation speedup :** It is necessary to run the computation activities as fast as possible. The solution to this is to break the processes or tasks into sub processes. This causes the parallel run of subtasks, which results into speedup of computation. Computation speedup can be achieved only if the computer has multiple processing elements such as CPU's I/O channels.
3. **Modularity :** Modularity divides the system in different modules so that running an operating system becomes easy. Each module performs its own function. This allows different functions to be separated in different processes or threads.
4. **Convenience :** User should be comfortable with the system as it should allow single user to do multiple tasks at the same time. e. g. user may want to do printing, typing , editing and many more operations at a time.

- There should be a mechanism for cooperating processes to communicate with each other. This mechanism is called as **interprocess communication (IPC)**. IPC allows processes to exchange data and information.
- There are two fundamental models which allows interprocess communication as follows:

1. Shared memory
2. Message passing

- In the shared memory model, sharing region is established. From this sharing region cooperating processes exchange data or information. They can read and write data from and to this region.

In message passing model the data or information is exchanged in the form of messages. Refer Fig. 3.4.1 to view both the models.



(a) Message passing (b) Shared memory

Fig. 3.4.1 : Interprocess communication models

- Shared memory as well as message passing models is very common and popular communication models used in an operating system. Message passing models are easy for exchanging small amount of data, as there are no much conflict. Message passing phenomenon is easier than shared memory model for inter-computer communication.
- Shared memory is useful for maximum speed and convenience of communication as it can be done at memory speeds when within a computer. Shared memory models are fast than message passing as message passing model uses system calls, it is time consuming and needs more intervention of kernel.
- Shared memory system calls are required only to establish shared memory regions. Once the shared memory region is established rest operations are routine memory access operations. It does not require any kernel assistance.

In short :

1. **Independent** process cannot affect or do not get affected by the execution of another process.
2. **Cooperating** process can affect or can get affected by the execution of another process.
3. Advantages of process co-operation :
   (i) Information sharing
   (ii) Computation speed-up
   (iii) Modularity
   (iv) Convenience

3.4.1 Shared Memory Systems

▶ (MSBTE - W-09, S-16, S-18)

Q. 3.4.3 Explain the two techniques for the inter process communication. (Ref. Sec. 3.4.1) **W-09, S-16, 8 Marks**

Q. 3.4.4 Explain the working of Inter-process communication considering.
(1) Shared memory
(2) Message passing
(Ref. Sec. 3.4.1) **S-18, 6 Marks**

- Shared memory systems allow processes to share data or information using shared memory region. Shared memory region resides in the address space of the process creating the shared memory segment.
- Other processes, which need to share this shared region should connect their address space with the segment of the memory region.
- Generally operating system tries to avoid one process accessing memory of another process. Shared memory concept could be used when two or more processes are ready to avoid this restriction. They can then read and write data to and from shared memory area.
- The important thing is the form of the data and its location that has to be determined by these processes. This time operating system does not have any control on these processes. Processes should ensure that they do not write at the same location at a time.
- To understand the concept of cooperating processes lets take an example of producer consumer problem. Producer process produces information that is

consumed by consumer process. It is like compiler, which produces an assembly code and assembler consumes it. In turn assembler may generate object code, which is consumed by the loader.

- The producer consumer is a metaphor (symbol) which describes the paradigm like client-server. We assume that producer is the server and consumer is client. The practical example is web server, which allows HTML files to run and produces HTML files and images. These are consumed by the client web browser requesting the resource.
- Producer consumer can be solved using shared memory option. For concurrent execution of producer processes and consumer processes a system should have buffer, which will be filled by producer processes and emptied by consumer processes. This buffer will reside in the shared memory area which is accessible to both producer and consumer.
- Here producer produces the item and consumer consumes the item. Both producer and consumer should be synchronized so that consumer will not try to consume the item that is not produced by the producer.
- The buffer used could be of two types as follows :
  1. Unbounded buffer
  2. Bounded buffer
- The **unbounded buffer** does not place any limit on the size of the buffer. When unbounded buffer is used, consumer has to wait for the new item to be produced but producer can produce items without waiting.
- The **bounded buffer** assumes a fixed buffer size. So in this type consumer should wait if the buffer is empty and producer has to wait if the buffer is full.
- Let's have a look in case of bounded buffer where buffer size is fixed.

```
typedef struct {
    . . .
} item;
item buffer[BUFFER_SIZE];
int in = 0;
int out = 0;
```

- From the above code we understand the shared buffer is implemented as a circular array with two logical points called **in** and **out. 'in'** points to the next free position in the buffer.
- **Out** variable points to the first position in the buffer. When **in=out** the buffer is empty. The buffer is full if the following condition is true. **((in+1) % BUFFER_SIZE)) = out**
- Producer code

```
while(true) {
  /* Produce an item */
    while (((in = (in + 1) % BUFFER SIZE count) ==
out)
      ;  /* do nothing -- no free buffers */
    buffer[in] = nextproduced;
    in = (in + 1) % BUFFER SIZE;
  }
```

Consumer code

```
while (true) {
    while (in == out)
        ; // do nothing -- nothing to consume

    // remove an item from the buffer
    nextConsumed = buffer[out];
    out = (out + 1) % BUFFER SIZE;
return item;
}
```

- The code described above is producer's and consumer's code. Producer uses local variable next produced in which newly produced item will be stored. Next consumed is a local variable in consumers code, which shows the item consumed. There could be a issue if consumer as well as producer tries to share the shared buffer at a time.
- In short :
  1. Paradigm for cooperating processes, producer process produces information that is consumed by a consumer process.

2. Unbounded-buffer places no practical limit on the size of the buffer.
3. Bounded-buffer assumes that there is a fixed buffer size.

3.4.2 Message Passing Systems

▶ (MSBTE - W-09, S-17)

Q. 3.4.5 Explain the two techniques for the inter process communication.
(Ref. Sec. 3.4.2) **W-09, 8 Marks**

Q. 3.4.6 With neat diagram, explain Message passing system. Also describe the following :
(i) Naming
(ii) Synchronization
(iii) Buffering
(Ref. Sec. 3.4.2) **S-17, 8 Marks**

- Just in the last section we have seen that cooperating processes uses shared memory technique to communicate with each other. In this technique processes uses shared memory region. Another way is message passing where individual processes passes messages to each other as a means of communication.
- In message passing mechanism processes can just pass the messages to each other without sharing of their address spaces. This type of communication is mainly used in distributed environments.
- In distributed computing the processes may reside even on different computers connected by the network. The best example to illustrate message passing system is the chat which takes place on the World Wide Web where multiple users chat from different locations. They exchange messages among them from different nodes or locations.
- There are at least two facilities called **send**(to send message) and **receive** (to receive message). Messages send by the processes are of either fixed or variable size. System level implementation is straightforward if the messages sent are of the fixed length.
- For variable length the complexity is more so system level implementation becomes complicated.
- Let's consider that P and Q are two processes, which want to communicate with each other. There should be a communication link, which will facilitate both processes to send and receive the messages. Here our discussion is restricted to logical link and not the physical link. This link should be useful to execute send ( ) or receive ( ) operations.
- There are certain methods to above said operations :
  1. Direct or indirect communication
  2. Synchronous or asynchronous communication.
  3. Automatic or explicit buffering.
- There are certain issues which are related to this method like :
  1. Naming
  2. Synchronization
  3. Buffering

3.4.2.1 Naming

- Processes which wish to communicate with each other need to know each other with the name for identification. There are two types of communications as follows :
  1. Direct communication
  2. Indirect communication
- In direct communication each process that want to communicate must be explicitly use name for the sender as well as receiver or recipient while communication. In this type the send ( ) and receive ( ) primitives are defined as follows :

Send (P, message)- Send message to process P

Receive (Q, message)-Receive a message from process Q.

- The communication in direct communication has following properties :
  1. A link is established automatically between every pair of process that wants to communicate. The process needs to know only each other's identity to communicate.
  2. A link is associated with exactly two processes.
  3. Between each pair of processes, there exists exactly one link.
- The recent scheme exhibits **symmetry** in addressing of both sender process and receiver process. Another slightly different method is **asymmetry** which allows only senders to give name to the receiver.
- Receiver does not need to address sender. In this case primitives like send() and receive() are as follows :

  Send (p, message) : Send a message to process P.

  Receive (id , message) : Receive a message from any process; the variable id is set to the name of the process with which communication has taken place.
- Disadvantage of both the schemes of direct communication is limited modularity of the resulting process definitions. These hard coding techniques of identifiers are not desirable than techniques involving indirection. We are now going to see indirect communication in detail.
- In an indirect communication the messages could be sent or received from mailboxes or ports. A mailbox can be viewed as an object in which messages can be kept or even removed. Each mailbox is associated with the unique number.
- E.g. POSIX message queues use an integer values to number or identify mailbox. Here the communication among the processes takes place with the mailbox numbers.
- Two processes can communicate with each other if they have shared mailbox by both sender and receiver processes. The send () and receive () primitives could be defined as follows :

  Send (A, message) : This sends a message from mailbox A.

  Receive (A, message) : This is for receiving message from mailbox A.
- In indirect type of communication the properties or attributes related to communication link are as listed below :
  1. If both member processes share same mailbox then only a communication link can be established between them.
  2. There could be association of more than two processes with the communication link.
  3. One mailbox may have multiple links.
- Imagine if there are three processes P_1, P_2, P_3 and all of them share mailbox A. If process P_1 sends messages to mailbox A and P_2 and P_3 both tries to receive () from A then who will receive the message depends on the following methods we choose :
  1. Allow a link to get associated with maximum two processes.
  2. Permit a single process at a time to execute receive () type of operation.
  3. Let the system choose randomly which process will receive the message. System can use an algorithm to make the choice of receiver. The system may recognize the receiver to the sender.
- A mailbox can be owned by process or even by an operating system. If process owns a mailbox that means mailbox is attached to address space of process.
- This process is called as owner of mailbox who has right of receiving mails from other process. The other category is user where other processes are user who can only send messages to this mailbox and owner can only accesses these messages.
- As each mailbox has only owner, when the owner process is terminated then mailbox gets disappear. User processes are smart enough that they do not send messages to the mailboxes, which do not exist.

- On the other hand the mailbox, which is own by operating system has its own existence. It is independent and not attached with any process. The operating system should support the following mechanisms for the processes :

1. Create a new mailbox.
2. Send and receive messages through the mailbox.
3. Delete a mailbox.

- The process which creates the mailbox becomes the owner of that mailbox which can by default receive the messages. The ownership and the privileges of the owner process can be transferred by the system to other process.

3.4.2.2 Synchronization

▶ (MSBTE - S-18)

Q. 3.4.7 Define synchronization Explain
(i) Blocking
(ii) Non Blocking in message passing
(Ref. Sec. 3.4.2.2) S-18. 4 Marks

- Communication between processes takes place through the system calls. To send () and receive() primitives a special design is required for the implementation of these primitives.
- Message passing is of two types as follows :
  (i) Blocking
  (ii) Nonblocking
- These are also known as synchronous and asynchronous communication. Let's see the following operations:

1. **Blocking send :** This is for blocking the sending process or operation till the message is received by the receiving process or mailbox.
2. **Nonblock sending :** Nonblock sending allows sending the message and resuming its operation.
3. **Blocking receive :** This blocks the receiver till the message is available to it.
4. **Nonblocking receive :** The receiver can receive a valid message or may receive null.

- There could be different combinations of send ( ) and receive ( ) are feasible. when send ( ) and receive( ) both are blocking, association between sender and receiver is possible.
- The producer and consumer problem here becomes small when we use blocking send ( ) and receive ( ) statements. The producer just invokes blocking send ( ) call and wait until the message is delivered to receiver or mailbox and receive ( ) or the mailbox. Similarly when consumer invokes receive ( ), it blocks till the message is available to it.

3.4.2.3 Buffering

- The communication could be direct or indirect. The message exchanged by the communicating processes resides or are stored in a temporary queue. There are three ways to implement this queue.

1. **Zero capacity :** In zero capacity queue the size or maximum length of the queue is zero. Thus in this case the link cannot have any messages waiting in it. The sender should be blocked until the recipient receives the message.
2. **Bounded capacity :** Bounded capacity means it has fixed length or fixed size. If the size of queue is **n** then number of messages can reside in this queue are **n**. If the queue is not full a new message can be send to it and sender can continue its execution without waiting. The link capacity is finite in this case. If the queue is full the sender must wait till space gets available in queue.
3. **Unbounded capacity :** In unbounded capacity the length or size of the queue is infinite. So any number of messages can reside on the queue. In this case sender never waits.

- The zero capacity is sometimes called as message system without buffer. The other type is known as system with automatic buffering.

3.4.3 Critical Sections (Critical Region)

▶ (MSBTE - S-17)

Q. 3.4.8 Describe the critical-section problem. (Ref. Sec. 3.4.3) **S-17, 4 Marks**

- Race condition occurs when two or more than two processes try to read or write the shared data and the final result is dependent of process that runs precisely.
- This condition occurs because of sharing file or same memory variables. To avoid this type of condition, we have to ensure that if one process is using some files or variable the other process should not be allowed to share it.
- In other words **Mutual Exclusion** is way to ensure that if one process is trying to use some shared files and variable then other process should be excluded or should not be allowed to do the same operation.
- Sometimes it may happen that process accesses shared memory or file, or may be doing some other critical things which may invite race condition. That part of a program where the shared memory is accessed is known as the **critical section** or **critical region**.
- To avoid the race condition, we have to see to it that no two processes will be there at a time in the critical region.
- To find out the solution on race condition, only mutual exclusion cannot be sufficient. Following are the four conditions important while solving the problem.

1. Two processes should not be present in their critical region at a time.
2. Assumptions regarding parameters like speeds or number CPUs should not be made.
3. The process that is present outside critical region cannot block other processes.
4. While process is trying to make an entry in its critical region, it should not face the problem like wait forever.

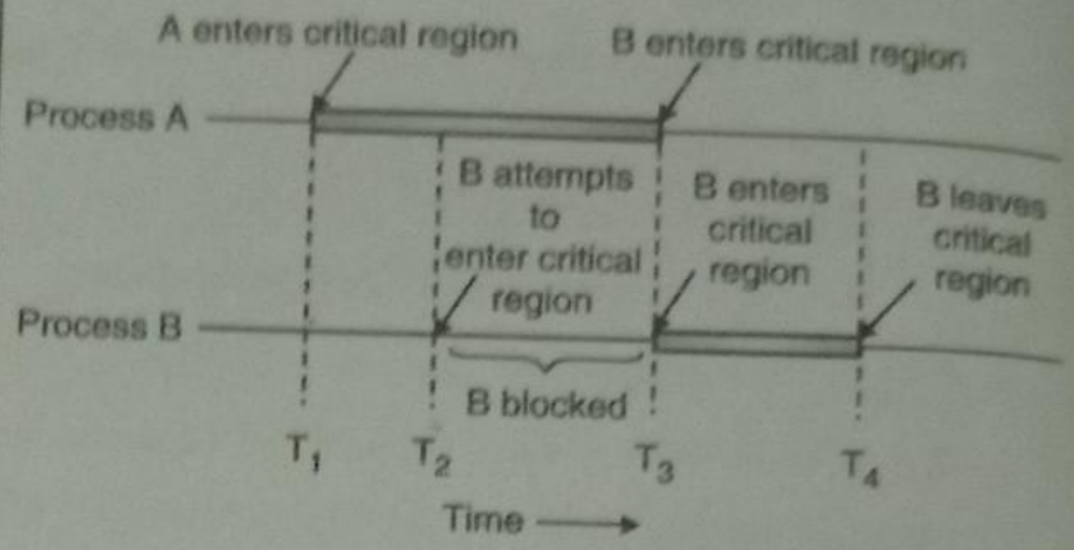


Fig. 3.4.2 : Critical region and Mutual exclusion

3.4.4 Critical Region and Mutual Exclusion

- As shown in the Fig. 3.4.2, there are two processes A and B. T_1, T_2, T_3, T_4 are time intervals. Process A enters in its critical region at time interval T_1.
- After some time, process B tries to enter in its critical region. Since process A is still in its critical region, between T_2 and T_3 process B will be blocked.
- Once the process A moves out from critical region, process B will enter in the critical region. It will remain there in the time interval T_3 and T_4. At T_4 it will leave the critical region.

3.4.5 Mutual Exclusion

▶ (MSBTE - S-10, S-14)

Q. 3.4.9 Explain the concept of mutual exclusion in detail. (Ref. Sec. 3.4.5) **S-10, S-14, 4 Marks**

- Mutual exclusion can be achieved by different ways or methods so that while one process is busy with shared memory operations in its critical region, no other process will enter its critical region and create problems.

- Following proposals can be used for that purpose :

1. Disabling interrupts	2. Lock variables
3. Strict alteration	4. Peterson's solution

3.4.5.1 Disabling Interrupts

- This is the simplest solution. When process enters in critical region, interrupts are disabled by it and once it leaves its critical region, it again enables interrupts.
- Because of either clock or interrupt, CPU makes switching from one process to other. When interrupts are disabled, CPU switching will not happen.
- Here process can complete its shared memory operations without any fear and bothering about other process to enter.
- Important thing is that we have to give power to a process to disable interrupt. That will be unwise because if the processes disables the interrupts and do not make them enable then many problems can occur. Only kernels can easily enable and disable the interrupts. So it is not a good approach for mutual exclusion.

3.4.5.2 Lock Variables

- This could be one of the software solutions. Here we can declare two variables one single and shared (lock) initialized to 0. If the process finds that lock is 0 then it enters into critical region and makes lock variable 1. And just before leaving it is again set to 0.
- If another process enters then it can see that lock variable is 1 then it waits till it becomes 0. There may be another case like this process keep waiting and if the lock variable becomes 0 then one more process may enters and tries to make that lock variable to 1 before waiting process makes it.
- So this approach is also not that effective because this solution will bring race condition.

3.4.5.3 Strict Alteration

- In strict alteration, a variable named turn is used to run two processes alternately. For example first process 1 runs and then process 2 runs. After process 2 again process 1 runs and so on.
- Following code is used by initializing turn=0.

```
while (TRUE) {
    while (turn !=0)  /* loop */;
    critical_region();
    turn = 1;
    noncritical_region()
}
```

(a) Process 0

```
while (TRUE) {
    while (turn!=1)  /*
loop */
    critical_region();
    turn = 0;
    noncritical_region();
}
```

(b) Process 1

- This method wastes CPU time. We need to minimize busy waiting. This solution can be used only when the waiting period is expected to be short. This approach is not a good solution.

3.4.5.4 Peterson's Solution

- The previous solution solves the problem of one process blocking another process while that process is outside its critical section. It is not a good mutual exclusion.
- Peterson's Solution is an efficient solution with busy waiting. It designs the procedures for process to enter and leave critical region.

3.4.6 Semaphores

▶ (MSBTE - S-16)

Q. 3.4.10 Describe how semaphores are useful for solving problems of interposes communication. (Ref. Sec. 3.4.6) **S-16, 6 Marks**

- There are certain hardware based solutions to the critical sections using Test And Set and Swap instructions. But they are very complicated for application programmers to implement.

- To find out solution to this problem, we can make use of synchronization tool known as semaphore.
- Semaphore S is integer variable. Except from its utilization, its value can be accessed with the help of only two atomic operations signal () and wait ().
- Wait() operation was originally referred as P and signal() operation was called as V.
- The definition of wait() is as below :

```
wait(S)
{
while S <= 0;        No operation
S--;                 // S is semaphore

}
```

The definition of signal () is :

```
signal(S)
{
S + +;       // S is semaphore
}
```

- Only one operation i.e. either wait () or signal () can access the value of semaphore and modify it. No two operations can access the value of semaphore at a time.
- Semaphores could be binary semaphores and counting semaphores. Operating system can distinguish them. Binary semaphores can be some time called as mutex locks for they provide locks for mutual exclusion.
- Binary semaphores can be used in critical section when multiple processes are involved.
- Counting semaphores can be used for controlling the access to the resources whose instances are finite number. Every process that needs resource performs the wait () operation on semaphore and decrements it.
- When process releases the resource it uses signal () operation and thus semaphore is incremented. When semaphore's value is zero, it indicated that all resources are used. At this stage if any process needs any resource, that processes will be blocked till the semaphore's value becomes greater than one.

3.5 Threads

▶ **(MSBTE - S-09, W-09, S-10, W-11, S-12, W-12, S-13, S-14, W-15, S-16, W-16)**

Q. 3.5.1 Write in brief about threading. (Ref. Sec. 3.5) **S-09, W-09, S-10, W-11, S-12, W-12, S-13, S-14, S-16, W-16, 6 Marks**

Q. 3.5.2 What is thread ? (Ref. Sec. 3.5) **W-15, 2 Marks**

- A **thread** is a basic unit of CPU utilization. Thread is associated with thread ID, a program counter, a register set, and a stack. Thread shares its code section, data section, and other operating system resources such as open files and signals with other threads which belong to the same type of process.
- Generally a traditional process has single thread but process can be multithreaded also. If the system is multithreaded control system then it is very powerful as it can perform multiple tasks at a time. Observe the Fig. 3.5.1.

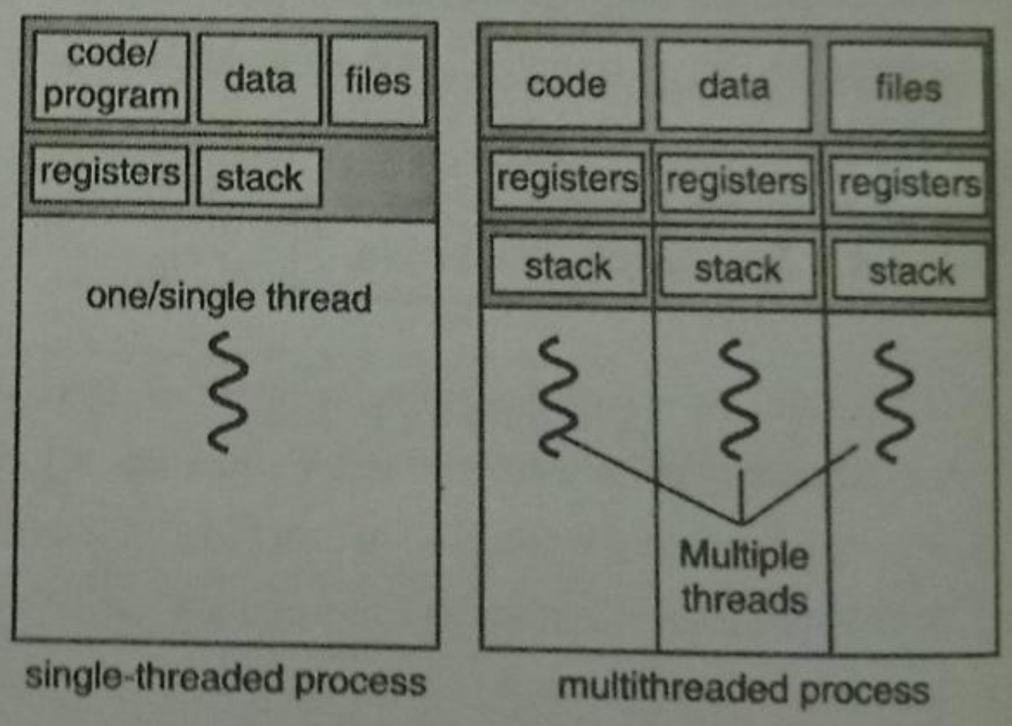


Fig. 3.5.1 : Single and multithreaded processes

- Modern operating systems are multithreaded operating systems and that makes them very powerful. One process if have multiple threads, then multiple tasks takes place. Just take an example of web browser which has many threads. One thread may display images, some may display text, some may be other links like data from network etc.

- Another example is word processor where one thread will display key strokes, one may show image, or one may do any other operation etc.
- In case of a server which is single processor system, it receives request from the different clients for the service. Now server can create different processes to serve each request. Creating process creates an overhead.
- Solution to this problem is creating a single process with multiple threads. The threads do not create overhead or time delay. So here multithreaded system can perform better than single threaded system.
- Threads plays very important role in case of Remote process call as server can collect requests as threads and can server many requests at a time. Example of this type is java's RMI.
- The conclusion is that now most of the operating systems have become multithreaded systems. The kernels have many threads which performs tasks like interrupt handling, device management, free memory management etc.

3.5.1 Benefits of Multithreaded Programming

▶ **(MSBTE - W-08, W-10, W-11, S-12, W-12, S-13, W-13, W-14, W-16, S-17)**

Q. 3.5.3 What are the benefits of multithreading. (Ref. Sec. 3.5.1) **W-08, W-10, W-11, S-12, W-12, S-13, W-13, W-14, W-16, S-17, 4 Marks**

- The benefits of multithreading programming are as follows :

1. Responsiveness
2. Resource sharing
3. Economy
4. Utilization of multiprocessor architecture

1. Responsiveness

- Multithreading operating system are interactive systems or applications. This environment may allow program to execute or run continuously even if the part of it is blocked or it may be performing a lengthy operation. This is known as responsiveness to the user.
- Due to multithreading system if user is using web browser one thread displays image while other thread does other operation. This is again part of responsiveness.

2. Resource sharing

- Threads belong to a particular process. The resources allotted or allocated to the process can be used by its threads. So many threads can share resources of its process to which they belong.
- The advantage of sharing code and data is that it allows an application to have many threads activity within the same address space of process.

3. Economy

- We have already seen that process creation is costly due to memory allocation and resource allocation. As threads shares all resources required from its process to which they belong, it is economical.
- The overhead of creating and managing processes is much more than creating and managing threads.
- An example of economy is: In Solaris creating a process is about thirty times slower than is creating a thread and context switching is also about five times slower.

4. Utilization of multiprocessor architectures

- The advantages of multithreading can be increased to the great extent in multiprocessor environment or architecture. Here many threads run on multiple processors.

- A single threaded process can run on one CPU though many processes are available. Multithreading is very useful on multi-CPU machine to increase concurrency.

3.5.2 User Thread and Kernel Thread

Q. 3.5.4 Explain users and Kernel threads. (Ref. Sec. 3.5.2) **W-15, 4 Marks**

- Support for creating thread is done on two levels i.e. user level and kernel level. User level threads are called as **user threads.** Kernel level threads are called as **kernel threads.**
- Support for the threads may be provided either at the user level called **user threads.** Kernel level threads provided by OS are called as **Kernel threads**.
- User threads are supported above the kernel and are managed without kernel support. On the other hand operating system supports and manages kernel threads. The kernel thread and user threads are related to each other.
- There are different models which shows the relationship between kernel thread and user threads as follows :
  1. Many to one model
  2. One to One model
  3. Many to many model

3.6 Multithreading Models

▶ **(MSBTE - W-08, S-09, W-10, S-11, W-11, S-12, S-13, W-13, S-15, W-15, W-16, S-17, W-17, W-18)**

Q. 3.6.1 State and explain multithreading models with diagram. (Ref. Sec. 3.6) **W-08, S-09, W-10, S-11, W-11, S-12, S-13, W-13, W-16, W-17, W-18, 2 Marks**

Q. 3.6.2 Describe many to one and one to one multithreading model with diagram. (Ref. Sec. 3.6) **S-15, 8 Marks**

Q. 3.6.3 Describe many to one and one to one multithreading model with diagram. Explain advantages of each (any two). (Ref. Sec. 3.6) **W-15, 8 Marks**

Q. 3.6.4 With neat diagram, explain many to one and many to many multithreading model with its advantages and disadvantages. (Ref. Sec. 3.6) **S-17, 6 Marks**

Q. 3.6.5 List different multithreading models. (Ref. Sec. 3.6) **W-18, 2 Marks**

- Multithreading models depicts three model as listed in the last section. We are going to learn about these three models one by one.

1. Many-to-one model

- In many to one model one kernel thread is connected with multiple user level threads. Thread management is done by thread library in user space. So it is efficient. The whole process will block if a thread makes a blocking system call.
- As only one thread can access the kernel at a time, multiple threads are unable to run in parallel on multiprocessors.
- E.g. Green Thread is the library available in Solaris operating system which implements the same model.

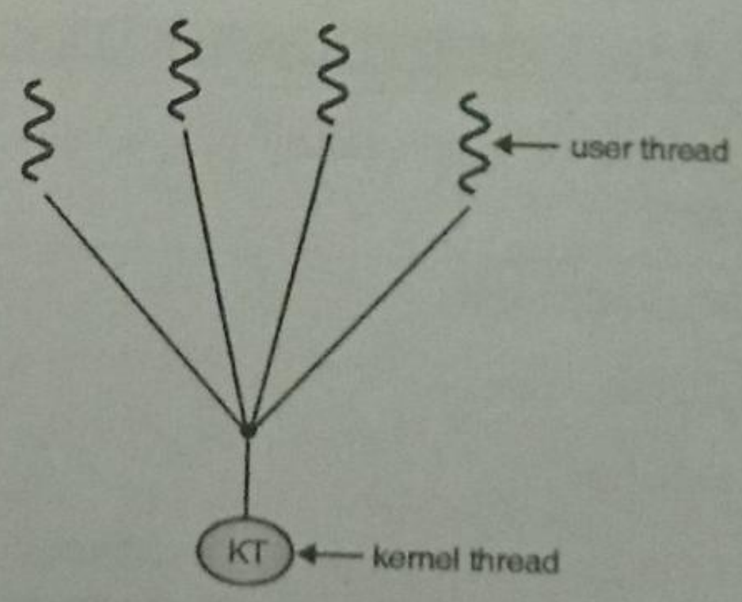


Fig. 3.6.1 : Many-to-one model

☞ **Advantages**

1. Use of library.
2. Efficient system in terms of performance.
3. One kernel thread controls multiple user threads.

☞ **Disadvantages**

1. Can not run multiple user threads parallel.
2. One block call blocks all user threads.

2. One-to-one model

▶ (MSBTE - W-09)

Q. 3.6.6 Explain one to one multithreading model of operating system. (Ref. Sec. 3.6) **W-09, 4 Marks**

- In one to one model each user thread is mapped to kernel thread. This model provides more concurrency by allowing other thread to run when a thread makes a blocking system call. This model also allows multiple threads to run in parallel on multiprocessors.
- The drawback of this model is that when user thread gets create then corresponding kernel thread should be created. Creating kernel thread is overhead for the system. So this model has this restriction that every time when user thread is created burden of creating kernel thread is there. This reduces the performances of an application.

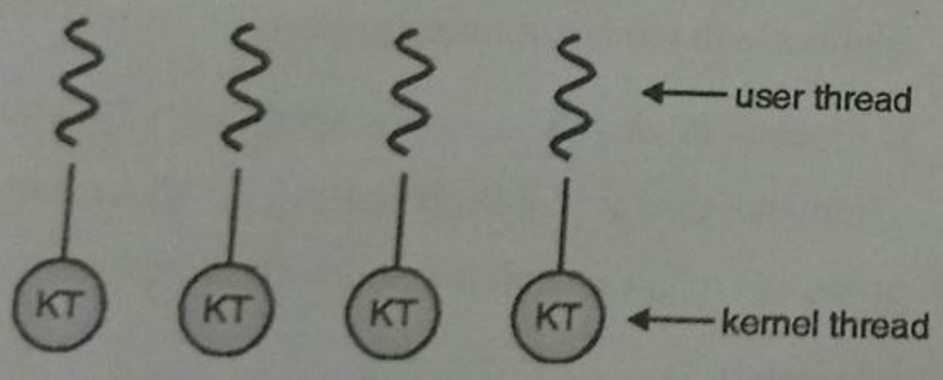


Fig. 3.6.2 : One-to-one model

- Some popular examples are the operating system which has implemented this model. e.g. Windows 95, 98, NT, XP, UNIX etc.

☞ **Advantages**

1. More concurrency.
2. Multiple threads can run parallely.
3. Less complications in the processing.

☞ **Disadvantages**

1. Every time with user's thread, kernel thread is created.
2. Kernel thread is an overhead.
3. It reduces the performance of system.

3. Many-to-Many models

▶ (MSBTE - W-09)

Q. 3.6.7 Explain many to many threading model with sample diagram. (Ref. Sec. 3.6) **W-09, 8 Marks**

- In many to many model, many user threads are attached with same number of kernel threads. The number of kernel threads dependent on the application or even on particular machine.
- Many to Many model allows developer to create many threads as user wishes. In this model true concurrency cannot be achieved because one kernel thread executes one thread at a time.
- On a multiprocessor system many kernel threads can run parallel. When one thread call block system call, other kernel thread calls for another system call.

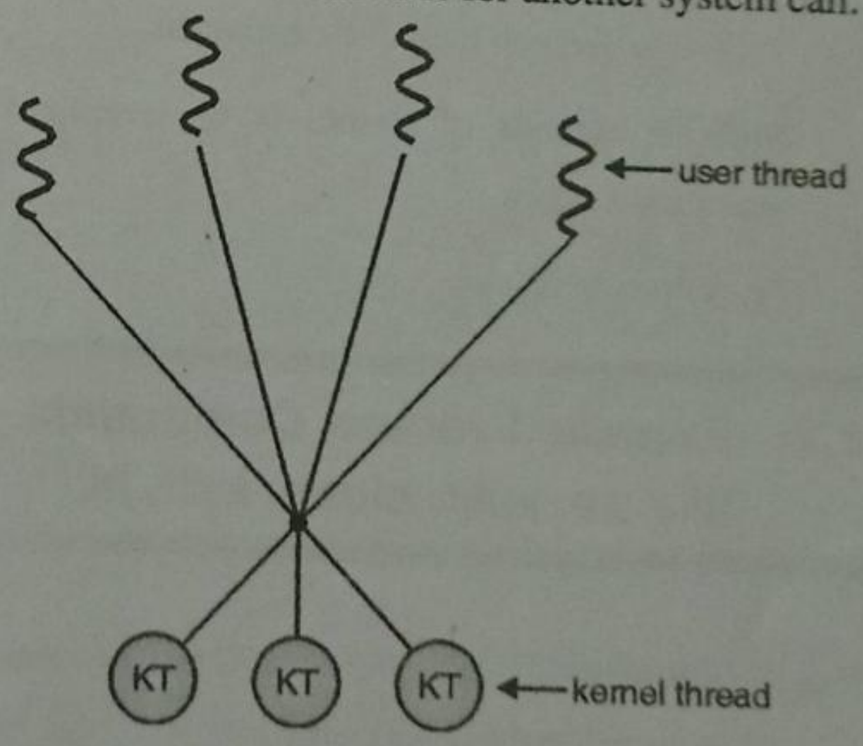


Fig. 3.6.3 : Many-to-many model

- There is one popular variation on the many to many model still multiplexes many user-level threads to a smaller or equal number of kernel threads. It also allows a user-level thread to be bound to a kernel thread.

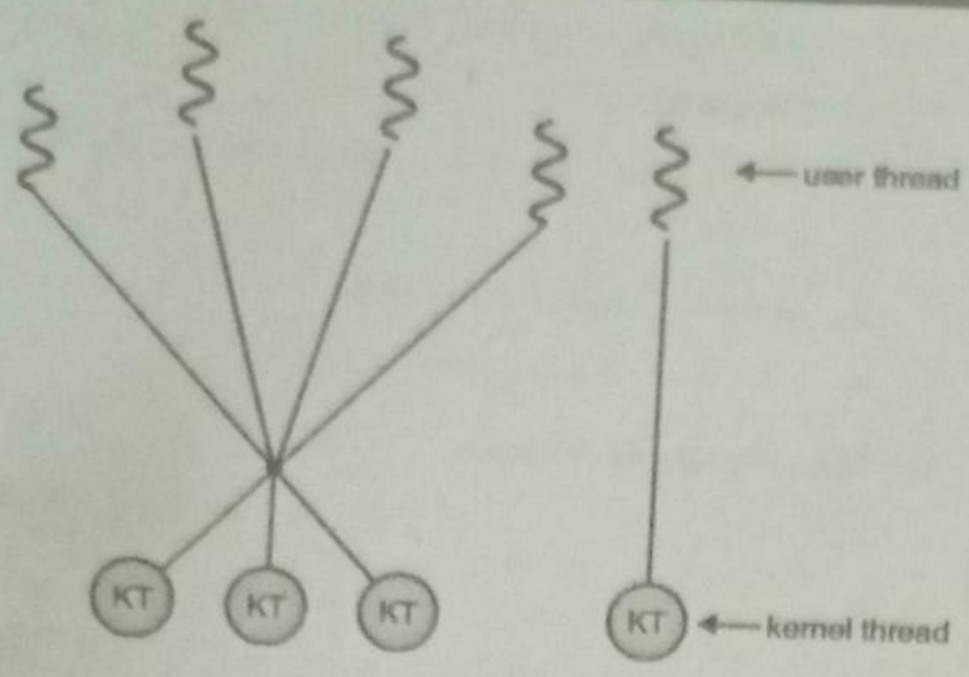


Fig. 3.6.4 : Two level model

- This variation is called as two-level model which is supported by operating system such as IRIX, HP-UX and Tru64 Unix.

☞ **Advantages**

1. Many threads can be created as per user's requirement.
2. Multiple kernel or equal to user threads can be created.

☞ **Disadvantages**

1. True concurrency cannot be achieved.
2. Multiple threads of kernel is an overhead for operating system.
3. Performance is less.

3.7 Execute Process Commands - like ps, wait, sleep, exit, kill

Process commands can be executed to perform different operations related to processes.

1. PS

- Program running on the system are called as processes. For examining these processes, users need to know about ps command.
- This command is useful in producing the information for all the processes that are running on the machine.
- The ps command is not easy to use as it presents number of parameters that makes it complicated

```
$ ps
```

Output

```
PID   TTY      TIME      CMD
 9    pts/0    00:00:00  sh
21 pts/0  00:00:00  ps
```

- The output has shown basic information like process ID, Terminal, time and command. The command ps has got different types of parameters.

(1) Unix-style parameters

(2) BSD-style parameters

(3) GNU long parameters

2. Wait Command

- **The wait is one of the built-in commands of Linux.** It waits for the completion of any running process.
- It can be used with a specific job id or process id. Current shell can know process id of last process whenever multiple processes are running. If wait command is used in this situation, wait will be applicable to the last running process.
- If process id or job id is not specified with the wait command, then it will keep waiting for all current child processes to get completed and will return exit status.
- Example :

```
#!/bin/bash
echo "testing wait command1" &
process_id=$!
echo "testing wait command2" &
wait $process_id
echo Job 1 exited with status $?
wait $!
echo Job 2 exited with status $?
```

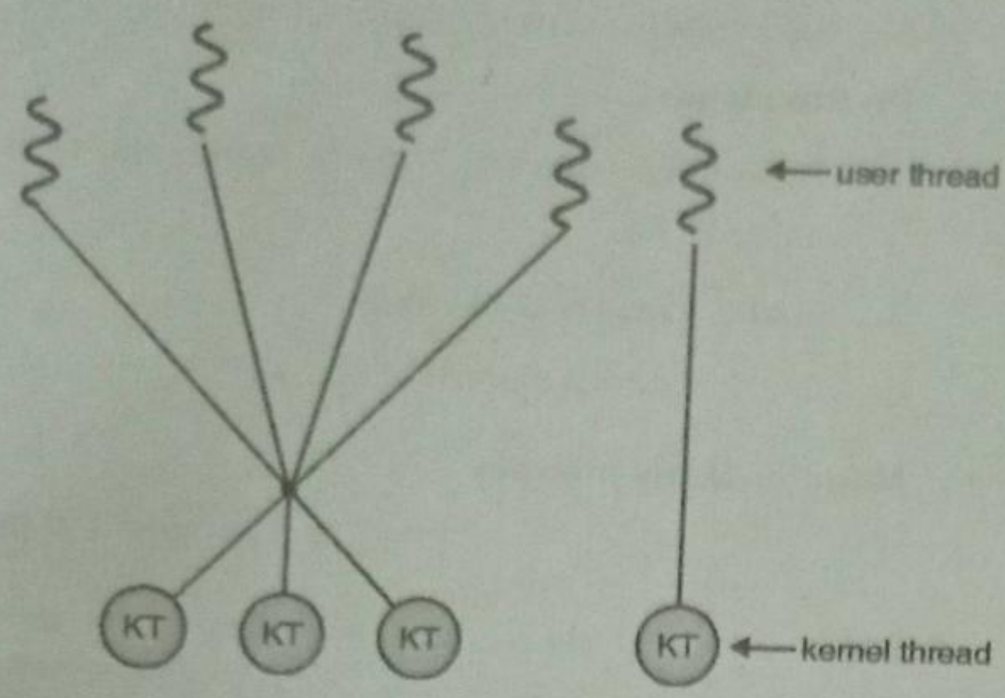


Fig. 3.6.4 : Two level model

- This variation is called as two-level model which is supported by operating system such as IRIX, HP-UX and Tru64 Unix.

☞ **Advantages**

1. Many threads can be created as per user's requirement.
2. Multiple kernel or equal to user threads can be created.

☞ **Disadvantages**

1. True concurrency cannot be achieved.
2. Multiple threads of kernel is an overhead for operating system.
3. Performance is less.

3.7 Execute Process Commands - like ps, wait, sleep, exit, kill

Process commands can be executed to perform different operations related to processes.

1. PS

- Program running on the system are called as processes. For examining these processes, users need to know about ps command.
- This command is useful in producing the information for all the processes that are running on the machine.
- The ps command is not easy to use as it presents number of parameters that makes it complicated

```
$ ps
```

Output

```
PID   TTY      TIME      CMD
 9    pts/0    00:00:00  sh
21 pts/0  00:00:00  ps
```

- The output has shown basic information like process ID, Terminal, time and command. The command ps has got different types of parameters.
  (1) Unix-style parameters
  (2) BSD-style parameters
  (3) GNU long parameters

2. Wait Command

- **The wait is one of the built-in commands of Linux.** It waits for the completion of any running process.
- It can be used with a specific job id or process id. Current shell can know process id of last process whenever multiple processes are running. If wait command is used in this situation, wait will be applicable to the last running process.
- If process id or job id is not specified with the wait command, then it will keep waiting for all current child processes to get completed and will return exit status.
- Example :

```
#!/bin/bash
echo "testing wait command1" &
process_id=$!
echo "testing wait command2" &
wait $process_id
echo Job 1 exited with status $?
wait $!
echo Job 2 exited with status $?
```

Output

```
$ bash wait1.sh

ubuntu@ubuntu-VirtualBox:~$ bash wait1.sh
testing wait command1
testing wait command2
job 1 exited with status 0
Job 2 exited with status 0
ubuntu@ubuntu-VirtualBox:~$
```

3. Sleep

- Sleep command is used to introduce delay for a specific time. This command helps to pause a process for a given time.
- Suffixes can be used with command to specify exact time. "s" can be used for seconds, "m" can be used for minutes, "h" can be used for hours and "d" can be used for days.

Syntax

```
sleep NUMBER[SUFFIX]...
```

Example

```
sleep 10
Delays for 10 seconds.
```

4. Exit command

Generally shell script will exit with the last command's exit status.

```
$ ./testfile9
```

Output

```
The answer is 2.2222

$ echo $?
```

Output

```
0
```

- User can change the status of the exit code by specifying it to exit code. Consider the example below for the same.

```
$ cat testfile10
#!/bin/bash
# Exit status testing
val1=30
val2=30
val3=$[ $val1 + val2 ]
echo The answer is $val3
exit 10

$ ./testfile10
```

Output

```
The answer is 60

$ echo $?
```

Output

```
$ 10
```

- In the above example, exit status is 10 as specified by the user in the shell script with exit command.
- User can return exit status as a value of a variable.

```
$ cat testfile11
#!/bin/bash
# Exit status testing
val1=30
val2=30
val3=$[ $val1 + val2 ]
echo The answer is $val3
exit $val1

$ ./testfile11
```

Output

```
The answer is 60

$ echo $?
```

Output

```
$ 30
```

- Maximum exit status can be upto 255 and can not exceed than that. You should be careful with this feature though, as the exit status codes can only go up to 255. If you try to return value more than 255, results will not be desirable.

5. Kill command

- Kill command is used to kill the process. User has to specify process ID for killing the process.
- Kill command sends TERMINATE signal to all the Process IDs that are listed on command line.
- It may be difficult to use kill command as it uses process ID instead of its command name.
- For killing the process user should be the owner of process or must have logged in from root user.

For example

```
$ kill 3940
```

3.8 MSBTE Questions and Answers

Winter 2008 : Total Marks 30

Q. 1 What are the benefits of multithreading. *(Ans. : Refer Section 3.5.1)* **(4 Marks)**

Q. 2 Explain process management in detail. *(Ans. : Refer Section 3.1)* **(4 Marks)**

Q. 3 State and explain any four process states. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 4 State and explain multithreading models with diagram. *(Ans. : Refer Section 3.6)* **(6 Marks)**

Q. 5 Draw and explain inter-process communication model. *(Ans. : Refer Section 3.4)* **(8 Marks)**

Q. 6 Draw and explain process control block in detail. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Summer 2009 : Total Marks 30

Q. 7 What is process? How is it different from a program? *(Ans. : Refer Section 3.1.1)* **(4 Marks)**

Q. 8 Describe various operations of process. *(Ans. : Refer Section 3.3)* **(4 Marks)**

Q. 9 Draw process state diagram and state its meaning. *(Ans. : Refer Section 3.1.2)* **(6 Marks)**

Q. 10 Explain the various types of multithreading models. *(Ans. : Refer Section 3.6)* **(8 Marks)**

Q. 11 Write difference between short term medium and long term scheduling. *(Ans. : Refer Section 3.2.2.1)* **(4 Marks)**

Q. 12 Write in brief about threading and list its advantages. *(Ans. : Refer Sections 3.5 and 3.5.1)* **(4 Marks)**

Winter 2009 : Total Marks 24

Q. 13 Explain the different process states with the help of state diagram. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 14 Explain one to one multithreading model of operating system. *(Ans. : Refer Section 3.6(2))* **(4 Marks)**

Q. 15 Explain the two techniques for the inter process communication. *(Ans. : Refer Sections 3.4.1 and 3.4.2)* **(8 Marks)**

Q. 16 What is a thread? Explain many-to-many threading model with sample diagram. *(Ans. : Refer Sections 3.5 and 3.6(3))* **(8 Marks)**

Summer 2010 : Total Marks 30

Q. 17 With neat diagram explain process control block (PCB). *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 18 State the meaning of scheduler. Explain any one type of scheduler in details. *(Ans. : Refer Section 3.2.2)* **(4 Marks)**

Q. 19 What are threads ? Explain benefits of multiprogramming using threads. *(Ans. : Refer Sections 3.5 and 3.5.1)* **(6 Marks)**

Q. 20 What is process ? Explain process in detail with the help of state diagram. *(Ans. : Refer Sections 3.1.1 and 3.1.2)* **(8 Marks)**

Q. 21 Explain process termination in detail. *(Ans. : Refer Section 3.3.2)* **(4 Marks)**

Q. 22 Explain the concept of mutual exclusion in detail. *(Ans. : Refer Section 3.4.3)* **(4 Marks)**

Winter 2010 : Total Marks 25

Q. 23 Draw and describe process state diagrams. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 24 Draw and explain process control block (PCB) in detail. *(Ans. : Refer Section 3.1.3)* **(8 Marks)**

Q. 25 Describe Inter Process Communication. *(Ans. : Refer Section 3.4)* **(3 Marks)**

Q. 26 Describe multithreading and its models. *(Ans. : Refer Section 3.6)* **(4 Marks)**

Q. 27 Describe the terms : **(6 Marks)**

(i) Scheduling Queues *(Ans. : Refer Section 3.2.1)*

(ii) Scheduler *(Ans. : Refer Section 3.2.2)*

(iii) Context Switch *(Ans. : Refer Section 3.2.3)*

Summer 2011 : Total Marks 28

Q. 28 Explain creation and termination operations on process. *(Ans. : Refer Section 3.3)* **(4 Marks)**

Q. 29 Explain PCB (Process Control Block) with suitable diagram. *(Ans. : Refer Section 3.1.3)* **(6 Marks)**

Q. 30 With advantages and disadvantages explain one-to-one model and many-to-many model for multithreading.*(Ans. : Refer Section 3.6)* **(6 Marks)**

Q. 31 With suitable diagram explain interprocess communication models. *(Ans. : Refer Section 3.4)* **(8 Marks)**

Q. 32 Define following terms with their diagram role in scheduling.

(i) Short term scheduler.

(ii) Long term scheduler. *(Ans. : Refer Section 3.2.2)* **(4 Marks)**

Winter 2011 : Total Marks 22

Q. 33 Draw types of schedulers. State and explain any one scheduler used in scheduling. *(Ans. : Refer Section 3.2.2)* **(4 Marks)**

Q. 34 Explain any two scheduling queues ? *(Ans. : Refer Section 3.2.1)* **(4 Marks)**

Q. 35 Define following terms : Threads *(Ans. : Refer Section 3.5)* **(2 Marks)**

Q. 36 What is multithreading? Explain with suitable example. *(Ans. : Refer Section 3.6)* **(4 Marks)**

Q. 37 State and explain any two process states. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 38 Describe the concept of inter-process communication. *(Ans. : Refer Section 3.4)* **(4 Marks)**

Summer 2012 : Total Marks 40

Q. 39 State and explain multithreading models with diagram. *(Ans. : Refer Section 3.6)* **(8 Marks)**

Q. 40 State the role of operating system in process communication. *(Ans. : Refer Section 3.4)* **(2 Marks)**

Q. 41 List model of threads. *(Ans. : Refer Section 3.6)* **(2 Marks)**

Q. 42 Draw process state diagram with neat labeling. *(Ans. : Refer Section 3.1.2)* **(2 Marks)**

Q. 43 What is a context switch? Define. *(Ans. : Refer Section 3.2.3)* **(2 Marks)**

Q. 44 Describe the terms : **(8 Marks)**

(i) Scheduling queues *(Ans. : Refer Section 3.2.1)*

(ii) Scheduler *(Ans. : Refer Section 3.2.2)*

(iii) Thread *(Ans. : Refer Section 3.5)*

(iv) Multithreading *(Ans. : Refer Section 3.5)*

Q. 45 Explain the concept of Inter-Process communication. *(Ans. : Refer Section 3.4)* **(8 Marks)**

Q. 46 What is the process? Describe process creation and termination along with necessary conditions. *(Ans. : Refer Section 3.1.1 and 3.3)* **(8 Marks)**

Winter 2012 : Total Marks 32

Q. 47 Explain different process states with the help of state diagram? *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 48 State the benefits of Multithreading ? *(Ans. : Refer Section 3.5.1)* **(4 Marks)**

Q. 49 Draw and explain Inter-process Communication Model ? *(Ans. : Refer Section 3.4)* **(4 Marks)**

Q. 50 Describe the terms : **(4 Marks)**

(i) Scheduling Queues *(Ans. : Refer Section 3.2.1)*

(ii) Context Switch *(Ans. : Refer Section 3.2.3)*

Q. 51 State and explain Multithreading model with diagram ? *(Ans. : Refer Section 3.5)* **(4 Marks)**

Q. 52 Draw and explain process control block in detail? *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 53 Differentiate between short term and long-term scheduler. *(Ans. : Refer Section 3.2.2.1)* **(4 Marks)**

Q. 54 What is process? How is it different from a program? *(Ans. : Refer Section 3.1.1)* **(4 Marks)**

Summer 2013 : Total Marks 28

Q. 55 Write a note on process management. *(Ans. : Refer Section 3.1)* **(4 Marks)**

Q. 56 Write note on context switch. *(Ans. : Refer Section 3.2.3)* **(4 Marks)**

Q. 57 Explain Multithreading and its model. *(Ans. : Refer Section 3.5 and 3.6)* **(4 Marks)**

Q. 58 Describe how process is terminated ? *(Ans. : Refer Section 3.3)* **(4 Marks)**

Q. 59 Explain Inter-process communication. *(Ans. : Refer Section 3.4)* **(4 Marks)**

Q. 60 How process control block used in process creation? Describe with diagram. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 61 What is thread ? Explain benefits of multithreaded programming. *(Ans. : Refer Sections 3.5 and 3.5.1)* **(4 Marks)**

Winter 2013 : Total Marks 36

Q. 62 List multithreading models. Explain any one with suitable diagram. *(Ans. : Refer Section 3.6)* **(4 Marks)**

Q. 63 What is scheduler? Explain multilevel queue scheduler with suitable example. *(Ans. : Refer Sections 3.2.1 and 3.2.2)* **(4 Marks)**

Q. 64 Draw and explain inter-process communication model. *(Ans. : Refer Section 3.4)* **(4 Marks)**

Q. 65 What is context switching? *(Ans. : Refer Section 3.2.3)* **(4 Marks)**

Q. 66 What are I/O bound processes? What are CPU bound processes? *(Ans. : Refer Section 3.2.2)* **(4 Marks)**

Q. 67 Draw the process state diagram and explain any one state. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 68 Explain : **(8 Marks)**

(i) Process creation

(ii) Process termination. *(Ans. : Refer Section 3.3)*

Q. 69 With neat diagram, explain process control block. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Summer 2014 : Total Marks 28

Q. 70 With neat diagram explain process control block. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 71 State the benefits of multithreading. *(Ans. : Refer Section 3.5.1)* **(4 Marks)**

Q. 72 Explain the concept of mutual exclusion in detail. *(Ans. : Refer Section 3.4.5)* **(4 Marks)**

Q. 73 Draw and explain inter process communication model. *(Ans. : Refer Section 3.4)* **(4 Marks)**

Q. 74 Describe the terms :

(i) Scheduling queues *(Ans. : Refer Section 3.2.1)* **(2 Marks)**

(ii) Context switch *(Ans. : Refer Section 3.2.3)* **(2 Marks)**

Q. 75 Explain process termination. *(Ans. : Refer Section 3.3.2)* **(4 Marks)**

Q. 76 Differentiate between short term and long term schedular. *(Ans. : Refer Section 3.2.2.1)* **(4 Marks)**

Winter 2014 : Total Marks 34

Q. 77 What is process ? Explain the different process state with the help of state diagram. *(Ans. : Refer Section 3.1.1 and 3.1.2)* **(4 Marks)**

Q. 78 Describe the term :

(1) Scheduling queues *(Ans. : Refer Section 3.2.1)* **(2 Marks)**

(2) Scheduler *(Ans. : Refer Section 3.2.2)* **(2 Marks)**

(3) Context switch *(Ans. : Refer Section 3.2.3)* **(2 Marks)**

Q. 79 Define PCB (Process Control Block) with suitable diagram. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 80 Describe creation and termination operation on process.
(Ans. : Refer Sections 3.3.1 and 3.3.2) **(4 Marks)**

Q. 81 What is multithreading ? Explain with suitable diagram. *(Ans. : Refer Section 3.5)* **(4 Marks)**

Q. 82 Describe types of scheduler used in scheduling.
(Ans. : Refer Section 3.2.2) **(4 Marks)**

Q. 83 With suitable diagram explain interprocess communication model.
(Ans. : Refer Section 3.4) **(8 Marks)**

Summer 2015 : Total Marks 34

Q. 84 Define process. Explain process state in detail with the help of state diagram.
(Ans. : Refer Sections 3.1.1 and 3.1.2) **(Chap 3, 4 Marks)**

Q. 85 Describe any four operations performed on process.
(Ans. : Refer Section 3.3) **(Chap 3, 4 Marks)**

Q. 86 Explain context switch with suitable example. *(Ans. : Refer Section 3.2.3)* **(Chap 3, 4 Marks)**

Q. 87 State and describe types of scheduler.
(Ans. : Refer Section 3.2.2) **(Chap 3, 4 Marks)**

Q. 88 Describe many to one and one to one multithreading model with diagram.
(Ans. : Refer Section 3.6.(1)(2)) **(Chap 3, 6 Marks)**

Q. 89 Explain Interprocess communication models with diagram.
(Ans. : Refer Section 3.4) **(Chap 3, 8 Marks)**

Q. 90 Draw and explain process control block in detail.
(Ans. : Refer Section 3.1.3) **(Chap 3, 4 Marks)**

Winter 2015 : Total Marks 30

Q. 91 Draw and explain process state diagram.
(Ans. : Refer Section 3.1.2) **(Chap 3, 4 Marks)**

Q. 92 Explain interprocess communication.
(Ans. : Refer Section 3.4) **(Chap 3, 4 Marks)**

Q. 93 What is thread ? Explain users and Kernel threads.
(Ans. : Refer Sections 3.5 and 3.5.2) **(Chap 3, 4 Marks)**

Q. 94 What is process ? Explain the different process states with diagram.
(Ans. : Refer Sections 3.1.1 and 3.1.2) **(Chap 3, 6 Marks)**

Q. 95 Describe many to one and one to one multithreading model with diagram. Explain advantages of each (any two).
(Ans. : Refer Section 3.6) **(Chap 3, 8 Marks)**

Q. 96 Explain the concept of context switching.
(Ans. : Refer Section 3.2.3) **(Chap 3, 4 Marks)**

Summer 2016 : Total Marks 34

Q. 97 With neat diagram describe use of Process Control Block (PCB).
(Ans. : Refer Section 3.1.3) **(4 Marks)**

Q. 98 Draw the process state diagram and describe each state in one/two sentences.
(Ans. : Refer Section 3.1.2) **(4 Marks)**

Q. 99 What is inter process communication? Describe any one technique of it.
(Ans. : Refer Sections 3.4 and 3.4.1) **(4 Marks)**

Q. 100 Differentiate between long term scheduler and short term scheduler on basis of :

(i) Selection of job

(ii) Frequency of execution

(iii) Speed

(iv) Accessing which part of system. **(4 Marks)**

Q. 101 Describe how semaphores are useful for solving problems of interposes communication.
(Ans. : Refer Section 3.4.6) **(6 Marks)**

Q. 102 Describe following terms :

(1) Scheduling queues

(2) Schedular

(3) Thread

(4) Multithreading.
(Ans. : Refer Sections 3.2.1, 3.2.2 and 3.5) **(8 Marks)**

Q. 103 Describe how context switch is executed by operating system.
(Ans. : Refer Section 3.2.3) **(4 Marks)**

Winter 2016 : Total Marks 34

Q. 104 Define thread. State any three benefits of thread. *(Ans. : Refer Sections 3.5 and 3.5.1)* **(4 Marks)**

Q. 105 List and explain various types of multi-threading models. *(Ans. : Refer Section 3.6)* **(4 Marks)**

Q. 106 Explain process control block with suitable diagram. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 107 Draw and explain inter-process communication model. *(Ans. : Refer Section 3.4)* **(6 Marks)**

Q. 108 Differentiate between short term, medium term and long term scheduling. *(Ans. : Refer Section 3.2.2.1)* **(4 Marks)**

Q. 109 Define process. Describe process creation and termination. *(Ans. : Refer Sections 3.1.1, 3.3.1 and 3.3.2)* **(8 Marks)**

Q. 110 Explain context switch with suitable example. *(Ans. : Refer Section 3.2.3)* **(4 Marks)**

Summer 2017 : Total Marks 42

Q. 111 Describe the following : **(4 Marks)**

(i) Schedulers

(ii) Context switch

(Ans. : Refer Sections 3.2.2 and 3.2.3)

Q. 112 Describe the critical-section problem. *(Ans. : Refer Section 3.4.3)* **(4 Marks)**

Q. 113 Describe Process Control Block (PCB) with suitable diagram. *(Ans. : Refer Section 3.1.3)* **(4 Marks)**

Q. 114 Write use of following system calls. **(4 Marks)**

(1) fork( ) (2) exec( )

(3) abort( ) (4) end( )

(Ans. : Refer Section 3.3.1)

Q. 115 Write benefits of using threads. *(Ans. : Refer Section 3.5.1)* **(4 Marks)**

Q. 116 With neat diagram, explain many to one and many to many multithreading model with its advantages and disadvantages. *(Ans. : Refer Section 3.6(1)(3))* **(6 Marks)**

Q. 117 With neat diagram, explain Message passing system. Also describe the following : **(8 Marks)**

(i) Naming

(ii) Synchronization

(iii) Buffering

(Ans. : Refer Sections 3.4.2, 3.4.2.1, 3.4.2.2 and 3.4.2.3)

Q. 118 Describe Process in memory with diagram. *(Ans. : Refer Section 3.1.1)* **(4 Marks)**

Winter 2017 : Total Marks 34

Q. 119 Differentiate between short term and long term scheduler. *(Ans. : Refer Section 3.2.2.1)* **(4 Marks)**

Q. 120 State and explain different process state. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 121 Explain context switch with help of diagram. *(Ans. : Refer Section 3.2.3)* **(4 Marks)**

Q. 122 Describe how process is terminated. *(Ans. : Refer Section 3.3.2)* **(4 Marks)**

Q. 123 With neat diagram describe Process Control Block (PCB). *(Ans. : Refer Section 3.1.3)* **(6 Marks)**

Q. 124 List and explain various type of multi-threading models with diagram. *(Ans. : Refer Section 3.6)* **(8 Marks)**

Q. 125 Draw and explain Inter-process communication model. *(Ans. : Refer Section 3.4)* **(4 Marks)**

Summer 2018 - Total Marks 24

Q. 126 List and draw a neat labelled diagram of process state. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 127 Explain the working of Inter-process communication considering.

(1) Shared memory

(2) Message passing **(6 Marks)**

(Ans. : Refer Section 3.4)

Q. 128 Define synchronization Explain

(i) Blocking

(ii) Non Blocking in message passing

(Ans. : Refer Section 3.4.2.2) **(4 Marks)**

Q. 129 Differentiate between long term scheduling and medium term scheduling.

(Ans. : Refer Section 3.2.2.1) **(4 Marks)**

Q. 130 Explain working of CPU switch from process to process with diagram.

(Ans. : Refer Section 3.1.3) **(6 Marks)**

Winter 2018 - Total Marks 34

Q. 131 Define process.

(Ans. : Refer Section 3.1.1) **(2 Marks)**

Q. 132 List different multithreading models.

(Ans. : Refer Section 3.6) **(2 Marks)**

Q.133 List any two condition leading to process suspension.

(Ans. : Refer Section3.2.3) **(2 Marks)**

Q. 134 Draw process state diagram and describe each state. *(Ans. : Refer Section 3.1.2)* **(4 Marks)**

Q. 135 What is process management? State any four functions of process management.

(Ans. : Refer Section 3.1) **(4 Marks)**

Q. 136 Draw and explain process control block in detail.

(Ans. : Refer Section 3.1.3) **(4 Marks)**

Q. 137 Draw and explain inter-process communication model. *(Ans. : Refer Section 3.4)* **(4 Marks)**

Q. 138 Explain multithreading model with diagram.

(Ans. : Refer Section 3.6) **(4 Marks)**

Q. 139 How context switching is done?

(Ans. : Refer Section 3.2.3) **(4 Marks)**

Q. 140 Why is process creation necessary? State the role of fork process in the context.

(Ans. : Refer Section 3.3.1) **(4 Marks)**

Chapter Ends...

❑❑❑