

Computer Memory:

→ The memory unit is an essential component of any digital computer since it stores the program and data.

Key characteristics of Computer Memory Systems:① Location

- Processor
- Internal (Main memory)
- External (Secondary)

② Capacity

- word size
- Number of words

③ Unit of Transfer

- Word
- Block

④ Access Method

- sequential
- direct
- Random
- Associative

⑤ Performance

- Access time
- Cycle time
- Transfer rate

⑥ Physical Type

- Semiconductor
- Magnetic
- optical
- Magneto-optical

⑦ Physical Characteristics

- volatile / non volatile
- Erasable / non-erasable

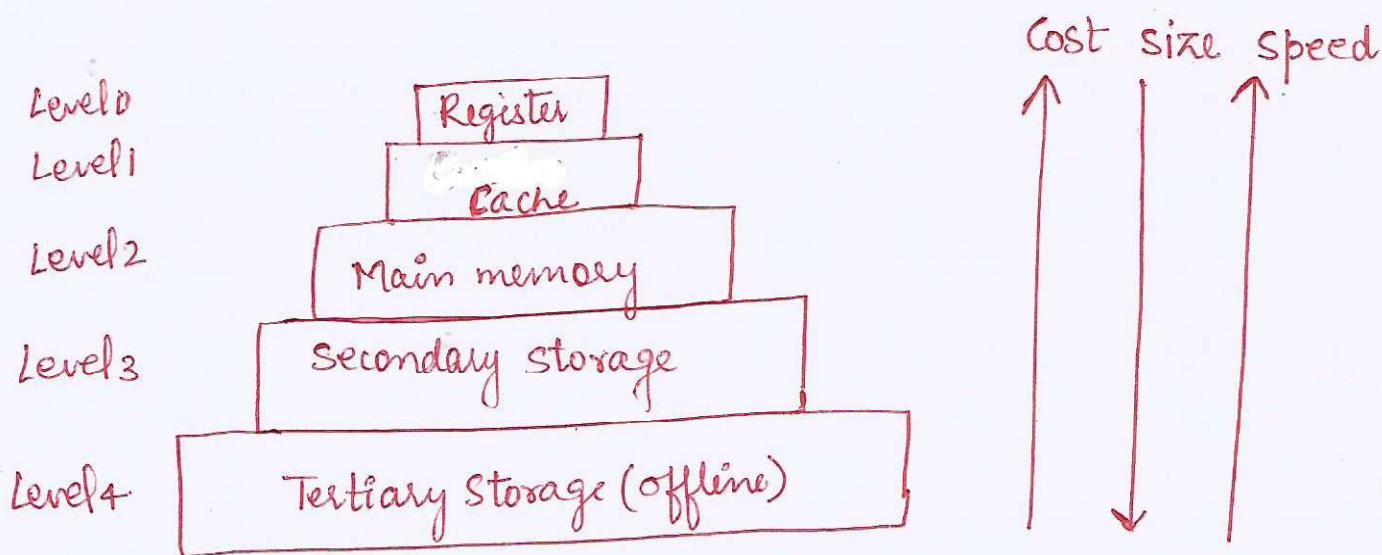
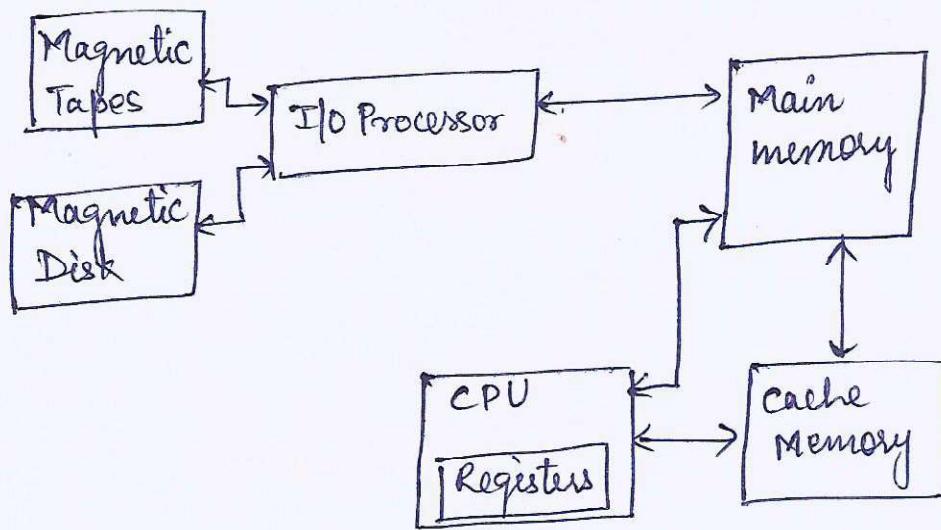
⑧ Organization

②

Memory Hierarchy:

→ The memories used in computer system are organized in a hierarchy.

Memory hierarchy in computer System



Registers:

- Registers are the part of CPU
- fastest
- smallest

- costly
- implemented using Flip Flops

Cache Memory:

- Cache memory is placed between CPU and main memory.
- Faster than Main memory.
- Most recently used information is stored in cache.
- Cache memory is implemented using static RAM.

Main Memory:

- Main memory can directly communicate to the CPU.
- The information is first kept in the main memory to get accessed by the CPU.
- Main Memory is implemented using Dynamic RAM (DRAM)

Secondary Storage: (External Memory)

Level 3: Secondary storage like magnetic disk are present.

- Non-volatile type of memory.
- used as backup storage.
- cheaper and larger than main memory.

Level 4: (Removable Media or offline media)

- The data not being currently used can be kept in removable media. e.g. CD, Pendrive etc.
- Non-volatile
- cheaper
- Larger in size.

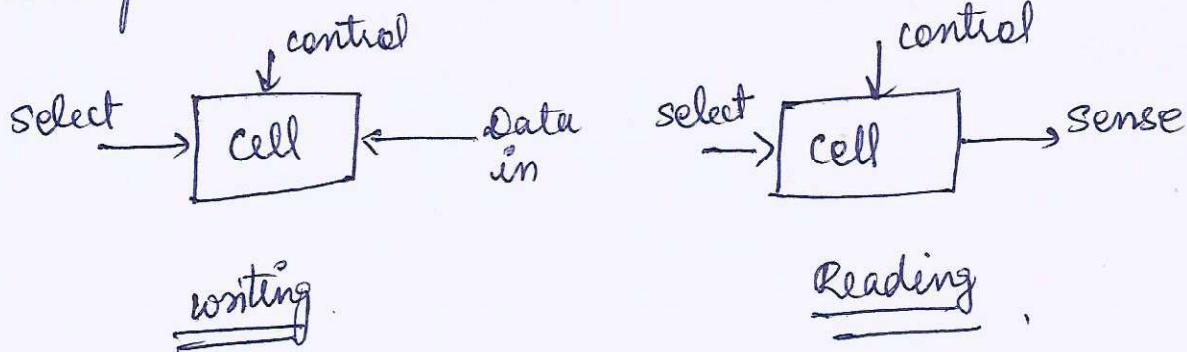
④

Main memory: (Primary Memory or simply memory)

- Semiconductor memory and internal memory.
- can directly communicate to the processor and auxilliary memory with the help of I/O processor.
- Two types of main memory
 - ROM (Read Only Memory) ⇒ Non volatile
 - RAM (Random Access Memory) ⇒ Volatile
- Data transfer b/w main memory and CPU takes place by
 - MAR ⇒ Memory Address Register
 - MDR ⇒ Memory Data Register

Organization: The basic element of semiconductor memory is 'Memory cell'.

- Memory cell represents two states zero and one.
- Memory cell is capable of being written into (at least once), to set the state.
- Memory cell is capable of being read to sense the state.



Types of RAM:

- Static RAM
- Dynamic RAM

Static RAM:

- Faster than DRAM
- Expensive
- Used for cache memory.
- single block of memory needs 6 transistors
- complex uses transistors and latches.

DRAM:

- Slower than SRAM
- Cheaper than SRAM
- used in Main memory
- simple and uses capacitors and very few transistors
- stores charges on capacitors
- needs periodic charging (or refreshing)

ROM: - ROM stands for Read only Memory. The ~~mem~~

- we can only read from this type of memory.
- information is stored in such memories during manufacturing
- used as Bootstrap (a program to ~~to~~ start computer).

Types of ROM

- PROM
- EEPROM
- EEPROM
- Flash Memory

(6)

PROM (Programmable ROM):

- can be modified only once by a user.
- it is not erasable.

EPROM: (Erasable and Programmable ROM)

- can be altered or modified multiple times
- updated each time from initial stage.
- Data erased completely.
- More expensive than PROM.

EEPROM: (Electrically Erasable Programmable ROM)

- can be altered or modified multiple times
- selectively programmed and erased
- More flexible than EPROM.
- More expensive than EPROM.

Flash Memory:

- is a type of EEPROM.
- is programmed and erased in large blocks
- may also be a standalone memory storage device like such as a USB drive.
- can be put between EPROM and EEPROM in both the cost and functionality.

EPROM



entire
chip
erasable.

Flash



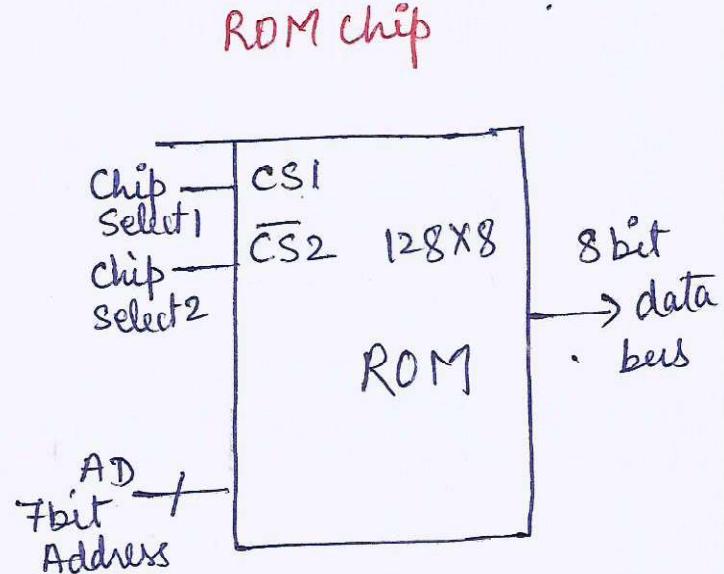
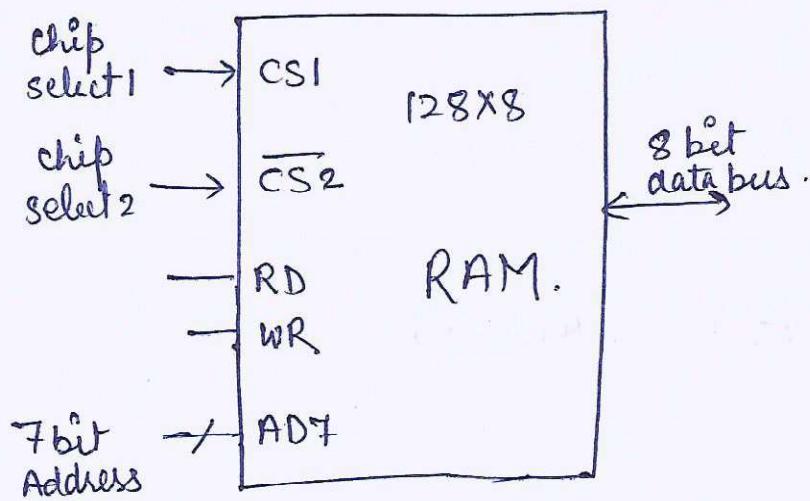
block
erasable

EEPROM



byte erasable

RAM chip:



RAM size $\Rightarrow m \times n$ ————— word size of each word.
 ↑
 denotes
 number of words
 in RAM.

Ques: if RAM chip size is 128x8. Then how many RAM chip will be used to make 2048 bytes of RAM.

$$\text{RAM chip size} = 128 \times 8$$

$$\text{RAM size required} = 2048 \times 8$$

because 2048 bytes

$$\text{Number of chips required} = \frac{2048}{128} = \frac{2^{11}}{2^7} = 2^4 = 16 \text{ chips}$$

The decoder to select 16 chips
 $4 \times 16 \rightarrow$ size of decoder required to select one of the 16 chips.

Ques:

(8) RAM chip size = 128×8

Memory size required 512×8

Number of chips required $\Rightarrow \frac{512}{128} \Rightarrow 4$

Address lines in RAM chip $\Rightarrow 7$

Address bits for memory $\Rightarrow 9$

Inputs of decoder

$$= 9 - 7$$

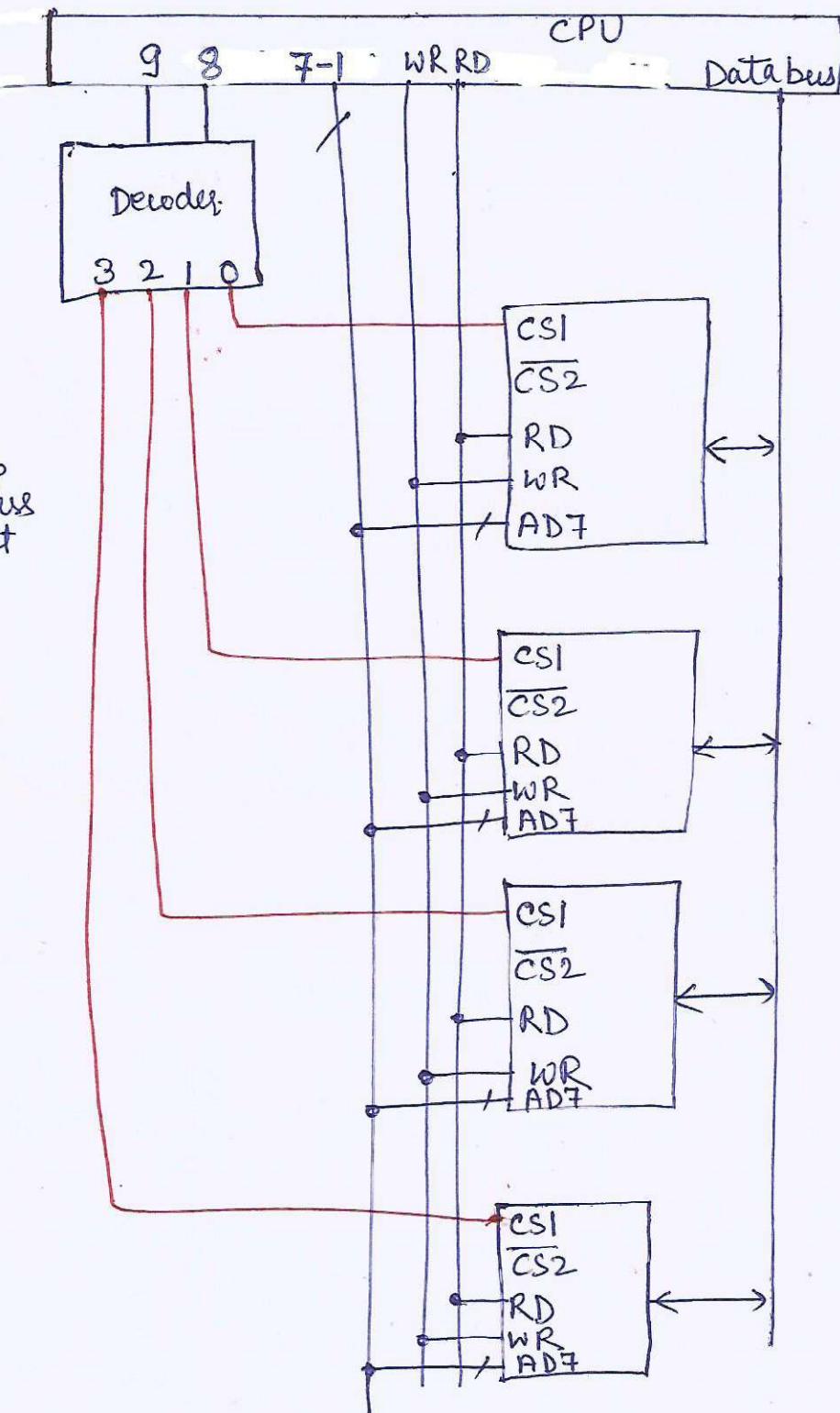
$$= 2$$

\Rightarrow Address bits - chip address bit

Size of decoder

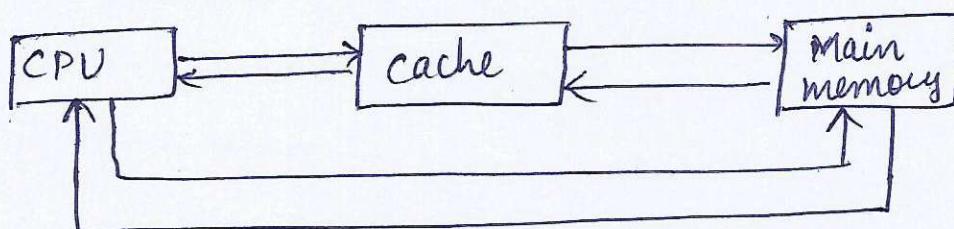
$$= 2 \times 2^2$$

$$= 2 \times 4$$



Cache Memory:

- Cache memory is an internal memory placed between CPU and main memory.
- Most recently used information is kept in cache memory.



- * The cache contains the copy of some portions of main memory when the CPU tries to read a word from main memory. Cache checked is performed, if that word is present in cache or not.
 - if yes, the word is delivered to CPU
 - if no, a block of memory consisting of that word is read into the cache and is delivered to the CPU.

L1

- * Also, cache memory satisfies 'Locality of Reference'. That is the block of data which is fetched from the main memory is likely that there will be more future reference for that block of word.

- Temporal Locality of Reference
- Spatial Locality of Reference.

(10)

Cache Fundamentals :

Cache Hit: when data is found in cache.

Cache Miss: when data is not found in cache.

Hit time: Time taken to access the cache. ($CPU \leftarrow Cache$)

Miss Penalty: Time taken to move data from main memory to CPU cache and then to CPU. when search for data is not found in cache
($CPU \leftarrow Cache \leftarrow Main\ Memory$)

Hit Ratio: The hit ratio is the fraction of accesses which are a hit

Miss Ratio: The miss ratio is the fraction of accesses which are a miss.

Structure of Cache / Main memory:

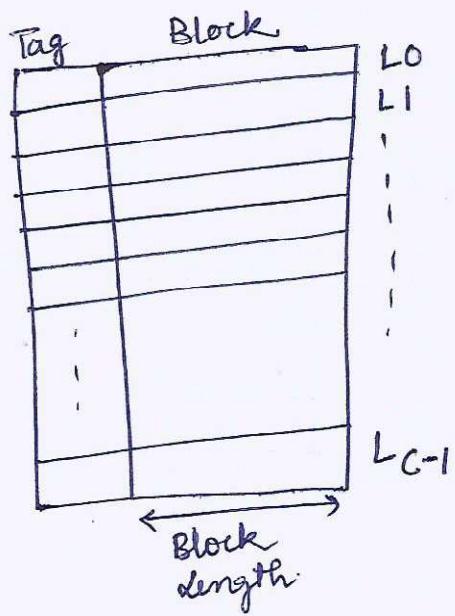
The main memory has 2^n addressable words with n-bit addresses each. For mapping purpose, main memory is divided into fixed length blocks with ~~x~~ words each.

$$\text{Number of blocks } M = \frac{2^n}{X} \text{ blocks}$$

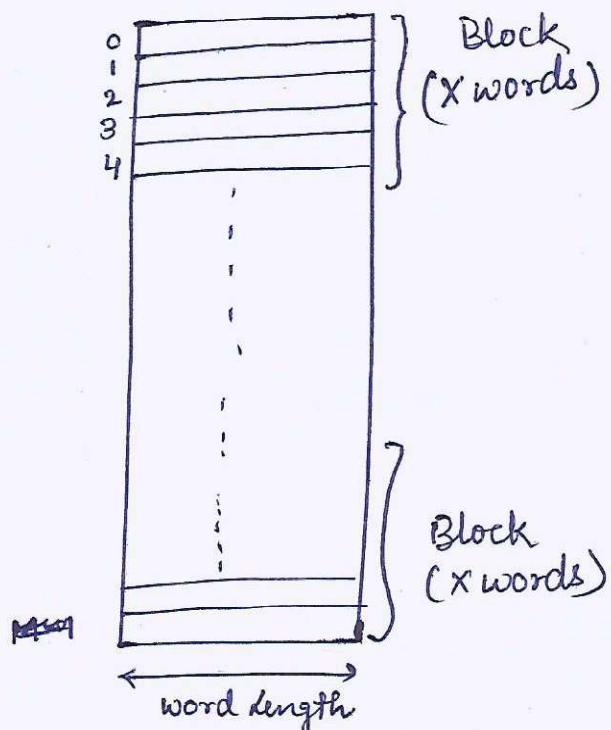
$$= \frac{2^n \text{ words in main Memory}}{X \text{ number of words in each block}}$$

- Cache consists of C lines of X words each
- Each line consists of X words.

$$C \ll M$$



$C \Rightarrow$ Number of lines in cache.
 $M \Rightarrow$ Number of blocks in MM



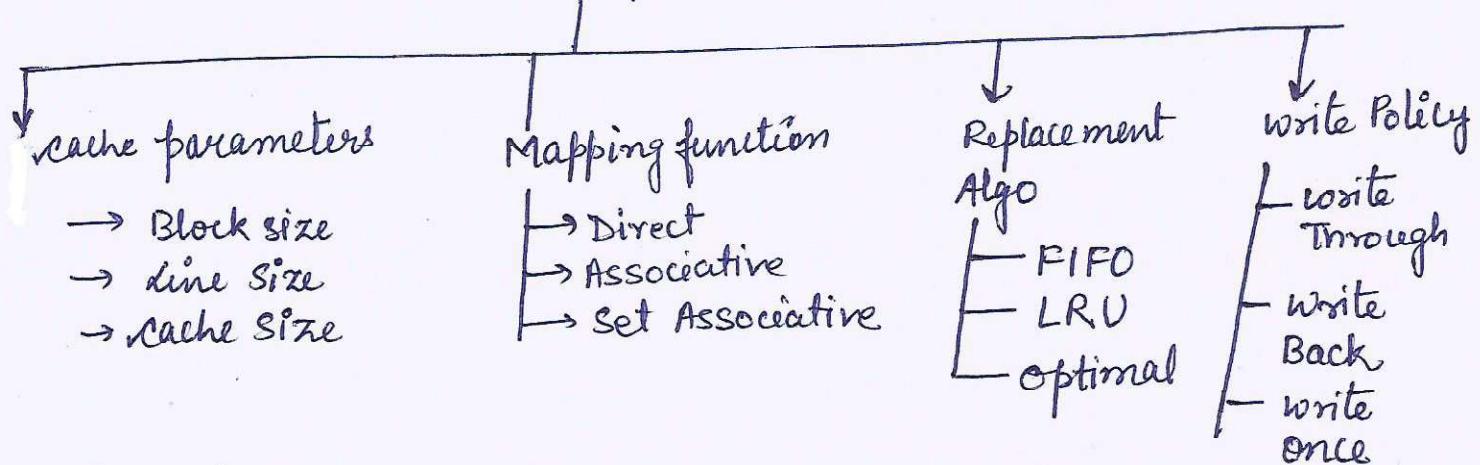
Main memory structure

When a block of memory is read, it is transferred to one ~~one~~ of the lines of cache. Because there are more blocks than lines, one line can not be assigned permanently to one particular block. So each cache line has 'tag' that identifies which particular block is being stored.

(12)

Cache Design issues:

Element of Cache Design



Cache Parameters:

- The size of the cache should be small enough, so that the cost per bit is less but the size should be large enough.
- The larger block size reduce the number of blocks that fits into the cache, so block size should not be large.

Mapping Function:

There are fewer cache lines than memory blocks, an algorithm is required ~~for~~ for mapping of memory blocks into cache lines.

* **Mapping function:** maps the memory block into a cache line

* **Types of mapping functions:** cache mapping techniques

- Direct
- Associative
- Set Associative

Address Mapping or Cache Mapping

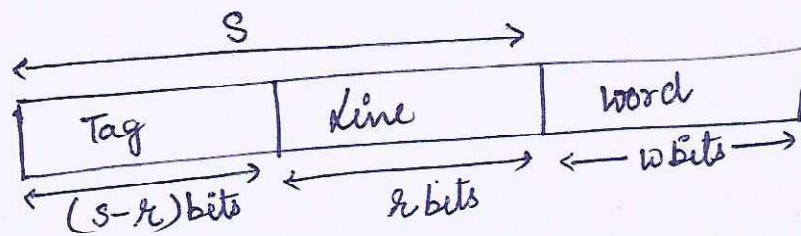
(13)

- The size of memory is smaller than main memory. Therefore, there should be some mechanism to fill the data in the cache. Here the mapping techniques can be used.
- Address mapping is mapping of main memory blocks into cache lines.
- Three techniques:
 - Direct Mapping
 - Associative mapping
 - Set-Associative Mapping

Direct Mapping:

- In this mapping, each block of main memory maps into only one possible cache line.
 - It can be expressed as
- $$i = j \bmod m$$
- where
 $i \Rightarrow$ cache line number
 $j \Rightarrow$ Main Memory block number.
 $m \Rightarrow$ number of cache lines.

- Each main memory address can be divided into three parts



Tag \Rightarrow tells which block is currently being stored in cache
 Line \Rightarrow line bits identifies one of the cache lines ($m = 2^R$)
 W \Rightarrow number of lines $\geq 2^R$

(ii) word bit or Block offset bit identifies which word (or byte in case of bytes in block) is in the block.

~~S → address length~~

Address length = $(S + W)$ bits

Number of words or addressable unit = 2^{S+W} words or bytes

Block size = line size = 2^W words or bytes

Number of blocks in main memory = $\frac{2^{S+W}}{2^W} \Rightarrow \frac{\text{Total number of words in main memory}}{\text{Block size}}$

$$= 2^S$$

Number of bits to identify the blocks in main memory = S bits = Block index

Number of lines in cache = $\frac{\text{Total number of lines in cache}}{\text{Line size}} = 2^R$ words.

Number of bits to identify lines = R = Cache index

Tag bits = Block index - Cache index
= $(S - R)$ bits

Example Direct Mapping

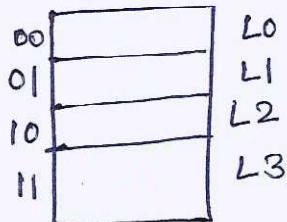
Main memory size = 128 bytes

Cache size = 32 bytes

Block size = 8 bytes

128 Bytes

| | |
|----|------|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |
| 10 | 1010 |
| 11 | 1011 |
| 12 | 1100 |
| 13 | 1101 |
| 14 | 1110 |
| 15 | 1111 |



Block (8 bytes)

(BI) Block index \Rightarrow 4 bits
 (CI) Cache index = 2 bits

Formula \Rightarrow

$$i = j \bmod m$$

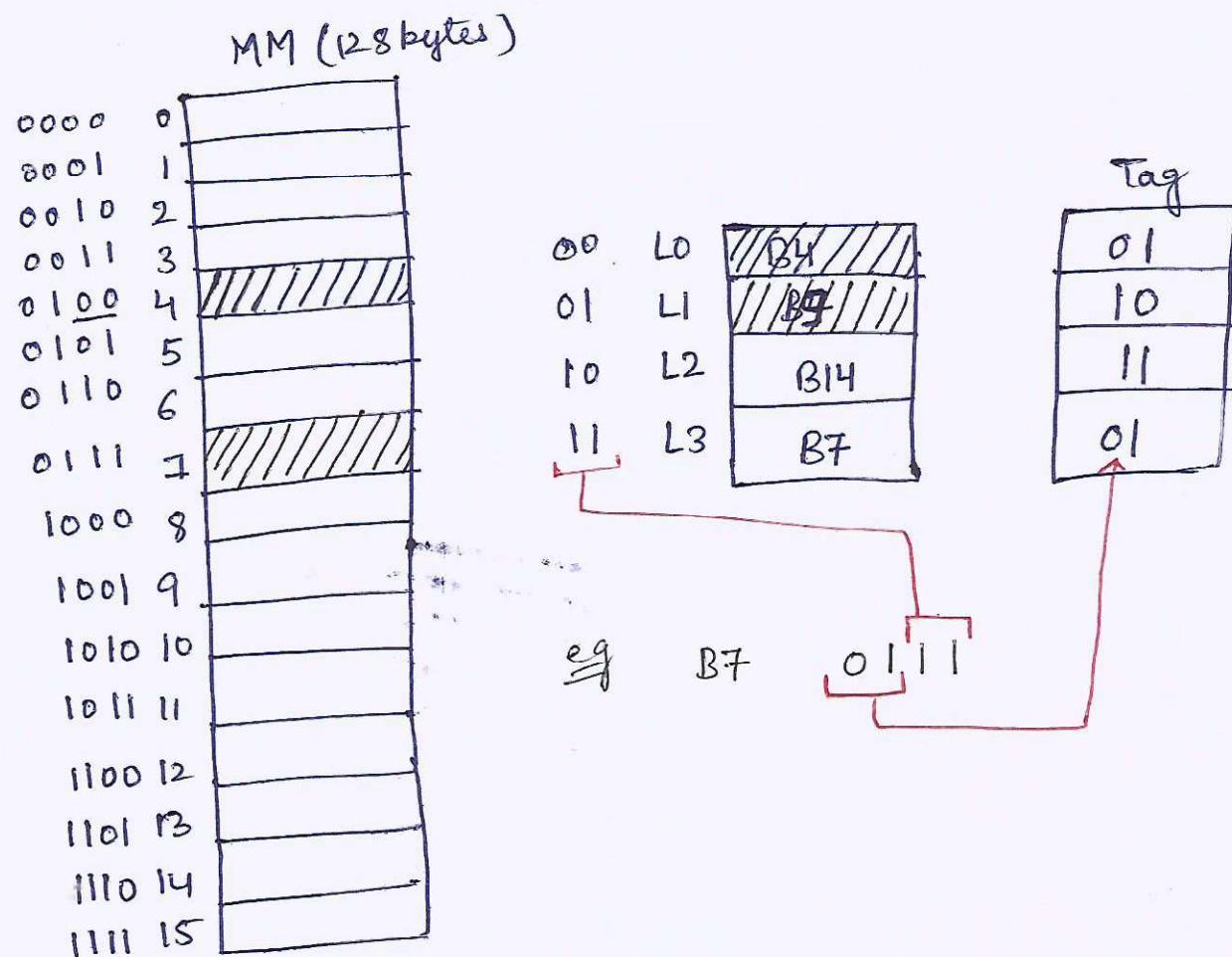
$i \rightarrow$ line number
 $j \rightarrow$ block number
 $m \rightarrow$ number of lines in cache
 $m=4$

| Line number | Blocks can be placed |
|-------------|----------------------|
| L0 | B0, B4, B8, B12 |
| L1 | B1 B5 B9 B13 |
| L2 | B2 B6 B10 B14 |
| L3 | B3 B7 B11 B15 |

* The least significant Block index bits are matched with cache index bits

So line 0 can hold B0, B4, B8 and B12. So CPU to understand which block in the cache, it maintains some extra information bit which is called tag bits.
 → So for each line CPU holds tag bits

- * In direct mapping Tag holds the ~~(BI-CI)~~ MSB bits of the block index BI.
- Tag holds the Most significant bits of the block.



Disadvantage of Direct mapping :

- Direct mapping imposes restriction on the placement of a particular memory block into a particular cache line.
- Poor hit ratio.

Advantages: simple to implement.

- Number of blocks in main memory = $\frac{\text{Main memory size}}{\text{Block size}}$
 $= \frac{128 \text{ bytes}}{8 \text{ bytes}} = 16 \text{ blocks}$

- Number of lines in cache = $\frac{\text{Cache memory size}}{\text{Block size}}$
 $= \frac{32 \text{ bytes}}{8 \text{ bytes}} = 4$

- Block index bits = $\log_2 (\text{Number of blocks})$
 $= \log_2 16 = \log_2 2^4 \Rightarrow 4$

Block index = 4 bits

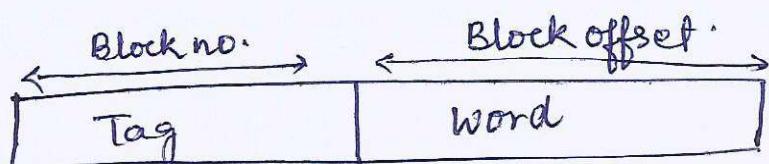
- Cache index bits = $\log_2 (\text{Number of lines})$
 $= \log_2 4 = \log_2 2^2$
 $= 2 \log_2 2$
 $= 2$

- Tag bits = Block index - Cache index
 $= 4 - 2$
 $= 2 \text{ bits}$

- Address bits = $\log_2 (\text{Main memory words})$
 $= \log_2 (128)$
 $= 7 \text{ bits}$

Associative Mapping: or Fully Associative Mapping

- Overcomes the problem of direct mapping method.
- In associative mapping any main memory block can be placed in any cache line.
- whenever a main memory block is copied into cache line, the complete block index is placed in the tag bits.
Tagbits = Block index bits
- Address bits can be divided into two parts

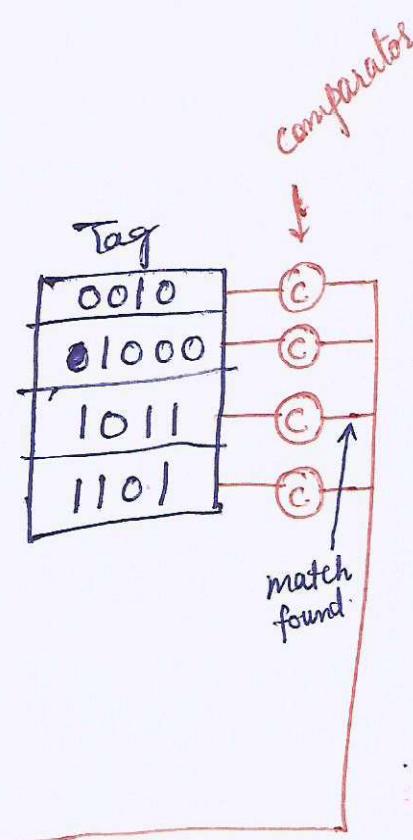
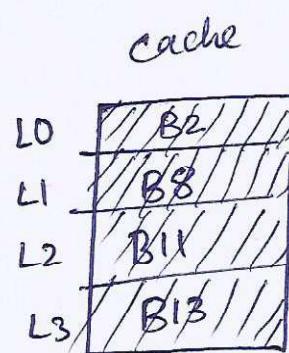
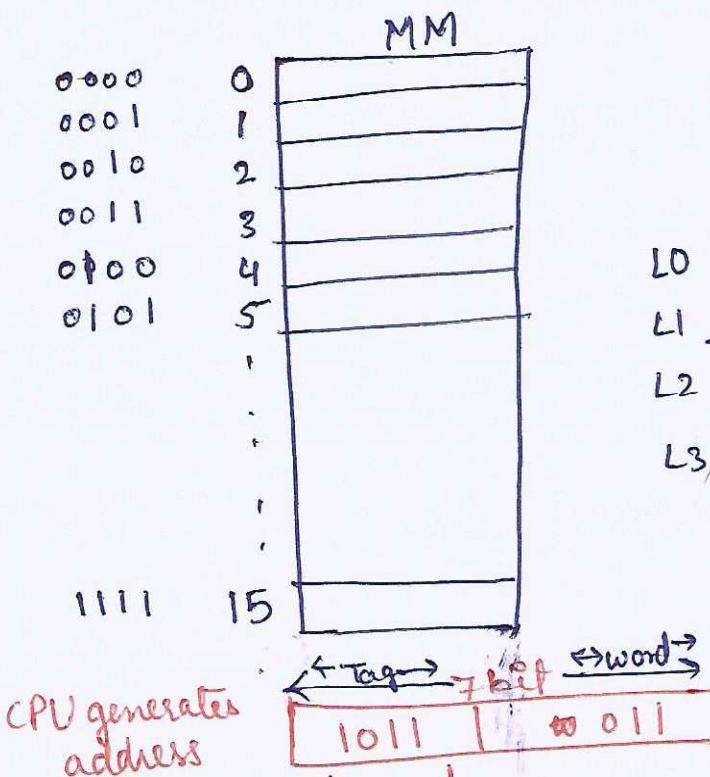


Example Main memory size = 128 bytes

Cache size = 32 bytes

Block size = 8 bytes.

- ⇒ Number of blocks = 16 BI = 4
- ⇒ Number of lines = 4 CI = 2



Advantages:

- Very High Hit ratio
- better block placement strategy which reduced contention.

Disadvantages:

- Associative memory is costly
- significant increase in tag length
- complex
- Searching is an overhead

Set Associative ~~Memory~~ Mapping

In set associative mapping, cache lines are grouped in sets. Grouping is done in K-way where K is the number of lines in a set.

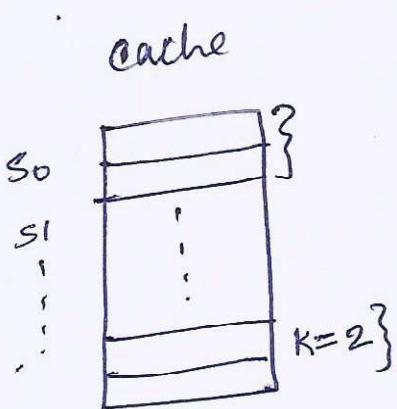
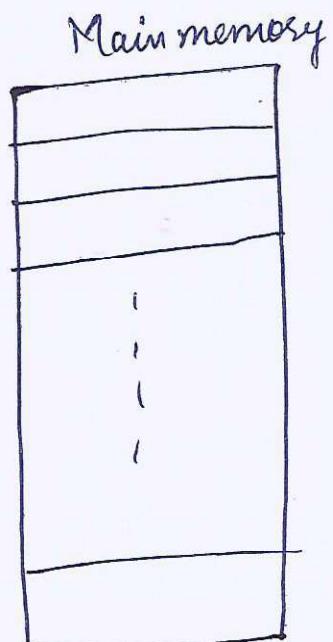
(20)

Set associative mapping technique is the combination of direct mapping and associative mapping.

In this mapping the memory blocks are connected to a set using the logic of direct mapping (this mapping is from main memory to set). and inside the set, the mapping is associative mapping.

Advantage: The advantage is, the associative mapping is only for a set. The number of lines in a set is only k. Therefore the number of comparisons will be less. So the search time will be reduced.

So the set associative is better than full associative mapping.



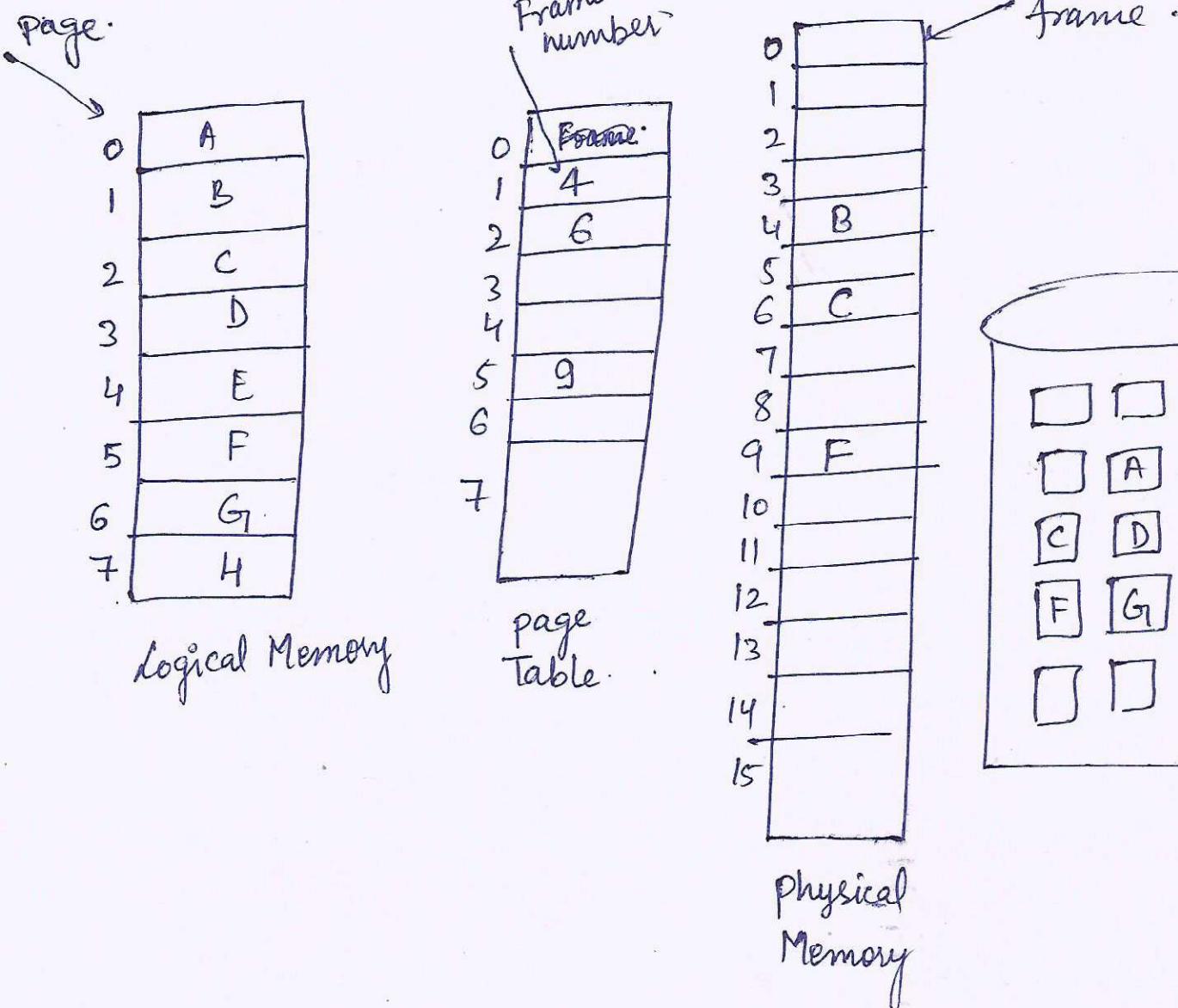
Grouping
K-way
Haserkar
e.g. \Rightarrow 2-way
Set

$K = \text{number of lines in a set}$

Virtual Memory

24)

- Virtual memory is a technique that lets the execution of different processes which are not totally in main memory.
- One main benefit of this method is that programs can be larger than the physical memory (main memory).
- The virtual address space of any process is defined as the logical (or virtual) view of how any process gets stored in memory. Normally this view is where a process begins at a certain logical address (say address location 0) and then exists in continuous memory. Although, the fact is physical memory (main memory might be structured in the form of page frames. and physical page frames are assigned to a process may not be adjacent.)
- Memory Management Unit (MMU) maps the logical pages to physical page frames in memory.
page = logical memory divided into fixed sized blocks called pages.
frame \Rightarrow physical memory divided into fixed sized blocks called frames
page size = frame size.



when a page is referenced, if it is found in main memory, then page hit occurs. If not then a page miss occurs, and is called a page fault.

⇒ A computer can address more memory than the amount physically installed on the system. This extra memory actually called a virtual memory and it is the section of a hard disk that's set up to emulate the computer's RAM.

Replacement Algorithm:

- when a page fault (page being accessed is not in memory), the operating system has to choose a page to remove from memory to make space for the page that has to be brought in.
- The criteria for selecting a page for replacement constitute a replacement policy.
- The main goal is choosing a replacement policy to maximize the hit ratio.
- Replacement Algorithms are:
 - FIFO
 - LRU
 - optimal

First In First Out Page Replacement Algorithm:

- FIFO select a page for replacement which has been least recently loaded.

- Easy to implement
- It does not give high hit ratio

Ques Assume that memory consists of 3 frames and following

pages are referenced

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

7, 0, 1,

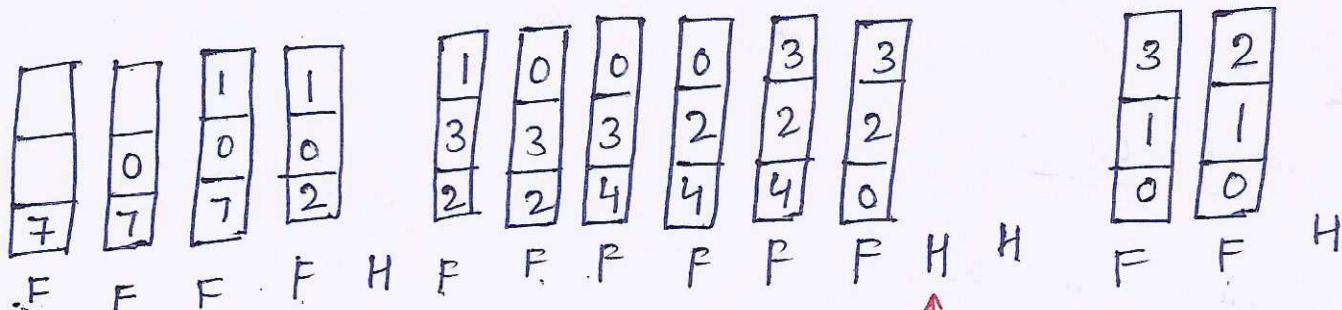
Show the behaviour of page replacement using FIFO.

Solution

Number of frames = 3, initially all frames are empty

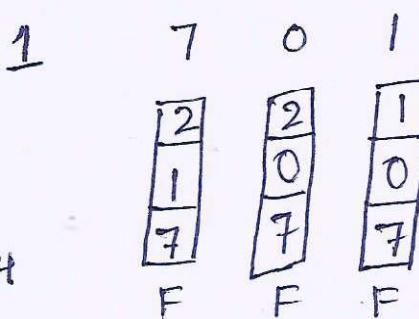
Reference string

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



F denotes page fault occurred. page 0 was not available in frames

H denotes page hit as page 3 found in frames.



Total number of page fault = sum of all page faults.
= 15

Least Recently Used :

- LRU policy replaces the page in memory that has not been referenced for longer time.
- This algo replaces a page that has been unused for the longest time

Reference string Given :

{ Number of frames = 3, all frames initially empty }



Here 2 replaces 7 because
Most recently used 1, then page 0
and least recently used is page 7

Number of page fault = 12

Optimal Page Replacement Algorithm:

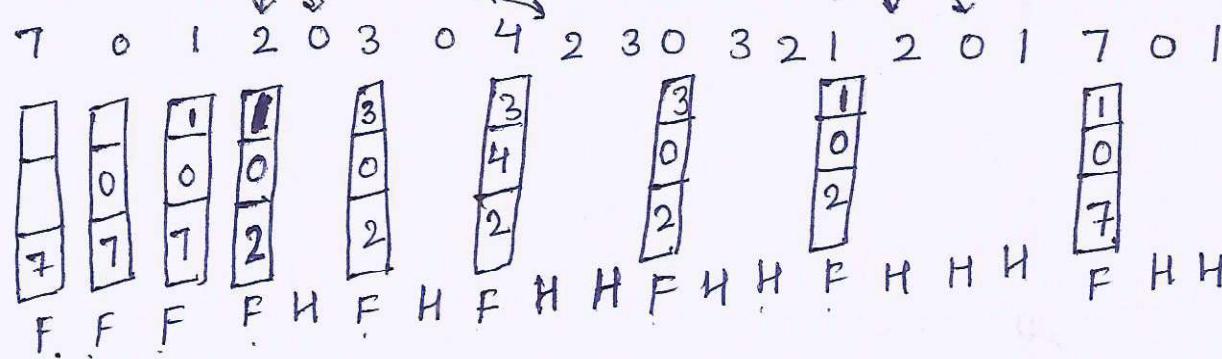
- It optimal page replacement policy, that page is replaced which will not be used for the longest period of time.
- It has lowest page fault.
- This technique is not implemented because it looks into future references.

Number of frames = 3

Reference string :

At this point, page 7 will be replaced

so page 3 will be replaced



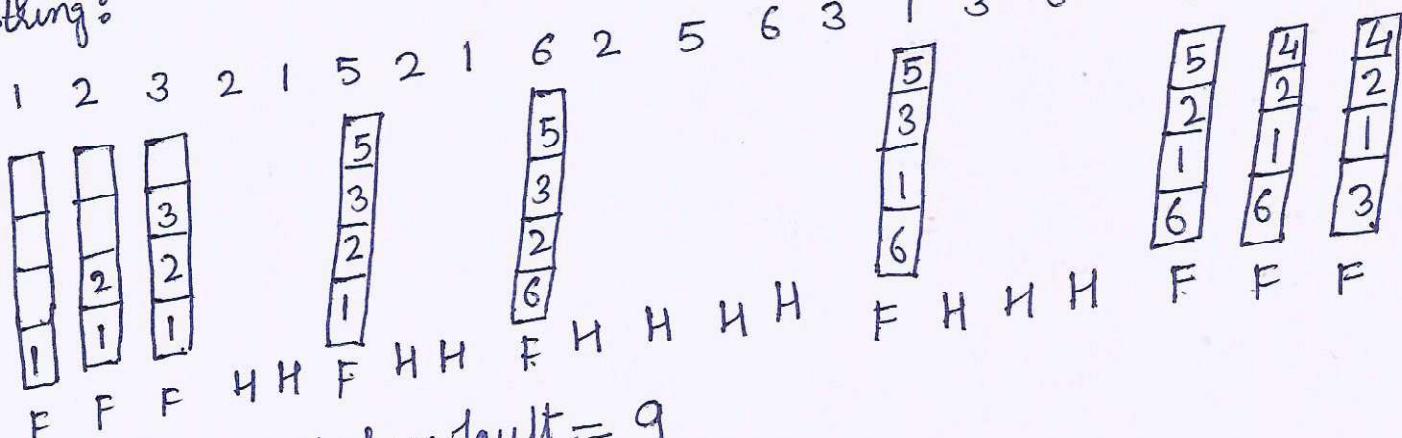
Number of page fault = 9

Ques 2: Consider a reference string 1, 2, 3, 2, 1, 5, 2, 1, 6, 2, 5, 6, 3, 1, 3, 6, 1, 2, 4, 3. The number of frames in memory are 4. Find out the number of page faults respective to: optimal page replacement, FIFO and LRU page replacement algorithm.

Solution : Number of frames = 4

FIFO Algo.

Reference String :



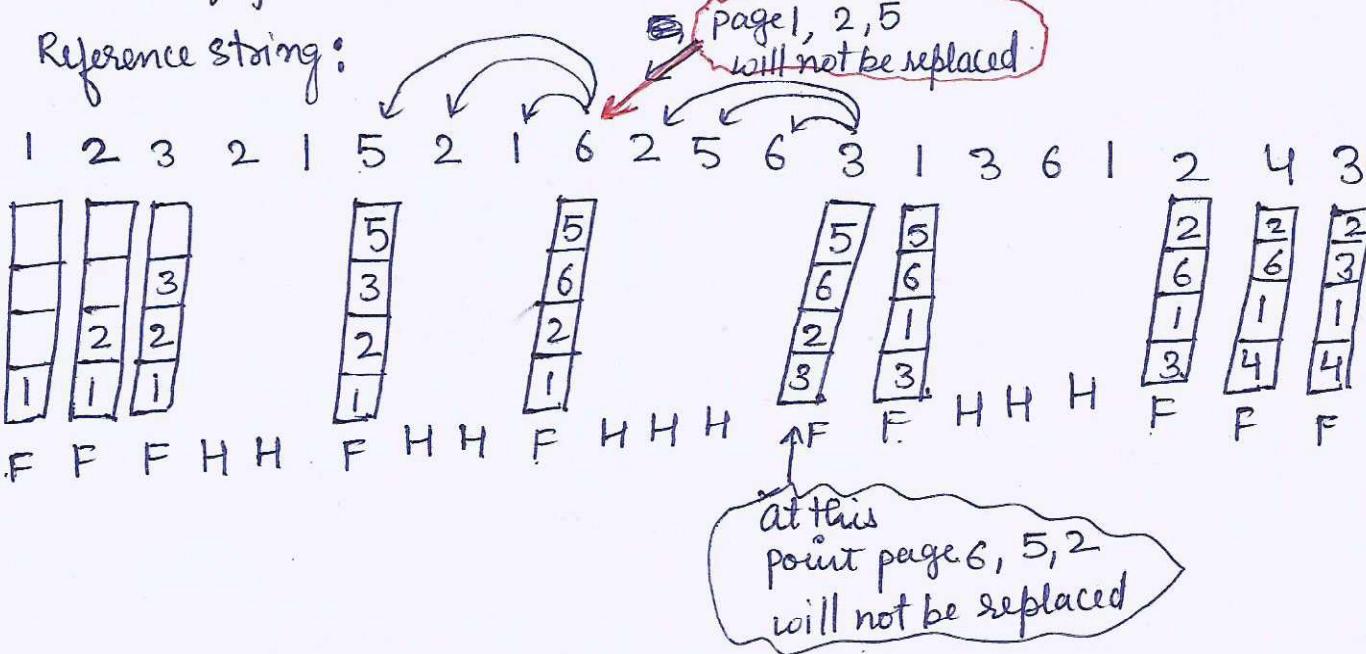
Number of page fault = 9

LRU:

Number of frames = 4

(30)

Reference string:

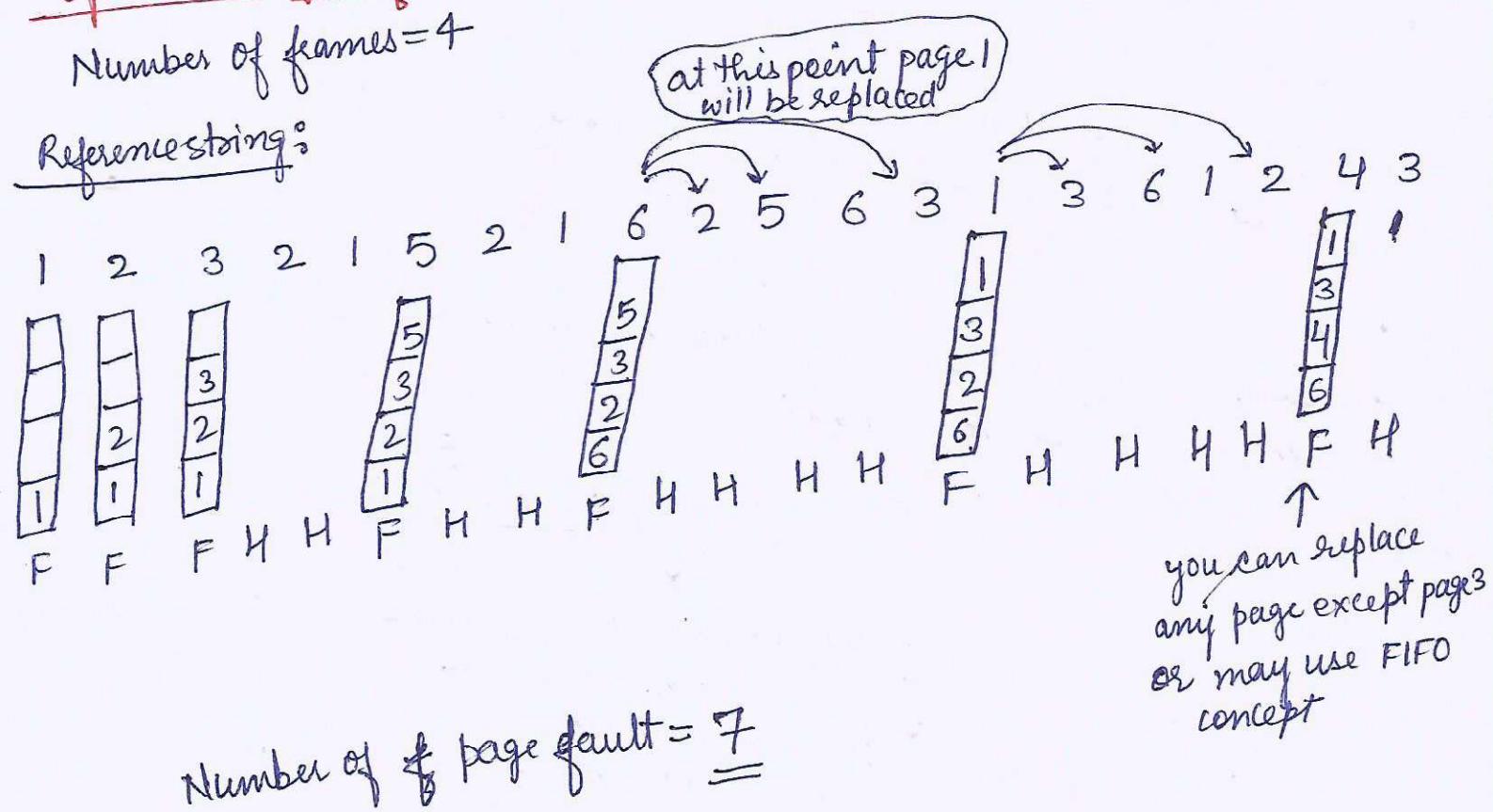


Number of page fault = 10

optimal Page Replacement

Number of frames = 4

Reference string:

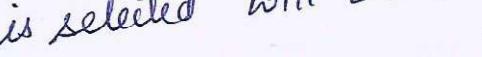


Number of page fault = 7

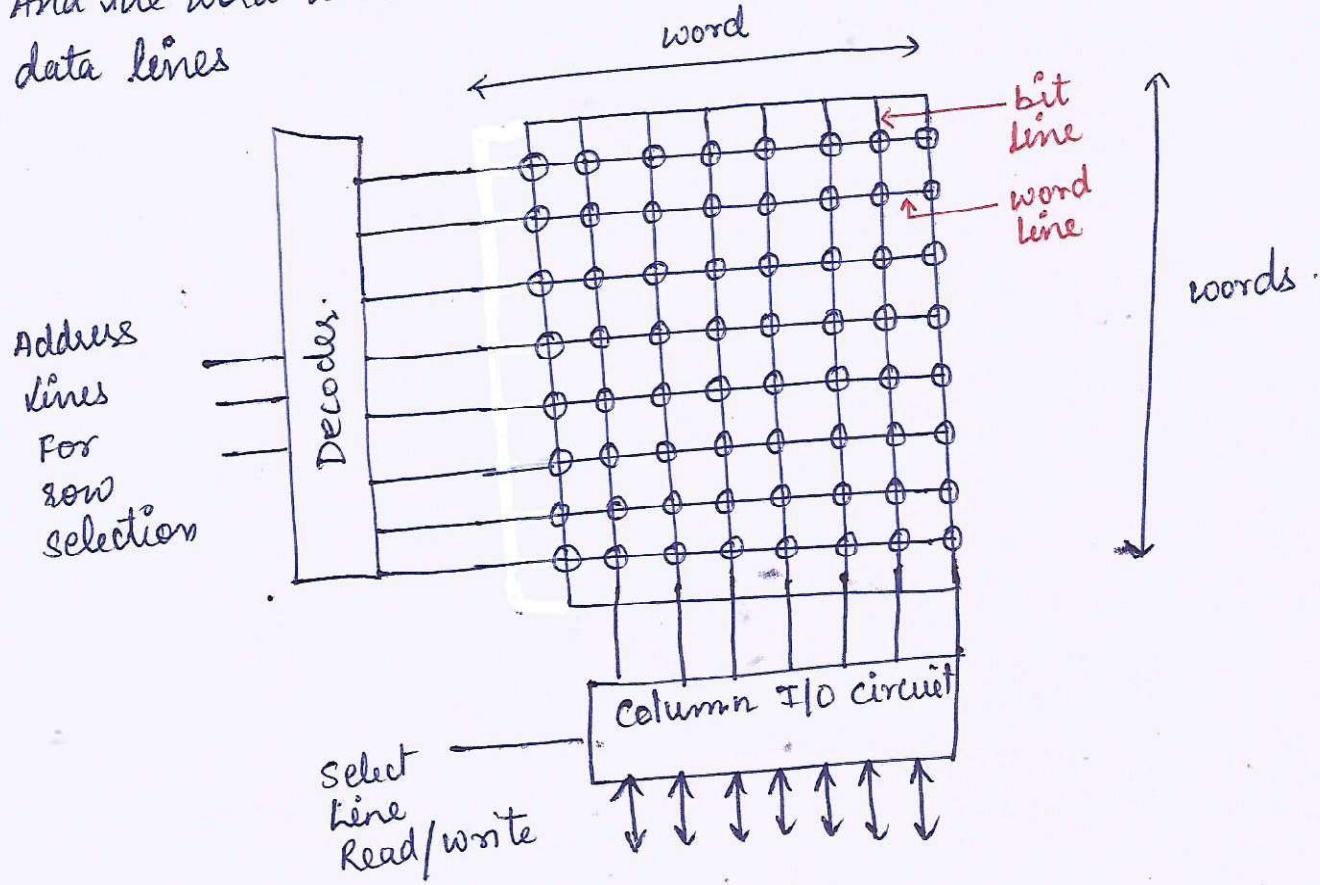
2 D Memory Organization

In 2D organization, memory is divided in the form of rows and columns. Each row contains a word. A decoder is used to select the row which address is ~~not~~ contained in MAR (Memory Address Register). The row it selected will be read or written through the

Select the row which address is stored in the Address Register). And the word which is selected will be read or written through the data lines



↑ ↑



memory size in this example \Rightarrow 8×8
 \downarrow each word size
8 number of words
~~word~~

2D Memory organization

2½ D Memory Organization:

(38)

- This organization, two decoders are used.
- One decoder to select the row (or words) and the column decoder selects the column.
- The address from MAR will go in ^{row} decoder inputs. The decoders will select the respective cell. Through the bit outline, the data from that location will be ~~be~~ read or through bit in line data will be written at that memory location.

