4.3 COMPUTER ORGANIZATION

L	T	P
3	-	-

RATIONALE

The subject provides the students with the knowledge of detailed organization of currently available personal computers in order to understand their functioning . The students will also get familiar with Architecture of multi processor systems.

LEARNING OUTCOMES

After undergoing the subject, students will be able to :

- Use CPU, register and stack.
- Compare micro programmed and hardwired control.
- Compare RISC and CISC architecture.
- Understand memory hierarchy and memory types.
- Explain the function of BIOS.
- Illustrate multi processor systems.

DETAILED CONTENTS

1. Hardware organisation of computer system (16 periods)
   - CPU organisation : general register organisation, stack organisation, instruction formats(three address, two address, one address, zero address and RISC instruction).
     Addressing modes: Immediate, register, direct, in direct, relative, indexed.
   - CPU Design : Microprogrammed vs hard wired control.
   - Reduced instruction set computers: CISC characteristics, RISC characteristics, and their comparison.
2. Memory organisation (14 periods)
   - Memory Hierarchy
   - RAM and ROM chips, Memory address map, Memory connections to CPU.
   - Auxillary memory : Magnetic disks and magnetic tapes.
   - Associative memory
   - Cache memory
   - Virtual memory
   - Memory management hardware
   - Read and Write operation
3. I/O organisation (08 periods)

a. Basis Input output system(BIOS)
   - Function of BIOS
   - Testing and initialization
   - Configuring the system

b. Modes of Data Transfer
   - Programmed I/O : Synchronous, asynchronous and interrupt initiated.
   - DMA data transfer

4. Architecture of multi processor systems (10 periods)

- Forms of parallel processing
- Parallel processing and pipelines, basic characteristics of multiprocessor
- General purpose multiprocessors'
- Interconnection networks : time shared common bus, multi port memory, cross bar switch, multi stage switching networks and hyper cube structures.

INSTRUCTIONAL STRATEGY

Since the subject is theoretical one, the practical aspects should be taught along with the theory instruction. The students be given quiz tests and asked to give seminars on small topics. There is sufficient time in the subject and the students can be taken to laboratory for demonstration.

MEANS OF ASSESSMENT

- Assignments and quiz/class tests, mid-term and end-term written tests
- Viva-voce

LIST OF RECOMENEDED BOOKS

1. Computer Architecture and Organisation by Moris Mano
2. Computer Architecture by J.P.Hayes
3. Structured Computer Organisation By Tanenbaum Andrew S, PHI
4. e-books/e-tools/relevant software to be used as recommended by AICTE/HSBTE/NITTTR.

Websites for Reference:

http://swayam.gov.in

Unit-2

(Computer organization)

Memory Hierarchy

A memory unit is an essential component in any digital computer since it is needed for storing programs and data.

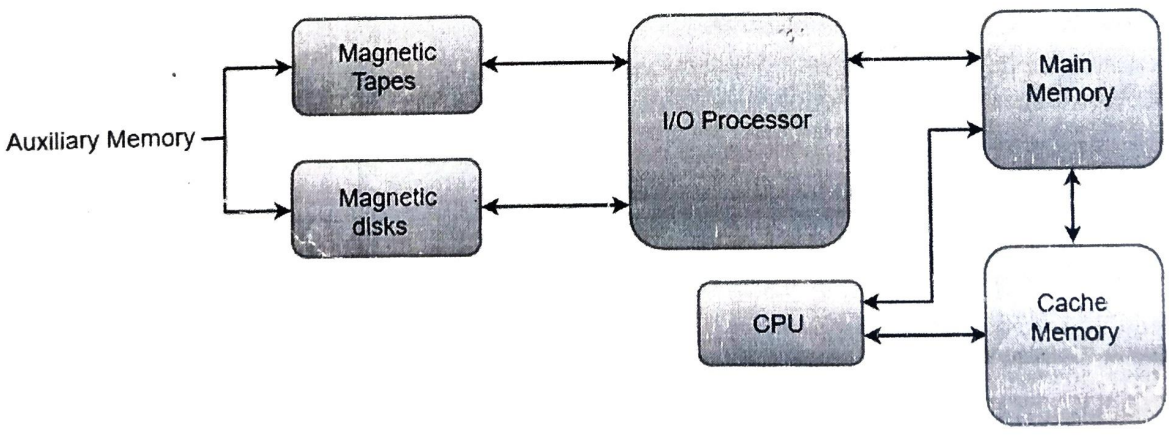
Typically, a memory unit can be classified into two categories:

1. The memory unit that establishes direct communication with the CPU is called **Main Memory**. The main memory is often referred to as RAM (Random Access Memory).
2. The memory units that provide backup storage are called **Auxiliary Memory**. For instance, magnetic disks and magnetic tapes are the most commonly used auxiliary memories.

Apart from the basic classifications of a memory unit, the memory hierarchy consists all of the storage devices available in a computer system ranging from the slow but high-capacity auxiliary memory to relatively faster main memory.

The following image illustrates the components in a typical memory hierarchy.

Memory Hierarchy in a Computer System:



Auxiliary Memory

Auxiliary memory is known as the lowest-cost, highest-capacity and slowest-access storage in a computer system. Auxiliary memory provides storage for programs and

data that are kept for long-term storage or when not in immediate use. The most common examples of auxiliary memories are magnetic tapes and magnetic disks.

A magnetic disk is a digital computer memory that uses a magnetization process to write, rewrite and access data. For example, hard drives, zip disks, and floppy disks.

Magnetic tape is a storage medium that allows for data archiving, collection, and backup for different kinds of data.

Main Memory

The main memory in a computer system is often referred to as **Random Access Memory (RAM)**. This memory unit communicates directly with the CPU and with auxiliary memory devices through an I/O processor.

The programs that are not currently required in the main memory are transferred into auxiliary memory to provide space for currently used programs and data.

I/O Processor

The primary function of an I/O Processor is to manage the data transfers between auxiliary memories and the main memory.

Cache Memory

The data or contents of the main memory that are used frequently by CPU are stored in the cache memory so that the processor can easily access that data in a shorter time. Whenever the CPU requires accessing memory, it first checks the required data into the cache memory. If the data is found in the cache memory, it is read from the fast memory. Otherwise, the CPU moves onto the main memory for the required data.

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.

The primary technology used for the main memory is based on semiconductor integrated circuits. The integrated circuits for the main memory are classified into two major units.

1. RAM (Random Access Memory) integrated circuit chips

2. ROM (Read Only Memory) integrated circuit chips

RAM integrated circuit chips

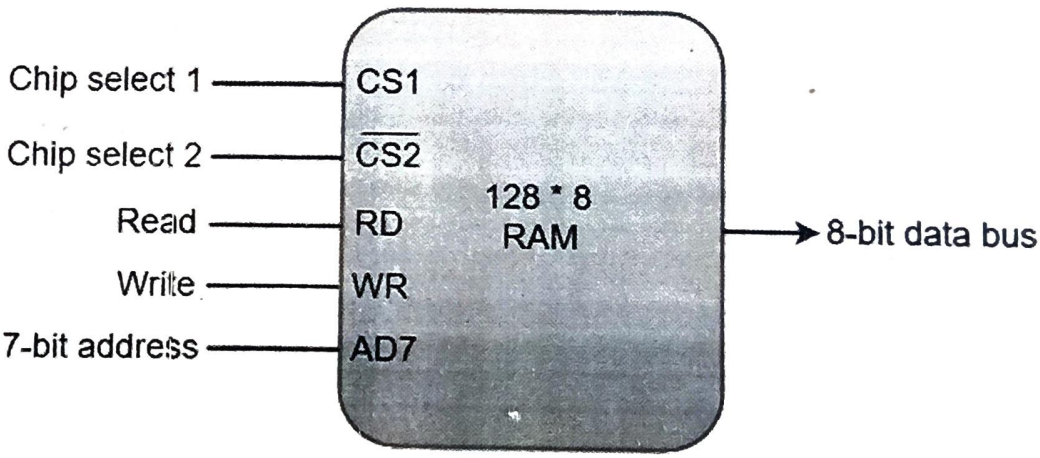
The RAM integrated circuit chips are further classified into two possible operating modes, **static** and **dynamic**.

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.

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 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.

Typical 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.

Note: The logic 1 and 0 are standard digital signals whereas the high-impedance state behaves like an open circuit, which means that the output does not carry a signal and has no logic significance.

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 $\overline{CS2}$ = 0. The bar on top of the second select variable indicates that this input is enabled when it is equal to 0.

ROM integrated circuit

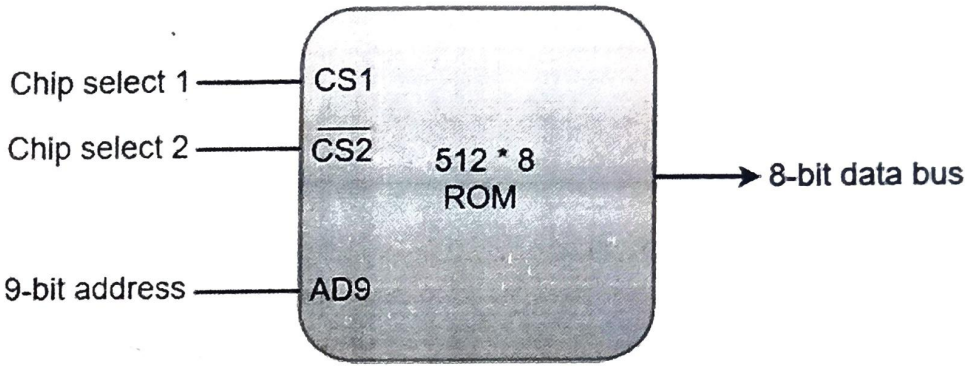
The primary component of the main memory is RAM integrated circuit chips, but a portion of memory may be constructed with ROM chips.

A ROM memory is used for keeping programs and data that are permanently resident in the computer.

Apart from the permanent storage of data, the ROM portion of main memory is needed for storing an initial program called a **bootstrap loader**. The primary function of the **bootstrap loader** program is to start the computer software operating when power is turned on.

ROM chips are also available in a variety of sizes and are also used as per the system requirement. The following block diagram demonstrates the chip interconnection in a 512 * 8 ROM chip.

Typical ROM chip:



- A ROM chip has a similar organization as a RAM chip. However, a ROM can only perform read operation; the data bus can only operate in an output mode.
- The 9-bit address lines in the ROM chip specify any one of the 512 bytes stored in it.
- The value for chip select 1 and chip select 2 must be 1 and 0 for the unit to operate. Otherwise, the data bus is said to be in a high-impedance state.

Memory Address Map

- **The designer** of a computer system must calculate the amount of memory required for the particular application and assign it to either RAM or ROM.
- **The interconnection** between memory and processor is then established from knowledge of the size of memory needed and the type of RAM and ROM chips available.
- **The addressing** of memory can be established by means of a table that specifies the memory address assigned to each chip.
- **The table**, called a memory address map, is a pictorial representation of assigned address space for each chip in the system.
- **To demonstrate** with a particular example, assume that a computer system needs 512 bytes of RAM and 512 bytes of ROM. The RAM and ROM chips to be used are specified in Figs. 2 and 3.
- **The memory address** map for this configuration is shown in Table 1.

TABLE 1 Memory Address Map for Microprocomputer

Component	Hexadecimal address	Address bus 10	9	8	7	6	5	4	3	2	1
RAM 1	0000–007F	0	0	0	x	x	x	x	x	x	x
RAM 2	0080–00FF	0	0	1	x	x	x	x	x	x	x
RAM 3	0100–017F	0	1	0	x	x	x	x	x	x	x
RAM 4	0180–01FF	0	1	1	x	x	x	x	x	x	x
ROM	0200–03FF	1	x	x	x	x	x	x	x	x	x

- **The component column** specifies whether a RAM or a ROM chip is used. The hexadecimal address column assigns a range of hexadecimal equivalent addresses for each chip.
- **The address** bus lines are listed in the third column. Although there are 16 lines in the address bus, the table shows only 10 lines because the other 6 are not used in this example and are assumed to be zero.
- **The small x's** under the address bus lines designate those lines that must be connected to the address inputs in each chip. The RAM chips have 128 bytes and need seven address lines.
- **The ROM chip has 512** bytes and needs 9 address lines. The x's are always assigned to the low-order bus lines: lines 1 through 7 for the RAM and lines 1 through 9 for the ROM.
- **It is now necessary** to distinguish between four RAM chips by assigning to each a different address. For this particular example we choose bus lines 8 and 9 to represent four distinct binary combinations.
- **Note that any other** pair of unused bus lines can be chosen for this purpose. The table clearly shows that the nine low-order bus lines constitute a memory space for RAM equal to 29 = 512 bytes.
- **The distinction** between a RAM and ROM address is done with another bus line. Here we choose line 10 for this purpose. When line 10 is 0, the CPU selects a RAM, and when this line is equal to 1, it selects the ROM.
- **The equivalent** hexadecimal address for each chip is obtained from the information under the address bus assignment. The address bus lines are subdivided into groups of four bits each so that each group can be represented with a hexadecimal digit.
- **The first hexadecimal** digit represents lines 13 to 16 and is always 0. The next hexadecimal digit represents lines 9 to 12, but lines 11 and 12 are always 0.
- **The range of hexadecimal** addresses for each component is determined from the x's associated with it. These x's represent a binary number that can range from an all-0's to an all-1's value.

Memory Connection to CPU

- **RAM and ROM chips** are connected to a CPU through the data and address buses.

- **The low-order lines** in the address bus select the byte within the chips and other lines in the address bus select a particular chip through its chip select inputs.
- **The connection of memory** chips to the CPU is shown in Fig. 4. This configuration gives a memory capacity of 512 bytes of RAM and 512 bytes of ROM. It implements the memory map of Table 1. Each RAM receives the seven low-order bits of the address bus to select one of 128 possible bytes.
- **The particular RAM chip** selected is determined from lines 8 and 9 in the address bus. This is done through a 2 x 4 decoder whose outputs go to the CS1 inputs in each RAM chip.
- **Thus, when address** lines 8 and 9 are equal to 00, the first RAM chip is selected. When 01, the second RAM chip is selected, and so on. The RD and WR outputs from the microprocessor are applied to the inputs of each RAM chip.
- **The selection between** RAM and ROM is achieved through bus line 10. The RAMs are selected when the bit in this line is 0, and the ROM when the bit is 1.
- **The other chip select** input in the ROM is connected to the RD control line for the ROM chip to be enabled only during a read operation. Address bus lines 1 to 9 are applied to the input address of ROM without going through the decoder.
- **This assigns addresses** 0 to 511 to RAM and 512 to 1023 to ROM. The data bus of the ROM has only an output capability, whereas the data bus connected to the RAMs can transfer information in both directions .
- **The example just shown** gives an indication of the interconnection complexity that can exist between memory chips and the CPU.
- **The more chips that** are connected, the more external decoders are required for selection among the chips . The designer must establish a memory map that assigns addresses to the various chips from which the required connections are determined.

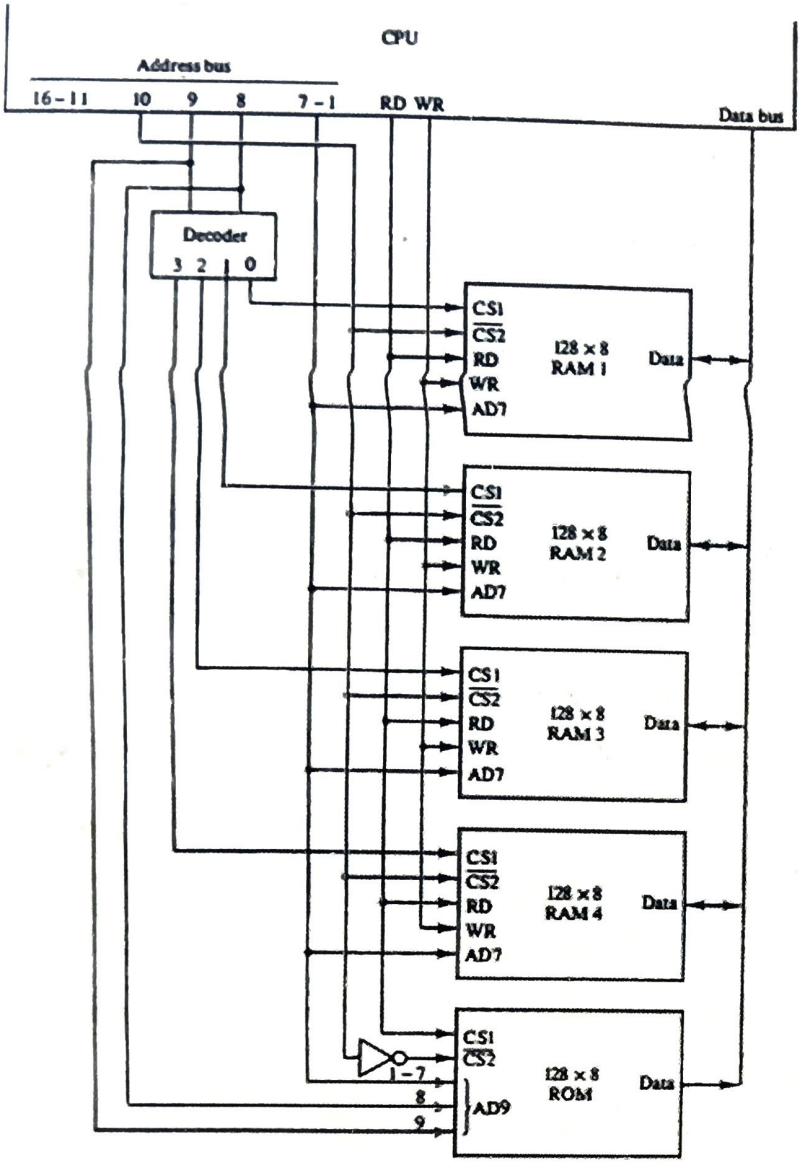


Figure 4 Memory connection to the CPU.

Auxiliary Memory

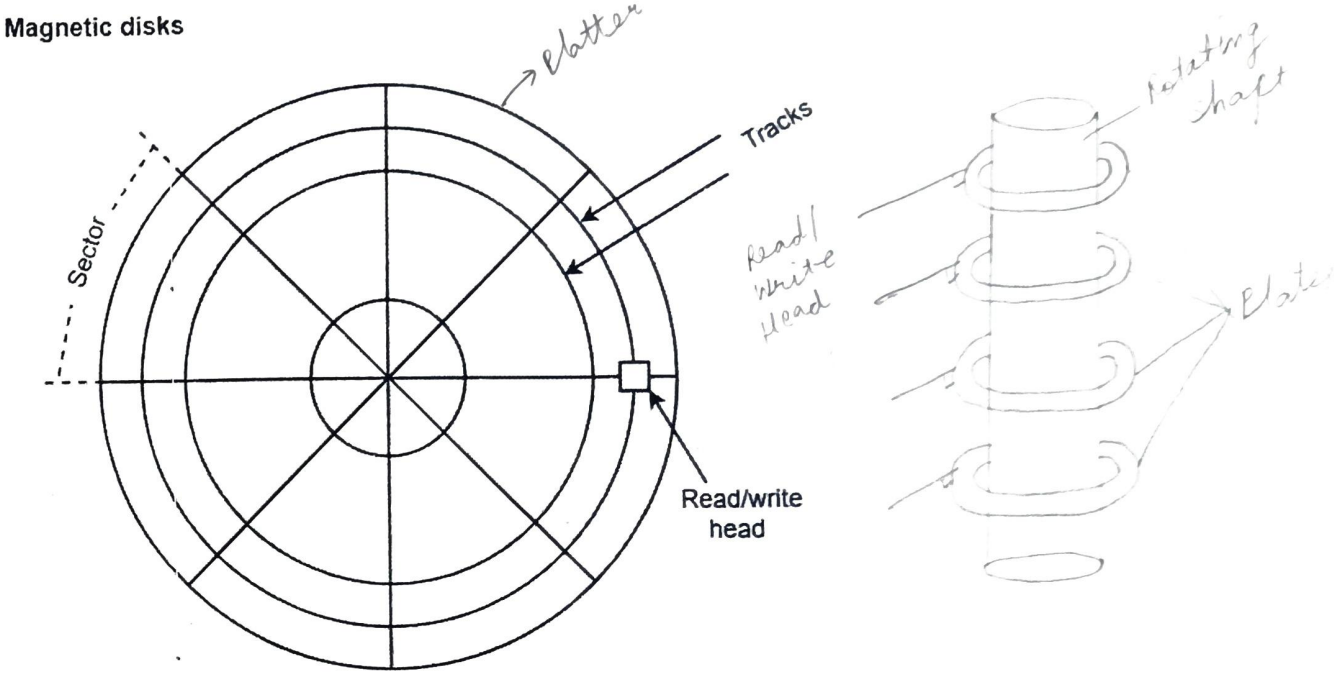
An Auxiliary memory is known as the lowest-cost, highest-capacity and slowest-access storage in a computer system. It is where programs and data are kept for long-term storage or when not in immediate use. The most common examples of auxiliary memories are magnetic tapes and magnetic disks.

Magnetic Disks

A magnetic disk is a type of memory constructed using a circular plate of metal or plastic coated with magnetized materials. Usually, both sides of the disks are used to

carry out read/write operations. However, several disks may be stacked on one spindle with read/write head available on each surface.

The following image shows the structural representation for a magnetic disk.



- The memory bits are stored in the magnetized surface in spots along the concentric circles called tracks.
- The concentric circles (tracks) are commonly divided into sections called sectors.

Magnetic Tape

Magnetic tape is a storage medium that allows data archiving, collection, and backup for different kinds of data. The magnetic tape is constructed using a plastic strip coated with a magnetic recording medium.

The bits are recorded as magnetic spots on the tape along several tracks. Usually, seven or nine bits are recorded simultaneously to form a character together with a parity bit.

Magnetic tape units can be halted, started to move forward or in reverse, or can be rewound. However, they cannot be started or stopped fast enough between individual characters. For this reason, information is recorded in blocks referred to as records.

Associative Memory

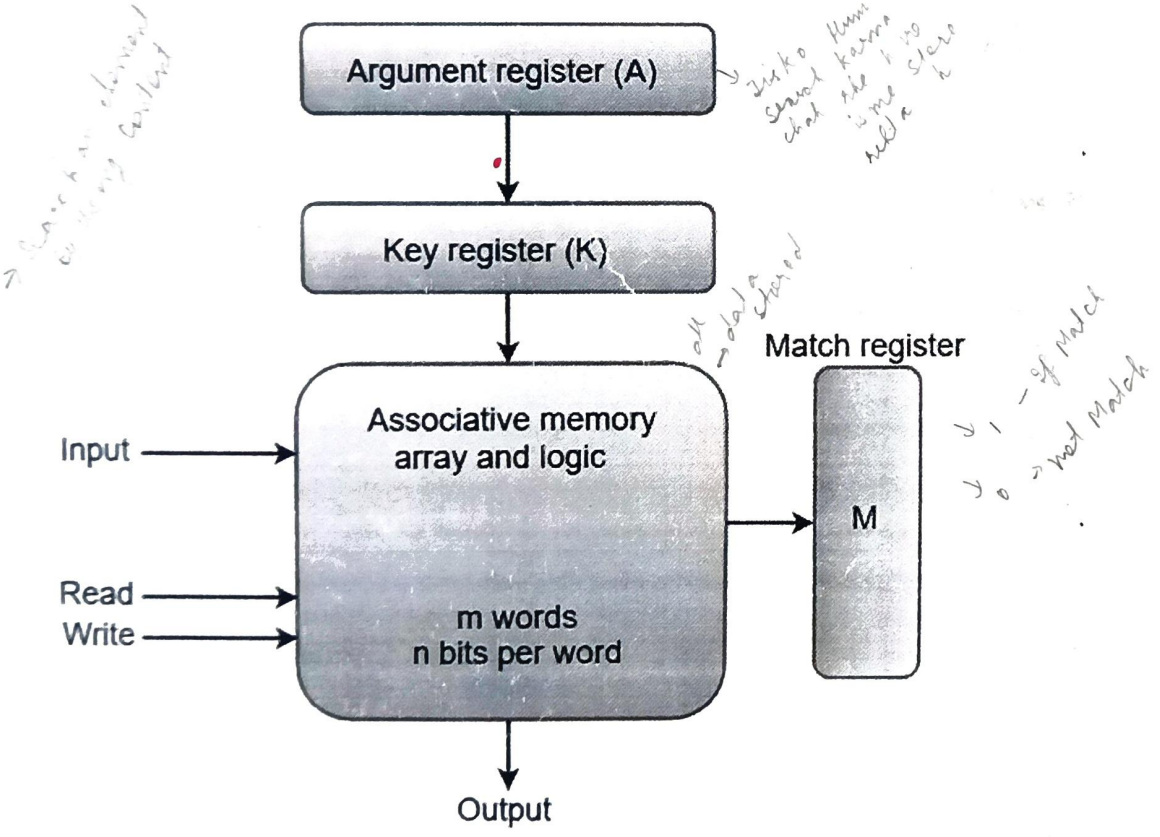
An associative memory can be considered as a memory unit whose stored data can be identified for access by the content of the data itself rather than by an address or memory location.

Associative memory is often referred to as **Content Addressable Memory (CAM)**.

When a write operation is performed on associative memory, no address or memory location is given to the word. The memory itself is capable of finding an empty unused location to store the word.

On the other hand, when the word is to be read from an associative memory, the content of the word, or part of the word, is specified. The words which match the specified content are located by the memory and are marked for reading.

The following diagram shows the block representation of an Associative memory.



From the block diagram, we can say that an associative memory consists of a memory array and logic for 'm' words with 'n' bits per word.

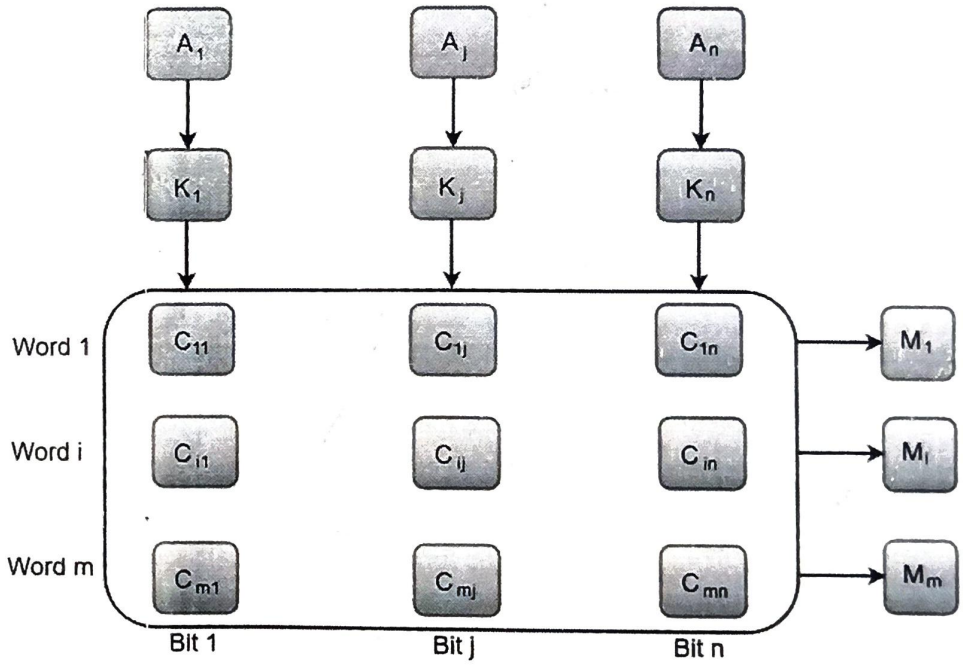
The functional registers like the argument register **A** and key register **K** each have **n** bits, one for each bit of a word. The match register **M** consists of **m** bits, one for each memory word.

The words which are kept in the memory are compared in parallel with the content of the argument register.

The key register (K) provides a mask for choosing a particular field or key in the argument word. If the key register contains a binary value of all 1's, then the entire argument is compared with each memory word. Otherwise, only those bits in the argument that have 1's in their corresponding position of the key register are compared. Thus, the key provides a mask for identifying a piece of information which specifies how the reference to memory is made.

The following diagram can represent the relation between the memory array and the external registers in an associative memory.

Associative memory of m word, n cells per word:



The cells present inside the memory array are marked by the letter C with two subscripts. The first subscript gives the word number and the second specifies the bit position in the word. For instance, the cell C_{ij} is the cell for bit **j** in word **i**.

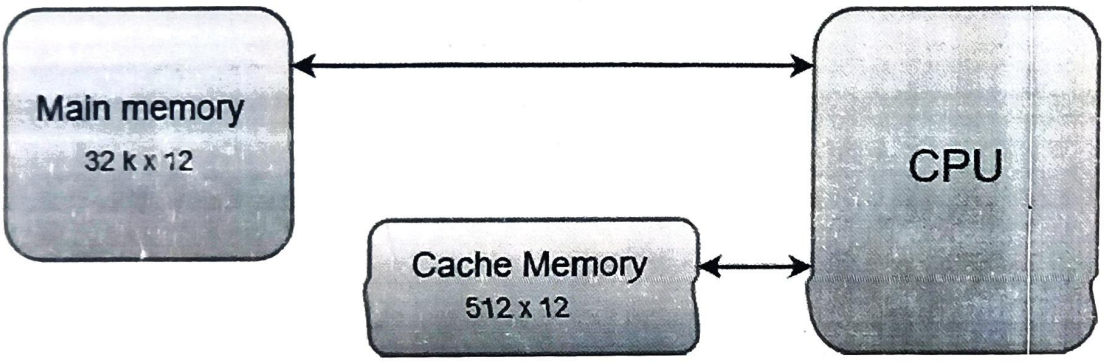
A bit A_j in the argument register is compared with all the bits in column **j** of the array provided that $K_j = 1$. This process is done for all columns **j** = 1, 2, 3....., n.

If a match occurs between all the unmasked bits of the argument and the bits in word **i**, the corresponding bit M_i in the match register is set to 1. If one or more unmasked bits of the argument and the word do not match, M_i is cleared to 0.

Cache Memory

The data or contents of the main memory that are used frequently by CPU are stored in the cache memory so that the processor can easily access that data in a shorter time. Whenever the CPU needs to access memory, it first checks the cache memory. If the data is not found in cache memory, then the CPU moves into the main memory.

Cache memory is placed between the CPU and the main memory. The block diagram for a cache memory can be represented as:



The cache is the fastest component in the memory hierarchy and approaches the speed of CPU components.

The basic operation of a cache memory is as follows:

- 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.
- A block of words 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.
- The performance of the 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 **hit**.

- If the word is not found in the cache, it is in main memory and it counts as a **miss**.
- The ratio of the number of hits divided by the total CPU references to memory (hits plus misses) is the hit ratio.

memory management

Memory management is the process of controlling and coordinating computer memory, assigning portions called blocks to various running programs to optimize overall system performance. Memory management resides in hardware, in the OS (operating system), and in programs and applications.

In hardware, memory management involves components that physically store data, such as RAM (random access memory) chips, memory caches, and flash-based SSDs (solid-state drives). In the OS, memory management involves the allocation (and constant reallocation) of specific memory blocks to individual programs as user demands change. At the application level, memory management ensures the availability of adequate memory for the objects and data structures of each running program at all times. Application memory management combines two related tasks, known as allocation and recycling.

- When the program requests a block of memory, a part of the memory manager called the allocator assigns that block to the program.
- When a program no longer needs the data in previously allocated memory blocks, those blocks become available for reassignment. This task can be done manually (by the programmer) or automatically (by the memory manager).

Virtual Memory

Virtual Memory is a storage scheme that provides user an illusion of having a very big main memory. This is done by treating a part of secondary memory as the main memory.

In this scheme, User can load the bigger size processes than the available main memory by having the illusion that the memory is available to load the process.

Instead of loading one big process in the main memory, the Operating System loads the different parts of more than one process in the main memory.

By doing this, the degree of multiprogramming will be increased and therefore, the CPU utilization will also be increased.

How Virtual Memory Works?

In modern word, virtual memory has become quite common these days. In this scheme, whenever some pages needs to be loaded in the main memory for the execution and the memory is not available for those many pages, then in that case, instead of stopping the pages from entering in the main memory, the OS search for the RAM area that are least used in the recent times or that are not referenced and copy that into the secondary memory to make the space for the new pages in the main memory.

Since all this procedure happens automatically, therefore it makes the computer feel like it is having the unlimited RAM.

Demand Paging

Demand Paging is a popular method of virtual memory management. In demand paging, the pages of a process which are least used, get stored in the secondary memory.

A page is copied to the main memory when its demand is made or page fault occurs. There are various page replacement algorithms which are used to determine the pages which will be replaced. We will discuss each one of them later in detail.

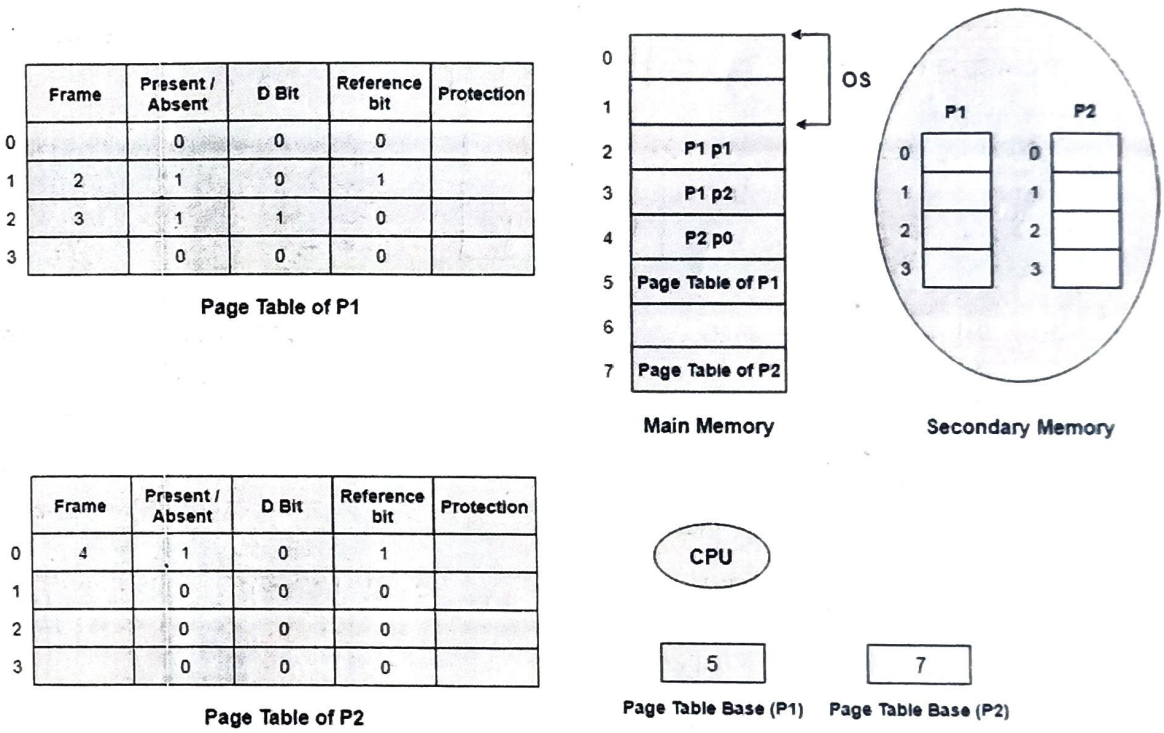
Snapshot of a virtual memory management system

Let us assume 2 processes, P1 and P2, contains 4 pages each. Each page size is 1 KB. The main memory contains 8 frame of 1 KB each. The OS resides in the first two

partitions. In the third partition, 1st page of P1 is stored and the other frames are also shown as filled with the different pages of processes in the main memory.

The page tables of both the pages are 1 KB size each and therefore they can be fit in one frame each. The page tables of both the processes contain various information that is also shown in the image.

The CPU contains a register which contains the base address of page table that is 5 in the case of P1 and 7 in the case of P2. This page table base address will be added to the page number of the Logical address when it comes to accessing the actual corresponding entry.



Advantages of Virtual Memory

1. The degree of Multiprogramming will be increased.
2. User can run large application with less real RAM.
3. There is no need to buy more memory RAMs.

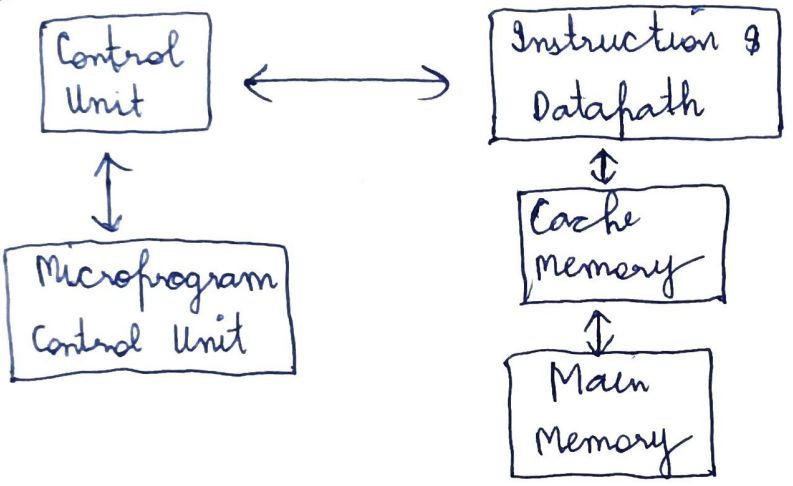
Disadvantages of Virtual Memory

1. The system becomes slower since swapping takes time.
2. It takes more time in switching between applications.
3. The user will have the lesser hard disk space for its use.

Characteristics of CISC -

CISC :- CISC Stands for Complex Instruction Set Computer. To Minimize the no. of Instructions Program and ignoring the no. of cycles per instructions. It is used in desktops, laptops. In this, less Memory (RAM) is Required to store Instructions.

Architecture :-



characteristics :-

1. Many addressing mode.
2. large no. of Instruction.
3. Variable length of Instruction format.
4. Several cycles may Be Required to execute one Instruction.
5. Instruction decoding logic is Complesc.

Cache Memory-

Cache memory bridges the speed mismatch between the processor and the main memory.

When cache hit occurs,

- The required word is present in the cache memory.
- The required word is delivered to the CPU from the cache memory.

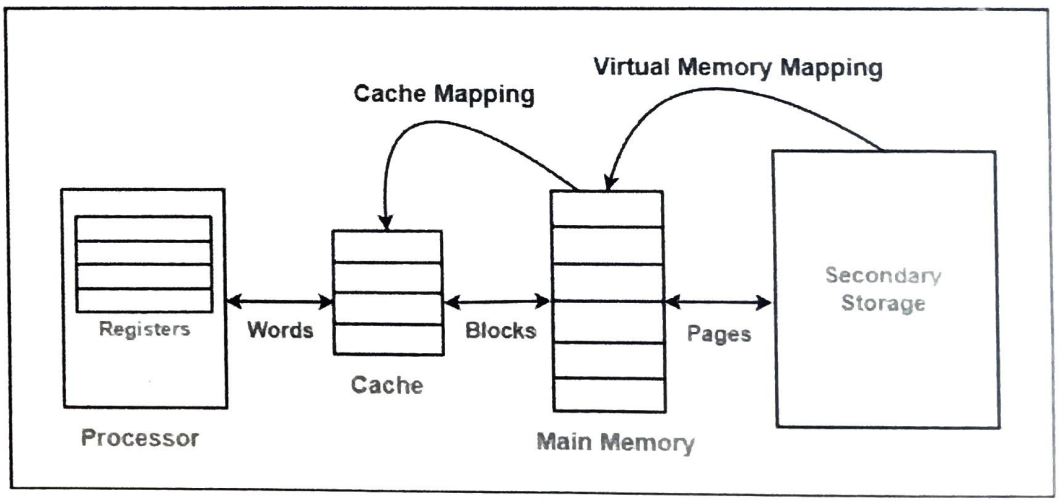
When cache miss occurs,

- The required word is not present in the cache memory.
- The page containing the required word has to be mapped from the main memory.
- This mapping is performed using cache mapping techniques.

Cache Mapping-

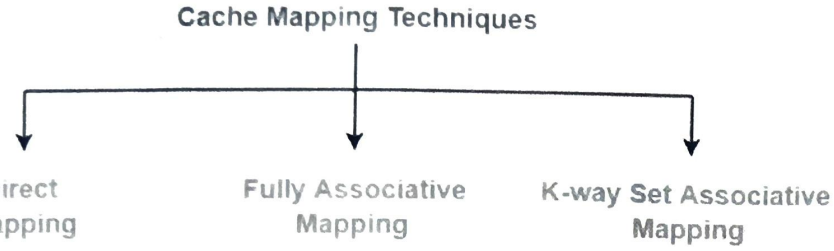
- Cache mapping is a technique by which the contents of main memory are brought into the cache memory.

The following diagram illustrates the mapping process-



Cache Mapping Techniques-

Cache mapping is performed using following three different techniques-



1. Direct Mapping
2. Fully Associative Mapping
3. K-way Set Associative Mapping

1. Direct Mapping-

In direct mapping,

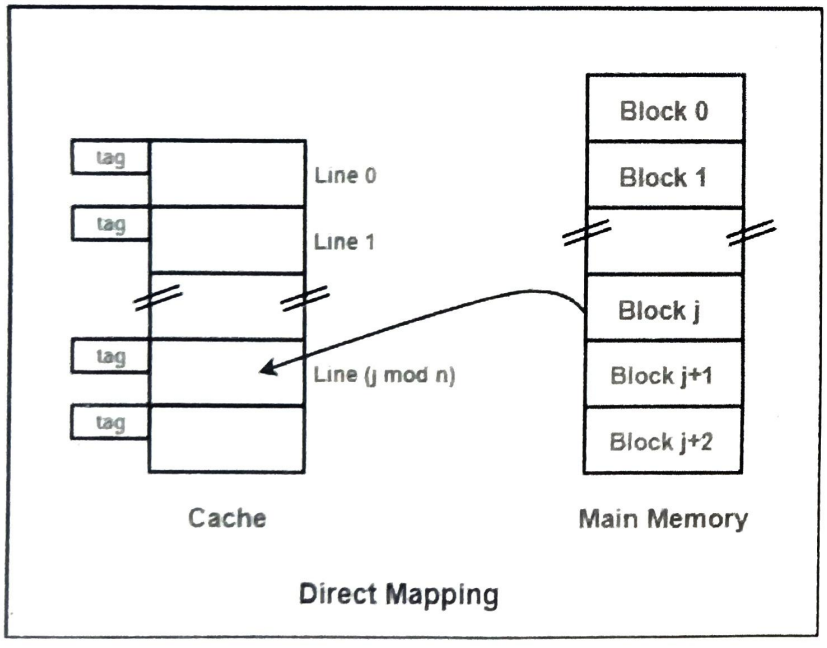
- A particular block of main memory can map only to a particular line of the cache.
- The line number of cache to which a particular block can map is given by-

Cache line number

= (Main Memory Block Address) Modulo (Number of lines in Cache)

Example-

- Consider cache memory is divided into 'n' number of lines.
- Then, block 'j' of main memory can map to line number (j mod n) only of the cache.



Direct Mapping

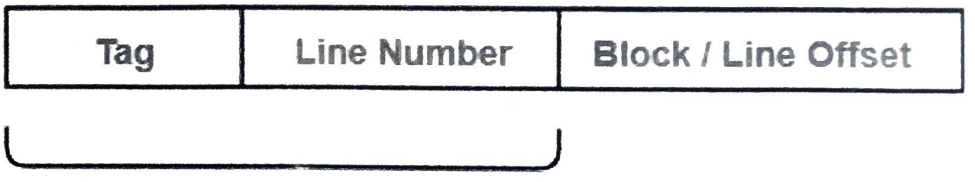
Need of Replacement Algorithm-

In direct mapping,

- There is no need of any replacement algorithm.
- This is because a main memory block can map only to a particular line of the cache.
- Thus, the new incoming block will always replace the existing block (if any) in that particular line.

Division of Physical Address-

In direct mapping, the physical address is divided as-



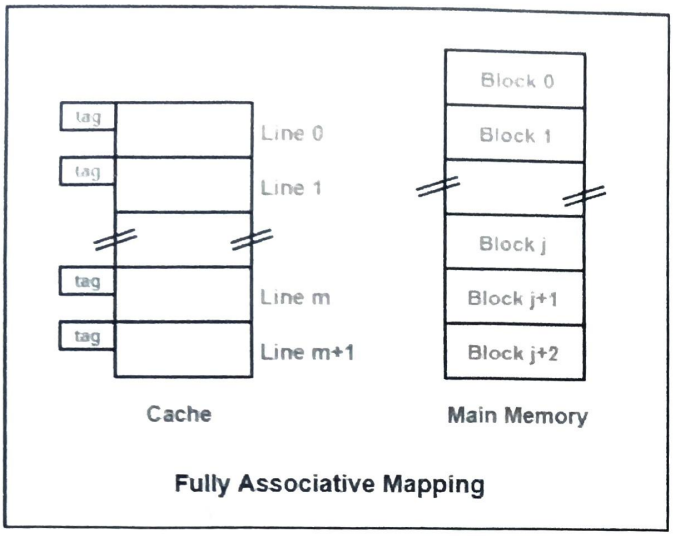
Division of Physical Address in Direct Mapping

2. Fully Associative Mapping-

- A block of main memory can map to any line of the cache that is freely available at that moment.
- This makes fully associative mapping more flexible than direct mapping.

Example-

Consider the following scenario-



Fully Associative Mapping

Here,

- All the lines of cache are freely available.
- Thus, any block of main memory can map to any line of the cache.
- Had all the cache lines been occupied, then one of the existing blocks will have to be replaced.

Need of Replacement Algorithm-

In fully associative mapping,

- A replacement algorithm is required.
- Replacement algorithm suggests the block to be replaced if all the cache lines are occupied.
- Thus, replacement algorithm like FCFS Algorithm, LRU Algorithm etc is employed.

Division of Physical Address-

In fully associative mapping, the physical address is divided as-

Block Number / Tag	Block / Line Offset

Division of Physical Address in Fully Associative Mapping

3. K-way Set Associative Mapping-

In k-way set associative mapping,

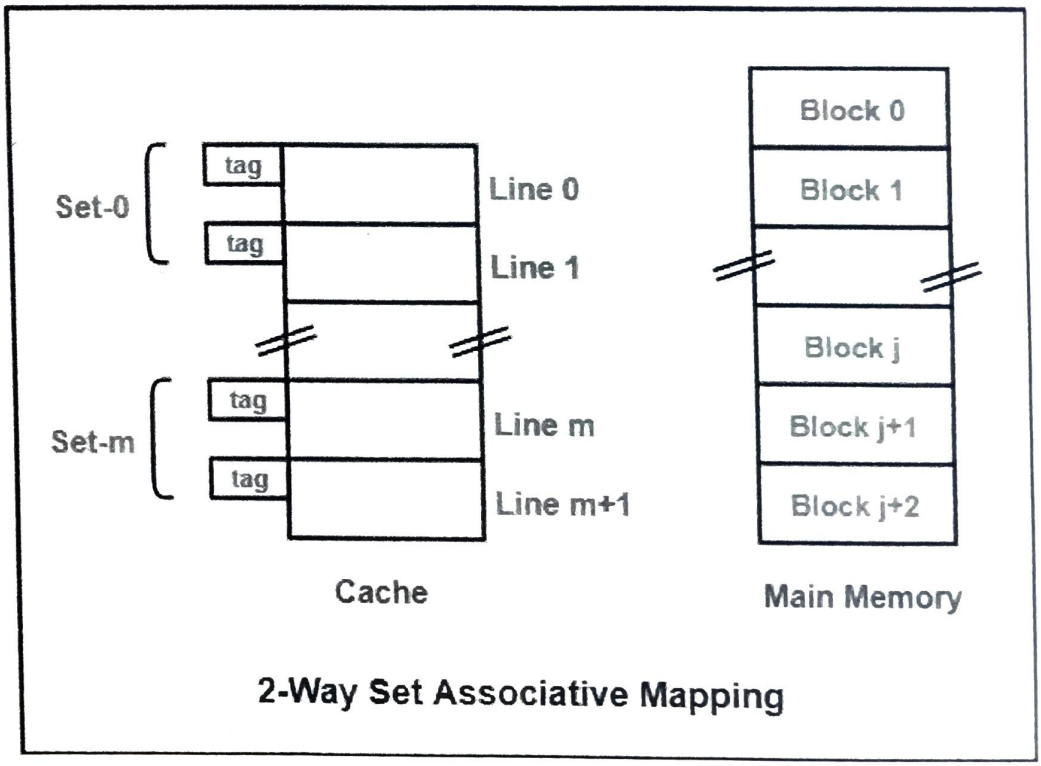
- Cache lines are grouped into sets where each set contains k number of lines.
- A particular block of main memory can map to only one particular set of the cache.
- However, within that set, the memory block can map any cache line that is freely available.
- The set of the cache to which a particular block of the main memory can map is given by-

Cache set number

= (Main Memory Block Address) Modulo (Number of sets in Cache)

Example-

Consider the following example of 2-way set associative mapping-



2-Way Set Associative Mapping

Here,

- k = 2 suggests that each set contains two cache lines.
- Since cache contains 6 lines, so number of sets in the cache = 6 / 2 = 3 sets.
- Block 'j' of main memory can map to set number (j mod 3) only of the cache.
- Within that set, block 'j' can map to any cache line that is freely available at that moment.

- If all the cache lines are occupied, then one of the existing blocks will have to be replaced.

Need of Replacement Algorithm-

- Set associative mapping is a combination of direct mapping and fully associative mapping.
- It uses fully associative mapping within each set.
- Thus, set associative mapping requires a replacement algorithm.

Division of Physical Address-

In set associative mapping, the physical address is divided as-

Division of Physical Address in K-way Set Associative Mapping