UNIT- 4 (Part II)

NON- CONTIGUOUS MEMORY ALLOCATION:-

It is a memory allocation technique. It allows to store Parts of a single Process in non-contiguous fashion. This different Parts of the same Process can be stored at different Places in the main memory.

Non-contiguous memory Allocation Techniques

- Paging
- Segmentation

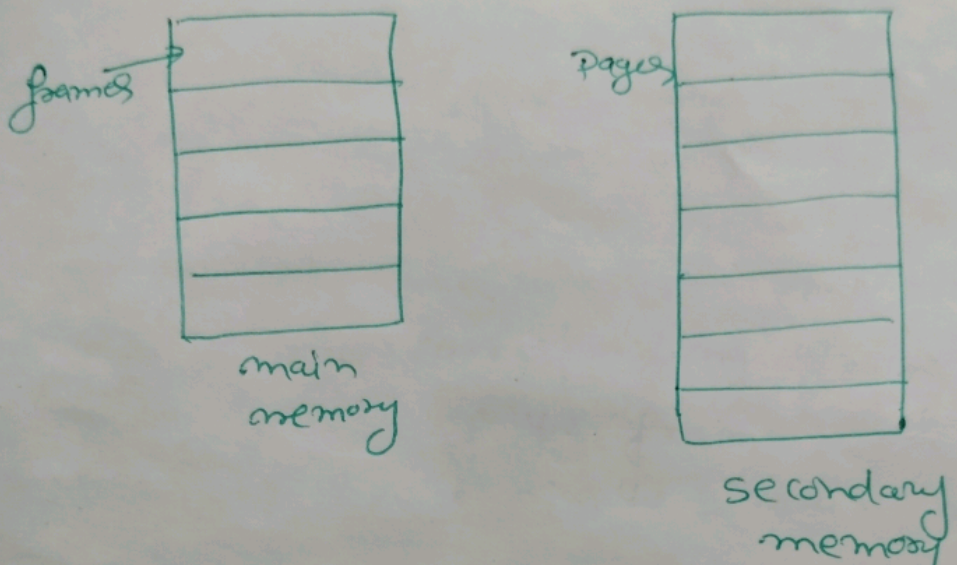
PAGING: ⇒ Paging is a fixed size Partitioning scheme.

⇒ In Paging, secondary and main memory are divided into equal fixed size partitions.

⇒ The Partitions of secondary memory are called as Pages.

⇒ The Partitions of main memory are called as frames.

each process divided into parts where size of each part is same as page size



The Pages of Process are Stored in the frams of main memory depending upon their availability.

LOGICAL ADDRESS AND PHYSICAL ADDRESS:-

⇒ CPU always generated a logical Address. A Physical Address is needed to access the main memory.

⇒ CPU generates a logical address consisting of two parts.

(1) Page number:- specifies the specific Page of the Process from which CPU wants to read the data.

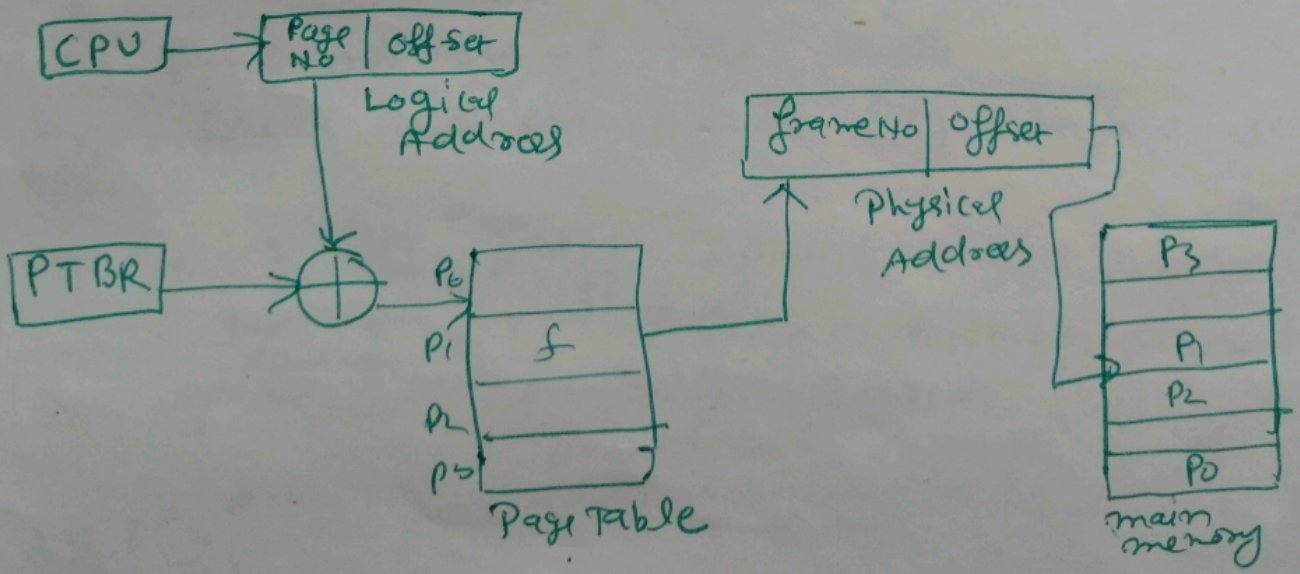
(2) Page offset :- Page offset specifies the specific word on the page that CPU wants to read.

⇒ For the Page number generated by the CPU.

Page Table :- Page Table Provides the corresponding frame number (base address of the frame) where that Page is Stored in the main memory.

⇒ The frame number combined with the page offset forms the required Physical Address.

⇒ Page off-set specifies the specific word that has to be read from that Page.

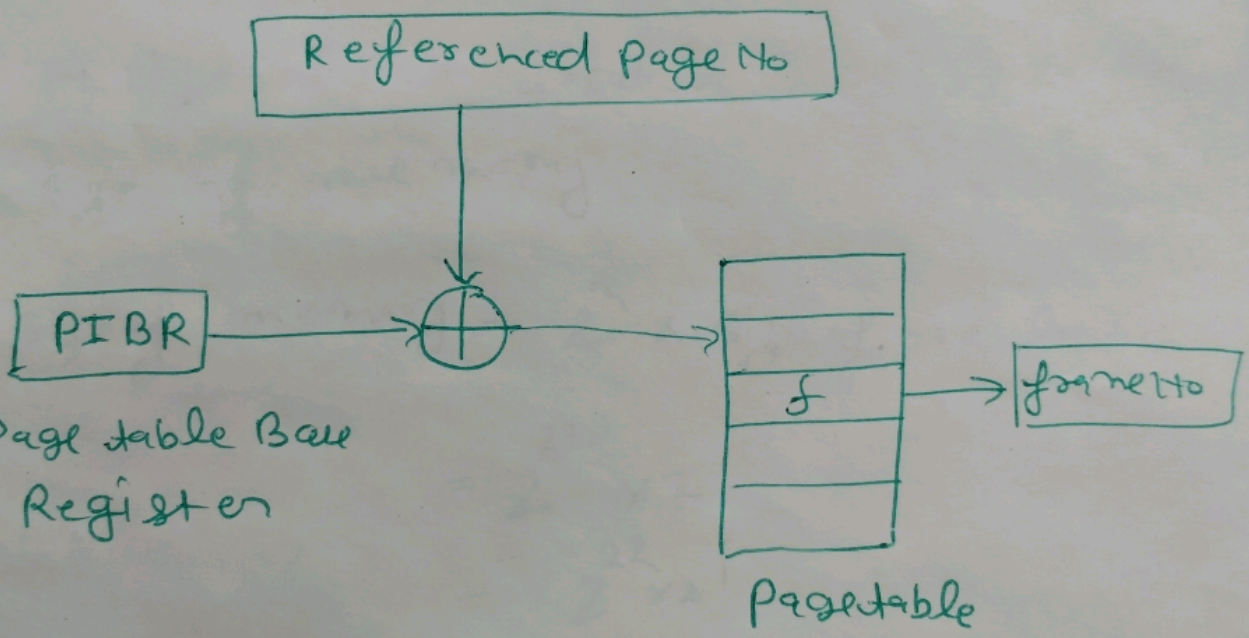


Page Table Base Register (PTBR).

PAGE TABLE:-

Page table is a data structure. Page table is stored in the main memory. Each process has its own independent page table. PTBR (Page table Base Register) contains the Base address of page table.

working:



PTBR (Page table Base Register) provides the Base Address of Page table.
TH Base address of the Page table is added.
It gives the entry of the Page table containing the frame number where the referenced page is stored.

Ex Paging

① Calculate the size of memory if its address consists of 22 bits and the memory is 2-byte addressable.

Sol We have

Number of location possible with = 22 bits

$= 2^{22}$ location

The size of one location = 2 bits.

size of memory

Size of memory = 2^n x Size of one location

$= 2^{22} \times 2$

$= 2^{22} \times 2^1$

$= 2^{23}$ bytes Ans

$= 8$

② Calculate the number of bits required in the address for memory having size at 16 GB Assume the memory is 4-byte addressable.

3a)

Let 'n' number of bits are required

Size of memory = $2^n \times 4$ bytes

$= 2^n \times 4$ bytes

$= 2^{16}$ bytes

$= 16$ GB

$2^n \times 4 = 16G$

$2^n \times 2^2 = 2^{34}$

$2^n = 2^{32}$

$n = 32$ bits

SEGMENTATION:-

Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a Segment which can be allocated to a Process

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 of starting address of the segment in the main memory.

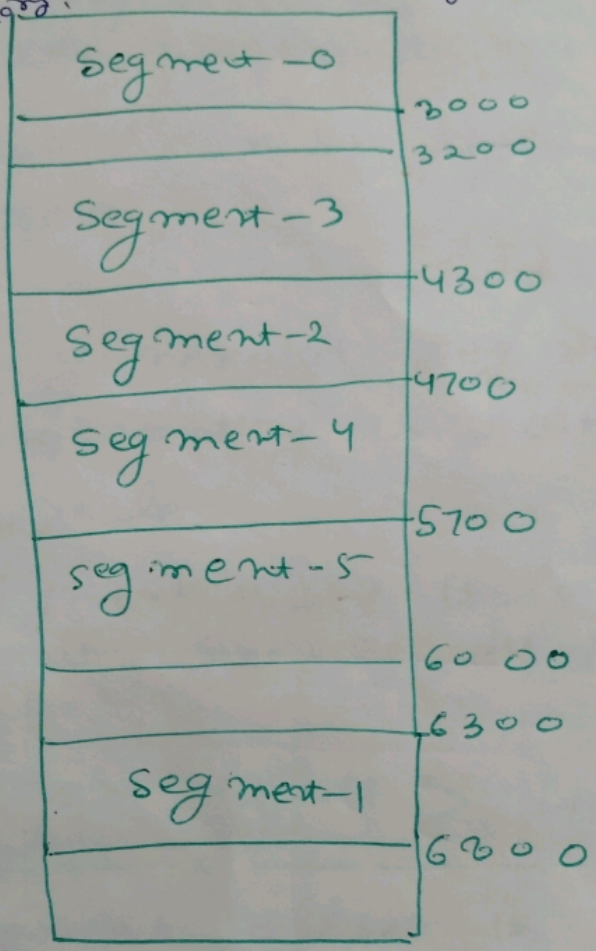
Segment-0	Limit	Base
S_0	1500	1500
S_1	500	6300
S_2	400	4800
S_3	1100	3200
S_4	1000	4700
S_5	300	5700

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.

the Segment table - the segment are stored in the main memory.



Translating Logical Address into Physical Address:-

CPU always generates a Logical address. A Physical Address is needed to access the main memory.

CPU generates a Logical Address consisting of two Parts,

(1) Segment Number
(2) Segment offset.

⇒ Specific segment of the Process from which CPU wants to read the data.

⇒ Specific word in the segment that CPU wants to read.

(2) ⇒ for the generated segment Number corresponding entry is located in the segment table.

⇒ Then, segment offset is compared with the Limit (size) of the segment.

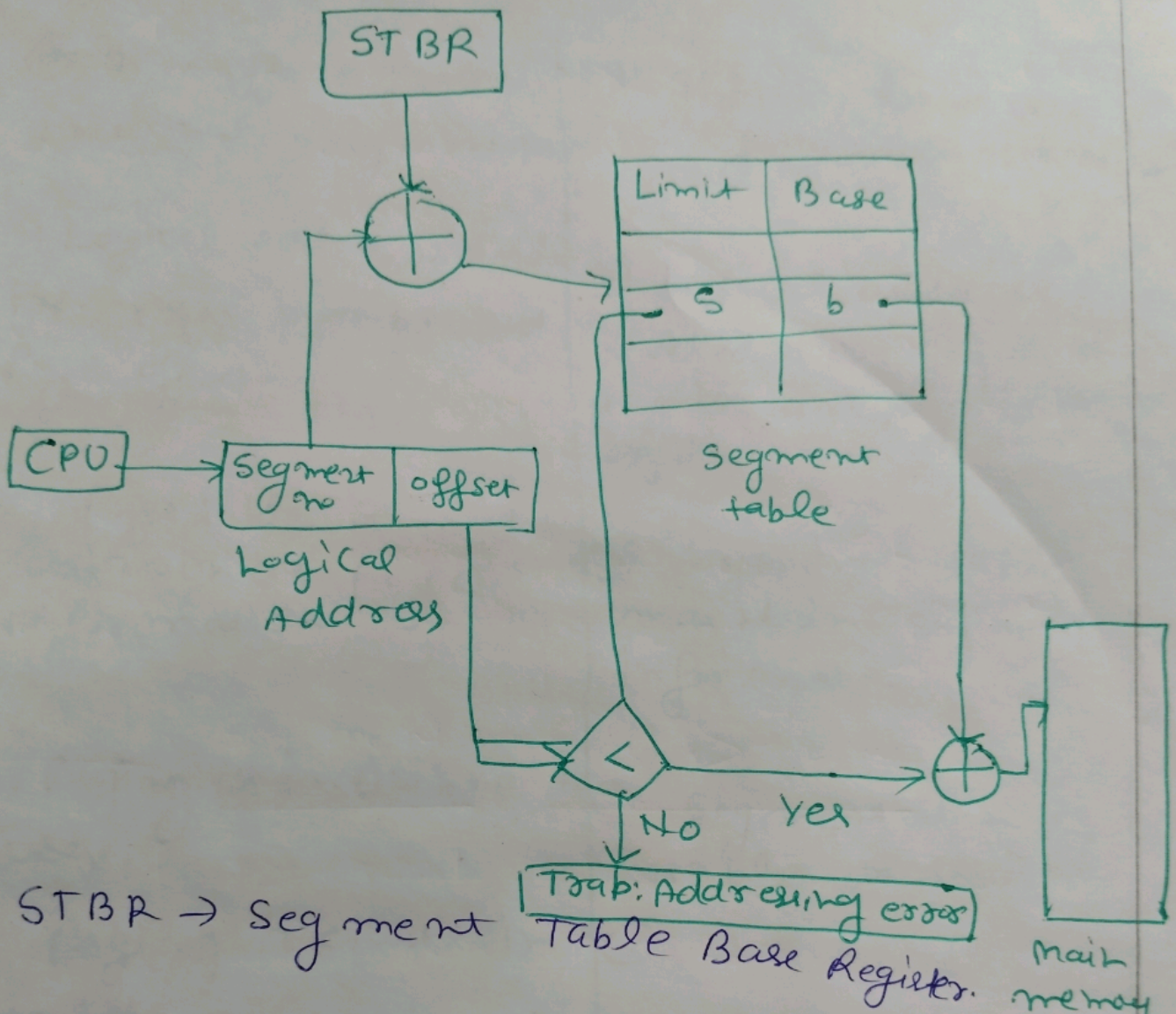
Case-1 Segment offset >= Limit !

⇒ if segment offset is found to be greater then or equal to the Limit, a trap is generated.

Case-2 Segment offset < Limit !

⇒ if segment offset < Limit then request is treated as a valid request.

⇒ The segment offset must always lie in the range [0, limit - 1].



STBR → Segment Table Base Register.

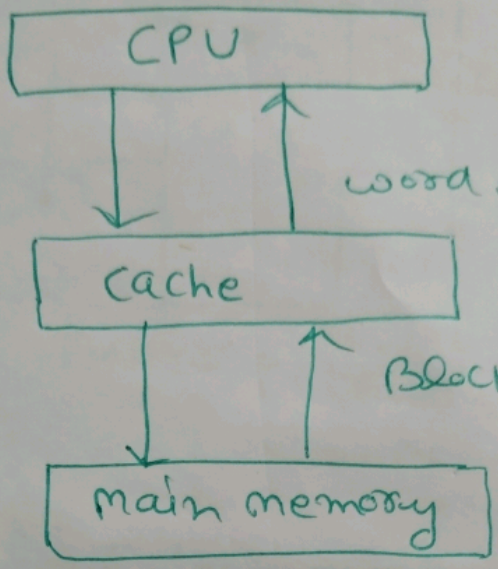
Difference Between Paging and Segmentation

Paging	Segmentation
① Paging divides Programs into fixed size Pages.	① segmentation divides Programs into variable size segments.
② Paging is closer to operating system	② segmentation is closer to user.

(3) It suffers from internal fragmentation.	It suffers from external fragmentation.
(4) Logical address is divided into page number and page offset.	(4) Logical address is divided into segment number and segment offset.
(5) Page table is used to maintain the page information.	(5) Segment table maintain segment information.
(6) Paging is invisible to user.	(6) Segmentation is visible to user.
(7) Paging results in a less efficient system.	(7)

Cache memory:- Cache memory is a Random Access memory. The main advantage of cache memory is its very fast speed.

It can be accessed by the CPU at much faster speed than main memory.

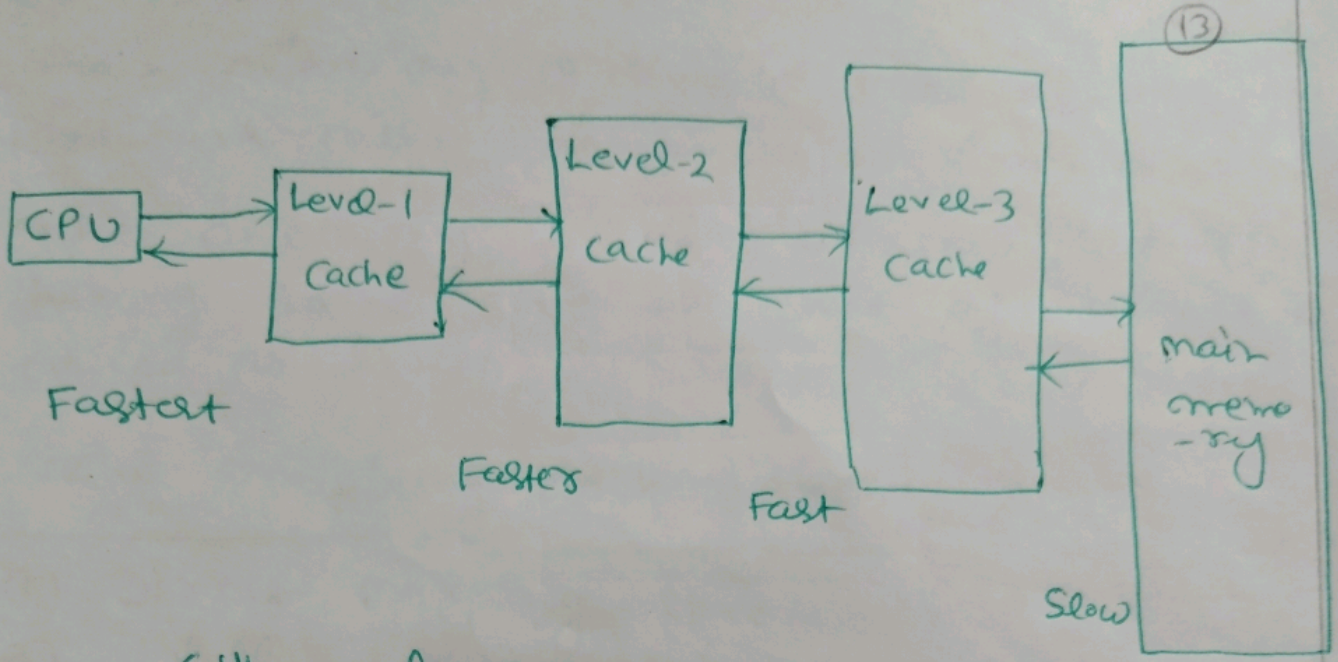


Multiple Cache organization!

⇒ A multiple cache organization is an organization where cache memories of different sizes are organized at multiple levels to increase the processing speed to a greater extend.

⇒ The smaller the size of cache the faster its speed.

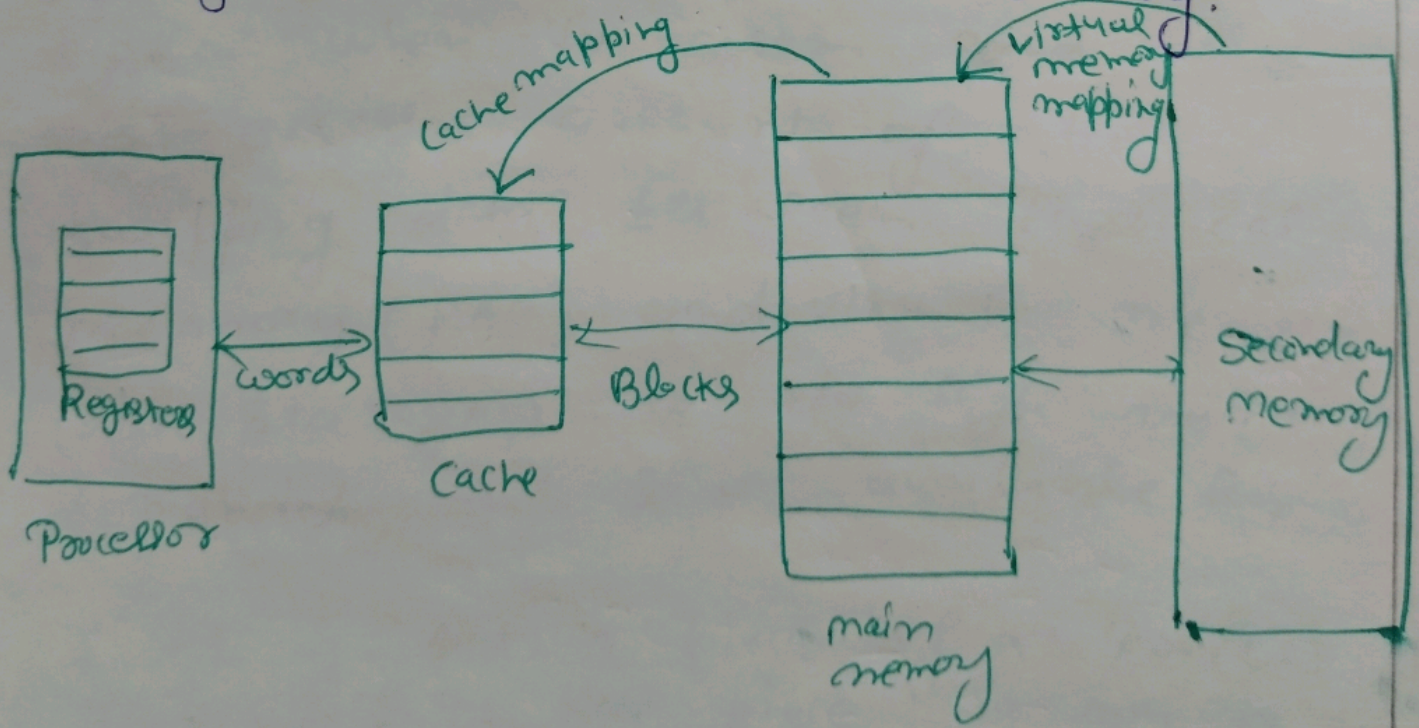
Ex three level cache organization consists of three cache memories different size organized at three different level as —



(three level cache organization)

Cache mapping:-

Cache mapping is a technique by which the contents of main memory are brought into the cache memory.



main memory is divided into equal size Partitions called as blocks or frames.

cache memory is divided into Partitions having same size as that of blocks called as lines.

Cache mapping techniques:-

(1) Direct mapping.
(2) fully Associative mapping
(3) K-way set Associative mapping.

THRASHING:-

Thrashing in operating System is a phenomenon that occurs in computer operating system when the system spend an excessive amount of time swapping data between Physical memory (RAM) and virtual memory (disk storage) due to High memory demanded and low available Resource.

Swapping out a piece of a Process just before that piece is needed.

THRASHING

The Processor spends most of its time swapping pieces rather than executing user instruction. This activity is called the thrashing.

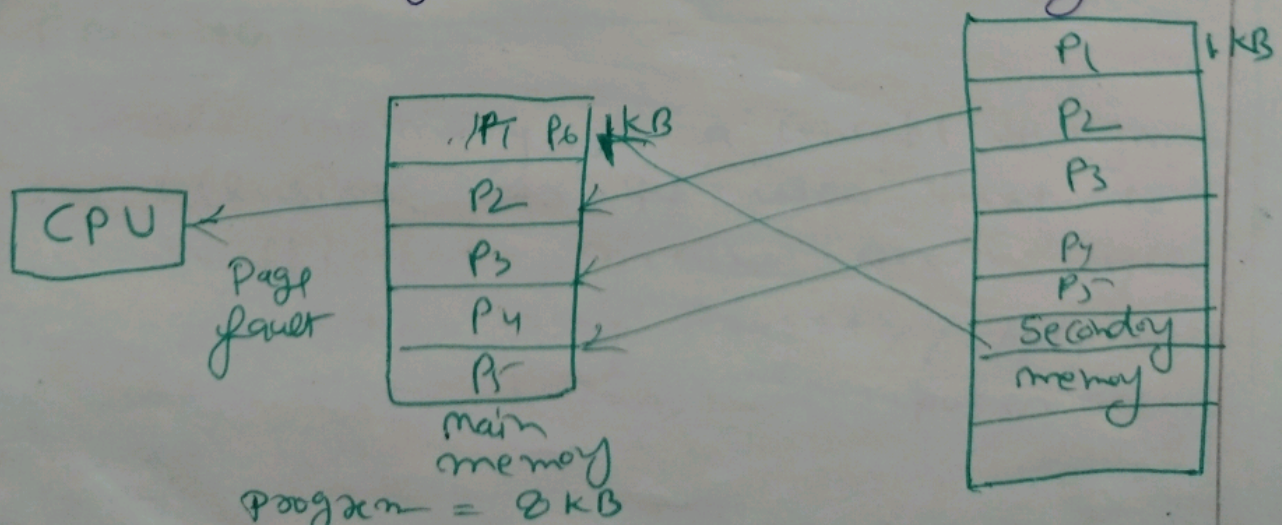
If a Process does not have enough pages the Page fault rate is very high. This leads to

⇒ Low CPU utilization

⇒ operating system think that it needs to increase the degree of multiprogramming.

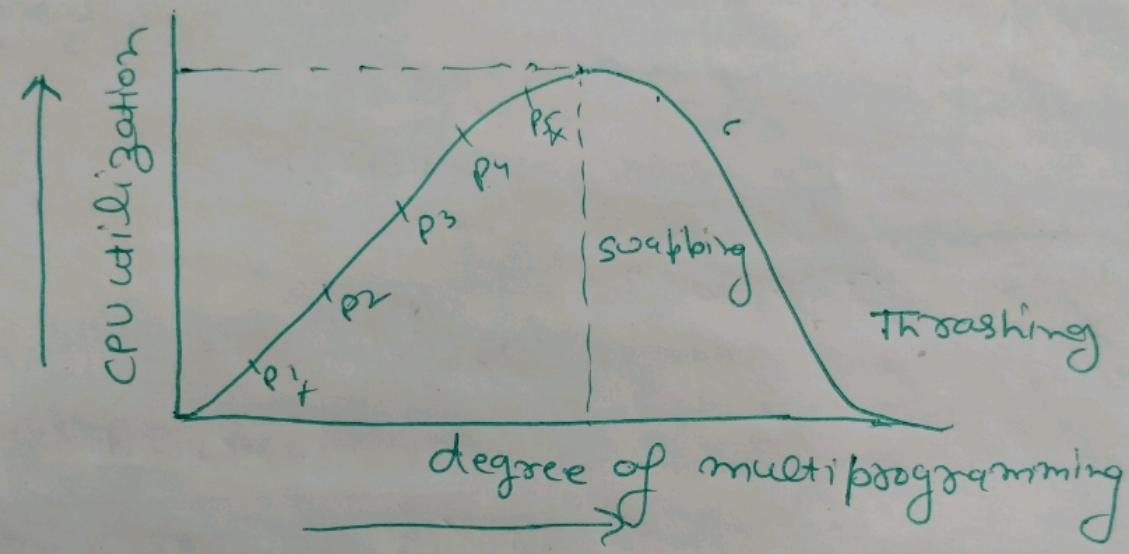
⇒ Another Process added to the system.

* A Process is busy swapping pages in and out.
* This high Paging activity is called thrashing.
* more Paging and less CPU utilization.



P_6 & P_1 & P_2 & P_3

CPU is utlize its complete time in swapping rather than executing instructions

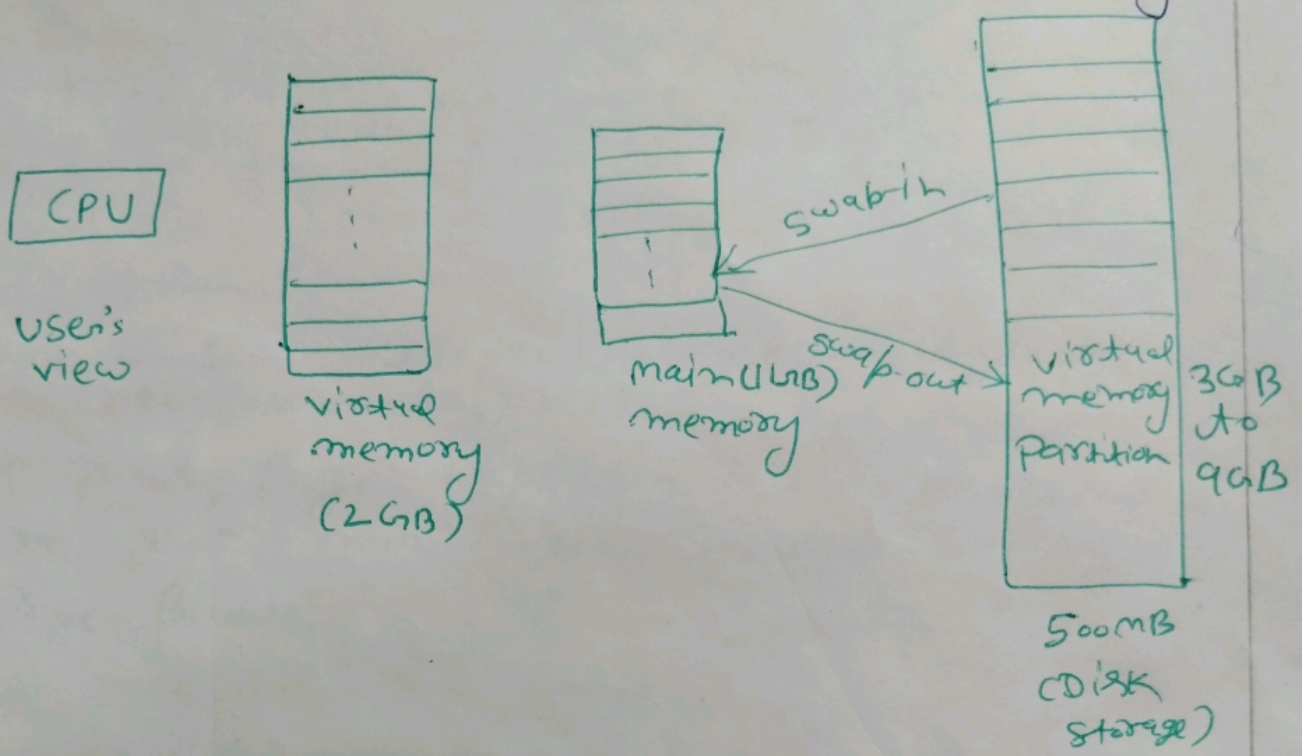


swapping is done the CPU utilization decreasing.

VIRTUAL MEMORY IMPLEMENTATION USING PAGING:

virtual memory is a concept that gives an illusion to the user that he has sufficient main memory (RAM) to execute any Program/application of any size, however computer actually

have relatively smaller main memory.



PAGE REPLACEMENT :-

Page replacement is a process of swapping out an existing page from the frame of a main memory and replacing it with the required page.

All the frames of main memory are already occupied, this a page has to be replaced to create a room for the required page.

Page Replacement Algorithms :-

⇒ FIFO Page Replacement Algorithms,
⇒ LIFO Last in first out
⇒ LRU Least recently used
⇒ Optimal
⇒ Random Page replacement Algorithms,
⇒ MRU (Most recently used)

A Good Page Replacement algorithms is one that minimizes the number of Page faults.

FIFO (first in first out) :-

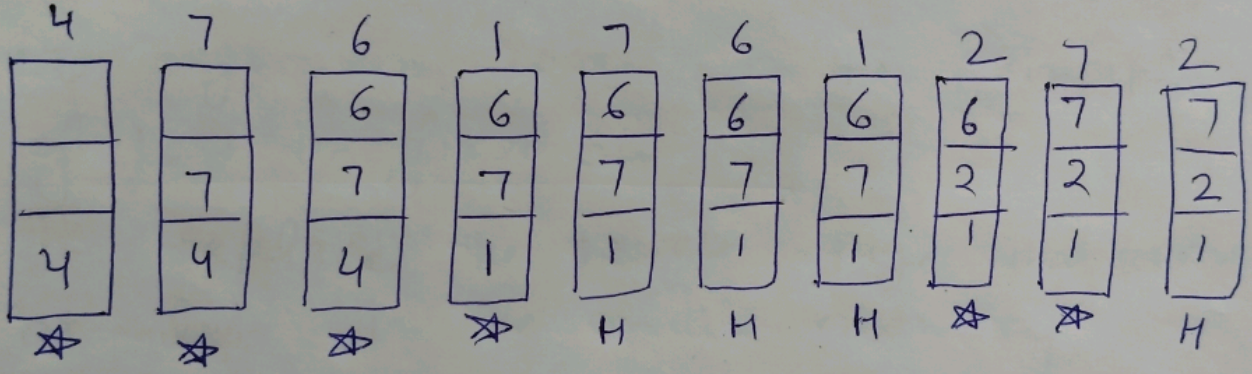
This Algorithm works on the Principle of "first in first out".
It replaces the oldest Page that has been Present in the main memory for the longest time.

It is implemented by keeping track of all page in a queue.

Ex A system uses 3 Page frames for storing Process Page in the main

in the main memory. It uses the first in-first out (FIFO) Page Replacement Policy Assume that all the Page frames are initially empty. what is the total no of Page faults that will occur while Processing the Page reference string given 4, 7, 6, 1, 7, 6, 1, 2, 7, 2

So Page frames = 3



Number of Page Hit = 4

total Page faret = 6

Hit ratio != Total no of Page hit / Total no of references

= 4/10

= 0.4 or 40% Ag

Calculating miss ration:

total Page fault = 6

$$\text{miss ratio} = \frac{\text{Total No of Page miss}}{\text{total No of references}}$$

$$= \frac{6}{10} = 0.6 \text{ or } 60\%$$

Ans

LIFO Page Replacement Algorithms:-

This algorithm work on the principle of Last in first out.

It replaces the newest page that arrived at last in the main memory

ex Page reference String:-

4, 7, 6, 1, 7, 6, 1, 2, 7, 2,

Sol reference String: 4, 7, 6, 1, 7, 6, 1, 2, 7, 2

4	7	6	1	7	6	1	2	7	2
		6	1	1	6	1	2	2	2
	7	7	7	7	7	7	7	7	7
4	4	4	4	4	4	4	4	4	4
*	*	*	*	H	*	*	*	H	H

Hit ratio = $\frac{\text{Total no of Page hit}}{\text{total no of references}}$

$= \frac{3}{10}$

$= 0.30$ or 30%

Miss ratio = $\frac{\text{Total no of Page miss}}{\text{Total no of references}}$

$= \frac{7}{10}$

$= 0.70$ or 70% Ans

LRU (least recently used) Page Replacement Algorithm :-

This Algorithm work on the Principle of "Least recently used".

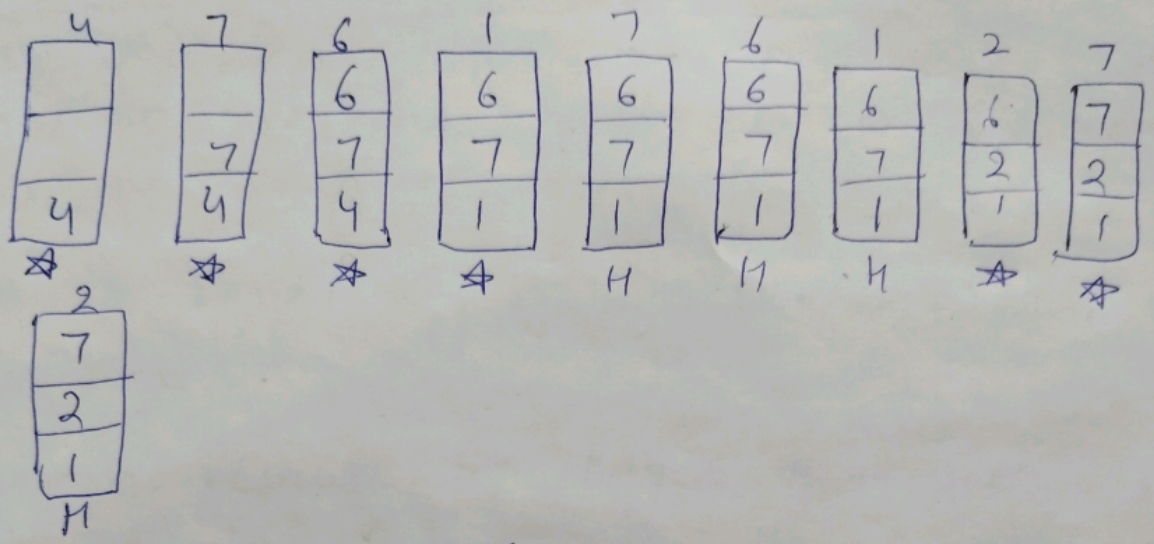
It replace the Page that has not been referred by the CPU for the longest time.

Ex Page reference String

4, 7, 6, 1, 7, 6, 1, 2, 7, 2,

frame size = 3

Ans



Page fault = 6

Page Hit = 4

$$\text{Hit ratio} = \frac{\text{Total no of Page Hit}}{\text{Total no of Page reference}}$$

$$= \frac{4}{10} = 0.4 \Rightarrow 40\%$$

$$\text{miss ratio} = \frac{\text{Total no of Page miss}}{\text{Total no of Page reference}}$$

$$= \frac{6}{10} = 0.6 = 60\%$$

Optimal Page Replacement Algorithm:-

This algorithm replaces the page that will not be referred by the CPU in future for the longest time.

It is practically impossible to implement this algorithm.

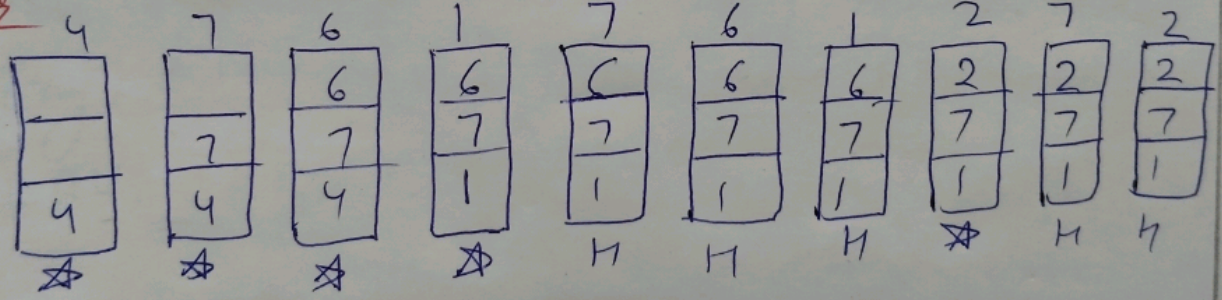
This is because the pages that will not be used in future for the longest time can not be predicated.

Ex Page reference string

4, 7, 6, 1, 7, 6, 1, 2, 7, 2

Page frame size = 3

Sol



Number of Page Hit = 5

No number of Page fault = 5

$$\text{Hit ratio} = \frac{\text{Total no of Hit}}{\text{Page Reference}}$$

$= \frac{5}{10} = 0.5$ or 50%.

$$\text{miss ratio} = \frac{\text{total no of page miss}}{\text{total no of page reference}}$$

$= \frac{5}{10} = 0.5 = 50\%$. Ans

RANDOM PAGE REPLACEMENT Algorithm:-

Random Page Replacement Algorithm as the name Srggest, this algorithm randomly replaces any Page.

⇒ So this algorithm may behave like any other algorithm like FIFO, LIFO, LRU, optimal etc.

Eg reference string

2, 7, 5, 1, 7, 2, 1, 2, 7, 2,

Do yourself

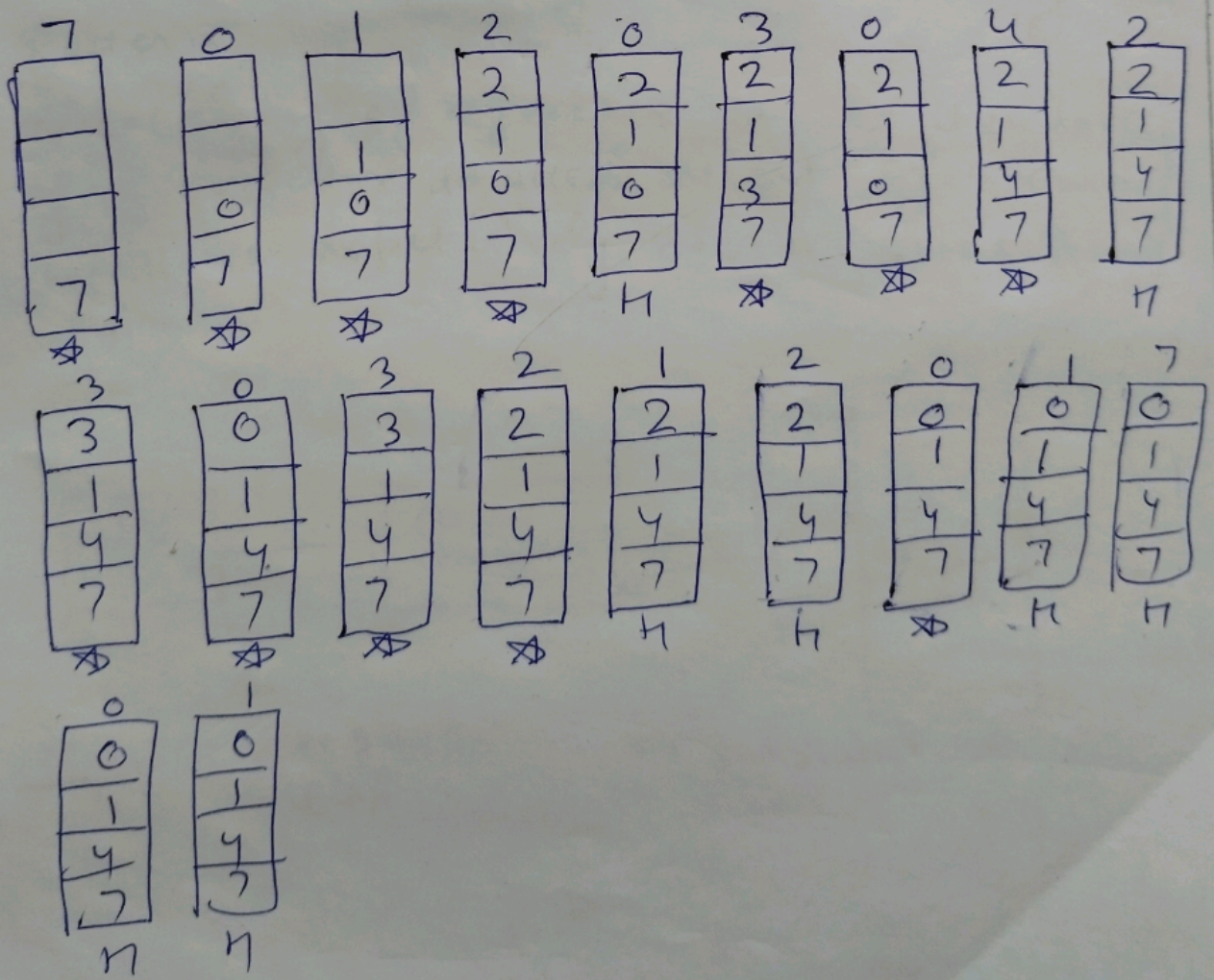
MRU (most recently (frequently used) Page replacement Algorithm)

Replace the most recently (frequently) used Page in Past.

Eg Reference string

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1,

No of frame size = 4



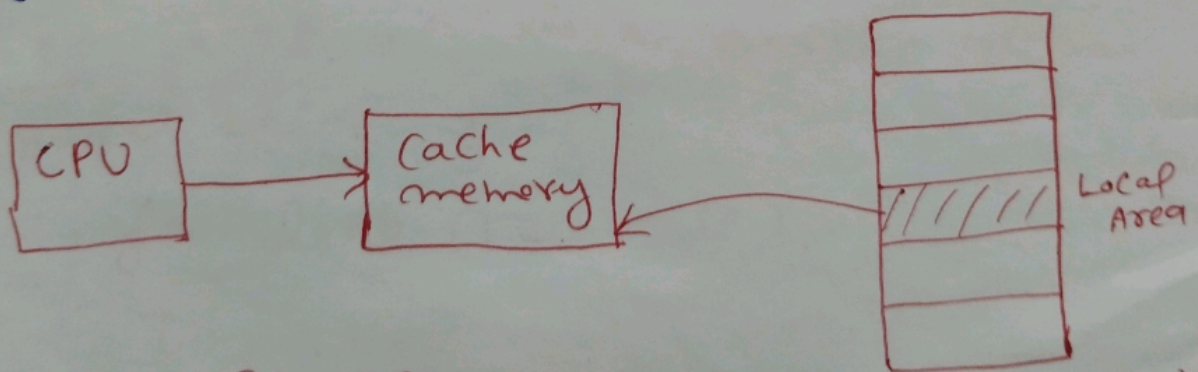
Number of Page fault = 12

No of Page Hit = 8

LOCALITY OF REFERENCE IN CACHE MEMORY

Locality of Reference: Locality of reference is also known as Principle of Locality. Locality of reference is a term for the phenomenon, in which the same values or related storage locations are frequently accessed do depending on the memory access Pattern.

⇒ Locality of reference is the tendancy of a Processor to access the set of memory locations repetitively over a short Period of time.



Ex (1) Program
loop
{
}

(2) Subroutine (functions)
sub()
{
}

(3) Array
a[i]
i = 1, 2, --- n

(4) Table Look up Procedure

Types of Locality of Reference :

① Temporal Locality
② Spatial LOcality

(1) Temporal Locality :- Temporal Locality says that if a program access on memory address there is a good cache that it will access the same address again.

⇒ if refers to reuse of specific data resources within a short period of time.

Ex Loop EXcellent example of temporal Locality

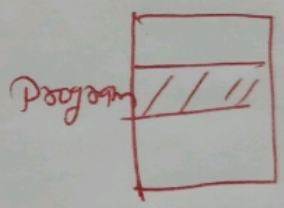
```
for loop (1 to --- 50)
{
 - -
 - -
}
```

(2) Spatial Locality:- Spatial Locality says that if a Program access one memory address. other nearby addresses

⇒ It refers to use of data elements within relatively close storage location.

Ex Nearly "every Program" exhibits Spatial Locality (Because instruction are usually executed in Sequence.

```
Jn( Data )
    Array a[i]
    i-0 , i=1, i=2 - —
```



employees (Id, name, Add (PhoneNo)