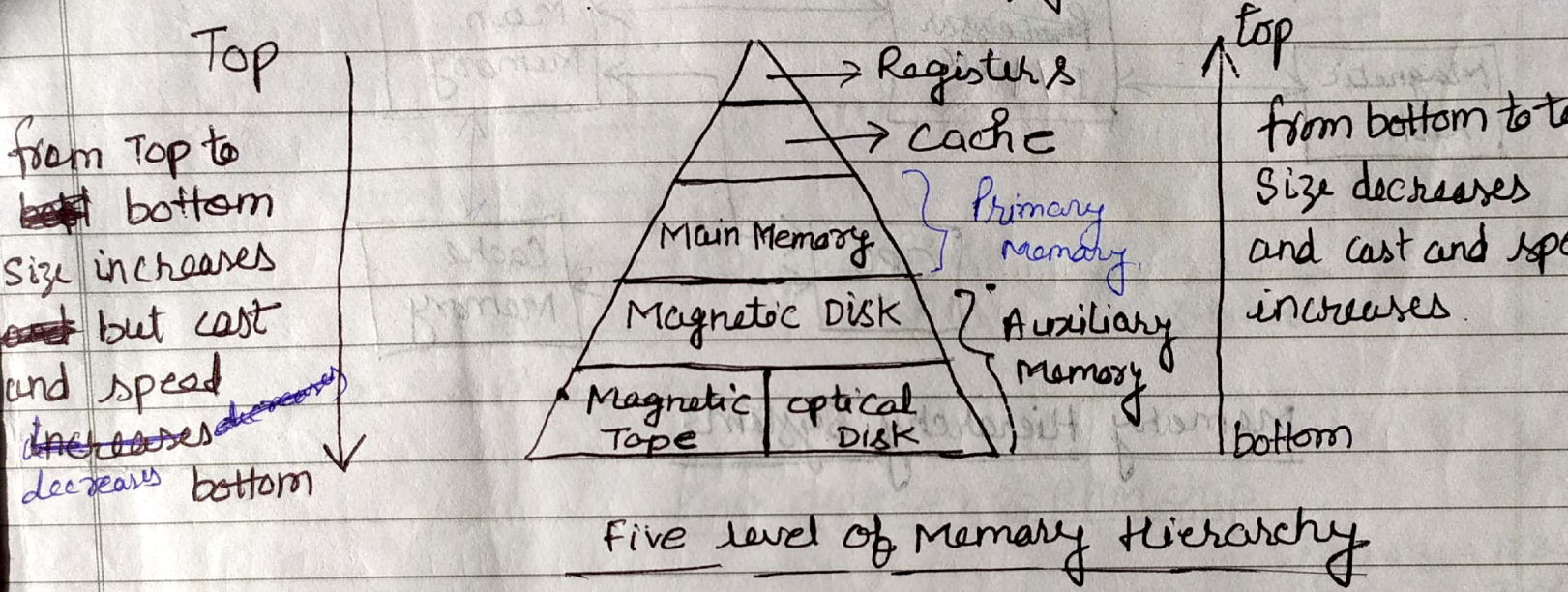
UNIT-04 Memory Organization

Memory Hierarchy :- A five-level memory hierarchy is shown in fig below.



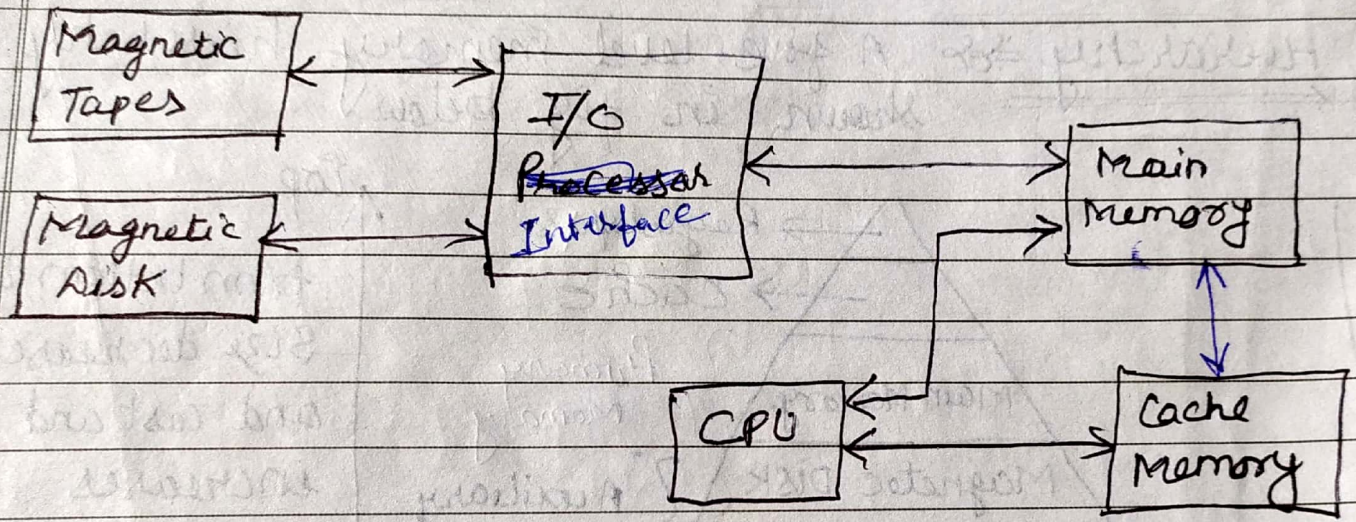
Five level of Memory Hierarchy

⇒ At the top of this memory hierarchy, there are CPU registers which is access at full CPU speed. These are local memory to CPU.

⇒ Next comes cache memory. which is slower than registers but faster than main memory. it has less cost than register but higher than main memory

⇒ Next comes main memory, which is slower than cache memory but faster than magnetic Disk. also it's cost is less than cache but higher than magnetic Disk.

⇒ Next comes magnetic Disk which is slower than main memory but faster than magnetic tape and optical disk.



Memory Hierarchy System

Main Memory

(Volatile) → RAM (Random Access memory)

Non volatile → ROM (Read only memory)

SRAM (Static RAM)

1. Called static because Data does not change till power is ON
2. It uses flip-flop as memory cell.
3. Size of one memory cell is larger
4. Flip-flop stores '0' and '1'
5. Memory density is low So access time is less
6. Faster and costlier
7. used in cache memory

DRAM (Dynamic RAM)

1. Called dynamic RAM because data changes in the presence of power also
2. It uses capacitor as memory cell
3. Size of one memory cell is smaller.
4. Presence of electric charges on capacitor shows '1' and absence shows '0'.
5. memory density is high charge decay on capacitor the data '1' becomes '0' after electric charge decay below threshold level.
6. Slower and cheaper
7. used in main memory.

PROM

1. Programmable Read only memory
2. one time programmable only
3. Data once written can not be changed

EPROM

1. Erasable read only memory
2. Data once written can be changed (erased by UV rays)
3. If error complete chip needs to be erased

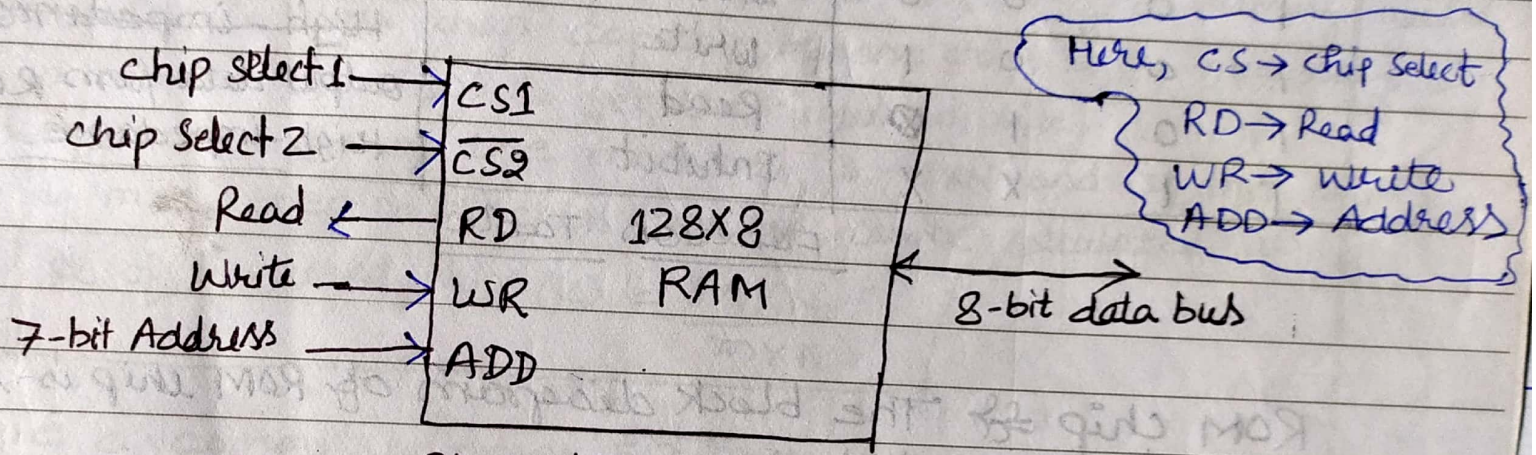
EEPROM

1. Electrically EPROM
2. Data once written can be erased by using electric signal
3. Byte/Block level erasing can be done so whole chip need not erased

Flash Memory

1. Electrically EPROM
2. Data once ~~can be~~ written ~~can be~~ erased using electric signal
3. Bit level erasing can be done no need to erase the whole Byte/block

RAM chips → fig below shows the block diagram of a RAM chip. The capacity of RAM is 128 words of eight bits per word. This requires a 7-bit address and 8-bit bidirectional data bus.



Block diagram of RAM chip

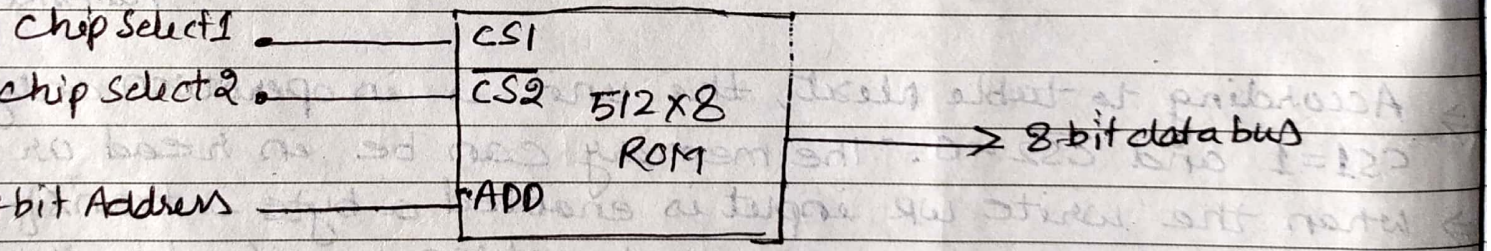
∴ Ram size = $128 = 2^7$
= 7-bit Address

⇒ According to table Next, the unit is in operation only when CS1=1 and $\overline{CS2}=0$. The memory can be in read or write mode.
⇒ When the write WR input is enabled a byte is transferred from the data bus to the memory location specified by the address lines.
⇒ When the read RD input is enabled, a byte from the specified memory location by address line is placed into data bus.
⇒ In other chip selection memory function is inhibit and state of data bus is in high impedance.

CS1	$\overline{CS2}$	RD	WR	Memory Function	State of data bus
0	0	X	X	Inhibit	High-impedance
0	1	X	X	Inhibit	High-impedance
1	0	0	0	Inhibit	High-impedance
1	0	0	1	Write	High-impedance Input data to RAM
1	0	1	⊗	Read	output data from RAM
1	1	X	X	Inhibit	High-impedance.

Function Table

ROM chip ⇒ The block diagram of ROM chip is shown below:



⇒ The two chip select line must be CS1=1 and $\overline{CS2}$=0 for the unit to be operational, otherwise, the data bus is in high-Impedance.

⇒ There is no need for read and write input control because the unit can only read. Thus, when the chip is select the bytes selected by the address line appears to be in data-bus.

Memory Address Map

⇒ The interconnection between memory and processor is established from the knowledge of size of memory required and type of RAM and ROM chip is available.

⇒ RAM and ROM chips are available in different sizes. If a memory needed for the computer is larger than capacity of one chip, it is necessary to combine a number of chips to get the required size.

⇒ If the required size of the memory is MXN and if the chip capacity is mxn, then no of chips required can be calculated as

$$\text{No. of required chips } (k) = \frac{M \times N}{m \times n}$$

Ex ⇒ Suppose a computer needs 512 bytes of RAM and 512 bytes of ROM. The capacity of RAM chip is 128X8 and that of ROM is 512X8. Hence, the number of RAM chips required

$$K = \frac{M \times N}{m \times n} = \frac{512 \times 8}{128 \times 8} = 4 \text{ RAM chips}$$

required, ROM chips $= \frac{M \times N}{m \times n} = \frac{512 \times 8}{512 \times 8} = 1$ ROM chip

⇒ The memory address map for the system is shown in table next page. The table consist of three columns. The First column specifies whether a RAM or a ROM chip is used. Next column shows a range of Hexadecimal addresses for each chip. The third column show Address bus.

⇒ The Table shows only 10 address lines although the address bus consists of 16 lines since $512 = 2^9$, 9 lines of RAM and 8 lines for ROM. These six lines are assumed to be zero.

Component	Hexadecimal Address	Address Bus									
		10	9	8	7	6	5	4	3	2	1
RAM 1	0000 - 007F (0 - 127)	0	0	0	x	x	x	x	x	x	x
RAM 2	0080 - 00FF (128 - 256)	0	0	1	x	x	x	x	x	x	x
RAM 3	0100 - 017F (257 - 383)	0	1	0	x	x	x	x	x	x	x
RAM 4	0180 - 01FF (384 - 511)	0	1	1	x	x	x	x	x	x	x
ROM	0200 - 03FF (512 - 1023)	1	x	x	x	x	x	x	x	x	x

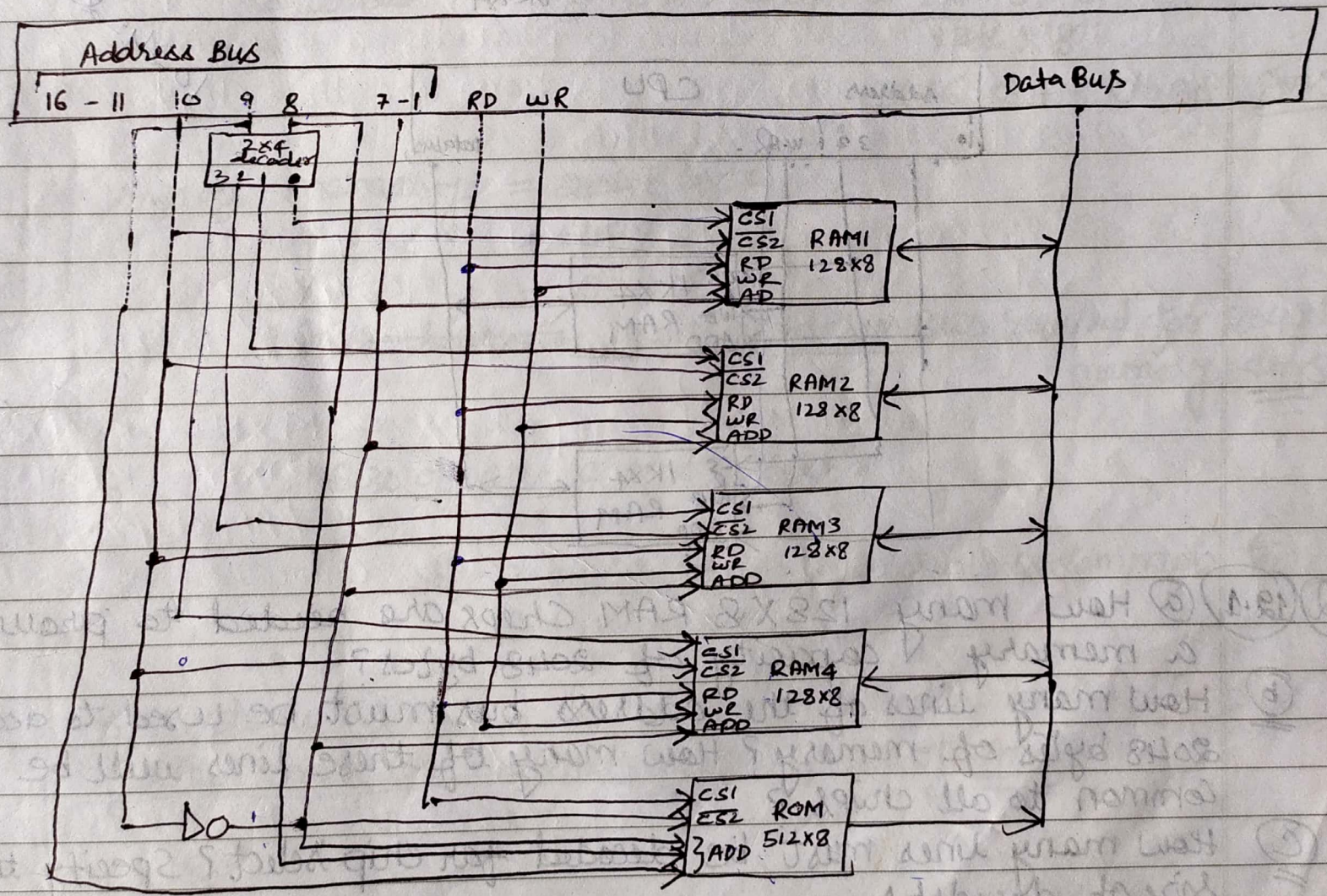
⇒ The RAM chips have 128 bytes and need seven address lines which are common to all four RAM chips. The ROM chip has 512 byte and need nine address lines. Thus, X's are assigned to the low-order bus lines, line 1 to 7 for RAM chips and lines 1 through 9 for the ROM chip where these X's represents a binary number which is a combination of all possible 0's and 1's value.

⇒ Lines 8 and 9 are used to differentiate between the four RAM chips. If line 8 and 9 is 00 then RAM1 is selected, 01 it is RAM2, 10 it is RAM3 11 it is RAM4.

⇒ To differentiate between RAM and ROM chips line 10 is used if line 10 is 0 CPU select RAM if it is 1 CPU select ROM chip.

Memory connection to CPU

⇒ The CPU connect the RAM and ROM chips through the data and address buses. The memory chip connection to CPU is shown below in fig



Memory connection to CPU

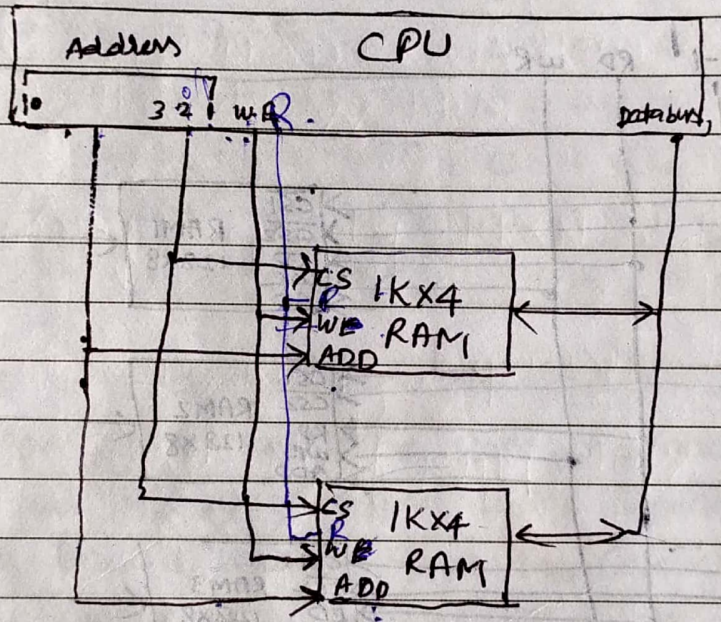
Q. 1KX4 RAM chips are used to construct 1KX8 RAM. How many chips are required? Draw the connection diagram?

1K = 2^{10}

Soln

$$\text{No. of chips} = \frac{M \times N}{m \times n} = \frac{1K \times 8}{1K \times 4} = 2 \text{ chips}$$

$10 + 1 = 11$



Q (12.1) (a) How many 128X8 RAM chips are needed to provide a memory capacity of 2048 bytes?

(b) How many lines of the address bus must be used to access 2048 bytes of memory? How many of these lines will be common to all chips?

(c) How many lines must be decoded for chip select? Specify the size of decoders.

Ans (a) ~~memory~~ Memory capacity M×N = 2048 bytes
= 2048×8

Size of one RAM chip m×n = 128×8

Required No. of RAM chips = $\frac{2048 \times 8}{128 \times 8}$

= 16 RAM chips Ans

(b) size of memory = 2048 bytes
= 2048 × 8
= $2^{11} \times 8$

So, ~~11 lines to~~ 11-bits of Address Bus required for 2048 bytes memory Ans

∴ Size of one chip = 128×8
= $2^{7} \times 8$

So, 7-bits of Address bus will be common to all RAM chips Ans

(c) Since, there 16 chips, so Size of decoder = 4×16 Ans

4 lines must be decoded for chip select Ans

Q (12.2) A computer uses RAM chips of 1024 X 1 capacity.

(a) How many chips are needed, and how should their Address lines be connected to provide a memory capacity of 1024 bytes?

(b) How many chips are needed to provide a memory capacity of 16K bytes? Explain in words how are the chips are to be connected to the address bus.

Sol^n

(a) Memory capacity M x N = 1024 bytes
= 1024 x 8

Size of one RAM chip m x n = ~~1024 x 8~~ 1024 X 1

Required No of RAM chips = $\frac{M \times N}{m \times n} = \frac{1024 \times 8}{1024 \times 1}$

= 8 RAM chips. Ans

8 chips are needed with address lines connected in parallel.

(b) Memory capacity M x N = 16K bytes
= 16 X 1024 X 8

Size of one RAM chip m x n = 1024 X 1

Required No of RAM chips = $\frac{16 \times 1024 \times 8}{1024 \times 1}$

= 128 RAM chips Ans

⇒ ∴ size of Memory = 16 K bytes
= 16 x 1024 x 8
= $2^4 \times 2^{10} \times 8$ = $2^{14} \times 8$

= 14-bits Address bus | one chip size = 1024×1 = 2^{10} ⇒ 10-bits Address for chip

where 10-bits specify chip address
4-bits lines are decoded into 16 chip-select inputs

2 (12.5) Book Moris Mano

Soln (a) Size of ~~RAM~~ one chip ~~=~~ m×n = 256×8
Memory Capacity M×N = 2K bytes
= 2×1024×8

Required No. of RAM chips = $\frac{M \times N}{m \times n} = \frac{2 \times 1024 \times 8}{256 \times 8} = 8$ RAM chips Ans

Size of one ROM chip ⇒ m×n = 1024×8
Memory capacity of ROM M×N = 4K bytes
= 4×1024×8

Required No. of ROM chips = $\frac{4 \times 1024 \times 8}{1024 \times 8} = 4$ ROM chips Ans

(b) interface 4×4 = 16 = 2^4

Size of one RAM = 256×8 = 2^8×8, Size of one ROM = 1024×8 = 2^{10}×8

Component	Address	16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
RAM1	0000-	00FF 16×16 = 256
RAM2	0001-2	

000-255
256-511
512-1023

(b)

Size of RAM = 256 X 8 = 2^8 ⇒ 8-bit Address need for RAM

Size of one ROM = 1024 X 8 = 2^{10} ⇒ 10-bits Address need for ROM

Component	Hexadecimal Address	Address Bus															
		16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
RAM 1	0000 - 00FF	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X
RAM 2	0100 - 01FF	0	0	0	0	0	0	0	1	X	X	X	X	X	X	X	X
RAM 3	0200 - 02FF	0	0	0	0	0	0	1	0	X	X	X	X	X	X	X	X
RAM 4	0300 - 03FF	0	0	0	0	0	0	1	1	X	X	X	X	X	X	X	X
RAM 5	0400 - 04FF	0	0	0	0	0	1	0	0	X	X	X	X	X	X	X	X
RAM 6	0500 - 05FF	0	0	0	0	0	1	0	1	X	X	X	X	X	X	X	X
RAM 7	0600 - 06FF	0	0	0	0	0	1	1	0	X	X	X	X	X	X	X	X
RAM 8	0700 - 07FF	0	0	0	0	0	1	1	1	X	X	X	X	X	X	X	X
ROM 1	4000 - 43FF	0	1	0	0	0	0	X	X	X	X	X	X	X	X	X	X
ROM 2	4400 - 47FF	0	1	0	0	0	1	X	X	X	X	X	X	X	X	X	X
ROM 3	4800 - 4BFF	0	1	0	0	1	0	X	X	X	X	X	X	X	X	X	X
ROM 4	4C00 - 4FFF	0	1	0	0	1	1	X	X	X	X	X	X	X	X	X	X
Interface	8000 - 800F	1	0	0	0	0	0	0	0	0	0	0	X	X	X	X	X

and interface 4 X 4 = 16 = 2^4 ⇒ 4-bit address need.

(c)

The address Range in Hexadecimal for RAM, ROM and Interface are as:

	Hexadecimal Address
RAM	0001 - 07FF
ROM	4000 - 4FFF
interface	8000 - 800F

Ans

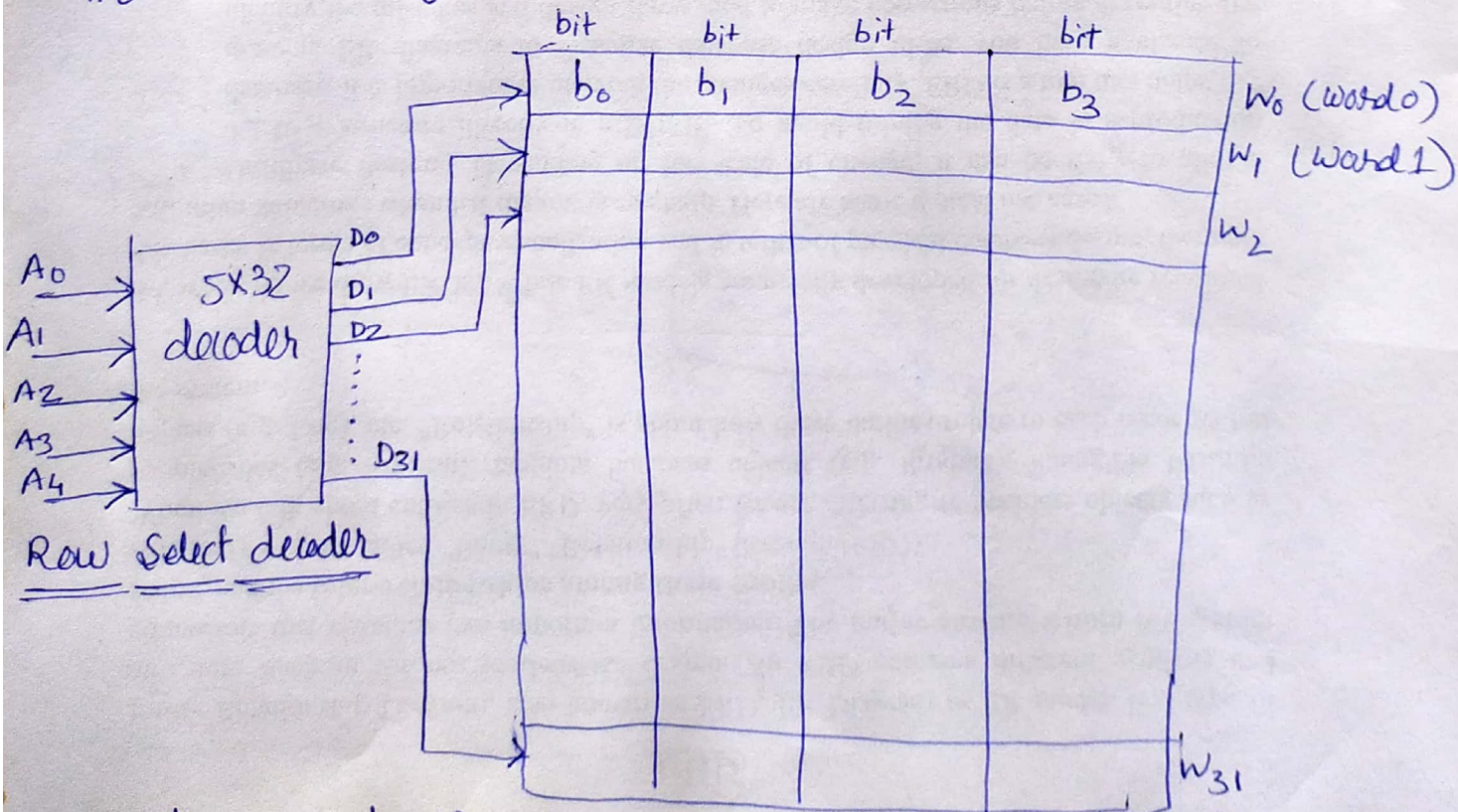
2D Organization of RAM

⇒ In 2D organization, memory is devided in the form of rows and columns. Each row contains a word, and there is a decoder to select each row.

⇒ A decoder is a combinational circuit that contains n input lines and 2^n output lines.

⇒ One of the output lines will select the row ~~which~~ whose address is contained in the MAR and the word which is represented by that row that will get selected and either read or write through the data lines.

ex ⇒ Organization of 32×4 2D RAM. In which 32 word and each word contains 4-bits. and since $2^5 = 32$, so it uses 5×32 decoder to represent the address of 32 word. It is called row select decoder



⇒ Each row have one column for each individual bits.

⇒ In 2D RAM organization, hardware is fixed.

⇒ It requires more numbers of logic gates. It is more complex

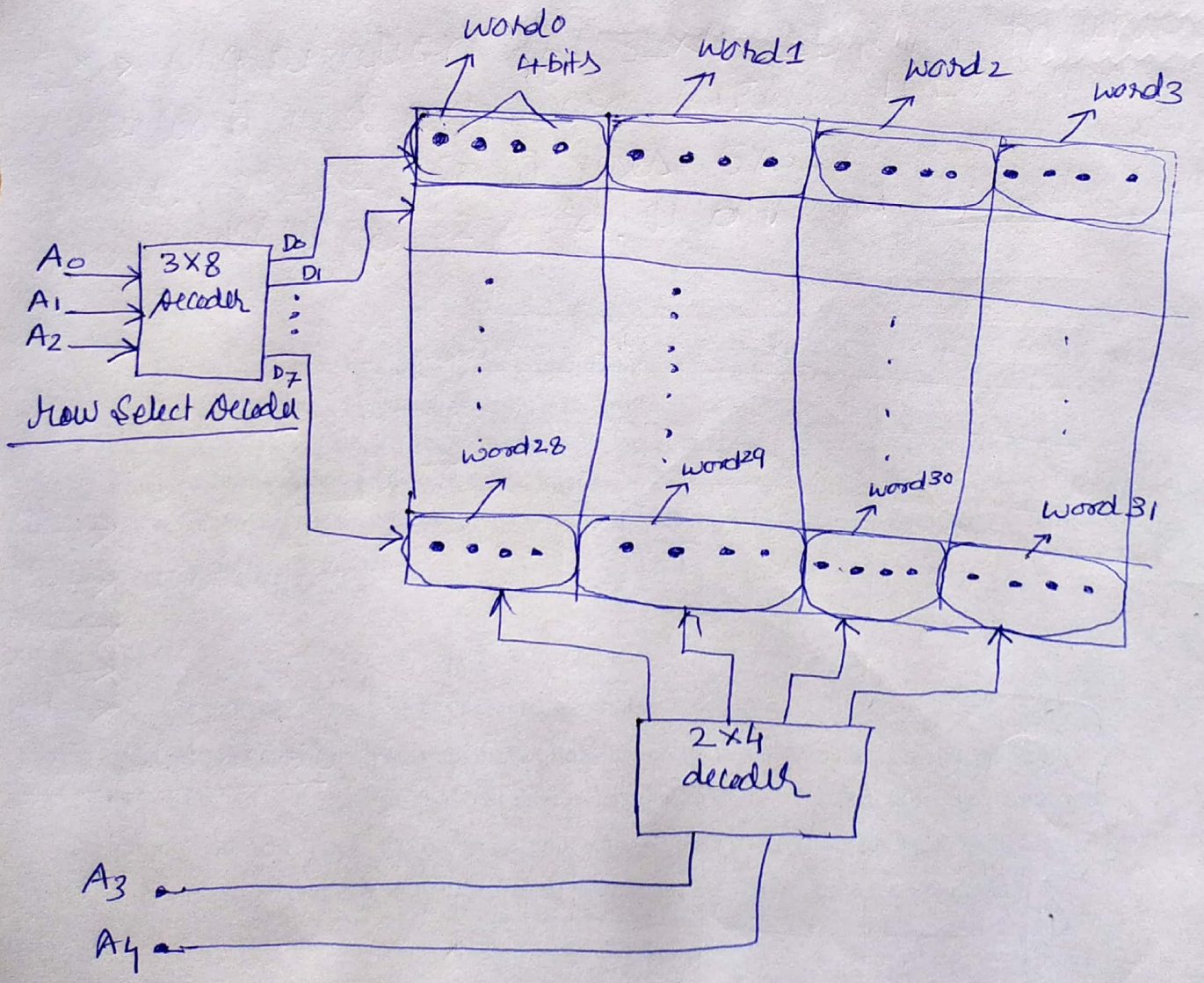
$2\frac{1}{2}$ D RAM Organization

⇒ In $2\frac{1}{2}$ D organization of RAM the number of address lines are divided into approximitly equal parts, one ~~for row decoder~~ decoder for selecting row is called row select decoder and one decoder for selecting column is called column select decoder.

⇒ In $2\frac{1}{2}$ D RAM hardware is variable.

⇒ It requires less number of logic gates.

⇒ $2\frac{1}{2}$ D RAM is less complex.



Cache Memory

⇒ A very high speed memory called a cache memory is used to increase the speed of processing by making current programs and data available to the CP at a rapid rate.

⇒ It is placed between the CPU and the main memory.

⇒ When the CPU needs to access memory, the cache is examined first. If the word is found in the cache, it is read from the cache memory. If the word is not found in cache, the main memory is accessed to read the word. A block of words containing the one just accessed is then transferred from the main memory to the cache memory.

Cache Performance ⇄

⇒ The performance of cache memory is measured in terms of a quantity called Hit ratio

⇒ "When the CPU refers to memory and finds the word in the cache, it is said hit. If the word is not found in cache, it is in the main memory and it count as a miss. The ratio of the number of hits divided by the total CPU references to memory (No. of hits + No. of miss) is called Hit ratio.

$$\text{Hit ratio} = \frac{\text{No. of Hits}}{\text{No. of Hits} + \text{No. of miss}}$$

⇒ let t_c is cache access time

h is Hit ratio

t_m is the main memory access time

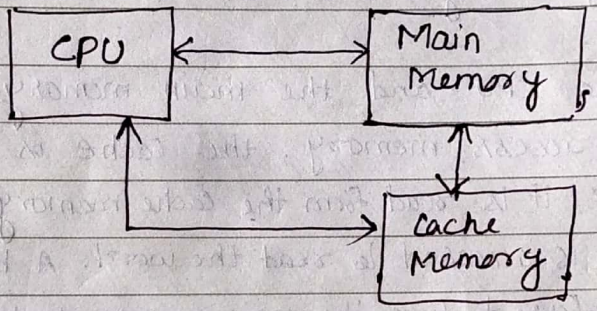
Then, The Average access time T is given by

$$T = ht_c + (1-h)(t_c + t_m)$$

NOTE ⇒ Hit ratio always lies between 0 and 1.

Cache Mapping Process or Cache Mapping techniques

"The transformation of data from main memory to cache memory is called cache mapping process".



⇒ The CPU will send an n-bit address to the cache memory. If there is a hit, the word is fetched from the cache and if there is a miss, the required word is searched in the main memory and then copied from main memory into cache.

⇒ There are three mapping process to take word from main memory to cache memory.

(i) ~~Associative mapping~~ (i) Direct Mapping
(ii) ~~Direct mapping~~ (ii) Associative mapping
(iii) Set-Associative mapping

(i) ~~Associative Mapping~~ Associative Mapping

⇒ In ~~associative~~ mapping, both the word and the address of the word in the main memory are stored in the cache as shown fig next page.

⇒ The address bits, sent by the CPU to search, are matched with the address stored in the cache memory. If any address is matched, the corresponding word is fetched from the cache and sent to the CPU.

⇒ If no match is found in cache memory, the word is searched in the main memory. The word along with address is then copied from main memory into cache.

If cache is full, then the existing word along with its address must be removed to main memory

V.V.IMP

Cache Memory Mapping Techniques

⇒ Cache memory mapping means how data is copied (mapped) from main memory to cache memory.

⇒ There are three mapping techniques
1. Direct mapping
2. Associative mapping
3. Set-Associative mapping

① Direct mapping ⇒ In direct mapping technique, main memory blocks are copied to a fixed block of cache memory, but one at a time.

⇒ In which cache memory block, main memory block will go for that we use

Cache memory block NO = K Mod N

where K = Main memory block number
N = No. of block in cache.

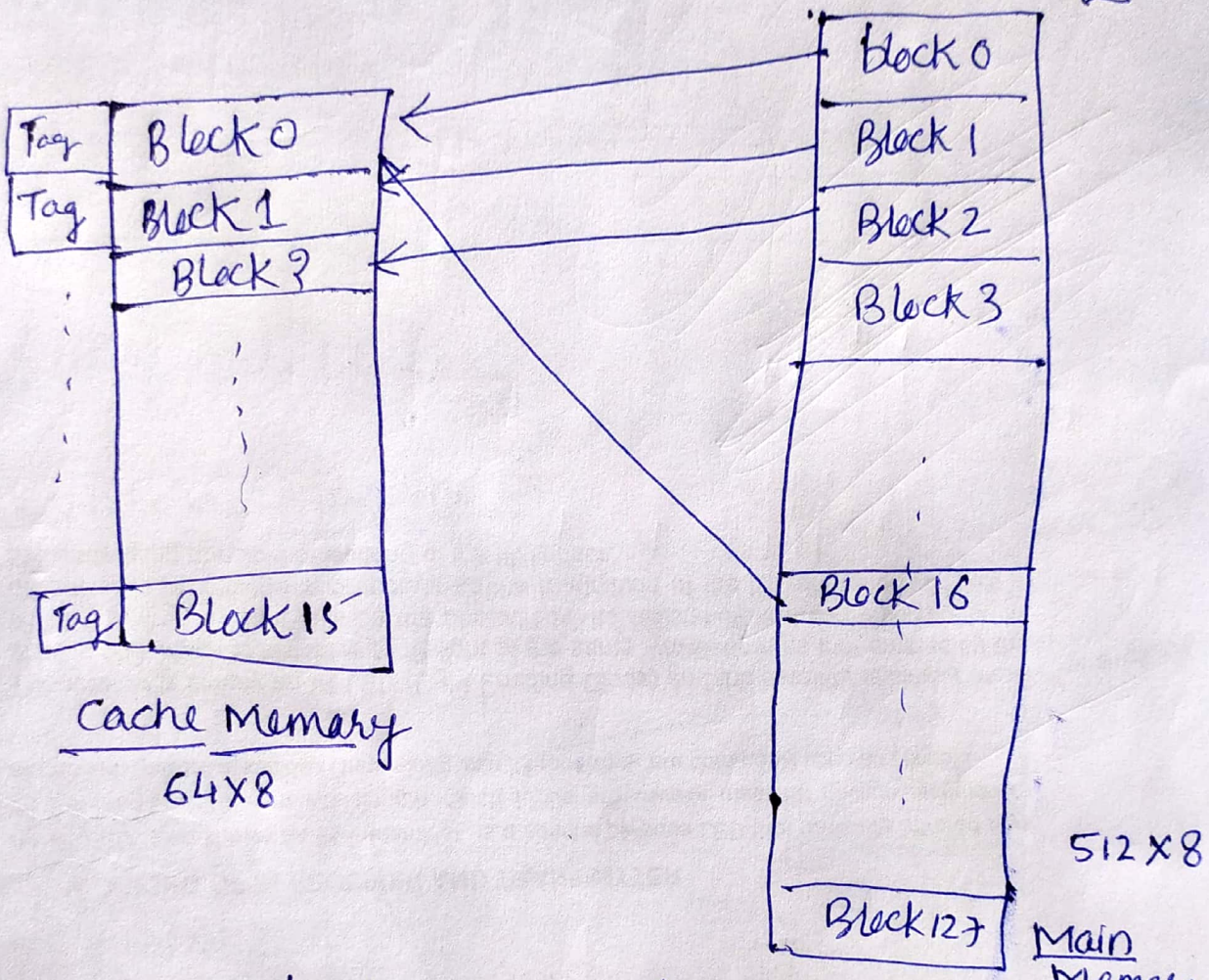
Let Main memory size = 512 X 8
Cache memory size = 64 X 8
Block size = 4 words

No. of Main memory blocks = $\frac{\text{Total main memory words}}{\text{Block size}}$

$= \frac{512}{4} = 128$ blocks in main memory

No. of cache Memory Block = $\frac{\text{Total cache Memory word}}{\text{Block size}}$

$= \frac{64}{4} = 16$ blocks in cache



Cache Memory block No = K Mod N

$\Rightarrow 0 \bmod 16 = 0$
$\Rightarrow 1 \bmod 16 = 1$
$\Rightarrow 2 \bmod 16 = 2$
$\Rightarrow 17 \bmod 16 = 1$
$\Rightarrow 33 \bmod 16 = 1$

$16 \bmod 16 = 0$
$32 \bmod 16 = 0$
$48 \bmod 16 = 0$

No. of main memory blocks in one block of cache = $\frac{\text{No. of main memory block}}{\text{No. of cache Memory block}}$

$= \frac{128}{16} = 8$ block

But one block at a time

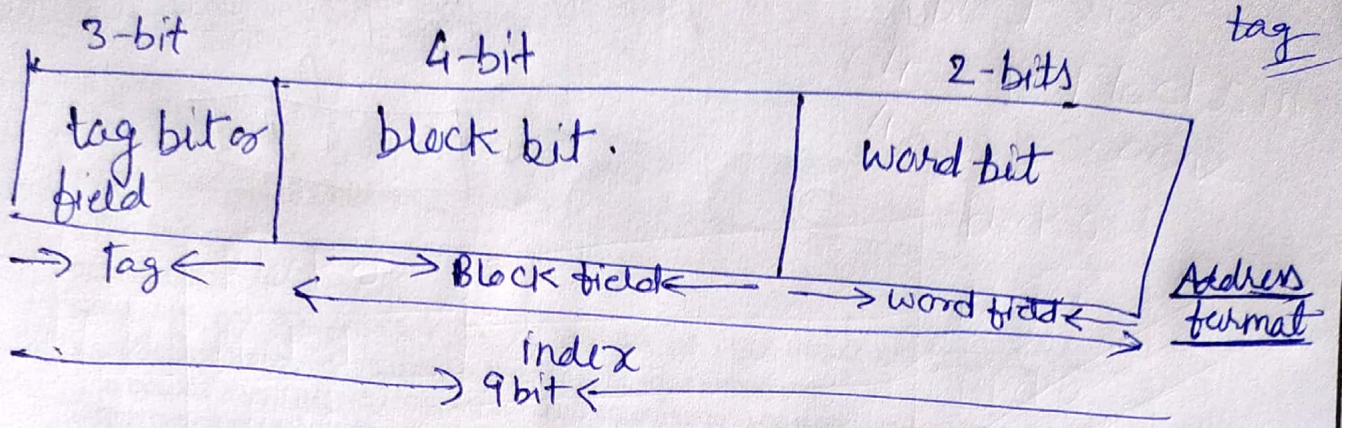
Main memory size = 512×8

Main memory Address = 2^9

= 9-bits

2^9 2^6

9-6 = 3 tag



No. of cache block = 16

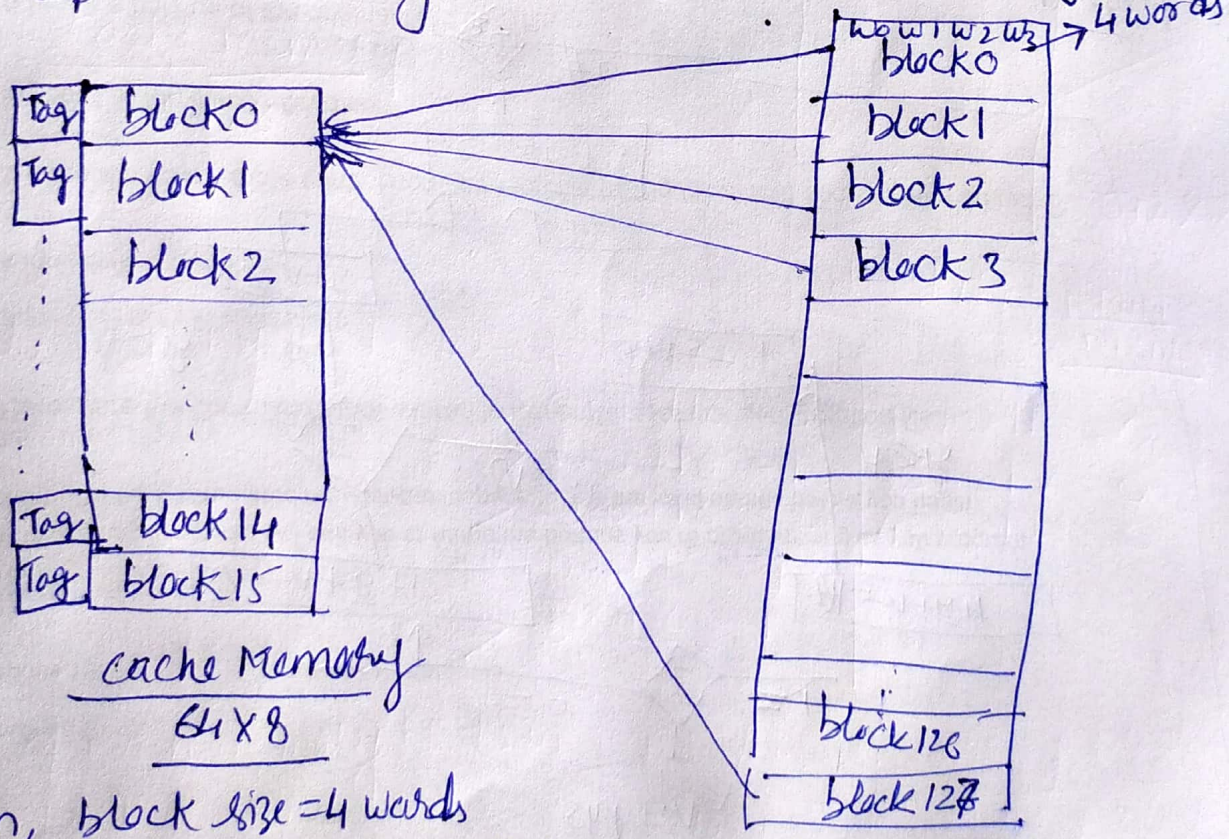
$= 2^4 \Rightarrow$ 4-bit

block size = 4 word

$= 2^2 =$ 2-bits

Associative Mapping →

In associative mapping, main memory blocks are copied into any block of cache memory.

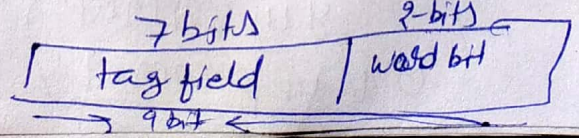


given, block size = 4 words

No. of main memory block = $\frac{512}{4}$ = 128 blocks

No. of cache memory block = $\frac{64}{4}$ = 16 blocks

Address Format



(3) Set-Associative mapping ⇒ It is a combination of direct mapping and associative mapping.

Set-Associative Mapping	=	Direct Mapping	+	Associative Mapping

⇒ In this, cache mapping is divided into set.

⇒ Set ⇒ Group of blocks

⇒ Block ⇒ Group of words

$$\text{Cache Memory Set No} = (\text{Main memory block No.}) \bmod (\text{No of sets in cache memory})$$

$$\text{No. of set in cache memory} = \frac{\text{No. of Cache Memory blocks}}{\text{Set Size}}$$

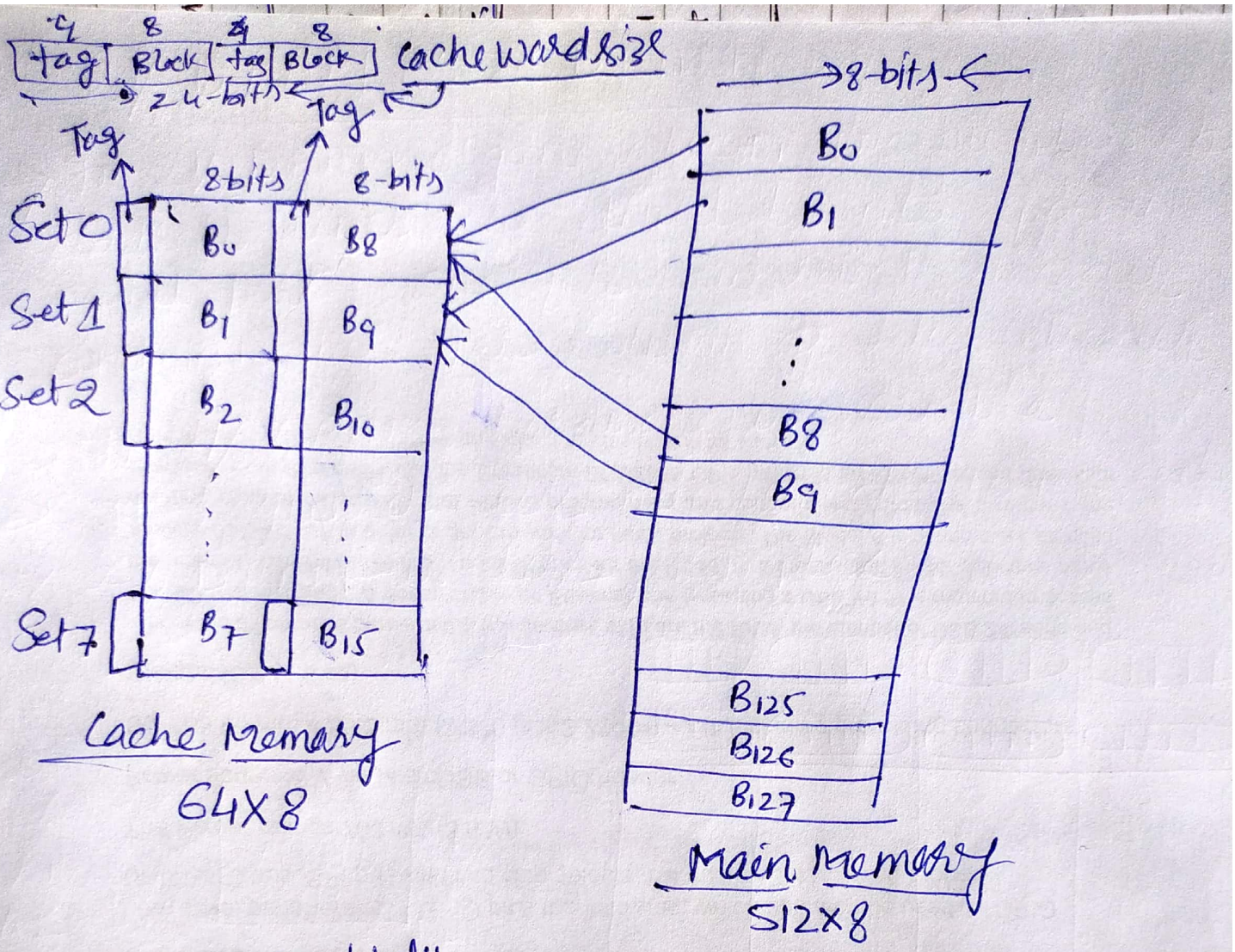
Let Set size = 2 (Two-way set associative)
Block size = 4 words, main memory = 512 x 8
cache memory = 64 x 8

No. of main memory blocks = $\frac{512}{4} = 128$ block
No. " cache " " = $\frac{64}{4} = 16$ block
No. of set in cache memory = $\frac{16}{2} = 8$ set

cache memory set No.

⇒ 0 mod 8 = 0
⇒ 8 mod 8 = 0
⇒ 16 mod 8 = 0
⇒ 24 mod 8 = 0

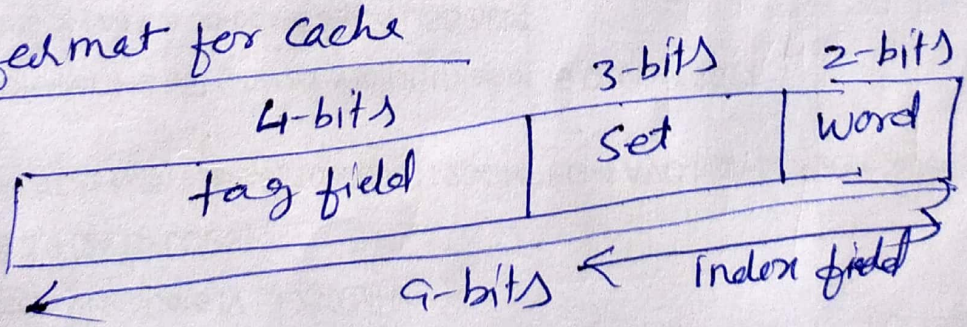
⇒ 1 mod 8 = 1
⇒ 9 mod 8 = 1
⇒ 17 mod 8 = 1
⇒ 25 mod 8 = 1



No. of main memory blocks in one set of cache (2 at a time) = $\frac{\text{main memory block}}{\text{cache memory block}}$

$= \frac{128}{8} = 16$ blocks

Address format for cache



(2) A digital computer has a memory unit of 64K X 16 and a cache memory of 1K words. The cache uses "Direct mapping" with a block size of 4 words

(a) How many bits are there in the TAG, INDEX and word field of Address format

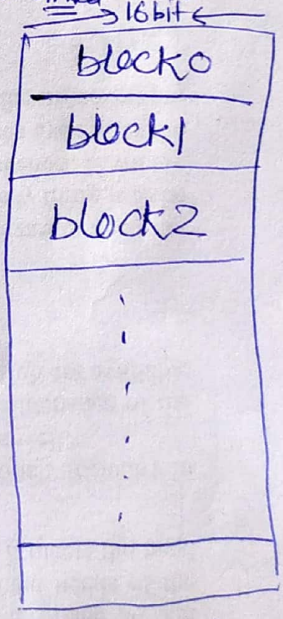
(b) How many bits are there in each word of cache, and how ~~they~~ are they divided into functions? include a Valid bit.

(c) How many blocks can the cache accomodate?

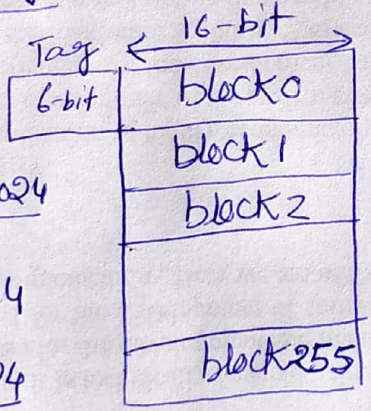
Soln ⇒ Given, Main memory size = 64K X 16 ⇒ 64 X 1024 X 16
⇒ $2^6 \times 2^{10} \times 16$

cache memory size = 1K ⇒ 1024 ⇒ 2^{16} ⇒ 16-bit Address
2^{10} ⇒ 10-bit Add

Direct Mapping: block size = 4 words



$$\text{No. of Block in Main Memory} = \frac{64 \times 1024}{4} = 16 \times 1024$$



$$\text{No. of Block in cache memory} = \frac{1024}{4} = 256 \text{ block}$$

$$\frac{1K}{1 \times 1024} = 2^{10}$$

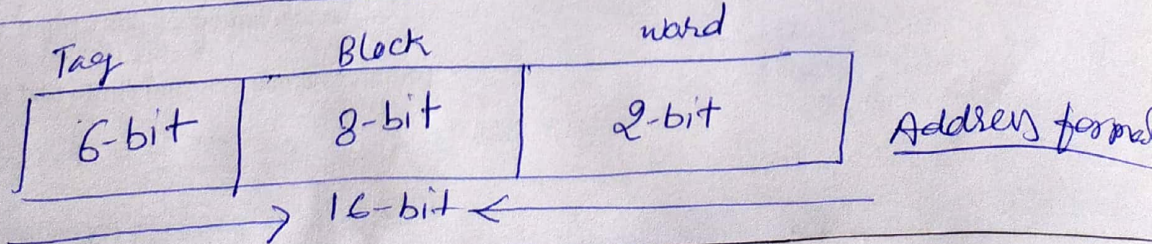
⇒ 10-bit Address

$$\frac{64 \times 1024 \times 16}{2^6 \times 2^{10}} = 2^{16}$$

⇒ 16-bit address

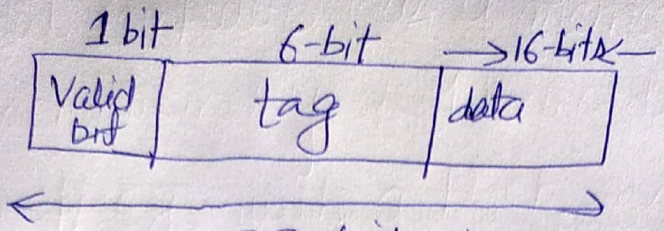
tag = M − N = 16 − 10 = 6

(a)



Address format

(b)



23-bits in each word of cache

(i)

(c) No. of blocks in cache can accomodate

$= 2^8 = 256$ Block

(ii)

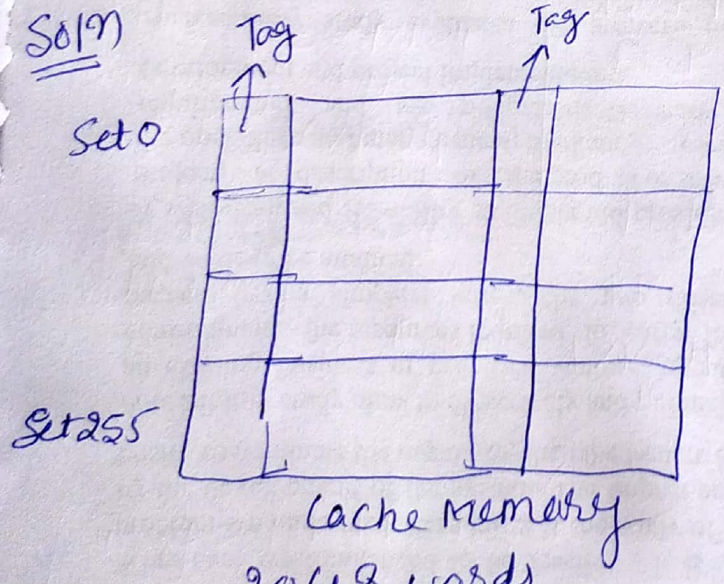
(iii)

(a) A two-way set associative cache memory uses block of four words. The cache can accomodate a total of 2048 words of Main Memory. The main Memory size is 128K X 32

(a) Formulate all pertinent information required to construct the cache memory.

(b) What is the size of cache memory

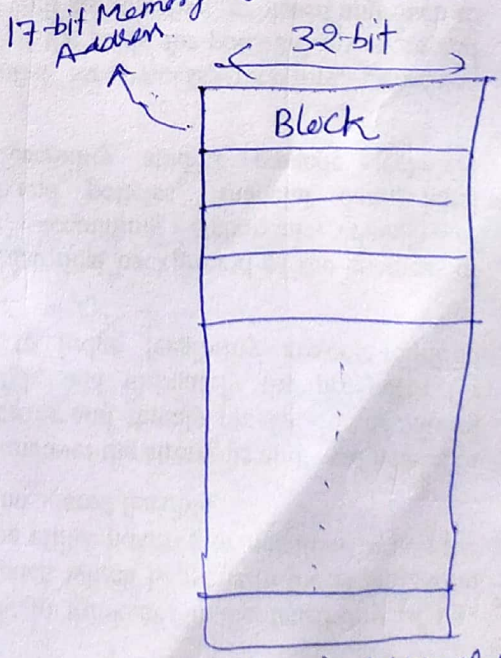
Soln



Cache Memory

2048 words

$\Rightarrow 2^{20}$



Main Memory

128K X 32

$\Rightarrow 128 \times 1024$

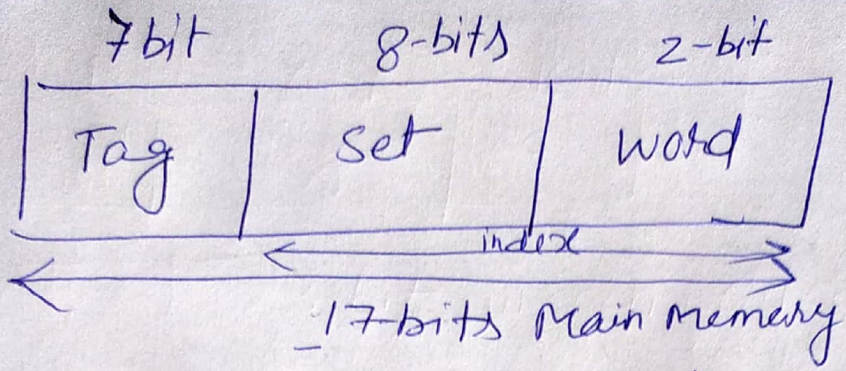
$\Rightarrow 2^7 \times 2^{10} \Rightarrow$ 17 bit Addr

No. of Block in cache $= \frac{2048}{4} = 512$ block $\Rightarrow$ Set $= \frac{512}{2} = 256$ Set

No. of Block in Main Memory $= \frac{128 \times 1024}{4}$

Sol

(a)



Tag = Main memory ~~Address bit~~ - ~~cache memory Address~~ bit

$$\text{Tag} = \frac{\text{Main memory blocks}}{\text{Cache memory sets}} = \frac{32k}{256} = \frac{32 \times 1024}{256}$$

$$= 128$$

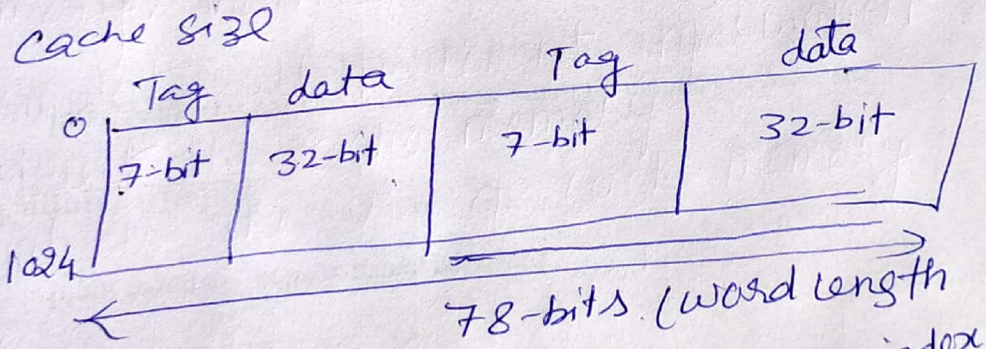
$$\Rightarrow 2^7 \Rightarrow 7 \text{ bit}$$

Set = No. of set in cache memory

$$= 256 = 2^8 \Rightarrow 8\text{-bits Set Address}$$

(b)

Cache size



Cache Size = 1024×78

Size of cache memory = No. of words × word length → index ~~field~~ field

$$= 2^{10} \times 78$$

$$= 1024 \times 78 \text{ ans}$$

Writing into cache

⇒ When CPU need any data it searches it in a cache if data is not present in cache then it is called Miss. and processor retrieved data from main memory and a block of data is transferred from main memory to cache memory.

⇒ When the ~~processes~~ processor changes the contents of cache it is also necessary to update its main memory copy also. This can be done in two ways.

(i) Write-through (ii) Write-Back

(i) Write-through

⇒ In write-through method, whenever the CPU updates a cache contents the same updation is made in the main memory copy immediately.

⇒ This method has the advantage that main memory always contains the updated data as in cache memory.

(ii) Write-Back

⇒ In write-back method, whenever the CPU (update) writes something into a cache block, that block is marked as flag. when a block that is marked as a flag is to be replaced by a new block, the flag marked block is copied back into the main memory before it is overwritten by the new block.

(2) 12.16 : The access time of a cache memory is 100ns and that of main memory 1000ns. It is estimated that 80% of memory requests are for read and the remaining 20% for write. The hit ratio for read accesses only is 0.9. A write-through procedure is used.

(i) What is the average access time of the system considering only memory read cycles?

(ii) What is the average access time of the system for both read and write requests

(iii) What is the hit ratio taking into consideration the write cycles?

Sol^{n} ⇒ given, $t_c = 100ns$, $t_m = 1000$, $h = 0.9$ (hit ratio of read)

(i) average access time of read $T_c = ht_c + (1-h)(t_m + t_c)$

$= 0.9 \times 100 + (1-0.9)(1000+100)$

$= 90 + 0.1 \times 1100$

$= 90 + 110$

Average access time of Read $T_c = 200ns$ Ans

(ii) For both read and write cycle average access time = $P_R \times$ Avg. access time for read $+ (1-P_R) \times t_m$

$= 80\% \times 200 + (1-80\%) \times 1000$ (Here, P_R = Percentage of Read)

$= \frac{80}{100} \times 200 + (1 - \frac{80}{100}) \times 1000$

$= 160 + (1-0.8) \times 1000$

$= 160 + 0.2 \times 1000$

$= 160 + 200$

$= 360ns$ Ans

(iii) Hit ratio when write cycle also considerd

$= P_R \times$ hit ratio of read

$= 80\% \times 0.9$

$= \frac{80}{100} \times 0.9$

$= 0.8 \times 0.9$

$= 0.72$ Ans

(2) 12.16 : The access time of a cache memory is 100ns and that of main memory 1000ns. It is estimated that 80% of memory requests are for read and the remaining 20% for write. The hit ratio for read accesses only is 0.9. A write-through procedure is used.

(i) What is the average access time of the system considering only memory read cycles?

(ii) What is the average access time of the system for both read and write requests

(iii) What is the hit ratio taking into consideration the write cycles?

Sol^n ⇒ given, $t_c = 100ns$, $t_m = 1000$, $h = 0.9$ (hit ratio of read)

(i) average access time of read $T_c = ht_c + (1-h)(t_m + t_c)$

$= 0.9 \times 100 + (1-0.9)(1000+100)$

$= 90 + 0.1 \times 1100$

$= 90 + 110$

Average access time of Read $T_c = 200ns$ Ans

(ii) For both read and write cycle average access time $= P_R \times$ Avg. access time for read $+ (1-P_R) \times t_m$

$= 80\% \times 200 + (1-80\%) \times 1000$ (Here, P_R = Percentage of Read)

$= \frac{80}{100} \times 200 + (1 - \frac{80}{100}) \times 1000$

$= 160 + (1-0.8) \times 1000$

$= 160 + 0.2 \times 1000$

$= 160 + 200$

$= 360ns$ Ans

(iii) Hit ratio when write cycle also considerd

$= P_R \times$ hit ratio of read

$= 80\% \times 0.9$

$= \frac{80}{100} \times 0.9$

$= 0.8 \times 0.9$

$= 0.72$ Ans

Virtual Memory

"Virtual memory is a technique that allows the execution of processes that may not be completely in main memory"

⇒ The main advantage of this scheme is that programs can be larger than the physical memory. Portion of the program or data are brought into main memory from secondary memory when they are required by the CPU.

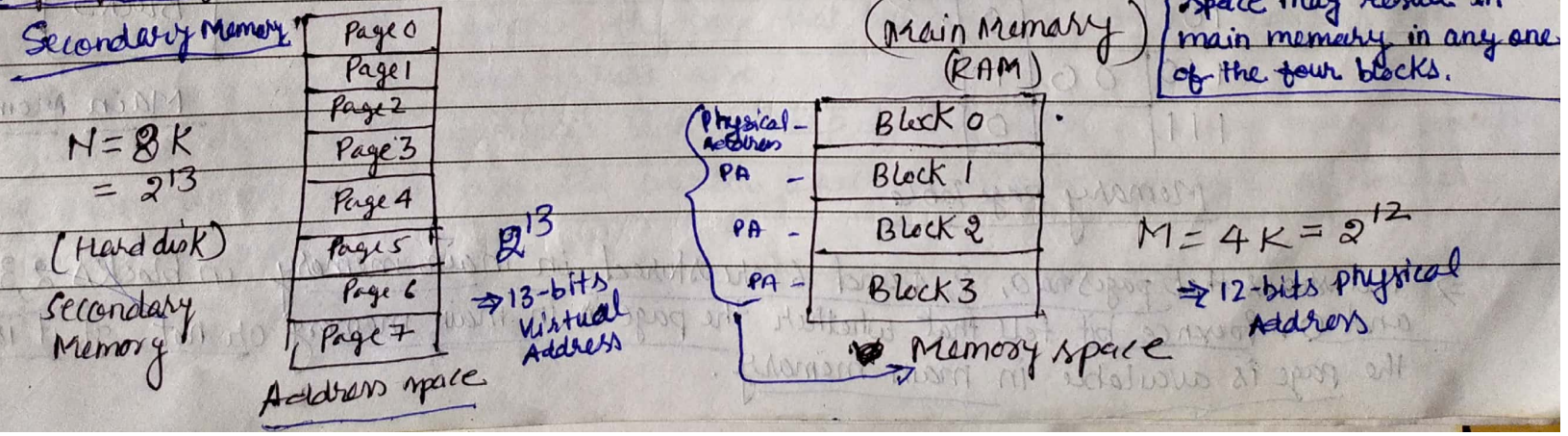
⇒ An address generated by user program is called virtual address and the set of virtual addresses make the virtual address space. Virtual Address are in secondary memory.

⇒ A Main memory address is called physical address and set of such addresses are called memory space.

⇒ Virtual Address are always greater than physical address.

⇒ The address space and memory space can be divided into groups of equal size. The address space is broken into group of equal size known as page and memory space is broken into group of same size known as blocks.

⇒ Consider, a computer with an address space 8K and memory space of 4K if the each group is of 1K words, we have 8 pages and 4 blocks. At any given time, upto four pages of address space may reside in main memory in any one of the four blocks.



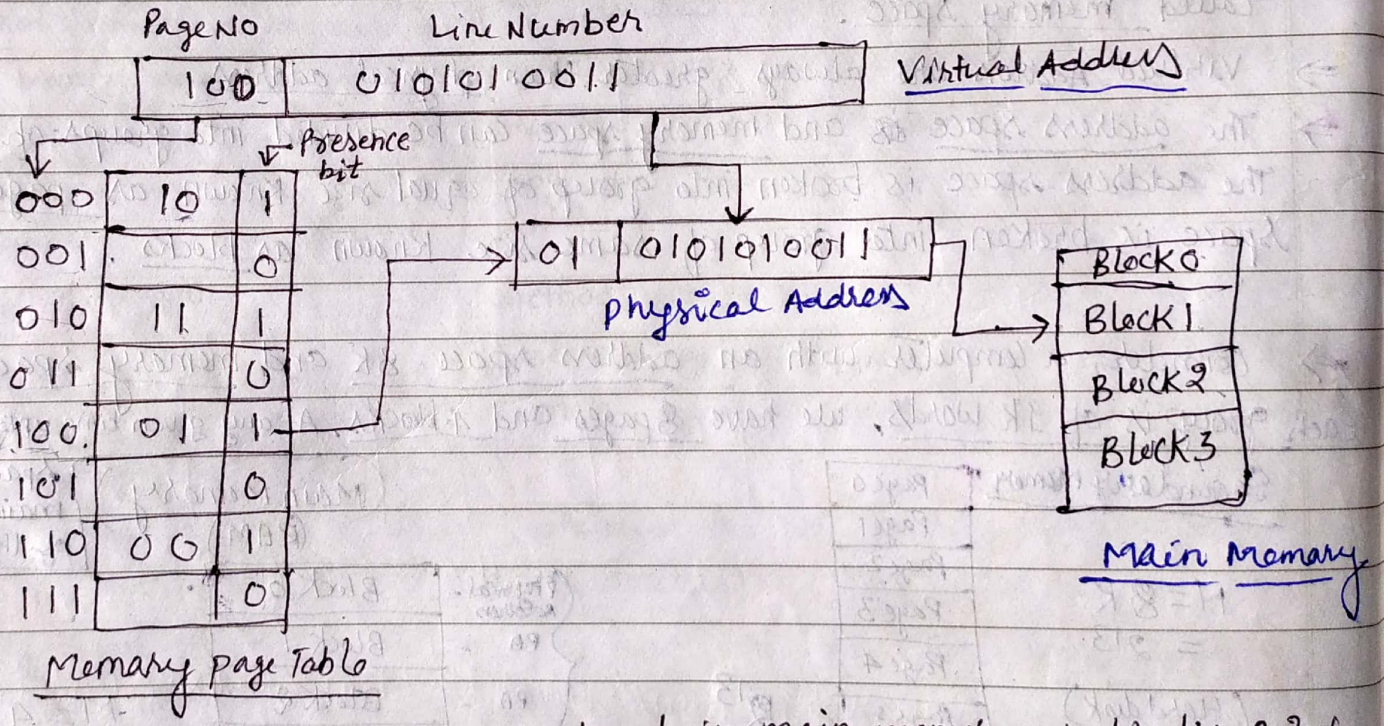
Every virtual address generated by the CPU is divided into two pars:

Mapping of Virtual memory

(1) a page number address and (2) a line within the page. In a computer with 2^p words per page, p bits are used for line-address and remaining high-order bits of the virtual address specify page number.

⇒ In the example, virtual address is 13 bits each page consists of 1K words i.e 2^{10} words so, 10 bits will be used to specify line number and three high-order bits of the virtual address will specify one of the eight pages of the address space.

⇒ The line number in address space and memory space is same Hence only mapping required from a page number to a block number.



⇒ This shows that pages no. 0, 2, 4 and 6 are stored in main memory in blocks 2, 3, 1 and 0. Presence bit tell that whether the page is in main memory or not. If it is 1 the page is available in main memory.

Page Replacement Policies

The most commonly used replacement algorithms are first-in-first-out (FIFO) and least recently used (LRU).

FIFO ⇒ When a page is to be replaced from main memory, the oldest page in the main memory is chosen. i.e the page that has been in memory for the longest time.

⇒ When the page loaded into memory, an identification number is pushed into the FIFO stack. The page whose identification number is at the top of of FIFO stack is removed.

⇒ The advantage of FIFO algorithm is that it is easy to understand and implement but its performance is not always good.

LRU (Least Recently Used)

⇒ In LRU algorithm, the page that has not been used for the longest period of time is replaced by a new page.

⇒ The principle is very simple, the page that has not been used till now is not going to be used in near future also,

⇒ This algorithm can be implemented with the help of a counter. Each page that is presently in main memory has its associated counter which is incremented by 1 at regular interval of time.

⇒ The page with highest count is the ~~least recently used page~~ and is replaced by the new page when required.

Q 12.19 ⇒ An address space is specified by 24 bits and the corresponding memory space by 16 bits.

(a). How many words are there in the address space?
(b). How many words are there in the memory space?
(c). If a page consists of 2K words, How many pages and blocks are there in the system?

Q (12.19) Soln given address space = 24 bits
i.e size of Address space = 2^{24}

memory space = 16 bits
size of main memory = 2^{16}

(a) No. of words in Address space = $2^{24} = 2^4 \times 2^{20}$
= 16M words Ans

(b) No. of words in memory space = $2^{16} = 2^6 \times 2^{10}$
= 64 K words Ans

(c) size of one page = 2K words

No. of Pages = $\frac{\text{size of secondary memory}}{\text{size of one page}} = \frac{16M}{2K} = \frac{2^4 \times 2^{10} \times 2^{10}}{2 \times 2^{10}}$

$= 2^3 \times 2^{10}$
$= 8K$ pages
$= 8 \times 1024$ pages
$= 8192$ pages Ans

No of Blocks = $\frac{64K}{2K} = 32$ blocks Ans

Q 12.20 → Morris Mano (Book)

given, Virtual memory page size = 1K words

No. of Pages = 8

Size of virtual memory = 8 × K = 8K words

memory page table Contains the following entries:

Page	Block
0	3
1	1
4	2
6	0

Since there are 8 Pages only four block can be filled at time since there are only 4 blocks remaining pages and address that will cause fault are

Page	address	Address that will Cause fault
2	2K	2048-3071
3	3K	3072-4095
5	5K	5120-6143
7	7K	7168-8191

Ans

Q (12.21) Book (Moris Mano)

(a)

In case of FIFO

4 2 0 1 2 6 1 4 0 1 0 2 3 5 7

Since memory space is 4K word so there can be four Pages at one time

FIFO

Page Reference	Page in Main Memory	contents of FIFO	
initial	0 1 2 4	4 2 0 1	
2	0 1 2 4	4 2 0 1	// 2 is already in queue
6	0 1 2 6	2 0 1 6	
1	0 1 2 6	2 0 1 6	// 1 is already
4	0 1 4 6	0 1 6 4	
6	0 1 4 6	0 1 6 4	
2	1 2 4 6	1 6 4 2	
3	2 3 4 6	6 4 2 3	
5	3 3 4 5	4 2 3 5	
7	2 3 5 7	2 3 5 7	

In case of LRU (Least Recently used)

Page Reference	Pages in memory	LRU ↓ Most recently used ↓
initial	0 1 2 4	4 2 0 1
2	0 1 2 4	4 0 1 2
6	0 1 2 6	0 1 2 6
1	0 1 2 6	0 2 6 1
4	0 2 4 6	2 6 1 4
0	0 1 4 6	6 1 4 0
1	0 1 4 6	6 4 0 1
0	0 1 4 6	6 4 1 0
2	0 1 2 4	4 1 0 2
3	0 1 2 3	1 0 2 3
5	0 2 3 5	0 2 3 5
7	2 3 5 7	2 3 5 7

Q 12.23 → Book Moris Mano

Size of segment = 128
$= 2^7 \Rightarrow$ 7-bit segment ~~Address~~

Page size = 32 = $2^5 \Rightarrow$ 5-bits page Address

word = ~~4K~~ $= 2^{12} \Rightarrow$ 12-bits Address

Logical Address :

7 bits	5 bits	12 bits
Segment	page	word

(b)

Physical address :

12 bits	12 bits
Block	Word

Auxiliary Memory or Secondary Memory

⇒ The storage device that provide backup storage is called auxiliary memory.

⇒ Auxiliary memory has a large storage capacity and is relatively inexpensive. bus has low access speed as compared to main memory.

⇒ The most commonly used auxiliary memory devices used in computer system are magnetic disks, optical disks, flash memory, Magnetic tape.

Magnetic Disk

⇒ Magnetic disk are circular metal plate coated with magnetized material on both sides.

⇒ Several disks are stacked to a spindle one below the other with read/write head to make a disk pack.

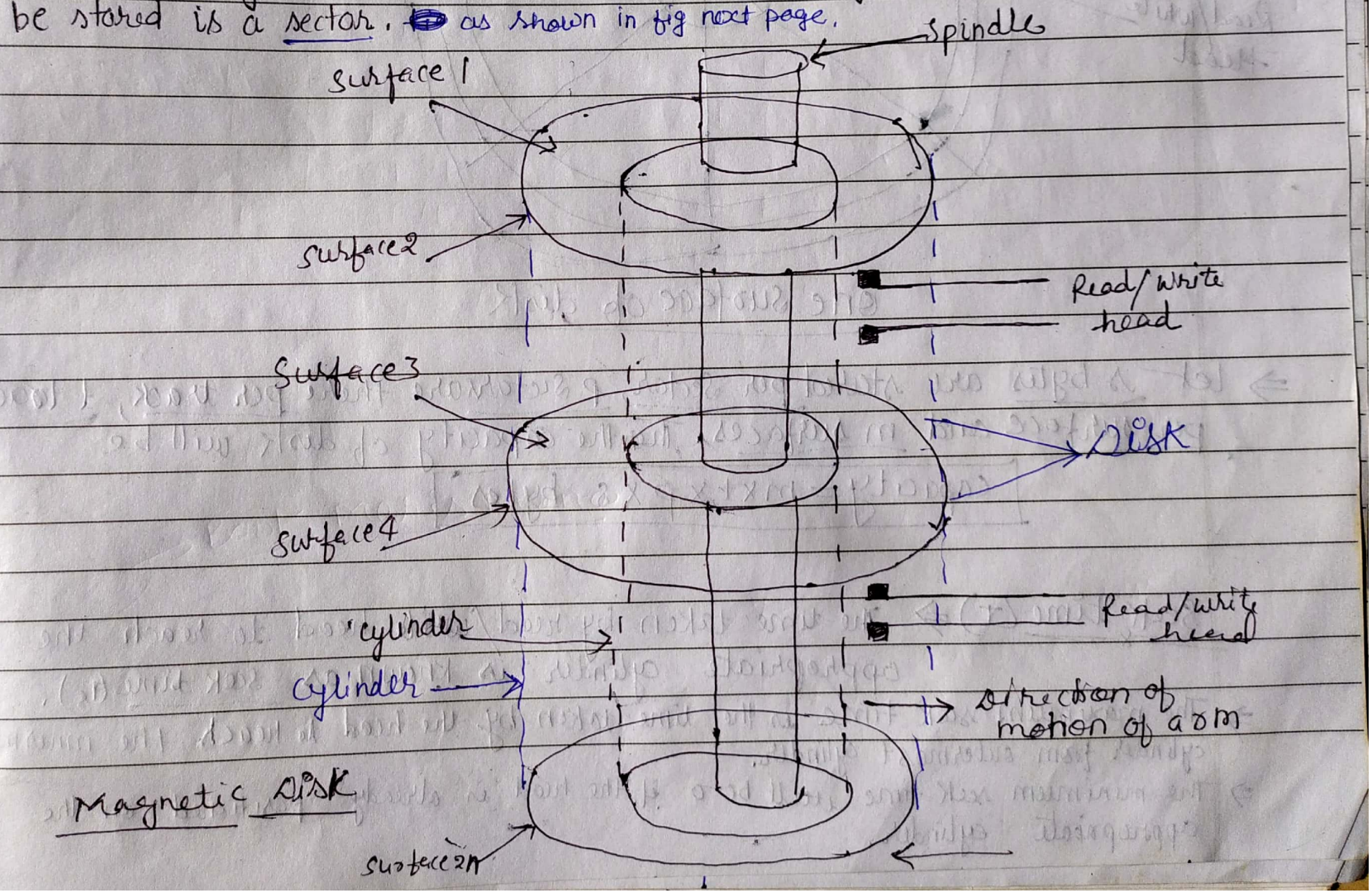
⇒ The disk drive consists of a motor and all disks rotate together at very high speed.

⇒ Informations are stored on the surface of a disk along set of rings called tracks. These tracks are divided into sections

called Sector.

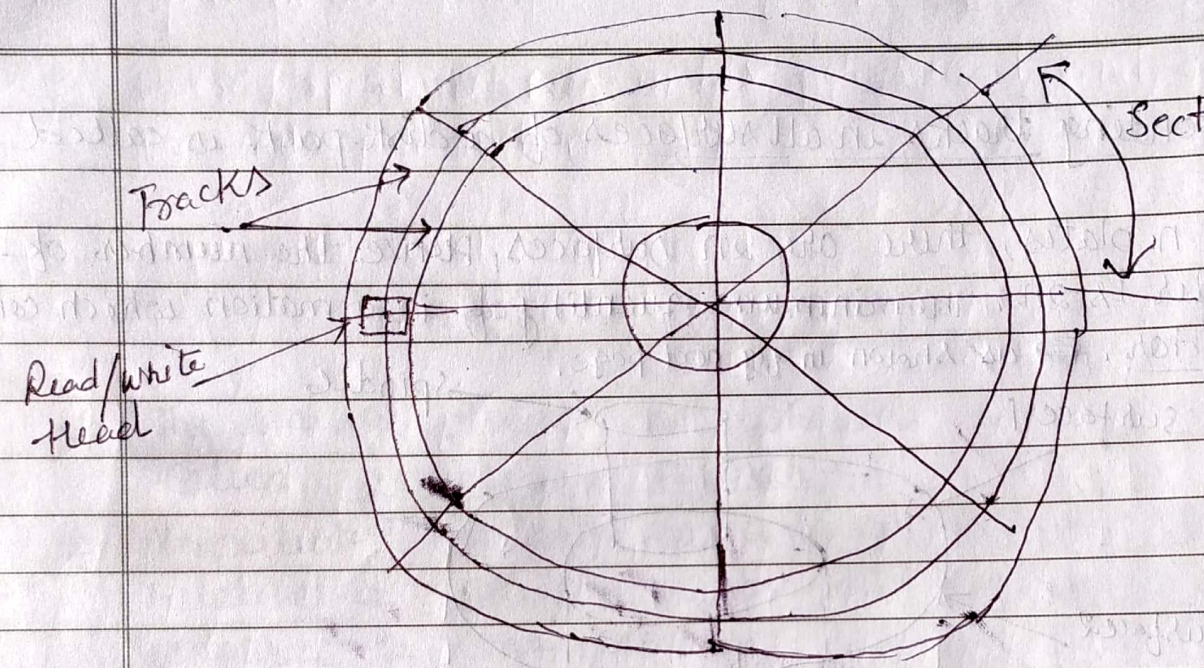
⇒ A set of corresponding tracks in all surfaces of a disk pack is called cylinder.

⇒ If a disk has n plates, there are 2n surfaces, Hence the number of tracks per cylinder is 2n. The minimum quantity of information which can be stored is a sector. as shown in fig next page.



Magnetic Disk

The one surface of disk with tracks and sector is show below



one surface of disk

⇒ let s bytes are stored per sector, p sectors are there per track, t tracks per surface and m surfaces. Then the capacity of disk will be

$$\text{capacity} = m \times t \times p \times s \text{ bytes}$$

Seek time (T) ⇒ The time taken by read/write head to reach the appropriate cylinder is known as seek time (T_s).

⇒ The maximum seek time is the time taken by the head to reach the innermost cylinder from outermost cylinder.

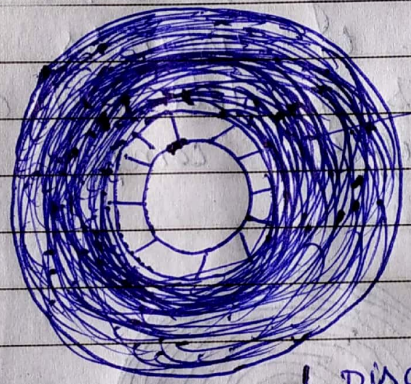
⇒ The minimum seek time will be 0 if the head is already positioned on the appropriate cylinder.

Magnetic Tape :- In magnetic tape only one side of ribbon is used for storing data. It is sequential memory which contains thin plastic ribbon to store data and coated by magnetic oxide.

⇒ Data read/write is slower because of sequential access.

⇒ It is highly reliable which requires magnetic tape drive for writing and reading data.

⇒ The width of the ribbon varies from 4mm to 1 inch and it has storage capacity 100 MB to 200 GB.



Magnetic Tape Memory

Advantages :

1. These are inexpensive
2. It provide backup storage.
3. It can be used for large files.
4. It is a reusable memory.
5. It can be used for copying from disk files.
6. It is compact and easy to store on racks.

Disadvantages :

1. It does not allow access randomly.
2. It require caring to store. like humidity, dust free.
3. If stored data cannot be easily updated or modified.

Optical Disk :-

⇒ An optical disk is any computer disk that uses optical storage techniques and technology to read and write data.

⇒ It is a computer storage disk that stores data digitally and uses laser beams to read and write data.

⇒ An optical disk is primarily used as a portable and secondary storage device.

⇒ Compact disks (CD), digital video disk (DVD) and Blue-ray disks are example of optical disk.

⇒ These disk are commonly used to:

* Distribute software to customers.
* Store large amounts of data such as music, images and videos.
* Transfer data to different computers or devices.

Optical Disk

Flash Memory ⇒

⇒ Flash memory is a non-volatile memory chip used for storage and for transfering data between a computer and digital devices.

⇒ It has the ability to be electronically reprogrammed and erased.

⇒ Flash memory is a type of electronically erasable programmable read only memory (EEPROM)

⇒ It is less expensive.

⇒ It can erase all the data on an entire chip at one time like a camera's flash.

Example: Portable devices such as: Digital camera, smartphones, MP3 player, USB drives, etc...

Advantages: ① less power consumed
② Non-volatile memory
③ Portable
④ large store space