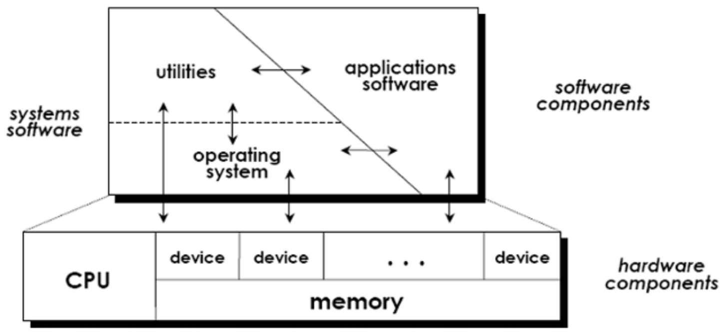
UNIT 05 SHORTNOTE



Initial operations of a computer

- When powers on, CPU activates the BIOS.
- The BIOS runs POST (Power On Self-Test) using CMOS memory to check hardware functionality.
- BIOS reads the MBR (Master Boot Record) from the boot drive using the bootstrap loader.
- The operating system is loaded from the boot drive
- into RAM.
- The operating system takes control and displays the user interface.

This process is called **booting**, which means loading the operating system into RAM.

Categories of System Software

1. **Operating System (OS)**
   - Manages hardware and software, allowing user to access and use the computer's functions.
   - Acts as an interface between system software, application software, and hardware.
2. **Utility Software**
   - Manages and analyzes computer resources.
   - Performs specific tasks such as:
     - **Anti-Virus Software**
     - **Disk Formatting**
     - **Disk Defragmenters**
     - **Disk Cleaners**.
3. **Language Translators**
   - Convert high-level or assembly languages into machine language (binary code). Examples:
     - **Compiler**: Converts the entire program at once, reporting errors after.
     - **Interpreter**: Translates and executes code line by line, reporting errors immediately.
     - **Assembler**: Converts assembly language into machine language.

Application Software

- Runs on the Operating System, used for performing user-specific tasks such as creating documents, performing calculations, Playing games

Evolution of Operating System

1) No OS (late 1940s – mid 1950s) → Manual Program Scheduling, Uni-programming, processor sat idle when loading programs and doing I/O.

2) Simple Batch Systems → No direct access to hardware, Uni-programming, High response time, Processor sat idle during I/O

3) Multi-Programmed batch Systems → Introduced in the 3rd generation, focuses on reducing processor idle time during I/O operations. By partitioning memory to hold multiple programs, the OS switches the processor to another program when one is waiting for I/O.

4) Time Sharing Systems → Introduced to minimize the response time and maximize the user interaction during program execution, Uses context switching, Enables to share the processor time among multiple programs o Rapidly switching among programs.

Different types of Operating Systems

1. Based on the users: Single user, Multi User
2. Based on Number of tasks: Single Task, Multi Task

Different types of modern Operating Systems

- Single user-single task
- Single user-Multi task,
- Multi user-Multi task
- Multi-threading
- Real Time
- Time Sharing Systems

Important functions of an operating System

1. User Interfacing - provides an easy way for users to interact with the hardware, hiding its complexity.

- **Command Line Interface (CLI)** – Commands are entered using the keyboard, Requires correct syntax, Not visually appealing, Uses very little system resources.

- **Graphical User Interface (GUI).** – Uses WIMP: [Windows, Icons, Menus, and Pointer], Visually appealing and user-friendly, Requires more system resources, Includes a cursor for navigation, Slower compared to CLI.

2. Process Management – OS divides a program or task into multiple processes during execution, Only one process runs on the CPU at a time, enables multitasking,

3. Memory Management – OS handles memory allocation for tasks, stores data before processing, keeps results after processing, and frees memory when applications are closed. OS optimizes the use of primary and secondary memory to enhance computer performance.

4. **Device Management** → OS manages peripheral devices used for input and output, Plug and Play features because the OS includes built-in device drivers.

5. **Security Management** → OS manages user permissions, maintains log files, and allows the creation of profiles with usernames, passwords, and different access levels.

6. **Network Management** → Manages network connections, Handles IP address configurations, device settings of connecting in a network. Use of protocols, monitors network performance and usage, Provides tools for diagnosing and troubleshooting network issues.

7. **File Management**

Files : A file is a named collection of related information, usually a sequence of bytes.

A file can be viewed in two different ways.

1. **Logical** (programmer's) view: how the users see the file.
   liners collection of records.
   Image File – cells(pixels) of intensity values
   Linear sequence of bytes.
2. **Physical** (operating system) view: how the file is stored on secondary storage.

Directories are continues used to organize files logically.

File Attributes : file name , type, location(s), organization, access permissions, time and date of creation, File size...
File Types : One of implementation techniques of file type is to include the type as an extension to the file name.
File can be classified into various types based on the content - Executable(.exe), Text(.txt, .docx), Image(.png)...

FAT (File Allocation Table)	NTFS (New Technology File System)
Up to 4 GB per file	Supports files larger than 4 GB up to 16 TB
Uses a FAT to keep track of files in storage	Complex, with advanced features. Supports Unicode
Low security	High security
Up to 8 characters (8 for file name and 3 for extension)	Up to 255 characters
No/low compression and encryption	Supports compression and encryption
MS-DOS, Win95/98/ME/XP	Windows XP, Vista, Windows 7, 8, etc.

Fille Access Methods

1. Sequential Method → All the data are in sequential Manner. Basically, use to take backups, Not for running applications. E.g. – Magnetic Tapes

2. Direct Access Method → Data is stored in random locations on the disk. Accessing any data takes almost the same amount of time (Direct Access/Random Access). E.g. - All types of memory except magnetic tapes.

The Utility Programs in an OS

• Utility programs are used to improve the efficiency of a computer.

1. Backup Software → Backups are copies of important data, often stored on removable media.

2. Disk Scanner → Disk scanning detects and fixes physical errors on the disk surface. It repairs damaged data (1's and 0's) on the disk.

3. File Compression → Minimizes the size of a file without damaging its content. Saves hard disk space and increases file transfer speed.

4. Anti-virus Software → Protects the computer by detecting and removing malicious software (malware).

5. Disk Partitioning → Divides a single disk into multiple logical drives. Usually done before installing the operating system.

6. Disk Formatting → Formatting prepares a disk to store data. A disk cannot be used until it is formatted.

Fragmentation → Fragmentation occurs when data blocks are scattered across the disk. This reduces the efficiency of file access, making the system slower.

7. Defragmentation → Defragmentation reorganizes scattered file blocks to minimize fragmentation.

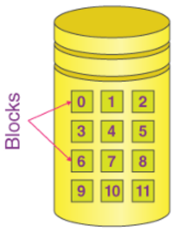
Compaction is the process of collecting all free memory space into a single large block. It helps allocate memory to processes by eliminating scattered fragments.

Internal Fragmentation → Wasted memory inside allocated blocks. Happens when fixed-sized memory blocks are given to processes, and the process doesn't use the entire block.

External Fragmentation → Wasted memory outside allocated blocks. Occurs when free memory is split into small, non-contiguous blocks, making it hard to allocate for new processes even if enough total space exists.

File Storage Management

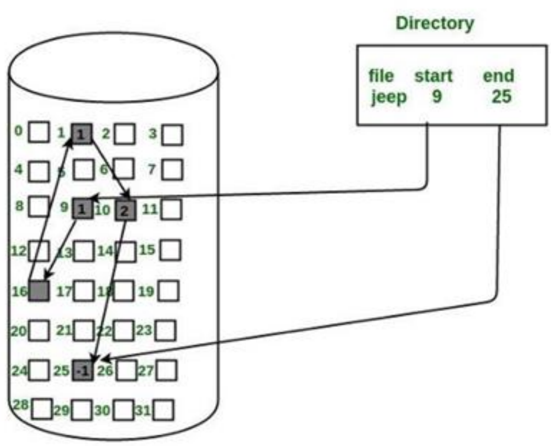
1. Contiguous Allocation → Allocate disk space as a collection of adjacent/contiguous blocks. This technique needs to keep track of unused disk space. Easy Access, Extending file size is difficult, External fragmentation (free unusable space between allocation)



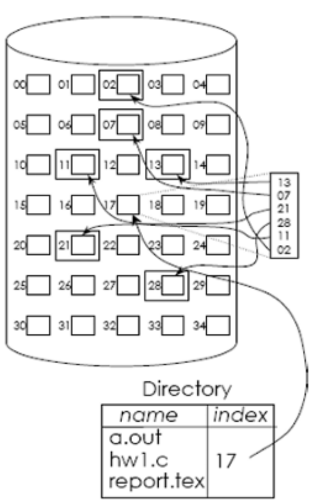
File Name	Start	Length	Allocated Blocks
abc.text	0	3	0, 1, 2
video.mp4	4	2	4, 5
jtp.docx	9	3	9, 10, 11

Directory

2. Linked Allocation → solves all problems of contiguous allocation. Inside each block a link is maintained to point to where the next block of the file is. It Avoids fragmentation but has slower access due to the need to follow links.



3. Indexed Allocation → Creates a table of pointers (index) at the time of the file creation. This table is modified as new blocks are allocated for the file or removed from the file. The index table is also saved in a block/s.
Example : UNIX file system



The two main parts of an Operating System (OS)

1. Kernel → The core component of the OS that directly interacts with hardware. Responsible for managing system resources. Provides essential services to the rest of the OS and applications, like process management, memory management, and device control.

2. Shell (or User Space) → The interface between the user and the OS. Includes utilities and programs that allow users to interact with the system (e.g., command line interface, graphical user interface). Provides application-level functionality, such as file management and task execution.

What is Process? → Process is a fundamental concept in modern operating systems. A process is basically a program in execution. Process is not a program. A program may have many processes.

Types of processes

- I/O bound processes
- Processor bound processes

A process must have (at least) → ID, Executable code, Data needed for execution, Execution context (PC, priorities, waiting for I/O or not)

Reasons for process creation → New batch job, User starts a program, OS creates process to provide a service, Running program starts another process

Reasons for process termination → Normal termination, Execution time-limit exceeded, A resource requested is unavailable, An execution error, A memory access violation, An operating system or parent process request, Parent process has terminated.

❖ On process termination, OS reclaims all resources assigned to the process.

Interrupts → Interrupt is an event that alters the sequence of execution of process. can occur due to a time expiry an OS service request I/O completion. For example when a disk driver has finished transferring the requested data, it generates an interrupt to the OS to inform the OS that the task is over.

Interrupt Handling → Generally I/O models are slower than CPU. After each I/O call, CPU has to sit idle until I/O device complete the operation, and so processor saves the status of current process and executes some other process. When I/O operation is over, I/O devices issue an interrupt to CPU then stores the original process and reserves execution.

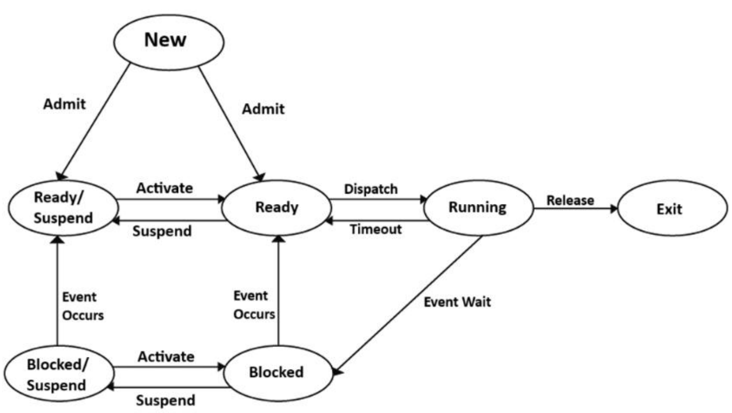
Process Management → In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called *process scheduling*. • Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.

• Allocates the processor (CPU) to a process.

• De-allocates processor when a process is no longer Required

Swapping is a mechanism in which a process can be swapped temporarily out of main memory (or move) to secondary storage (disk) and make that memory available to other processes. At some later time, the system swaps back the process from the secondary storage to main memory.

Seven State Process Transition diagram



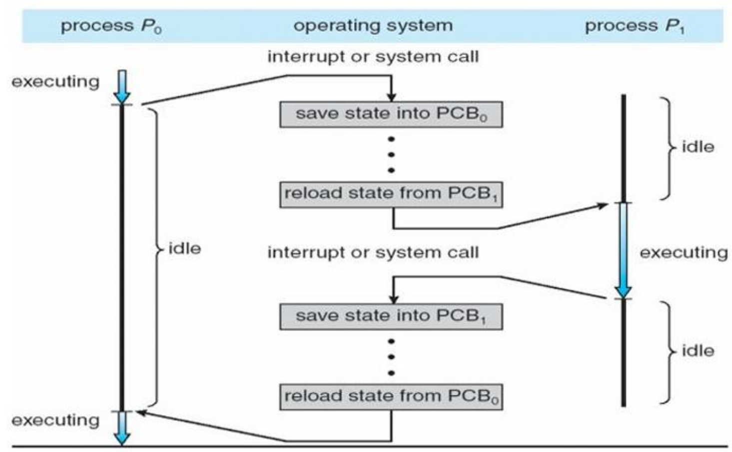
Process Control Block (PCB)

A Process Control Block is a data structure maintained by the Operating System for every process. The PCB is identified by a process ID (PID). A PCB keeps all the information needed to keep track of a process.

A simplified diagram of a PCB:

Process ID
State
Pointer
Priority
Program Counter
CPU Register
I/O Information
Accounting Information
etc

Context switching is the process of storing a CPU's state in a process control block so that a process can resume execution later, enabling multiple processes to share a single CPU in a multitasking operating system. During a context switch, the scheduler saves the current process's CPU registers and loads the registers of the next process. While essential for multitasking, context switching adds overhead and can impact performance due to the time required to save and restore modern processors' numerous registers.



Types of Scheduling

1. Long-term scheduling (Job scheduling): It determines which programs are admitted to the system for processing. Job scheduler selects processes from the queue and loads them into memory for execution.

2. Medium-term scheduling: Medium term scheduling is in charge of swapping processes between the main memory and the secondary storage.

3. Short-term scheduling (low-level scheduling): Determines which ready process will be assigned the CPU when it next becomes available.

Long Term Scheduler	Short Term Scheduler	Medium Term Scheduler
Job Scheduler	CPU scheduler	Processes swapping scheduler
Selects processes from a pool and loads them into the memory for execution	Selects those processes which are ready to execute for dispatching	Swapped out/Re-introduces the processes into memory and execution can be continued.
Controls the degree of multiprogramming	Provides lesser control over the degree of multiprogramming	Controls the degree of multiprogramming
Speed is lesser than short term scheduler	Speed is fastest among other two	Speed is in between (short and long term schedulers)

Process Schedulers → Assigning the processor to the processes.

• **Turnaround time** : Time required for a particular process to complete, from submission time to completion.
• **Response time** : The time taken in an interactive program from the issuance of a command to the commence of a response to that command.
• **Throughput** : Number of processes completed per unit time. May range from 10 / second to 1 / hour depending on the specific processes.
• **Waiting time** : How much time a process spends in the ready queue waiting its turn to get on the CPU.
• **Burst Time**: Time required by a process for CPU execution.

Scheduling Policies

• **Non-preemptive** : Once a process is in the running state, it will continue until it terminates or blocks itself for I/O.
• **Preemptive** : Currently running process may be interrupted and moved to the Ready state by the OS. Allows for better service since any one process cannot monopolize the processor for very long.

Scheduling Algorithms

→ The Purpose of a Scheduling algorithm
Maximum CPU utilization, Fare allocation of CPU
Maximum throughput, Minimum turnaround time,
Minimum waiting time 6. Minimum response time

1. First Come First Serve algorithm: The FCFS scheduling algorithm processes tasks in order they arrive. The first task in the queue gets the CPU first. However, if the first task takes a long time, others may have to wait, leading to starvation.
Disadvantages → A process runs until it finishes, delaying others. Long tasks in the beginning can make shorter ones wait too long. High average waiting time compared to other methods, despite being simple to use.

2. Shortest Job First (SJF) Algorithm: This selects the process with the smallest burst time for execution. It minimizes the average waiting time by prioritizing shorter tasks. However, it may cause starvation if longer tasks keep being delayed.

3. Round Robin Scheduling Algorithm: widely used in operating systems. It is a preemptive version of the First Come First Serve (FCFS) algorithm, designed for time-sharing. In RR, each process is given CPU access for a fixed time slice, called the time quantum. If a process finishes during its time quantum, it terminates; otherwise, it returns to the ready queue to wait for its next turn.

4. Priority Scheduling Algorithm: each process is assigned a priority number. Depending on the system, a lower number may indicate higher priority, or vice versa. The process with the highest priority is given the CPU first.

A **Deadlock** happens when two or more processes are waiting for resources that other processes are holding, and none of them can continue. It's like a traffic jam where every car is waiting for another to move, but no one can proceed. Since both processes hold a resource the other process needs, the processes will have to wait indefinitely. OS should avoid deadlocks.

Spooling stands for Simultaneous Peripheral Operations On-Line. It is the process of storing data temporarily in a buffer (like print jobs in a queue) before sending it to a device. It allows slower devices, like printers, to handle tasks at their own speed without holding up the CPU or other processes.

Buffering Data → Device Synchronization → Parallel Operations

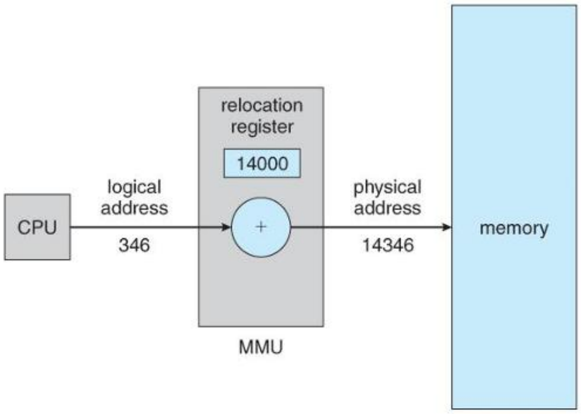
Memory management is an operating system function that efficiently handles primary memory by tracking all memory locations, allocating memory to processes, and moving them between main memory and disk during execution. It ensures efficient usage by deciding how much memory to allocate, determining which process gets memory and when, and updating the status when memory is freed or unallocated.

Memory Management Unit (MMU) - This is the hardware device that translates virtual addresses (used by programs) into physical addresses (used by memory). This ensures efficient memory management and security in a system.

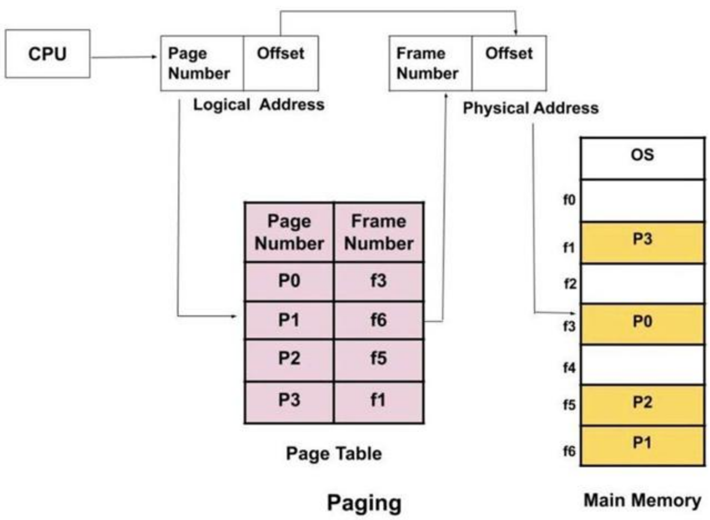
How MMU Works:

• **Address Translation:** The MMU adds the value in the relocation (base) register to each logical address generated by the program. This translated address is the physical address, where data is actually stored in memory.
• **Logical vs. Physical Addresses:** A program works only with logical addresses. The program never directly interacts with real physical memory addresses.

Physical Address=Base Register Value + Logical Address



Paging allows a process's logical address space to be non-contiguous by dividing physical memory into fixed-sized blocks called frames and logical memory into equally sized pages. The operating system tracks free frames and uses a page table to map logical addresses to physical addresses. To run a program, the OS loads its pages into available frames, though this can lead to internal fragmentation.

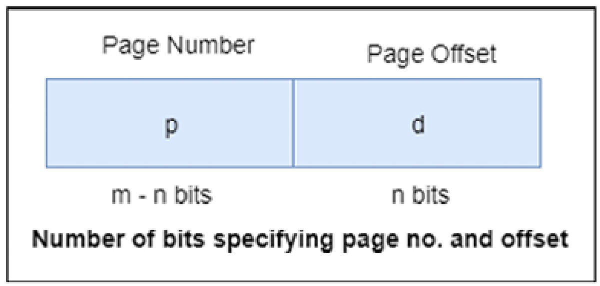


What is Mapping? → refers to the process of converting logical addresses (used by programs) into physical addresses (used by hardware). This is managed by the OS during memory allocation. At runtime, the Memory Management Unit (MMU) – a hardware component— handles this mapping dynamically.

Page Table - A special data structure used in virtual memory systems to map virtual addresses to physical addresses. It plays a crucial role in memory management. Role of the Page Table: The page table contains mappings (or relationships) between pages and frames. Each logical address maps to a specific physical frame via this table.

Virtual memory allows a computer to use more memory than physically available by using disk space to emulate RAM. It enables larger programs and multitasking by loading only required parts of programs into memory, supporting applications larger than physical memory, and allowing multiple programs to run simultaneously. Developers benefit from simplified memory management, and shared memory regions enhance efficiency.

Logical Address



Physical Address