UNIT-4
Memory management

M**emory management** is the function responsible for managing the computer's primary memory.

The memory management function keeps track of the status of each memory location, either *allocated* or *free*.

It determines how memory is allocated among competing processes, deciding which gets memory, when they receive it, and how much they are allowed.

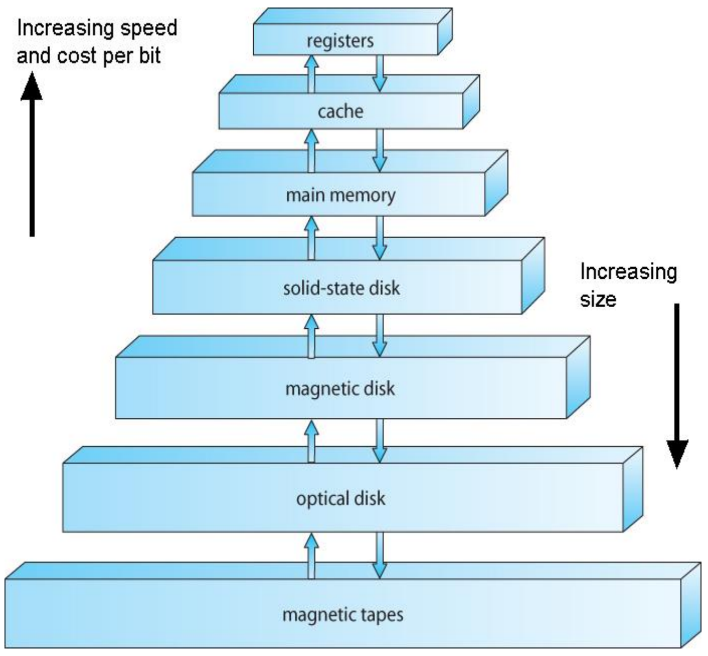
When memory is allocated it determines which memory locations will be assigned. It tracks when memory is freed or *unallocated* and updates the status.

Functions of Memory management

- Keeping track of each part of memory i.e. Either allocated or free.
- Allocation—if the process request for memory then the question arises whether the memory is free is not.
- If memory is free ---How to allocate it
  When to allocate it
  To whom allocate it
- If not free then what to do?
- Deallocation –After completion the task, the process has to release its memory.
- Protection and sharing is also provided in multiprogramming OS.

Memory Hierarchy

- The memory in a computer can be divided into five hierarchies based on the speed as well as use. The processor can move from one level to another based on its requirements.
- The five hierarchies in the memory are registers, cache, main memory, magnetic discs, and magnetic tapes.
- The first three hierarchies are volatile memories which mean when there is no power, and then automatically they lose their stored data. Whereas the last two hierarchies are not volatile which means they store the data permanently.

- 

Advantages of Memory Hierarchy

- The need for a memory hierarchy includes the following.
- Memory distributing is simple and economical
- Removes external destruction
- Data can be spread all over
- Permits demand paging & pre-paging
- Swapping will be more proficient
- Memory Hierarchy Design
- The memory hierarchy in computers mainly includes the following.

- Registers
- Usually, the register is a static RAM or SRAM in the processor of the computer which is used for holding the data word which is typically 64 or 128 bits. The program counter register is the most important as well as found in all the processors. Most of the processors use a status word register as well as an accumulator. A status word register is used for decision making, and the accumulator is used to store the data like mathematical operation. Usually, computers like complex instruction set computers have so many registers for accepting main memory, and RISC- reduced instruction set computers have more registers

  .
- Cache Memory

  Cache memory can also be found in the processor, however rarely it may be another IC (integrated circuit) which is separated into levels. The cache holds the chunk of data which are frequently used from main memory. When the processor has a single core then it will have two (or) more cache levels rarely. Present multi-core processors will be having three, 2-levels for each one core, and one level is shared.

- Main Memory

  The main memory in the computer is nothing but, the memory unit in the CPU that communicates directly. It is the main storage unit of the computer. This memory is fast as well as large memory used for storing the data throughout the operations of the computer. This memory is made up of RAM as well as ROM.

- Optical disk

  In the computer system, the optical disk is the recording technology. It is flat, a circular disk which can encode the binary data in the form of pits and lands on the special material. It is the computer storage disk that stores the data digitally and uses a laser beam to read and write the data.

  The Optical disk is primarily used as the portable and secondary storage device. This disk can store more data than the previous generation of magnetic storage media. The compact disk, digital versatile or video disk and the Blue-ray disk are the most commonly used form of optical disks. These types of the disk are generally used for:

  1. Distribute the software to customers.
  2. Store a large amount of data in the form of music, images, and videos.
  3. Transfer the data to different computer systems or devices.
  4. Back up the data from the local machine.

- Magnetic Disks

  The magnetic disks in the computer are circular plates fabricated of plastic otherwise metal by magnetized material. Frequently, two faces of the disk are utilized as well as many disks may be stacked on one spindle by read or write heads obtainable on every plane. All the disks in computer turn jointly at high speed. The tracks in the computer are nothing but bits which are stored within the magnetized plane in spots next to concentric circles. These are usually separated into sections which are named as sectors.

  Magnetic Tape

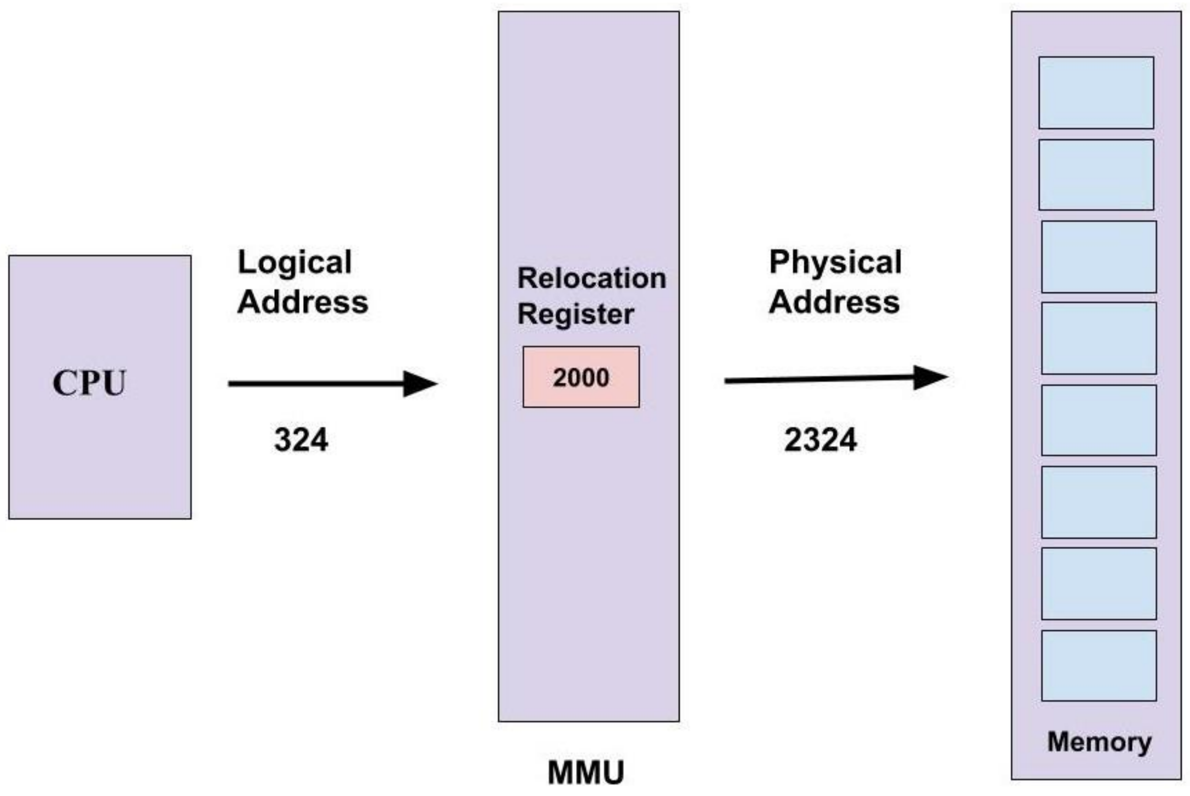
- This tape is a normal magnetic recording which is designed with a slender magnetizable covering on an extended, plastic film of the thin strip. This is mainly used to back up huge data. Whenever the computer requires to access a strip, first it will mount to access the data. Once the data is allowed, then it will be unmounted. The access time of memory will be slower within magnetic strip as well as it will take a few minutes for accessing a strip.

Logical and Physical Address in Operating System

There are two types of addresses that are used for memory in the operating system i.e. the physical address and logical address about which we will learn in this blog. So, let's get started.

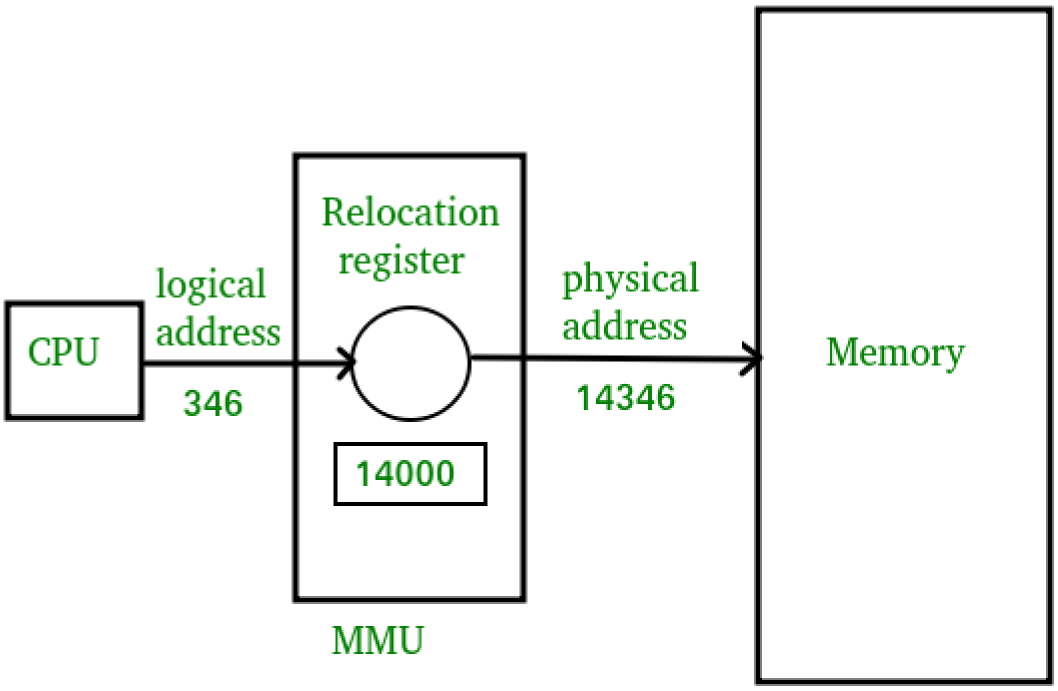
- **Logical Address** is generated by CPU while a program is running. The logical address is virtual address as it does not exist physically, therefore, it is also known as Virtual Address. This address is used as a reference to access the physical memory location by CPU. The term Logical Address Space is used for the set of all logical addresses generated by a program's perspective.
  The hardware device called Memory-Management Unit is used for mapping logical address to its corresponding physical address.

  The diagram below explains how the mapping between logical and physical addresses is done.



- The CPU generates the logical address(here, 324).
- The MMU will generate the base address (here, 2000) which is stored in the Relocation Register.
- The value of Relocation Register(here, 2000) is added to the logical address to get the physical address. i.e. 2000+324= 2324(Physical Address).

- **Physical Address** identifies a physical location of required data in a memory. The user never directly deals with the physical address but can access by its corresponding logical address. The user program generates the logical address and thinks that the program is running in this logical address but the program needs physical memory for its execution, therefore, the logical address must be mapped to the physical address by MMU before they are used. The term Physical Address Space is used for all physical addresses corresponding to the logical addresses in a Logical address space.



- **Comparison Chart:**

PARAMENTER	LOGICAL ADDRESS	PHYSICAL ADDRESS
• Basic	• generated by CPU	• location in a memory unit
• Address Space	• Logical Address Space is set of all logical addresses generated by CPU in reference to a program.	• Physical Address is set of all physical addresses mapped to the corresponding logical addresses.
• Visibility	• User can view the logical address of a program.	• User can never view physical address of program.
• Generation	• generated by the CPU	• Computed by MMU
• Access	• The user can use the logical address to access the physical address.	• The user can indirectly access physica
• Basic	• It is the virtual address generated by CPU	• The physical address is a location in a memory unit.

PARAMENTER	LOGICAL ADDRESS	PHYSICAL ADDRESS
• Address Space	• Set of all logical addresses generated by CPU in reference to a program is referred as Logical Address Space.	• Set of all physical addresses mapped to the corresponding logical addresses is referred as Physical Address.
• Visibility	• The user can view the logical address of a program.	• The user can never view physical address of program
• Access	• The user uses the logical address to access the physical address.	• The user can not directly access physical address.
• Generation	• The Logical Address is generated by the CPU	• Physical Address is Computed by MMU

What is Swapping

Swapping is a memory management scheme in which any process can be temporarily swapped from main memory to secondary memory so that the main memory can be made available for other processes. It is used to improve main memory utilization. In secondary memory, the place where the swapped-out process is stored is called swap space.

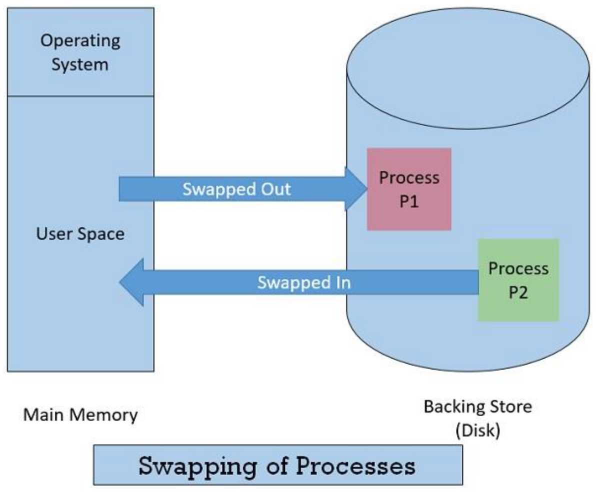
The purpose of the swapping in operating system is to access the data present in the hard disk and bring it to RAM so that the application programs can use it. The thing to remember is that swapping is used only when data is not present in RAM.

The concept of swapping has divided into two more concepts: Swap-in and Swap-out.

- Swap-out is a method of removing a process from RAM and adding it to the hard disk.
- Swap-in is a method of removing a program from a hard disk and putting it back into the main memory or RAM.

Example : Suppose the user process's size is 2048KB and is a standard hard disk where swapping has a data transfer rate of 1Mbps. Now we will calculate how long it will take to transfer from main memory to secondary memory.

User process size is 2048Kb
Data transfer rate is 1Mbps = 1024 kbps
Time = process size / transfer rate
= 2048 / 1024
= 2 seconds
= 2000 milliseconds
Now taking swap-in and swap-out time, the process will take 4000 milliseconds.



Benefits of Swapping

1. It helps the CPU to manage multiple processes within a single main memory.
2. It helps to create and use virtual memory.
3. Swapping allows the CPU to perform multiple tasks simultaneously. Therefore, processes do not have to wait very long before they are executed.
4. It improves the main memory utilization.
5. It offers a higher degree of multiprogramming.
6. Allows dynamic relocation.

Disadvantages of Swapping

1. If the computer system loses power, the user may lose all information related to the program .
2. If the swapping algorithm is not good, then it decreases the overall processing performance.
3. Inefficiency may arise in the case where a resources or a variable is commonly used by the processes which are participating in the swapping process.

Note:

In a single tasking operating system, only one process occupies the user program area of memory and stays in memory until the process is complete.

In a multitasking operating system, a situation arises when all the active processes cannot coordinate in the main memory, then a process is swap out from the main memory so that other processes can enter it.

What is Memory allocation?

Memory allocation is a process by which computer programs and services are assigned with physical or virtual memory space.

Memory allocation is the process of reserving a partial or complete portion of computer memory for the execution of programs and processes. Memory allocation is achieved through a process known as memory management.

Memory allocation is a process by which computer programs are assigned memory or space.

Main memory is divided into two types of partitions

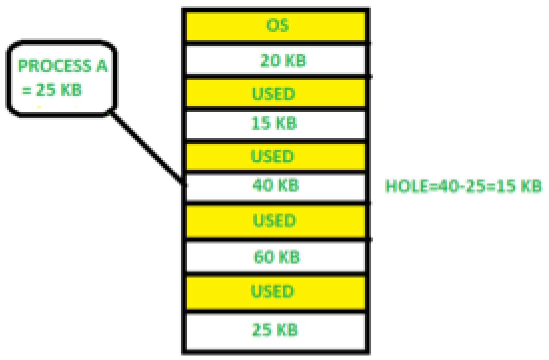
1. **Low Memory** - Operating system resides in this type of memory.
2. **High Memory**- User processes are held in high memory.

Partition Allocation

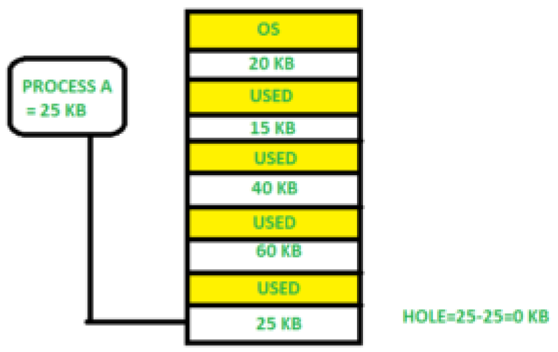
Memory is divided into different blocks or partitions. Each process is allocated according to the requirement. Partition allocation is an ideal method to avoid internal fragmentation.

The various partition allocation schemes :

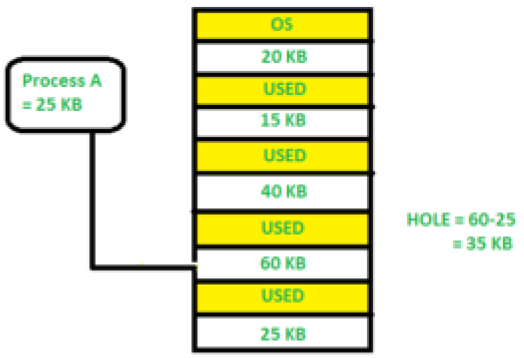
- **First Fit**: In this type fit, the partition is allocated, which is the first sufficient block from the beginning of the main memory.



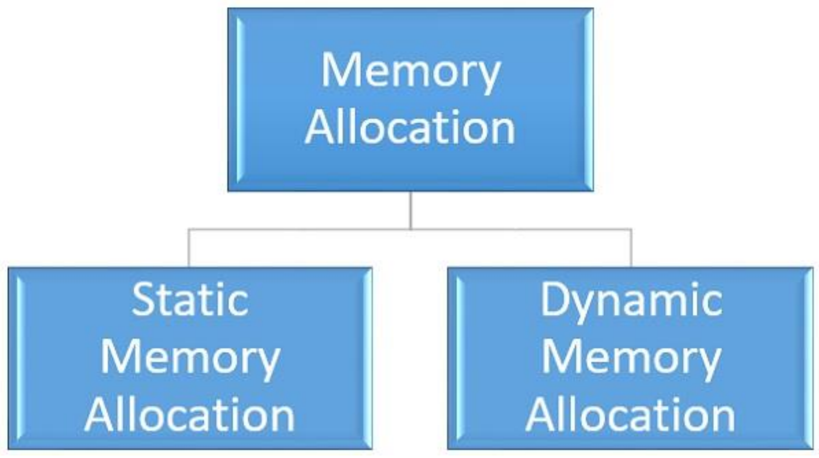
- **Best Fit:** It allocates the process to the partition that is the first smallest partition among the free partitions.



- **Worst Fit:** It allocates the process to the partition, which is the largest sufficient freely available partition in the main memory.
- **Next Fit:** It is mostly similar to the first Fit, but this Fit, searches for the first sufficient partition from the last allocation point.



Memory Allocation



Types of Memory Allocation

1. **Static Memory Allocation**

Static memory allocation is performed when the compiler compiles the program and generate object files, linker merges all these object files and creates a single executable file, and loader loads this single executable file in main memory, for execution. In static memory allocation, the size of the data required by the process must be known **before** the execution of the process initiates.

Static memory allocation method does not need any memory allocation operation during the execution of the process.

So, it leads to **faster** execution of a process.

2. **Dynamic Memory Allocation**

Dynamic memory allocation is performed while the program is in execution.

The actual size, of the data required, is known at the run time so, it allocates the **exact** memory space to the program thereby, reducing the memory wastage.

Dynamic memory allocation provides **flexibility** to the execution of the program. As it can decide what amount of memory space will be required by the program.

If the program is large enough then a dynamic memory allocation is performed on the different parts of the program, which is to be used currently. This reduces memory wastage and improves the performance of the system.

Allocating memory dynamically creates an overhead over the system.

The dynamic memory allocation is flexible but slower than static memory allocation.

Advantages of static memory allocation

Static Memory Allocation

1. Static memory allocation provides an **efficient** way of assigning the memory to a process.
2. All the memory assigning operations are done before the execution starts. So, there are *no* **overheads** of memory.
3. Static memory allocation provides **faster** execution, as at the time of execution it doesn’t have to waste time in allocation memory to the program.

Advantages of Dynamic Memory Allocation

1. Dynamic memory allocation provides a **flexible** way of assigning the memory to a process.
2. Dynamic memory allocation also reduces memory wastage and indeed improves **system performance**.

Disadvantages of static memory allocation

Static Memory Allocation

1. In static memory allocation, the system is **unaware** of the memory requirement of the program. So, it has to guess the memory required for the program.
2. Static memory allocation leads to memory **wastage**. As it estimates the size of memory required by the program. So, if the estimated size is larger, it will lead to memory **wastage** else if the estimated size is smaller, then the program will execute **inappropriately**.

Disadvantages of Dynamic Memory allocation

1. Dynamic memory allocation method has an **overhead** of assigning the memory to a process during the time of its execution.
2. Sometimes the memory allocation actions are repeated several times during the execution of the program which leads to more **overheads**.

3.The overheads of memory allocation at the time of its execution **slowdowns** the execution to some extent.

Memory Management Techniques:

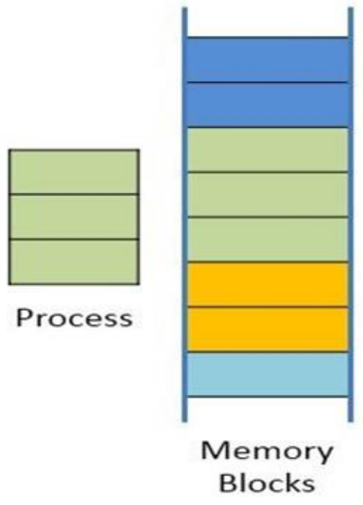
There are two Memory Management Techniques--

Contiguous, and **Non-Contiguous**.

In Contiguous Technique, executing process must be loaded entirely in main-memory.

In **contiguous memory allocation**, all the available memory space remains together in one place.

In the **contiguous memory allocation** when any user process request for the memory a single section of the contiguous memory block is given to that process according to its need. Which can be achieve by dividing memory into the fixed-sized partition.



Contiguous Memory Allocation

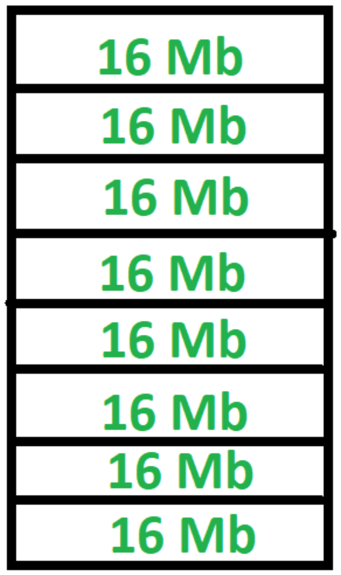
Fixed sized partition

In the fixed sized partition the system divides memory into fixed size partition (may or may not be of the same size).

In this partitioning, number of partitions (non-overlapping) in RAM are fixed but size of each partition may or may not be same.

The memory is assigned to the processes in contiguous way.

The entire partition is allowed to a process and if there is some wastage inside the partition is allocated to a process and if there is some wastage inside the partition then it is called internal fragmentation.



Example :

first process is only consuming 1MB out of 4MB in the main memory.
Hence, Internal Fragmentation in first block is (4-1) = 3MB.

Sum of Internal Fragmentation in every block

= (4-1)+(8-7)+(8-7)+(16-14)= 3+1+1+2 = 7MB.

Suppose process P5 of size 7MB comes. But this process cannot be accommodated inspite of available free space because of contiguous allocation (as spanning is not allowed). Hence, 7MB becomes part of External Fragmentation.

There are some advantages and disadvantages of fixed partitioning.

Advantages of Fixed Partitioning –

1. **Easy to implement**
2. **Little OS overhead**

Disadvantages of Fixed Partitioning –

1. **Internal Fragmentation:**
   Main memory use is inefficient. Any program, no matter how small, occupies an entire partition. This can cause internal fragmentation.

2. **External Fragmentation:**
   The total unused space (as stated above) of various partitions cannot be used to load the processes even though there is space available but not in the contiguous form (as spanning is not allowed).
3. **Limit process size:**
   Process of size greater than size of partition in Main Memory cannot be accommodated.
4. **Limitation on Degree of Multiprogramming.**

Variable size partition

In this technique, the partition size is not declared initially. It is declared at the time of process loading.

The first partition is reserved for the operating system. The remaining space is divided into parts. The size of each partition will be equal to the size of the process. The partition size varies according to the need of the process so that the internal fragmentation can be avoided.

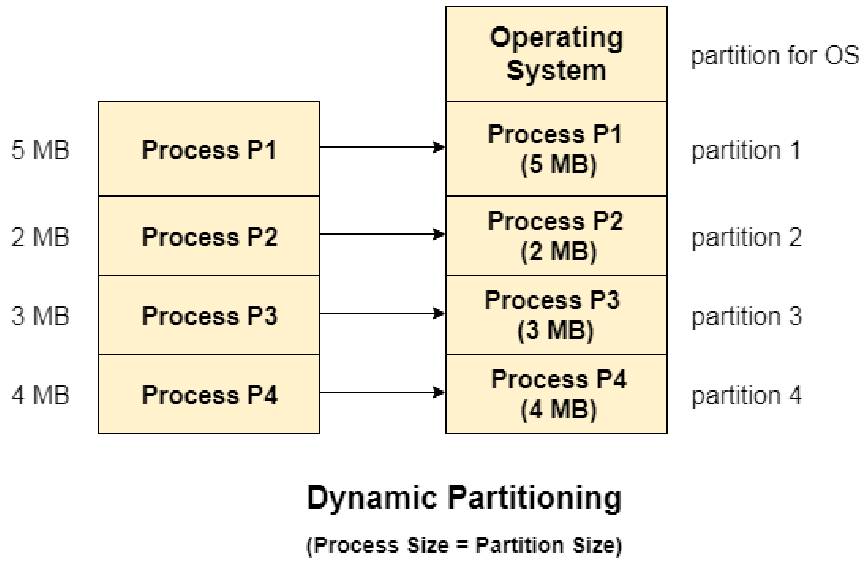


Figure --1

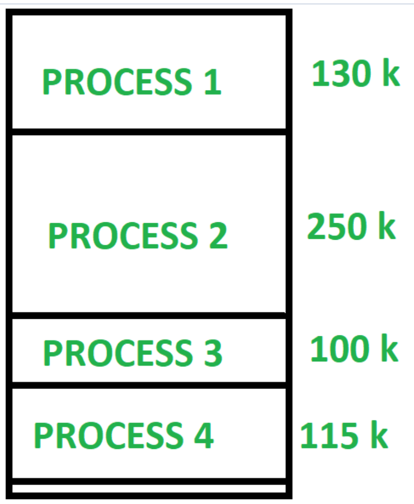


Figure --2

Advantages of Dynamic Partitioning

1. No Internal Fragmentation
2. No Limitation on the size of the process
3. Degree of multiprogramming is dynamic

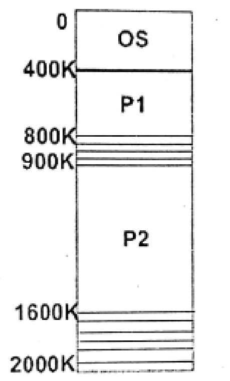
Compaction

In Compaction the idea is to move all processes towards one end of memory and all the free blocks of memory towards the other end.

Compaction refers to combining all the empty spaces together and processes.

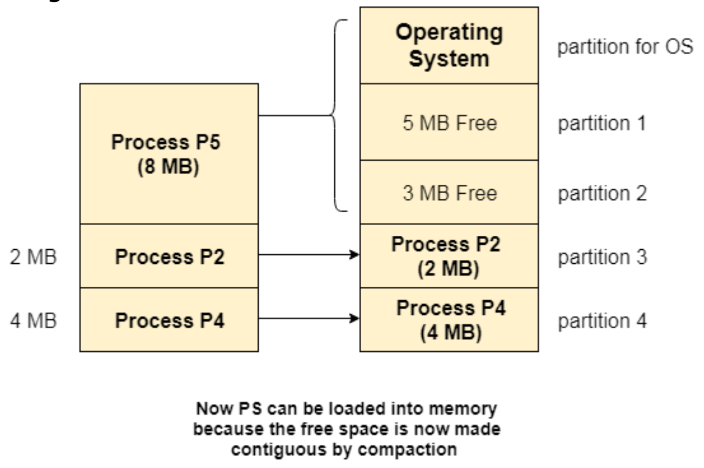
A possible remedy to the problem of external fragmentation is compaction.

For example in Diagram--2
consider the case in which the blocks are distributed between 2 processes as shown in the below image.



Now, using the compaction algorithm, we move process P2 upwards and the free block of 100 K between P1 and P2 downwards thus forming a big block of 500 K. However, compaction is possible only when dynamic relocation is used i.e. when the address binding is done at runtime.

Diagram--2



Compaction

As shown in the image above, the process P5, which could not be loaded into the memory due to the lack of contiguous space, can be loaded now in the memory since the free partitions are made contiguous.

Problem with Compaction

The efficiency of the system is decreased in the case of compaction due to the fact that all the free spaces will be transferred from several places to a single place.

Huge amount of time is invested for this procedure and the CPU will remain idle for all this time. Despite of the fact that the compaction avoids external fragmentation, it makes system inefficient.

Disadvantages of Compaction;

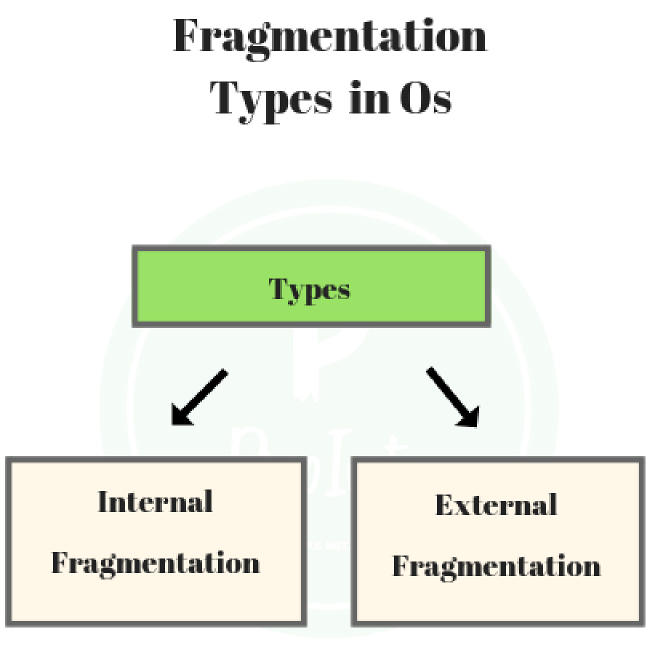
- Decrease the efficiency due to transfer of all the free spaces from several places to a single place.
- Wastage of Time.
- Poor response time because the system must stop everything while Compaction.
- Relocation information lost due to relocating the processes.

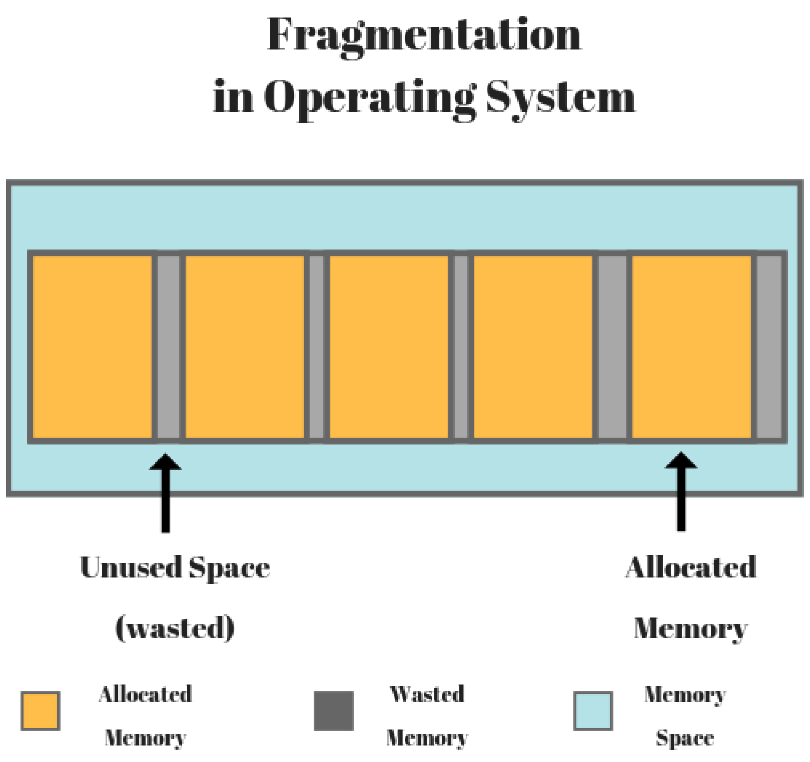
Fragmentation in OS :

Memory space in the system constantly busy in loading and releasing processes and their resources because of which the total memory spaces gets broken into a lot of small pieces, this causes creation small non utilized memory spaces, which are so small so that normal processes can not fit into those small fragments, therefore these memory spaces not getting utilized at all, this is called memory Fragmentation in operating system.

Fragmentation is of the following two types –

1. Internal Fragmentation
2. External Fragmentation



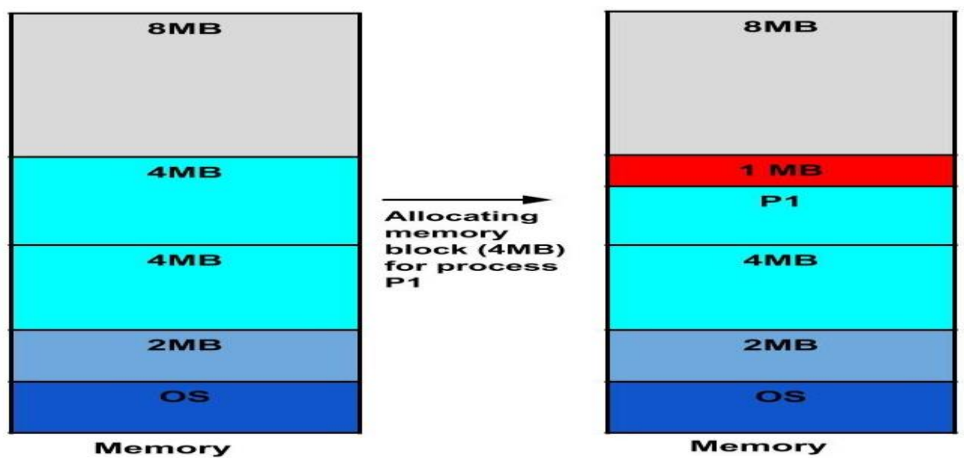


Internal Fragmentation

In this fragmentation, the process is allocated a memory block of size more than the size of that process. Due to this some part of the memory is left unused and this cause internal fragmentation.

Example: Suppose there is fixed partitioning (i.e. the memory blocks are of fixed sizes) is used for memory allocation in RAM. These sizes are 2MB, 4MB, 4MB, 8MB. Some part of this RAM is occupied by the Operating System (OS).

Now, suppose a process P1 of size 3MB comes and it gets memory block of size 4MB. So, the 1MB that is free in this block is wasted and this space can't be utilized for allocating memory to some other process. This is called internal fragmentation.



Q.How to remove internal fragmentation?

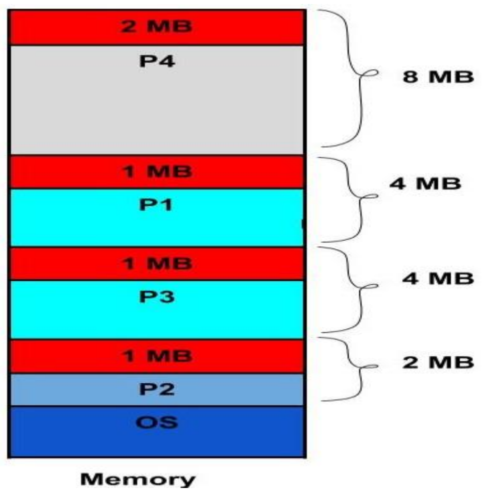
Give one word answer.

External Fragmentation

In this fragmentation, although we have total space available that is needed by a process still we are not able to put that process in the memory because that space is not contiguous. This is called external fragmentation.

Example: Suppose in the above example, if three new processes P2, P3, and P4 come of sizes 2MB, 3MB, and 6MB respectively. Now, these processes get memory blocks of size 2MB, 4MB and 8MB respectively allocated.

So, now if we closely analyse this situation then process P3 (unused 1MB)and P4(unused 2MB) are again causing internal fragmentation. So, a total of 4MB (1MB (due to process P1) + 1MB (due to process P3) + 2MB (due to process P4)) is unused due to internal fragmentation.



Now, suppose a new process of 4 MB comes. Though we have a total space of 4MB still we can't allocate this memory to the process. This is called external fragmentation.

Q.How to remove external fragmentation?

Name two.

Difference between Internal and External fragmentation

S.NO	INTERNAL FRAGMENTATION	EXTERNAL FRAGMENTATION

1.	In internal fragmentation fixed-sized memory, blocks square measure appointed to process.	In external fragmentation, variable-sized memory blocks square measure appointed to method.
2.	Internal fragmentation happens when the method or process is larger than the memory.	External fragmentation happens when the method or process is removed.
3.	The solution of internal fragmentation is best-fit block.	Solution of external fragmentation is compaction, paging and segmentation.
4.	Internal fragmentation occurs when memory is divided into fixed sized partitions.	External fragmentation occurs when memory is divided into variable size partitions based on the size of processes.
5.	The difference between memory allocated and required space or memory is called Internal fragmentation.	The unused spaces formed between non-contiguous memory fragments are too small to serve a new process, is called External fragmentation.

Difference between Fixed Partitioning and Variable Partitioning:

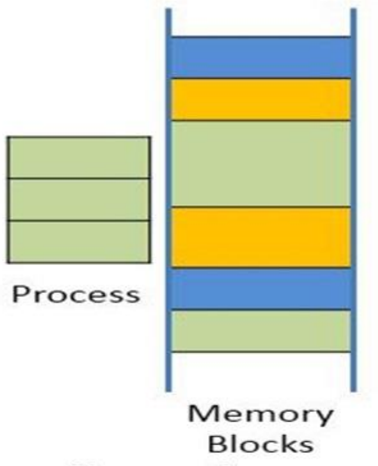
S.NO.	FIXED PARTITIONING	VARIABLE PARTITIONING
1.	In multi-programming with fixed partitioning the main memory is divided into fixed sized partitions.	In multi-programming with variable partitioning the main memory is not divided into fixed sized partitions.
2.	Only one process can be placed in a partition.	In variable partitioning, the process is allocated a chunk of free memory.
3.	It does not utilize the main memory effectively.	It utilizes the main memory effictively.
4.	There is presence of internal fragmentation and external fragmentation.	There is external fragmentation.
5.	Degree of multi-programming is less.	Degree of multi-programming is higher.
6.	It is more easier to implement.	It is less easier to implement.
7.	There is limitation on size of process.	There is no limitation on size of process.

Non-contiguous memory allocation

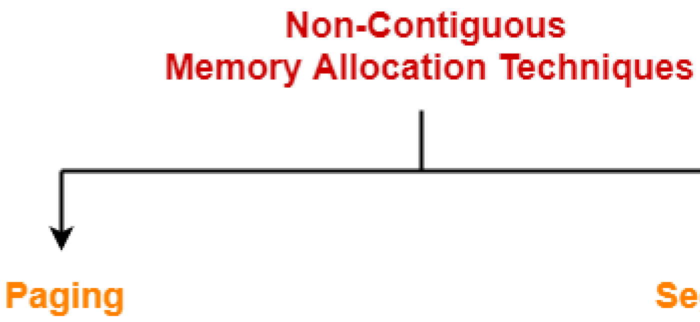
In the non-contiguous memory allocation the available free memory space are scattered here and there and all the free memory space is not at one place. So this is time-consuming.

In the non-contiguous memory allocation, a process will acquire the memory space but it is not at one place it is at the different locations according to the process requirement.

This technique of non-contiguous memory allocation reduces the wastage of memory which leads to internal and external fragmentation. This utilizes all the free memory space which is created by a different process.



Noncontiguous Memory Allocation



Paging

- Paging is a fixed size partitioning scheme.
- In paging, secondary memory and main memory are divided into equal fixed size partitions.
- The partitions of secondary memory are called as pages.
- Every process will have a separate page table.
- The entries in the page table are the number of pages a process.
- The partitions of main memory are called as frames.
- Each process is divided into parts where size of each part is same as page size.
- PCB contains a register value PTBR (page table base register) which leads to the page table.

Note: At each entry either we have an invalid pointer which means the page is not in main memory or we will get the corresponding frame number.

Advantages:

- It is independent of external fragmentation.
- Paging technique is very important in implementing virtual memory.
- It allows storing parts of a single process in a non-contiguous fashion.
- It solves the problem of external fragmentation.

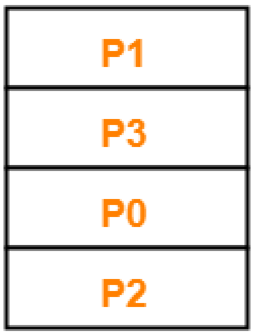
Disadvantages-

- It suffers from internal fragmentation.
- There is an overhead of maintaining a page table for each process.The time taken to fetch the instruction increases since now two memory accesses are required.

Example-

Consider a process is divided into 4 pages P0, P1, P2 and P3.

Depending upon the availability, these pages may be stored in the main memory frames in a non-contiguous fashion as shown-



Main Memory

The Paging Process

When the frame number is combined with instruction of set D than we will get the corresponding physical address.

The MMU uses page tables to translate virtual addresses to physical ones. Each table entry indicates where a page is located.

OS keep track of the states of each frame by using MMT (Memory Map Table).

MMT indicates whether the frame is free or not.

F= Frames

M=Memory Capacity

F=M/P

P=Page Size

A page table stores the definition of each page. When an active process requests data, the MMU (Memory management Unit) retrieves corresponding pages into frames located in physical memory for faster processing. **The process is called paging.**

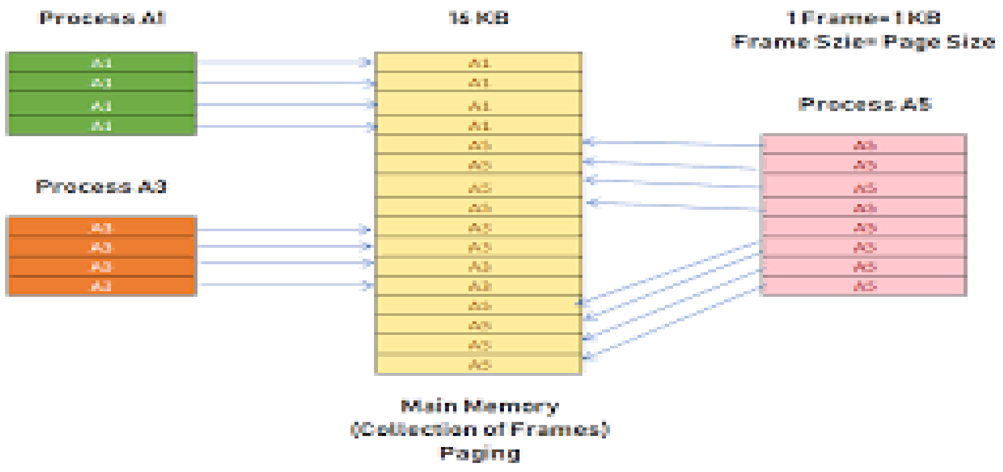
If the size of the process is not exact multiple of 2, last frame is partly used and some space is left in that. **This is known as Page Fragmentation or Page Break.**

Example

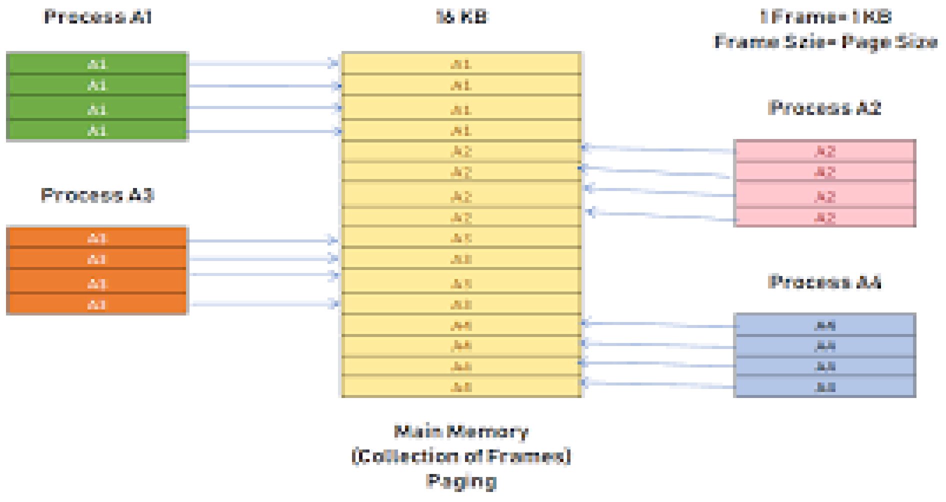
For example, if the main memory size is 16 KB and Frame size is 1 KB. Here, the main memory will be divided into the collection of 16 frames of 1 KB each.

There are 4 separate processes in the system that is A1, A2, A3, and A4 of 4 KB each. Here, all the processes are divided into pages of 1 KB each so that operating system can store one page in one frame.

At the beginning of the process, all the frames remain empty so that all the pages of the processes will get stored in a contiguous way.
In the following example A2 and A4 are moved to the waiting state after some time. Therefore, eight frames become empty, and so other pages can be loaded in that empty blocks. The process A5 of size 8 pages (8 KB) is waiting in the ready queue.



In the following example, there are eight non-contiguous frames which is available in the memory, and paging offers the flexibility of storing the process at the different places. This allows us to load the pages of process A5 instead of A2 and A4.



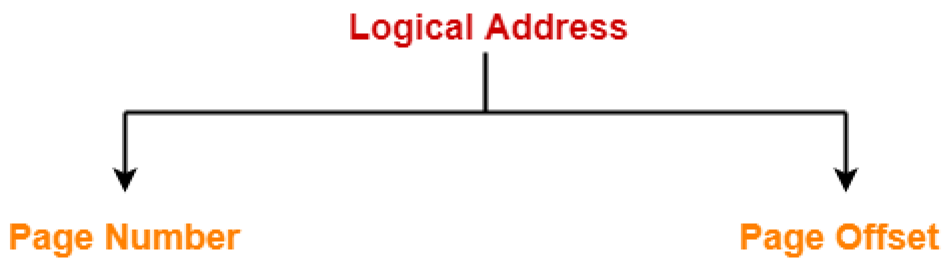
Translating Logical Address into Physical Address-

- CPU always generates a logical address.
- A physical address is needed to access the main memory.

Following steps are followed to translate logical address into physical address-

Step-01:

- CPU generates a logical address consisting of two parts-
- Page Number
- Page Offset



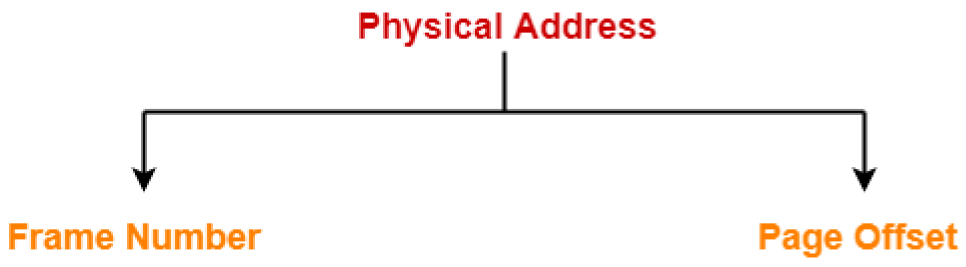
- Page Number specifies the specific page of the process from which CPU wants to read the data.
- Page Offset specifies the specific word on the page that CPU wants to read.

Step-02:

- For the page number generated by the CPU,
- **Page Table** provides the corresponding frame number (base address of the frame) where that page is stored in the main memory.
-

Step-03:

- The frame number combined with the page offset forms the required physical address.

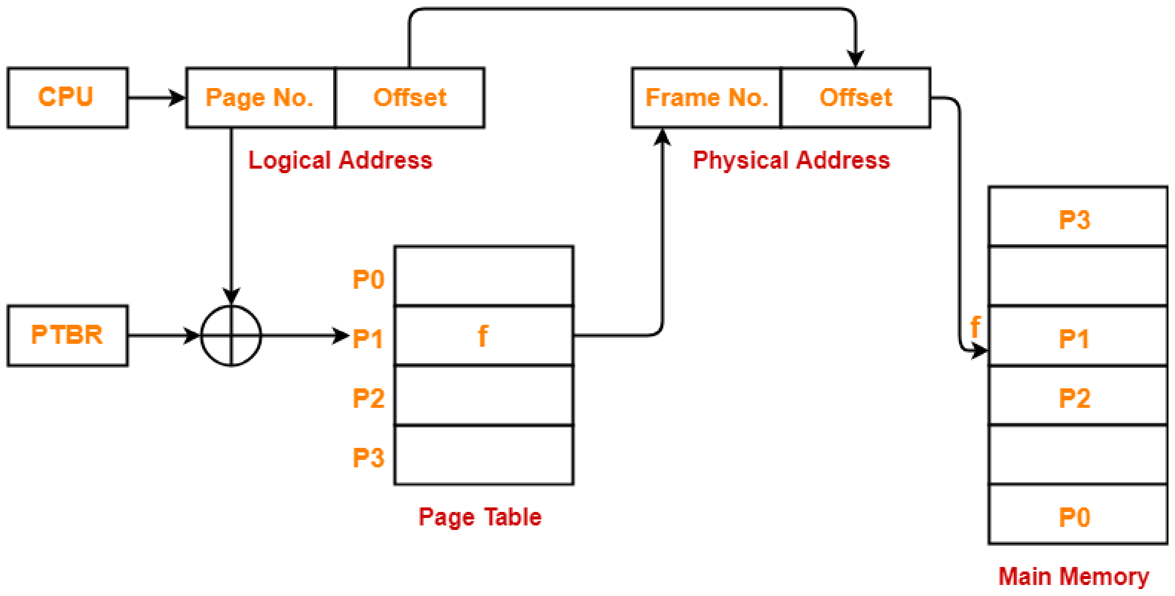


- Frame number specifies the specific frame where the required page is stored.

- Page Offset specifies the specific word that has to be read from that page.

Diagram-

- The following diagram illustrates the above steps of translating logical address into physical address-



Translating Logical Address into Physical Address

Segmentation :

What is segmentation?

Segmentation is a memory management technique in which we divide the process into smaller segments. The process is segmented module by module. In main memory, we only store the segments of the process. Process segment table is used to keep the record of segments, its size, and its memory address.

When a process executes, segmentation assigns related data into segments for faster processing. The segmentation function maintains a segment table that includes physical addresses of the segment, size, and other data.

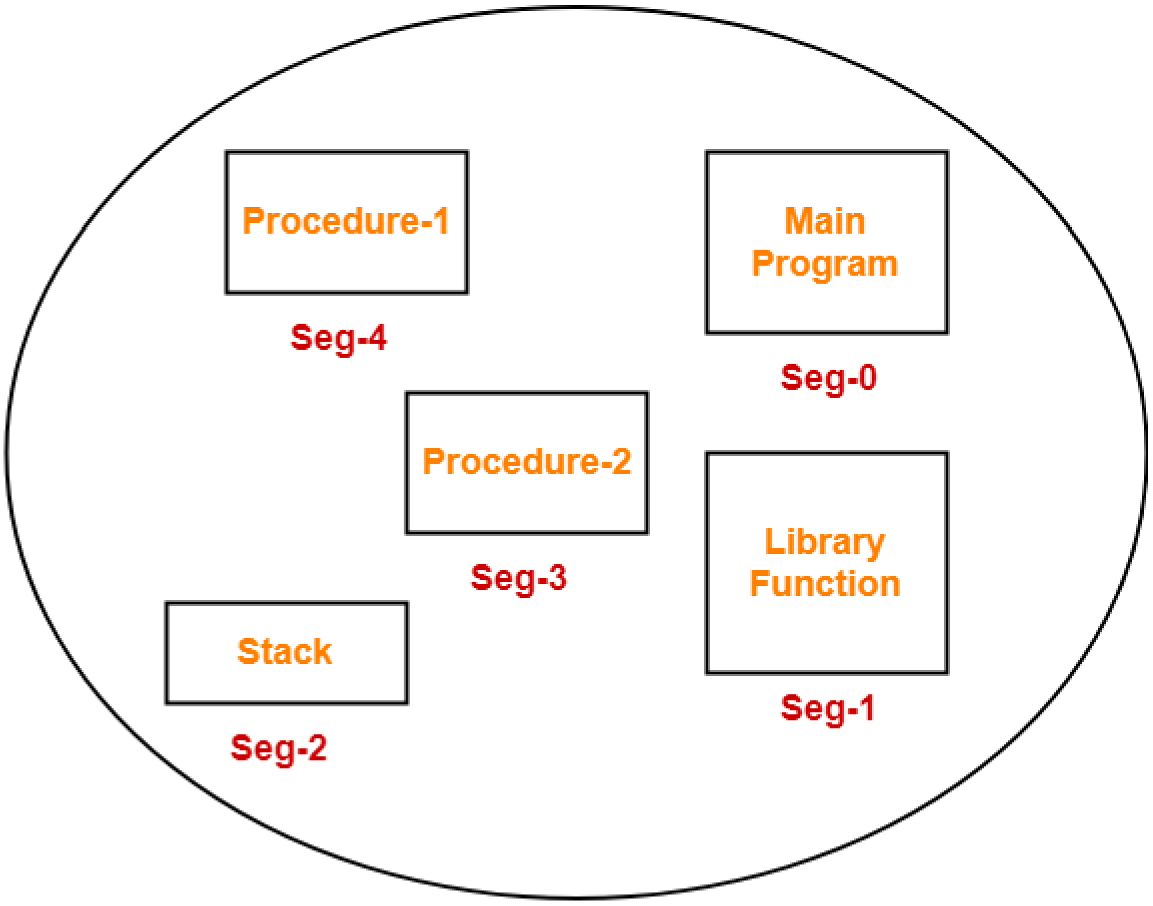
Segmentation speeds up a computer's information retrieval by assigning related data into a "segment table" between the CPU and the physical memory.

Types of Segmentation

1. **Virtual memory segmentation**
   Each processor job is divided into several segments, It is not essential all of which are resident at any one point in time.
2. **Simple segmentation**
   Each process is divided into many segments, and all segments are loaded into the memory at run time, but not necessarily contiguously.

Example-

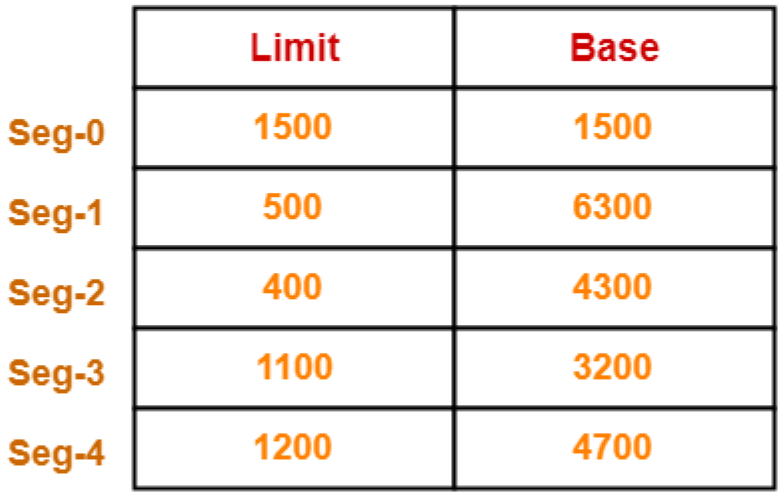
Consider a program is divided into 5 segments as-



Segment Table-

- Segment table is a table that stores the information about each segment of the process.
- It has two columns.
- First column stores the size or length of the segment.
- Second column stores the base address or starting address of the segment in the main memory.
- Segment table is stored as a separate segment in the main memory.
- Segment table base register (STBR) stores the base address of the segment table.

For the above illustration, consider the segment table is-

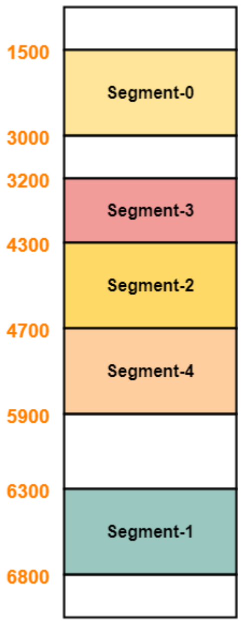


	Limit	Base
Seg-0	1500	1500
Seg-1	500	6300
Seg-2	400	4300
Seg-3	1100	3200
Seg-4	1200	4700

Segment Table

- Limit indicates the length or size of the segment.
- Base indicates the base address or starting address of the segment in the main memory.

- In accordance to the above segment table, the segments are stored in the main memory as-



Main Memory

The Segmentation Process

Segmentation method works almost similarly to paging, only difference between the two is that segments are of variable-length whereas, in the paging method, pages are always of fixed size.

A program segment includes the program's main function, data structures, utility functions, etc. The OS maintains a segment map table for all the processes. It also includes a list of free memory blocks along with its size, segment numbers, and it's memory locations in the main memory or virtual memory.

Following required for mapping:

Segment Table

The correlation and mapping for the logical address which is 2 dimensional addresses and the physical address is done with the help of segment table.

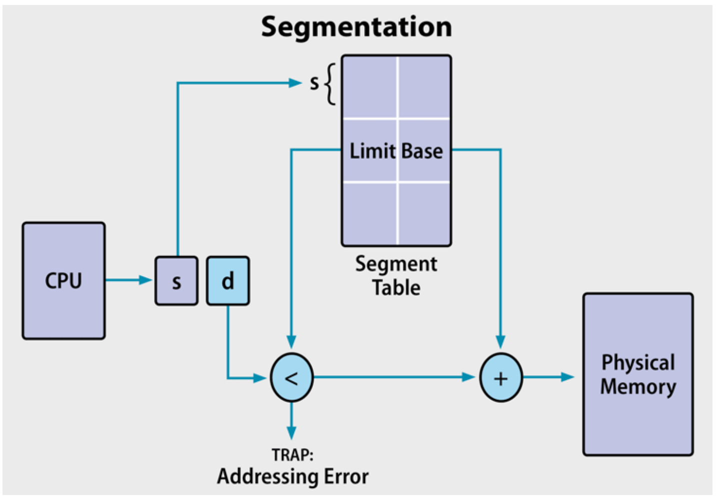
Base Address

The starting address of the memory where the data is stored in its logical format in the system. For example Base address is 8000, then the data will be available in addresses from 8000 onwards.

Limit

Since the size of each segment is different, thus each segment has its own limit defined and it tells the system about the size or length of any particular segment.

1. **Segment Number (s) –** It is basically the total number of bits that are required to store any particular address of the segment in consideration
2. **Segment Offset (d) –** Number of bits required for the size of the segment.



Advantage of Segmentation

- Simple to relocate segments than the entire address space.
- The absence of internal fragmentation as external fragmentation has to be done.
- The segment table is of lesser size compared with the page table in paging.
- The average size of the segment is larger to the actual size of the page
- Offer protection within the segment
- Segment tables use lesser memory than paging
- Due to small segment table, memory reference is simple, which lends itself to sharing data among processes.

Disadvantages of Segmentation

- In segmentation method, processes are loaded/ removed from the main memory. Therefore, the free memory space is separated into small pieces which may create a problem of external fragmentation
- Costly memory management algorithm
- Un-equal size of segments is not good in the case of swapping.
- It demands programmer intervention.

- This is costly memory management algorithm.

Difference between Paging and Segmentation:

Paging

- Page size is fixed
- Can lead to internal fragmentation
- Size is decided by the Hardware
- Address mapping is simple and page table is sufficient

Segmentation

- Segment size is variable
- Can lead to external fragmentation
- Size is decided by the user
- Address mapping is complex even with the help of segment table, segment number and offset.

Hardware support for Paging:

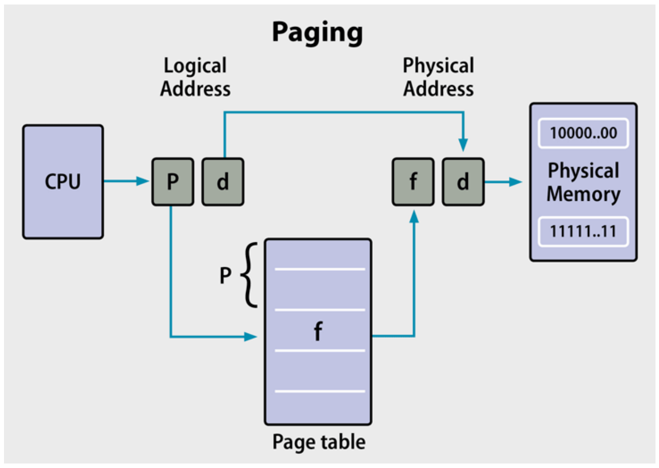
The hardware support for paging is illustrated in figure below.

Every address generated by the CPU is divided into two parts: a page number (p) and a page offset (d).

The page number is used as an index into a page table.

The page table contains the base address of each page in physical memory.

This base address is combined with the page offset to define the physical memory address that is sent to the memory unit.



Hardware support for segmentation

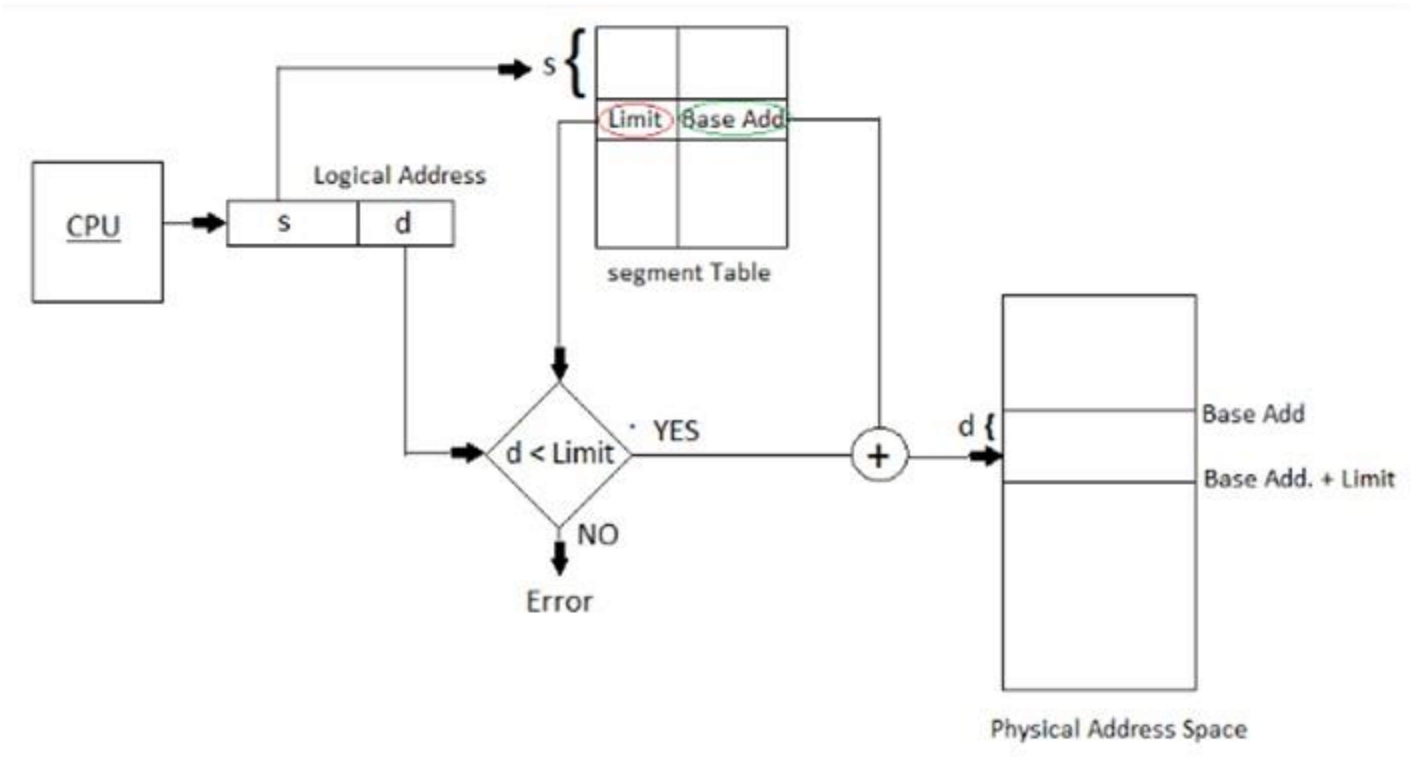
In the program, the user refers to objects by a two-dimensional address, the actual physical memory is still, of course, a one- dimensional sequence of bytes. Thus we have to define an implementation to map two-dimensional user-defined addresses into one-dimensional physical addresses.

This mapping is affected by a segment table. In the segment table, each entry has a segment base and a segment limit.

Segment Base – It contains the starting physical address where the segment kept in memory.

Segment Limit – It specifies the length of the segment.

The use of the segment table illustrated in this figure:



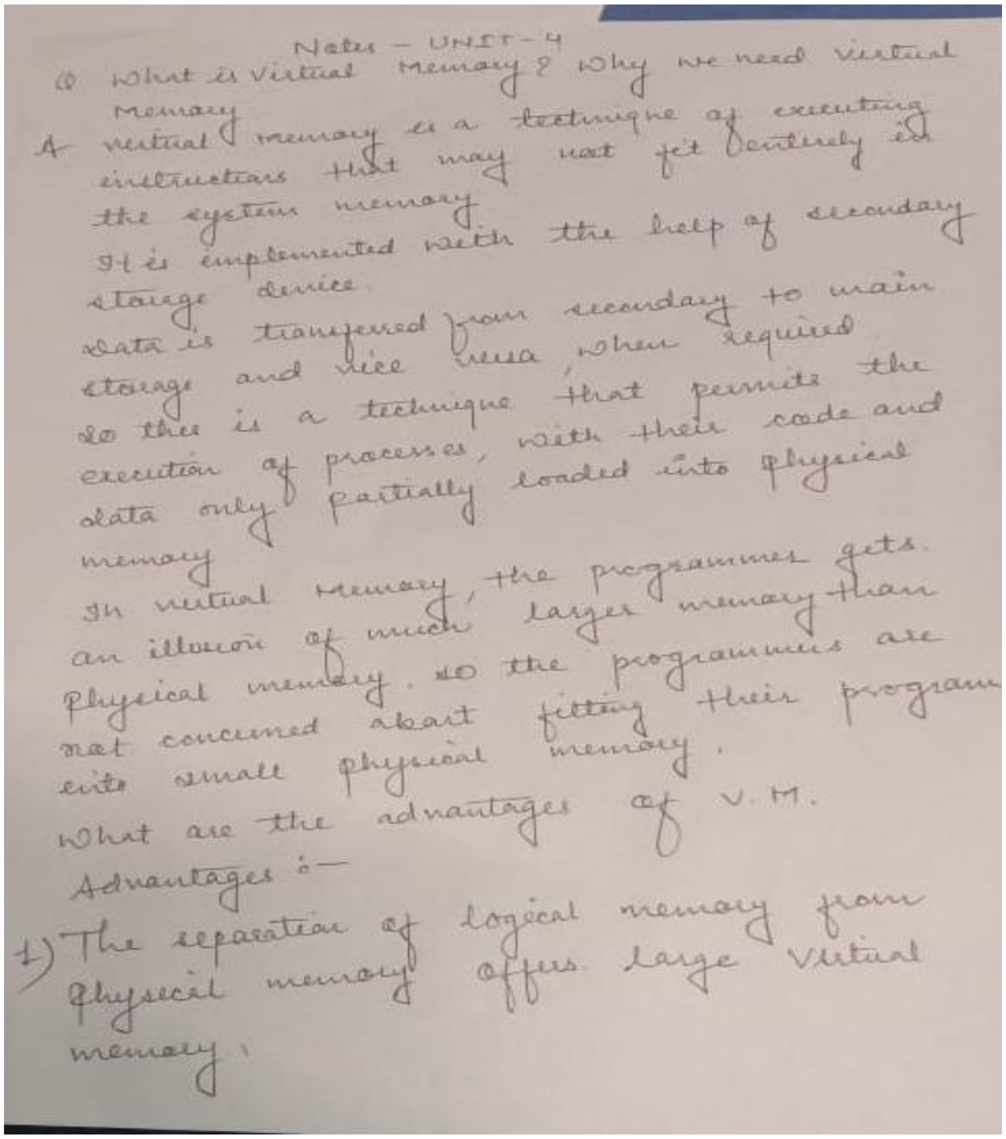
Segmentation Hardware

- The logical address consists of two parts: a segment number (s) and an offset (d) into that segment.
- The segment number used as an index into the segment table.
- The offset d of the logical address must be between 0 and the segment limit.
- If offset is beyond the end of the segment, we trap the Operating System.
- If offset is in the limit, then it is combined with the segment base to produce the address in physical memory, hence the segment table is an array of base limit and register pairs.

Virtual Memory:

Virtual memory is a memory management technique where secondary memory can be used as if it were a part of the main memory. Virtual memory is a very common technique used in the operating systems (OS) of computers.

Virtual memory uses hardware and software to allow a computer to compensate for physical memory shortages, by temporarily transferring data from random access memory (RAM) to disk storage. In essence, virtual memory allows a computer to treat secondary memory as though it were the main memory.



Notes – UNIT – 4

Q What is Virtual Memory? Why we need virtual Memory

A Virtual memory is a technique of executing instructions that may not fit entirely in the system memory

It is implemented with the help of secondary storage device

data is transferred from secondary to main storage and vice versa, when required

so there is a technique that permits the execution of processes, with their code and data only partially loaded into physical memory

In virtual memory, the programmer gets an illusion of much larger memory than physical memory. so the programmers are not concerned about fitting their program into small physical memory.

What are the advantages of V.M.

Advantages :–

1) The separation of logical memory from physical memory offers large virtual memory.

Advantages of V. M.

2) Increase degree of multiprogramming

3) Reduces external fragmentation.

4) Programmer can concentrate only on programming without bothering for memory

<u>Example</u> :- A 1 MB program can run on a 640K. RAM machine by carefully choosing 640K. of program out of 1 MB. to keep memory at each instant and swapping pieces of a program between disk and memory as needed

Virtual Memory Management :-

The implementation of virtual memory is done by paging. which is memory management scheme.

For active process (entry) it requires page map table (PMT)

<u>Note</u> :

As virtual memory is greater than the main memory. therefore page map Table (PMT) size is also larger than PMT of paging.

When manually resetting virtual memory, the minimum and maximum amount of hard drive space to be used for virtual memory must be specified.

Allocating too little HDD space for virtual memory can result in a computer running out of RAM. If a system continually needs more virtual memory space, it may be wise to consider adding RAM.

Common operating systems may generally recommend users not increasing virtual memory beyond 1.5 times the amount of RAM.

Managing virtual memory may be a different experience on different types of operating systems, however.

Virtual memory Working:

Modern microprocessors intended for general-purpose use, a memory management unit, or MMU, is built into the hardware. The MMU's job is to translate virtual addresses into physical addresses. A basic example is given below –

Example:

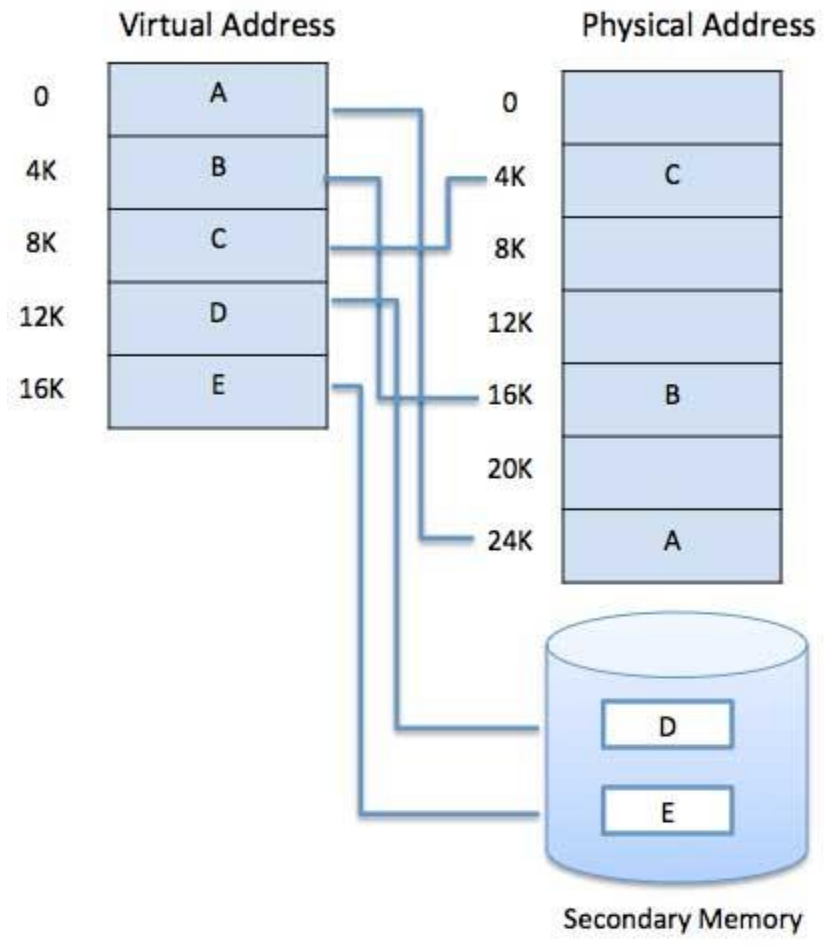
Assume that an OS requires 300 MB of memory to store all the running programs. However, there's currently only 50 MB of available physical memory stored on the RAM.

The OS will then set up 250 MB of virtual memory and use a program called the Virtual Memory Manager(VMM) to manage that 250 MB.

So, in this case, the VMM will create a file on the hard disk that is 250 MB in size to store extra memory that is required.

The OS will now proceed to address memory as it considers 300 MB of real memory stored in the RAM, even if only 50 MB space is available.

It is the job of the VMM to manage 300 MB memory even if just 50 MB of real memory space is available.



Virtual memory is commonly implemented by demand paging. It can also be implemented in a segmentation system. Demand segmentation can also be used to provide virtual memory.

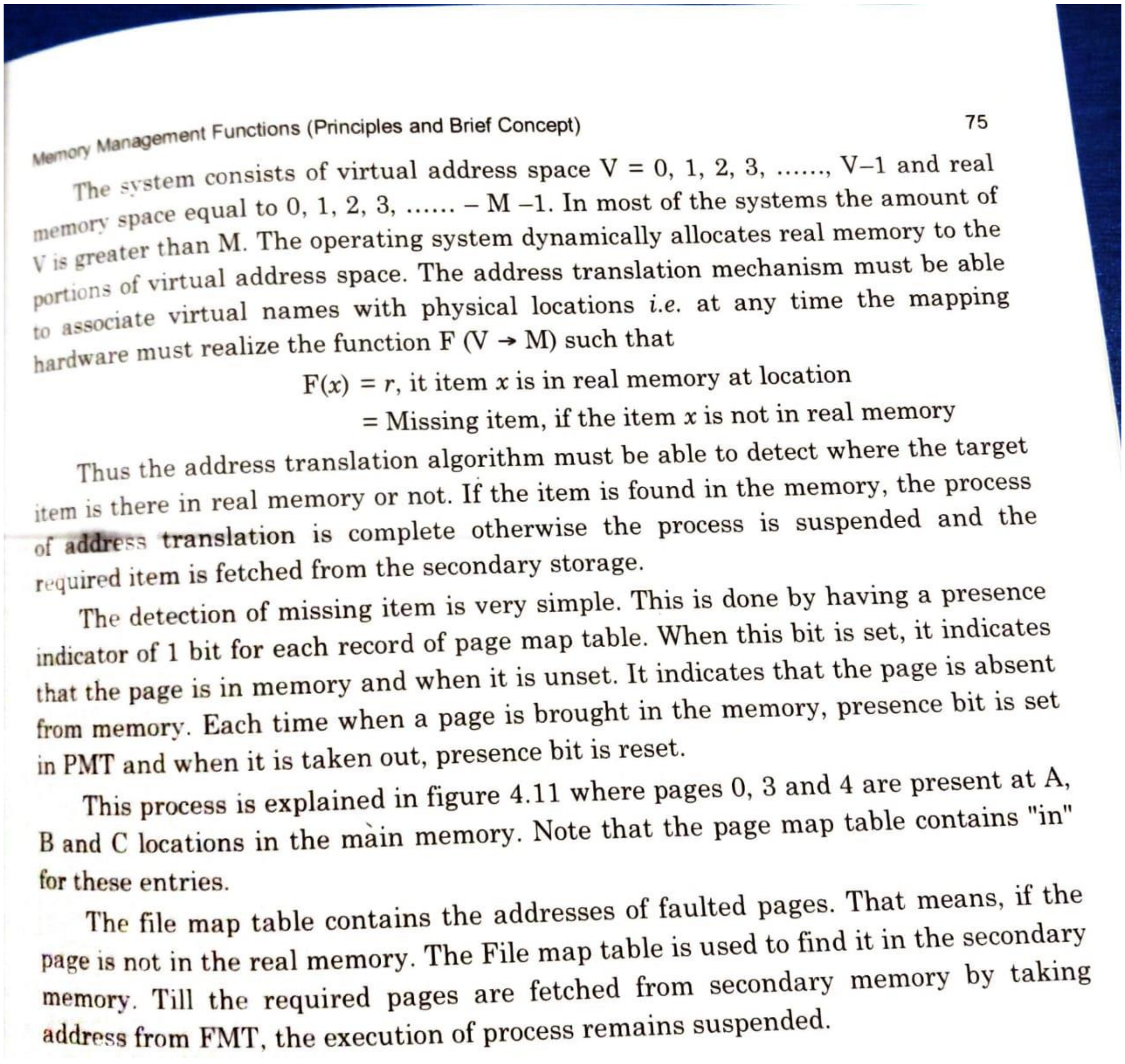
Virtual memory uses both computer hardware and software to work. When an application is in use, data from that program is stored in a physical address using RAM.

More specifically, virtual memory will map that address to RAM using a memory management unit (MMU).

The OS will make and manage memory mappings by using page tables and other data structures. The MMU, which acts as an address translation hardware, will automatically translate the addresses.

If at any point later the RAM space is needed for something more urgent, the data can be swapped out of RAM and into virtual memory.

The computer's memory manager is in charge of keeping track of the shifts between physical and virtual memory.



The system consists of virtual address space V = 0, 1, 2, 3, ..., V–1 and real memory space equal to 0, 1, 2, 3, ... – M –1. In most of the systems the amount of V is greater than M. The operating system dynamically allocates real memory to the portions of virtual address space. The address translation mechanism must be able to associate virtual names with physical locations *i.e.* at any time the mapping hardware must realize the function F (V → M) such that

$$F(x) = r, \text{ it item } x \text{ is in real memory at location}$$
$$= \text{Missing item, if the item } x \text{ is not in real memory}$$

Thus the address translation algorithm must be able to detect where the target item is there in real memory or not. If the item is found in the memory, the process of address translation is complete otherwise the process is suspended and the required item is fetched from the secondary storage.

The detection of missing item is very simple. This is done by having a presence indicator of 1 bit for each record of page map table. When this bit is set, it indicates that the page is in memory and when it is unset. It indicates that the page is absent from memory. Each time when a page is brought in the memory, presence bit is set in PMT and when it is taken out, presence bit is reset.

This process is explained in figure 4.11 where pages 0, 3 and 4 are present at A, B and C locations in the main memory. Note that the page map table contains "in" for these entries.

The file map table contains the addresses of faulted pages. That means, if the page is not in the real memory. The File map table is used to find it in the secondary memory. Till the required pages are fetched from secondary memory by taking address from FMT, the execution of process remains suspended.

Disadvantages of Virtual Memory

- The system becomes slower since swapping takes time.
- It takes more time in switching between applications.
- The user will have the lesser hard disk space for its use.