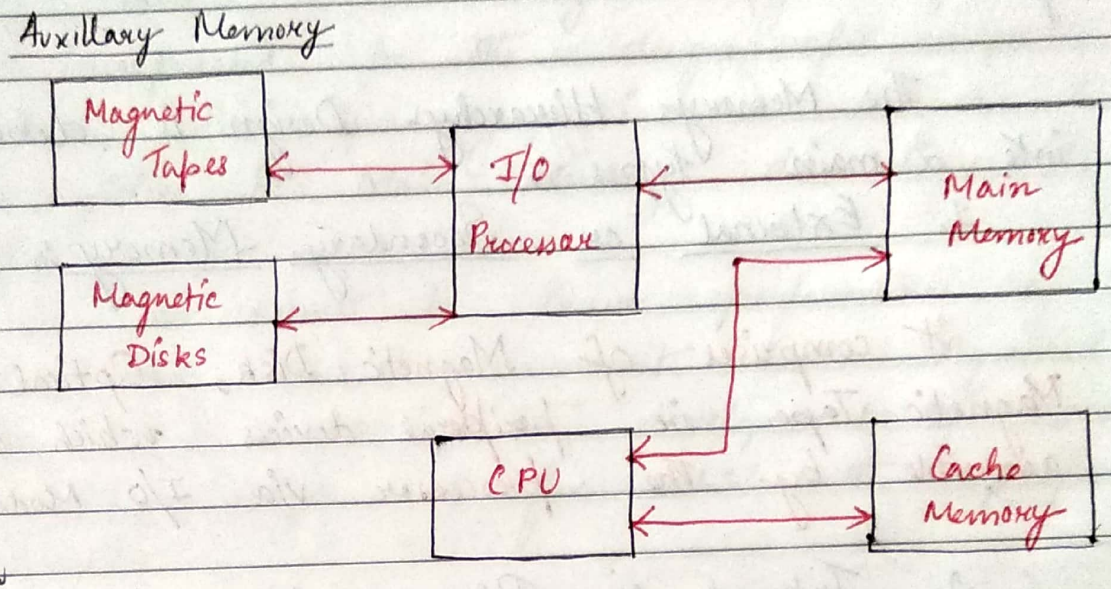
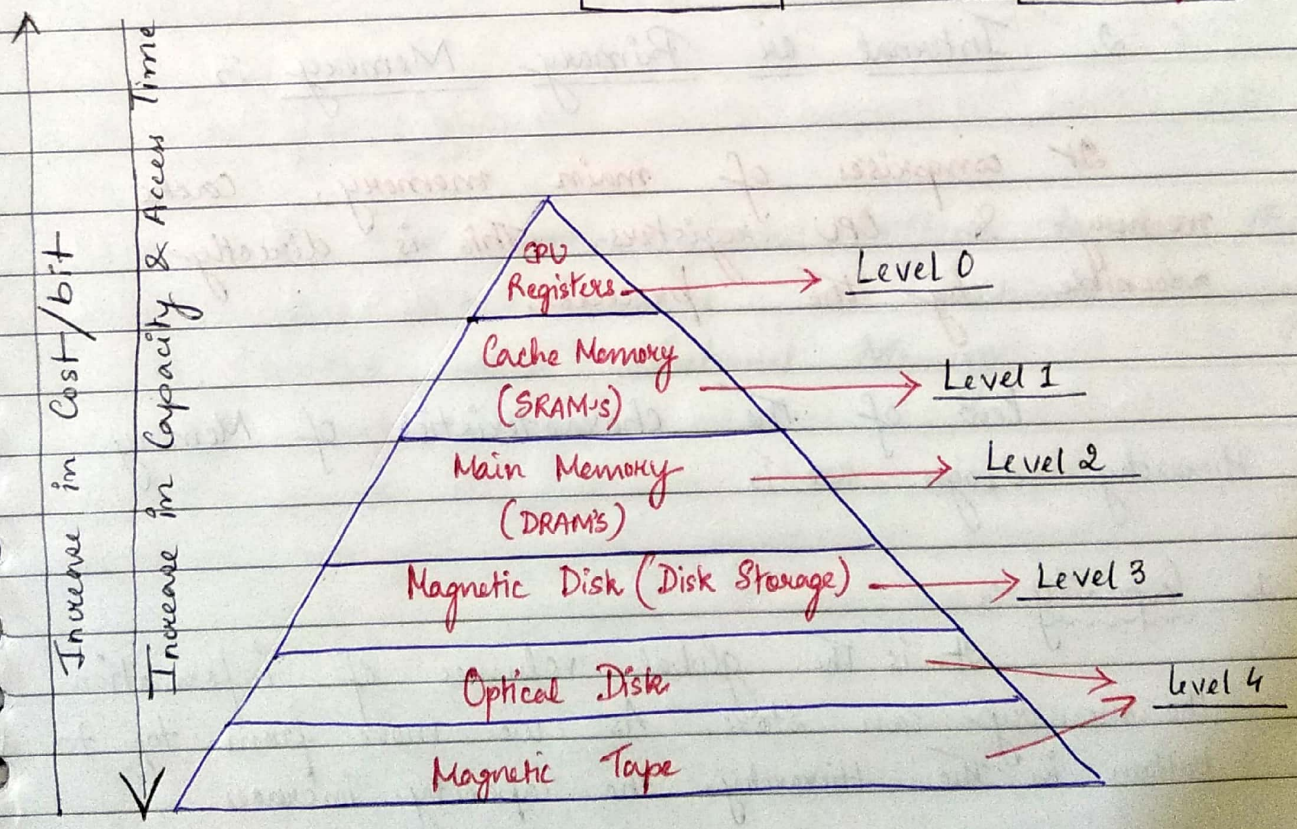
Date.....

MEMORY HIERARCHY IN COMPUTER SYSTEM

General Flow Diagram :-





Spiral

Scanned with CamScanner

Date.....

In computer system Design, Memory hierarchy is an enhancement to the organisation of memory such that it can minimize the access time.

The memory hierarchy was developed based on a program behaviour known as Locality of references.

The Memory Hierarchy Design is divided into 2 main types :

1. External or Secondary Memory :

It comprises of Magnetic Disk, Optical Disk, Magnetic Tape i.e peripheral devices which are accessible by the processor via I/O Module.

2. Internal or Primary Memory :

It comprises of main memory, cache memory & CPU registers. This is directly accessible by the processor.

Few of the characteristics of Memory Hierarchy design are :

1. Capacity :

It is the global volume of information the memory can store. As we move from top to bottom in the Hierarchy, the capacity increases.

2. Access Time :

It is the time interval between the read/write request & the availability of data. As we move from

Spiral

Date.....

top to bottom in the hierarchy, the access time increases.

3. Performance :-

Earlier when Memory Hierarcy design was not incorporated in the design of computer system, the speed gap increases between the CPU registers & Main Memory due to a large difference in access time.

This results in lower performance of the system and thus, enhancement was required. Then this results in the development of memory hierarchy design, & for which the performance increased.

4. Cost per bit :-

As we move from bottom to top the cost per bit increases i.e Internal Memory is costlier than External Memory.

— o —

Scanned with CamScanner

Date.....

MAIN MEMORY

↳ Main memory is where programs & data are kept when the processor is actively using them. When programs & data become active, they are copied from secondary memory into main memory where the processor can interact with them. A copy remains in secondary memory.

↳ Main memory is intimately connected to the processor, so moving instructions & data into & out of the processor is very fast.

Types of Main Memory :

1.) RAM (Random Access Memory) :

↳ It is volatile in nature i.e it is used to store the programs & data being used by the CPU in real-time.

↳ The data on the random-access memory can be read, written & erased any number of times.

↳ Writing data is faster.

2.) ROM (Read Only Memory) :

↳ It is a type of memory where the data has been pre-recorded.

↳ It is Non-volatile in nature.

Scanned with CamScanner

Date.....

↳ Data stored in ROM can be retained even after the computer is turned off.
↳ It is used only for read.

Types of RAM :-

1. SRAM (Static - RAM) :-

↳ Called as static RAM, because data is retained (does not change) till power is 'ON'.
↳ Uses Flip - Flop as memory cell.
↳ Flip - Flop Stores '0' & '1'.
↳ Here, size of memory cell is larger
↳ It is costlier in comparison to DRAM.

2. DRAM (Dynamic - RAM) :-

↳ called as dynamic RAM, because data can be changed during the Power 'ON'.
↳ It uses capacitor as memory cell.
↳ Size of memory cell is smaller
↳ Presence of Electric charge on capacitor is shown by '1', where as the absence of electric charge in the capacitor is shown by '0'.
↳ Slower & cheaper.

Scanned with CamScanner

Date.....

TYPES OF ROM :-

1. PROM :-

- Known as "Programmable ROM".
- One time programmable by user.
- Here data is written after the memory chip has been created.
- Data once written cannot be erased.

2. EPROM :-

- Known as "Erasable PROM"
- Here, multitime programmable is possible by the user.
- Data once written can be erased by UV rays.
- More time is required in re-programming.

3. EEPROM :-

- Known as "Electrical EPROM"
- Data once written can be erased by using electrical signals/ field electron emission.
- Less amount of time is required in re-programming.

4. FLASH MEMORY :-

- It is a non-volatile memory
- It is used for storage & for transferring data between a PC & digital devices.

Spiral

Scanned with CamScanner

Date.....

↳ It has the ability to electrically re-programmed & erased.

↳ Eg :- USB flash drives, MP3 players, Digital cameras etc.

—— o ——

MEMORY – CHIP ORGANIZATION

Scanned with CamScanner

Date.....

* Example on Memory address map for RAM, ROM :-

Question: A computer employs RAM chips of 256 X 8 & ROM chips of 1024 X 8. The computer system needs 2K bytes of RAM, 4K Bytes of ROM & 4 interface units, each with 4 Registers. A memory mapped I/O Configuration is used. The 2 highest order bits of the address bus are assigned as 00 for RAM, 01 for ROM & 10 for Interface Registers.

(i) How many RAM & ROM chips are needed ?
(ii) Draw a memory address map for the system ?
(iii) Give the address range in hexadecimal for RAM, ROM & I/O Interface.

Ans :- For RAM chips:
256 X 8 i.e 256 memory locations & 8 bit assigned for each memory locations.
lly for 1024 X 8 for ROM chips.

Given parts

Also given :- 2K bytes = RAM
4K bytes = ROM } requirements
4 units / 4 registers required

4 units means 4 I/O devices which are connected with the help of Interface units.

∴ 4 units / 4 registers = 16 registers are required.

& 2 highest order bits of the address bus are assigned as :- 00 → RAM ; 01 → ROM &
10 → Interface Registers.

Spiral

Date.....

Sol^n (i) : To find :- No. of RAM & ROM chips.

Formula for :-

$$\boxed{\text{No. of RAM chips} = \frac{\text{Total RAM size}}{\text{1 RAM size}}}$$

$$\boxed{\text{No. of ROM chips} = \frac{\text{Total ROM size}}{\text{1 ROM size}}}$$

$\therefore$ No of RAM chips $= \dfrac{\text{Total RAM size}}{\text{1 RAM size}}$

$$= \frac{2K \text{ bytes}}{256 \times 8}$$

$\boxed{1K = 1024.}$

$$= \frac{2K \times \cancel{8} \text{ bits}}{256 \times \cancel{8}}$$

$$= \frac{2K}{256}$$

$$= \frac{2 \times 1024}{2^8} = \frac{2 \times 2^{10}}{2^8}$$

$$= 2^{n}/2^{8} = 2^3$$

$= 8$ RAM chips.

& lly No. of ROM chips $= \dfrac{\text{Total ROM size}}{\text{1 ROM size}}$

$$= \frac{4K \text{ Bytes}}{1024 \times 8}$$

$$= \frac{4K \times \cancel{8}}{1024 \times \cancel{8}}$$

Date.....

$$= \frac{4K}{1024}$$

$$= \frac{4 \times \cancel{1024}}{\cancel{1024}}$$

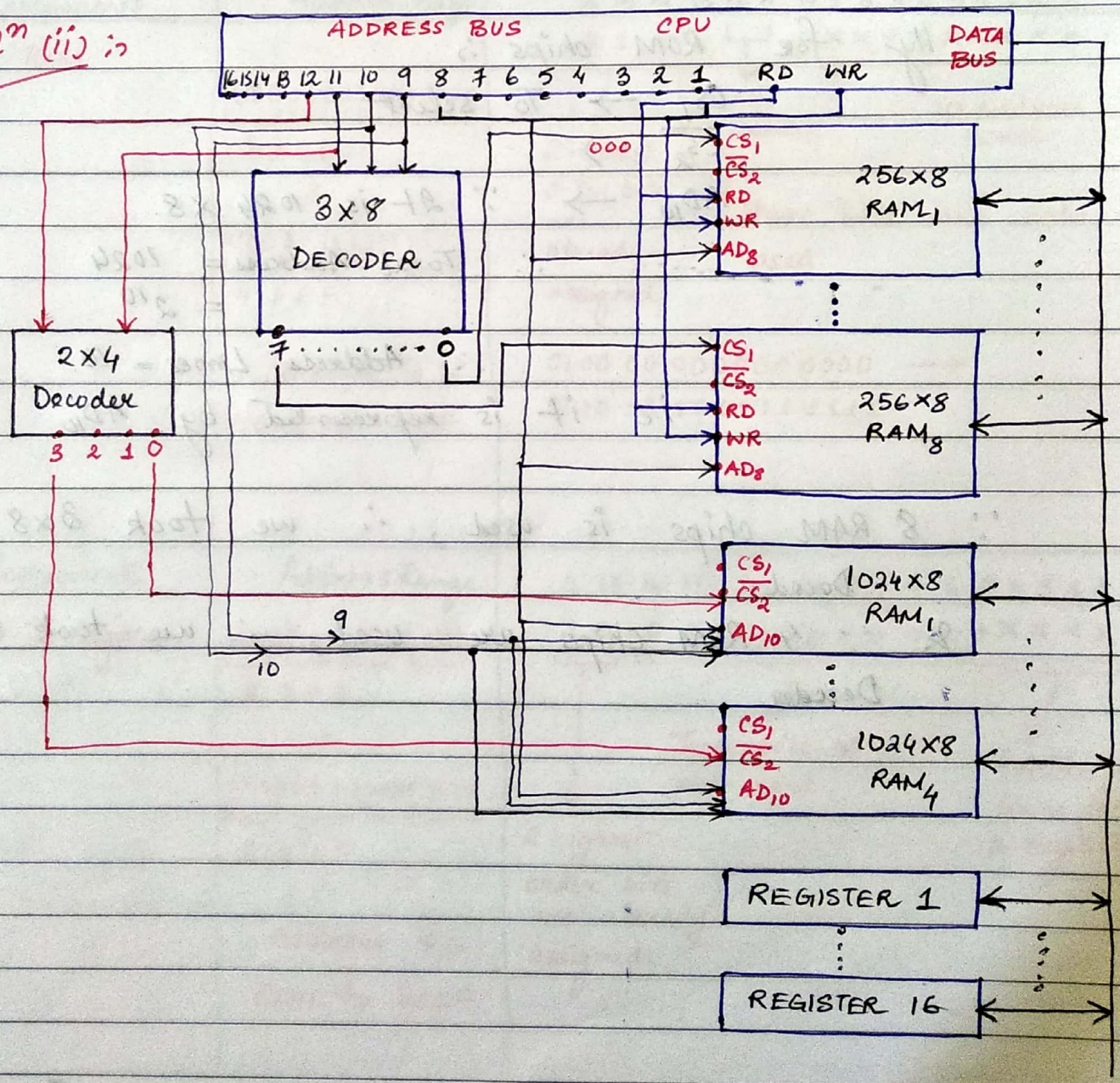
$= 4$ ROM chips

& No. of I/O Interface Registers

$= 4 \times 4$

$= 16$ Registers.

Solⁿ (ii) :-



Spiral

Date.....

Here,

$CS_1 \rightarrow$ To select RAM chips.

$\overline{CS_2} \rightarrow$ To select ROM chips

$RD \rightarrow$ To read RAM. chips

$WR \rightarrow$ To write control in RAM chips

$AD_8 \rightarrow$ $\because$ 256 × 8 i.e 256 locations having 8 bit of data.

$\because$ Total Address = 256

$= 2^8$

$\therefore$ Address Lines = 8.

$\therefore$ It is represented by AD_8.

Ily for ROM chips :

$CS_1 \rightarrow$ To select

$\overline{CS_2} \rightarrow$

$AD_{10} \rightarrow$ $\because$ It is 1024 × 8.

$\therefore$ Total Address = 1024

$= 2^{10}$

$\therefore$ Address Lines = 10.

i.e it is represented by AD_{10}

$\because$ 8 RAM chips is used, $\therefore$ we took 3×8 Decoder.

& $\because$ 4 ROM chips are used, $\therefore$ we took 2×4 Decoder.

Scanned with CamScanner

Sol^n (iii)

Date.....

Now let us perform memory address map :-

Components	Address Range	16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
RAM	00000000 00000000 0 0 0 0 to 00000111 11111111 0 7 F F ∴ Address is :- 0000 to 07FF	00 000 ___ XXXXXXXX 2 highest order bits are already assigned. 3x8 Decoder 8 Address Lines This bits are not used. ∴ 00 00000000000000 → 00000111 11111111
Components	Address Range	16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
ROM	0100 0000 0000 0000 4 0 0 0 to 0100 0111 1111 1111 4 F F F ∴ Address is :- 4000 to 4FFF	01 00 ___ XXXXXXXXXX 2x4 Decoder 10 Address lines 2 highest order bits are already assigned These bits are not used. ∴ 0100 00 0000000000 → 0100 11 1111111111
Components	Address Range	16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
I/O Interface	1000 0000 0000 0000 8 0 0 0 to 1000 0000 0000 1111 8 0 0 F ∴ Address is :- 8000 to 800F	10 0000000000 XXXX These bits are not used 4 Address lines for 16 Registers 2 highest order bits are already assigned.

Spiral

Date.....

Numerical :> A computer system employs 128 × 8 of RAM chips, 512 × 8 of ROM chips. The system needs a total of 512 bytes of RAM & 512 bytes of ROM. Find the no. of RAM, ROM chips & draw memory address map.

Solⁿ: (i) No of RAM chips = $\frac{\text{Total RAM size}}{\text{One RAM size}}$

$= \frac{512 \text{ bytes}}{128 \times 8}$

$= \frac{{}^{4}\cancel{512} \times \cancel{8}}{\cancel{128} \times \cancel{8}}$

= 4 RAM chips

(ii) No of ROM chips = $\frac{\text{Total ROM size}}{\text{One ROM size}}$

$= \frac{512 \text{ bytes}}{512 \times 8}$

$= \frac{512 \times \cancel{8}}{512 \times \cancel{8}}$

= 1 ROM chip.

(iii) Memory Address Map :>

↳ It specifies the pictorial way to represent address range of various RAM & ROM chips.

∴ The required memory address map can be done by representing into hexadecimal form.

Scanned with CamScanner

Date.....

CHIPS	ADDRESS RANGE [Start Address → Last Address]	ADDRESS LINES.
		16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		0 ⌊⌋ × × × × × × ×
		2x4 Decoder
RAM_1	0000H to 007FH	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
		to
		0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1
		16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		0
RAM_2	0080H to 00FFH	0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
		to
		0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
		16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		0
RAM_3	0100H to 017FH	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
		to
		0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1
		16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		0
RAM_4	0180H to 01FFH	0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
		to
		0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1
		16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		0 0 0 0 0 0 1 × × × × × × × ×
ROM_1	0200H to 03FFH	0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
		to
		0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1
∵ 4 RAM CHIPS.		
∴ 2x4 Decoder used.	RAM: 128 X 8	
∵ 1 ROM CHIP.	= 128 = 2^7	
∴ No Decoder Req.	= Address Lines = 7	

Note: Here, Bit possition of the Address Line 10 is used for detecting whether it is RAM or ROM.

If Bit position (10) = 0 then it is RAM &
If " " (") = 1 " " " ROM.

Spiral

Scanned with CamScanner

Date.....

Numerical 3.

a) How many 128 × 8 RAM chips are needed to provide a memory capacity of 2048 bytes.

b) How many lines of 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 decoder?

Ans: a) No. of RAM chips $= \dfrac{\text{Total RAM size}}{\text{1 RAM size}}$

$$= \frac{2048 \text{ bytes}}{128 \times 8}$$

$$= \frac{2048 \times \cancel{8}}{128 \times \cancel{8}}$$

$$= \frac{2^{11}}{2^{7}} = 2^{4} = 16 \text{ Chips.}$$

b) No. of address lines to access 2048 bytes of memory $= 2048$ bytes

$= 2048 \times 8$

where 2048×8 represents 2048 address lines & 8 data lines.

$$= 2^{11} \times 8$$

$\therefore$ No. of address lines $= 11$.

Spiral

Scanned with CamScanner

Date.....

$\therefore$ No. of common address lines
= No. of address lines in 1 chip
$= 128 \times 8$
$= 2^7 \times 8$

$\therefore$ 7, common address lines are needed.

c) To find :- Decoder size & also No. of decoded lines.

$\because$ Number of RAM chips = 16
$\therefore$ Decoder required = 4×16, which is the decoder size.

$\because$ Decoder size = 4×16.

$\therefore$ No. of decoded lines = 4.

———— o ————

Scanned with CamScanner

Date.....

"CACHE MEMORY MAPPING"

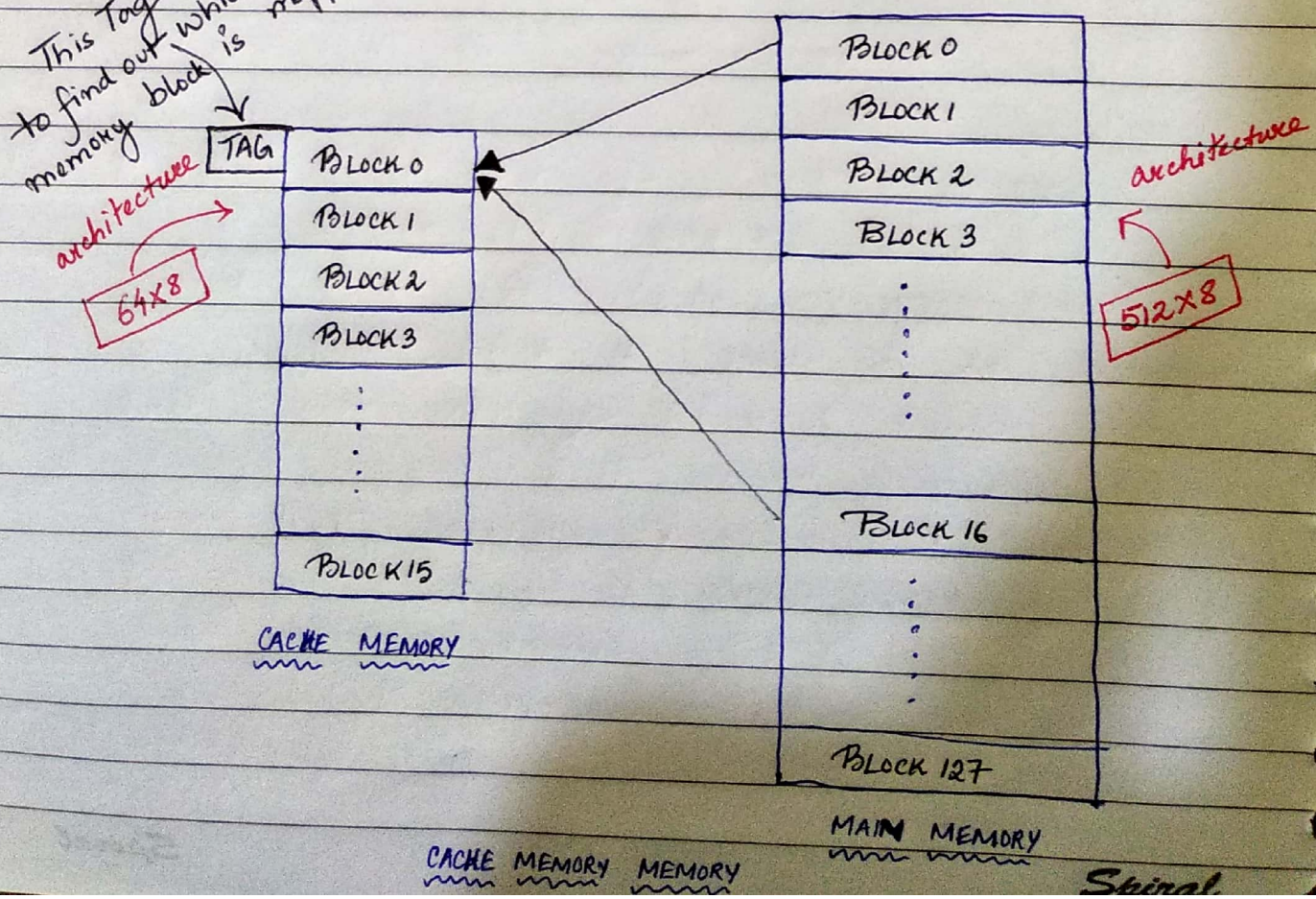
↳ In Cache Memory, mapping is done, which explains how data is copied (mapped) from main memory to cache memory.

There are few techniques with the help of which cache memory mapping can be done. The techniques are :-

1. DIRECT MAPPING :-

↳ In this mapping, main memory blocks are copied to a fixed block of cache memory, but one at a time.

Let cache memory block no = C_b
main memory block no = m_b
No. of block in cache = c
No. of block in main memory = m



CACHE MEMORY MEMORY

Date.....

Formula for finding cache memory block number :-

$$\begin{bmatrix}\text{CACHE MEMORY} \\ \text{BLOCK No.}\end{bmatrix} = \begin{bmatrix}\text{MAIN MEMORY} \\ \text{BLOCK No.}\end{bmatrix} \text{ mod } \begin{bmatrix}\text{No. of} \\ \text{Cache Blocks}\end{bmatrix}$$

For example :-

Main memory Size = 512 x 8
Cache memory Size = 64 x 8
& Block size = 4 words

$$\text{No of Main memory Blocks} = \frac{\text{Total Main Memory Words}}{\text{Block Size}}$$

$$= \frac{512}{4} = 128 \text{ Blocks}$$

$$\& \text{ No. of Cache Memory Blocks} = \frac{\text{Total Cache Memory Words}}{\text{Block Size}}$$

$$= \frac{64}{4} = 16 \text{ Blocks}$$

Note: "Block means a group of words."

∴ No of main memory blocks = 128 Blocks & No of cache memory block = 16 Blocks are shown in the figure above.

Now, Let us understand the mapping process :-
The mapping process can be mapped with the above formula like :-

Spiral

Scanned with CamScanner

Date.....

Cache memory Block No :-

C_b = 0 mod 16 = 0 , → Cache memory block.
16 mod 16 = 0 ,
32 mod 16 = 0 ,
48 mod 16 = 0 ,
64 mod 16 = 0 ,
⋮

main memory block

i.e Main memory block no 0, 16, 32, 48, 64, 80, etc will be mapped with 0 cache memory block.

That is, Remainder 0 will be mapped with Block 0
" 1 " " " " " 1
⋮
" 15 " " " " " 15.

based on the above formula.

Point to note :- In case of Direct mapping, if suppose Main memory block 0 is mapped with Cache memory block 0 & suppose block 16 needs to be mapped with cache memory block 0 which are already mapped with main memory block 0, then that mapping process will overlap the new mapping of main memory block 16 with cache memory block 0.

But mapping is done one at a time.

Scanned with CamScanner

Date.....

In order to find :-

Number of main memory block to be mapped with number of cache memory block, but one at a time, can be calculated with the following formula :-

$$\text{No. of main memory blocks in 1 block of Cache} = \frac{\text{No. of main memory blocks}}{\text{No. of cache memory blocks}}$$

∴ If we consider the above example :-

No of main memory blocks in 1 block of cache, one at a time $= \frac{128}{16} = 8.$

This is one possible question!

Another question may arise ?

In the above example, Main memory size = 512 × 8 i.e 512 words.

∴ Main memory address bit $= 512 = 2^9$
$= 9$ bits.

This is another question!

Scanned with CamScanner

Date.....

Q: How to find the Main memory address division?

Ans: Three parts:

1. Tag bits
2. Block bits
3. Word bits

represented as:

Tag bits	Block bits	Word bits

← index bits. →

Fig: Format for Main Memory Address.

e.g.: Let Block size = 4 words.

$= 2^2$

$\therefore$ $\boxed{\text{Word bits} = 2 \text{ bits}}$

If Block size = 16 words

$\therefore$ Word bits $= 2^4 = 4$ bits etc.

As discussed in previous example

$\because$ Number of blocks in Cache Memory = 16 blocks

$\therefore$ Block bits $= 16 = 2^4$

$\therefore$ $\boxed{\text{Block bits} = 4 \text{ bits}}$

& Lastly,

Tag bits = No of main memory blocks in 1 block of Cache.

For our previous example:

No of main memory blocks in 1 block of cache = 8

Spiral

Date.....

$\therefore$ $\boxed{\text{Tag bits} = 8 = 2^3 = 3 \text{ bits}}$

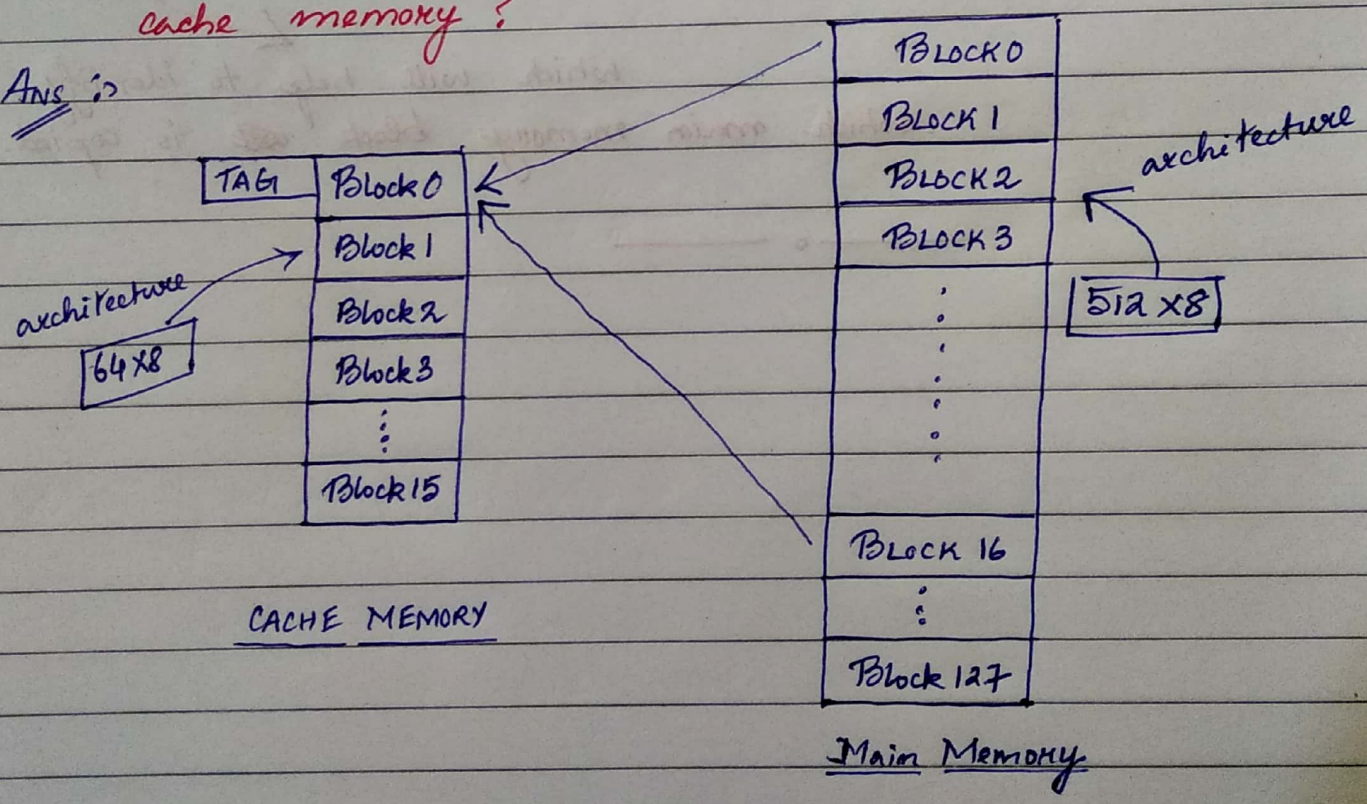
$\therefore$ Finally, the Main memory address can be represented as :-

3 bits	4 bits	2 bits

$\longleftarrow$ 6 bits $\longrightarrow$

Q:- How to find the size the word size of the cache memory ?

Ans :-



Here, No. of Main Memory blocks = 128 blocks
& No of Cache Memory blocks = 16 blocks.

$\therefore$ No. of main memory blocks present in 1 block of cache $= \frac{128}{16} = 8$.

Spiral

Date.....

$\therefore$ Tag size = $8 = 2^3 = 3$ bits.

Now, in order to find the Cache memory word length = [3 bits (Tag bits)] + [8 bits]

= 11 bits.

$\because$ 8 bits $\Rightarrow$ 000 to 111 } Tags

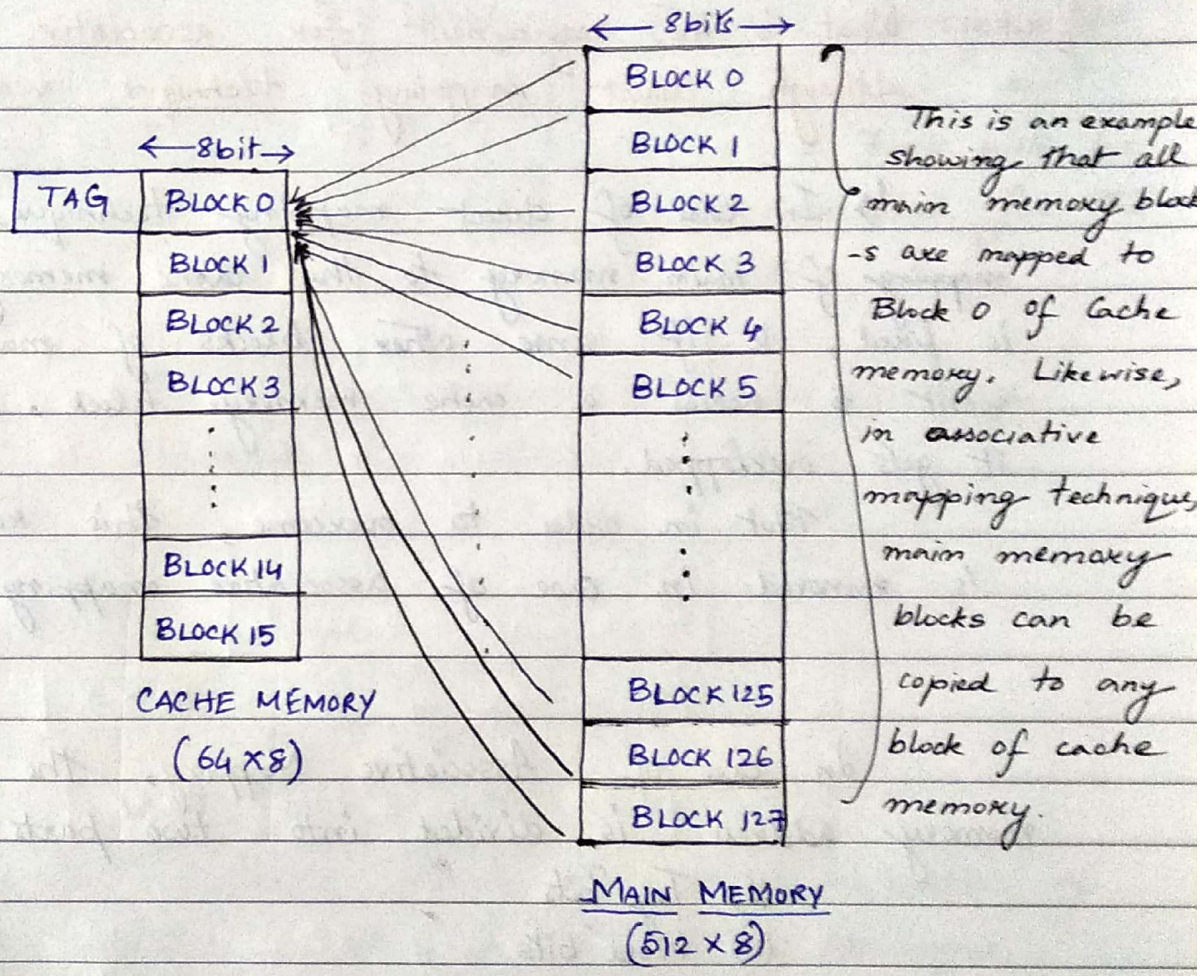
which will help to identify which main memory block is copied.

Date

2.) ASSOCIATIVE MAPPING :-

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

In order to understand the working of associative mapping, let us consider an example.



Let main memory size = 512 x 8 , where
512 = Words & 8 = Word size

lly cache memory size = 64 x 8 , where
64 = Words & 8 = word size

& Block size = 4 words

Date.....

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

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

& No. of main memory blocks in 1 block of cache $= 128 = 2^7$

Question: What is the requirement for associative mapping although direct mapping technique was there?

↳ In case of direct mapping technique, the mapping of main memory to the cache memory block is fixed, & if some other blocks of main memory want to access a cache memory block, then it gets overlapped.

But in order to overcome, this restriction is removed in case of associative mapping.

In case of Associative Mapping, the main memory address is divided into two parts :-

(i) Tag bits
(ii) Word bits.

Tag generally defines how many main memory block can map in 1 block of cache memory.

7 bits	2 bits
Tag bits	Word bits

← 9 bits →

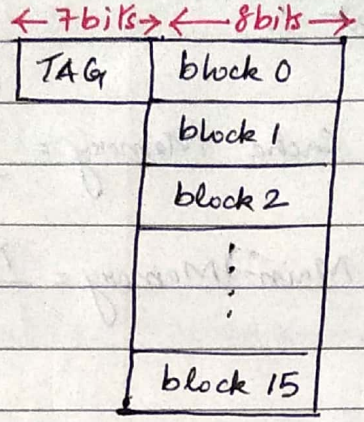
Spiral

Scanned with CamScanner

Date.....

∵ Block size = 4 words = 2^2 ∴ Word bits = 2 bits.

Question: What is the memory size of the Cache Memory address?



∵ No of Main memory blocks in 1 block of cache memory = 128
$= 2^7$
= 7bits will be associated for the TAG.

∴ Memory size of the cache memory address = 7bits + 8 bits
= 15 bits.

———— o ————

3.) SET-ASSOCIATIVE MAPPING :-

↳ This mapping technique is a combination of direct mapping & associative mapping techniques.

↳ Here, cache memory is divided into SET's & Blocks,

Sets = group of blocks
Blocks = group of words.

Scanned with CamScanner

Date.....

Now, Let us understand the set-associative memory mapping technique with the help of the following formula's :-

∵ We know that :-

$$\text{No. of blocks in Cache Memory} = \frac{\text{No. of words in Cache memory}}{\text{Block Size}}$$

& lly, $$\text{No. of blocks in Main Memory} = \frac{\text{No. of words in Main Mem.}}{\text{Block Size}}$$

∴ Then, Let us assume that,

$\boxed{\text{Set size} = 2}$ (ie known as Two-way set associative)

& if set-size = 4 then it is termed as 4-way set associative.

$\boxed{\text{Block-size} = 4 \text{ words}}$

$\boxed{\text{Main Memory Size} = 512 \times 8}$

& $\boxed{\text{Cache Memory Size} = 64 \times 8}$

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

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

& No. of sets in cache memory $= \frac{16}{2} = \boxed{8 \text{ sets}}$

Scanned with CamScanner

Date.....

How to find the cache memory set no :-

Formula :-

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

e.g :-

Cache Memory Set No. 0 :-

$\Rightarrow$ 0 mod 8 = 0
$\Rightarrow$ 8 mod 8 = 0
$\Rightarrow$ 16 mod 8 = 0
$\Rightarrow$ 24 mod 8 = 0

Ily Cache Memory Set No. 1 :-

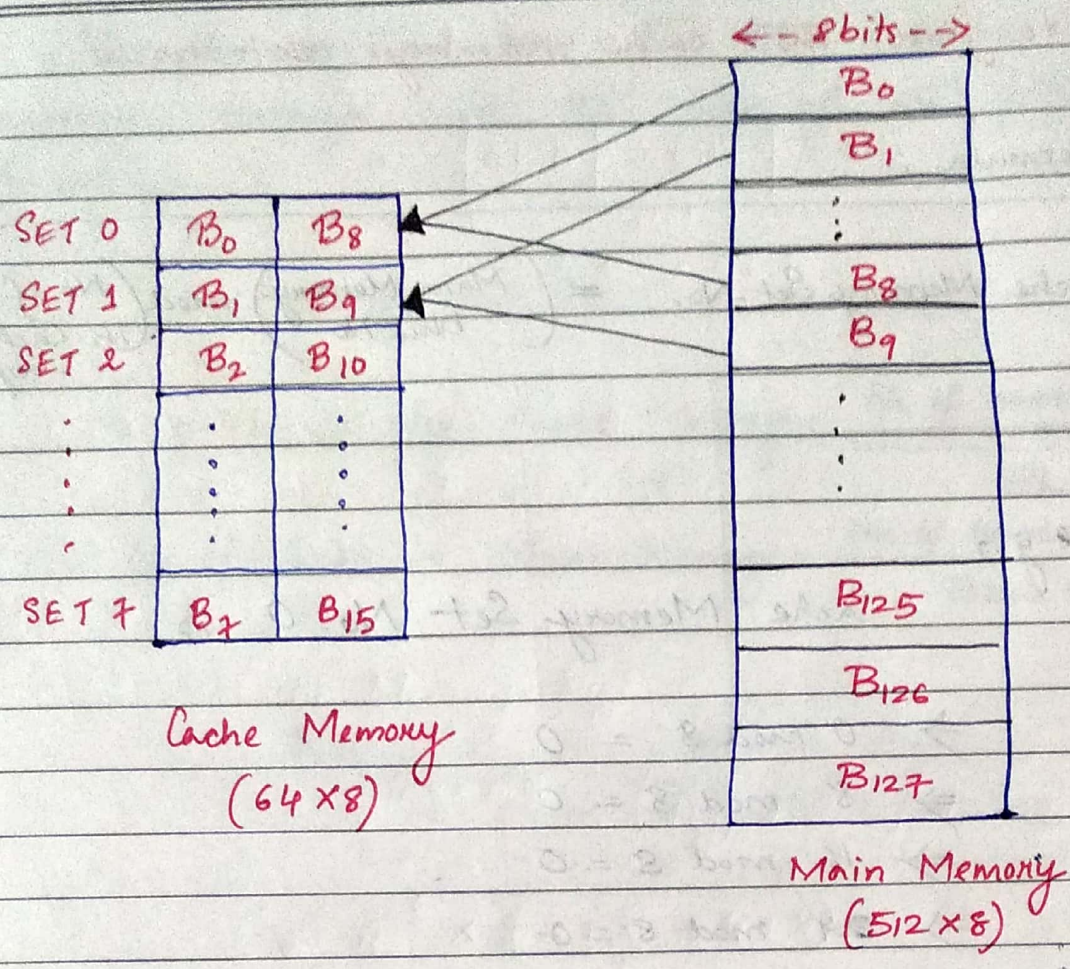
$\Rightarrow$ 1 mod 8 = 1
$\Rightarrow$ 9 mod 8 = 1
$\Rightarrow$ 17 mod 8 = 1
$\Rightarrow$ 24 mod 8 = 1

Now let us understand this above ⇑ mapping with the help of a diagram :-

{P.T.O}

Scanned with CamScanner

Date.....



SET- ASSOCIATIVE MAPPING PROCESS

Question : How to find out the number of main memory blocks in 1 set of cache (twice a time) ?

Sol^n: No. of main memory blocks in 1 set of cache (2 at a time)

$$= \frac{\text{Main Memory Blocks}}{\text{Cache Memory Sets}}$$

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

$$= 16.$$

Scanned with CamScanner

Date.....

Question :- In case of set-associative mapping what is the main memory address bits ?

Solⁿ:

In case of set-associative, the main memory address is divided into three parts :-

(i) tag bits
(ii) set bits
(iii) word bits.

Address format

i.e

Tag bits	Set bits	Word bits

← index bits →

∴ According to our previous example :-

∵ Main Memory words = 512 = 2^9

∴ $\boxed{\text{Total Address bits} = 9 \text{ bits.}}$

∵ Block size = 4 words = 2^2 = 2 bits.

∴ $\boxed{\text{Block bits} = 2 \text{ bits}}$

∵ No of set in cache memory = $\frac{16}{2}$

= 8 sets

= 2^3

∴ $\boxed{\text{Set bits} = 3 \text{ bits}}$

& Finally,

∵ No. of main memory blocks in 1 set of cache (2 at a time)

= 16 = 2^4

∴ $\boxed{\text{Tag bits} = 4 \text{ bits}}$

Scanned with CamScanner

Date.....

$\therefore$ The required main memory address bit

$$= \boxed{4\text{ bits}}\boxed{3\text{ bits}}\boxed{2\text{ bits}}$$

$\longleftarrow$ 9 bits $\longrightarrow$

$\longleftarrow$ 5 bits $\longrightarrow$

Question : How to find the cache memory size in case of set-associative mapping technique?

Sol^n:

Cache Memory Size =

$$2^{\text{index bits}} \times (tag_1 + data_1 + tag_2 + data_2)$$

$$= 2^5 \times (4+8+4+8)$$

$$= 32 \times 24 \quad \text{Size} \quad \text{Ans}$$

—o—

Date.....

WRITE BACK & WRITE THROUGH IN CACHE MEMORY

↳ If the data is in the cache, it is called as READ or WRITE hit.

↳ If the data is not present in the cache, then a READ miss or WRITE miss occurs.

READ MISS :

↳ Block of words containing this requested word is transferred from the memory.

↳ After the Block is transferred, the desired word is forwarded to the processor.

↳ The desired word may also be forwarded to the processor as soon as it is transferred without waiting for the entire block to be transferred. This is called load-through or early-restart.

WRITE - MISS :

When CPU writes to cache, we may use one of the two policies :

a) WRITE THROUGH
b) WRITE BACK.

Spiral

Scanned with CamScanner

Date

WRITE THROUGH : [Also known as STORE THROUGH]

Every "write" updates both current & next level of cache to keep them in sync. (i.e coherant). Later when Block is evicted, no writeback is needed.

WRITE BACK :

Let the CPU keep writing to cache at fast rate, not updating the next level. Later when block is evicted, entire block is written back (because bookkeeping is kept on a per block basis).

Let us discuss an example :

Before	WRITE BACK	After	WRITE THROUGH	After
X = 100	CPU	X = X + 200	CPU	X = X+200
X = 100	CACHE MEMORY	X = 300	CACHE MEMORY	X = 300
X = 100	MAIN MEMORY	X = 100	MAIN MEMORY	X = 300

Here,

WRITE BACK's before value will be in Before & Write BACK's after value will be at After.

Ily,

WRITE THROUGH's before value will be at Before & WRITE THROUGH's after value will be at second After.

Scanned with CamScanner

Understanding WRITE BACK policy :-

Date.....

Let us assume that the value initialy is at Main Memory i.e $X = 100$, then this value is transferred to the Cache Memory & from Cache Memory this $X = 100$ will be transfered to the "Registers" present in the CPU.

After that operation is performed i.e $X = X + 200$ $\Rightarrow X = 300$ & it will be transfered to the Cache Memory.

And what WRITE BACK policy says that this new update value of X, i.e $X = 300$ will be updated in Cache Memory; but it will not be updated in Main Memory. i.e $X = 100$ will be there in Main Memory.

Now in near future whenever CPU will require $X = 300$, it will not access Main Memory, rather it will access Cache Memory.

Now when this variable, i.e $X = 300$ becomes useless in Cache Memory then it will be updated in the main memory. Then in main memory it will be now $X = 300$ in place of $X = 100$.

Spiral

Scanned with CamScanner

Date.....

WRITE BACK	WRITE THROUGH
1. Data is updated only in Cache Memory.	1. Data is updated both in Cache Memory & main memory.
2. Commonly used method	2. Unpopular method
3. No Data Redundancy	3. Data Redundancy
4. When a Cache block is not required then selected Cache block is written back to main memory to UPDATE the main memory.	4. When cache block is not required, there is no need to written back to main memory.
5. Simple method to implement	5. Complex method to implement.

Scanned with CamScanner

Date.....

AUXILIARY MEMORY

(also referred to as SECONDARY STORAGE)

It is the non-volatile memory with lowest-cost, having having highest-capacity & slowest access storage in a computer system. It is where programs & data kept for long-term storage or when not in immediate use.

Auxiliary memory also refer to as auxiliary storage, secondary storage, secondary memory, external storage or external memory.

Types :-

1. MAGNETIC DISKS

↳ A magnetic disk consists of one or more aluminium platters with a magnetisable coating.

↳ A disk head containing an induction coil floats just over the surface, resting on a cushion of air (except for floppy disks, where it touches the surface).

↳ When a current passes through the head, it magnetizes the surface beneath the head.

↳ When the head passes over a magnetized area a current is induced in the head,

Spiral

Date.....

making it possible to read back the previous stored bits.

↳ Thus the platter rotates under the head, a stream of bits can be written & later read back.

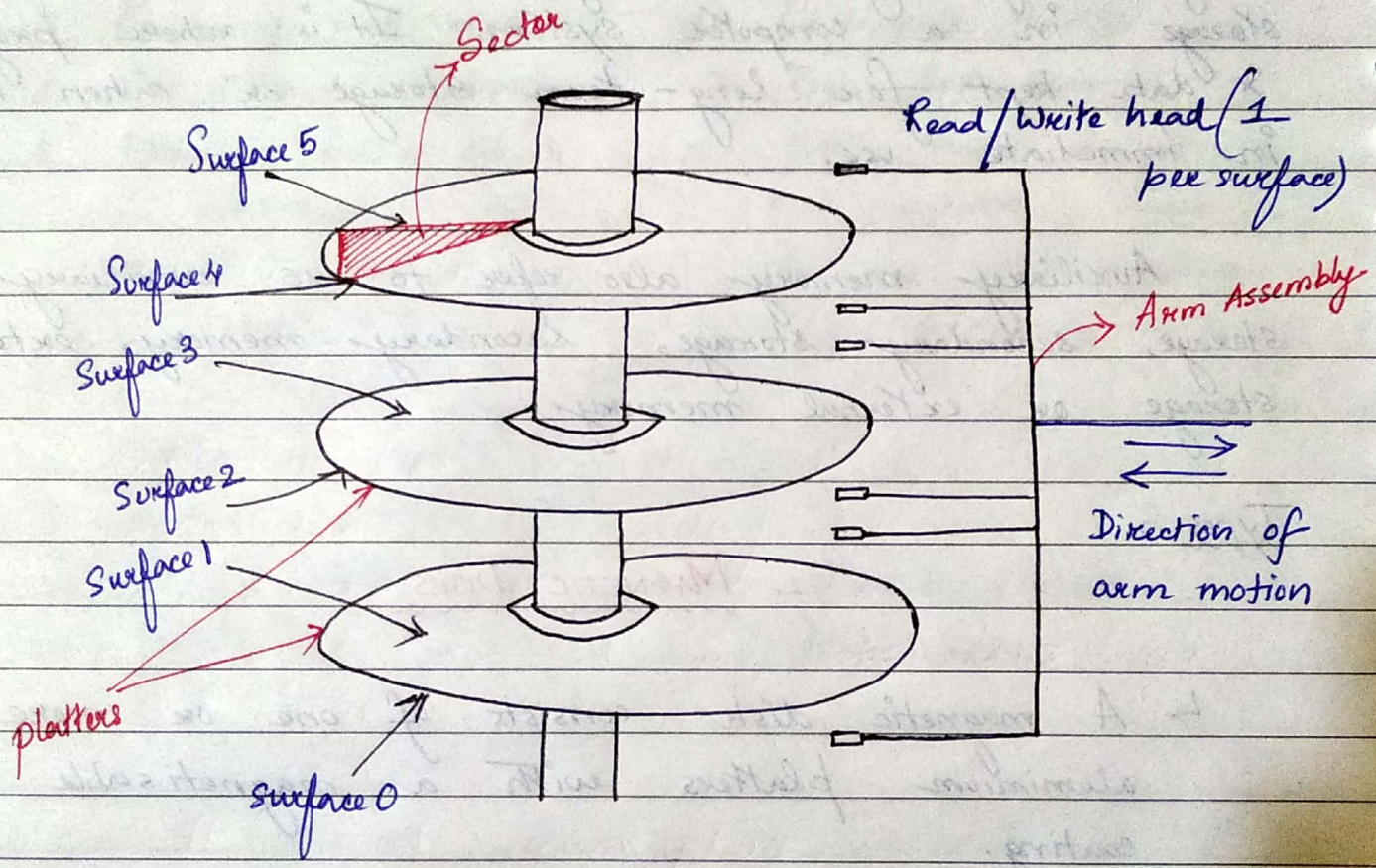


Fig: A Disk with 8 Platters

Components of a disk :

↳ Disk contains concentric tracks.

↳ Tracks are divided into sectors

↳ A sector is the smallest addressable unit in a disk.

Scanned with CamScanner

Date.....

↳ The arm assembly is moved in or out to position a head on a desired track. Tracks are under heads making a cylinder in shape.

↳ Only read/write head at any one time.
↳ BLOCK SIZE is a multiple of sector size (which is often fixed).

↳ DISK CONTROLLER :- It is typically embedded in the disk drive, which acts as an interface between the CPU & the disk hardware.
The controller has an internal cache (typically a number of MBs) that it uses to buffer data for read/write requests.

↳ Cylinder :- A cylinder is the set of tracks at a given radius of a disk pack.
↳ i.e a cylinder is the set of tracks that can be accessed without moving the disc arm.

↳ Estimating Capacities :-

1. TRACK CAPACITY = No. of sectors/track * bytes/sector.
2. CYLINDER CAPACITY = No. of tracks/No. of cylinder * track capacity.
3. Drive CAPACITY = No. of cylinders * cylinder capacity.
4. NUMBER OF CYLINDERS = No. of tracks on a surface.

Spiral

Scanned with CamScanner

Date.....

2. MAGNETIC TAPE

Magnetic Tape is a medium for magnetic recording, made of a thin, magnetizable coating on a long, narrow strip of plastic film.

Devices that record & playback audio & video using magnetic tape are called as TAPE RECORDERS & VIDEO TAPE RECORDERS.

A device that stores computer data on magnetic tape is known as a TAPE DRIVE.

Magnetic Tape revolutionized sound recording & reproduction & broadcasting. It allowed radio, which had always been broadcast live, to be recorded for later or repeated airing.

It allowed gramaphone records to be recorded in multiple parts, which were then mixed & edited with tolerable loss in quality. It was a key technology in early computer.

Construction :

↳ Magnetic tape is usually recorded on only one side.

↳ The opposite side is a substrate to give the tape strength & flexibility.

Scanned with CamScanner

Date.....

↳ The magnetic side of most tapes is magnetically manipulated by a "TAPE HEAD" to store the information.

↳ In all tape formats, a "TAPE DRIVER" uses motors to wind the tape from one reel to another, passing over tape heads to read, write or erase as it moves.

3. OPTICAL DISK

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

It is a computer storage disk that stores data digitally & uses "laser beams" that is being transmitted from a laser head mounted on an optical disk drive); to read & write data.

An optical disk

Scanned with CamScanner

Date.....

VIRTUAL MEMORY

Virtual Memory :-

↳ It is a concept that gives an illusion to the user that he has sufficient memory to execute any application / program of any size.

Few points to be noted :-

1. Virtual memory allows a number of applications having total size more than the main memory size, to run at the same time

2. Virtual memory is a simulated memory that is written to a file on the hard drive. This file is called a page file or swap file. It is used by Operating System to simulate physical RAM by using hard disk space.

3. In windows 1.0, 2.0 versions, there was no virtual memory; so we were not able to run a number of applications due to run out of RAM space. But, from Windows 3.0 onwards, concept of virtual memory was introduced.

Scanned with CamScanner

Date.....

In order to implement virtual memory, a portion of hard drive is made "reserve" by the system. This reserved portion can be either a "file" or a "seperate partition".

Foreg: In case of Linux OS, a seperate partition is used for virtual memory. And in case of WINDOWS OS, a file is used namely "pagefile.sys" which is used for Virtual memory.

⇓ Implementation of this file or seperate partition

So, when the system needs more memory, it maps some of its memory address out to the hard disk drive. This extra memory does not actually exist in RAM, it is the storage space on the disk drive.

Accessibility is done based on FIFO or Least Recently used

4. There is this concept that if your RAM size is more, then more faster your computer works. But, if there is lack of this RAM size, then it will slow your computer, therefore we can increase the speed by increasing the Virtual Memory.

Spiral

Scanned with CamScanner

Date.....

For example,

Let us take an example for Windows 10,

Formula → & let $\boxed{\text{initial virtual memory size} = 1.5 \times \text{RAM size}}$

If we want to add 6 GB of virtual memory,

then we have to use 4GB of RAM.

then,

Virtual Memory Size = 1.5 x 4 GB
= 6 GB.

$\boxed{\text{Formula, for maximum Virtual Memory Size} = 3 \times \text{Initial Size}}$

→ For this example,

Max^m Virtual Memory Size = 3 x 6 GB
= 18 GB.

(Used Hard Disk Space As a Virtual Memory)

Scanned with CamScanner

Practical Demonstration :

Date.....

- **How to change Virtual Memory Size :**

↳ Go to Control Panel, then System & Security, then System & then Advanced System Setting & then click Advance.

↳ Click on Setting Button in "PERFORMANCE SECTION"

↳ Click on Advance Tab in "PERFORMANCE OPTIONS"

↳ Click on Change Button of Virtual Memory Section

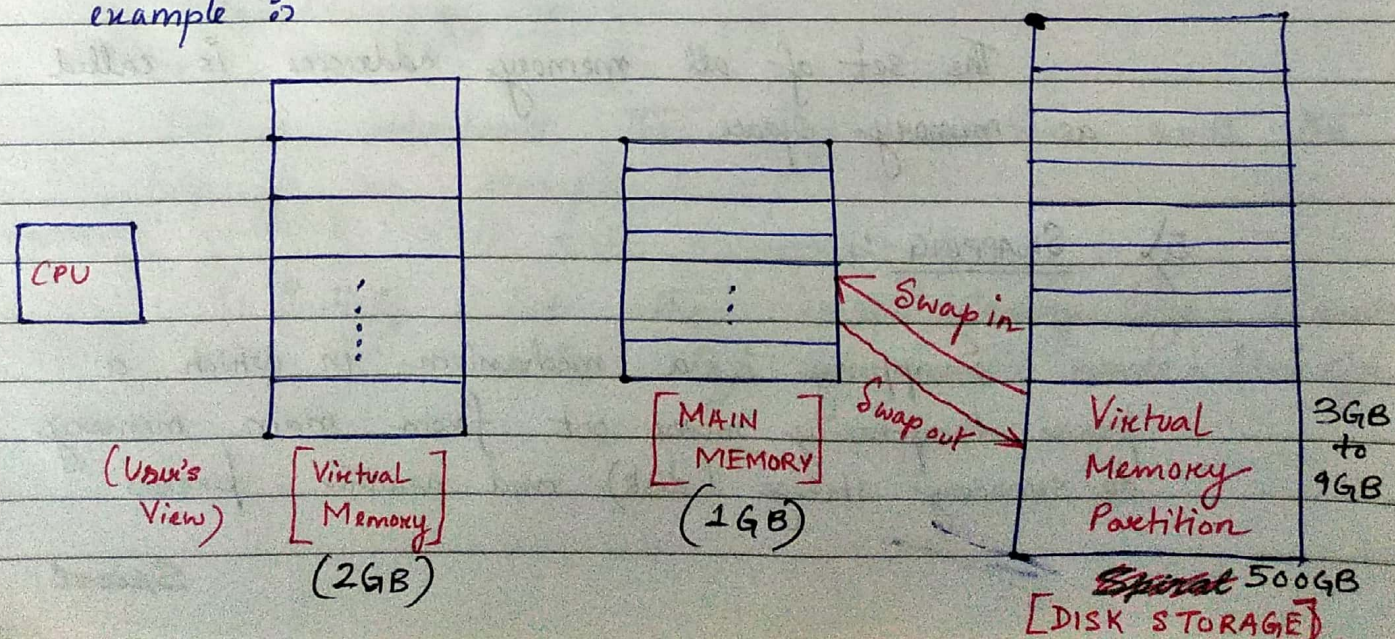
↳ Click on CUSTOM SIZE & set initial & maximum size.

—— • ——

IMPLEMENTATION OF VIRTUAL MEMORY

(By Using PAGING)

Let us understand the process with the help of an example :



Date.....

The above diagram generally explains the working of Virtual Memory.

Few terms must be known in order to understand the implementation :-

1.) VIRTUAL ADDRESS (LOGICAL ADDRESS) :-

Each address in Virtual Memory is called as Virtual Address.

2.) ADDRESS SPACE :-

Set of all virtual addresses is called as "address space."

3.) MEMORY ADDRESS (PHYSICAL ADDRESS) :-

Each address in main memory is termed as memory address.

4.) MEMORY SPACE :-

The set of all memory addresses is called as memory space.

5.) SWAPPING :-

Swapping is a mechanism in which a process temporarily moved out from main memory to secondary storage (disk) and another process

Scanned with CamScanner

Date.....

moved in from secondary storage (disk) to main memory. After some time, the first process again brought back to main memory.

6.) ADDRESS MAPPING :-

Address mapping specify how to convert virtual address to memory address.

The implementation of Virtual Memory ~~dapper~~ can be performed by two ways :-

(i) USING PAGING :- (ii) Using PAGE SEGMENTATION.

Address mapping using pages :-

Page :-

Virtual Memory is divided into equal number of groups of equal size. Each such group is called a "page".

Generally, virtual memory with its address space is a group of pages.

Each page consists of a number of words called "page size"

Let us understand the implementation with the help of an example :-

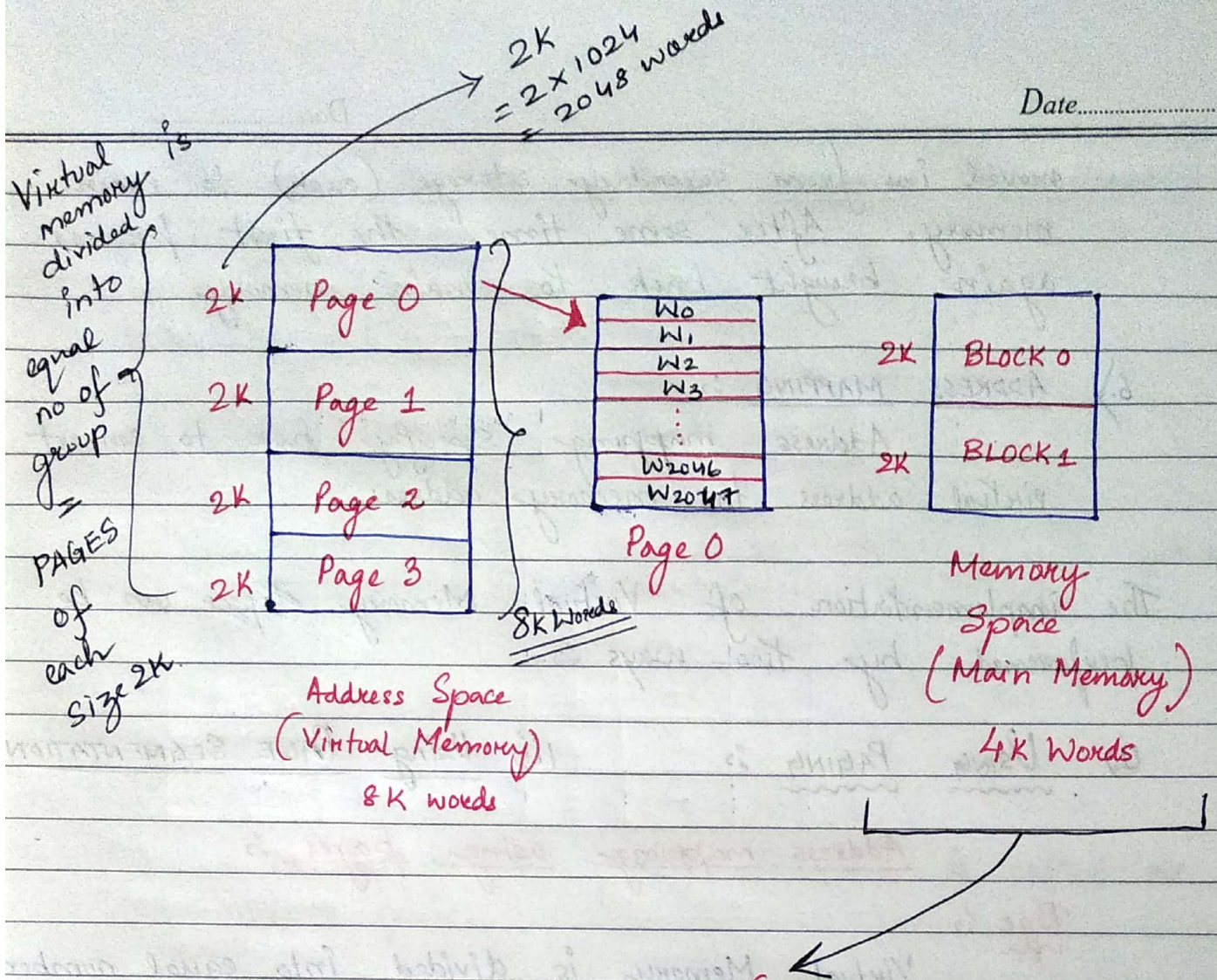
Initially Let the Virtual Memory Size = 8K words.
& Page Size = 2K words = block size.

i.e :-

Spiral

Scanned with CamScanner

Date.....



BLOCK (Frame) :-

Main memory is divided into equal size of groups. Each group is called BLOCK [FRAME].

Main memory is a group of frames (blocks) & each blocks (frames) consists of a number of words called BLOCK SIZE.

∴ In order to map Virtual & Main Memory :-

↳ **PAGE SIZE = BLOCK SIZE**

Spiral

Scanned with CamScanner

Date.....

Some Formula's

1. No. of pages = $\frac{\text{Address Space Size}}{\text{Page Size}}$

2. No. of blocks = $\frac{\text{Memory Space Size}}{\text{Block Size}}$

3. Virtual address bits = n
   then address Space = 2^n

4. Memory address bits = m bits
   then Memory Space = 2^m

—— o ——

Spiral

Scanned with CamScanner

Date.....

* EXAMPLE TO DEMONSTRATE ADDRESS MAPPING USING PAGING :-

Let Page size = 2K words = Block Size

$\therefore$ No. of pages = $\frac{8K}{2K}$ = 4

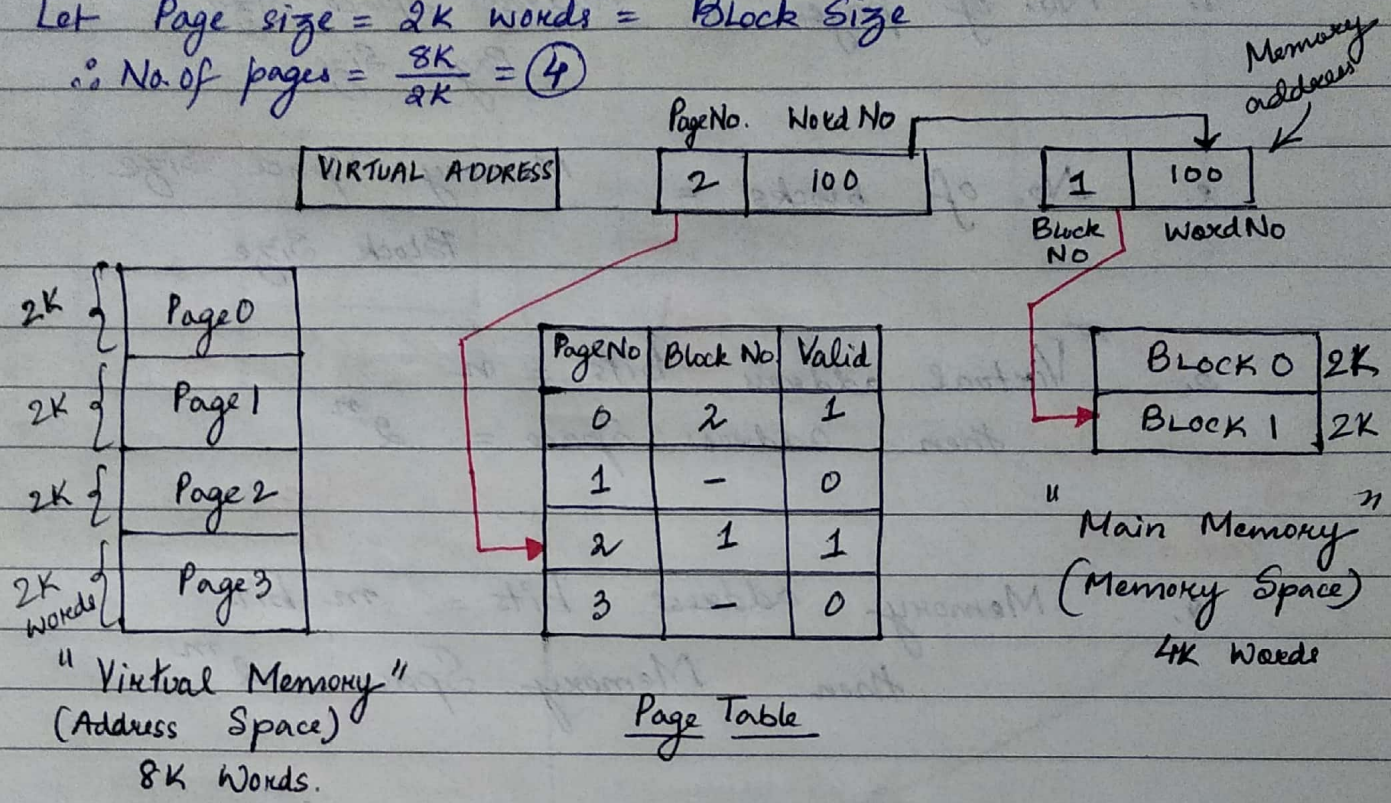


Fig: ADDRESS MAPPING (Virtual to Memory Address) USING PAGING.

ADDRESS MAPPING TO PAGE SEGMENTATION

SEGMENTED PAGING :-

↳ In this method, the virtual memory is first divided into segments, which are further divided into pages, & then pages are further divided into words.

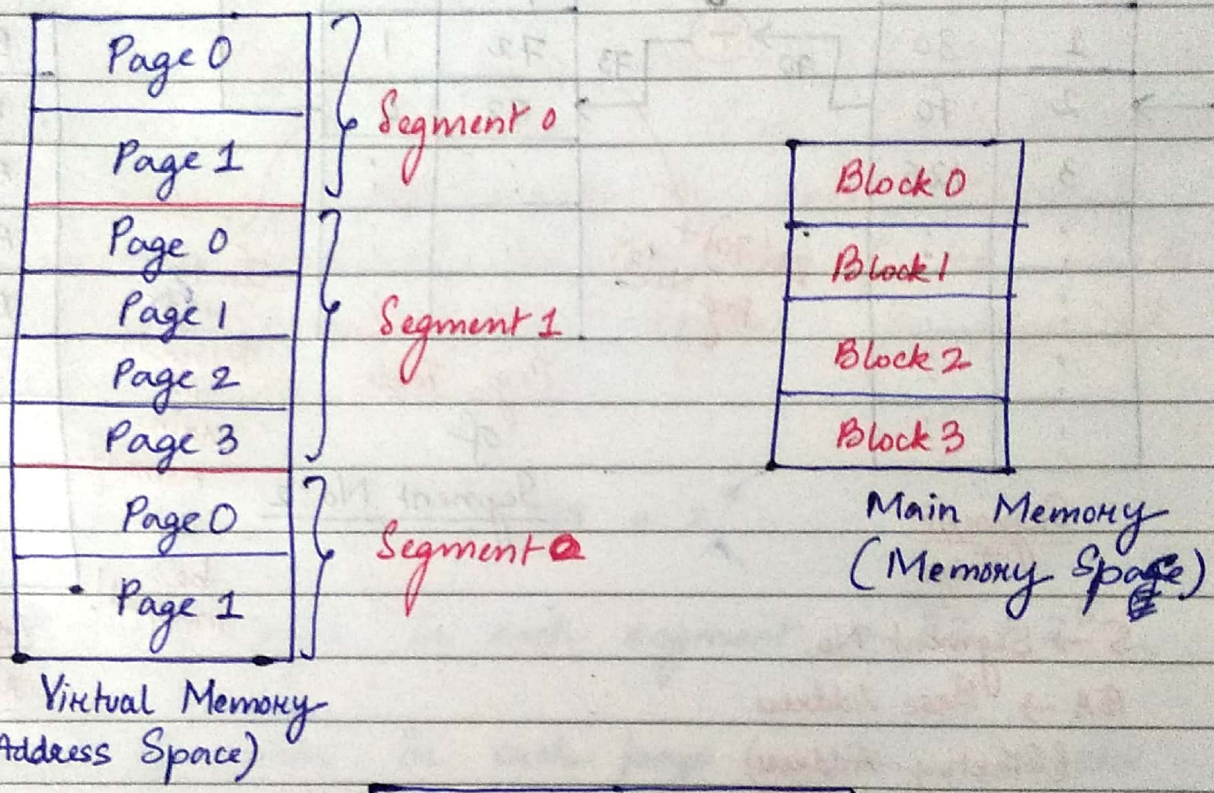
Spiral

Date.....

These Segments can be subroutine, array of data, table of symbols, Stack or can be main programs etc.

Each Segment is a Group of "pages."

& Each Page is a Group of "Words"



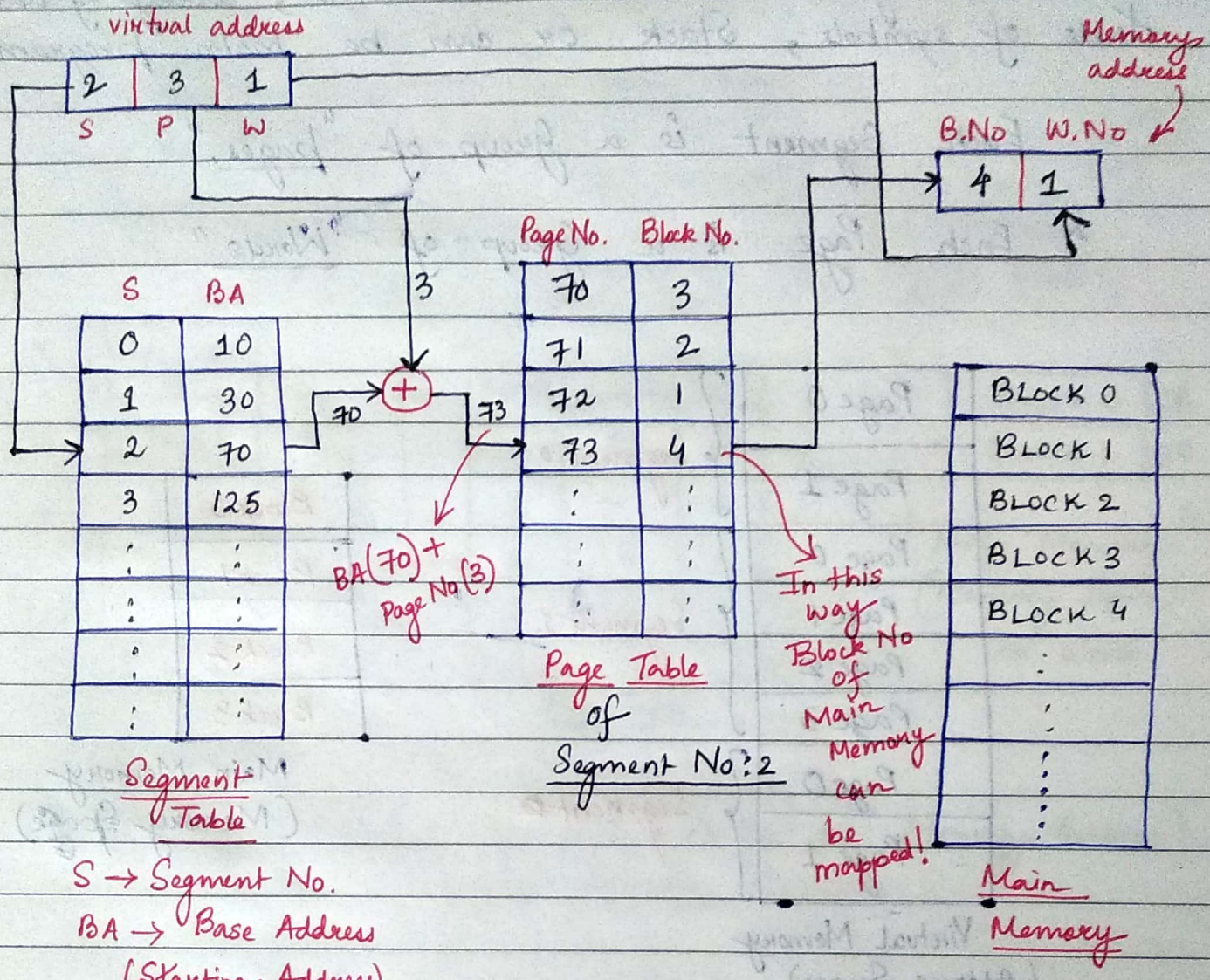
$\boxed{\text{Page Size} = \text{Block Size}}$

Example to understand the address mapping in Segmented paging is

{P.T.O}

S → Segment No.
P → Page No.
W → Word No.

Date.....



S → Segment No.
BA → Base Address (Starting Address)

∴ Virtual Address =

Segment No.	Page No.	Word No.

& Memory Address :-

← Block No →	Word No.

Spiral

Date.....

If Virtual address is asked, then it can find by:-

Segment No's → Page No's → Word No

will be the Virtual Address.

NUMERICAL :-

1. The Logical Address space in a Computer System consists of 128 Segments. Each Segment consists of 32 pages of 4K words in each. Physical memory consists of 4K blocks of 4K words in each. Formulate the Logical & physical address formats ?

Ans

No. of segments $= 128 = 2^7$

& No of pages in each segment $= 32 = 2^5$

No. of words in each page $= 4K = 4 \times 1024$
$= 2^2 \times 2^{10}$
$= 2^{12}$

Logical Address Format

Segment	Page	Word
7 bits	5 bits	12 bits

No. of blocks $= 4K = 4 \times 1024 = 2^2 \times 2^{10} = 2^{12}$

& No. of Words $= 4K = 4 \times 1024 = 2^2 \times 2^{10} = 2^{12}$.

Physical Address Format

Block	Word
12 bits	12 bits

Spiral

Scanned with CamScanner