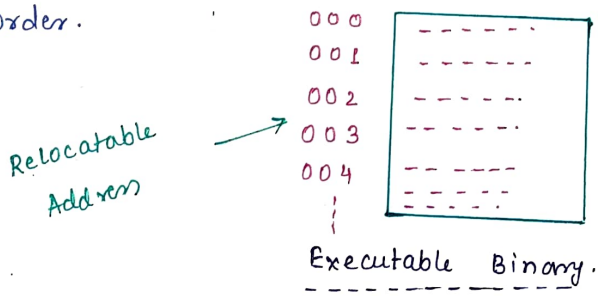
Date: 26/09/2021 to 28/09/2021. O.S. 37. Address Binding Abhishek Sharma NOTES. Written by from GFG course.

Address Binding: Mapping of Relocatable Addresses to Physical Addresses.

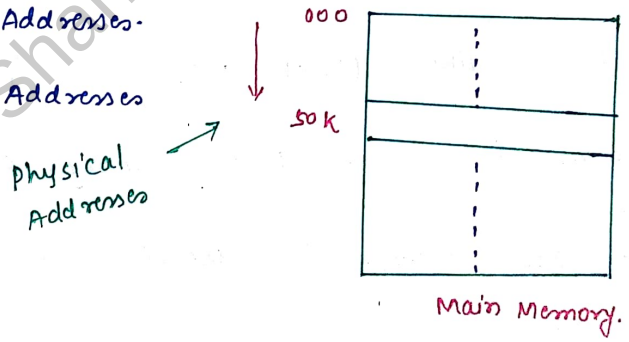
Q) What is Relocatable address and physical Address ?

Relocatable Address: When we produced a Binary of a program. It contains Relocatable Addresses. These are in sequencial order.



TO run this Binary File we have to brought it in Main Memory.

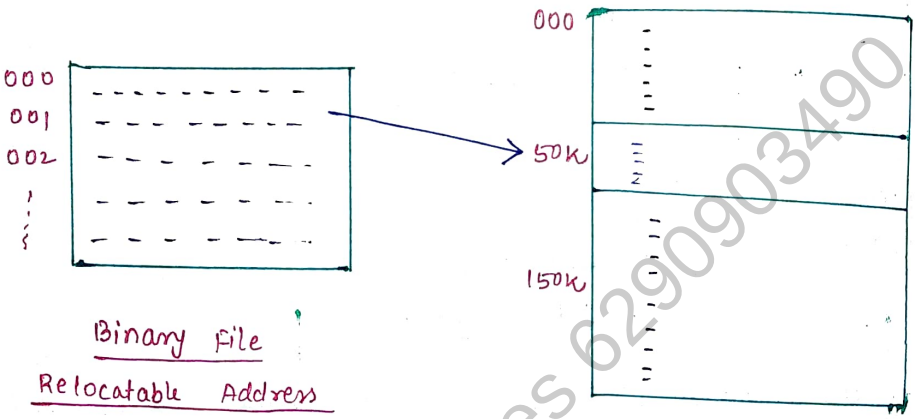
Physical Addresses: The Addresses in the main Memory are called Physical Addresses. It is the actual Addresses in Ram.



* Address Binding can Happen at different stages:

1) **Compile Time:** Compile Time Address Binding is that means your compiler knows where this program is going to be loaded in physical memory and it used to have in very very old days. Ex. .com file and MSDOS. They used to use compile Time Address Binding.

ii) Load Time Binding: your loader has the executable file which has all the relative address. It puts it in the main memory with some base location and all the relative addresses. They are recomputed with reference to base Address.



Binary File
Relocatable Address

Physical Address.
Main Memory.

If our Binary File Program went into the main memory at 50K Then counting starts from There onwards like (50K+1, 50K+2 ---) and so on. Base Address is in our Main Memory. So all the addresses are readjusted according to Base Addresses.

* Problem with Load Time Binding.

Once a program is loaded in main memory, it can't be moved to other location in the main memory.

Q) Why we need to move our Program's location in the main memory ??

Ans: your program might be doing I/O. and when it does I/o. it waits for the I/o. So we can do that move this program to another location and Bring other program for work

iii) Run Time Binding: Run Time Binding Happens in all of the modern Operating systems. In Run Time Binding your process can be moved to a different section of memory at runtime. The Addresses Generated by CPU is not physical Addresses. CPU Generates Logical Addresses and These address are converted to physical Addresses at RUN Time.

TO Implement Run Time Binding Hardwares support is required. In compile Time Binding, Load Time Binding we could do it through software.

* There are some Memory Management unit which converts these Logical Addresses into physical Addresses.

27/09/2021 34. RunTime Binding.

Run Time Binding: As we discussed above, The addresses generated by CPU is Logical Addresses in Run Time Binding. And These Logical Addresses are converted to physical Address during RunTime.

Purpose Of Runtime Binding: your process can move to different location while it is being Run.

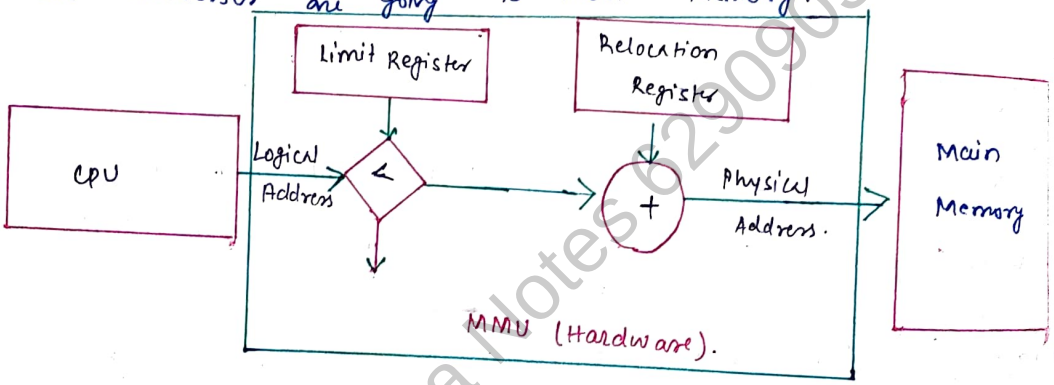
Working.

Runtime Binding Happens in our Hardware. In Hardware we have MMU (Memory Managment unit) which have two registers Limit Register and Relocation Register.

When a process is loaded into Memory. These Registers are Loaded by the Operating System. (Limit & Relocation Register).

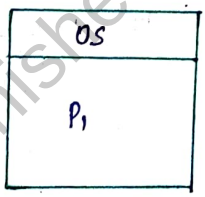
CPU generates the logical Addresses and after the generation of Locigical Addresses These 2 Register converts these Addresses into physical Addresses. Hardware helps you in doing Security Also. It provides you security. Yours process can not access beyond its Limit.

After converting the logical Addresses to physical Addresses. These Addresses are going to Main Memory.



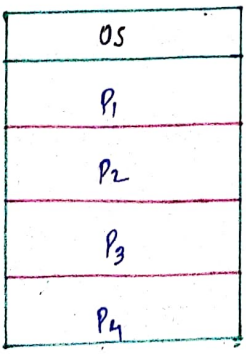
35. Evolution of Memory Management.

* Single Tasking System:



It is the Initial system. At a time only one process can be run (Ex. Process P_1) After completion of one process another Process can be run.

* Multitasking System:



You can run Multiple Processes at a Time. CPU can be utilised in a Proper way.

You might have multiple processes available in memory. So that CPU can be utilized better.

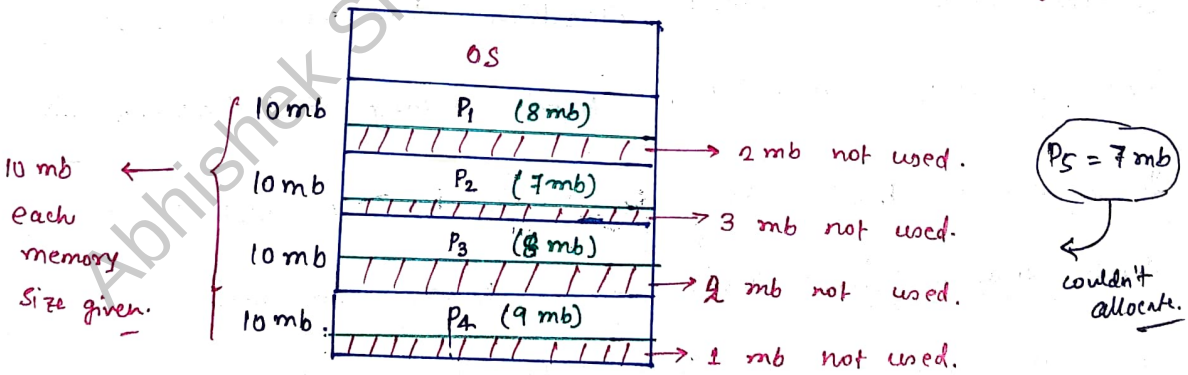
Memory Allocation in Multitasking system

- Static
  - Equal size
  - Unequal size.
- Dynamic
  - i) Contiguous
  - ii) Non-contiguous.
    - a) paging
    - b) segmentation
    - c) paging with segmentation.

Now we will see examples of each and every Multitasking system.

Equal size Partition :

Static Memory Allocation in Multitasking Syste



We have 10 mb memory for each process and $P_1 = 8mb$, $P_2 = 7mb$, $P_3 = 6mb$, $P_4 = 9$ mb consumes the memory space

Now a process $P_5 = 7mb$ wants to access the memory but it can't allocate. Even though we have total $(2+3+2+1) = 8mb$ memory which is free but we can't assign $P_5 = 7mb$.. (but in different areas)

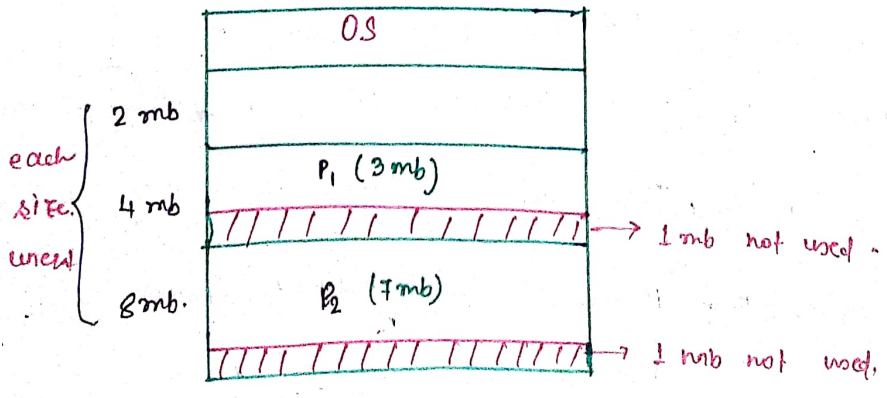
This is called External Fermentation.

External Fermentation: There is a process waiting for some memory for allocation. But there is memory Available but in different parts. So that it is not allocated to the process. is called External fermentation.

Internal Fermentation: When more memory is allocated than needed. Like in our Equal size memory example. we allocate 10 mb memory for each Process But the process is consumeing less memory than needed. This is Internal Fermentation.

Unequal Size Partition

The idea is to Better utilise the memory.
In this type of memory allocation we have unequal memory slots rathar than fixed memory slot (equal partition) If a process need 3 mb and the memory slot is of 4 mb than it goes there. If a process has need 7 mb and the slot is of greater than 7 mb then it goes there



The unequal size partition is better than equal size partition. Bcz. Here Memory is better utilized. Still it has also Internal Fermentation and External Fermentation.

* Both of the Static Memory Allocation in Multitasking System has Internal Fermentation and External Fermentation. (Equal size and unequal size).

Dynamic Memory Allocation in Multitasking System.

In dynamic Memory Allocation we need memory Empty initially. When Processes came we allocate them into memory accordig to their sizes. Durig Runtime you allocate Memory according to the Process size.

Contiguous Dynamic Memory Allocation.

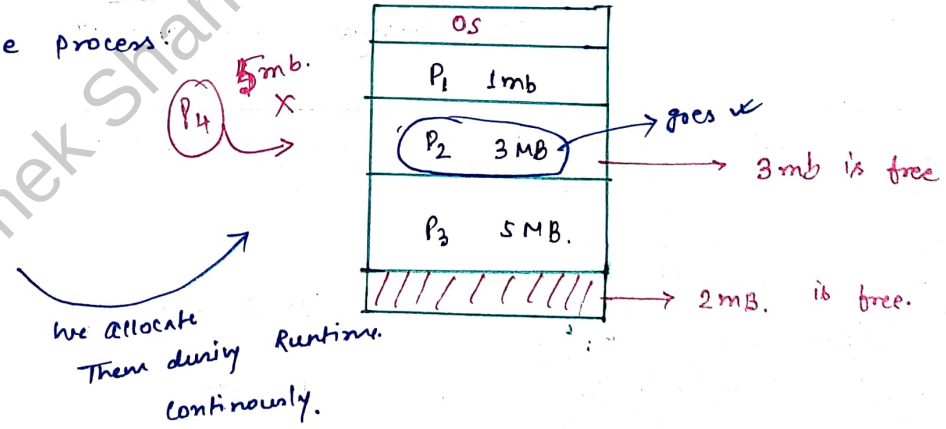
We allocate Memory dynamically contiguous (one by one).

We have Three process:

$P_1 = 1$ mb

$P_2 = 3$ MB

$P_3 = 5$ Mb.



Let's Assume, At a time P_2 goes from the Memory and the memory space is now empty i.e. (3 mB) we have another (2 MB) free Memory. Now we have (3+2) = 5 mb Free on different spaces. and a Process P_4 = 5 mb want to come but it is not allowed. and not allocated.

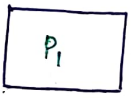
In Contiguous Dynamic Memory Allocation, we don't have Internal Fermentation. But we have External Fermentation.

TO Avoid External fermentation we have non-contigeous Dynamic Memory Allocation. In which we use paging and segmentation to avoid External fermentation.

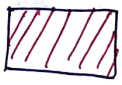
we devide the processes into pages and these pages are loaded seperately at different location. Or we devide the Processes into Segment and these segments are loaded at different locations.

These pages and segments do not have to be at contigeous Location. We will study in the next lect. about paging and segmentation

36. Dynamic partitioning:



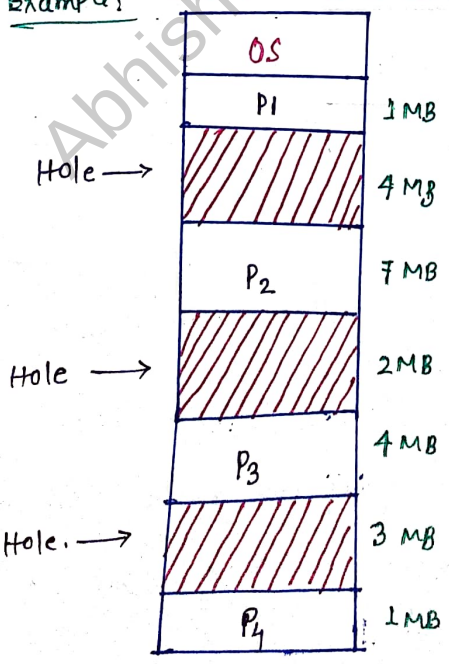
⇒ P_1 Process is running and occupying memory.



⇒ Process Left the memory and there is a hole.

Non-Contigeous Dynamic Memory Allocation..

Example:



If A process P_5 = 5mb. came and we have to allocate.

How will be we allocate ??

There are no continious memory.

How will we allocate ??

A: There are 2 ways. to allocate Memory.

There are 2 ways to allocate Memory in non-Contiguous Dynamic Memory Allocation.

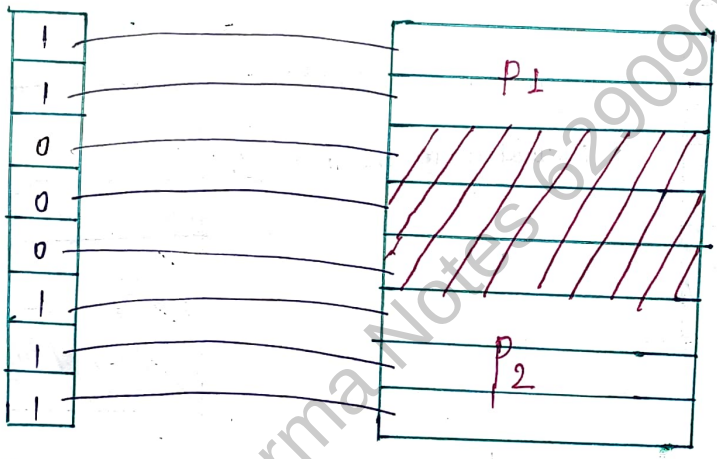
i) Bitmap

ii) Linked List.

Bitmap : We use Bitmap. Where every entry has '1' and '0'.

1 means the particular slot of memory is occupied.

0 mean there is a Hole. (Memory is Free).



Bitmap. OS.

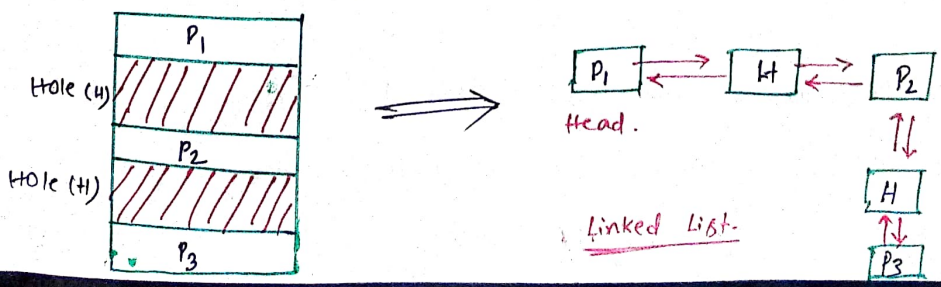
The problem with this Bitmap approach is that it consumes lots of memory. Ex. you have 32 bits size memory and 1 bit of it is occupy by this Bitmap.

So every 32 bit you need 1 bit for Bitmap.

That is why Bitmap is not used in Dynamic Memory Allocation.

Linked List : This is the other way to manage memory in Dynamic partitioning is Linked List.

H = Hole.



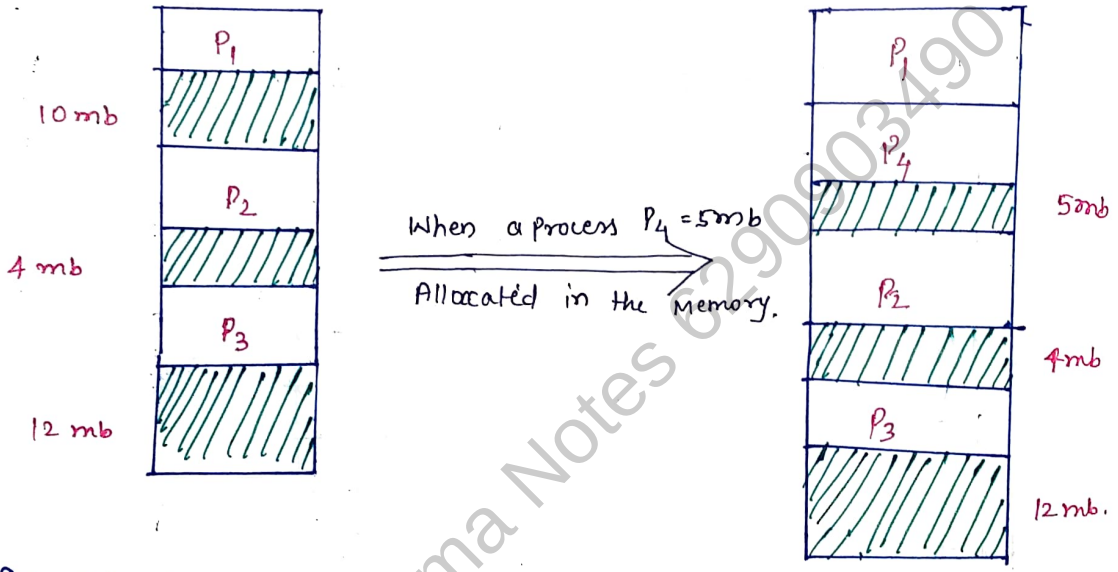
37. First Fit, Best fit, Next Fit & Worst Fit :- Abhishek Sharma Notes.

- Algorithms:

These algorithms (First, Best, Next, Worst Fit) are about How we allocate Memory in the Linked List Representation.
↓
To the process.

i) First Fit Algorithm:-

This is the Best Algo. in all algorithms.



In this Algo. we Traverse from the Linked List from Head. (from Beginning) and as soon as we find a Hole that can accomodate the process we allocate memory to the process.

ii) Best Fit Algorithm:-

Instead of Allocating the 1st node, we traverse the Whole list and we find the Hole that is best fit for the process having size closest or equal to the waiting process.

Example. In the above Example, (in First Fit) if a Process P_4 = 3 MB comes we allocate Memory. After P_2's Hole that is of (4 MB Hole).

The disadvantages of Best fit Algorithms is that we have to traverse through the whole List. to find the Hole.

iii) Next Fit Algorithm:

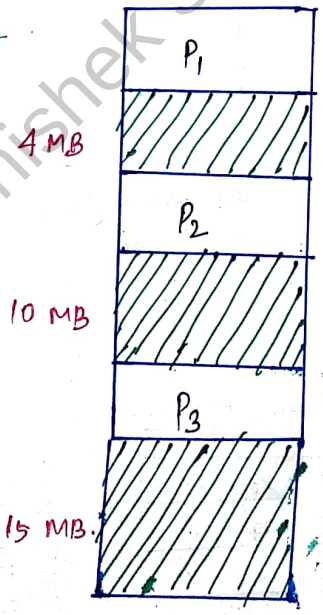
We start searching ^Hole from the point where we stopped last time.

Ex. If we have 2 processes P_4 = 8 mb and P_5 = 3 mb. Then we allocate P_4 in P_1's after Hole and we now start searching for P_5 from P_1's Hole after this point. That's How next Fit Algo works. It starts searching from the point of Last search. It's not begin from the beginning.

iv) Worst Fit Algorithm:

Always allocates the biggest slot to process request in the memory.

Example.



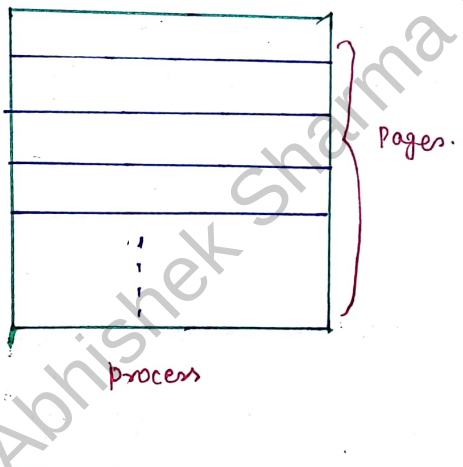
Whatever request it might come, it always assign to this Hole. bcz. it's size. is Biggest.

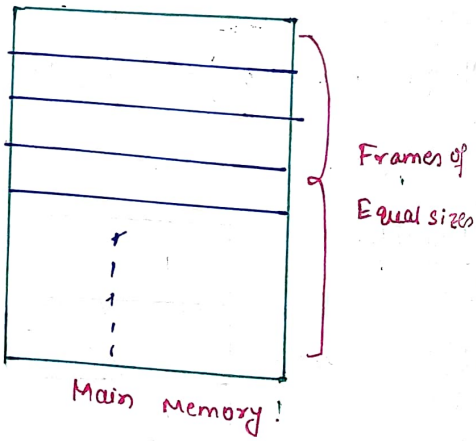
After Analysis, First- Fit Algorithm is the best Algorithm. bcz in Best Fit and Worst Fit Algo. we need to Traverse the whole list.

Contd.

Paging: The most popular idea of Memory Management is paging. This idea create abstraction where programmer do not worry for anything. They see their programme as contiguous memory location and internally your Programme is stored in different memory locations.

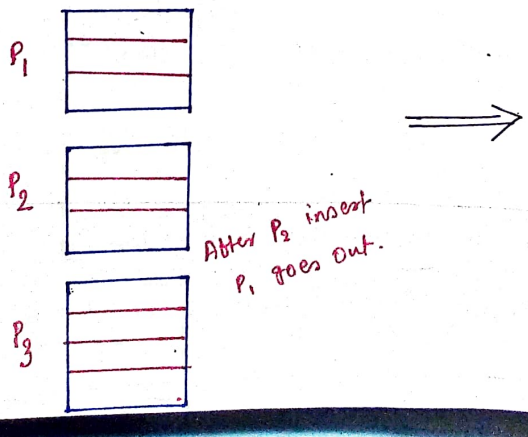
In paging your main memory is divided into slots of equal sizes. And all these slots are called Frames. Similarly your process is divided into equal sizes and all these slots are called slots of pages.





* Page size is always equal to frame size.

Example of Paging:



We can insert P3 in different Location as well in Paging.

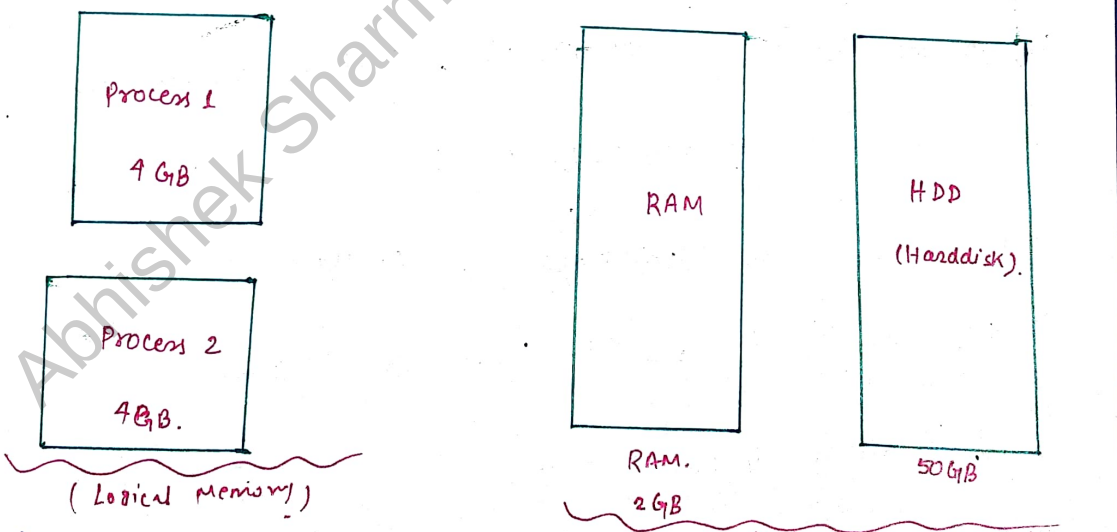
* To convert Logical Addresses to physical Addresses we use Page Table. These page tables are Mapping b/w pages and Frames. Page No. is Generated by the Logical Addresses is passed to the page table and page table tells you the frame no. in the Main Memory. (Physical Memory).

39. Virtual Memory.

Virtual Memory is very closely related to paging.

The Idea of virtual Memory is some pages of the process may be present in the memory and some pages of the process may not be present in the memory. We will talked about valid and invalid Bit.

Virtual Memory and Related Concepts.



A process Assumes that it has 4 GB of Memory. It doesn't know that it has 2 GB of Memory only.

* Whenever a page is not present and later required by the process then **Page fault** happens.

Idea of Virtual Memory:

Whenever there is shortage of physical Memory Happens. Virtual Memory compensate the shortage of physical Memory by Transferring Pages from Harddisk.

When a page is required and it is not in the Memory. It is loaded from the Harddisk.

* Page Fault is a really costly Operation.

OS has to Run some code to bring that page from Harddisk to Ram. There are Many Things involved.

* Harddisk access is Really really slow in compared to normal RAM Access.

* Performance Impact of page Fault:

Avg. Access Time = 0.99 x 10 ns + 0.01 x 5000000 ns

= 50,009.9 ns.

= ~~1500~~ = 50 µs.

* Page Fault increses the Avg. Access Time.

Advantages of Virtual Memory:

1) The Degree of Multiprogramming goes High.

Disadvantyes of Virtual Memory:

1) If you have very Few pages allocated to individual process Then There are going to many page faults.

40. TLB, Demand Paging, Thrashing, Page Replacement Algorithms.

TLB: Translation Lookaside Buffer.

It is an associate cache present in your CPU.

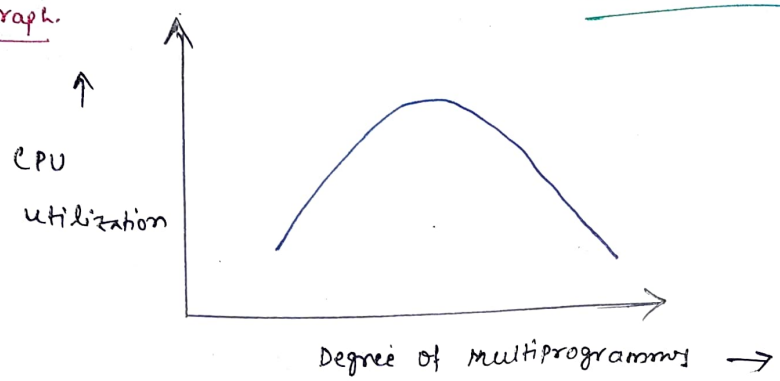
Purpose of TLB: CPU Generates Logical Address. and every Page may contain multiple Logical Addresses. So when a CPU Generates a Logical Address This address has to be convert into physical addresses. we look these physical Addresses in Page table to find the frame nos. in the memory. and This look up is very frequent it might happen very frequently for the same page no.

So The idea of This TLB is to optimise This lookup. we have the cache of this page table in our processor. and This cache is very fast bcz most of the time you need to access the same page no.

Demand Paging: You only keep working set part of the process in memory. and in pure Demand paging Enviornment you need keep the minimal part of Process in the Memory. and you will fetch the part from the memory which part is demanded. This is called Demand paging.

Thrashing: Thrashing is when the page fault and swapping happens very frequently at a higher rate, and the OS has to spend more Time swapping these pages. This state of OS is called Thrashing.

Thrashing Graph.



After a certain Time Graph decreases bcz as you increase Degree of Multiprogramig page fault happens. and This Situation is called Thrashing.

Page Replacement Algorithms.:

Page Replacement: When page fault occurs. you need to bring the page from Harddisk to RAM. and your RAM may not have space for accomodate this page. For TO accomodate this page we need to move one of the existing page from RAM to Harddisk.

Q How did you decide which page to move to bring a new require page ??

A TO decide this particular page (which has to move from RAM to Harddisk) we ~~need~~ use. Page Replacement Algorithm.

* PAGE Replacement can be Local Or Global.

Local Page Replacement: A Process has fixed working Set allocated to RAM and when page fault occurs you need to Replace one of the processes pages to Bring a new page.

This is called Local Startgy for page Replacement Algorithm.

* Local Startgy is the most popular startgy for page Replacement Algo.

Global page Replacement: You can pick pages of other processes and you move them to disk to bring a new page.

Algorithm for page Replacement.

It doesn't matter you have Global or Local page Replacement you need to follow the following Algo's.

i) First in First out (suffers from Belady's Anomaly).

ii) optimal (You remove that page which is access latest in time).

iii) Least Recently used. (This is the Most Popular page Replacement Algorithm).

It is not feasible for OS to know. which page has access latest in future.

Most popularly used.

Idea: If a page is being access now, it is very possible that it is going to access in near future also.

You replace the page which is accessed earliest in time.

41. Segmentation in Memory Management.

Segmentation is an alternate of paging.

In paging, we devide our process into small size equal partition and these equal partition may be there at non-contiguous location. Advantages of paging are our process can be loaded into non-contiguous Location also we get to implement virtual Memory. where we maintain page table and we map these pages.

To frames with main memory.

Segmentation is an alternate to implement both the ideas. Non-contigous Allocation of process in main Memory and implementing Virtual Memory.

idea: idea of segmentation is to brought the related items together into main Memory. and These related items might be of different sizes. Not of the same sizes as pages.

In segmentation, we devide our process accordig to User's view. User's view is related items together.

Advantages of Segmentation:

1) Your process can be loaded into non-contigous memory location.
2) We can Implement virtual Memory.

Q How we can load our process into non-contigous Memory location??

Let's see with the Help of Example:

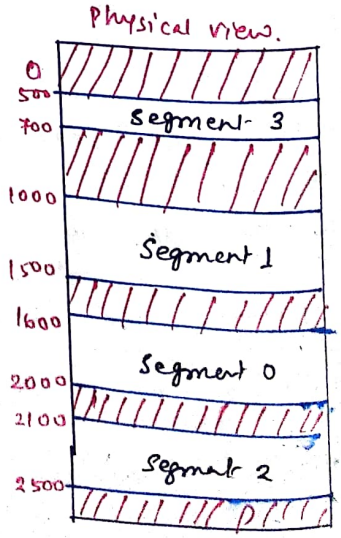
Logical view

Segment 0
segment 1
segment 2
segment 3.

Segment table

	Base	Limit
0	1600	400
1	1000	500
2	2100	400
3	500	200

Physical view.



If The limit is crossed Then it will give us error or segmentation fault.

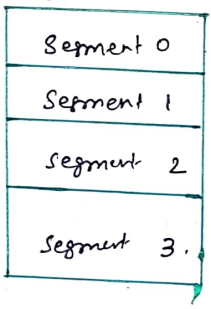
Simple Segmentation: All Segments are present.

Q. How can be we implement virtual Memory with use of Segmentation.

Let's see with the Help of Example:

We can add Absent (0) present (1) Bit in our segment table. Then all the segments don't need to present in your main Memory.

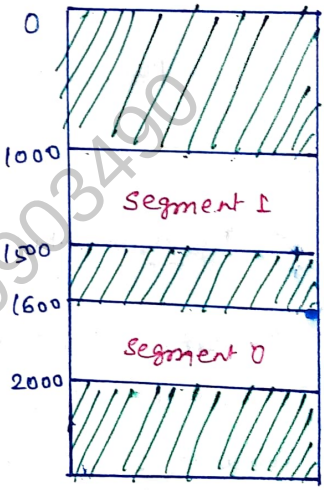
Logical view



Segment Table.

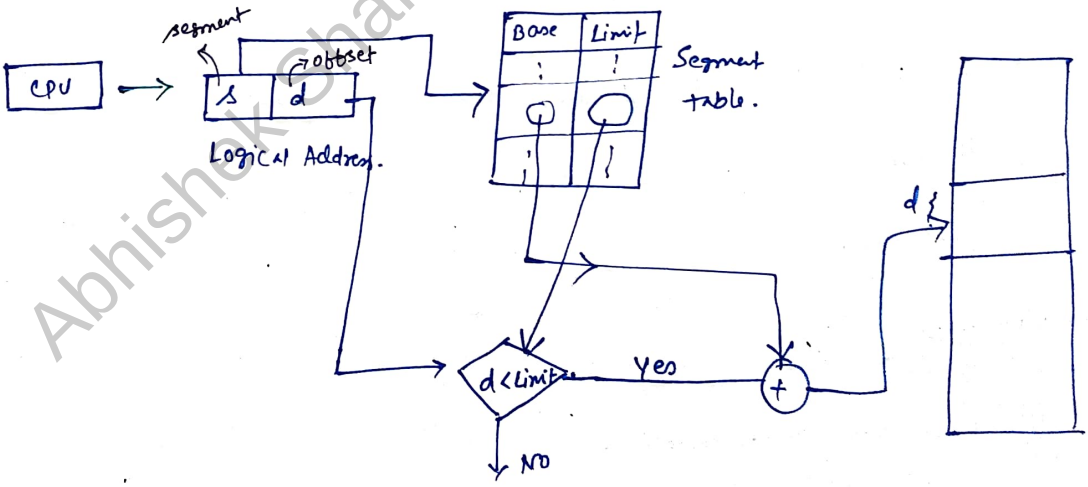
	Base	Limit	Absent/ Present Bit
0	1600	400	1
1	1000	1500	1
2			0
3			0

Physical view.



Virtual Memory Segmentation: All segments need not to be present.

Working of The segmentation



Error or Segmentation Fault.

This is How we access perticular process through Segmentation.

Overview of Segmentation:

Segmentation can be of pure segmentation. All the example we discussed are of pure segmentation. where we divide our process into segments and we are loding the individual segments as a whole.

Pure segmentation → Simple
Pure segmentation → Virtual.

* Segmentation with paging: This is implemented in the Architecture that we generally use.
We will talk about it further lect.

Advantages of Segmentation Over paging:

i) No internal Fragmentation.

ii) user view

iii) Segment table is smaller than Page table.
Therefore easier to implement sharing and Protection.

Disadvantages:

i) External Fragmentation.

42. Segmentation With paging.

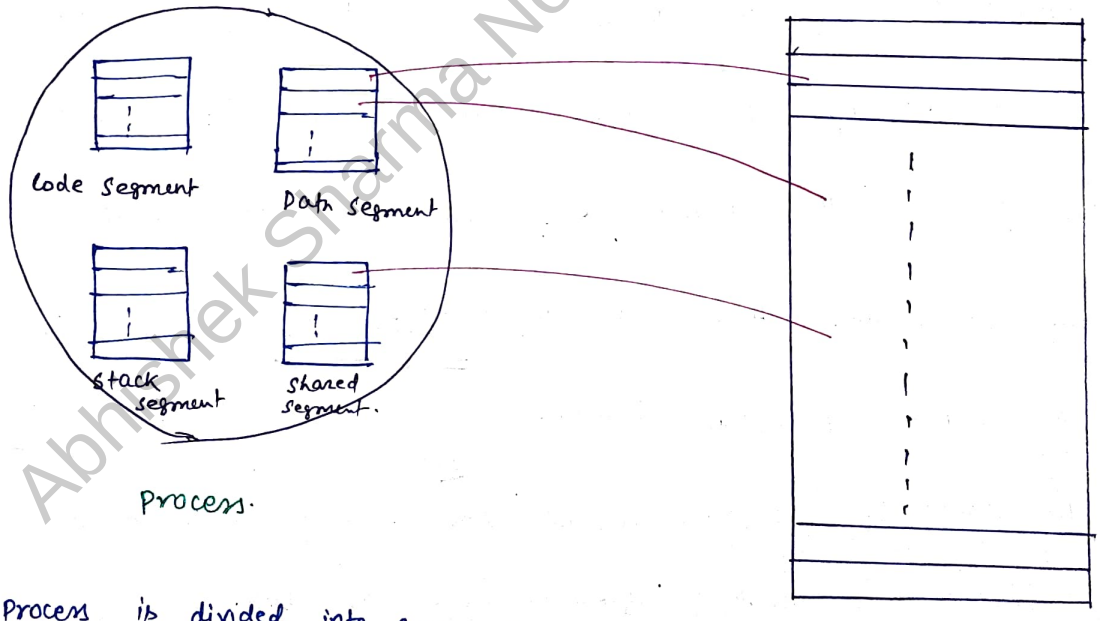
Idea of segmentation with paging :

In paging does better memory utilization bcz in segmentation you have External Fragmentation (Disadvantage).

In segmentation has the advantages of user view and sharing and production is easier in segmentation. bcz you do it at the segment level..

So the idea of segmentation with paging is to take the advantages of both the thing (above upper mentioned)..

Example Diagram of segmentation with paging.



A Process is divided into some segment- and each individual segment has some pages

Q1 How we will map these segment. pages (Logical Address) to our physical Memory Address ??

A

Abhishek Sharma Notes.
ph. 6290903490

Q How we will map now Logical Address to physical Memory Address ??

Ans Now we have Both Segment table and Page table. For a single process we have our own segment table. When we switch context from one process to other Processes there is a register in CPU which is set for context switching. named Segment Base Register. This Register gives us base address of the segment table. That is How we will figure out where is the Segment table of our process. using this Segment table we go to the Page Table and using this page table we go to the Frame No. and once we go to the frame no. we add the offset (d) to go to the actual physical Address.

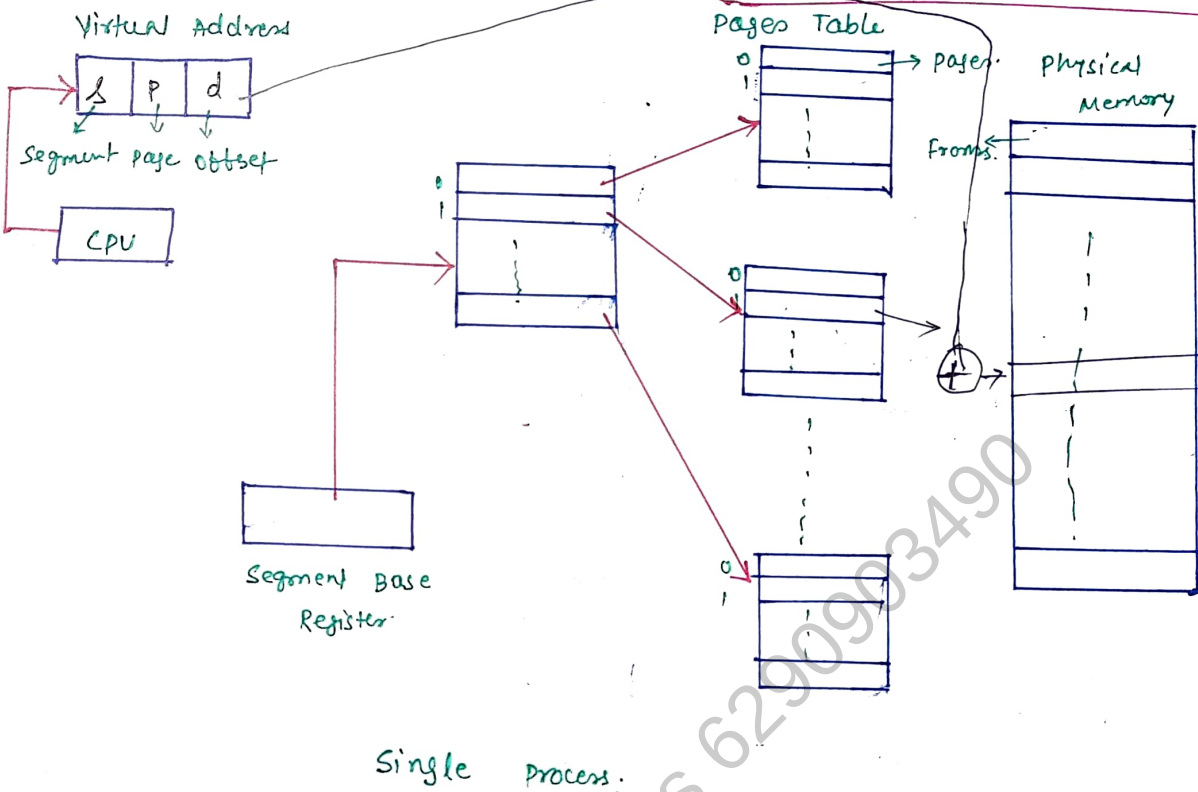
The last bits of the virtual Address are called offset. which is the location difference b/w the byte address we want and the start of the page.

Disadvantages:

- Segmentation with paging causes too lookups.
- We generally use segmentation with paging.

Abhishek Sharma Notes.
Ph: 6290903490.

* Working of segmentation with paging.



Single Process.

The end of Operating System.

Thank you for supporting me Guys. Date: 28/09/2021

12:40 Am.

Join the Group and share the group with your near and dear once.

Thank You

Abhishek Sharma.
3rd year undergraduate at NIT Durgapur.

Best of Luck!

If you like my notes Then please go to the Group and share your Thoughts.