CHAPTER 9
MEMORY ORGANIZATION
BSc. CSIT 3rd Semester

Contents

Memory Organization LH 4

9.1 Introduction: Memory Hierarchy, Main Memory, RAM and ROM Chips, Memory address Map, Memory Connection to CPU, Auxiliary Memory (magnetic Disk, Magnetic Tape)

9.2 Associative Memory: Hardware Organization, Match Logic, Read Operation, Write Operation

9.3 Cache Memory: Locality of Reference, Hit & Miss Ratio, Mapping, Write Policies

Memory Hierarchy

- Memory unit is essential component of digital computer since it is needed for storing programs and data.
- Memory unit that communicates directly with CPU is called Main memory.
- Devices that provide backup storage is called auxiliary memory.
- Only programs and data currently needed by processor reside in the main memory.
- All other information is stored in auxiliary memory and transferred to main memory when needed.

Table 4.1 Key Characteristics of Computer Memory Systems

Location

- Internal (e.g. processor registers, main memory, cache)
- External (e.g. optical disks, magnetic disks, tapes)

Capacity

- Number of words
- Number of bytes

Unit of Transfer

- Word
- Block

Access Method

- Sequential
- Direct
- Random
- Associative

Performance

- Access time
- Cycle time
- Transfer rate

Physical Type

- Semiconductor
- Magnetic
- Optical
- Magneto-optical

Physical Characteristics

- Volatile/nonvolatile
- Erasable/nonerasable

Organization

- Memory modules

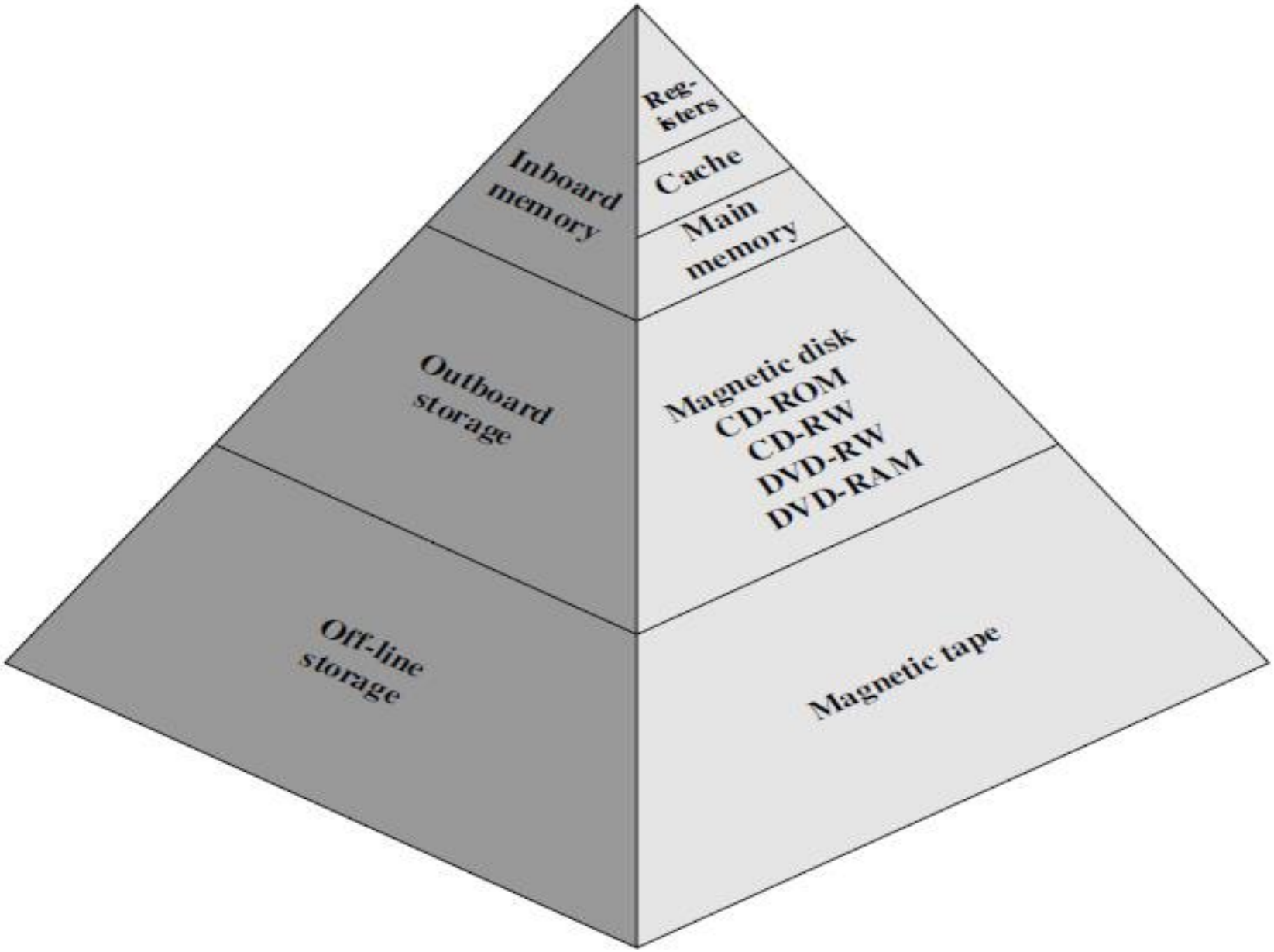


Figure 4.1 The Memory Hierarchy

- Memory hierarchy system consist of all storage devices from auxiliary memory to main memory to cache memory
- As one goes down the hierarchy :
  - Cost per bit decreases.
  - Capacity increases.
  - Access time increases.
  - Frequency of access by the processor decreases.

Primary and Secondary Memory

To identify the behavior of various memories certain characteristics are considered. These are as follows-

Memory types : On the basis of their location inside the computer, memory can be placed in four groups :

Continued…

- **CPU Registers**: these high speed registers in the CPU work as memory for temporary storage of instruction and data. The data can be read from or written into a register within a single clock cycle.
- **Main Memory or Primary Memory**: Main memory size is large and fast accessing external memory stores programs and data. This memory is slower compared to CPU registers because of main memory has large storage capacity is typically 1 and 2^{10} megabyte.

Continued…

- **Secondary Memory**: This memory has larger in capacity but slower than main memory. Secondary memory stores system programs, large data files and like the data are not continually required by the CPU. It also acts as an overflow memory when the capacity of the main memory is exceeded. Information in secondary storage is accessed indirectly via input output processor that transfer information between main and secondary memory.

Auxiliary Memory

- Also called as Secondary Memory, used to store large chunks of data at a lesser cost per byte than a primary memory for backup.
- It does not lose the data when the device is powered down—it is non-volatile.
- It is not directly accessible by the CPU, they are accessed via the input/output channels.
- The most common form of auxiliary memory devices used in consumer systems is flash memory, optical discs, and magnetic disks, magnetic tapes.

Types of Auxiliary Memory

- *Flash memory:* An electronic non-volatile computer storage device that can be electrically erased and reprogrammed, and works without any moving parts. Examples of this are **USB flash drives** and **solid state drives**.

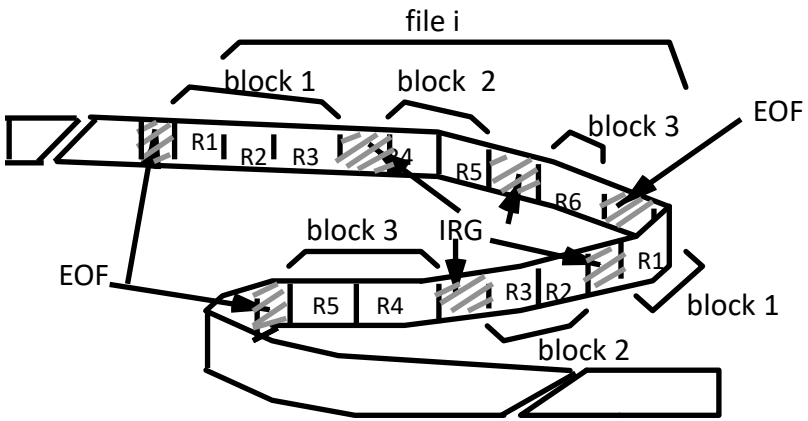
- *Optical disc:* Its a storage medium from which data is read and to which it is written by lasers. There are three basic types of optical disks: CD-ROM (read-only), WORM (write-once read-many) & EO (erasable optical disks).

Types of Auxiliary Memory

- *Magnetic tapes:* A magnetic tape consists of electric, mechanical and electronic components to provide the parts and control mechanism for a magnetic tape unit.

- The tape itself is a strip of plastic coated with a magnetic recording medium. Bits are recorded as magnetic spots on tape along several tracks called **RECORDS.**

- Each record on tape has an identification bit pattern at the beg. and the end.



- R/W heads are mounted in each track so that data can be recorded and read as a sequence of characters.
- Can be stopped , started to move forward, or in reverse, or can be rewound, but cannot be stopped fast enough between individual characters.

Types of Auxiliary Memory

Magnetic Disk:

- A magnetic disk is a circular plate constructed of metal or plastic coated with magnetized material.
- Both sides of the disk are used and several disks may be stacked on one spindle with read/write heads available on each surface.
- Bits are stored in magnetized surface in spots along concentric circles called tracks. Tracks are commonly divided into sections called sectors.
- Disk that are permanently attached and cannot removed by occasional user are called hard disks.

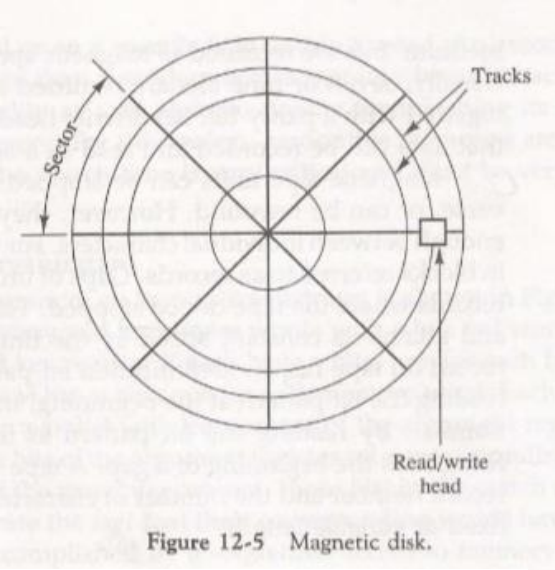
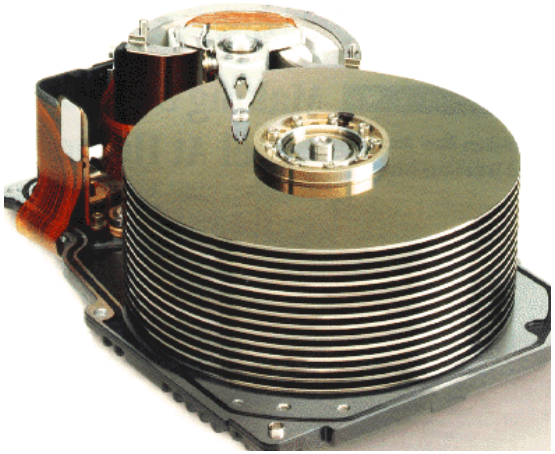


Figure 12-5 Magnetic disk.

From Computer Desktop Encyclopedia
Reproduced with permission.
© 1997 Singapore Technologies



Continued…

- **Cache Memory**: Most computers have another level of IC memory called cache memory. It is placed between CPU registers and main memory. A cache memory capacity is less than that of main memory but it is faster than that of main memory because some or all of it can reside on the same IC as the CPU. Cache memories are essential components of high performance computers.

Continued…

- **Location**: The memory which is inside the processor called the internal memory. The memory which is external to the processor is known as external memory.
- **Access Method**: Each memory is a collection of various memory location. Accessing the memory means finding and reaching desired location and than reading information from memory location. The information from locations can be accessed as follows:

Continued…

1. Random access
2. Sequential access
3. Direct access.

- **Random Access**: It is the access mode where each memory location has a unique address. Using these unique addresses each memory location can be addressed independently in any order in equal amount of time. Generally, main memories are random access memories.

Continued…

- **Sequential Access**: If storage locations can be accessed only in a certain predetermined sequence, the access method is known as serial or sequential access.
- **Direct Access**: In this access information is stored on tracks and each track has a separate read/write head. This features makes it a semi random mode which is generally used in magnetic disks.

Continued…

- **Volatile Memories**: The memories that looses their contents when the power is turned off called volatile memories.
- **Non-volatile Memories**: The memories that do not loose their contents when the power is removed called Non-volatile memories.

MEMORY CLASSIFICATION

In general the memory is classified in two types based on their mode of access of a memory system.

1. Random access memory
2. Sequential access memory

- **Random Access Memory**: The world of data reading or writing from or to the memory requires same time. We can access the data randomly.

  Example: hard disk.

Continued…

- **Sequential Access Memory**: The information stored in some medium is not immediately accessible but is available as certain intervals of time. The access time is variable.

  Example: magnetic tape.

TYPES OF RAM

- **Static RAM**: It consist of internal latches that store the binary information. The stored information remains valid as long as power is applied to the unit.
- **Dynamic RAM**: It stores the binary information in the form of electric charges on capacitors. The capacitors are provided inside the chip by MOS transistors. The stored charge on the capacitor tends to discharge with time and the capacitors must be periodically recharged by refreshing the dynamic memory.

ROM & THEIR TYPES

- **Read only memory**: It is non-volatile memory, which retains the data even when power is removed from this memory. Programs and data that can not be altered are stored in ROM. The required paths in a ROM may be programmed in four different ways:

1. **Mask Programming**: It is done by the company during the fabrication process of the unit. The procedure for fabricating a ROM requires that the customer fills out the truth table he wishes the ROM to satisfy.

Continued…

2. Programmable Read only memory(PROM):

PROM contain all the fuses intact giving all 1's in the bits of the stored words. A blown fuse defines binary 0 state and an intact fuse give a binary 1 state. This allows the user to program the PROM by using a special instruments called PROM programmer.

Continued…

3. **Erasable PROM(EPROM):** In a PROM once fixed pattern is permanent and can not be altered. The EPROM can be restructured to the initial state even through it has been programmed previously. When EPROM is placed under a special ultra-violet light for a given period of time all the data are erased. After erase, the EPROM returns to it initial state and can be programmed to a new set of values.

Continued…

4. **Electrically Erasable PROM(EEPROM):**

It is similar to EPROM except that the previously programmed connections can be erased with an electrical signal instead of ultra violet light. The advantage is that device can be erased without removing it from it's socket.

Main Memory

- It is the memory used to store programs and data during the computer operation.
- The principal technology is based on semiconductor integrated circuits.
- It consists of RAM and ROM chips.
- RAM chips are available in two form static and dynamic.

SRAM	DRAM
Uses capacitor for storing information	Uses Flip flop
More cells per unit area due to smaller cell size.	Needs more space for same capacity
Cheap and smaller in size	Expensive and bigger in size
Slower and analog device	Faster and digital device
Requires refresh circuit	No need
Used in main memory	Used in cache

- ROM is uses random access method.
- It is used for storing programs that are permanent and the tables of constants that do not change.
- ROM store program called bootstrap loader whose function is to start the computer software when the power is turned on.
- When the power is turned on, the hardware of the computer sets the program counter to the first address of the bootstrap loader.

Assignment

- Explain Memory Hierarchy in detail with diagram.
- Define cache memory
- Explain main memory with examples.
- Explain auxiliary memory with examples.
- Differentiate between RAM and ROM.
- Differentiate between SRAM and DRAM.

RAM and ROM chips

- RAM and ROM chips are available in a variety of sizes. If we need larger memory for the system, it is necessary to combine a number of chips to form the required memory size.

RAM Chips

- A RAM chip is better suited to communicate with CPU if it has one or more control inputs that select the chip only when needed. The block diagram of a RAM chip is shown below:

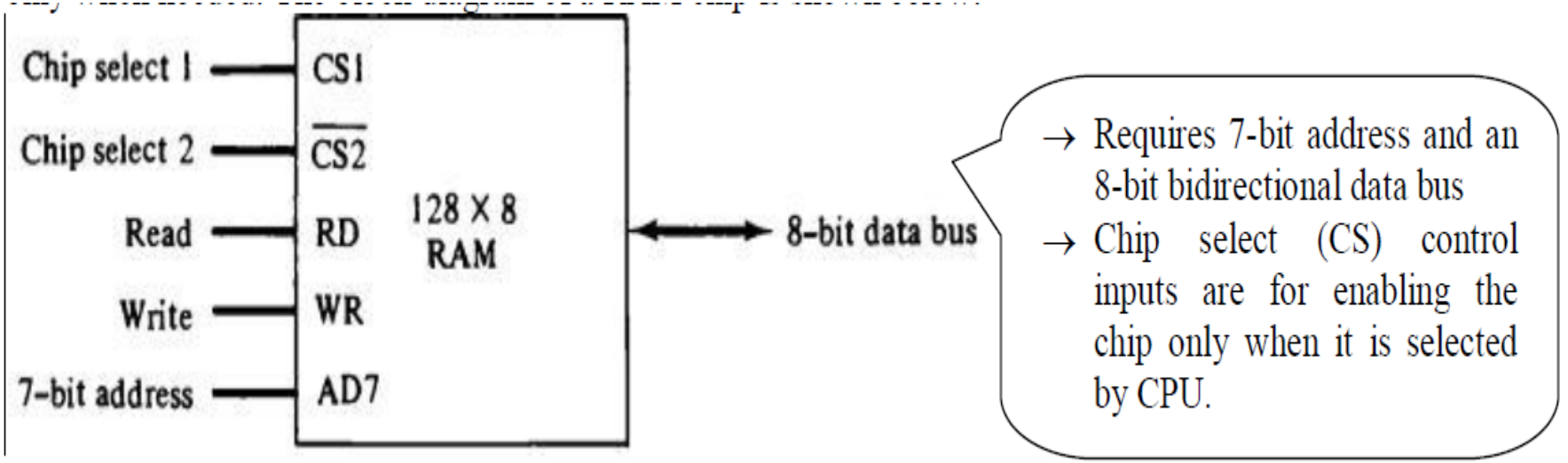


Fig: Typical RAM chip (128 words of eight bits each)

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 from RAM
1	1	x	x	Inhibit	High-impedance

Fig: Function table for RAM chip

→ The unit is in operation only when CS1=1 and (CS2)'=0.

→ High impedance state indicates open circuit i.e. output does not carry a signal and has no logic significance.

ROM Chips

- Since a ROM chip can only read, data bus is unidirectional (output mode only).

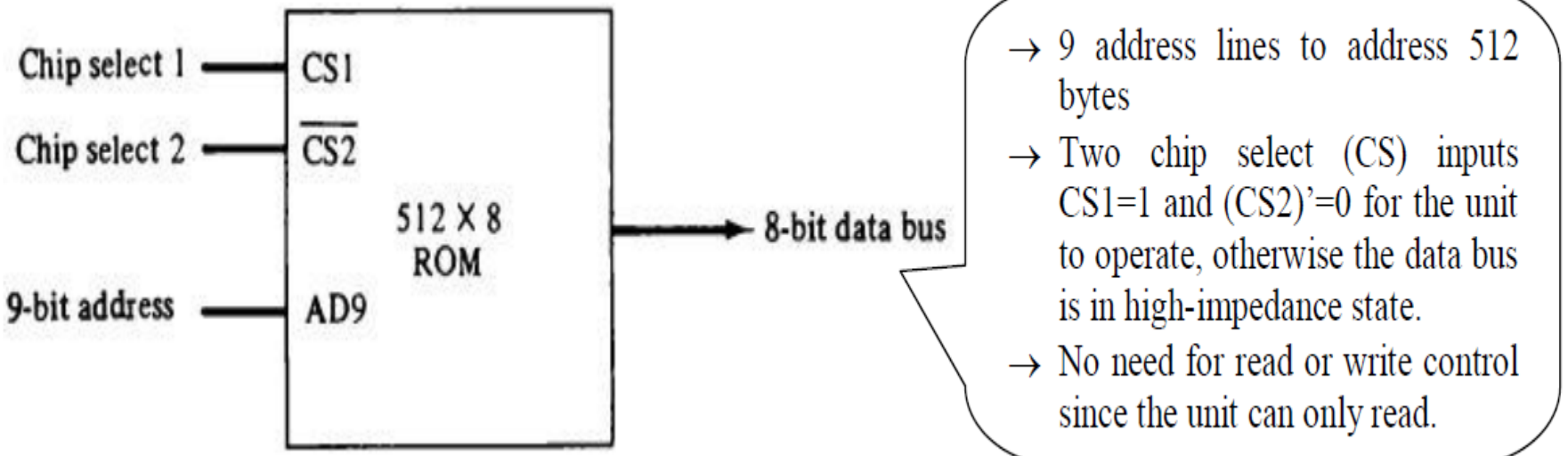


Fig: Typical ROM chip (512 byte ROM)

Memory Address Map

- The addressing of memory can be established by means of a table that specifies the memory address assigned to each RAM or ROM chip. This table is called memory address map and is a pictorial representation of assigned address space for particular chip.

Example: Suppose computer system needs 512 bytes of RAM and 512 bytes of ROM.

Solution,

We have a chip size of RAM =128=2^7

We have a chip size of ROM = 512=2^9

So, total RAM required = 512/128 = 4

So, total ROM required = 512/512 =1

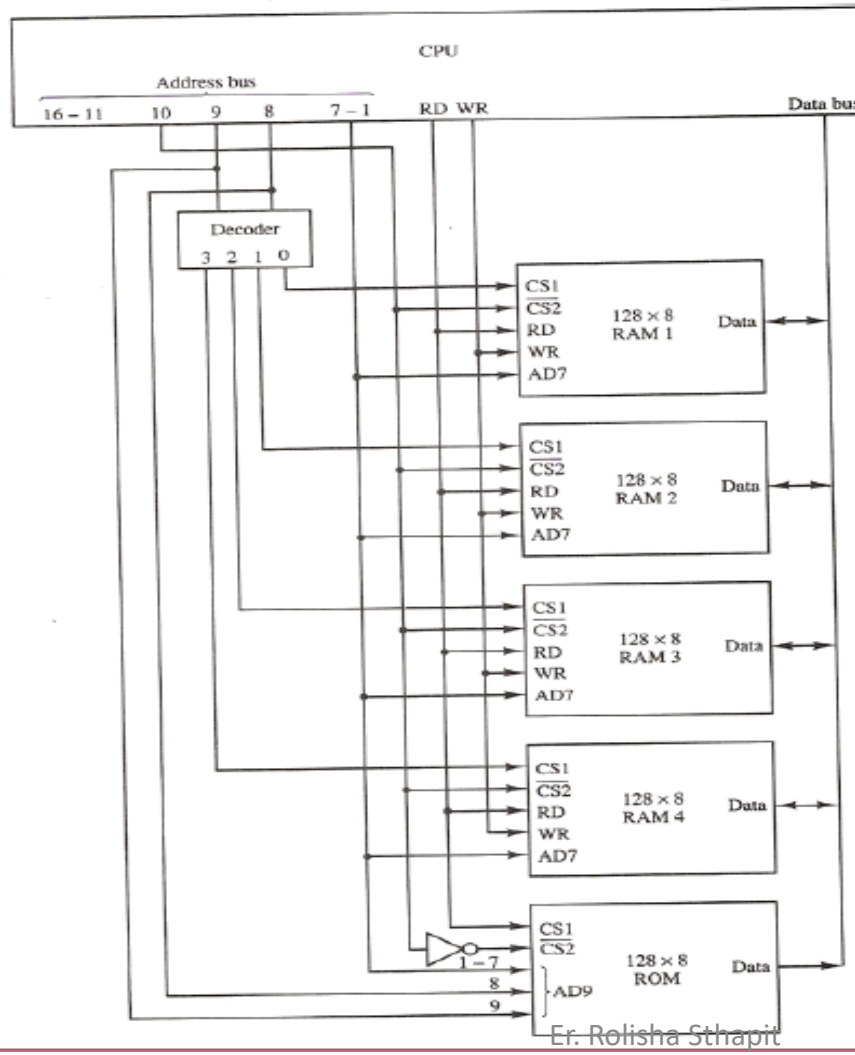
Therefore, 4 RAM and 1 ROM is required.

	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
RAM 1	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
RAM 2	0	0	0	0	0	0	0	0	1	X	X	X	X	X	X	X
	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
RAM 3	0	0	0	0	0	0	0	1	0	X	X	X	X	X	X	X
	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1
RAM 4	0	0	0	0	0	0	0	1	1	X	X	X	X	X	X	X
	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
ROM	0	0	0	0	0	0	1	X	X	X	X	X	X	X	X	X
	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

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

Memory-CPU Connection

RAM and ROM chips are connected to CPU through data and address buses.



Example gives an indication of the interconnection complexity that can exist between memory chips and CPU. More the chips, more external decoders are required for selection among the chips.

- This configuration gives 512 bytes of RAM and 5112 bytes of ROM
- Each RAM receives 7 low-order bits of the address bus to select a byte.
- RAM chips are selected with decoder with selection input of line 8 and 9.
- The selection between RAM and ROM is done by line 10. When 0, RAMs are selected and when 1 ROM get selected.

Associative Memory

- The time require to find an item store in the memory can be reduced considerably if the stored data can be identified for excess by the content of data itself rather than by address.
- A memory unit accessed by the content is called an associative memory or content addressable memory (CAM).
- This type of memory is accessed simultaneously and in parallel on the basis of data content rather then by specific address or location.
- When a word is written in a associative memory, no address is given.
- Memory is capable of finding empty unused location to store the word.

- When a word is to be read from an associative memory, the content of the word or the part of the word is specified.
- The memory locates all words which match the specified content and marks them for reading.
- An associative memory is more expensive than RAM, as each cell must have storage capability as well as logic circuits for matching its content with an external argument.
- Associative memories are used in applications where the search time is very critical and short.

Hardware Organization

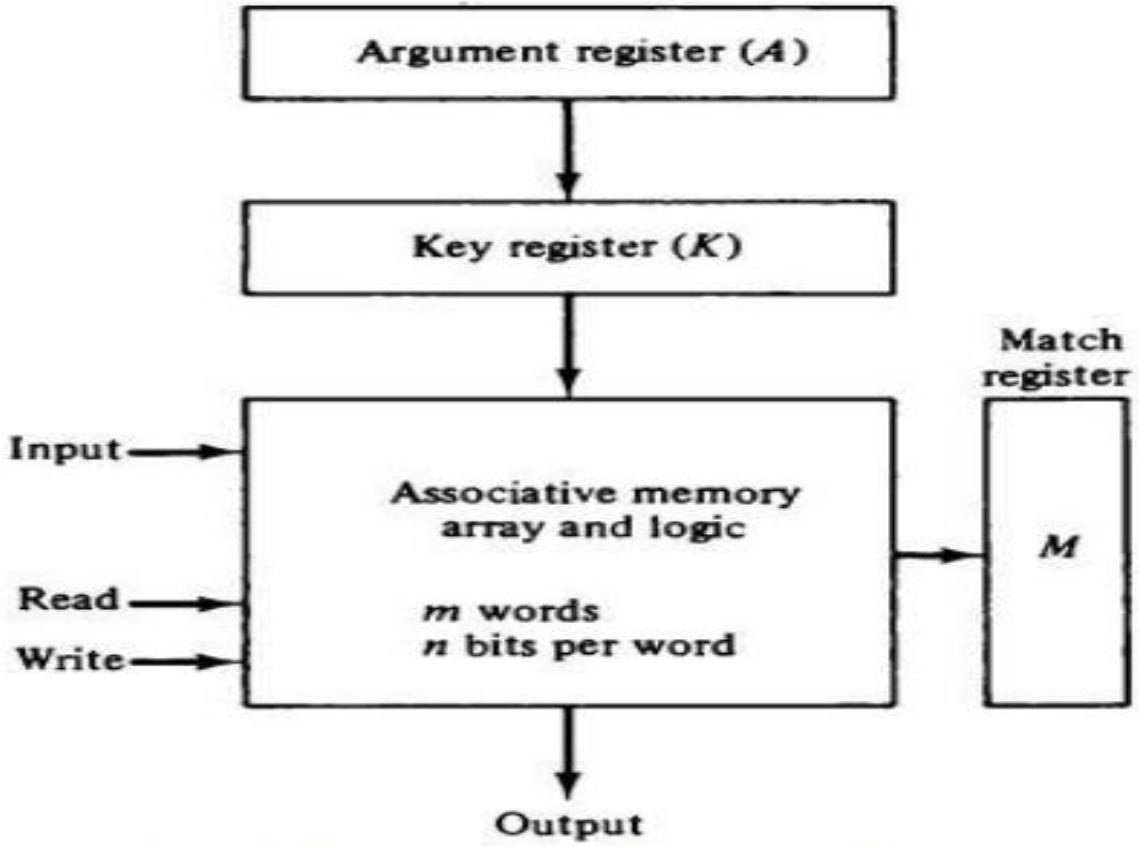


Fig: Block Diagram of Associative Memory

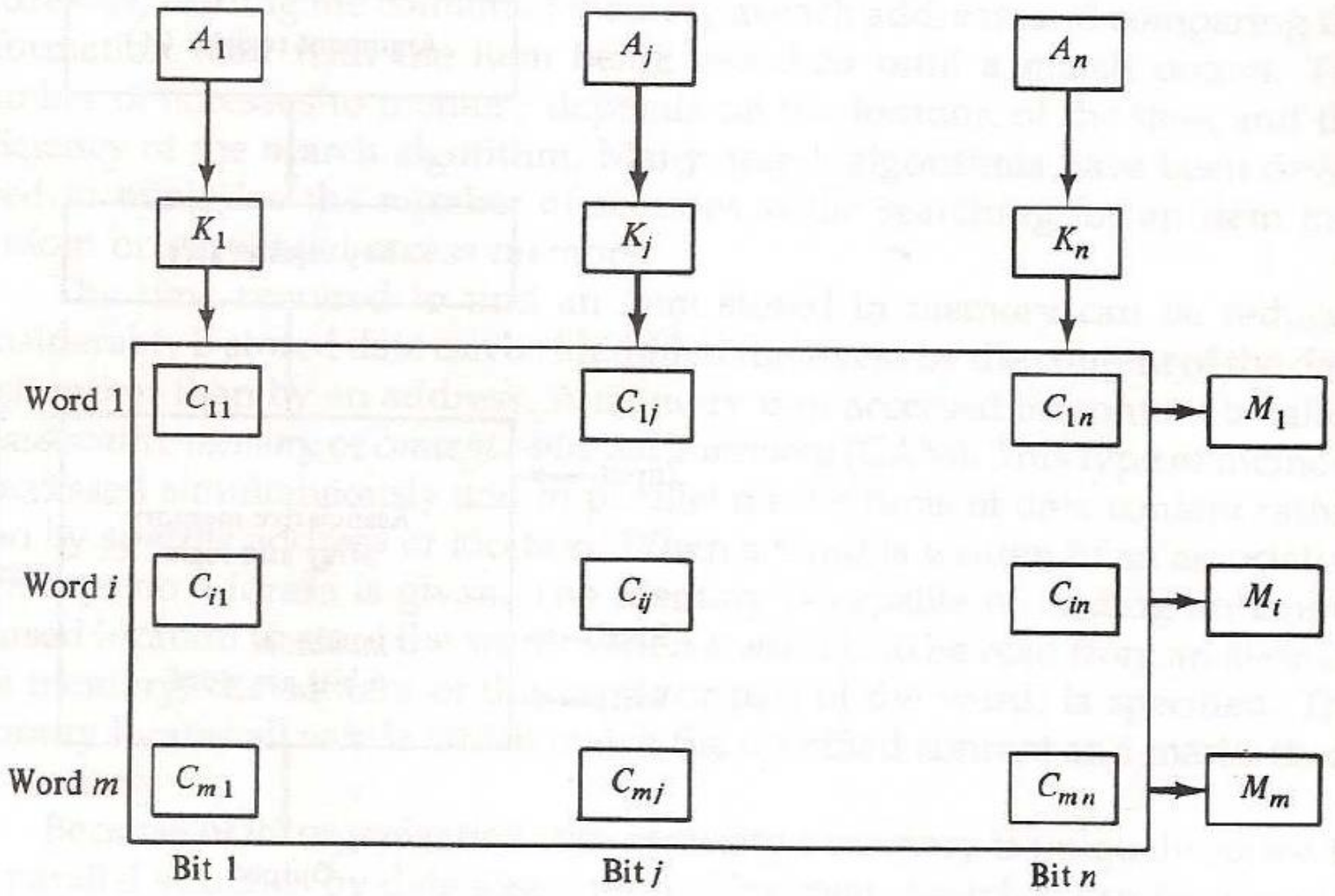
- Associative memory consists of a memory array and logic for m words and n bits per word.
- The argument register A and key register K each have n bits, one for each bit of a word. The match register M has m bits, one for each memory words. Each word in memory is compared in parallel with the content of the argument register. The words that match the bits of the argument register set a corresponding bit in the match register. After matching process, those bits in the match register that have been set indicate the fact that their corresponding words have been matched.
- Reading is accomplished by a sequential access to memory for those words whose corresponding bits in the match register have been set.

- The key register provides a mask for choosing a particular field or key in the argument word. The entire argument is compared with each memory word if the key register contains all 1's. 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 or identifying piece of information which specifies how the reference to memory is made.

E.g. A=10111100
K=11100000

Word1	100111100	no match
Word2	101000001	match

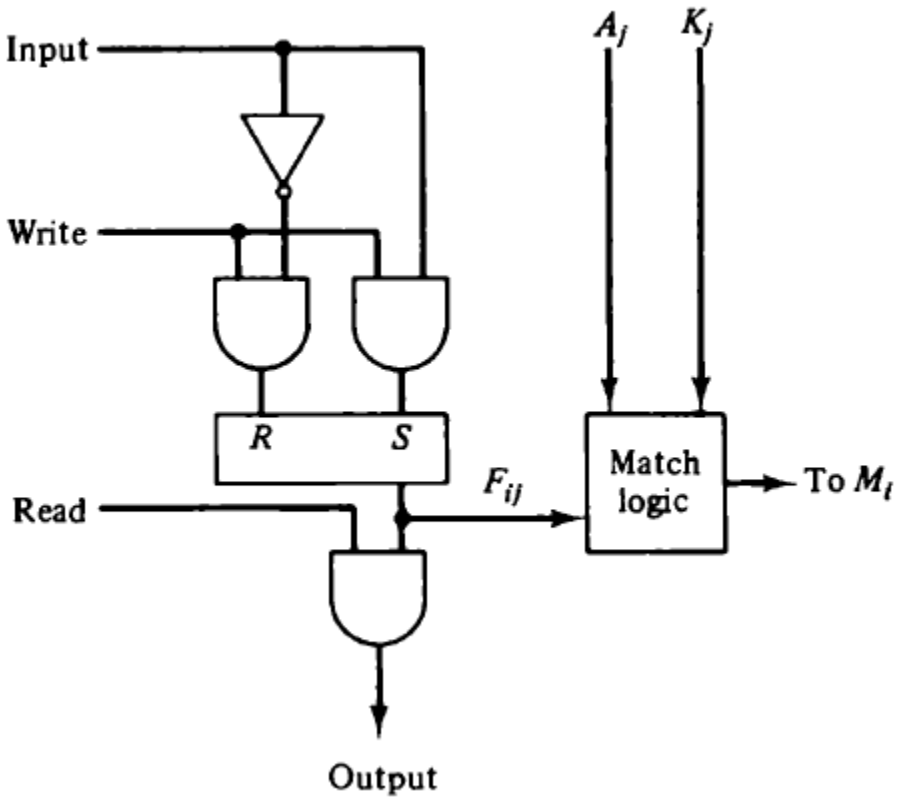
Figure Associative memory of m word, n cells per word.



The relation between the memory array and external registers in an associative memory is shown in Fig. 12-7. The cells in the array are marked by the letter C with two subcripts. The first subscript gives the word number and the second specifies the bit position in the word. Thus 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 is done for all columns $j = 1, 2, \ldots, n$. 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.

The internal organization of a typical cell C_{ij} is shown in Fig. 12-8. It consists of a flip-flop storage element F_{ij} and the circuits for reading, writing, and matching the cell. The input bit is transferred into the storage cell during a write operation. The bit stored is read out during a read operation. The match logic compares the content of the storage cell with the corresponding unmasked bit of the argument and provides an output for the decision logic that sets the bit in M_i.

Figure 12-8 One cell of associative memory.



Match Logic

The match logic for each word can be derived from the comparison algorithm for two binary numbers. First, we *neglect* the key bits and compare the argument in A with the bits stored in the cells of the words. Word i is equal to the argument in A if $A_j = F_{ij}$ for $j = 1, 2, \ldots, n$. Two bits are equal if they are both 1 or both 0. The equality of two bits can be expressed logically by the Boolean function

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

where $x_j = 1$ if the pair of bits in position j are equal; otherwise, $x_j = 0$.

For a word i to be equal to the argument in A we must have all x_j variables equal to 1. This is the condition for setting the corresponding match bit M_i to 1. The Boolean function for this condition is

$$M_i = x_1 x_2 x_3 \cdots x_n$$

and constitutes the AND operation of all pairs of matched bits in a word.

We now include the key bit K_j in the comparison logic. The requirement is that if $K_j = 0$, the corresponding bits of A_j and F_{ij} need no comparison. Only when $K_j = 1$ must they be compared. This requirement is achieved by ORing each term with K_j', thus:

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

When $K_j = 1$, we have $K_j' = 0$ and $x_j + 0 = x_j$. When $K_j = 0$, then $K_j' = 1$ and $x_j + 1 = 1$. A term $(x_j + K_j')$ will be in the 1 state if its pair of bits is not compared. This is necessary because each term is ANDed with all other terms so that an output of 1 will have no effect. The comparison of the bits has an effect only when $K_j = 1$.

The match logic for word i in an associative memory can now be expressed by the following Boolean function:

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

Each term in the expression will be equal to 1 if its corresponding $K_j = 0$. If $K_j = 1$, the term will be either 0 or 1 depending on the value of x_j. A match will occur and M_i will be equal to 1 if all terms are equal to 1.

If we substitute the original definition of x_j, the Boolean function above can be expressed as follows:

$$M_i = \prod_{j=1}^{n} (A_j F_{ij} + A_j' F_{ij}' + K_j')$$

where Π is a product symbol designating the AND operation of all n terms. We need m such functions, one for each word $i = 1, 2, 3, \ldots, m$.

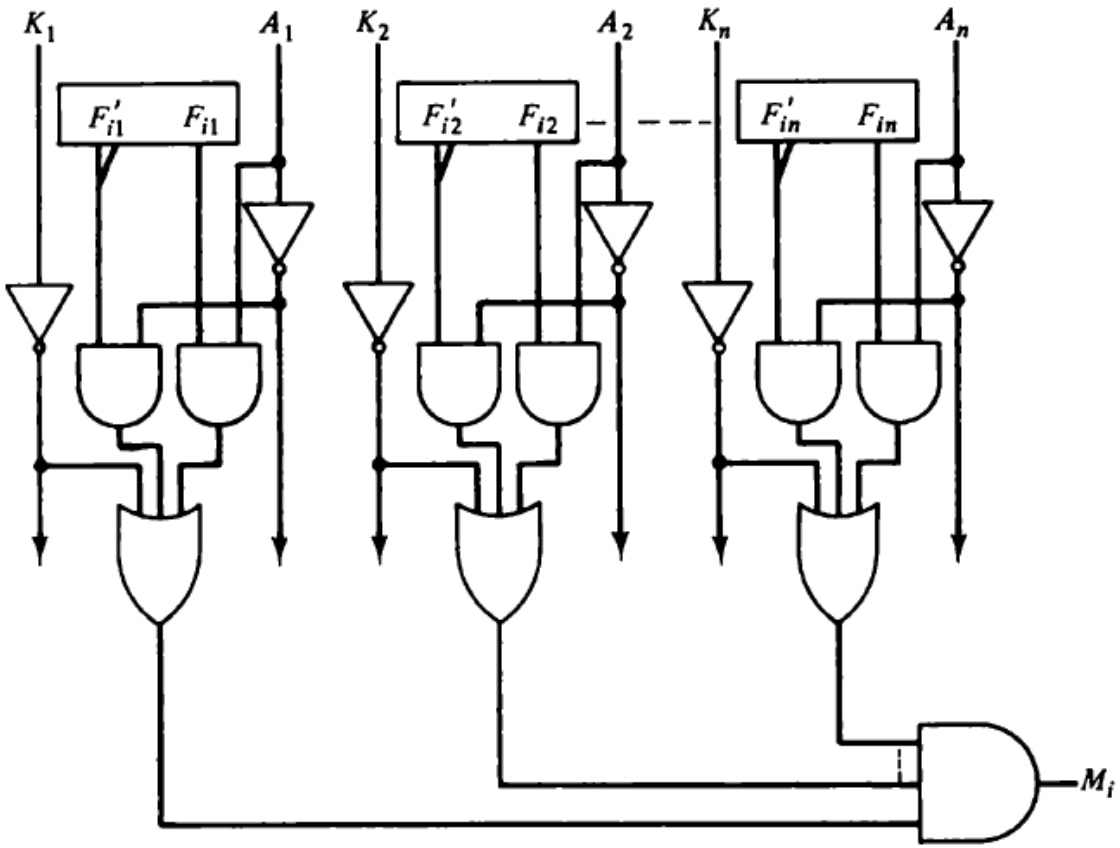


Figure 12-9 Match logic for one word of associative memory.

Read Operation

- If more than 1 word in memory matches the unmasked argument field, all matched words will have 1 in match register.
- Scan the bits of match register one at a time.
- Matched words are read by applying read signal, whose $M_i = 1$.
- When output M_i is connected directly to read line in same word, content of matched word will be presented automatically in output line.
- No special read command is required.
- If words containing zero are excluded, all zero output will indicate no match.

Write Operation

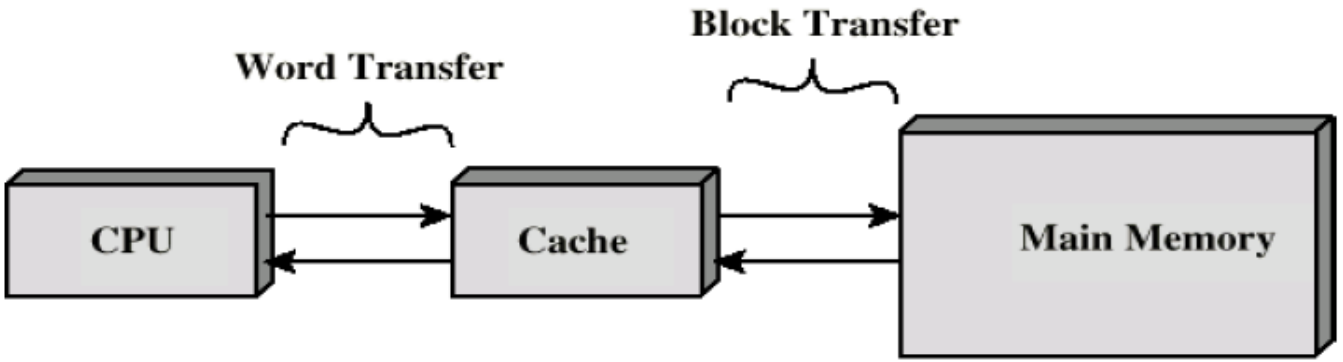
- It must have write capability to store the information to be searched.
- If entire memory is loaded with new information at once prior to search, then writing can be done by addressing each location in sequence.
- Device works as RAM for writing and Content addressable memory (CAM) for reading.
- Address for input can be decoded as in RAM.
- Instead of ‘m’ address lines, one for each word, Number of address lines are reduced to ‘d’ lines. $m = 2^d$
- If unwanted words are deleted and new words are inserted, we need special register (tag register) to distinguish between active and inactive words.

- It contain 1 for active word and 0 for inactive word.
- If the word is to be inserted, tag register is scanned until 0 is found and word is written at that position and bit is change to 1.

Cache Memory

- Cache is extremely fast memory that is built into a computer's central processing unit (CPU), or located next to it on a separate chip.
- The CPU uses cache memory to store instructions that are repeatedly required to run programs, improving overall system speed.
- The advantage of cache memory is that the CPU does not have to use the motherboard's system bus for data transfer.
- Whenever data must be passed through the system bus, the data transfer speed slows to the motherboard's capability. The CPU can process data much faster by avoiding the bottleneck created by the system bus.

- A cache is a small amount of very fast associative memory.
- It sits between normal main memory and CPU.
- When CPU needs to access contents of memory location, then cache is examined for this data.
- If present, get from cache (fast).
- If not present, read required block from main memory to cache, then deliver from cache to CPU.
- Cache includes tags to identify which block of main memory is in each cache slot.



- The performance of cache memory is frequently measured in terms of a quantity called hit ratio. When CPU refers to memory and finds the word in cache, then it is said to produce hit.
- If word is not found in cache, it is in main memory and it counts as a miss.
- The ratio of the number of hits (success in finding the words in cache) to the total CPU references to memory (hits + misses) is known as hit ratio.
- The basic characteristic of cache memory is its fast access time.
- Hit Ratio = $\frac{Hit}{Hit+Miss}$
- Average memory access time

  = hit time + miss rate * miss time

- Example:

A computer with cache access time of 100 ns, a main memory access time of 1000 ns at the hit ratio of 0.9, what is the average access time?

Solution,

Hit time = 100 ns

Memory access time (miss time) = 1000 ns

Hit ratio = 0.9

Miss ratio = 1 – Hit ratio = 1-0.9 = 0.1

- Average memory access time

= hit time + miss rate * miss time

= 100 + 0.1*1000

= 200 ns

- Given a cache memory with access time of 2ns and RAM with excess time 10 ns. If the hit ratio is 80%. Calculate the memory access time.

Hit time = 2 ns

Memory access time (miss time) = 10 ns

Hit ratio = 80% = 0.8

Miss ratio = 1 – Hit ratio = 1-0.8 = 0.2

- Average memory access time

= hit time + miss rate * miss time

= 2 + 0.2*10

= 4 ns

Locality of Reference

- Analysis of large number of program shows that reference to memory at any given interval of time tend to be confined to few localized area in memory. This is known as locality of reference.
- Since size of cache memory is less as compared to main memory. So to check which part of main memory should be given priority and loaded in cache is decided based on locality of reference.
- Phenomenon in which the same values, or related storage locations, are frequently accessed, depending on the memory access pattern.

- There are two basic types of reference locality
  - Temporal locality
  - Spatial locality

Temporal Locality –

Temporal locality means current data or instruction that is being fetched may be needed soon. So we should store that data or instruction in the cache memory so that we can avoid again searching in main memory for the same data. When CPU accesses the current main memory location for reading required data or instruction, it also get stored in cache memory which is based on the fact that same data or instruction may be needed in near future. This is known as temporal locality.

Spatial Locality –

Spatial locality means instruction or data near to the current memory location that is being fetched, may be needed soon in near future. This is slightly different from temporal locality. Here we are taking about nearly located memory locations while in temporal locality we were taking about the actual memory location that were being fetched.

Cache Mapping Techniques

The transformation of data from main memory to cache is known as mapping process. Three types of mapping procedures are:

- Associative Mapping
- Direct Mapping
- Set-Associative Mapping

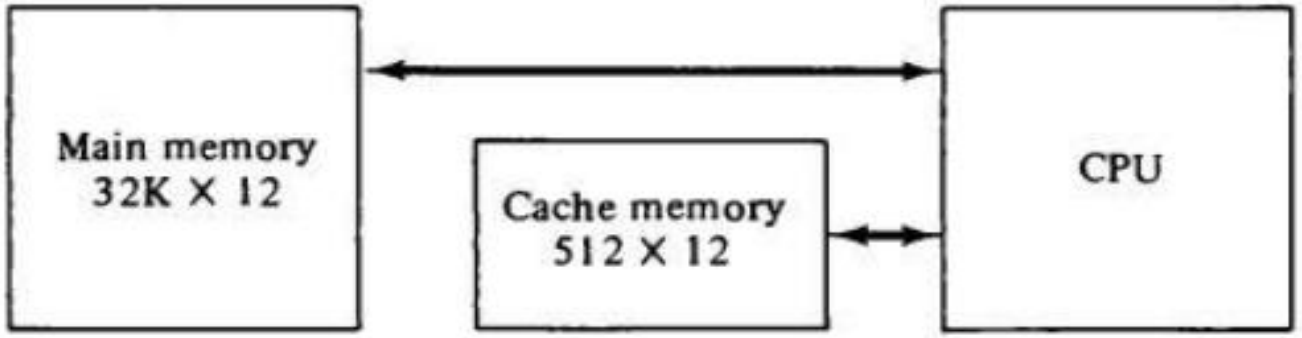


Fig: Example of Cache memory

- The main memory can store 32K words of 12 bits each. The cache is capable of storing 512 of these words at any given time.
- For every word stored in cache, there is a duplicate copy in main memory. The CPU communicates with both memories. It first sends a 15-bit address to cache.
- If there is a hit, the CPU accepts the 12-bit data from cache. If there is a miss, the CPU reads the word from main memory and the word is then transferred to cache.

a. Associative Mapping

- The fastest and most flexible cache organization uses an associative memory. The associative memory stores both the address and content (data) of the memory word.
- This permits any location in cache to store any word from main memory.
- The address value of 15 bits is shown as a five digit octal number and its corresponding 12 bit word is shown as a four digit octal number.
- A CPU address of 15 bits is placed in the argument register and the associative memory is searched for a matching address. If the address is found, the corresponding 12 bit data is read and sent to the CPU. If no match occurs, the main memory is accessed for the word.

- The address-data pair is then transferred to the associative cache memory.
- If the cache is full, an address-data pair must be displaced to make room for a pair that is needed and not presently in the cache.
- The decision as to what pair is replaced is determined from the replacement algorithm that the designer chooses for the cache.
- A simple procedure is to replace cells of the cache in round robin order whenever a new word is requested from main memory. This constitutes a first in first out (FIFO) replacement policy.

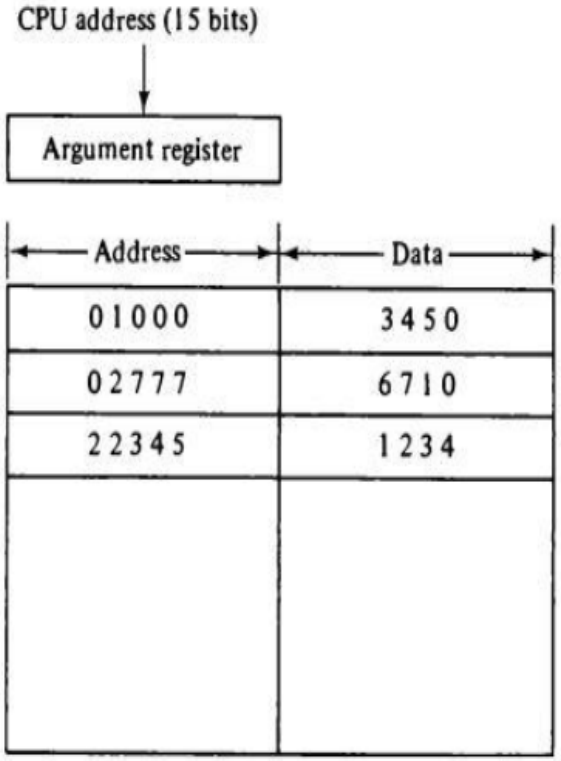


Fig: Associative mapping cache (all numbers in octal)

b. Direct Mapping

- The CPU address of 15 bits is divided into two fields. The nine least significant bits constitute the index field and the remaining six bits form the tag field.

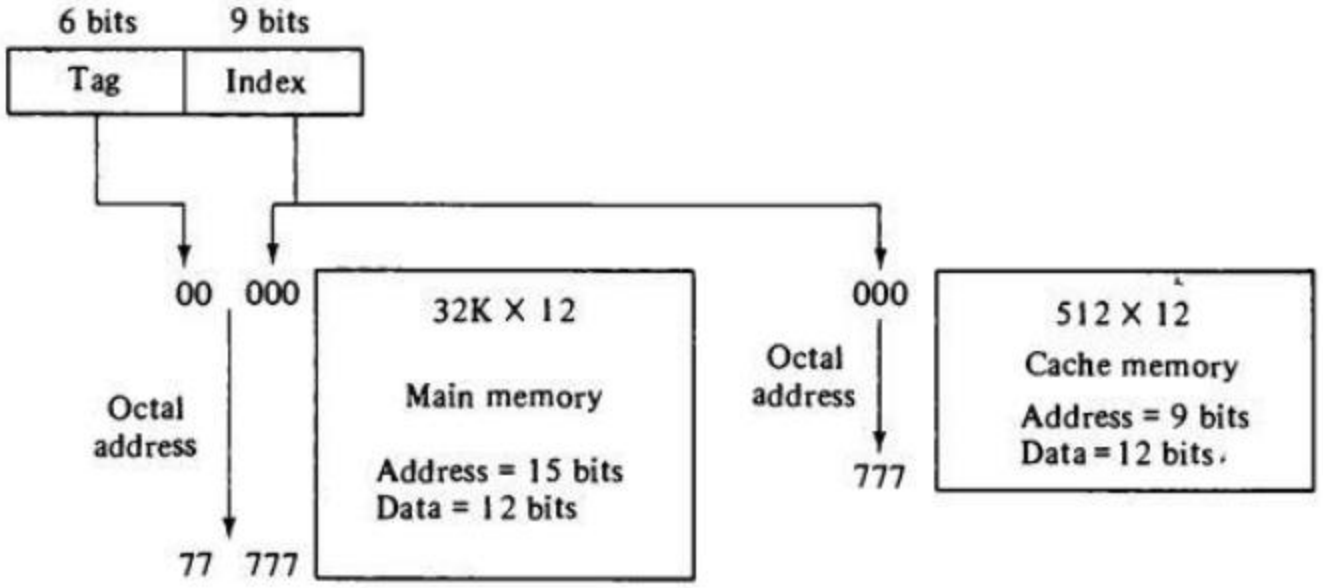


Fig: Addressing relationships between main and cache memories.

- The figure shows that main memory needs an address that includes both the tag and the index bits. The number of bits in the index field is equal to the number of address bits required to access the cache memory.
- There are 2K words in cache memory and 2^n in main memory.
- The n bit memory address is divided into two fields: k bits for the index field and n-k bits for the tag field. The direct mapping cache organization uses the n bit address to access the main memory and the k bit index to access the cache.
- Each word in cache consists of the data word and its associated tag. When a new word is first brought into the cache, the tag bits are stored alongside the data bits. When the CPU generates a memory request, the index field is used for the address to access the cache.

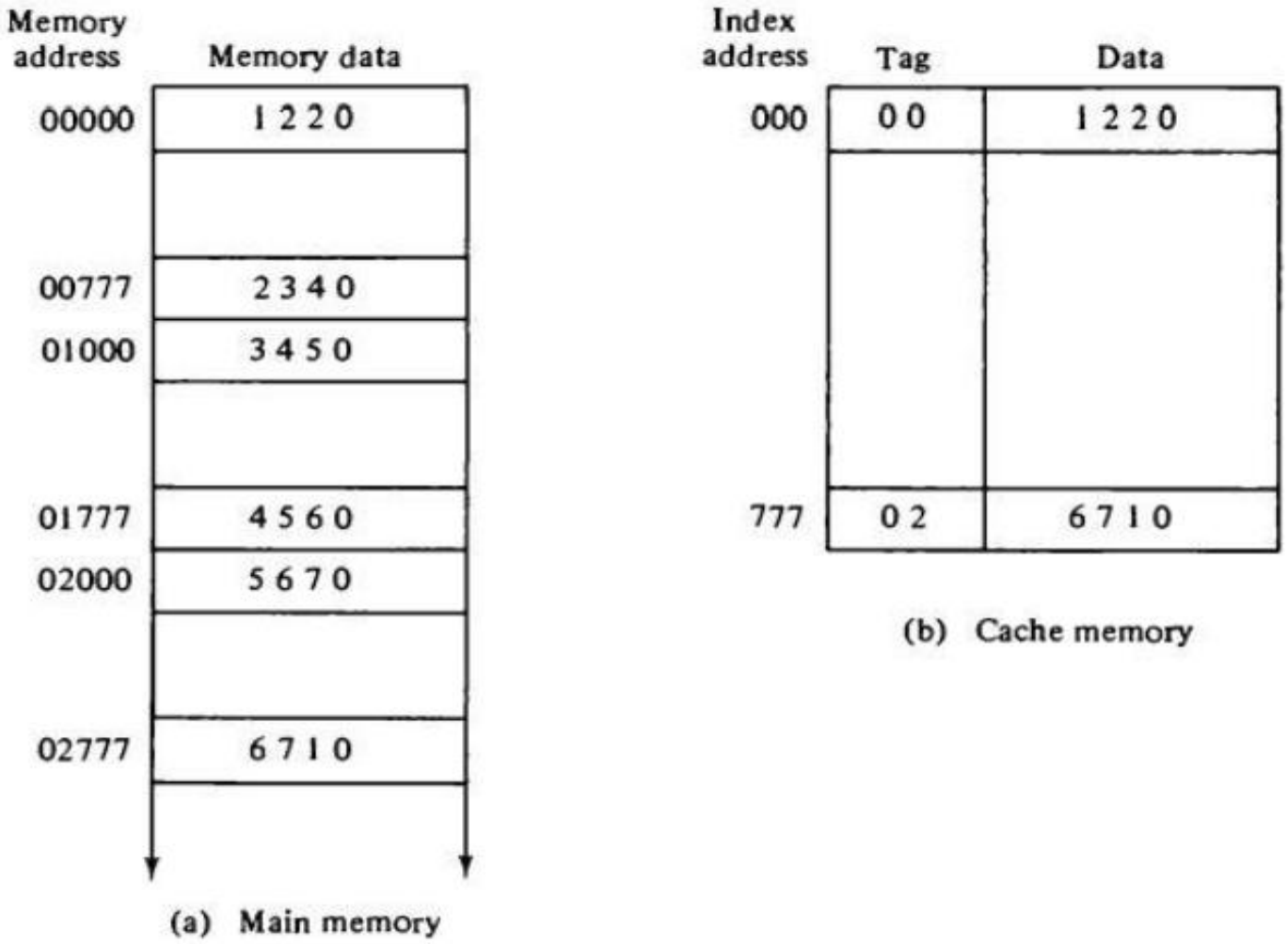
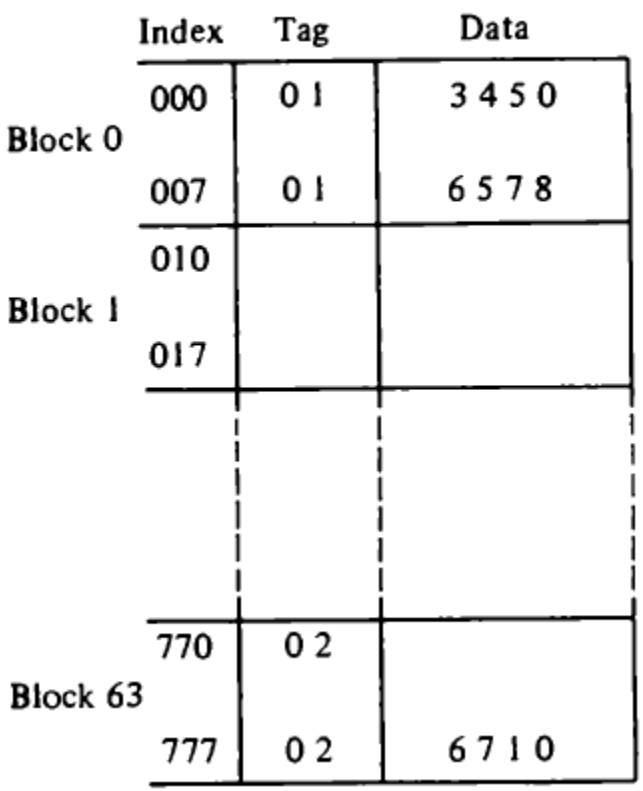
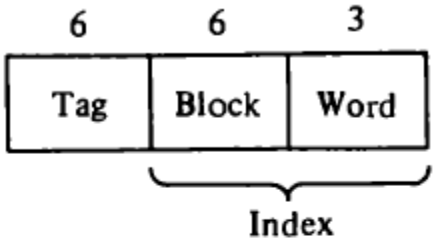


Fig: Direct mapping cache organization.

- The tag field of the CPU address is compared with the tag in the word read from the cache. If the two tags match, there is a hit and the desired data word is in cache. If there is no match, there is a miss and the required word is read from main memory. It is then stored in the cache together with the new tag, replacing the previous value.
- The disadvantage of direct mapping is that two words with the same index in their address but with different tag values cannot reside in cache memory at the same time.

To see how the direct-mapping organization operates, consider the numerical example shown in Fig. . The word at address zero is presently stored in the cache (index = 000, tag = 00, data = 1220). Suppose that the CPU now wants to access the word at address 02000. The index address is 000, so it is used to access the cache. The two tags are then compared. The cache tag is 00 but the address tag is 02, which does not produce a match. Therefore, the main memory is accessed and the data word 5670 is transferred to the CPU. The cache word at index address 000 is then replaced with a tag of 02 and data of 5670.





Direct mapping cache with block size of 8 words.

The direct-mapping example just described uses a block size of one word. The same organization but using a block size of 8 words is shown in Fig.

The index field is now divided into two parts: the block field and the word field. In a 512-word cache there are 64 blocks of 8 words each, since $64 \times 8 = 512$. The block number is specified with a 6-bit field and the word within the block is specified with a 3-bit field. The tag field stored within the cache is common to all eight words of the same block. Every time a miss occurs, an entire block of eight words must be transferred from main memory to cache memory. Although this takes extra time, the hit ratio will most likely improve with a larger block size because of the sequential nature of computer programs.

c. Set Associative Mapping

- It is an improvement over the direct mapping organization in that each word of cache can store two or more words of memory under the same index address.
- Each data word is stored together with its tag and the number of tag-data items in one word of cache is said to form a set.
- Each index address refers to two data words and their associated tags. Each tag requires six bits and each data word has 12 bits, so the word length is 2(6 + 12) = 36 bits.
- An index address of nine bits can accommodate 512 words. Thus, the size of cache memory is 512*36. It can accommodate 1024 words of main memory since each word of cache contains two data words.

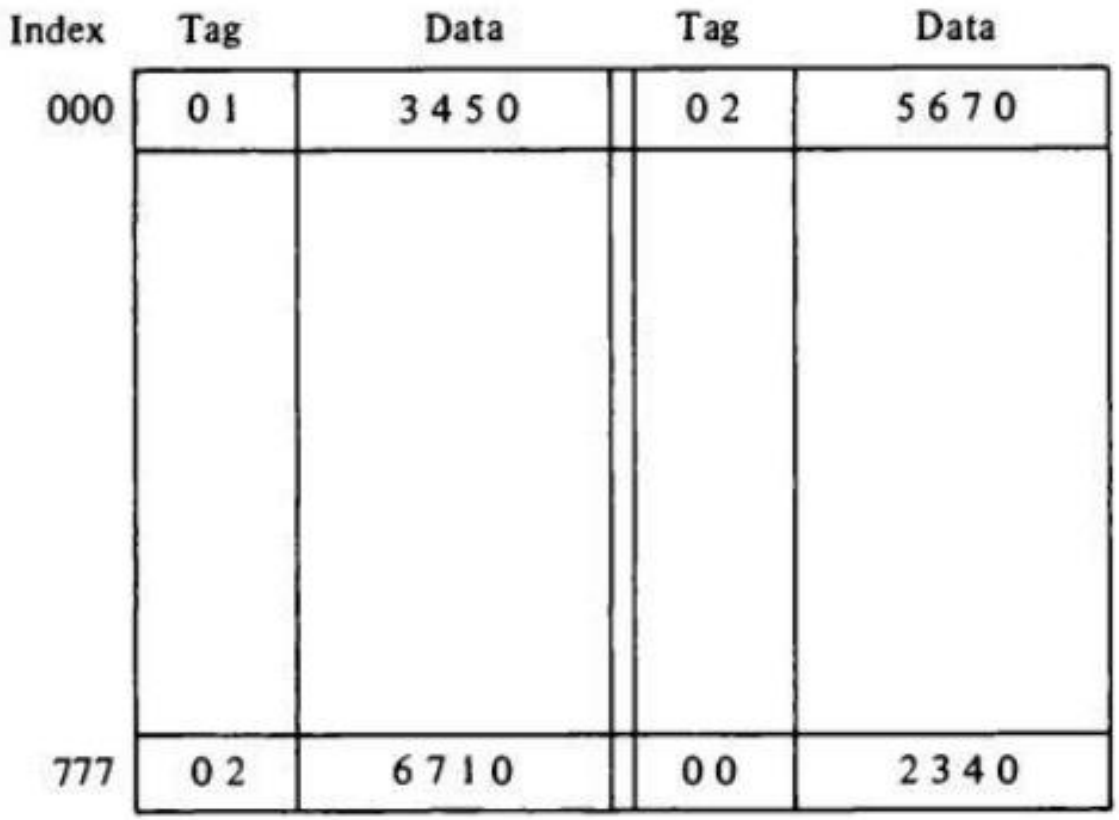


Fig: Two-way set associative mapping cache.

- The words stored at addresses 01000 and 02000 of main memory are stored in cache memory at index address 000. Similarly, the words at addresses 02777 and 00777 are stored in cache at index address 777.
- When CPU generates a memory request, index value of address is used to access the cache. The tag field of CPU address is compared with both tags in cache to determine if a match occurs. The comparison logic is done by an associative search of a tag in the set similar to an associative memory searched.
- When a miss occurs in a set associative cache and the set is full, it is necessary to replace one of the tag data items with a new value.
- The most common replacement algorithms used are: random replacement, first in first out (FIFO), and least recently used (LRU).

Writing into Cache

Writing into cache can be done in two ways:

- Write through
- Write Back

- In **write through**, whenever write operation is performed in cache memory, main memory is also updated in parallel with the cache.
- In **write back**, only cache is updated and marked by the flag. When the word is removed from cache, flag is checked if it is set the corresponding address in main memory is updated.

Bootstrap Loader:

- Bootstrap loader is a program that is stored in ROM and is used to start the loading of OS from hard disk to RAM when power is turned on in a computer. When a computer is turned on hardware of the computer sets PC (program counter) to first instruction of bootstrap loader so that execution of bootstrap loader begins when power is turned on.

Q. Design a system memory of 2048 words that includes 1024 words of RAM and 1024 word of ROM. Use RAM chip of 256 words and ROM chip of 1024 words.

Solution,

We have a chip size of RAM =256=2^8

We have a chip size of ROM = 1024=2^10

So, total RAM required = 1024/256 = 4

So, total ROM required = 1024/1024 =1

Therefore, 4 RAM and 1 ROM is required.

	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
	0	0	0	0	0	0	#	#	x	x	x	x	x	x	x	x
RAM 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
RAM 2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
RAM 3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1
RAM 4	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	1	x	x	x	x	x	x	x	x	x	x
ROM 1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1