


CMR INSTITUTE OF TECHNOLOGY	USN							 <small>CMRIT CENTRE FOR MANAGEMENT RESEARCH &amp; INNOVATION TECHNOLOGY &amp; INNOVATION</small>
	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	

**Internal Assessment Test - I**

Sub:	Operating Systems						Code:	18MCA25	
Date:	20.04.2019	Duration:	90 mins	Max Marks:	50	Sem:	II	Branch:	MCA

Answer Any One FULL Question from each part.

								Mar	OBE	
								ks	CO	RBT
<b>Part - I</b>										
1	What are the various services of an operating system? Explain briefly (or)						10	CO1	L2	
2	Describe in detail about the components of operating system and its responsibilities.						10	CO1	L2	
<b>Part – II</b>										
3	What is processor register? Explain different types of process registers? (or)						10	CO1	L1	
4	Explain the following types of OS: i) Real Time ii) Multiprocessor System iii) Clustered System.						10	CO1	L1	

<b>PART - III</b>										
5	(a)	Define system call. Classify the types of systems calls.						05	CO1	L1
	(b)	How does system call work? Explain with neat diagram. (or)						05	CO1	L2
6	Explain in detail about virtual machines						10	CO1	L2	
<b>Part – IV</b>										
7	Explain any two I/O communication techniques with flowchart. (or)						10	CO1	L1	
8	(a)	Explain layer structure with a neat diagram.						05	CO1	L1
	(b)	Write a short note on microkernel						05	CO1	L2
<b>Part – V</b>										
9	(a)	What is memory hierarchy? List and explain the category with neat diagram						06	CO1	L1
	(b)	Write a short note on cache memory (or)						04	CO1	L1
10.	Consider the following set of processes with given length of CPU burst:						10	CO2	L3	
	Processes	P1	P2	P3	P4	P5				
	Burst Time	10	1	2	1	5				
	Priority	3	1	3	4	2				
<p>All processes arrived at time 0 in the given order.</p> <p>i) Draw Gantt chart using SJF (non-preemptive), Priority (Non-preemptive) [Smallest number implies highest priority], and Round Robin [Quantum-2 ms] scheduling policies.</p> <p>ii) Find the average waiting time for each scheduling policy.</p>										

Internal Assessment Test 1–April 2019

Sub:	Operating Systems						
Date:	20.4.2019	Duration:	90 mins	Max Marks:	50	Sem:	2

Code:	17MCA 32
Branch:	MCA

1 What are the various services of an operating system? Explain briefly

10

An OS provides an environment for the execution of programs. Also, it provides certain services to the programs and its users. Though these services differ from one OS to the other, following are some general services provided by any OS. □ Program execution: The system must be able to load a program into memory and to run that program. The program must be able to end its execution, either normally or abnormally (indicating error).

□ I/O operations: A running program may require I/O. This I/O may involve a file or an I/O device. For efficiency and protection, users usually cannot control I/O devices directly. Therefore, the OS must provide a means to do I/O.

□ File-system manipulation: The OS must facilitate the programs to read and write the files. And also, programs must be allowed to create and delete files by name.

□ Communications: Processes may need to exchange information with each other. These processes may be running on same computer or on different computers.

Communications may be implemented via shared memory, or by the technique of message passing, in which packets of information are moved between processes by the OS.

□ Error detection: The OS constantly needs to be aware of possible errors. Errors may occur in any of CPU, memory hardware, I/O devices, user program etc. For each type of error, the OS should take the appropriate action to ensure correct and consistent computing. OS also has another set of functionalities to help the proper functioning of itself.

□ Resource allocation: When multiple users are logged on the system or multiple jobs are running at the same time, resources must be allocated to each of them. OS has to manage many resources like CPU cycles, main memory, and file storage etc. OS uses CPU scheduling routines for effective usage of CPU. These routines manage speed of the CPU, the jobs that must be executed, the number of registers available, and such other factors.

□ Accounting: We want to keep track of which users use how many and which kinds of computer resources. This record keeping may be used for accounting (so that users can be billed) or simply for accumulating usage statistics. Usage statistics may be a valuable tool for researchers who wish to reconfigure the system to improve computing services.

□ Protection: Information on a multi-user computer system must be secured. When multiple processes are executing at a time, one process should not interfere with the others. Protection involves ensuring that all access to system resources is controlled. System must be protected from outsiders as well. This may be achieved by authenticating the users by means of a password. It also involves defending external I/O devices, modems, network adapters etc. from invalid access.

2 (a) Describe in detail about the components of operating system and its responsibilities.

10

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 System

### Process Management

A *process* is a program in execution. A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.

- Processes can create sub-processes to execute concurrently.
  - A program by itself is not a process; a program is a passive entity, whereas a process is an active entity.
  - The execution of a process must progress in a sequential fashion. The CPU executes one instruction of the process after another until the process completes.
  - Operating System processes: Those execute system code. • User processes: Those that execute user code.
- The operating system is responsible for the following activities in connection with process management.
- Process creation and deletion.
  - Process suspension and resumption.
  - Provision of mechanisms for:
    - 1) process synchronization
    - 2) process communication
  - Deadlock handling

### Main Memory Management

Memory is a large array of words or bytes, each with its own address. It is a repository (storage) of quickly accessible data shared by the CPU and I/O devices.

- Main memory is a volatile storage device. It loses its contents in the case of system failure.
- The operating system is responsible for the following activities in connections with memory management:
  - Keep track of which parts of memory are currently being used and by whom.
  - Decide which processes to load when memory space becomes available.
  - Allocate and deallocate memory space as needed.

### File Management

A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.

- A file consists of a sequence of bits, bytes, lines, or records whose meanings are defined by their creators.
- The operating system is responsible for the following activities in connections with file management:
  - File creation and deletion.
  - Directory creation and deletion.
  - Support of primitives for manipulating files and directories.
  - Mapping files onto secondary storage.
  - File backup on stable (nonvolatile) storage media.

### I/O System Management

The I/O system consists of:

- A buffer-caching system
- A general device-driver interface
- Drivers for specific hardware devices
- The O.S. hides the peculiarities of specific hardware devices from the user.

### Secondary Storage Management

Since main memory (*primary storage*) is volatile and too small to accommodate all data and programs permanently, the computer system must provide *secondary storage* to back up main memory.

- Most modern computer systems use disks as the principle on-line storage medium, for both programs and data.
- The operating system is responsible for the following activities in connection with disk management:
  - Free space management
  - Storage allocation
  - Disk scheduling

### Networking

A *distributed* system is a collection of processors that do not share memory or a clock. Each processor has its own local memory and clock.

- The processors in the system are connected through a communication network.
- A distributed system provides user access to various system resources.
- Access to a shared resource allows:
  - Computation speed-up
  - Increased data availability
  - Enhanced reliability

### Protection System

• *Protection* refers to a mechanism for controlling access by programs, processes, or users to both system and user resources.

- The protection mechanism must:
  - distinguish between authorized and unauthorized usage.
  - specify the controls to be imposed.
  - provide a means of enforcement.

### Command-Interpreter System

Command-Interpreter system is a system program, which is the interface between the user and the operating system.

- Command-Interpreter system is known as the shell.
- Some operating systems provide a user-friendly interface (mouse-based window) such as, Macintosh and Microsoft Windows.
- Some operating systems provide text interface (commands are typed on keyboard) such as MS-DOS and Unix shells.

Many commands are given to the operating system by control statements which deal with:

- process creation and management
- I/O handling
- secondary-storage management
- main-memory management
- file-system access
- protection
- networking

- The program that reads and interprets control statements is called variously:
  - control-card interpreter
  - command-line interpreter

– shell (in UNIX)

- 3 What is processor register? Explain different types of process registers? 10  
A processor includes a set of registers that provide memory. But, this memory is faster and smaller than the main memory. These registers can be segregated into two types based on their functionalities as discussed in the following sections.

### User – visible Registers

These registers enable the assembly language programmer to minimize the main memory references by optimizing register use. Higher level languages have an optimizing compiler which will make a choice between registers and main memory to store variables. Some languages like C allow the programmers to decide which variable has to be stored in register.

A user visible register is generally available to all programs. Types of registers that are available are: data, address and condition code registers.

**Data Registers:** They can be assigned to different types of functions by the programmer. Sometimes, these are general purpose and can be used with any machine instruction that performs operations on data. Still, there are some restrictions like – few registers are used for floating-point operations and few are only for integers.

**Address Registers:** These registers contain main memory addresses of data and instructions. They may be of general purpose or may be used for a particular way of addressing memory. Few examples are as given below:

- o Index Registers: Indexed addressing is a common mode of addressing which involves adding and index to a base value to get the effective address.
- o Segment Pointer: In segmented addressing, a memory is divided into segments (a variable-length block of words). In this mode of addressing, a register is used to hold the base address of the segment.
- o Stack Pointer: If there is a user-visible stack addressing, then there is a register pointing to the top of the stack. This allows push and pop operations on instructions stored in the stack.

### \Control and Status Registers

The registers used by the processor to control its operation are called as Control and Status registers. These registers are also used for controlling the execution of programs. Most of such registers are not visible to the user. Along with MAR, MBR, I/OAR and I/OBR discussed earlier, following registers are also needed for an instruction to execute:

- Program Counter: that contains the address of next instruction to be fetched.
- Instruction Register(IR): contains the instruction most recently fetched.

All processor designs also include a register or set of registers, known as *program status word (PSW)*. It contains condition codes and status information like interrupt enable/disable bit and kernel/user mode bit.

*Condition codes* (also known as *flags*) are bits set by the processor hardware as the result of operations. For example, an arithmetic operation may produce a positive, negative, zero or overflow result. Condition code bits are collected into one or more registers. And, they are the part of a control register. These bits only can be read to know the feedback of the instruction execution, but they can't be altered

- 4 Explain the following types of OS: 10

- i) Real Time ii) Multiprocessor System iii) Clustered System.

## **REAL-TIME SYSTEMS**

A real-time system is used when there is a time requirements on the operation of a processor or the flow of data. Thus, it is used as a control device in a dedicated application.

Sensors bring data to the computer. The computer must analyze the data and possibly adjust controls to modify the sensor inputs. Systems that control scientific experiments, medical imaging systems, industrial control systems, and certain display systems are realtime systems. Some automobile-engine fuel-injection systems, home-appliance controllers, and weapon systems are also real-time systems.

A real-time system has well-defined, fixed time constraints. Processing must be done within the defined constraints, or the system will fail. For instance, a robot arm must be instructed to halt before it reaches a wall. A real-time system functions correctly only if it returns the correct result within its time constraints. So in real-time system, a quick response is expected; whereas in batch system time constraints are not at all there.

- Real-time systems may be hard or soft.
- A hard real-time system guarantees that critical tasks be completed on time.
- A soft real-time system is less restrictive, where a critical real-time task gets priority over other tasks, and retains that priority until it completes.
- But, soft real-time systems have limited utility than hard real-time systems. Given their lack of deadline support, they are risky to use for industrial control and robotics.
- They are useful in areas like multimedia, virtual reality, and advanced scientific projects-such as undersea exploration and planetary rovers.

## **Multiprocessor systems**

### **Advantages :**

□ Increased throughput: By increasing the number of processors, we hope to get more work done in less time. The speed-up ratio with N processors is not N; rather, it is less than N. When multiple processors cooperate on a task, a certain amount of overhead is incurred in keeping all the parts working correctly. This overhead, plus conflict for shared resources, lowers the expected gain from additional processors.

□ Economy of scale: Multiprocessor systems can save more money than multiple single-processor systems, because they can share peripherals, mass storage, and power supplies. If several programs operate on the same set of data, it is cheaper to store those data on one disk and to have all the processors share them, than to have many computers with local disks and many copies of the data.

□ Increased reliability: If functions can be distributed properly among several processors, then the failure of one processor will not halt the system, only slows down. If we have ten processors and one fails, then each of the remaining nine processors must take a share of the work of the failed processor. Thus, the entire system runs only 10 percent slower, rather than failing altogether. This ability to continue providing service proportional to the level of surviving hardware is called *graceful degradation*. Systems designed for graceful degradation are also called *fault tolerant*.

The most common multiple-processor systems now use symmetric multiprocessing (SMP), in which each processor runs an identical copy of the OS, and these copies communicate with one another as needed. Some systems use asymmetric multiprocessing, in which each processor is assigned a specific task. A

master processor controls the system; the other processors either look to the master for instruction or have predefined tasks. This scheme defines a master-slave relationship. The master processor schedules and allocates work to the slave processors. SMP means that all processors are peers; no master-slave relationship exists between processors. Each processor concurrently runs a copy of the operating system.

The difference between symmetric and asymmetric multiprocessing may be the result of either hardware or software. Special hardware can differentiate the multiple processors, or the software can be written to allow only one master and multiple slaves. For instance, Sun's operating system SunOS Version 4 provides asymm Clustered System

## CLUSTERED SYSTEM

---

### Clustered Systems

- Definition: Clustered computers which share storage and are closely linked via LAN networking.
- Advantages: high availability, performance improvement, etc.
- Types
  - Asymmetric/symmetric clustering
  - Parallel clustering – multiple hosts that access the same data on the shared storage.
  - Global clusters
- Distributed Lock Manager (DLM)

Asymmetric mode

- one machine is in hot standby mode while the other is running the applications.
- The hot standby host (machine) just monitors the active server.
- If that server fails, the hot standby host becomes the active server.

Symmetric mode

- two or more hosts are running applications, and they are monitoring each other. This mode is obviously more efficient, as it uses all of the available hardware

5 (a) Define system call. Classify the types of systems calls.

05

System calls provide the interface between a process and OS. These calls are normally assembly-language instructions and used by the assembly-language programmers. Some systems allow system calls to be made directly from a higher level language program. In such cases, the calls resemble predefined function or subroutine calls. They may generate a call to a special run-time routine that makes the system call or the system call may be generated directly in-line.

System calls can be grouped roughly into five major categories: process control, file management, device management, information maintenance, and communications. These are discussed in the following sections. Table 2.1 summarizes the types of system calls provided by OS.

**Table 2.1 Types of System Calls**

Category	System Calls
Process Control	<ul style="list-style-type: none"> <li>• end, abort</li> <li>• load, execute</li> <li>• create process, terminate process</li> <li>• get process attributes, set process attributes</li> <li>• wait for time</li> <li>• wait event, signal event</li> <li>• allocate and free memory</li> </ul>
File Management	<ul style="list-style-type: none"> <li>• create file, delete file</li> <li>• open, close</li> <li>• read, write, reposition</li> <li>• get file attributes, set file attributes</li> </ul>
Device Management	<ul style="list-style-type: none"> <li>• request device, release device</li> <li>• read, write, reposition</li> <li>• get device attributes, set device attributes</li> <li>• logically attach or detach devices</li> </ul>
Information Maintenance	<ul style="list-style-type: none"> <li>• get time or date, set time or date</li> <li>• get system data, set system data</li> <li>• get process, file, or device attributes</li> <li>• set process, file, or device attributes</li> </ul>
Communications	<ul style="list-style-type: none"> <li>• create, delete communication connection</li> <li>• send, receive messages</li> <li>• transfer status information</li> <li>• attach or detach remote devices</li> </ul>

(b) How does system call work? Explain with neat diagram.

To understand how system calls are used, consider an example of writing a program to read data from one file and to copy them to another file. The program needs names of two files as an input. These names may be asked from the user. Now, this will initiate a sequence of system calls: to write a prompting message on the screen, and to read the characters from keyboard which defines names of files. Then, the program opens an input file and creates the output file. This operation requires another system call that may possibly face error conditions. Each of the error conditions (like input file doesn't exist, no permission to read, output file can't be created, no write permission etc) require different system calls. When everything is proper, we enter a loop to read from input file and write into the output file. Both of these operations are system calls. Every read/write must return status information for any possible error or end-of-file. Again, this requires system call.

Once the task is done, both the files must be closed using system calls. Finally, the program will be terminated using final system call. Thus, we can make out that a very simple program also uses OS heavily. However, the users never see such details. Because, the set of built-in functions provided by the compiler of programming languages provides very simpler interface.

Occurrence of system calls differ from computer to computer. Apart from the identity of the system call, some more information may also be required. The type and nature of information vary according to OS and call. For example, to get input, we may need to specify the file or device to use as the source, and the address and length of the memory buffer into which the input should be read.

Three general methods are used to pass parameters to the operating system:

- The simplest approach is to pass the parameters in registers.



□ In some cases, there may be more parameters than registers. Then, the parameters are stored in a block or table in memory. And the address of the block is passed as a parameter in a register as shown in Figure 2.1. This is the approach taken by Linux. Parameters can also be placed, or pushed, onto the stack by the program, and popped of the stack by the operating system.

□ Some OS prefer the block or stack methods, because those approaches do not limit the number or length of parameters being passed. System calls can be grouped roughly into five major categories: process control, file management, device management, information maintenance, and communications. These are discussed in the following sections. Table 2.1 summarizes the types of system calls provided by OS.

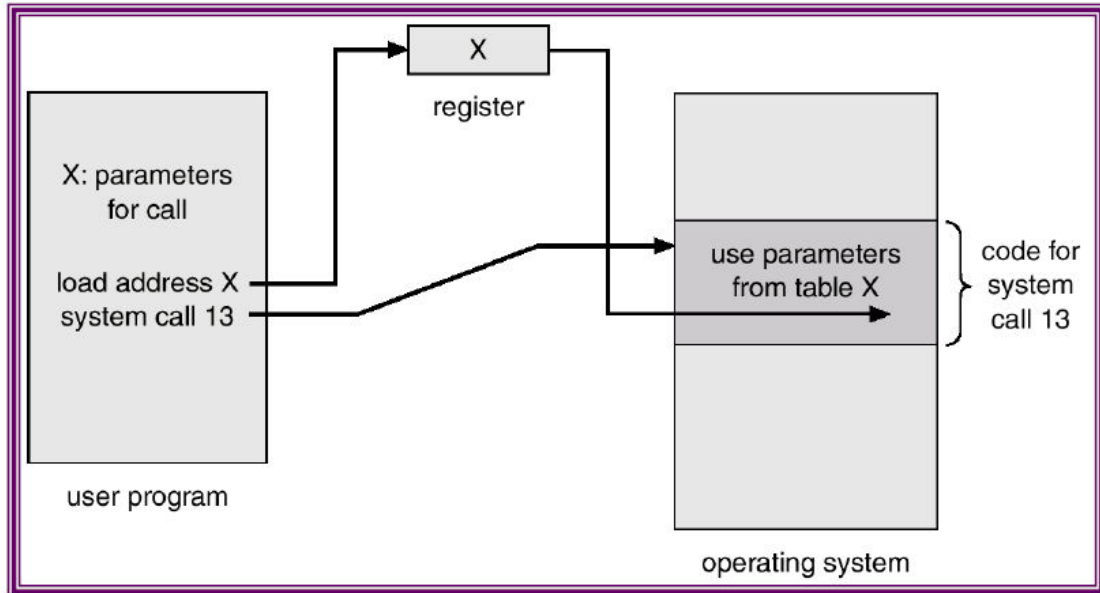


Figure 2.1 Passing of parameters as a table

6 (a) Explain in detail about virtual machines

10

- A virtual machine takes the layered approach to its logical conclusion.
- It treats hardware and the operating system kernel as though they were all hardware.
- A virtual machine provides an interface identical to the underlying bare hardware.
- The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory.
- The resources of the physical computer are shared to create the virtual machines. That is,
  - CPU scheduling can create the appearance that users have their own processor.
  - Spooling and a file system can provide virtual card readers and virtual line printers.
  - A normal user time-sharing terminal serves as the virtual machine operator's console.

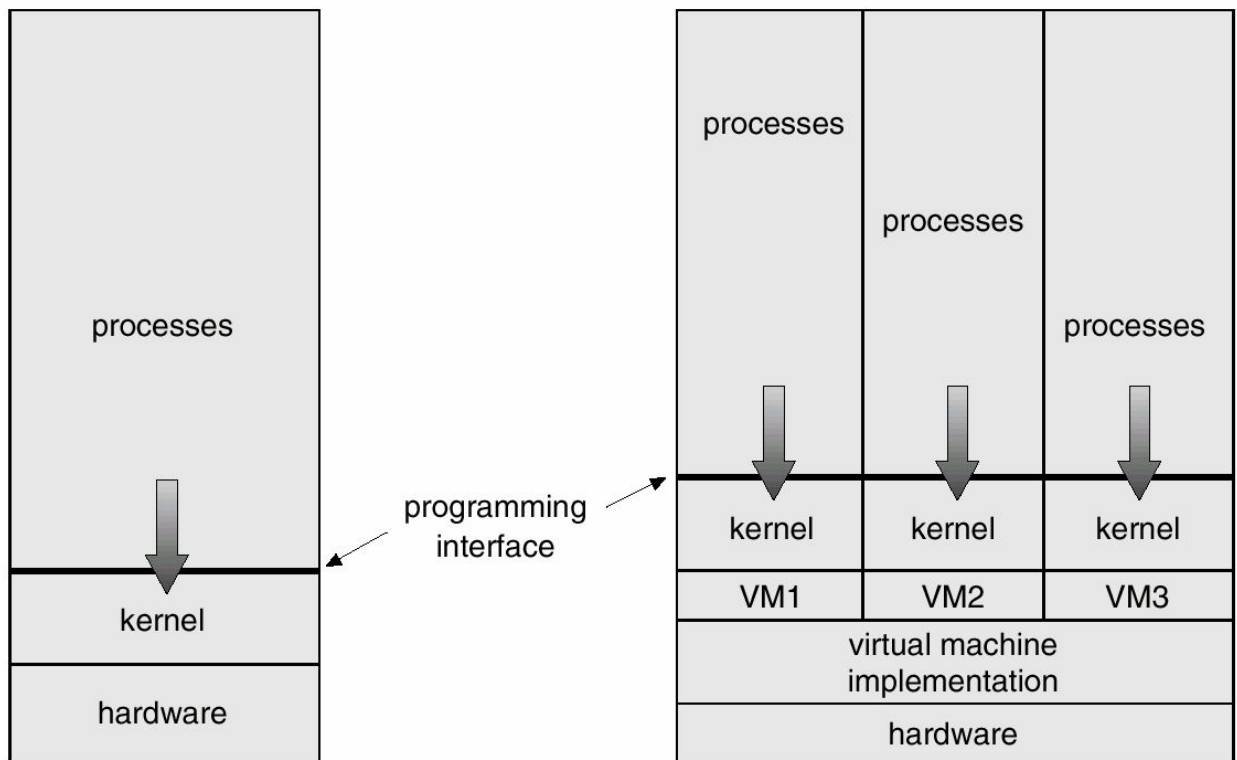
• Machine has two modes :

1. User Mode

2. Monitor mode

- VM Software run in monitor mode, since it is the OS
- VM itself can execute in only user mode.
- Both run in a physical user mode
- In real machine – transfer from user mode to monitor mode
- Virtual machine – transfer from virtual user mode to virtual monitor mode
- Though the virtual-machine (VM) concept is useful, it is difficult to implement due to the effort required to provide an exact duplicate of the underlying machine.
- Compared to actual I/O, the virtual I/O may take considerably more time, as it is interpreted.

- Also, since CPU is multiprogrammed among many virtual machines, the VM will slow down unpredictably.



#### Benefits

- The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines.
- This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect means for operating systems research and development.
- System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.

7 Explain any two I/O communication techniques with flowchart  
1.7.1 Programmed I/O

10

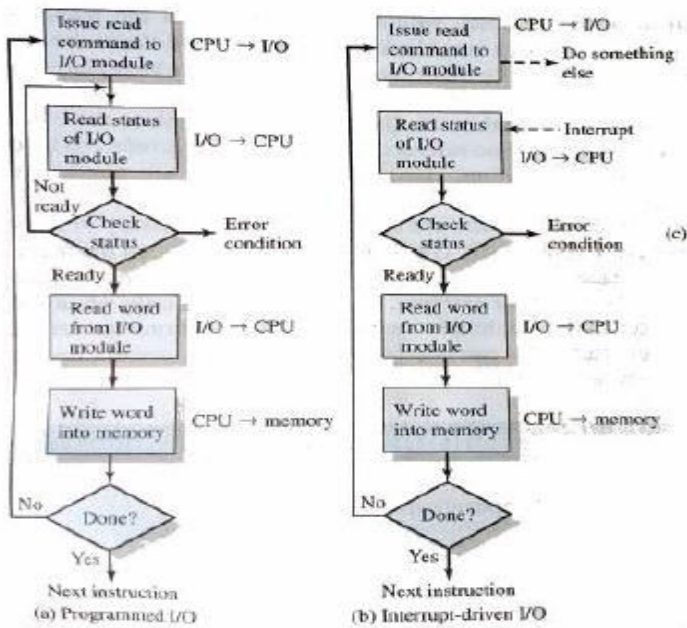
When the processor is executing a program and encounters an I/O instruction, then it will inform I/O module and executes that instruction. In case of programmed I/O, the I/O module performs the task but do not interrupt the processor about the completion of the task. Hence, the processor must periodically keep checking the I/O module for the completion of the task.

Thus, the processor is responsible for extracting/storing data from/to the main memory. So, the instruction set includes the I/O instructions in the following categories:

- Control: activates an external device and informs the action to be taken.
- Status: used to test various status conditions associated with I/O module.
- Transfer: to read/write data between processor registers and external devices.

Note that, the technique of programmed I/O is time-consuming and keeps the processor busy unnecessarily.

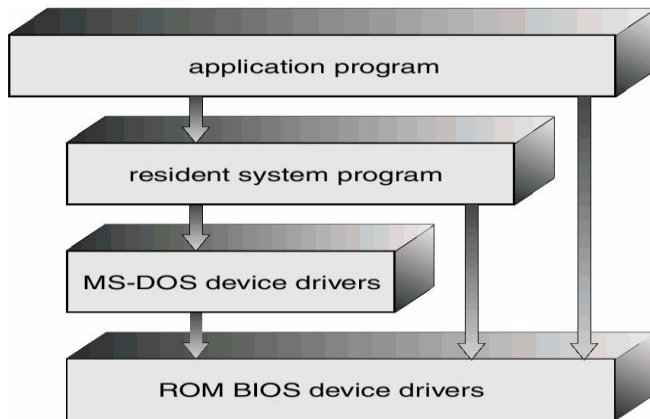
The programmed I/O technique will degrade the performance of the processor. An alternative way is to provide interrupt-driven I/O. In this technique, the processor will issue an I/O command to the I/O module and then continue its regular instruction execution. When the I/O module is ready, it will interrupt the processor. Then the processor will execute the requested task and then resume its former processing.

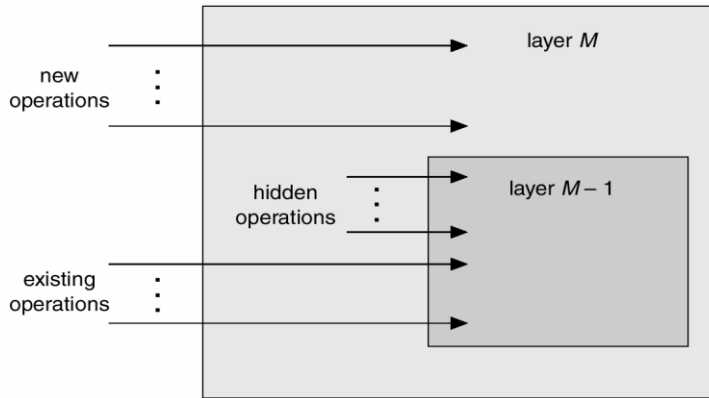


### Interrupt-Driven I/O

As it is observed, the programmed I/O technique will degrade the performance of the processor. An alternative way is to provide interrupt-driven I/O. In this technique, the processor will issue an I/O command to the I/O module and then continue its regular instruction execution. When the I/O module is ready, it will interrupt the processor. Then the processor will execute the requested task and then resume its former processing.

8 (a) Explain layer structure with a neat diagram.





In the layered approach, the OS is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.

The main advantage of layered approach is modularity. The layers are selected such that each uses functions (operations) and services of only lower-level layers. This approach simplifies debugging and system verification.

(b) Write a short note on microkernel

05

- All non-essential components of earlier UNIX kernel have been removed and only system and user level programs have been implemented. Hence, it became a smaller kernel.
- Microkernels provide minimal process and memory management, in addition to a communication facility.
- Communication takes place between user modules using message passing.
- The major benefits of using microkernels
  - easier to extend a microkernel
  - easier to port the operating system to new architectures
  - more reliable (less code is running in kernel mode)
  - more secure and reliable (services are running as user rather than kernel-processes)
- Tru64 UNIX – UNIX interface, implementation-Mach kernel
- QNX – realtime os based on microkernel
- Windows NT uses a hybrid structure (layering)
- Designed to run various applications, including win32, OS/2, and POSIX

9 (a) What is memory hierarchy? List and explain the category with neat diagram

06

To solve this problem, memory hierarchy has to be used instead of relying on a single memory component. A typical memory hierarchy would be as given in Figure 1.9. In this hierarchy, from top to bottom, the following occur:

(i) Decreasing cost per bit

(ii) Increasing capacity

(iii) Increasing access time

(iv) Decreasing frequency of access to the memory by the processor Hence, the smaller, expensive, faster memories are supplemented by larger, cheaper, slower memories. The levels of memory hierarchy are explained hereunder.

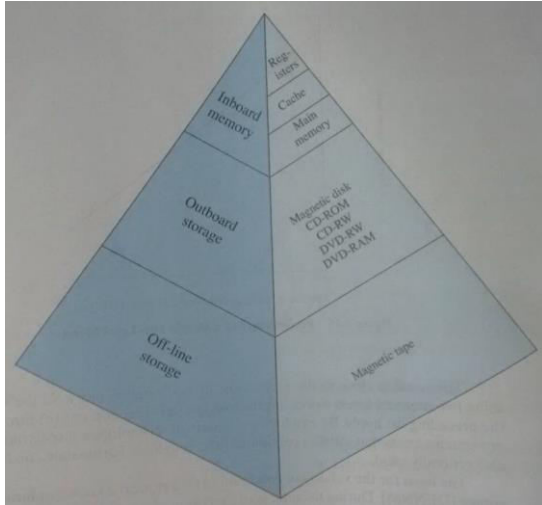
□ Registers: Generally every processor contains a few dozen or hundreds of registers which are faster, smaller but expensive.

□ Cache Memory: It is not usually visible to the programmer, but visible only to the processor. It is used for the movement of data between main memory and processor registers.

□ Main Memory: It is the principal internal memory system of the computer. Each location in the main memory has a unique address. Most of the machine instructions refer to one or more main memory

addresses. Main memory is usually extended with a higher speed, smaller cache.

□ Secondary/Auxiliary Memory: The next two levels of the hierarchy falls into this category. The data are stored permanently on external storage devices like hard disk,



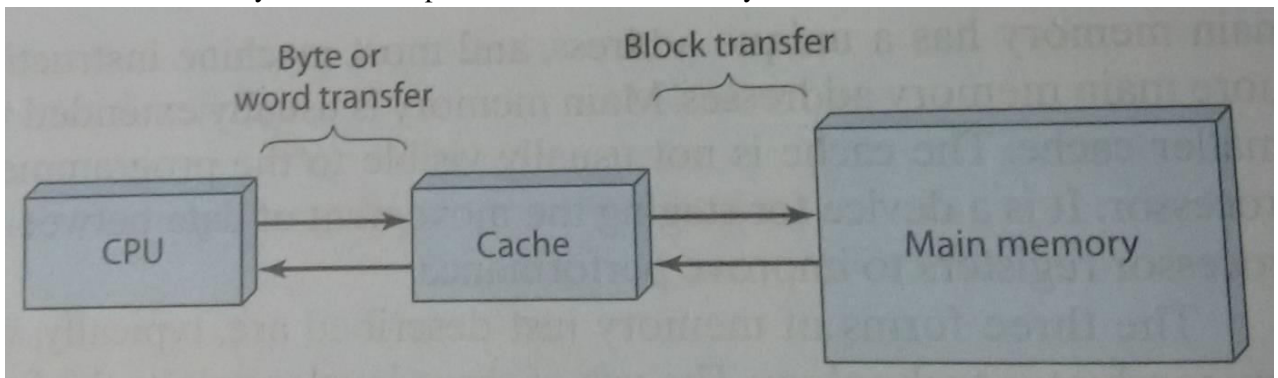
(b) Write a short note on cache memory

04

### 1.6 CACHE MEMORY

Since cache memory plays very important role in the performance of the processor and managing the memory hardware, it is being discussed in detail here.

On every instruction cycle, the processor fetches the memory at least once. If the instruction contains operands or it needs to store some data, then every fetch may need to access memory more than once. Thus, the processor speed is always restricted by the memory cycle time. And, in almost all computers, the processor speed is much higher than that of memory cycle speed. Hence, an intermediate repository of instructions is thought of to keep a portion of memory that needs to be executed by the processor. Such a small and fast memory between the processor and main memory is called as cache.



Cache Design

The design of a cache memory has to consider following aspects:

- Cache size: Small size caches have significant impact on the performance of the processor.
- Block size: It indicates the amount of data exchanged between cache and main memory. Optimum selection of this size is essential. Because, if the block size is too small, then it cannot hold many instructions and hence the main memory has to be hit more number of times. Whereas, if the block size is too large, then it becomes almost like a main memory and the very usage of cache will be void.
- Mapping function: This function determines the location in the cache to be occupied by the block. It has two constraints: (i) while one block is read, another may need to be replaced. (ii) As mapping function

becomes more flexible, the design circuitry becomes complex.

□ Replacement algorithm: This algorithm decides which block has to be replaced when a new block is loaded into the cache. And the design of this algorithm should work within the constraints of mapping function. In most of the cases, least-recently used (LRU) method is applied.

□ Write policy: If the contents of the block in the cache are modified, the same has to be written inside the main memory. The write policy indicates when the memory write operation has to take place.

10. Consider the following set of processes with given length of CPU burst:

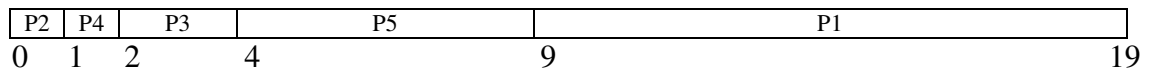
10

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

All processes arrived at time 0 in the given order.

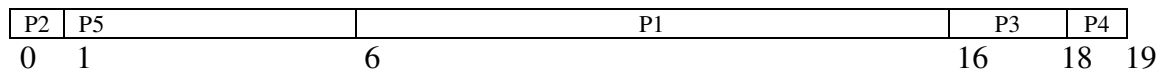
- Draw Gantt chart using SJF (non-preemptive), Priority (Non-preemptive) [Smallest number implies highest priority], and Round Robin [Quantum-2 ms] scheduling policies.
- Find the average waiting time for each scheduling policy.

### SJF (Non preemptive)



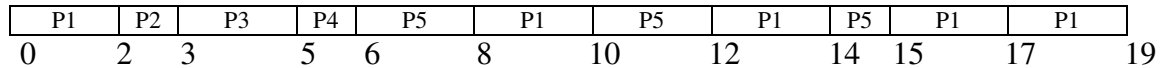
$$AWT = 9+0+2+1+4 = 16/5 = 3.2 \text{ ms}$$

### Priority (Non-preemptive)



$$AWT = 6+0+16+18+1 = 41/5 = 8.2 \text{ ms}$$

### Round Robin



$$\text{Waiting Time for P1} = 0+(8-2)+(12-10)+(15-14) = 0+6+2+1=9$$

$$\text{Waiting Time for P2} = 2$$

$$\text{Waiting Time for P3} = 3$$

$$\text{Waiting Time for P4} = 5$$

$$\text{Waiting Time for P5} = 6+(10-8)+(14-12) = 6+2+2=10$$

$$AWT = 9+2+3+5+10 = 29/5 = 5.8 \text{ ms}$$