Unit-V: Memory

⇒ It is a Storing device.

⇒ There are different types of memory and files which memory can store

memories are (Memory Hierarchy) Representation

i> Register
ii> Cache Memory
iii> primary memory (main memory)
iv> Secondary memory

Storage capacity increase from top - Bottom (Increased order).

Registes: Less Storage capacity in terms of KB's.

Buffer → set of Register.

Types of Register Ax, Bx, Cx, Dx .... 16 bits

Extend Accumulate /Base Register are EAx, EBx, ECx ... 32 bit capacity.

⇒ Storage capacity is in terms of Bytes.

Cache Memory:

L1 - Bytes
L2 - KB's.
L3 - MB's

⇒ To Access Data it take more time than Register

⇒ Storage capacity increases from top - Bottom

⇒ time increase from top - Bottom.

⇒ For Designing this memory we use SRAM (starting Random memory)

⇒ Recently Used Data is Available in Cache memory.

⇒ Loop Instructions /Recently Used functions are

Stored in Cache memory.

=> Recently Used information stored in cache memory.

=> Remove Blocks from cache memory, if Cache memory is Empty by Using Replacement Algorithm

=> Mapping Techniques – How it is Linked to the Primary memory

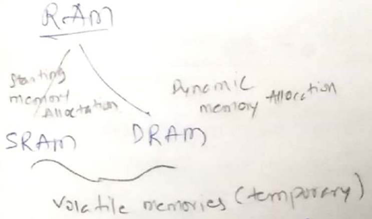
=> There are 3 mapping Techniques

Primary Memory: capacity is more than Cache memory.

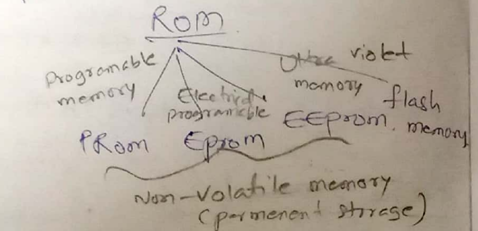
=> Storage capacity increase then time Access also increases

=> cache memory part of primary memory

=> primary memory 2types



Volatile memories (temporary)



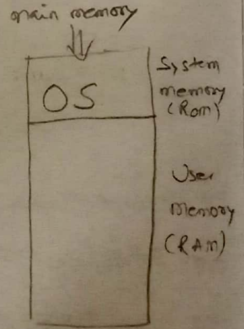
Non-Volatile memory (permenent strage)

Rom: Processor can Access only but can't write in Rom

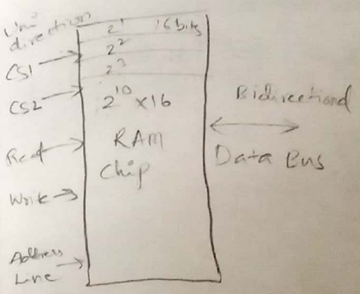
=> Non-Volatile Nature (can't be removed).

RAM processor can Access data from Any location.

=> Access in sequence of manner.



RAM Chip



cs – chip selection line

Total capacity : 2^{14} Bytes

SRAM

=> Used for cache memory Desigining

=> Data Can't Be erased After few milli Second.

=> Available for long period

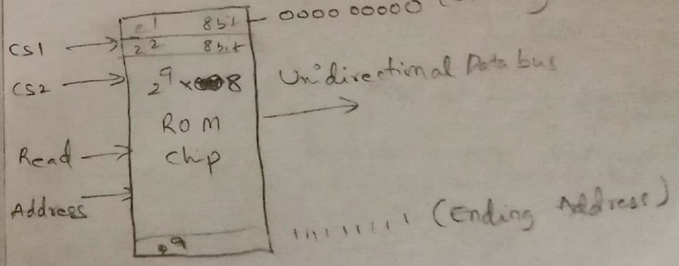
DRAM

=> Used for Desigining primary memory

=> Data can be erased Easily After few mill Second

=> Data Can't be Available for long period of time

Rom Chip



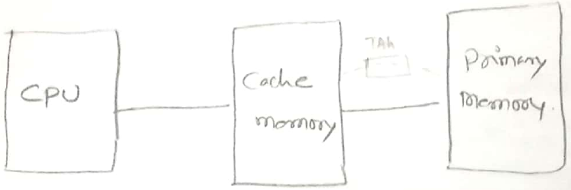
Total capacity : $2^9 \times 8$

=> only Read but can't write in ROM.

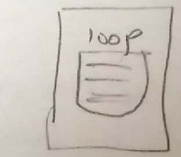
Secondary Memory: Storage [illegible]

time Access increases to Access Data

=> Storage in terms of tera byte. (2^{40} bytes)



Locality of reference: In some of the memory block Some of the Instructions are repeated like (loop Block).



Write Through method: Both memory (cache & main) Clock is updated for Executing those Instruction, processor refer write Through method

Dirty (or) modified bit: Before main memory ~~Tab~~ Tag bits are modified

Mapping Techniques

Three different mapping Techniques

i) Direct mapping Techniques

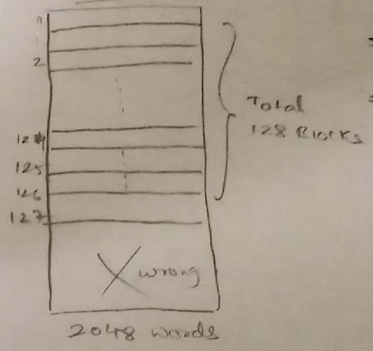
ii) Associative mapping Techniques

iii) Set Associative Mapping Techniques

Direct Mapping Techniques

Inside cache memory - 128 Blocks present

Cache (16 words)



=> Each Block size = 16 words.

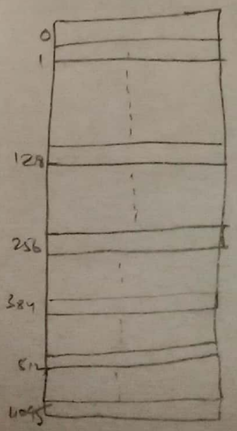
=> Cache Block & main memory Block equal size

∴ main memor size is

$$\frac{64k}{16} = 4k = 4 \times 1024 = 4096$$

Main memory

64k words



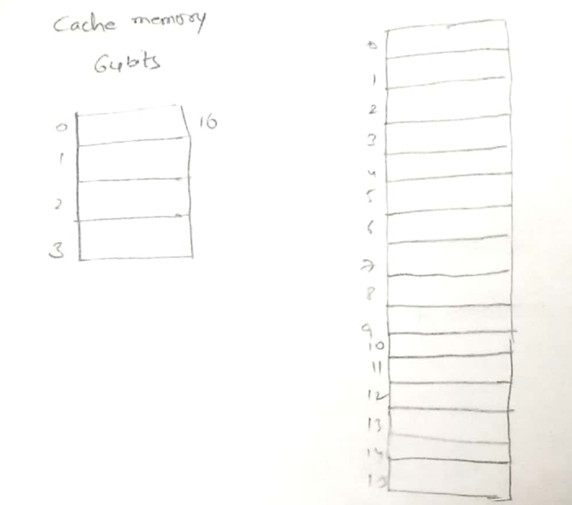
J modulo 128 └> cache memory Block

└> main memory Block No.

=> To identify m.m Block mapped with How main cache memory is

$$\frac{2^{12}}{2^7} = 2^5 = 32 \text{ bits.}$$

Direct ~~Associate~~ Mapping



8 bits

2	2	4
TAG	Block	word

TAG: How many Primary memory Block map with Each cache memory Block

$$= \frac{\text{Primary}}{\text{cache}} = \frac{2^4}{2^2} = 2^2 = \boxed{4}$$ Block map to Each cache memory

J mod 4 — No. of Blocks in Cachememory

↓

memory Block

Block No

Ex: 3 mod 4 = 3

2 mod 4 = 2

TAG values

3rd cache memory have

3 — 3, 7, 11, 15 Block of memory

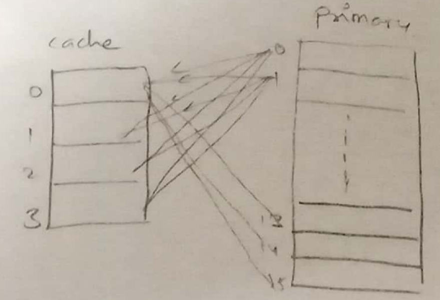
00 - 3
01 - 7
10 - 11
11 - 15

TAG	memory
00	0
01	4
10	8
11	12

Drawback

Associate Mapping

To overcome Drawback of Direct memory, Associative Block is used



→ Has more flexibility.

0 memory Block map with all cache memory Block

similarly All blocks are mapped with Every cache memory Block

Draw Back

To Identify a Block it take more time

i,e searching Block

See of Primary memory

Size = 16 × 16 = 256

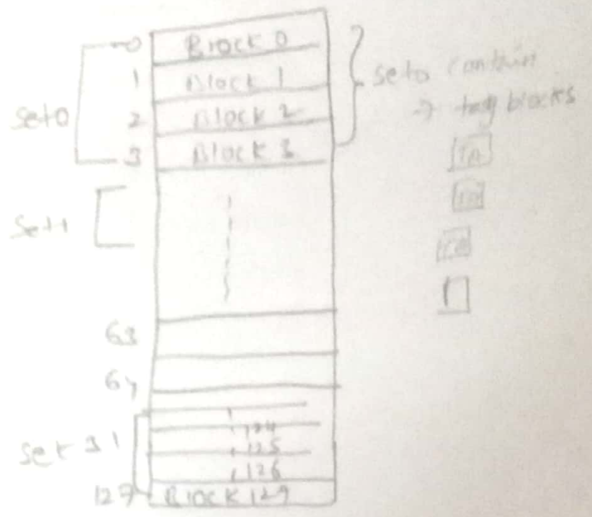
8 bits.

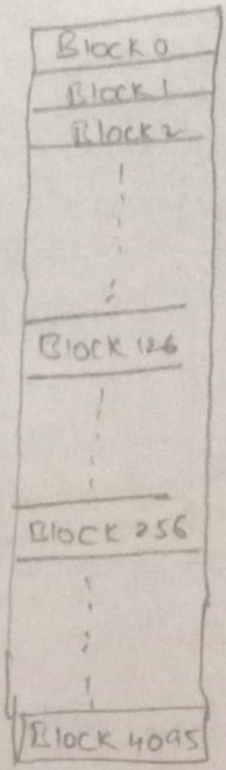
4	4
TAG	Word

Set Associative Mapping

Associative + Direct memory

cache memory





Total no, of sets = 32 = $\frac{4096}{128}$.

No, of Blocks in Each set = $\dfrac{\text{Total cache Block}}{\text{Total size}} = \frac{128}{32} = \frac{2^7}{2^5} = 2^2 = 4$

Size of the main memory = 4096 x 16 words

= 64 k words.

= 2^{16}

TAG = $\frac{4096}{32} = \frac{2^{12}}{2^5} = 2^7$

7	5	4
TAG	SET	Word

Identfying

M modulo S

↓ ↓

m.m Block No Total no, of sets

Ex:- 256 mod 32

= 0

which is mapped to set 0

~~Se~~ set 0 mapped with → 0, 32, 64, 96, 128, 160, 192 - - -

Total 128 Blocks mapped with set 0

Searching time is reduced compared to Associative mapping.

Replacement Algorithms

miss: Required word is not found in cache memory Block.

⇒ If not found then the processor load from main memory to cache memory.

Hit: If required word is found then we call it as hit

If it is available then processor load from cache memory.

Ex:

1	2	1	2	3	4	2	1	5	3
miss	miss	hit	hit	miss	miss	hit	hit	miss	hit

Cache memory

~~1~~ 5
2
3
4

Replacement is done in the form of (FIFO)

i,e 1 → 5.

⇒ Calculating Miss Ration: $\frac{\text{no, of miss}}{\text{Total no, of refrences}} = \frac{5}{10} = 0.5 = 50\%.$

→ Calculating hit ration: $\frac{\text{no, of hits}}{\text{Total no, of references}} = \frac{5}{10} = 0.5 = 50\%$

LRU: least recently Used Algorithm.

least rectntly Used Blocks.

Compare two three the previous one is older.

Ex: $1 \quad 2 \quad 1 \quad 2 \quad \overset{\text{older one}}{\underset{\overset{\uparrow}{5}}{\boxed{3}}} \quad \underbrace{4 \quad 2 \quad 1}_{\text{recently Used}}$

Cache memory

1
2
~~3~~ 5

MRU most Recently Used