2CEIT501:Computer Architecture & Organization

GANPAT UNIVERSITY

FACULTY OF ENGINEERING & TECHNOLOGY

Programme	Bachelor of Technology				Branch/Spec.	Computer Engineering / Information Technology		
Semester	V				Version	2.0.0.0		
Effective from Academic Year	2020-21				Effective for the batch Admitted in	July 2018		
Subject code	2CEIT501	Subject Name			Computer Architecture & Organization			
Teaching scheme					Examination scheme (Marks)			
(Per week)	Lecture (DT)		Practical (Lab.)		Total		CE	SEE	Total
	L	TU	P	TW					
Credit	3	0	1	-	4	Theory	40	60	100
Hours	3	0	2	-	5	Practical	30	20	50

Theory syllabus		
Unit	Content	Hrs
1	**Overview of Register Transfer and Micro Operations:** Register Transfer Language, Register Transfer, Bus & Memory Transfer, Arithmetic Micro-Operations, Logic Micro-Operations, Shift Micro Operations, Arithmetic Logic Shift Unit.	05
2	**Basic Computer Organization and Design:** Instruction Codes, Computer Registers, Computer Instructions, Timing & Control, Instruction Cycle, Memory-Reference Instructions, Input-Output & Interrupt, Complete Computer Description, Design of Basic Computer, Design of Accumulator Unit.	05
3	**Micro Programmed Control:** Control Memory, Address Sequencing, Micro Program Example, Design of Control Unit.	03
4	**Central Processing Unit:** Introduction, General Register Organization, Stack Organization, Instruction Format, Addressing Modes, Data Transfer & Manipulation, Program Control, Reduced Instruction Set Computer (RISC).	05
5	**Pipeline and Vector Processing:** Parallel Processing, Pipelining, Arithmetic Pipeline, Instruction, Pipeline, RISC Pipeline, Vector Processing, Array Processor.	04
6	**Input – Output Organization:** Input-Output Interface, Asynchronous Data Transfer, Modes Of Transfer, Priority Interrupt, DMA, Input-Output Processor (IOP), CPUIOP Communication, Serial Communication.	04
7	**Memory Organization:** Memory Sub System, Memory Hierarchy, Main Memory, Auxiliary Memory, Flash Memory, Associative Memory, Cache Memory, Virtual Memory, Memory Management Hardware.	05
8	**Microprocessor Architecture:** 8085 Architecture, Instruction Set, Instruction Types & Formats, Instruction Execution,	14

Architecture vs Organization

- **Architecture:**
  - Conceptual design & fundamental operational structure

Architecture vs Organization

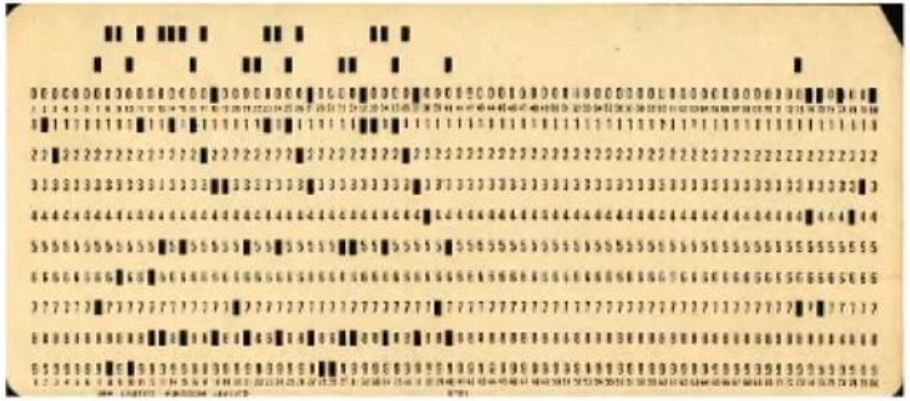
- **Organization:**
  +
  - Deals with physical devices and their interconnections
  - With a perspective of improving the performance

Computer Organization & Architecture

+

Computer Architecture	Computer Organization
• CPU Design	• I/O Organization
• Instructions	• Memory Organization
• Addressing modes	• Performance
• Data format	

Punch Card Computers



+

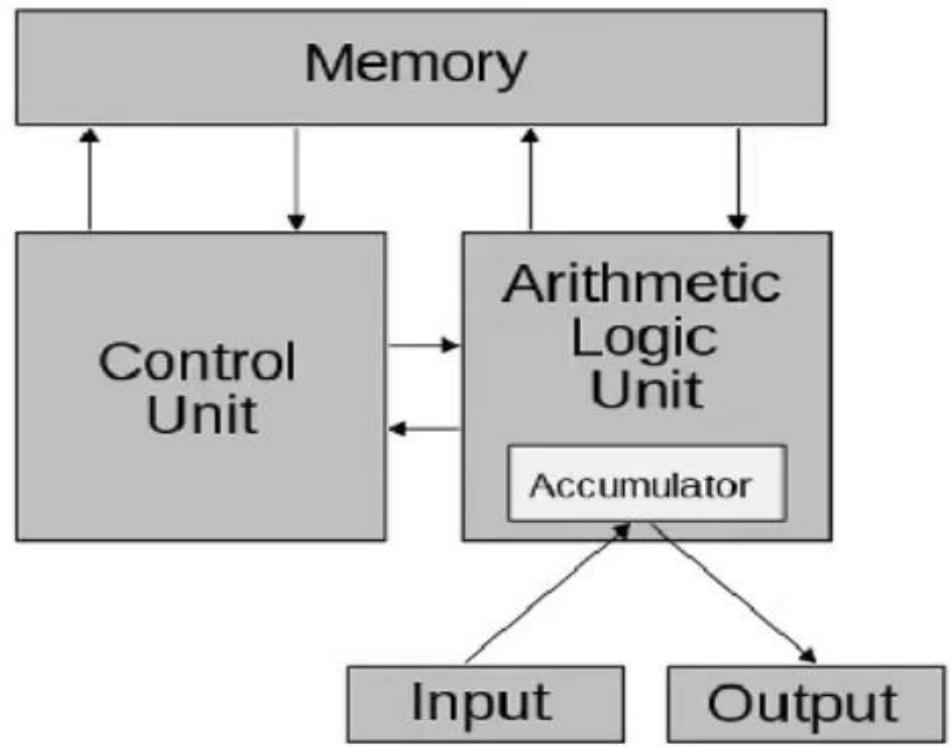
Von Neumann's Architecture

+

- Also known as 'Stored Program Architecture'

Von Neumann's Architecture

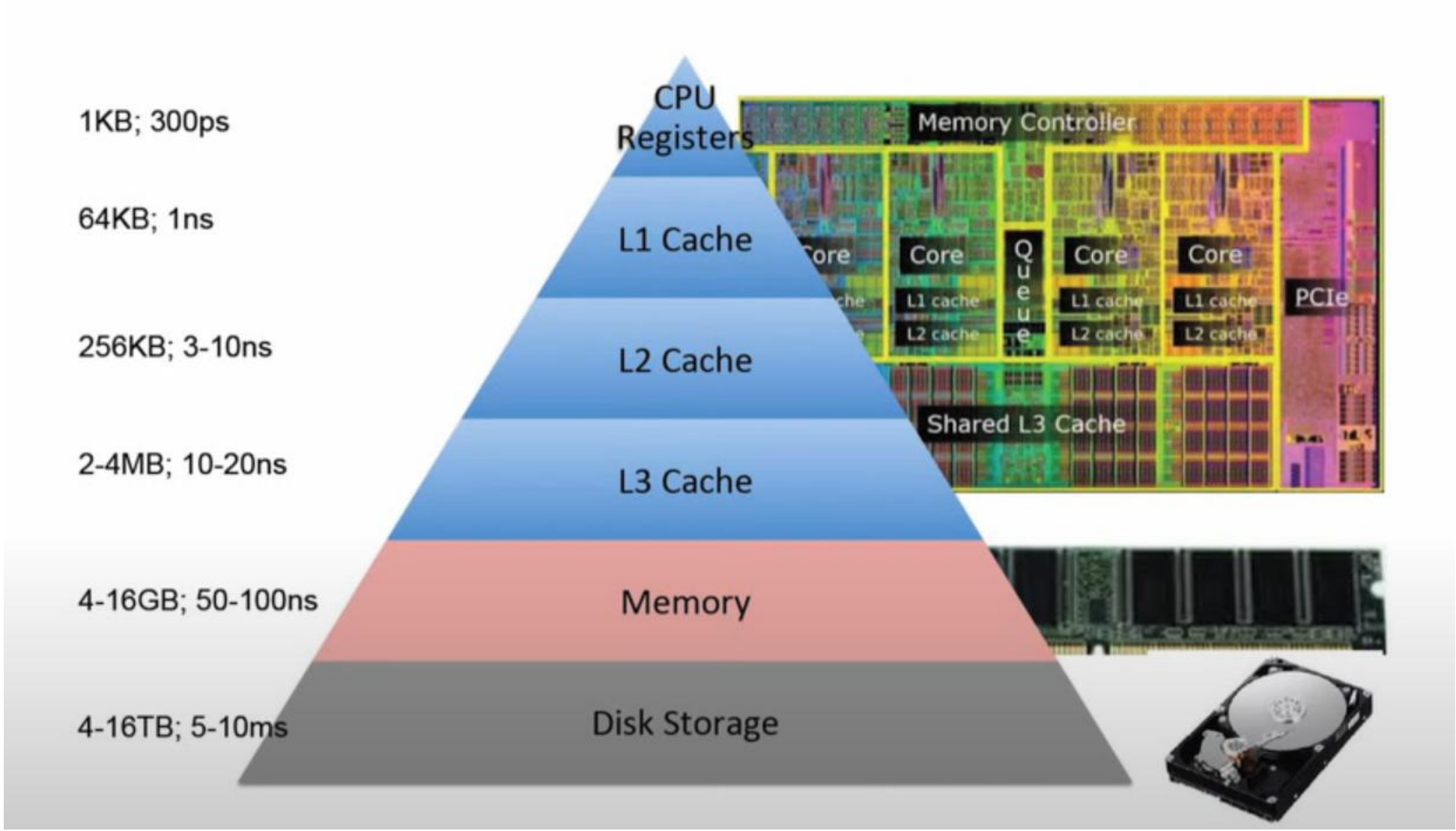
- Also known as 'Stored Program Architecture'



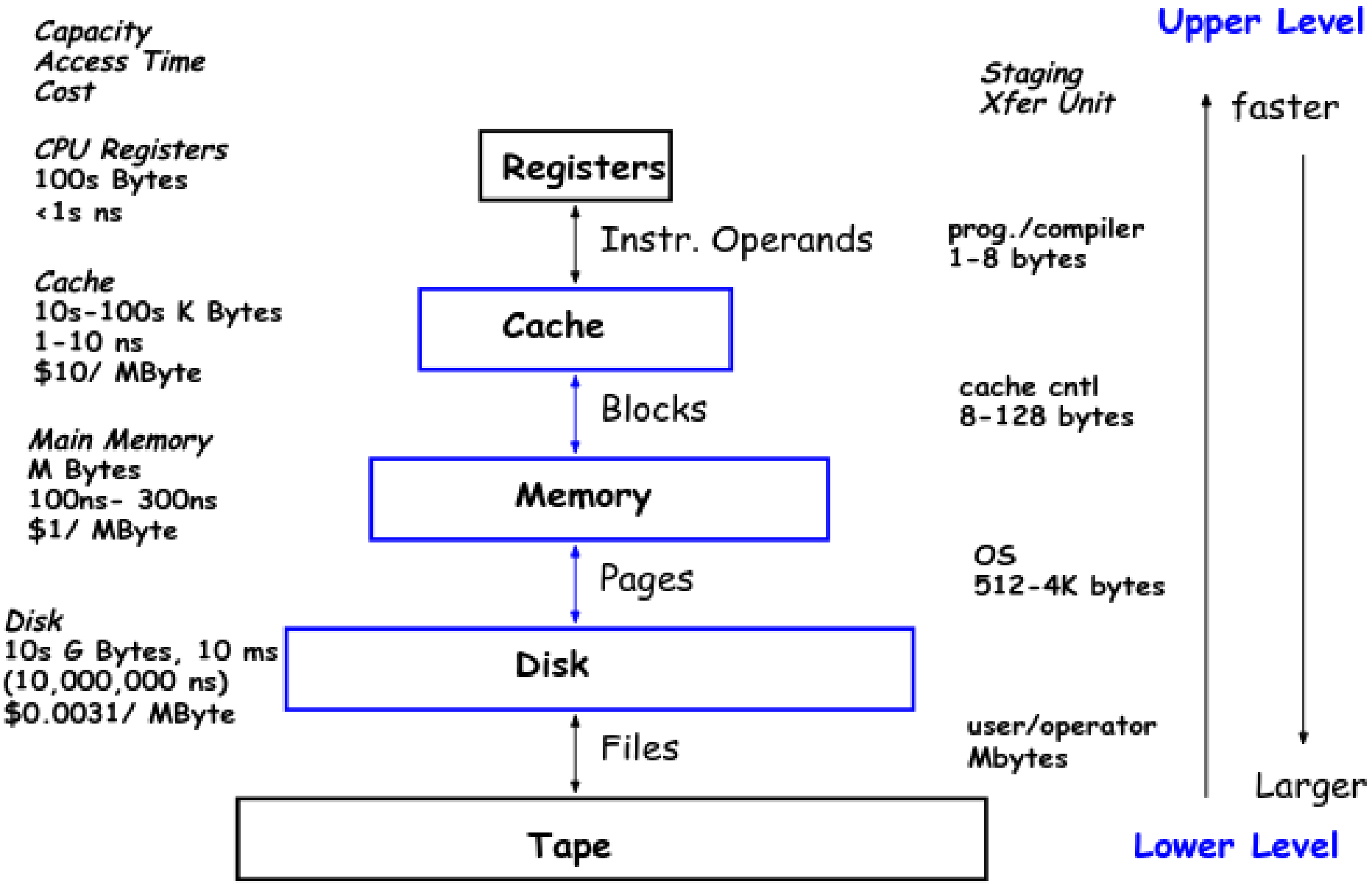
Harvard Architecture

Memory Organization

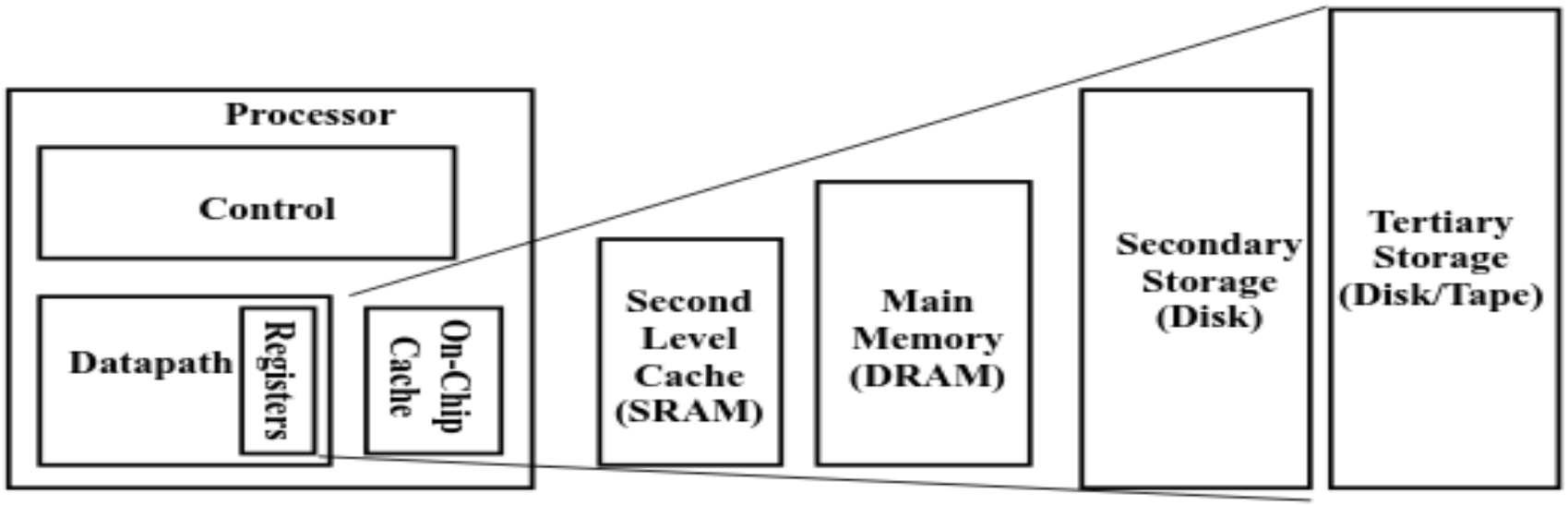
Memory Hierarchy



Levels of the Memory Hierarchy

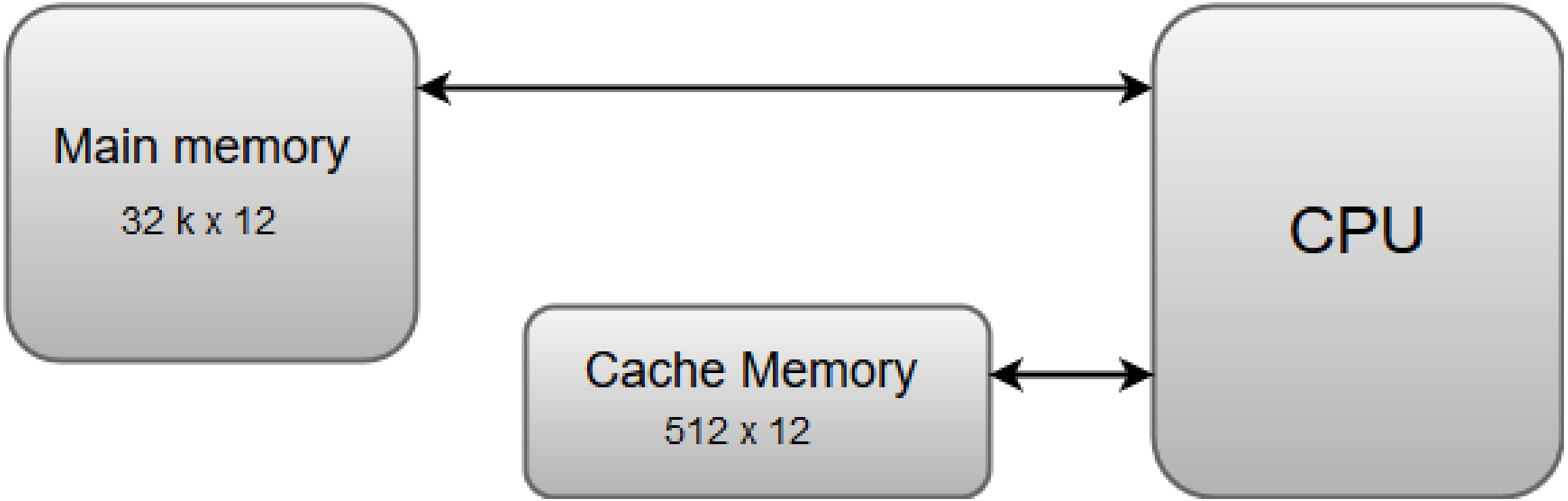


Levels of the Memory Hierarchy



Levels of the Memory Hierarchy

- Main memory directly communicates with the CPU.
- Devices that provides backup storage are called auxiliary memory.
- A special very high speed memory is sometimes used to increase the speed of processing
- by making current programs and data available to the CPU at the rapid rate.


Main memory
32 k x 12
Cache Memory
512 x 12
CPU

Memory

- A physical device used for storage in computer system.

- Stores program and associated data.

Memory hierarchy

- Computers have limited storage.
- Users tend to store more information and data with time.
- At a time, CPU runs only finite number of instructions.
- Computers can be benefited with extra storage.

Associative Memory

Associative Memory

- Content Addressable memory
- E.g.:
  - Student information
  - Bank information

Associative Memory

- RAM
  - Memory accessed using location address
  - Needs to search the content of each location one after other
  - Requires the hardware for read and write operations
- Associative Memory
  - Memory accessed using content
  - Search is done simultaneously
  - Requires the hardware for read and write as well as the search operations

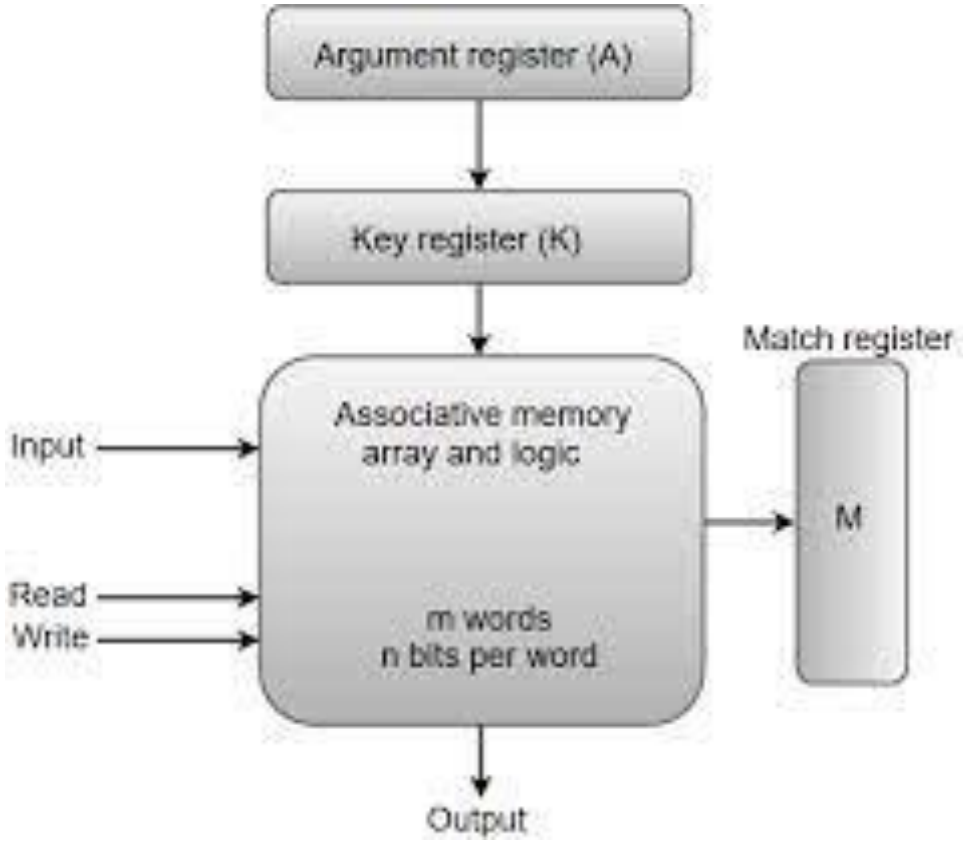
Associative Memory

- Faster access
- Expensive

Associative Memory

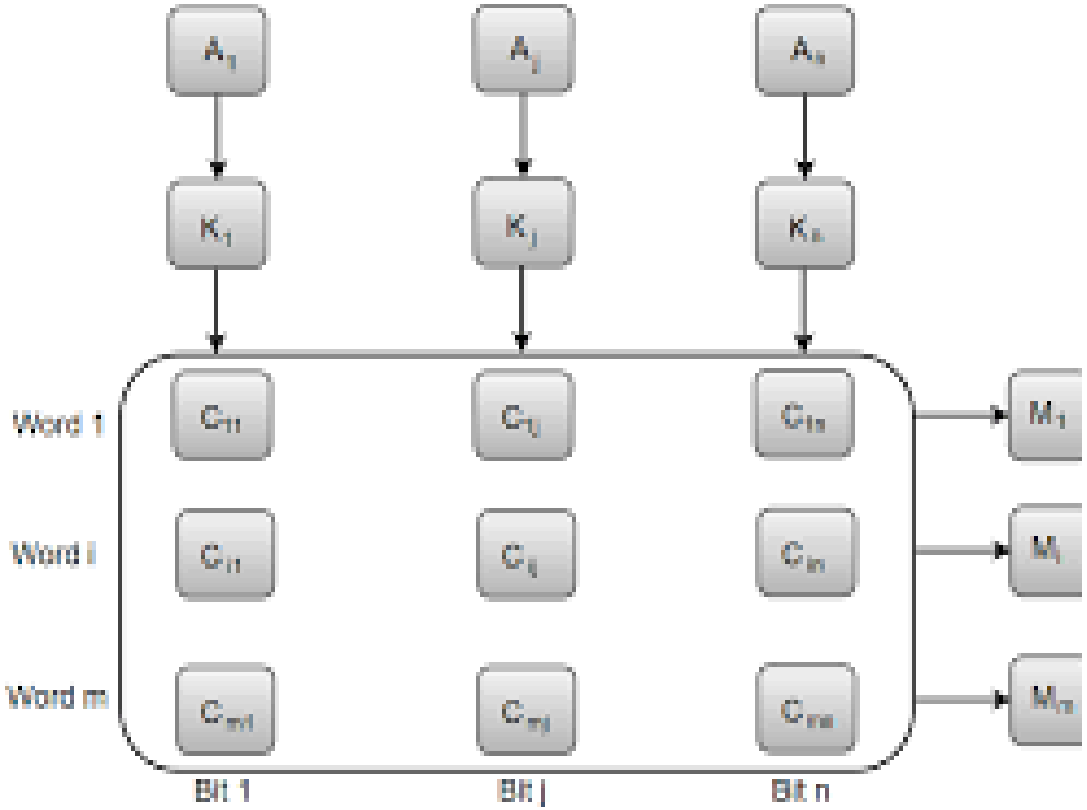
- Associative cache
- Translation look aside buffer

Associative Memory



- Arguement Register
  - For searching
- Key register
  - For masking (Conditional search)

Associative Memory



- Each cell implemented using flip flop and search logic

Associative Memory – Match logic

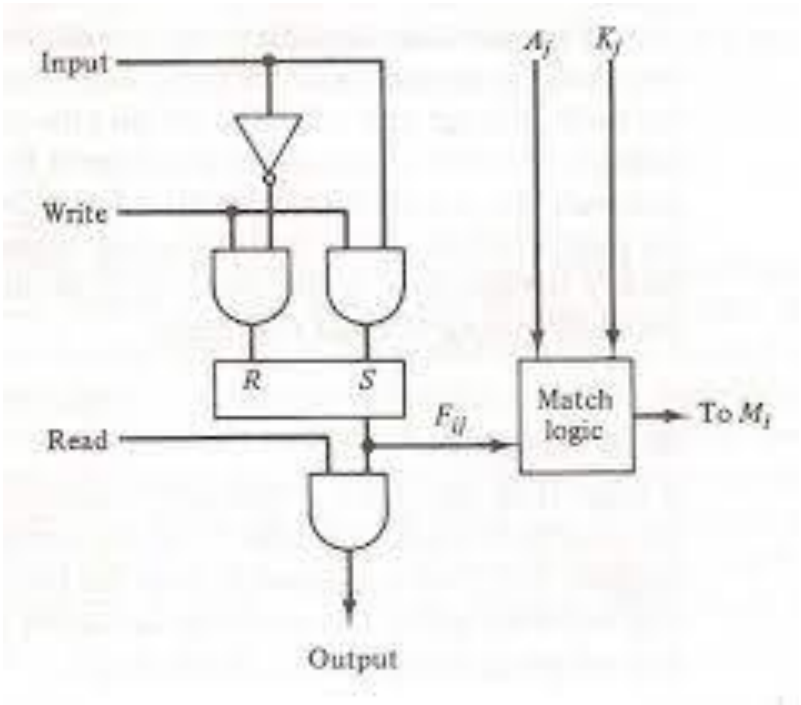
- Output $x_j = 1$ when A_j and F_{ij} matches.
- $M_i = 1$, when all the x_j's are 1

Associative Memory – One cell logic

- Cell using flip-flop

S	R	Q(t)	Q(t+1)
0	0	Q	Q (retain)
0	1	Q	0 (Reset)
1	0	Q	1 (Set)
1	1	Q	X (in valid)

Associative Memory

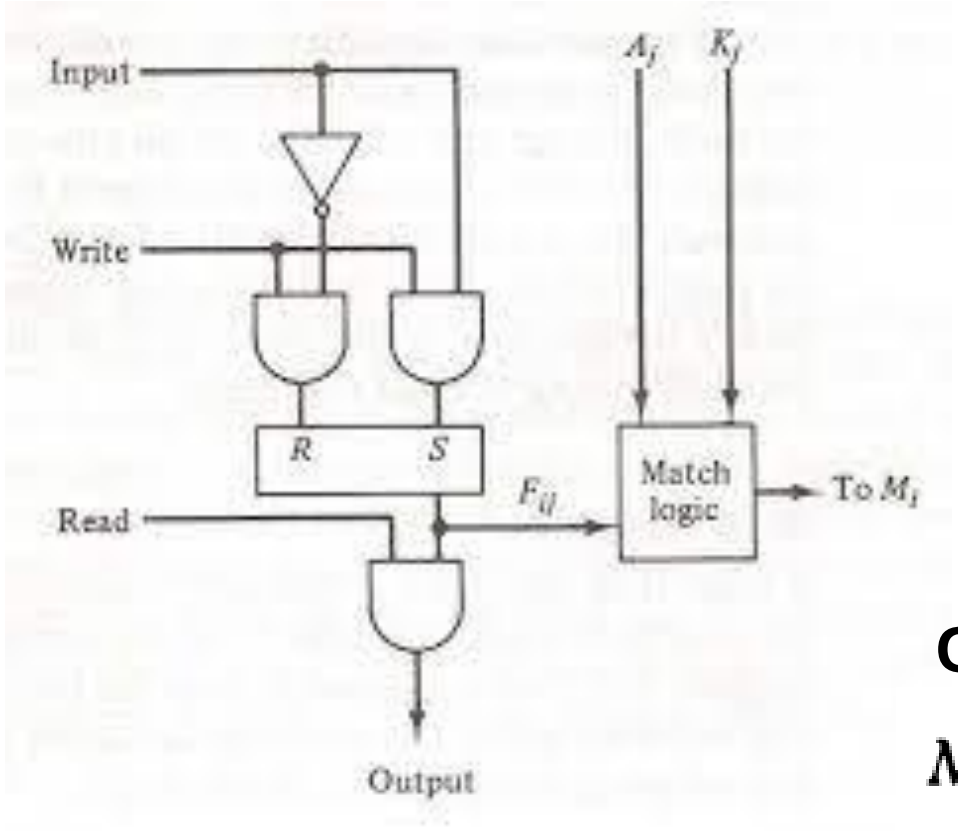


Aj	Fij	xi
0	0	1
0	1	0
1	0	0
1	1	1

Using NOR gate: Match logic can be

$$x_j = A_j F_{ij} + A_j' F_{ij}'$$

Inclusion of flag bit



$$x_j + K_j' = \begin{cases} x_j & \text{if } K_j = 1 \\ 1 & \text{if } K_j = 0 \end{cases}$$

Considering all the bits:

$$M_i = (x_1 + K_1')(x_2 + K_2')(x_3 + K_3') \cdots (x_n + K_n')$$

Associative Memory

Considering all the bits:

$$M_i = (x_1 + K_1')(x_2 + K_2')(x_3 + K_3') \cdots (x_n + K_n')$$

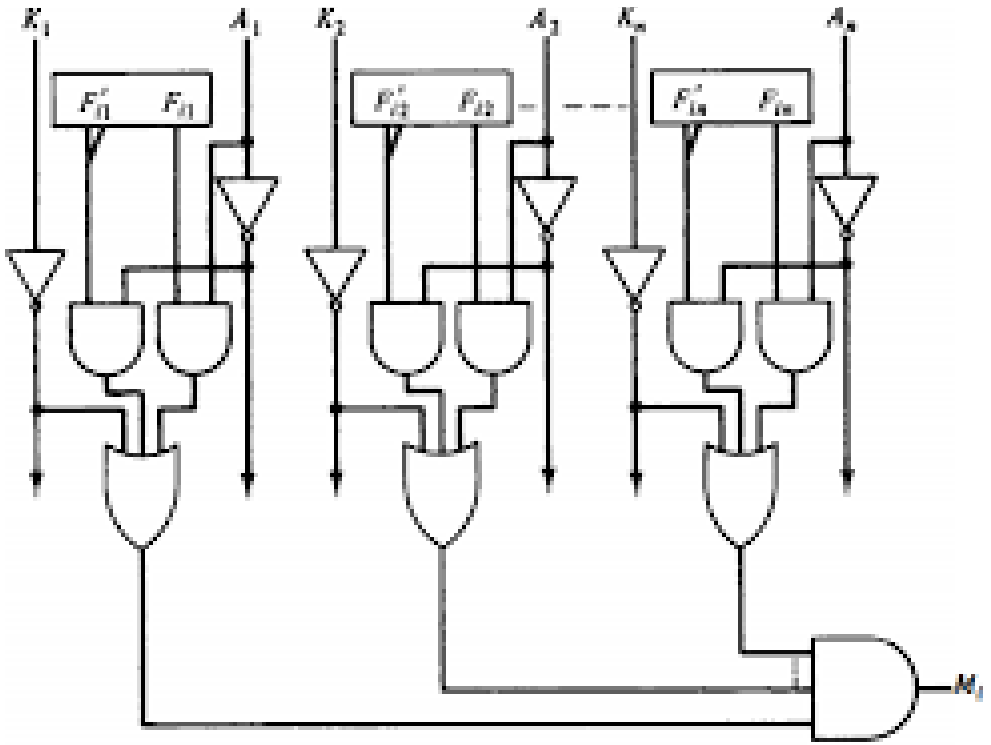


Figure 9 Match logic for one word of associative memory.

Associative Memory

- Read operation
  - Compare the argument register (after the masking effect of flag register) with all the location simultaneously.
  - The location corresponding to the matching content will have the match bit as 1.
  - The location with match bit as 1 are accessed in sequence.

Associative Memory

- Write operation
  - Case:1 Cache is empty (all tags=0)
    - Write the data sequentially.
  - Case:2 Cache is in use (some tags ≠ 0)
    - Find the first location with tag=0, which indicates that the location do not have valid input.
    - Write the content at the found location.

Cache

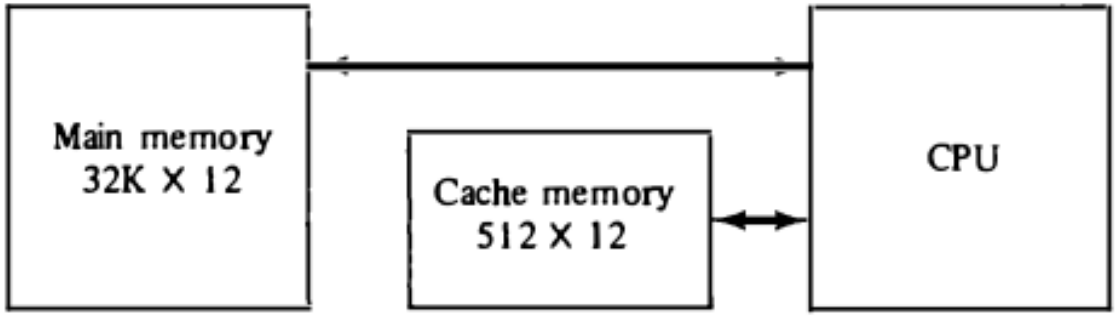
- Locality of reference: The processor accesses some data and instructions now, there is a high chance that the same data will be required in future or the neighboring data may be required in future.

```
for(i=0; i<3;i++)
        for (j=0; j<3;j++)
                printf("%d, %d", i, j);
```

- Execution of same instruction again: Temporal locality
- Execution of neighboring instruction: Spatial locality

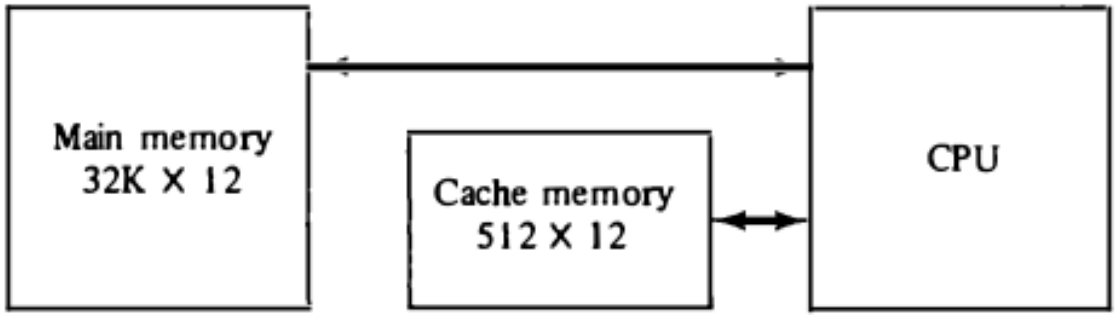
Cache

- If the active portions of the program and data are placed in a fast small memory, the average memory access time can be reduced, thus reducing the total execution time of the program. Such a fast small memory is referred to as a **cache memory.**



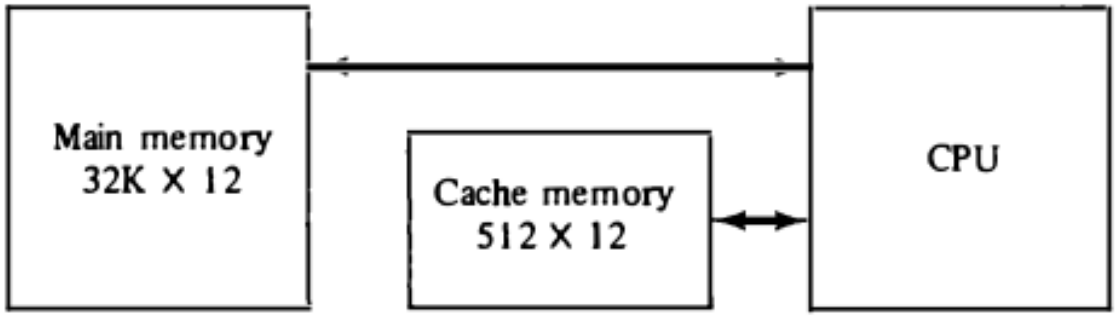
Cache

- When the CPU needs to access memory, the cache is examined.
- If the word is found in the cache, it is read from the fast memory.
- If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word.



Cache

- A block of words containing the one just accessed is then transferred from main memory to cache memory. The block size may vary from one word (the one just accessed) to about 16 words adjacent to the one just accessed.
- In this manner, some data are transferred to cache so that future references to memory find the required words in the fast cache memory.



Terms related to Cache

- Word: The content stored at one address location.
- Block: The local content comprising of group of words, which is currently in demand.

Terms related to Cache

- The performance of cache memory is frequently measured in terms of a quantity called hit ratio.
- When the CPU refers to memory and finds the word in cache, it is said to produce a main memory and it counts as a **hit**. If the word is not found in cache, it is in **miss.**
- The higher hit ratio indicates faster average access time.

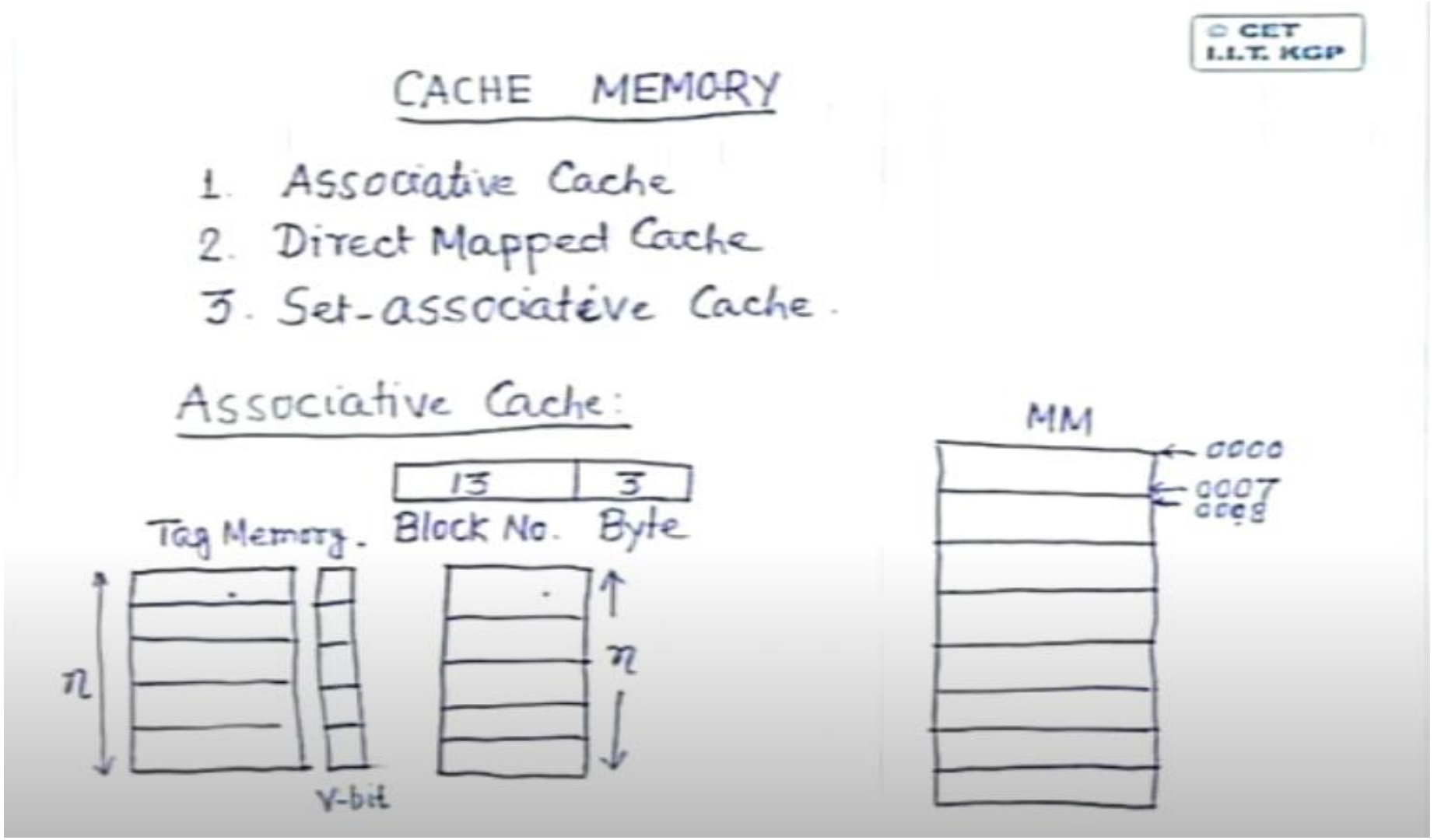
Terms related to Cache

- Average memory access time:
  - *hit * (hit access time) + miss * (miss access time)*

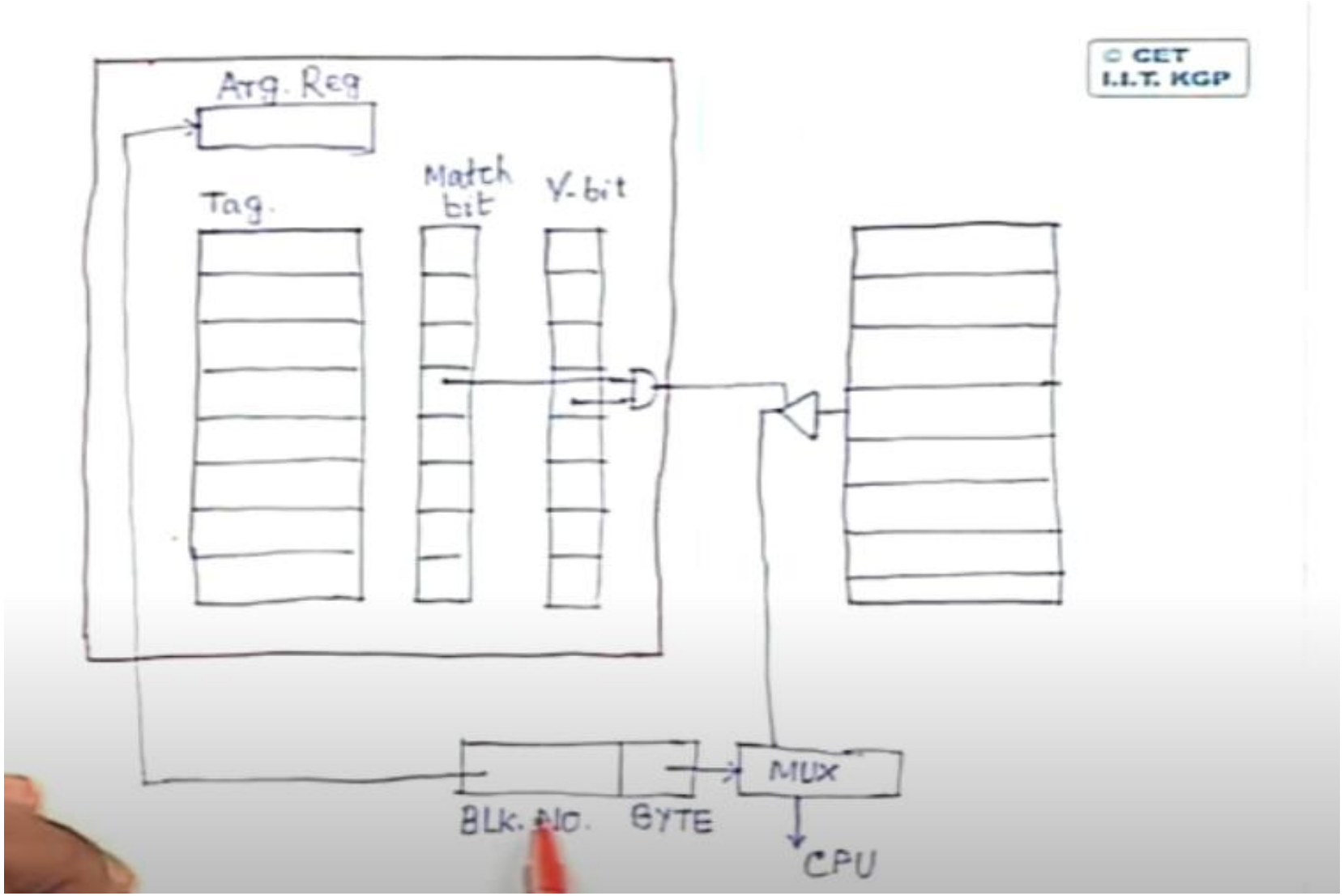
Cache mapping

- The transformation of data from main memory to cache memory is referred to as a mapping process.
  - Associative mapping
  - Direct mapping
  - Set associative mapping

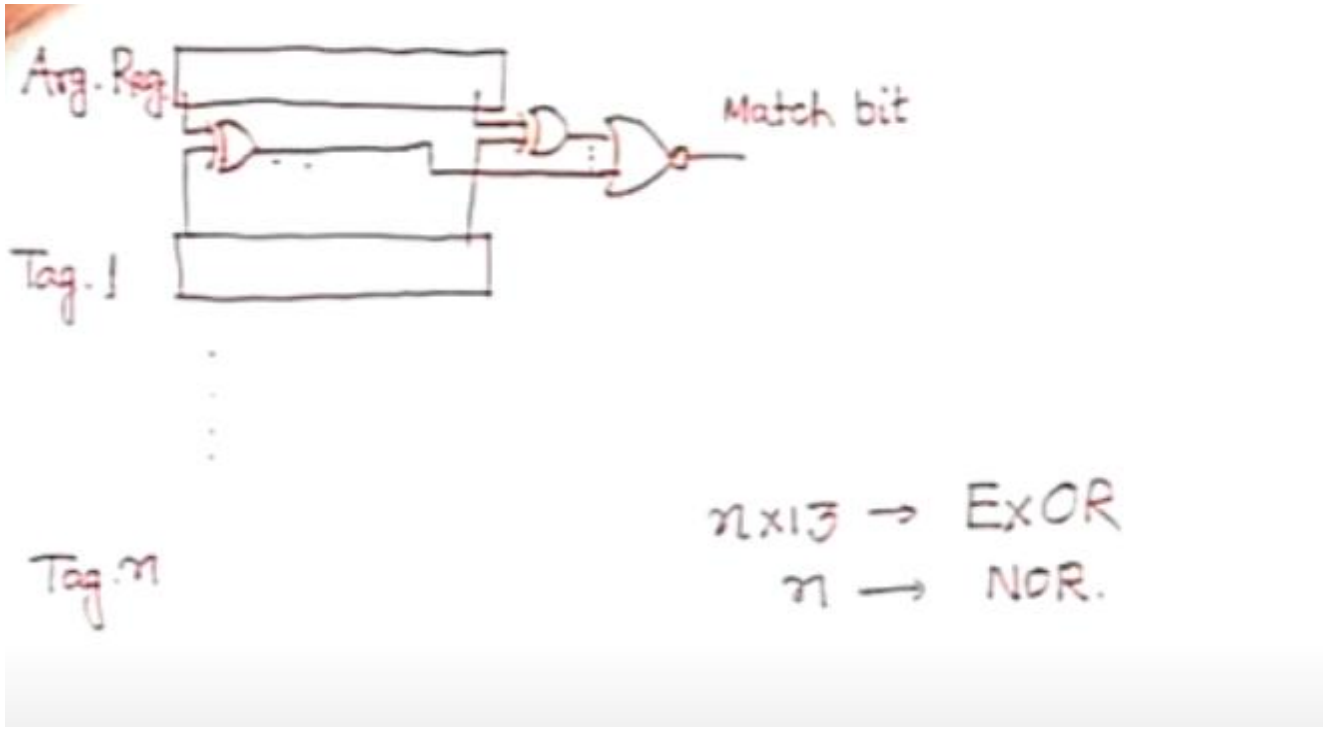
Cache Memory



Associative cache



Associative cache (Match Bit)



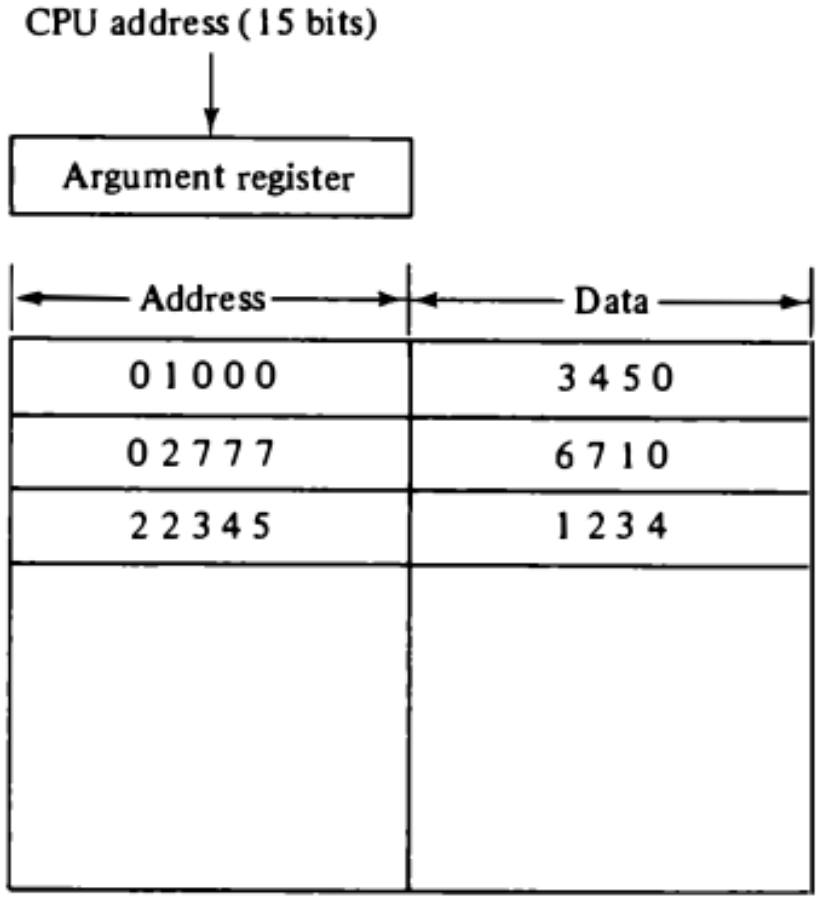
Associative cache

Write:

- It stores both the address and data of the memory word.
- This permits any location in cache to store any word from main memory.

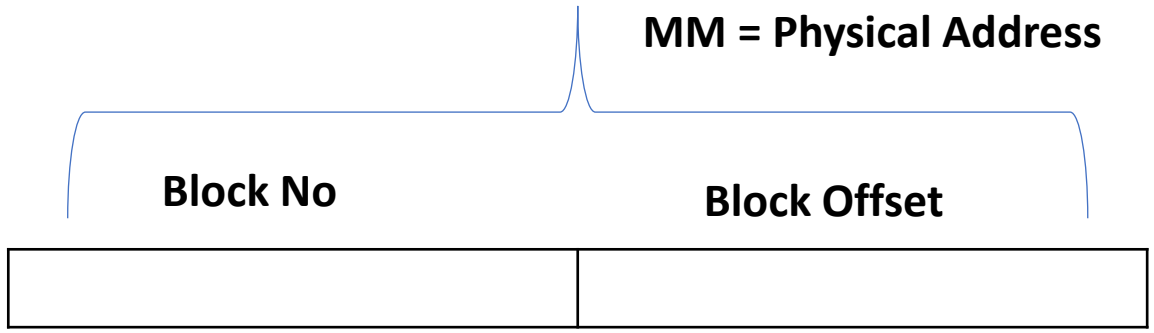
Read:

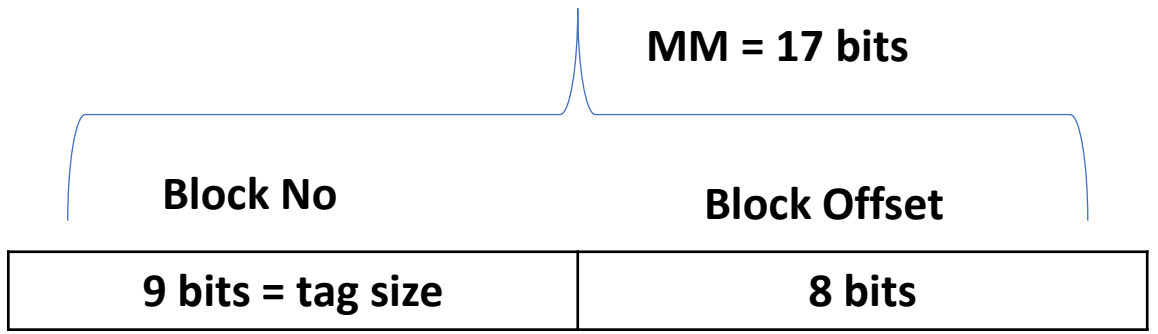
- Simultaneous search of address.
- Read the locations where exists.



Associative Memory Example :

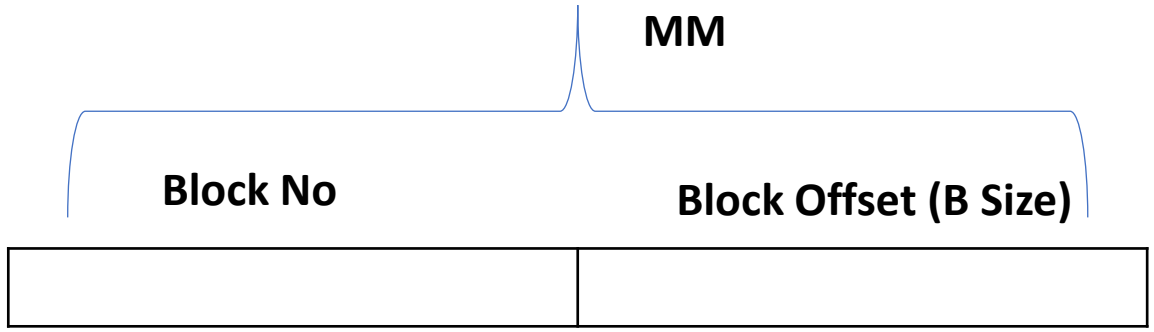
MM Size	Cache Size	Block Size	Tag Size	Tag directory Size	Comparator
128 KB	16KB	256 B	**9 bits**	**(9 * 64) bits**	**64**
32 GB	32 KB	1 KB	**25 bits**	**(25 * 32) bits**	**32**
128 MB	512 KB	1 KB	17 BITS	**(17 * 512) bits**	**512**
16 GB		4 KB	**22 bits**	Can not get	
64 MB		64 KB	**10 bits**	Can not get	
	512 KB		7 bits		

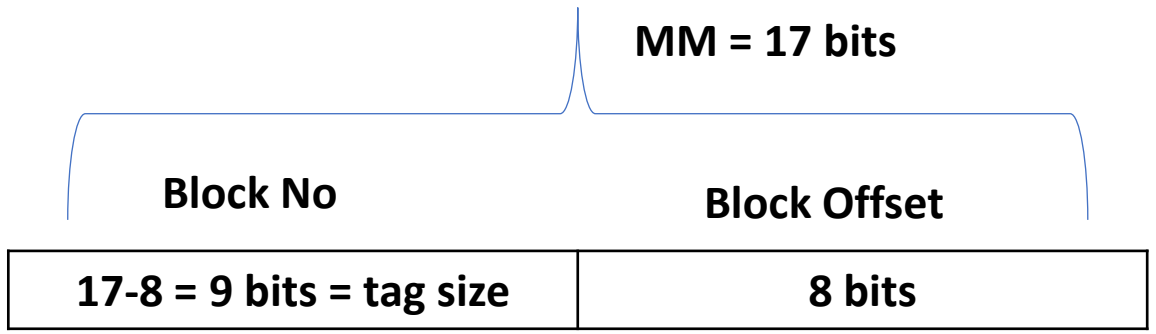




Associative Memory Example :

MM Size	Cache Size	Block Size	Tag Size	Tag directory Size	Comparator
128 KB	16KB	256 B	**9 bits**	**(9 * 64) bits**	**64**





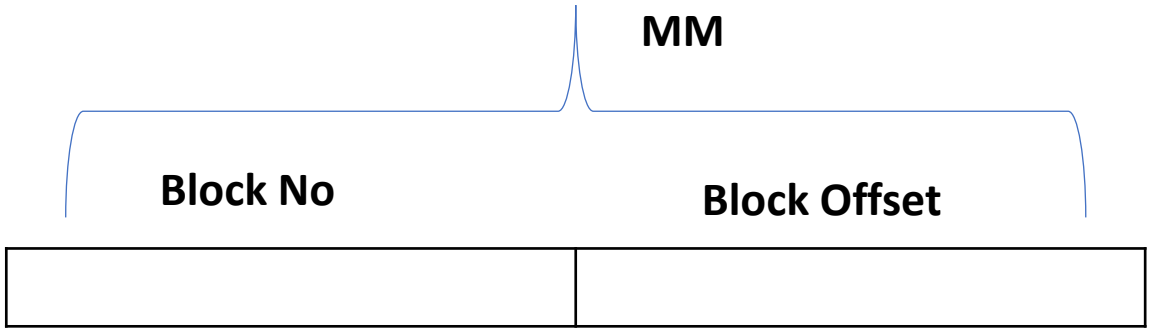
Cache size = 16 KB = 2^{14}

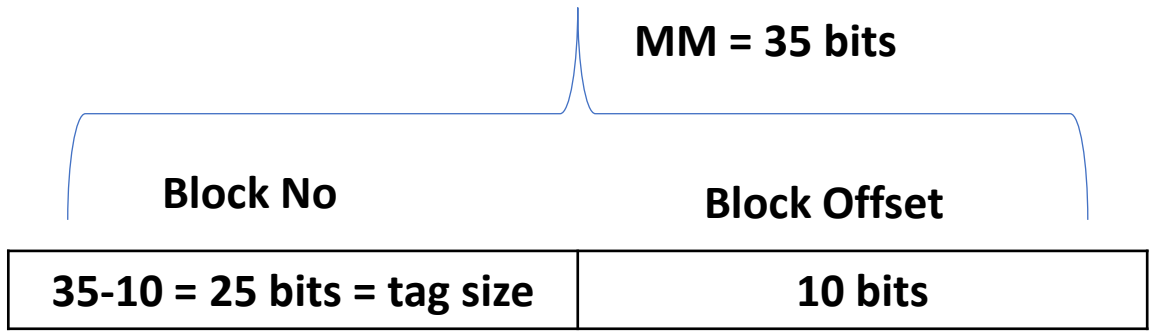
Block size = 256 B = 2^{8}

Cache size / **Block size** = $\frac{2^{14}}{2^{8}}$ = 2^{6} = 64 line in cache

Associative Memory Example :

MM Size	Cache Size	Block Size	Tag Size	Tag directory Size	Comparator
32 GB	32 KB	1 KB	**25 bits**	**(25 * 32) bits**	**32**





Cache size = 32 KB = 2^{15}

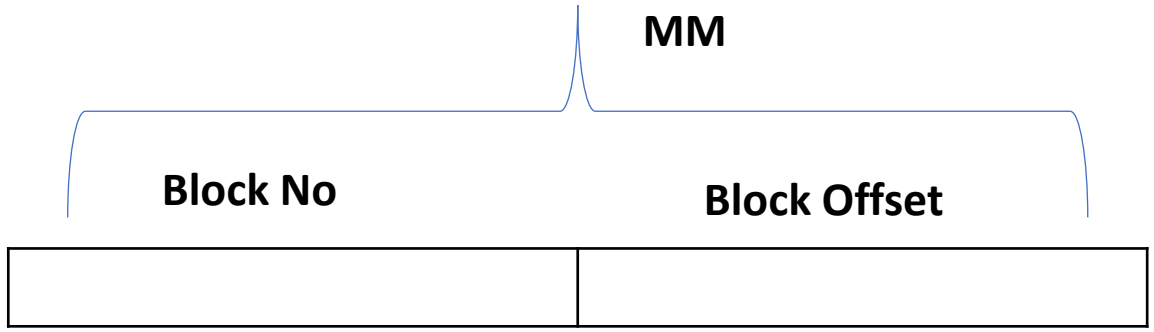
Block size = 1 KB = 2^{10}

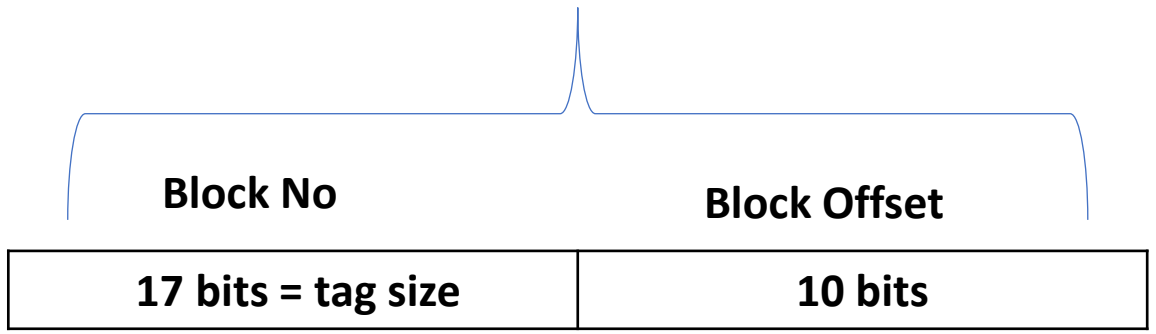
$$\frac{\textbf{Cache size}}{\textbf{Block size}} = \frac{2^{15}}{2^{10}} = 2^{5} = 32 \text{ line in cache}$$

Associative Memory Example :

MM Size	Cache Size	Block Size	Tag Size	Tag directory Size	Comparator
128 MB	512 KB	1 KB	17 Bits	**(17 * 512) bits**	**512**

MM = 17+10 = 27 bits = 2^{27} = 128 MB





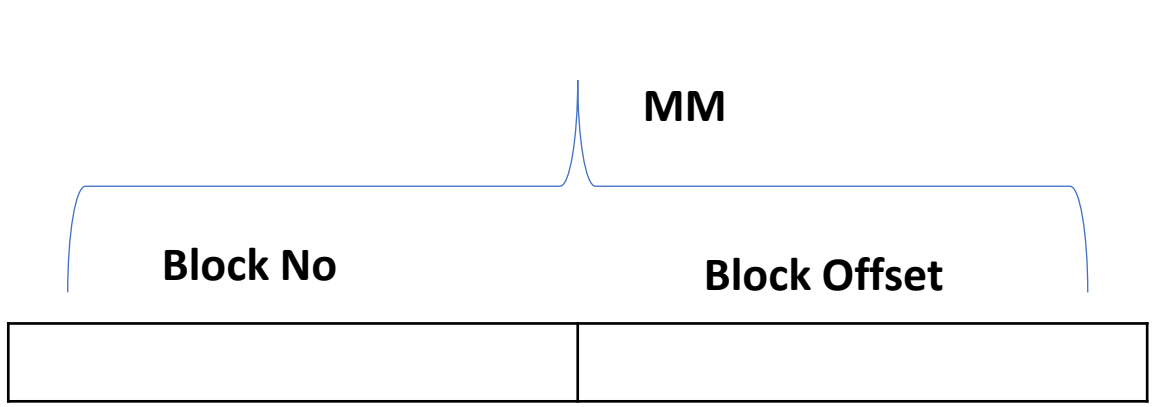
Cache size = 512 KB = 2^{19}

Block size = 1 KB = 2^{10}

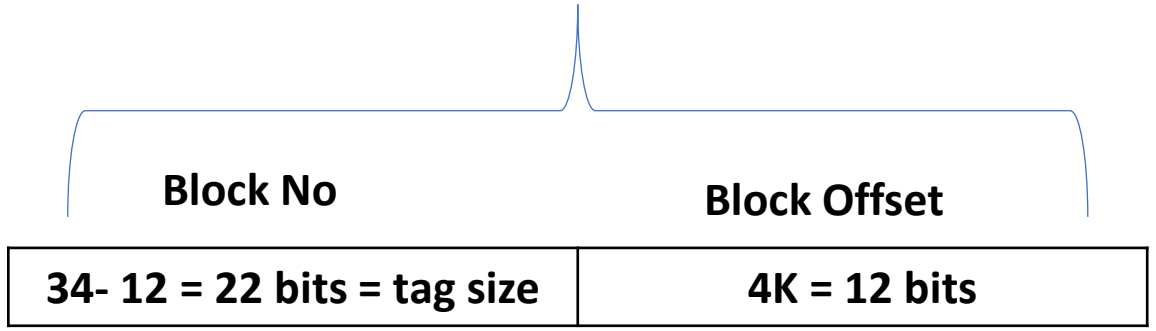
$$\frac{\textbf{Cache size}}{\textbf{Block size}} = \frac{2^{19}}{2^{10}} = 2^{9} = 512 \text{ line in cache}$$

Associative Memory Example :

MM Size	Cache Size	Block Size	Tag Size	Tag directory Size	Comparator
16 GB	X	4 KB	**22 bits**	Can not get	X

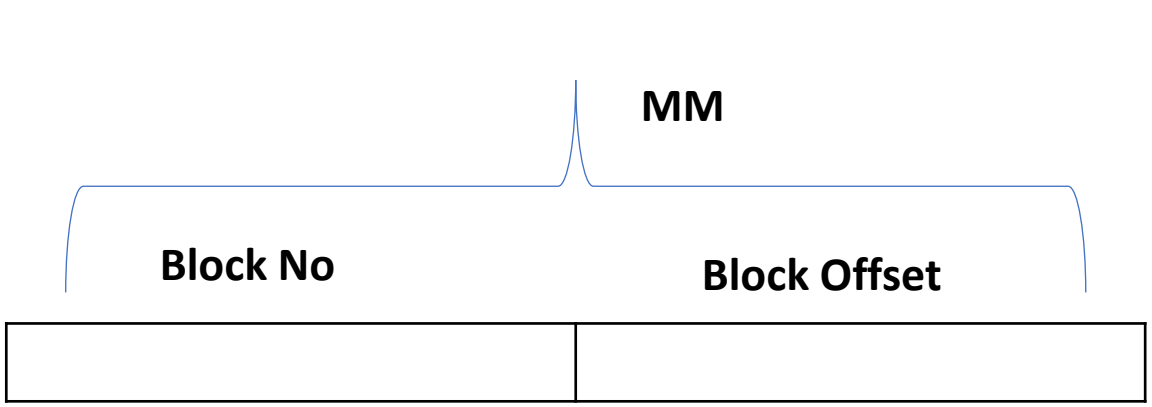


MM = 2^{34} = 34 bits = 16 GB

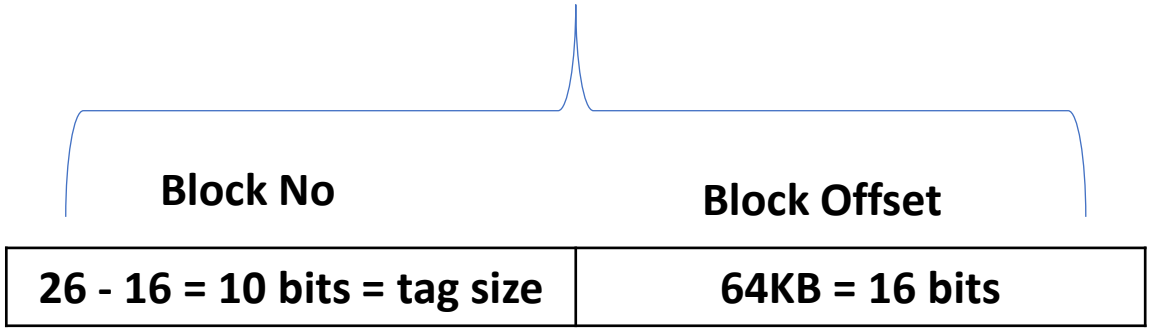


Associative Memory Example :

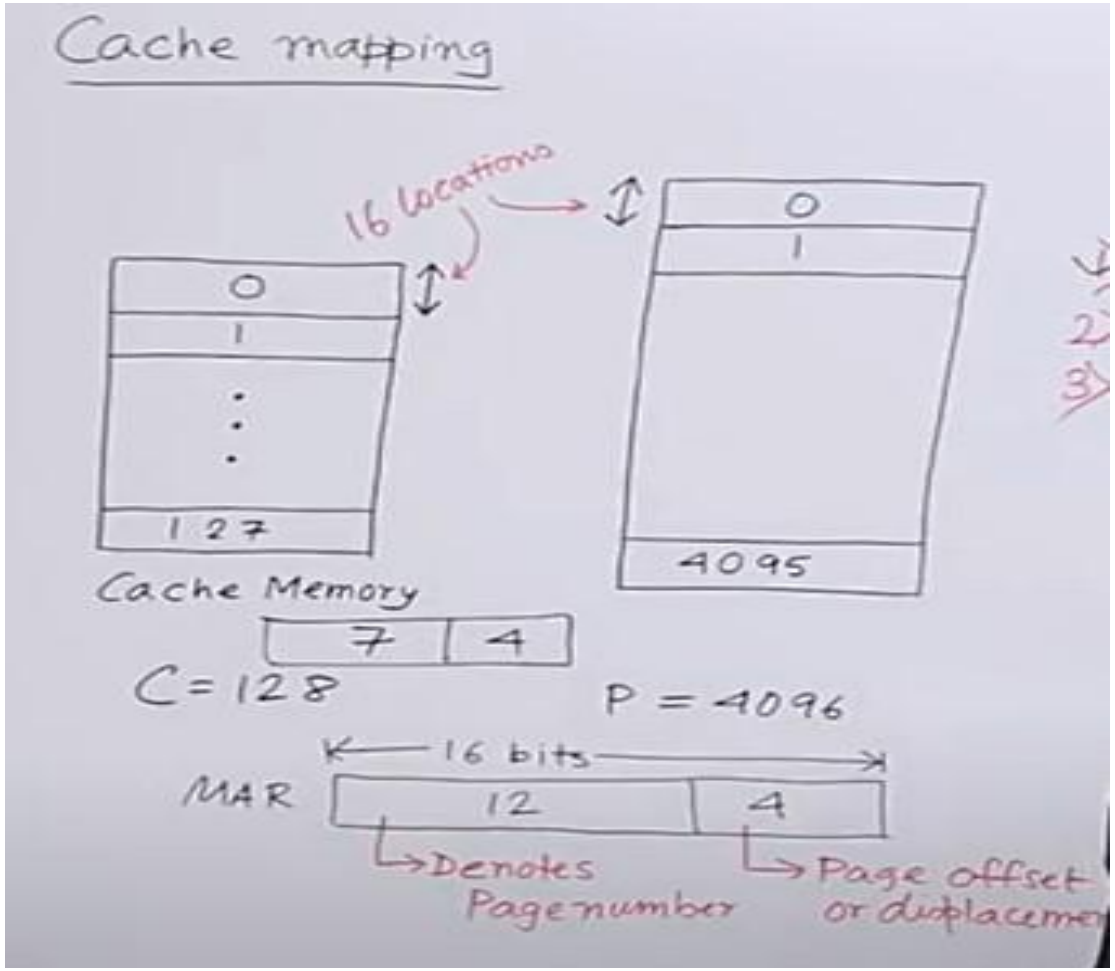
MM Size	Cache Size	Block Size	Tag Size	Tag directory Size	Comparator
64 MB	x	64 KB	**10 bits**	Can not get	x

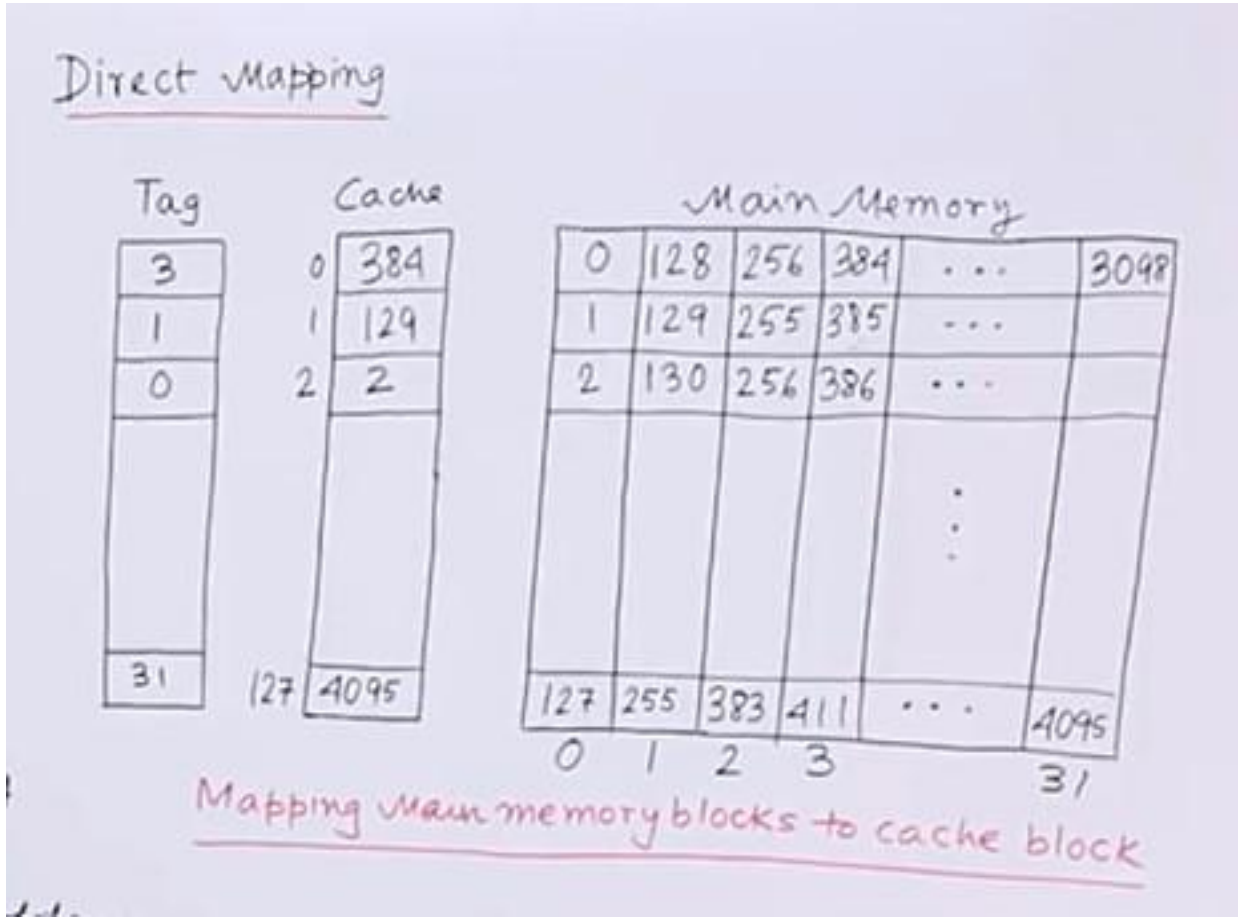


MM = 2^{26} = 26 bits = 64MB

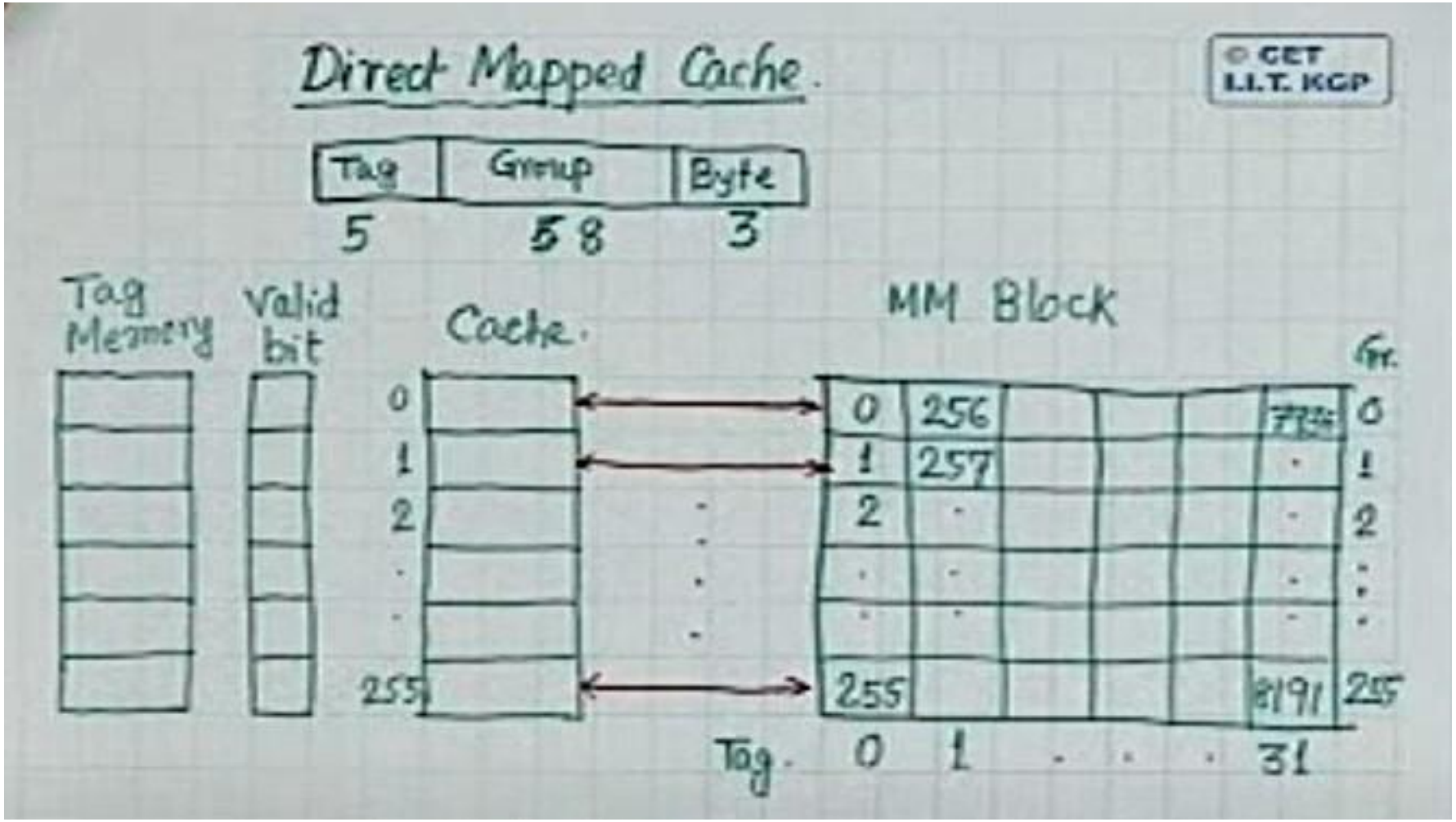


Direct Mapped Cache

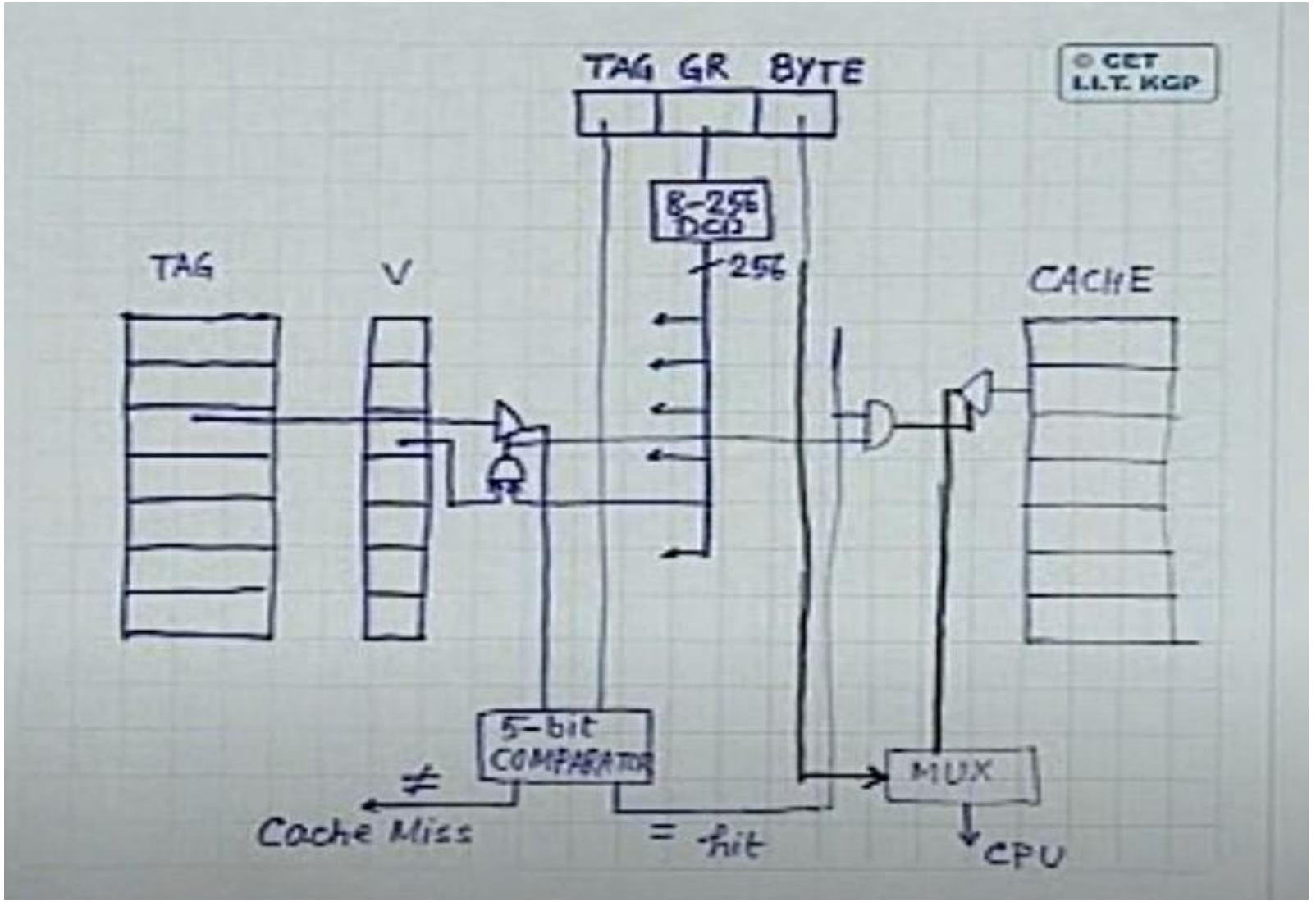




Direct Mapped Cache



Direct Mapped Cache



Discuss direct cache mapping technique.

Complete missing parameter in below table using direct cache mapping technique, assume memory is byte addressable.

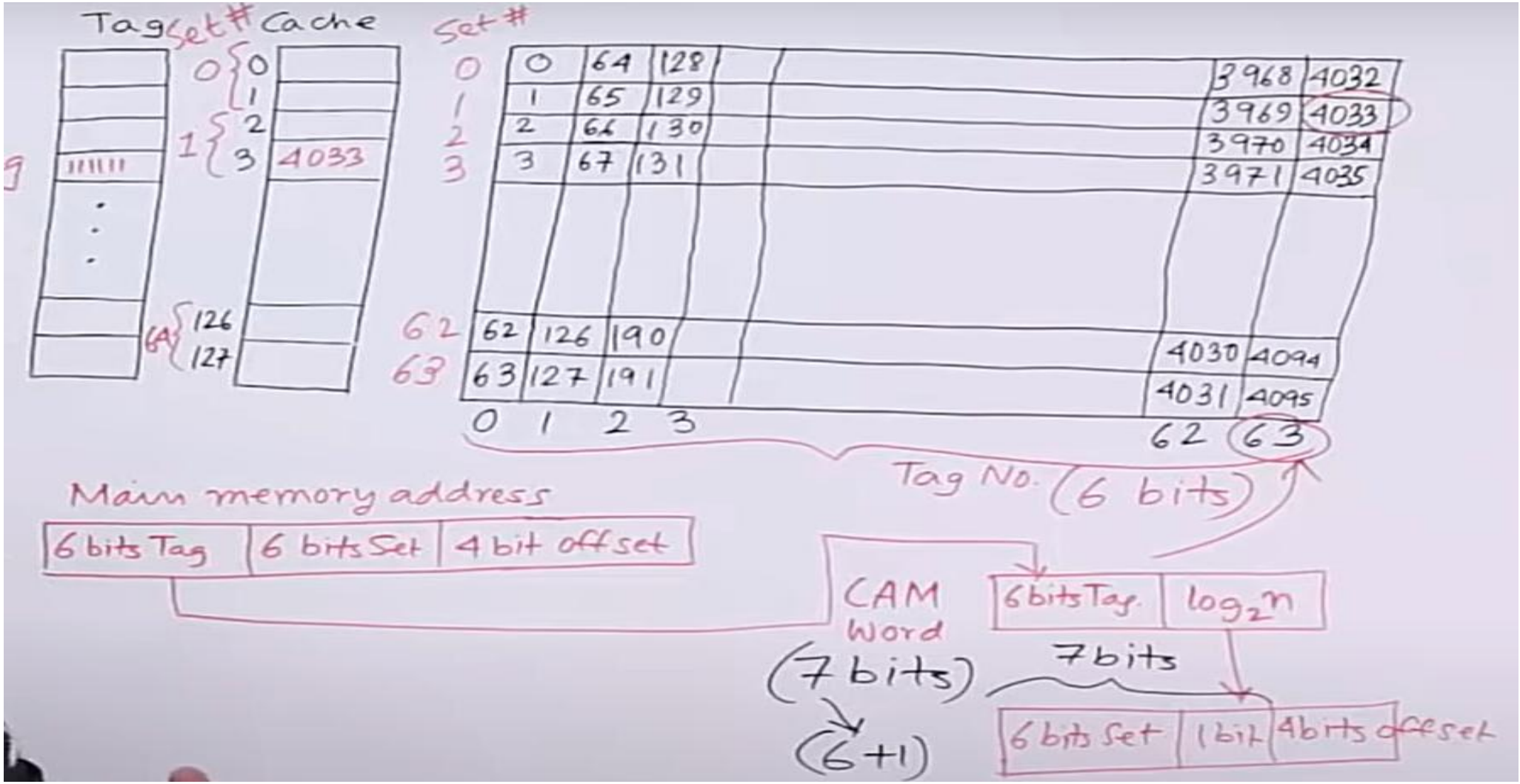
QueNO	M.M (size)	Cache Size	Block Size	Tag Bit	Tag Directory
1	128kb	16kb	256b	_____	_____
2	32 Gb	32kb	1 kb	_____	_____
3	_____	512 kb	1 kb	7	_____
4	16 Gb	_____	4 kb	10	_____
5	64MB	_____	_____	10	_____
6	_____	512 kb	_____	7	_____

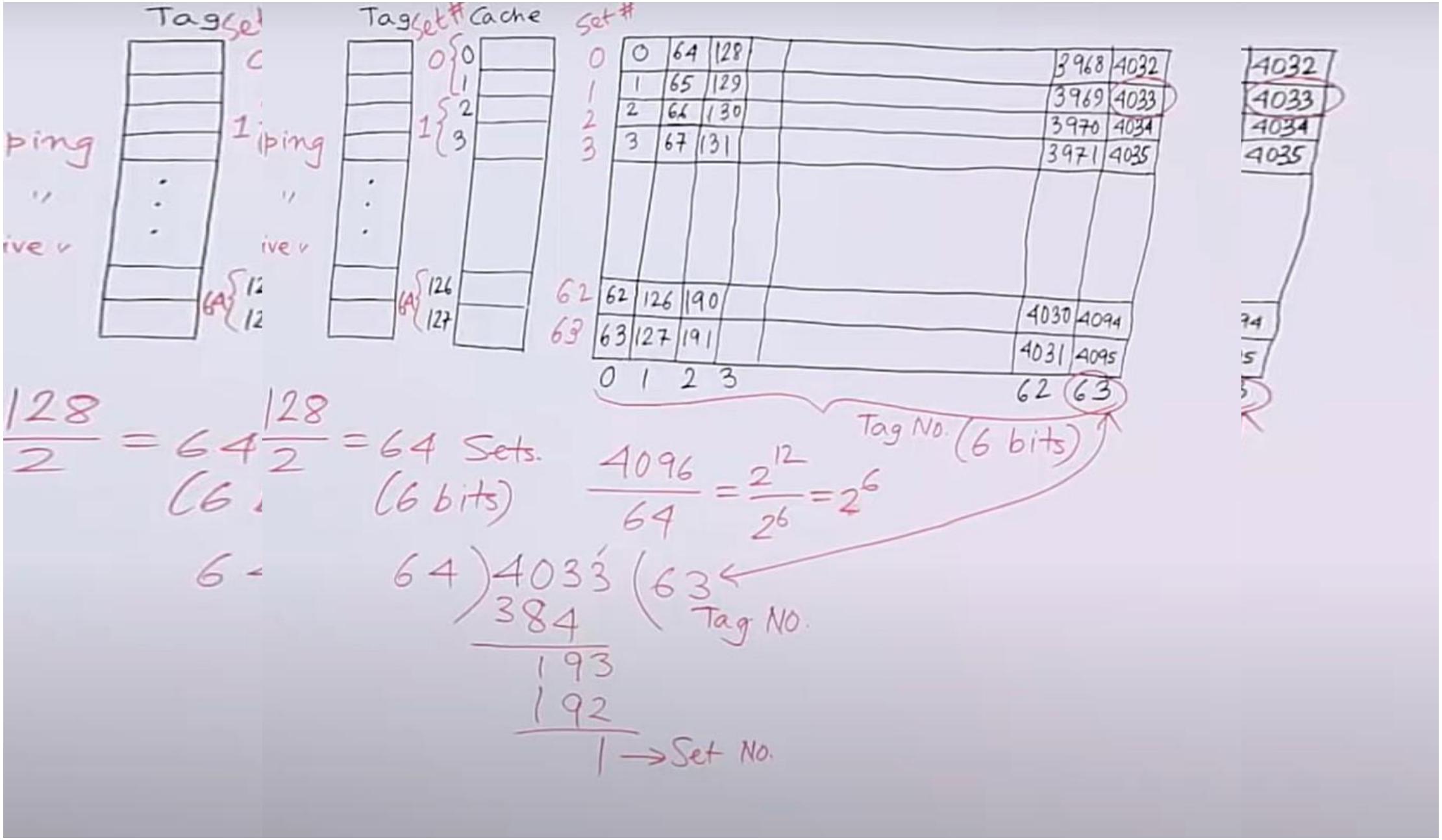
Set Associativity

Set Associativity

Memory address	Memory data
00000	1220
00777	2340
01000	3450
01777	4560
02000	5670
02777	6710

Index	Tag	Data	Tag	Data
000	01	3450	02	5670
777	02	6710	00	2340


Tag
Set #
Cache
Set #
0
1
2
3
4033
126
127
0 64 128
1 65 129
2 66 130
3 67 131
62 126 190
63 127 191
3968 4032
3969 4033
3970 4034
3971 4035
4030 4094
4031 4095
0 1 2 3
62 63
Tag No. (6 bits)
Main memory address
6 bits Tag
6 bits Set
4 bit offset
CAM
Word
(7 bits)
(6+1)
6 bits Tag
log2n
7 bits
6 bits Set
1 bit
4 bits offset


Tag
Set #
Cache
0
1
2
3
62
63
0 64 128
1 65 129
2 66 130
3 67 131
62 126 190
63 127 191
3968 4032
3969 4033
3970 4034
3971 4035
4030 4094
4031 4095
0 1 2 3
62 63
Tag No. (6 bits)
4032
4033
4034
4035
128/2 = 64
(6
128/2 = 64 Sets.
(6 bits)
4096/64 = 2^12/2^6 = 2^6
64) 4033 (63
384
193
192
1 → Set No.
Tag No.

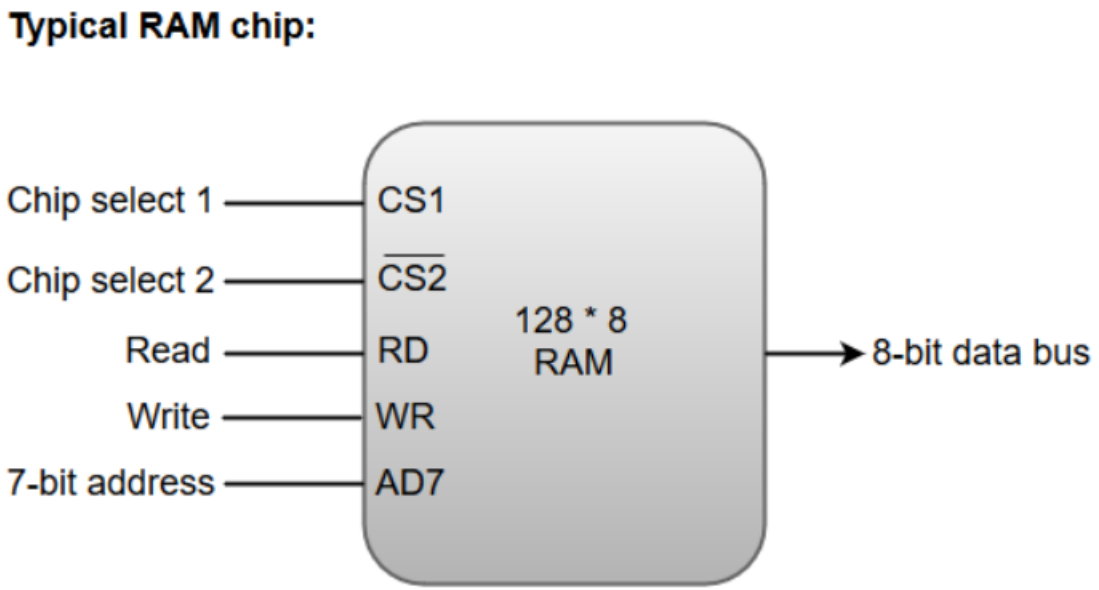
Main Memory

- The main memory acts as the central storage unit in a computer system. It is a relatively large and fast memory which is used to store programs and data during the run time operations.
- RAM integrated circuit chips: The RAM integrated circuit chips are further classified into two possible operating modes, **static** and **dynamic**.
- The primary compositions of a static RAM are flip-flops that store the binary information. The nature of the stored information is volatile, i.e. it remains valid as long as power is applied to the system. The static RAM is easy to use and takes less time performing read and write operations as compared to dynamic RAM.

Main Memory

- The dynamic RAM exhibits the binary information in the form of electric charges that are applied to capacitors. The capacitors are integrated inside the chip by MOS transistors. The dynamic RAM consumes less power and provides large storage capacity in a single memory chip.

RAM Chip



RAM chips are available in a variety of sizes and are used as per the system requirement. The following block diagram demonstrates the chip interconnection in a 128 * 8 RAM chip.

RAM Chip

- A 128 * 8 RAM chip has a memory capacity of 128 words of eight bits (one byte) per word. This requires a 7-bit address and an 8-bit bidirectional data bus.

- The 8-bit bidirectional data bus allows the transfer of data either from memory to CPU during a **read** operation or from CPU to memory during a **write** operation.

- The **read** and **write** inputs specify the memory operation, and the two chip select (CS) control inputs are for enabling the chip only when the microprocessor selects it.

- The bidirectional data bus is constructed using **three-state buffers**.

- The output generated by three-state buffers can be placed in one of the three possible states which include a signal equivalent to logic 1, a signal equal to logic 0, or a high-impedance state.

RAM Chip

The following function table specifies the operations of a 128 * 8 RAM chip.

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	Input data to RAM
1	0	1	x	Read	Output data to RAM
1	1	x	x	Inhibit	High-impedance

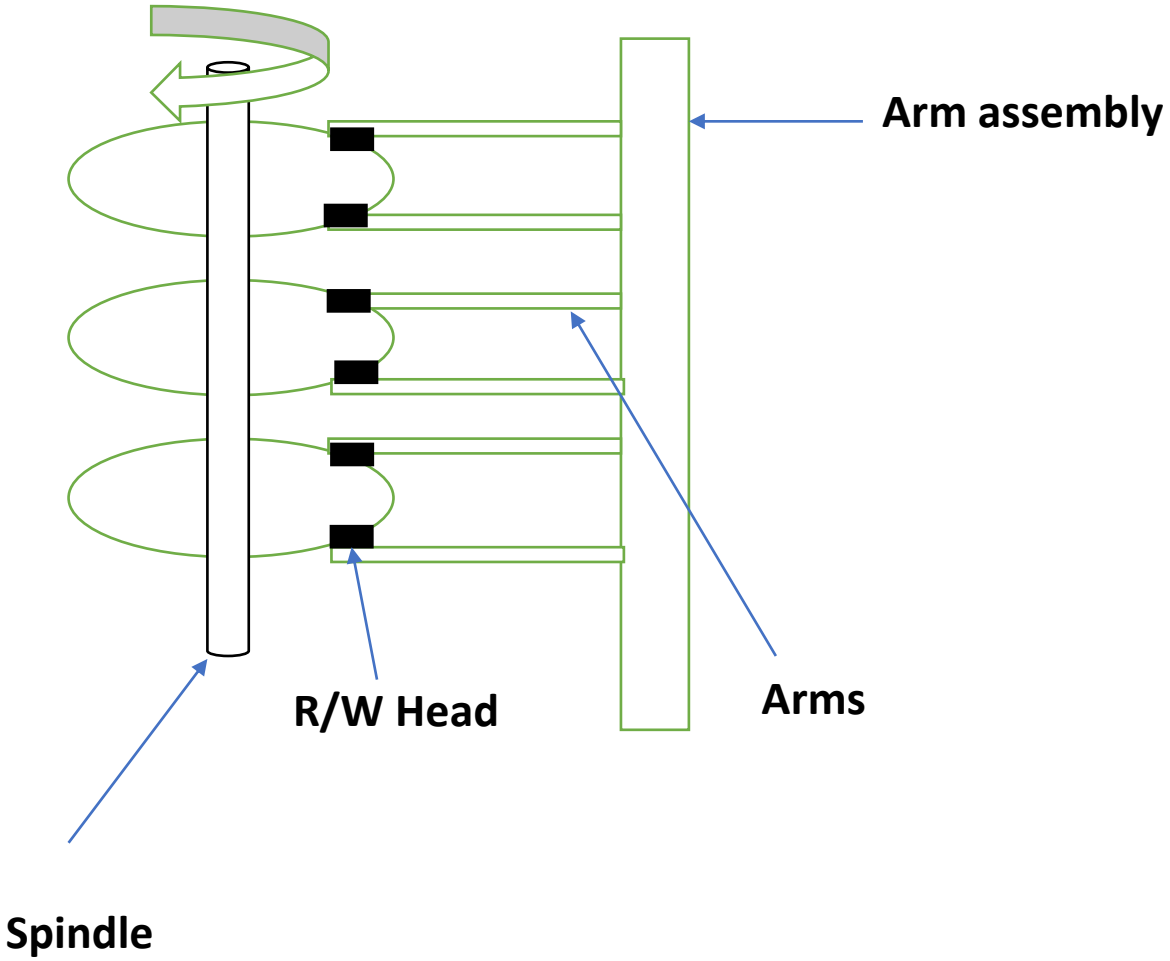
From the functional table, we can conclude that the unit is in operation only when CS1 = 1 and CS2 = 0. The bar on top of the second select variable indicates that this input is enabled when it is equal to 0.

Auxiliary Memory :

- Most common auxiliary memory devices – Magnetic Disks & Tapes.
- The average time required to reach a storage location in memory and obtain its content is called the ***access time***.

- Compact Disk – only one surface for storage
- Magnetic Disk – Both surface store data

- Number of disk are there in Magnetic Disks and it is called as ***platter***.
- 3 platter – 6 side storage available
- In magnetic surface store data is called recording.



Read or Write on the surface then required one small pointer is called **Read/Write head**.

Each Surface have particular R/W Head.

Attach R/W head to one hardware is called as **arms**.

The arms are attached with column wise structure is called **Arms assembly**.

Disk rotation happens in only single direction.

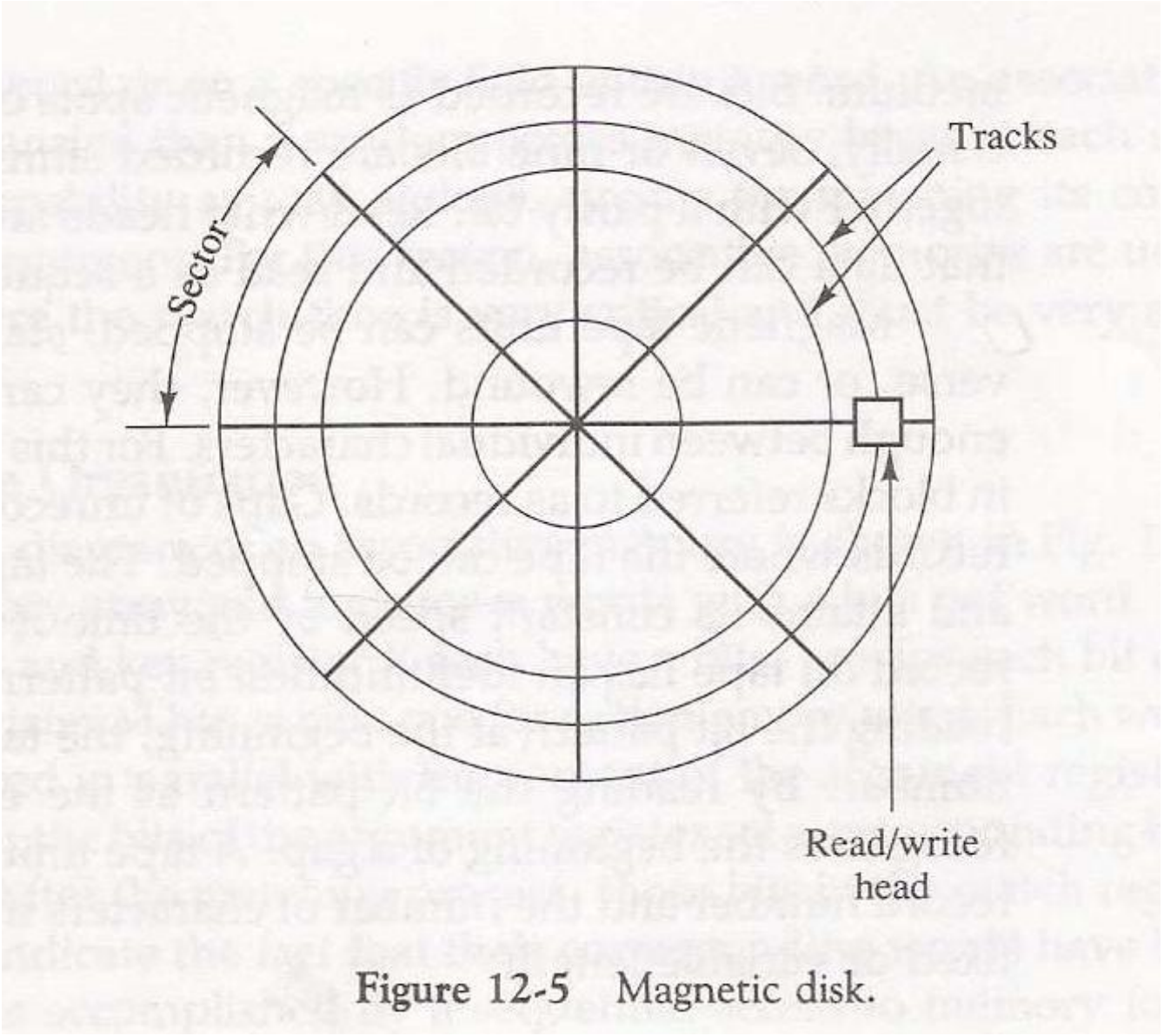


Figure 12-5 Magnetic disk.

Divide such surface in concentrical **track** .

Concentric track because there is only one centre.

Tracks also divided further into **sector.**

Collection of consecutive sector of track is called **cluster.** (only in one track)

- Disk access time = seek time + Rotational latency (delay) + 1 sector transfer time + additional delay

Sector is smallest unit of disk, which can be read or write at onces.

Sector is the addressable unit of the disc.

Disk access time is nothing but 1 sector access time.

All arms are move together. (1 R/W Head is outer most track then all R/W head will be on outer most track)

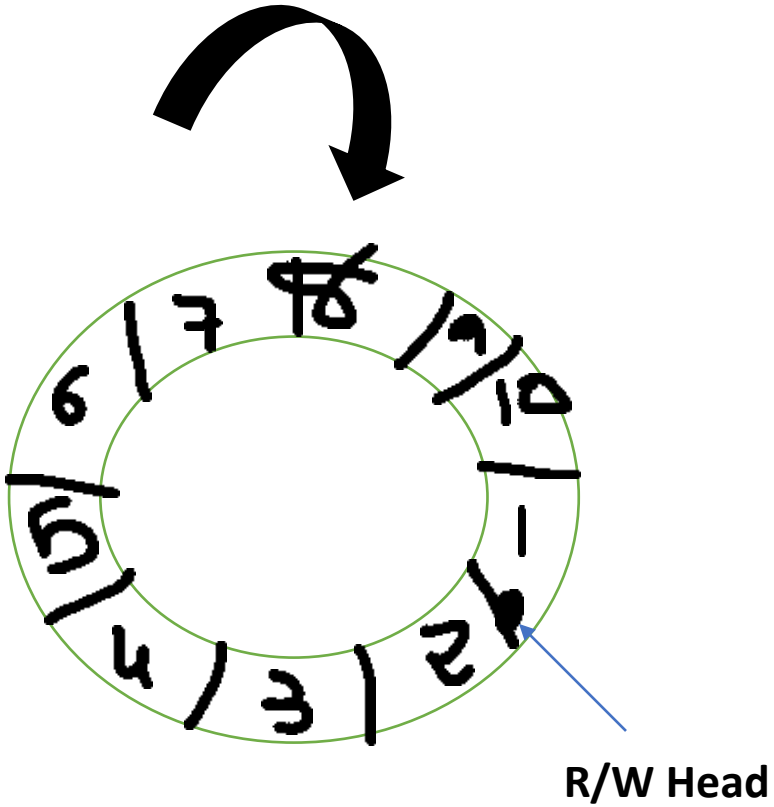
- First R/W head move to particular track.
- Move particular arm forward or backward some time is required. So that R/W head can reach to particular track & this time is called as ***seek time***.
- Then spindle come to a picture.
- The spindle rotate the disk & particular sector comes under R/W head, takes some time is called ***rotational latency (delay***).
- Then read or write happens. So time take is called ***transfer time***.
- Also required some additional delay if given then put otherwise zero.

Disk access time = seek time + Rotational latency (delay) + 1 sector transfer time + additional delay

- **Seek time :** Time required to position the arm over the track.

- **Rotational delay** : Time required to rotate desired sector under R/W head.
- **Transfer Time :** Time required to read or write 1 sector.

- **Additional delay :** Disk controller – hardware delay



R/W head always at end of some sector & Starting of some sector.

In fig now R/W head is at starting sector of 2 & end of sector 1.

- If read sector 2 then no any extra time required.

Note – Disk rotate in single direction.

If read sector 1 then directly not possible – rotate whole disc.