INDEX

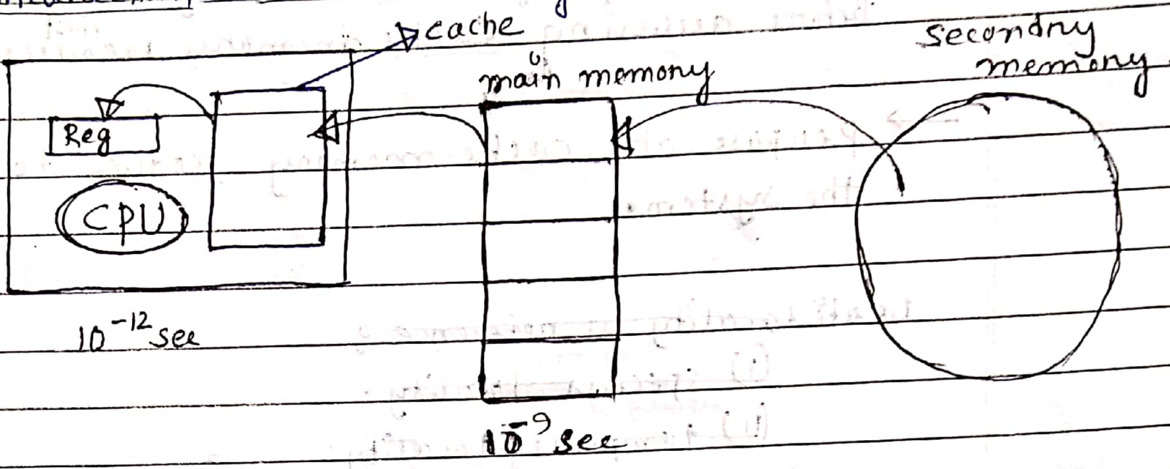
Name : Rakesh Nama. Subject : C.O.A

Class : Sec : Roll No. :

Sl. No.	Date	Title	Page No.	Teacher's Sign/Remarks
✓1.		Cache.		
✓2.		Memory Interfacing.		
✓3.		Secondary Memory.		
✓4.		Machine Instructions and Addressing modes.		
✓5.		ALU, Datapath and control Unit.		
✓6.		IO Interface.		
✓7.		pipelining.		

Cache memory

- Introduction to cache memory:



→ cache memory faster than main memory.

- ~~ch~~ cache hit: If required element present in cache, ~~then it~~ that is called 'cache hit'.

- hit latency: Time taken to findout whether ~~ere~~ element present on the cache or not, that is called "hit latency".

- cache miss: If required element not present in ~~cha~~ cache, that is called 'cache miss'.

- Miss latency: Time taken to get something from main memory and then place it into the cache and then read that's called "miss latency".

- page hit: If required element present in main memory

- Page fault: If required element not present in main memory.

- Tag directory: Tag directory says that required element present in tag or not.

→ Before ~~to~~ accessing main memory we first access page table,
Before accessing cache memory we first access Tags.

→ perpose of cache memory reduce to cost of the system.

~~Locali Locality of reference~~ :
~~(i) speciar locality.~~
~~(ii) temporal locality.~~

- Locality of reference :- is a term for the phenomenon in which the same values or related storage locations are frequently accessed, depending on the memory access pattern.

(Locality of reference)

temporal locality

Recently refferenced items are likely to be referenced in the near future.

spacial locality

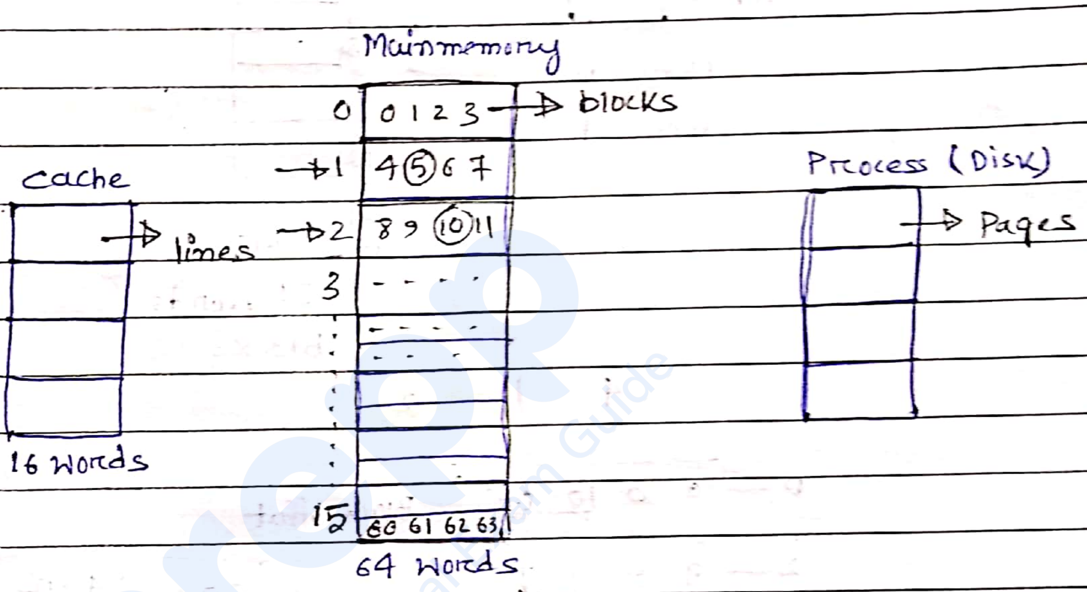
Items with nearby addresses tend to be referenced close together in time.

- ~~Introduction to Direct mapping~~ :

Introduction to Direct mapping :

→ talking about Disk and main memory He talking about pageing.
→ talking about cache and " " " " BlocKs.

→ Block size = lines size.



→ smallest addressable unit in the memory called word.
let's
1W = 1B (means ~~oute~~ system is byte addressable)

Block size = 4 words

No. of Block in main memory $= \frac{64}{4} =$ 16 Block.

No. of lines in cache $= \frac{16}{4} =$ 4 lines

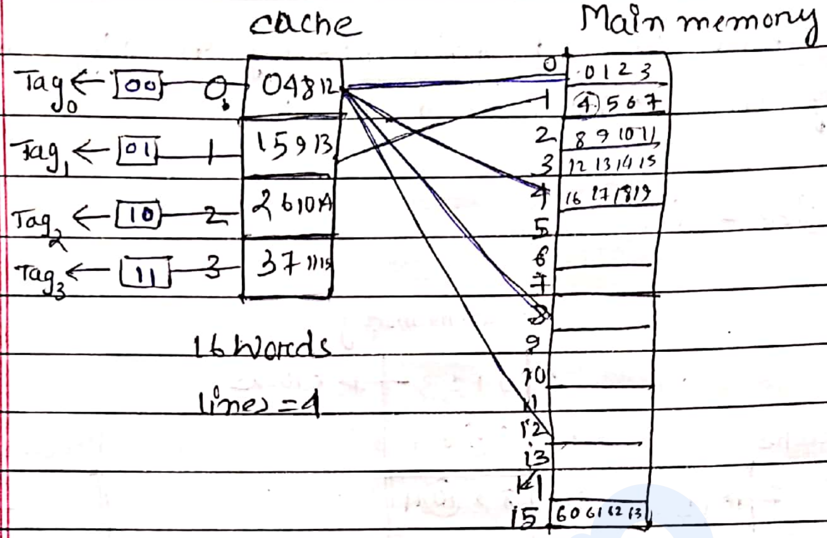
physical address contain = ~~0~~ 6 bits

$= 64$
$= 2^{6}$

~~o~~ processor generate address = $\underbrace{0001}_{\text{Blk num}}\underbrace{01}_{\text{Blk offset}}$ (to ~~50~~ word 5)

$= \underbrace{0010}_{\text{B.no}}\underbrace{10}_{\text{B off}}$ (to word 10)

• Direct Mapping :



16 Words

lines = 4

64 words

Block = 16

4 2

Block offset

0 —	0	0	0	0	✓
1 —	0	0	0	1	
2 —	0	0	1	0	
3 —	0	0	1	1	
4 —	0	1	0	0	✓
5 —	0	1	0	1	
6 —	0	1	1	0	
7 —	0	1	1	1	
8 —	1	0	0	0	✓
9 —	1	0	0	1	
10 —	1	0	1	0	
11 —	1	0	1	1	
12 —	1	1	0	0	✓
13 —	1	1	0	1	
14 —	1	1	1	0	
15 —	1	1	1	1	

00	–	0 4 8 12
01	–	1 5 9 13
10	–	2 6 10 14
11	–	3 7 11 15

0
1
2
3
4

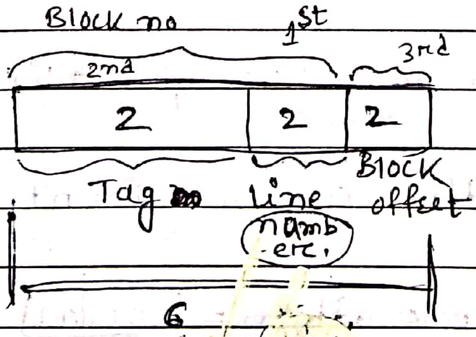
32 16 8 4 2 1
0 0 0 1 0 0

* 0 0 | 1 0 | 0 1 — 9
→ ~~Line~~ offT   ~~Tag~~ L.N   Blk off

address to word 3

~~0 0 | 0 1 | 0 0~~

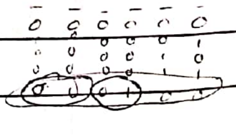
1 0 0 | 0 0 | 1 1
Tag no   line no   B offset



address to word 4

0 1 | 0 0 | 0 0

0 1 0 0 0 0



1Bit ~~[illegible]~~ Byte = 8 bit.
KB = 2^{10} B
MB = 2^{10} KB
GB = 10^{10} MB

Direct Mapping ~~Problem~~ - ①

[Problem - 1]

	MM size	cache size	Block size	Tag Bits	Tag Directory size
✓ Q1 —	128 KB	16KB	256B	✓3 bit	✓($3 * 2^6$) bit
✓ Q2 —	32 GB	32 KB	1KB	✓20bit	✓($20 * 2^5$) "
✓ Q3 —	✓ ~~[illegible]~~ 2^{26} B	512 KB	1KB	7	✓($7 * 2^9$) "
✓ Q4 —	16 GB	✓ ~~16KB~~ 2^{24} B	4KB	10	✓10 ~~[illegible]~~ $\times 2^{12}$
✓ Q5 —	64 MB	✓ 2^{16} B	can't guess	10	can't guess.
✓ Q6 —	2^{26} B	512KB	can't guess	7	can't guess

Assum~~s~~ing that memory is Byte addressable.

✓ **Q1:**

Main memory size = 128 KB
$= 2^{10} * 128$ B
$= 2^{17}$ B

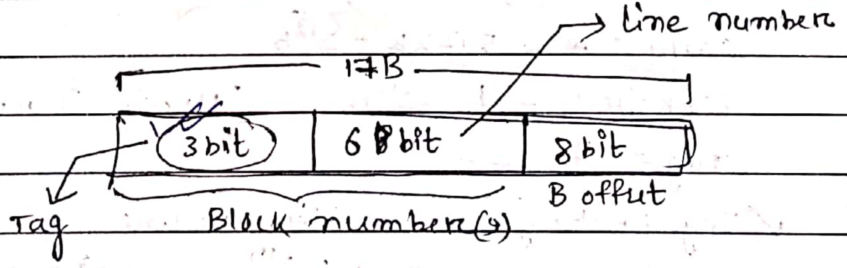
Block size = 256 B
$= 2^{⑧}$ B

no. of Block = $\frac{2^{17}}{2^8} = 2^{⑨}$ - bit

$\frac{2^{14}}{2^8}$

6

Cache size = 16 KB = 2^{14} B



no. of lines = $\frac{\text{C size}}{\text{line size}} = \frac{2^{14}}{2^8} = 2^{⑥}$ - bit

Line size = Block size

Tag Directory size = (Tag ~~#~~ size * no. of lines)
= ($3 * 2^6$) bit ✓✓

~~9~~

Q2:

MM size = 32 GB

$= ~~30~~\ 2^{10} * 32$ MB

$= 2^{10} * 2^{10} * 32$ KB

$= 2^{10} * 2^{10} * 2^{10} * 32$ B

$= 2^{30+5}$ B $= 2^{35}$ B

→ 35 bit

20 bit	5 bit	10 bit
Tag	line number	B offset

Block size = 1 KB

$= 2^{10}$ B (10 bit for Block ~~Bo~~ Offset)

Cache size = 32 KB

$= 2^{10} * 2^{5}$ B

$= 2^{15}$ B

no. of line = $\frac{\text{C size}}{\text{line size}}$

$= \frac{2^{15}}{2^{10}}$

$= 2^{5}$

Tag directory size = ~~10 * 2^10~~ Tag * no. of lines.

$= (20 * 2^{5})$

Q3:

26 B

Tag	Line no	B offset
7	19	10

Main memory size = ? (64)

Cache size = 512 KB = 2^{19} B

Block size = 1KB = 2^{10} B (10 bit B offset)

Tag = 7

✓ Tag directory size = 7 * ~~line~~ no of line

$= 7 * \frac{2^{19}}{2^{10}} = (7 * 2^{9})$ ~~B~~ bit

no. of line,

$\frac{2^{19}}{2^{10}} = 2^{9}$

~~no. of block = MM size / block size~~

address size = 7 + 19 + 10 = ~~36~~ 26

Main memory size = 2^{26} ~~Bit~~ B

$= ~~2^{32}\ B~~ = 2^{6}$ MB = 64 MB

~~$= 2^{30} \times 2^{2}$ B = 4 GB~~

Q4:

MM size = ~~164~~16 GB = $2^{30} * 2^{4}$ B = 2^{34} B

10	12	12
Tag	line num	Block offset = line offset

34 bit

$$\begin{array}{r} 34 \\ -22 \\ \hline 12 \end{array}$$

cache size = ?

Block size = 4KB = $2^{10} * 2^{2}$ B = 2^{12} B

tag = 10

Tag = Directory.

cache size = $2^{12} * 2^{12} = 2^{24}$ B

no of lines = $\frac{\text{c size}}{\text{line size}} = \frac{2^{24}}{2^{12}} = 2^{12}$

Tag directory = $10 * 2^{12}$

= $(10 * 2^{12})$ bit.

Q5:

MM size = 64 MB = $2^{20} * 2^{6}$ B = 2^{26} B (PA)

cache size = ?

Block size = ?

tag = 10

$$\begin{array}{r} 2^{26} \\ \hline 26 \end{array}$$

26 B

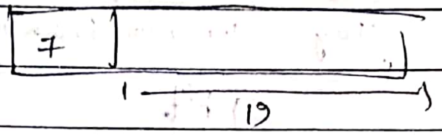
10		
Tag	16	

cache size = 2^{16} B

Q6.

Cache size = 512 KB = 2^{19} B

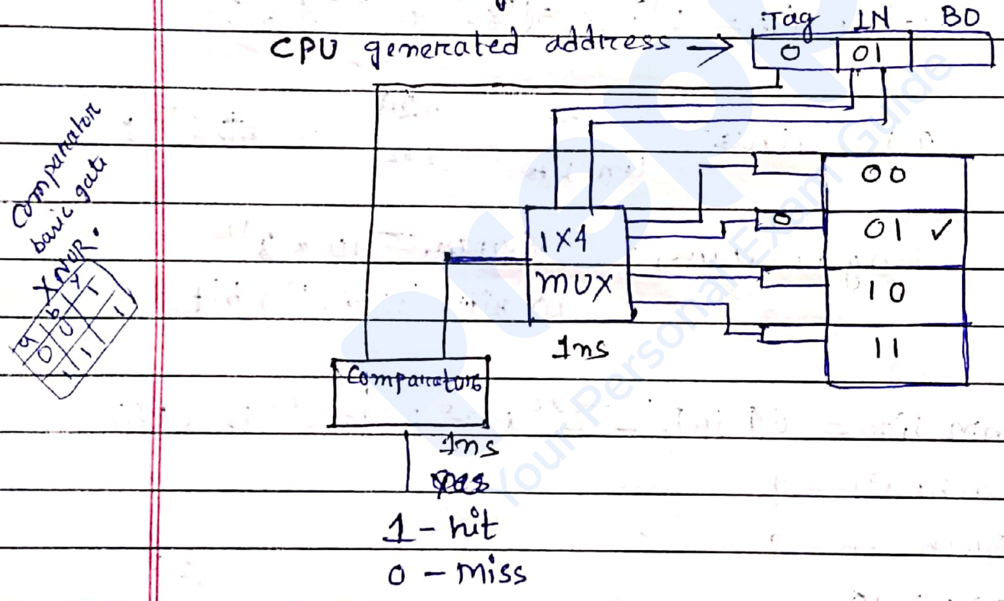
Tag = 7 bit



$19 + 7 = \boxed{26}$

✓ MM size = 2^{26} B

- **Direct Mapping HW Implementation** —



Total time taken = (latency of MUX + latency of comparator)

= 1ns + 1ns

= 2ns. (nanosecond)

→ no of MUX ~~and comparator~~ depends on ~~Tag~~ no. of Tag bits.

** Direct mapping →

Tag ~~bit~~ size = K bit

no. of MUX require = K

no. of Comparator = 1 (K bit compare)

↳ always (In Direct map)

Q1:

MM size = 1 GB

Cache ~~a~~ size = 1 MB

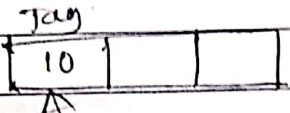
Comparator size = 10 Kns.

Hit latency = ?

here latency MUX negligible

⇒ ~~Hit Latency~~

~~MM = 2^{30} B~~



** Tag = $\frac{\text{Main M size}}{\text{Cache size}}$

$= \frac{1 \text{ GB}}{1 \text{ MB}} \Rightarrow$ ~~2^{10} KB~~ 1 KB ⇒ 2^{10} B

K=10

Hit Latency = ~~[illegible]~~ 10 Kns.

= 10 * 10 ns

= 100 ns.

→ ~~line number = $\frac{\text{Block number}}{\text{no. of lines}}$~~

line number = (B number) % (no. of lines)

K = m % n

• Disadvantage of direct mapping =

→ Conflict miss problem.

ex:

Block req by CPU – 5, 6, 4, 8, 9, 12, 15, 20

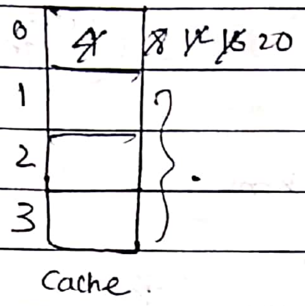
0	~~4~~	8/12 20
1	~~5~~	9
2	6	
3	15	→ place was ~~e~~ empty for long time, but

LN = (B.N) % (NL)

~~cache size~~ =
no. of lines = 4

line no. 0 used heavily ~~many time~~, this is conflict miss problem.

ex:



Block request by CPU

4 8 12 16 20 ~~04~~

$K = m \% n$ 　 n = ~~line~~ num of lines in cache

~~4/4=0~~ 4%4=0

~~8/4=0~~ 8%4=0

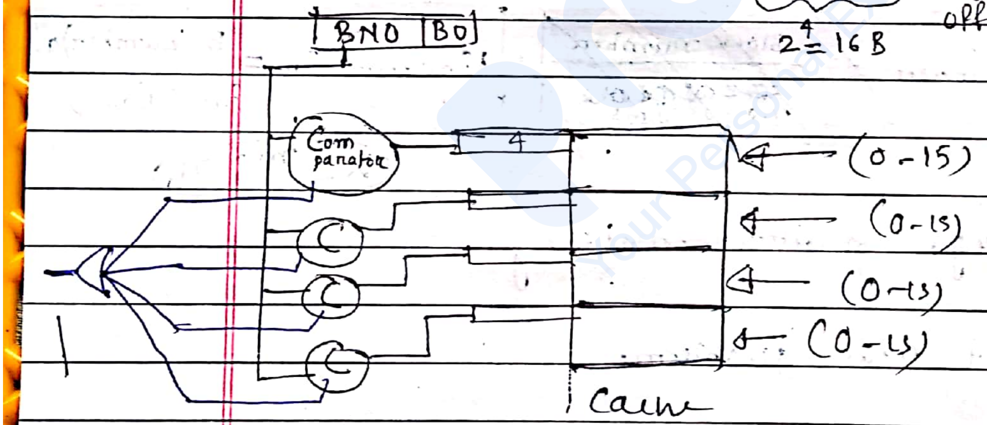
line no, 1, 2, 3, empty ~~but~~ then line 0 are heavily used it is conflict miss problem.

- Associative mapping = (Intro)

→ By Associative mapping can solve Conflict miss problem. ~~so~~

Phical address → $\underbrace{\overbrace{4}^{\to Tag}}_{B\cdot NO}$ | $\underbrace{2}_{\text{Block offset}}$

$2^4 = 16B$



1 - cache hit
0 - cache miss

→ 4 lines - 4 comparator need, so hardware cost Increase along with freedom.

→ *no of comparator requires = no of cache lines.

→ We get freedome to placed Block ~~an~~ any where in Cache, but requirement of Comparator Increase.

Q1:

Main memory size = 32 GB

Block size = 32 KB

✓ Tag(K) = ?

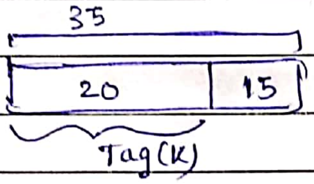
Propagation delay (PD) of Comp = 10 K ns

PD of OR Gate = 10 ns.

✓ Hit latency = ?

$$MM = 32\,GB = 2^{30+5}\,B = 2^{35}\,B$$

⇒



$$BS = 32\,KB = 2^{15}\,KB$$

✓ K = 20.

✓ hit latency = PD of comparator + PD of OR gate

= 10 K ns + 10 ns

= (10 * 20) ns + 10 ns

= 210 ns.

Q2:

	MM size	cache size	Block size	Tag size	Tag directory size
1 —	128 KB	16 KB	256 B	9 bits	(9×2^6) bits
2 —	32 GB	32 KB	1 KB	25 bits	(25×32) bits
3 —	128 MB	512 KB	1 KB	17	(17×2^9) bits
4 —	16 GB	? (can't)	4 KB	22 bits	(can't given)
5 —	64 MB.	? (can't) (dmy).	64 KB	10	(can't given)
6 —	not possi	512 KB	not possi	7	not possible.

① MM size = 128 KB

~~CS~~ Cache size = 16 KB

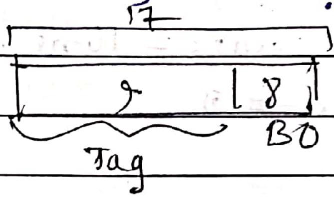
Block size(BS) = 256 B

Tag ~~size~~ size = ?

Tag directory size = ?

$$\text{no of lines} = \frac{CS}{LS(BS)} = \frac{2^{14}}{2^8} = 2^{\boxed{6}}$$

⇒



$BB = 128 = 2^7 * 2^{10}\,B = 2^{17}\,B$

$BS = 256\,B = 2^8\,B$

✓ Tag = 9

✓ Tag directory size = Tag size * no of lines.

$= (9 * 2^6)$ bits

$$\boxed{\text{no of com req} = 2^6 = 64}$$

②

$MM = 2^{35}\,B$

$CS = 2^{15}\,B$

$BS = 2^{10}\,B$

✓ Tag = $(35-10) = \boxed{25}$

✓ Tag directory = Tag * no of lines

$= \boxed{25 * 2^5}$

$$\text{no. of Line} = \frac{2^{15}}{2^{10}} = 2^5$$

③

MM = ?

$CS = 2^{19}\,B$

$BS = 2^{\boxed{10}}\,B$

Ts = 17

Tag directory = ?

✓ MM size = 2^{10+17}

$= 2^{27}\,B = 2^7$ MB

= 128 MB

$$\text{no. of line} = \frac{2^{19}}{2^{10}} = 2^9$$

✓ Tag D.S = $(17 * 2^9)$ bits.

(4) $MM = 2^{34} B$

~~CS~~ $CS = ?$ ✗

$BS = 2^{12} B$

✓ $TS = ~~10~~ ?$

$T.DS = ?$ ✗

Tag size = $(34 - 12)$

$= (22)$

(5) $MM = 2^{26} B$

$CS = ?$

✓ $BS = ?$ (2^{16})

$Tag = 10$

$Tag\ D.S = ?$

$BS = 2^{26-10}$

$= 2^{16} = 64 KB$

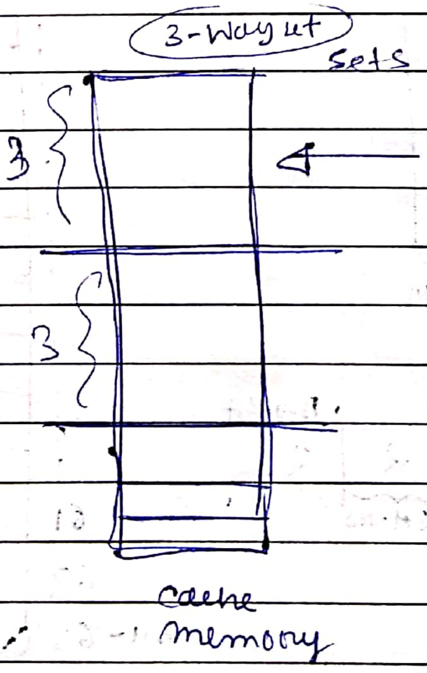
(6) $MM = ?$ ✗

$CS = 2^{19} B$

$TS = 7$

$BS = ?$ ✗

$T.DS = ?$ ✗ not possible.



- set Associative mapping == *(adv: no. of comparator reduced)

Ex: (how set associative work)

MM size = 64 B

cache s = 32 B

Block s = 4 B

✓ set size = 2 Blocks (lines)

OR

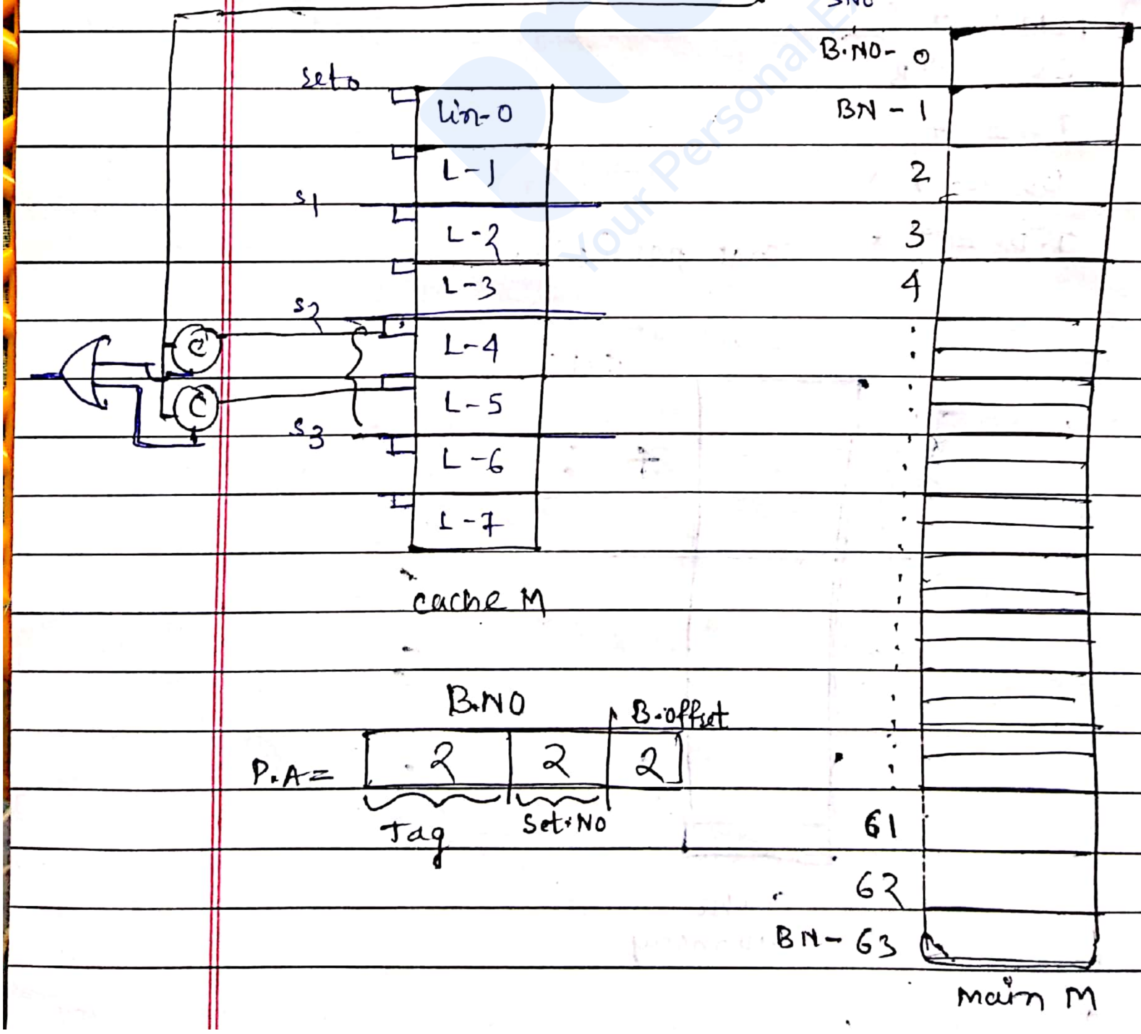
✓ (2-way set associative)

* K-Way set associative required K-Comparators

$\text{lines} = \frac{CS}{Bs} = \frac{32}{4} = 8 \text{ lines}$

$\text{sets} = \frac{\text{Lines}}{\text{sets}} = \frac{8}{2} = 4 \text{ sets}$

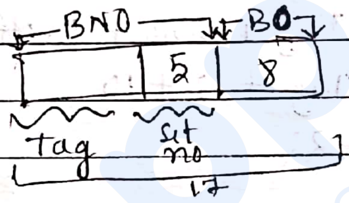
give address (P.A) = | 01 | 10 | 11 |



Q1:

	MM size	cache size	Block size	Tag bits	Tag directory size	set associative
1	128 KB	16 KB	256 B	4	$(4 * 2^6)$ bits	2-way set
2	32 GB	32 KB	1KB	22	$(22 * 2^5)$ bits	4
3	2^3 MB	512 KB	1 KB	7	$(7 * 2^9)$ bit	8
4	16 GB	64 MB	4 KB	10	$(10 * 2^{14})$ bit	4
5	64 MB	~~2~~256 KB	X	10	X	4
6	~~8~~8 MB	512 KB	X	7	X	8

1



$MM\,S = 2^{17} B$ | PA = 17 bit.

$CS = 2^{14} B$

$BS = 2^{8} B$

$$\text{no. of lines} = \frac{CS}{BS} = 2^6 \text{ lines.}$$

$$\text{no. of sets} = \frac{\text{no. of lines}}{\text{set size}}$$

$$= \frac{2^6}{2} = 2^5 = (32 \text{ sets})$$

PA = Tag + set no + B.offset

$\Rightarrow 17 = Tag + 5 + 8$

$\Rightarrow Tag = 4$

Tag directory = Tagsize * no. of lines.

$= (4 * 2^6)$ bits.

(2)

MM = 32GB = 2^{35}B

CS = 32KB = 2^{15}B

BS = 1KB = 2^{10}B

T = ? (22)

T.D = ?

Set-ass = 4-way

$$\text{no. of line} = \frac{CS}{BS} = \frac{2^{15}}{2^{10}} = 2^5$$

$$\text{no. of set} = \frac{\text{~~tir~~ no. of line}}{\text{set size}} = \frac{2^5}{2^2} = 2^3$$

→ P.A = Tag + S.no + B.off

→ 35 = Tag + 3 + 10

→ Tag = 22

T.D = $(22 * 2^5)$

(3) MM = ?

CS = 512KB = 2^{19}B

BS = 1KB = 2^{10}B

T = 7

T.D = ?

set-ass = 8-way

~~sez~~ 2^3.

7	6	10
Tag	Set no	B.O

= ~~200 bit 24~~ 23 bit.

& ~~MM~~ =

$$\text{no. of line} = \frac{2^{19}}{2^{10}} = 2^9$$

$$\text{no. of sets} = \frac{\text{~~ces~~ no. of line}}{\text{size of sets}} = \frac{2^9}{2^3} = 2^6$$

MM size = 2^{23} = 8MB

T.D = T * no. of lines.

= $(7 * 2^9)$ bits.

④ MM = 16 GB = 2^{34} B

CS = ?

BS = 4 KB = 2^{12} B

T = 10

T.D = ?

set ass = 4-way set.

T.D = ~~of~~ T * no. of lines
= $10 * 2^{14}$
= $\boxed{160 \text{KB}}$

C.S = no. of lines * line size
= $2^{14} * 2^{12}$
= 2^{26} B
= 2^{6} MB
= ~~256~~ $\boxed{64 \text{MB}}$

34

10	12	12
Tag	Set no	B.of

no. of sets = 2^{12}

no. of sets = $\dfrac{\text{no. of lines}}{\text{a set size}}$

no. of lines = set size * no. of sets
= $4 * 2^{12}$
= 2^{14}

⑤ MM = 64 MB = 2^{26} B

CS = - (2^{18} B)

BS = -

Tag = 10

tag.D = -

set. ass = 4 way

26

10	?	?
Tag	set NO	B.offset

26

10	x	y
	16	

~~⑥ MM = 4~~

~~CS = 512 KB~~

~~BS = -~~

~~T = 4~~

~~T.D = -~~

~~set A = 8 way.~~

~~4~~

~~Tag~~

Cache size = no. set * ~~size of each set~~ lines per set * line size (Bs)
= $2^{x} * 2^{2} * 2^{y}$
= 4.2^{x+y}
= $4 \cdot 2^{16}$
= $2^{18} = 2^{8}$ KB = 256 K

Q3:

MM Size	Cache size	Block size	Tag size	Tag directory	Set associative
64MB	—	—	10	—	4-way
—	512KB	—	7	—	8-way

⑥ MM = ? — (2^{23} B)

CS = . 512KB = 2^{19} B

BS = —

TS = 7

TDS = —

Set ass = 8-way

TAG	set	Block
7	x	y

cache size = no. of sets * line's per set * Block size

$2^{19} = 2^{x} * 2^{3} * 2^{y}$

=> $2^{16} = 2^{x+y}$

=> $x + y = 16$

P.A = 7 + (x + y)

= 7 + 16 = $\boxed{23}$

MM = 2^{23} = 8 MB

• Comparing all the mappings :-

MM size = 4GB $= 2^{32}$ B

Cache = 4MB $= 2^{22}$ B

Block size = 1KB $= 2^{10}$ B

P.A = 32 Bits.

no. of Lines $= \frac{CS}{BS} = \frac{2^{22}}{2^{10}} = \boxed{2^{12} \text{ lines}}$

no. of set $= \frac{\text{no. line \#}}{\text{size of set}} = \frac{2^{12}}{2^{2}}$
$= 2^{10}$

Direct mapping :-

B.NO		B.O
10	12	10
Tag	Line no	

32 bits

associative Mapping :-

32 bits

22	10
B.NO (Tag)	B.O Offset

4-way set asso :-

32 bits

B.NO		
12	10	10
Tag	set NO	BO

~~Relationship [illegible]~~

	TAG	Comparator required	Size of ~~comparator~~ (Tag directory)	Lines	Size of comparator
Direct	10	1	$10 * 2^{12}$	2^{12}	10
Associative	22	2^{12}	$22 * 2^{12}$	2^{12}	22
4-way	12	4	$12 * 2^{12}$	2^{12}	12

✓ Questions - gate

g-1995 Q1

Cache size = 4 K words = 2^{12} words

Block size = 64 words. = 2^{6} words

set size = 4 blocks.

The number of bits in "$\underset{(4)}{\text{set}}$" and '$\underset{(6)}{\text{word}}$" field of MM address are:-

→

	4	6
	set no	B.O or (word)

no of sets = $\frac{\text{no of lines}}{\text{set size}}$ no. of lines = $\frac{2^{12}}{2^{6}} = 2^{6}$

$= \frac{2^{6}}{2^{2}} = 2^{4}$ (sets)

Q2

4-way set associative.

cache lines = 128

lines size = 64 words

P.A. = 20 bits

Tag, set and word fields are ?

→ $\overbrace{\qquad\qquad\qquad}^{20}$

9	5	6
Tag	set no	BO or (word)

no.of lines = $\frac{\cancel{\text{cs}}\ \text{cache size.}}{\text{lines size.}}$

cache size = $2^{6} * 2^{7} = 2^{13}$ words.

sets = $\frac{\text{lines}}{\text{set size}} \Rightarrow \frac{2^{7}}{2^{2}} \Rightarrow 2^{5}$

Tag = 9, set = 5, words (BO) = 6.

(Q3) Blocks in cache = 128
4-way set associative.
MM contains 2^{14} blocks.
Block size is 256 eight bit words.

(i) How many bits are required for addressing MM ? (22)

(ii) How many bits are needed to represent the TAh, set and word fields.
(9) (3) (8)

→ cache lines $= 2^7$
set size $= 4$
MM Blocks $= 2^{14}$
Block ~~size~~ size $= 2^8$ ~~bit~~ word.

$$\underbrace{\boxed{9} \boxed{5} \boxed{8}}_{22}$$

sets $= \frac{2^7}{2^2} = 2^5$

~~no of lines~~

MM size $= 2^{14} * 2^8$ words
$= 2^{22}$ words

(Q4)

Direct mapped cache
cache size $= 32\,KB = 2^{15}\,B$
Block size $= 32\,B = 2^5\,B$
PA = 32 bits

The number of bits needed for cache indexing and tag bits are respectively.

→ $$\overbrace{\underset{\text{tag}}{\boxed{17}}\ \underset{\text{line.no}}{\boxed{10}}\ \underset{\text{B.O}}{\boxed{5}}}^{32}$$

no.of line. ~~n~~ $= \frac{2^{15}}{2^5} = 2^{10}$

cache indexing = 10 bit
Tag bit = 17.
Ans-(10, 17)

P-227

g-2006 (Q5)

1-20-1-21

Considere three cache organizations

① first one

set associative

cache size = 32 KB = 2^{15} B

2-way set associative

Block size = 32 B = 2^5 B

② 2nd one

different

cache size same

Direct mapped

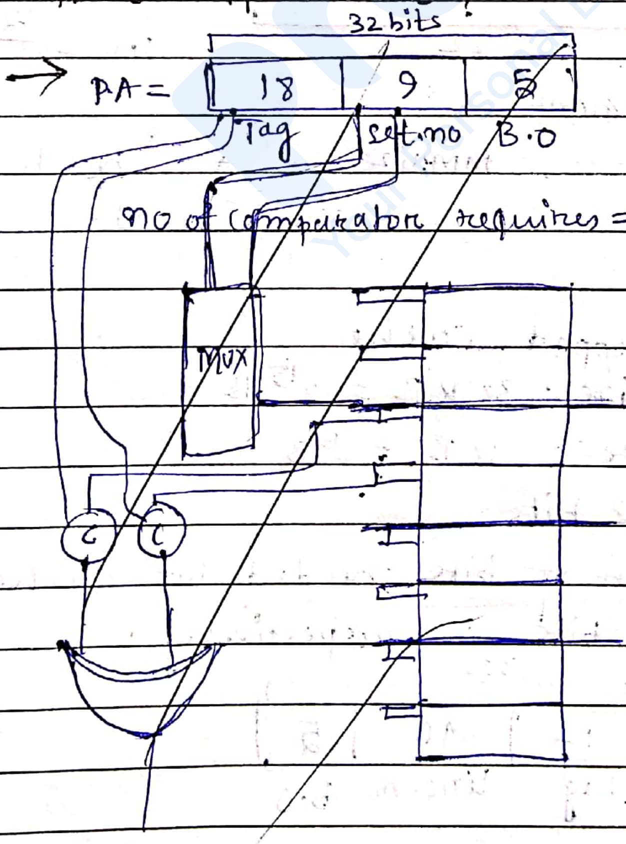
P.A = 32 bits

2-to-1 MUX latency = 0.6 ns.

K bit comparator has latency = K/10 ns.

The hit latency of set associative organization is h_1 and " " Direct mapped is h_2.

find h_1 and h_2?



no of comparator requires = 2^1

line.no = $\frac{2^{15}}{2^5} = 2^{10}$

no. of set = $\frac{2^{10}}{2^1} = 2^9$

32
14
18

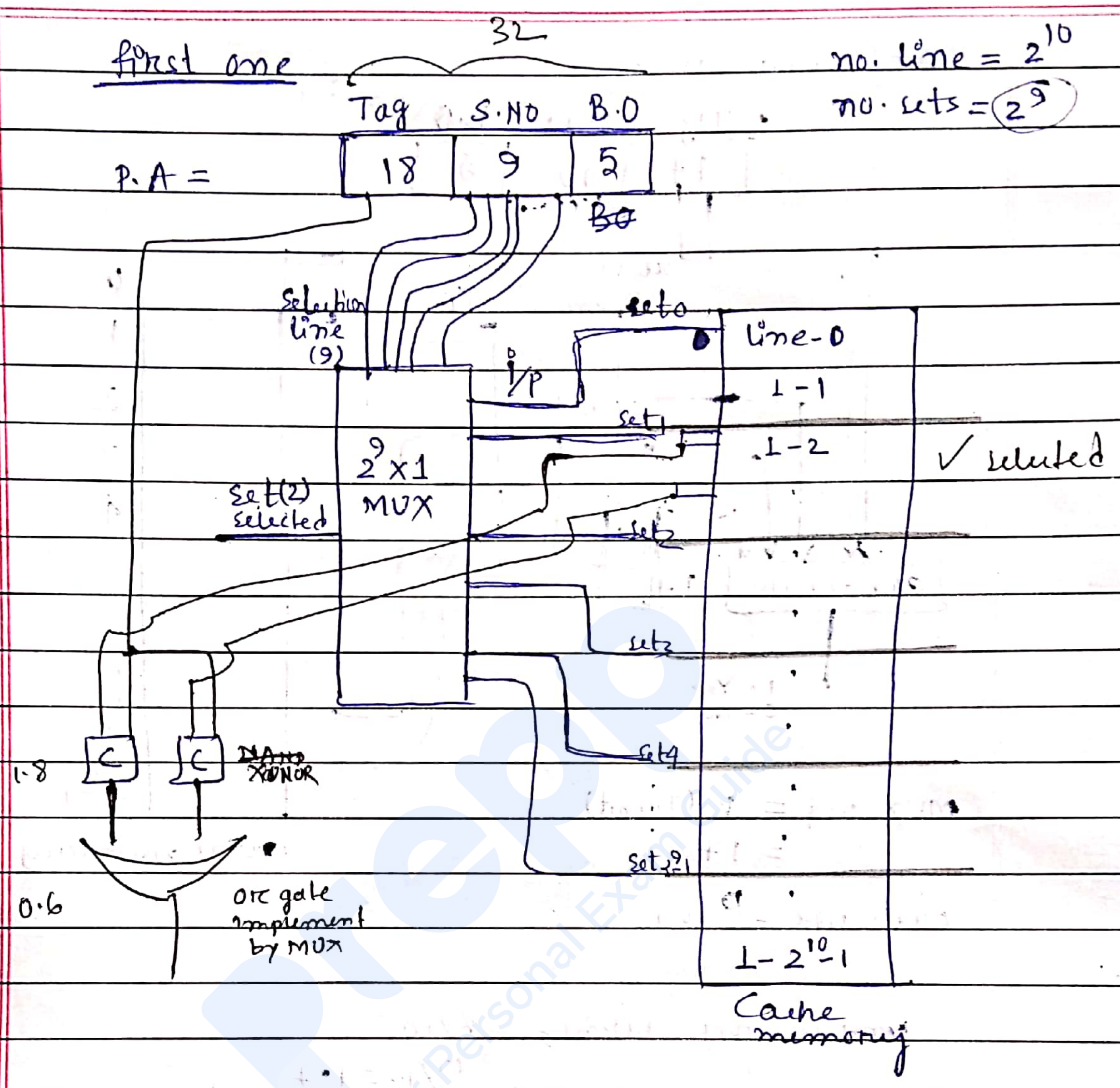
here, MUX required
= 2 X Tag bits
= K X Tag bits

hit latency =

first one

no. line = 2^{10}

no. sets = 2^9



MUX req = Tag bit * 2

K-way set associative ~~⊕~~ (MUX req) = T * K.

size of each MUX = 2^S to 1

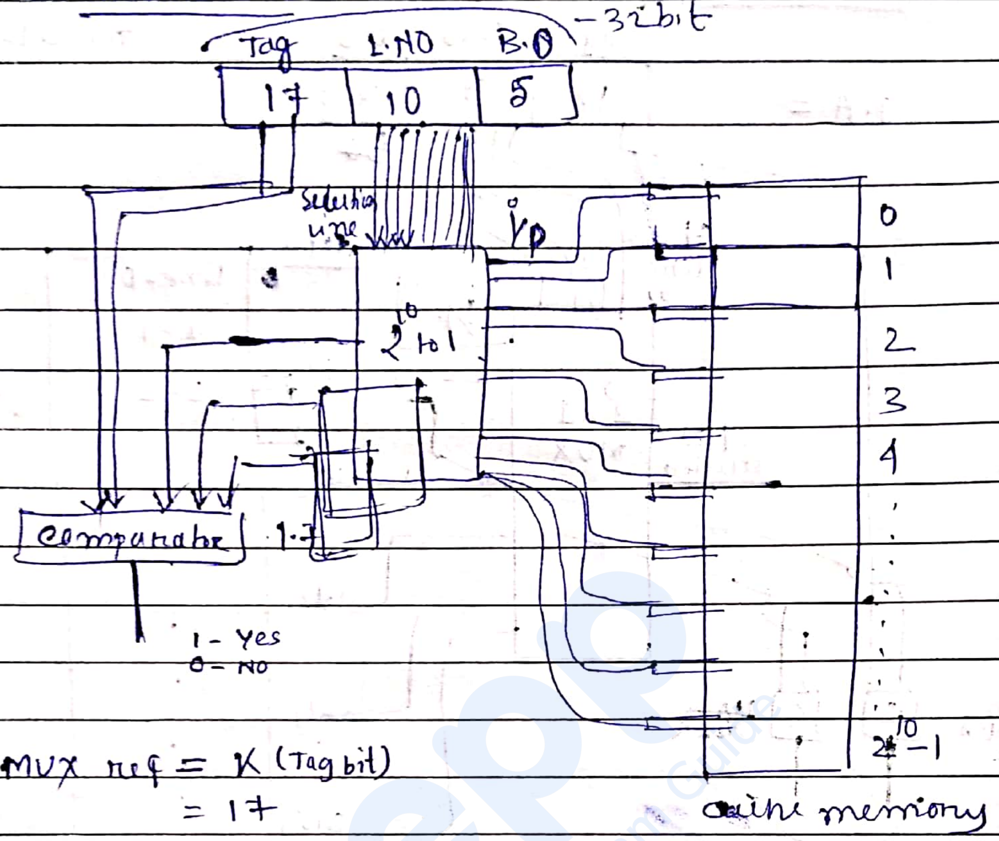
Comparator need = K

comparator latency = $K/10$ = $18/10$ = 1.8 ns.

h_1 = 1.8 + 0.6

= 2.4 ns

~~A~~ Second one-



MUX ref = K (Tag bit)
= 17

each size = 2^{10} to 1

comparator latency = K/10
= 17/10 = 1.7

~~Total~~ $h_2 = 1.7\,ns$ ✓

9-2011

Q6 Direct mapping
cache size = 8KB
BS = 32 byte.
PA = 32 bites.

The cache controller maintains tag information for each cache block compriising of following.

1 valid bit, 1 modified bit and ~~as as as~~ as many bits as the minimum needed to identify ~~block~~ ~~mapped in~~ the memory block mapped to the cache

What is the total size of memory needed at the cache controller of store meta-data (tags) for the cache?

→ lines = $\frac{CS}{BS} = \frac{2^{13}}{2^5} = 2^8$

19	8	5

(19+1+1) = 21

T. Directory = $(21 * 2^8)$ bits
= $21 * 256$
= 5376 bih,

P-229, 1.40, 1.41
g-2012

Q7 cache size = 256 KB = 2^{18} B
set size = 4.
B.S = 32 B = 2^5 B
P.A = 32 bits

16	11	5
Tag	set	BO

no of lines = $\frac{~~BS~~ CS}{BS}$
= 2^{13}

sets = $\frac{~~8~~ \text{lines}}{2^2}$
= $\frac{2^{13}}{2^2} = 2^{\boxed{11}}$

(1) The number of bits in the tag field of an address — (16).

(2) The size of the cache tag directory is – lines * Tag

$2^{13} * (16+2+1+1)$
= $2^{13} * 20$
= $2^3 * 20$ KB
= 160 KB.

= $2^{13} * 16$
= $2^{13} * ~~8~~ 4$
= 2^{17}
= 2^7 KB
= 128 KB

9-2019 Q8

4-way set associativity

Cache size = 16KB = 2^{14} B

Block size = 8 Words = (8 * 32) bits = $\frac{2^8}{2^3}$ Bs = 2^5 B

Word size = 32 bits

PAS = 4GB = 2^{32} B

Tag bits = ?

→

32

20	7	5
Tag	Set no	B.O

Tag bits = 20

lines = $\frac{CS}{BS}$

= 2^9

Sets = $\frac{\text{lines}}{\text{set size}}$ = $\frac{2^9}{2^2}$

= 2^7

Memory Interfacing

www.gatenotes.in

Computer Architetutre :

It deals with Instructions, addressing modes, ALU, pipelining etc (Internal Design)

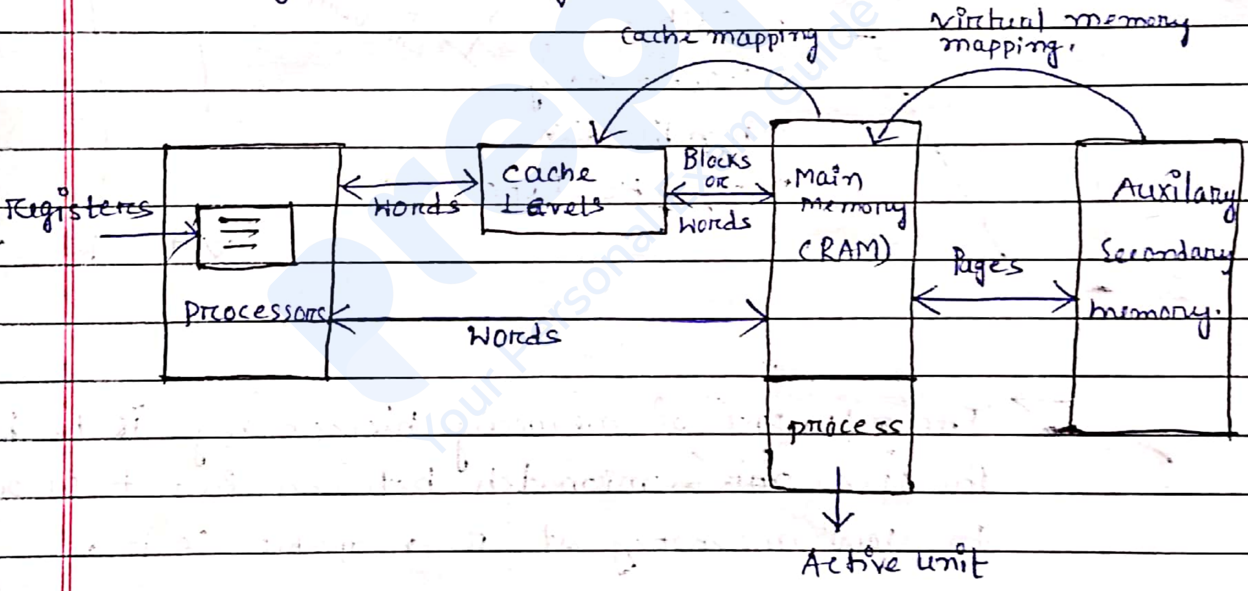
Computer Organization :

It deals with how various memory and I/O interact with a system.

Computer design :

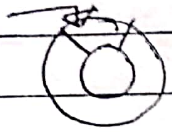
It deals with hardware design.

- Memory Interfacing –

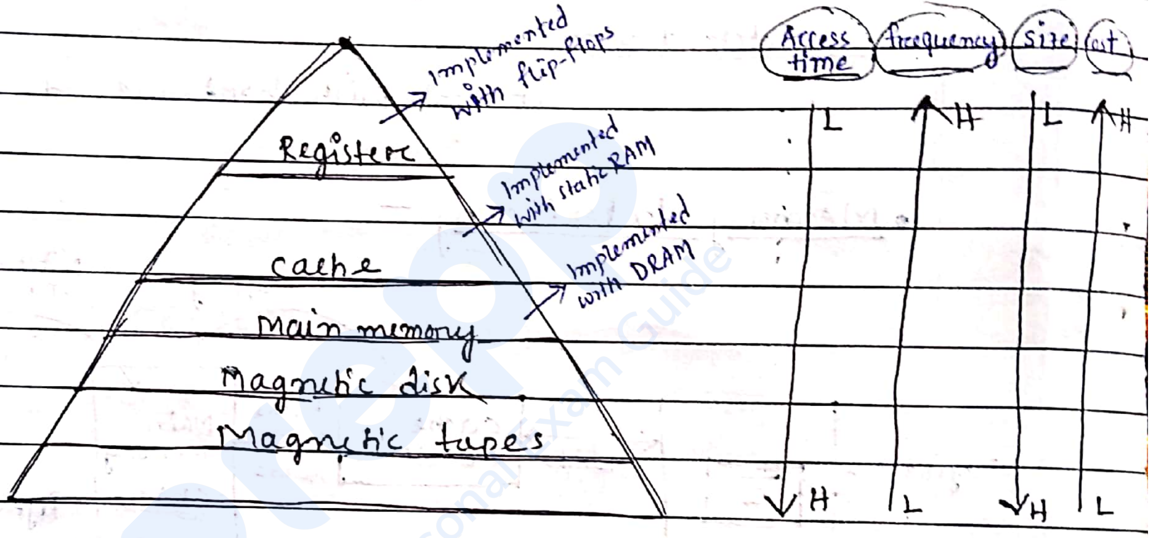


- **Memory Hierarchy :**

Cache levels }
Main memory } Random Access.

Magnetic disk → semi Random Access. 

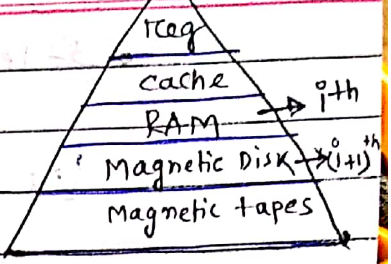
Magnetic tapes → sequential Access.



→ The ~~p~~ purpose of memory hierarchy is to bridge the speed ~~mis~~ ~~s~~ mismatch between fastest processor to slow memory at reasonable cost.

→ The goal of memory hierarchy is to minimize average access time of entire memory system.

- **Level memory—**



• 2 level memory –

Information in $i^{th} \subset (i+1)^{th}$ level

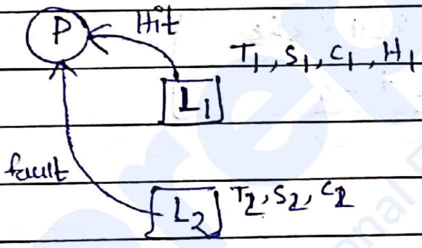
→ If processor refers to ith level memory is found then " Hit" otherwise "Miss (or) fault".

→ There are two way in which the processor is connected to various levels of memory

Case 1 :

$T_1 < T_2$

~~Cases~~



$$\text{Hit rate} = \frac{x}{100}$$

$$\text{Miss rate} = \left(1 - \frac{x}{100}\right)$$

T_1 → Time to access

S_1 → size of level 1 memory.

C_1 → cost per bit.

H_1 → Hit rate.

✓✓ $$\boxed{T_{avg} = H_1T_1 + (1-H_1)T_2.}$$

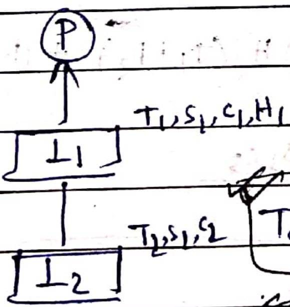
✓ $$\boxed{Cost_{avg} = \frac{C_1S_1 + C_2S_2}{S_1+S_2}}$$

100 → x

$$T_{avg} = \frac{xT_1 + (100-x)T_2}{100}$$

$$= H_1T_1 + (1-H_1)T_2$$

Case 2 :



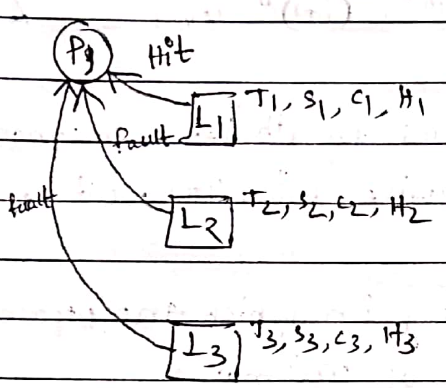
✓ $$\boxed{T_{avg} = H_1T_1 + (1-H_1)(T_1+T_2)}$$

✓ $$\boxed{Cost_{avg} = \frac{S_1C_1 + S_2C_2}{S_1+S_2}}$$

3-level memory –

case-1

$T_1 < T_2 < T_3$



~~Best case time~~

Best case time

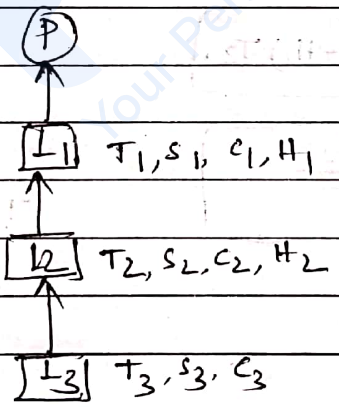
Worst time taken

$\boxed{T_1 \leq T_{avg} \leq T_3}$

$$\boxed{T_{avg} = H_1T_1 + (1-H_1)H_2T_2 + (1-H_1)(1-H_2)T_3}$$

$$\boxed{C_{avg} = \frac{C_1S_1 + C_2S_2 + C_3S_3}{S_1 + S_2 + S_3}}$$

Case-2



Best time taken

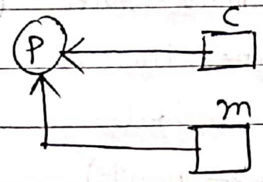
Worst time taken

$\boxed{T_1 < T_{avg} < (T_1 + T_2 + T_3)}$

$$\boxed{\begin{array}{l} T_{avg} = H_1T_1 + (1-H_1)H_2(T_1+T_2) + (1-H_1)(1-H_2)(T_1+T_2+T_3) \\ C_{avg} = \dfrac{s_1c_1 + s_2c_2 + s_3c_3}{s_1+s_2+s_3} \end{array}}$$

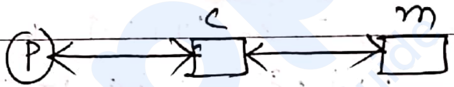
Q: The average memory access time for a machine with a cache hit rate of 80% where the cache access time is 5ns and memory access time is 100ns is ?

→ ~~Tavg~~ (1)



$$T_{avg} = H_1 T_c + (1-H_1) T_m$$
$$= (0.8)(5ns) + (1-0.8)(100ns)$$
$$= \boxed{24\ ns}$$

(2)



$$T_{avg} = H_1 T_c + (1-H_1)\ (T_c + T_m)$$
$$= (0.8)(5ns) + (1-0.8)\ (5ns + 100ns)$$
$$= 4ns + (0.2)\ (105ns)$$
$$= 4ns + 21.0ns.$$
$$= \boxed{25\ ns}.$$

- Cache replacement policy :-

↳ Replacement policy is required for associative mapping and set associative mapping but not for direct mapping.

↳ Replacement policies ~~and~~ are aimed to minimize miss penality for ~~references~~ future references.

Replacement Policies

- Random (No specific criteria to replace the block)
- FIFO (The block which entered first is the candidate replacement)
- LRU (The block which is not refferred for longest time which is replaced)
- LFU (The block ~~eto~~ which fewer references is replaced)

Q:1 Consider a direct mapped cache with 8 cache blocks (0-7), if the memory block requests are in the order 3, 5, 2, 8, 0, 6, 3, 9, 16, 20, 17, 25, 18, 30, 24, 2, 63, 5, 82, 17, 24 which one of the memory blocks will ~~be~~ not be in the cache at the end of the sequence ?

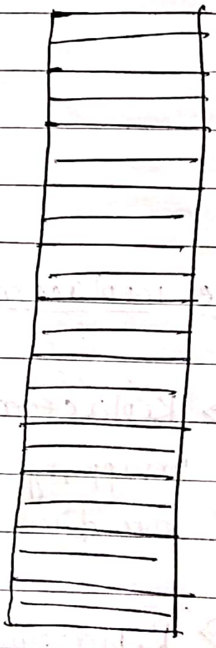
(a) 3 [✓ (b) 18] (c) 20 (d) 30.

⇒

0	~~8~~ ~~0~~ ~~16~~	24
1	~~9~~ ~~17~~	~~25~~ 17
2	~~2~~ ~~18~~	~~2~~ 82
3	3 ✓	
4	20 ✓	
5	5	
6	~~6~~ 30	✓
7	63	

no. of Block (Line) = 8 ~~[illegible]~~

cache



main memory

$\boxed{i = j \bmod 8}$

i → ~~block~~ Line no. of cache.

j → block no. of MM.

number of Lines on cache (8).

$\boxed{Q}$: 2 Conciders a 4-Way set ~~ass~~ouiative mapping with 16 cache blocks, the more the memory block request are in the order (0, 255, 1, 4, 3, 8, 133, 159, 216, 219, 63, 8, 48, 32, 73, 92, 155) which one of the following memory block, will ~~[illegible]~~ be present in the cache if LRU is used.

✓ a) 3 ~~b)~~ 8 ✓ c) 129 ✓ d) 216.

⇒ (LRU — least recently used)

S_0	~~0~~, ~~4~~, ~~8~~, ~~216~~, ~~48, 32, 92~~ 8, 48, 32, 92
S_1	1, 133, 73,
S_2	
S_3	(~~255~~, ~~3~~, ~~159~~), 219, ~~[illegible]~~ 63, 155

cache memory (4-way set associative) (means ~~one~~ each set contain 4 lines)

$\boxed{\text{Set-number} = (i) \bmod 4}$

i → Block number.

$\boxed{Q}$: 3 consider a small 2-way set associative mapping with a total of 4 blocks, for choosing the block to be replace use LRU scheme, the number of cache misses for the following sequence of block addresses 8, 12, 0, 12, 8 is 4 ?
M M M H M

⇒

S_0	~~8~~, ~~12~~, ~~0~~, 12, 8
S_1	

cache

$\boxed{\text{s.no} = (i) \bmod 2}$

✓ miss rate $= \left(\frac{4}{5} \times 100\right) = 80\%$

✓ Hit rate $= \left(\frac{1}{5} \times 100\right) = 20\%$

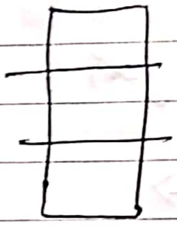
$\boxed{Q}$: 4 Consider a 2-way set associative mapping consisting of 2^c memory blocks and 2c cache blocks, the cache Blocation for the memory block 'K' is ____

a) K mod 2c

b) K mod 2^c

✓ c) K mod c

d) K mod K



no. of lines in cache = 2C

2-way set-associative.

no. of set = $\frac{2c}{2} = c$

~~set no = K mod~~

Set no = K mod c

$\boxed{Q}$: 5

Consider the cache has 4 blocks, for the memory reference (5, 12, 13, 17, 4, 12, 13, 17, 2, 13, 19, 13, 43, 61, 19) what is the hit ratio for the following cache ~~re~~ replacement algorithms –

(i) FIFO (ii) LRU (iii) Direct mapping.

(iv) 2-way set Associate (LRU):

→ (i) $\boxed{\text{FIFO}}$. $(\overset{m}{5}, \overset{m}{12}, \overset{m}{13}, \overset{m}{17}, \overset{m}{4}, \overset{✓}{12}, \overset{✓}{13}, \overset{✓}{17}, \overset{m}{2}, \overset{✓}{13}, \overset{m}{19}, \overset{m}{13}, \overset{m}{43}, \overset{m}{61}, \overset{✓}{19})$

~~5~~	~~4~~ 43
~~12~~	~~2~~ 61
~~13~~	~~13~~ ~~[illegible]~~ 19
~~17~~	13

hit ratio $= \left(\frac{5}{15} \times 100\right)$

miss ratio $= \left(\frac{10}{15} \times 100\right)$

→ (II) LRU. $(\overset{m}{5}, \overset{m}{12}, \overset{m}{13}, \overset{m}{17}, \overset{m}{4}, \overset{✓}{12}, \overset{✓}{13}, \overset{✓}{17}, \overset{m}{2}, \overset{✓}{13}, \overset{m}{19}, \overset{✓}{13}, \overset{m}{43}, \overset{m}{61}, \overset{✓}{19})$

$(\cancel{5}, \cancel{12}, \cancel{13}, \cancel{17}, \cancel{4}, \cancel{12}, \cancel{13}, \cancel{17}, \cancel{2}, \cancel{13}, \cancel{19}, 13, 43, 61, 19)$

hit ratio $= (\frac{6}{15} \times 100)$ miss ratio $= (\frac{9}{15} \times 100)$

(III)

→ Direct mapping $(\overset{m}{5}, \overset{m}{12}, \overset{m}{13}, \overset{m}{17}, \overset{m}{4}, \overset{m}{12}, \overset{m}{13}, \overset{m}{17}, \overset{m}{2}, \overset{m}{13}, \overset{m}{19}, \overset{✓}{13}, \overset{m}{43}, \overset{m}{61}, \overset{m}{19})$

0	$\cancel{12}$	$\cancel{4}$ $\cancel{12}$ 2
1	$\cancel{5}$	$\cancel{13}$ $\cancel{17}$ $\cancel{13}$ $\cancel{17}$ $\cancel{13}$ 61
2		
3	$\cancel{19}$	$\cancel{43}$ 19

4)61(15
4
21
20
1

line.no = (B.NO) mode 4

cache hit ratio $= (\frac{1}{15} \times 100)$

miss ratio $= (\frac{14}{15} \times 100)$

→ (IV) 2-Way set Associate (LRU) –

$(\underset{m}{5}, \underset{m}{12}, \underset{m}{13}, \underset{m}{17}, \underset{m}{4}, \underset{✓}{12}, \underset{✓}{13}, \underset{✓}{17}, \underset{m}{2}, \underset{✓}{13}, \underset{m}{19}, \underset{✓}{13}, \underset{m}{43}, \underset{m}{61}, \underset{m}{19})$

S0	$\cancel{12}, \cancel{4}$	, 12, 2
S1	$\cancel{5}, \cancel{13}$	, $\cancel{17}$, $\cancel{13}$, $\cancel{17}$, $\cancel{13}$, $\cancel{19}$, $\cancel{13}$, $\cancel{43}$, 61, 19

line.no = (i mod 2)

i → Block no.

hit ratio $= (\frac{5}{15} \times 100)$.

miss ratio $= (\frac{10}{15} \times 100)$.

$\boxed{Q}$: 6

A hierarcal memory system has the following speification 20MB main storage with access time of 300ns, 256 B cache with access time of 50ns. Word size 4B, page size 8 words. What will be the hit ratio if the page address trace of a program has the pattern $\underset{m}{0}, \underset{m}{1}, \underset{m}{2}, \underset{m}{3}, \underset{\checkmark}{0}, \underset{\checkmark}{1}, \underset{\checkmark}{3}, \underset{\checkmark}{0}, \underset{\checkmark}{1}2, \underset{m}{4}$ following LRU page replacement technique.

→

Cache size = 256 B

Word size = 4B

page size = 8 Words = (8×4) B = 32 B

$$\text{No. of cache page} = \frac{\text{size of cache}}{\text{cache page size}} = \frac{2^8}{2^5} = 2^3$$

$(\cancel{0}, \cancel{1}, \cancel{2}, \cancel{3}, \cancel{0}, \cancel{1}, 3, 0, 1, 2, 4)$

hit ratio $= (\frac{3}{8} \times 100)$

$\boxed{Q}$: 7

Considers an array A[100] and each element Occupies 4 words, A 32-word cache is used and divided into 8-word blocks.

a) What is the hit ratio for the statement.

for (i = 0, i < 100, i++)

A[i] = A[i] + 10.

→ cache size = 32 W

Block size = 8 W

no. of Block in cache = $\frac{32}{8}$ = 4 Block.

B_0	A_0	A_1
B_1	A_2	A_3
B_2	A_4	A_5
B_3	A_6	A_7

cache.

$\underset{m}{\underline{A_0}}, \underset{H}{\underline{A_0}}, \underset{H}{\underline{A_1}}, \underset{H}{\underline{A_1}}$

$\underset{m}{\underline{A_2}}, \underset{H}{\underline{A_2}}, \underset{H}{\underline{A_3}}, \underset{H}{\underline{A_3}}$

hit ratio $= \left(\frac{3}{4} \times 100\right)$ ✓ $= 75\%$ hit rate.

$\boxed{Q}$: 8

Consider an array has 100 elements and each elements occupies 4 words. A 32 bit word cache is used and divided into a blocks of 8 words. What is the hit rate of.

```
~~ff~~ for ( i = 0 ; i<10 ; i++)
        for ( j=0 ; j<10 ; j++ )

                A[i][j] = A[i][j] +10.
```

→

Cache size = 32 Word
Block size = 8 W
no. of Block = 4.

i = 0 10 . . . .
i = 1 . . . .
j = 2 . . . .

array = $[A_0, A_1, A_2, \cdots \cdots A_{100}]$

B_0	A_0	A_4
B_1		
B_2		
B_3		

cache.

$$\begin{bmatrix} 00 & 01 & 02 & \cdots & 09 \\ 10 & 11 & 12 & \cdots & 19 \\ 20 & & & & \\ 30 & & & & \\ \vdots & & & & \\ 90 & 91 & 92 & \cdots & 99 \end{bmatrix}$$

RMO

$\boxed{00}\ \boxed{01}\ \boxed{02}\ \boxed{03}\ \boxed{04}\ \boxed{05}$

CMO

$\boxed{00}\ \boxed{10}\ \boxed{20}\ \boxed{30}$

RMO :

00	01

m	H	H	H
00	00	01	01
R	W	R	W

TM = (50) 1

TH = (50) ~~4~~3

TRefference = (50) 4

Hit ratio = $\left(\frac{3}{4} \times 100\right)$.
= 75 %

(column major order)

CMO :

m	H	m	H
00	00	01	01

TM = (50) 2

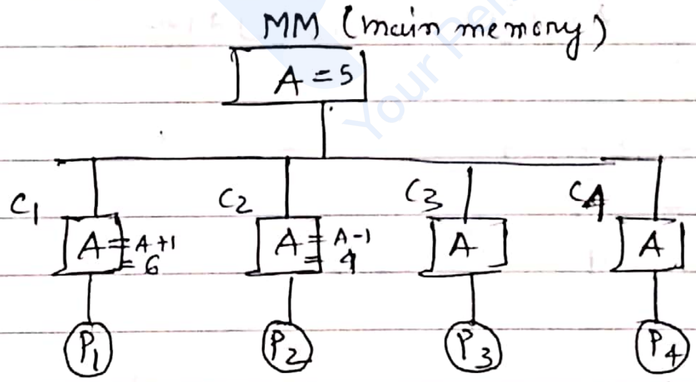
TH = (50) 2

TR = (50) 4

Hit ratio = $\left(\frac{2}{4} \times 100\right)$
= 50 %

- Cache Coherence problem :

→ cache coherence problem : Multiple copies of same data can exist indifferent caches simultaneously, and if processors are allow to Update their own copies freely an ~~ma~~ insoncistent view of memory can result.

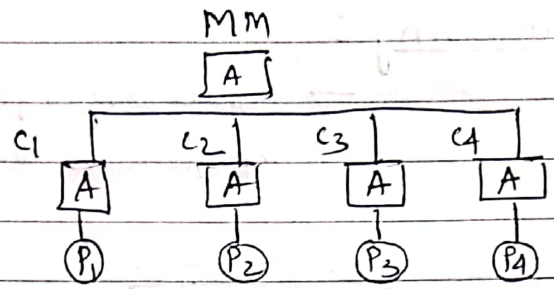


C1 → cache
P1 → processor

- Methods to avoid cache coherence problem -

→ There are 4 - method to avoid cache coherence problem

1) write Update - write through.
2) write Update - write back.
3) write invalidate - write through.
4) write Invalidate - Write back.



1) Write update - write through: update simultaniously of a word in cache & M memory

2) Write update - ~~Back~~ write Back : The updation of MM is postponed until the associated block is replaced.

~~3) Write Invalidate - write through: ~~Changing a value in~~ Updaty simultaniously of a value in ~~one cache,~~ the value will be invalid ~~ate to other cache.~~ MM. ~~[illegible]~~ and other cach can't used this value.~~

~~4) write Invalidate - write Back :~~

3) write invalidate - write through: if processor P1 ~~[illegible]~~used a value and change, then ~~other other Processor P2 can't w~~ the value will be Invalied for other processor and update simultaniously of value In ~~[illegible]~~ MM.

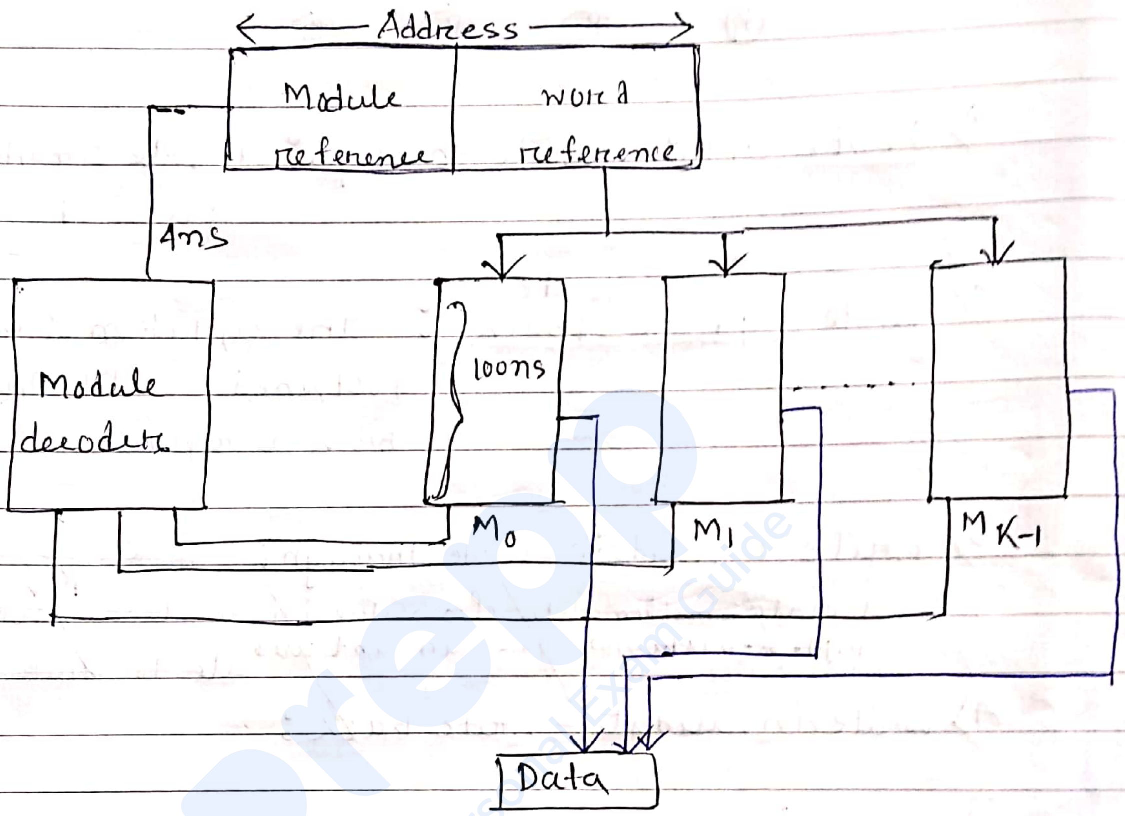
4) Write Invalidate - write Back :

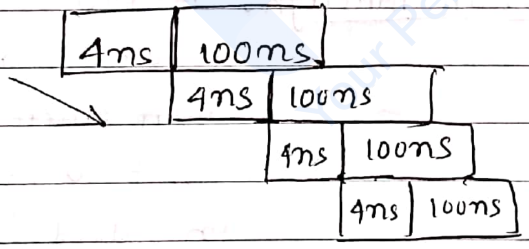
if P1 ~~and~~ change a value then this value will be Invalid to other processor and updation of MM is postponed until the associated block is replaced.

- **Memory interleaving :**

→ "Reduce average access time".

→ "Improve data transfer rate".





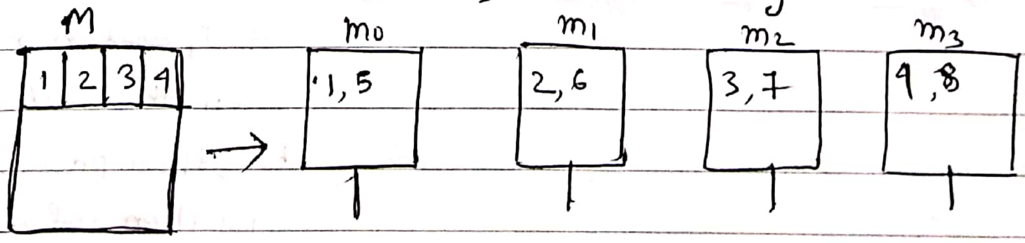
If read 'K' words then, total time taken,- T = K(4ns)+100ns.

AND

Without Interleaving concept total time, T = K(100ns).

~~K(4ns)+100ns~~

Memory Interleaving -



TO DOWNLOAD THE COMPLETE PDF

CLICK ON THE LINK GIVEN BELOW

WWW.GATENOES.IN

GATE CSE NOTES