# COMPUTER SYSTEM AND ARCHITECTURE

## Course code: ACSD04

## B.Tech III semester

## Regulation: IARE UG -23

## Prepared by:


**Dr.P.L.Srinivasa Murthy**
**Professor**

**COMPUTER SCIENCE AND ENGINEERING**
**INSTITUTE OF AERONAUTICAL ENGINEERING**
**(Autonomous)**
**DUNDIGAL, HYDERABAD - 500 043**

# Course Outcomes

| The course should enable the students to: | |
|---|---|
| CO 1 | Understand the organization and levels of design in computer architecture and To understand the concepts of programming methodologies. |
| CO 2 | Describe Register transfer languages, arithmetic micro operations, logic micro operations, shift micro operations address sequencing, micro program example, and design of control unit. |
| CO 3 | Understand the Instruction cycle, data representation, memory reference instructions, input-output, and interrupt, addressing modes, data transfer and manipulation, program control. Computer arithmetic: Addition and subtraction, floating point arithmetic operations, decimal arithmetic unit. |
| CO 4 | Knowledge about Memory hierarchy, main memory, auxiliary memory, associative memory, cache memory, virtual memory Input or output Interface, asynchronous data transfer, modes of transfer, priority interrupt, direct memory access. |
| CO 5 | Explore the Parallel processing, pipelining-arithmetic pipeline, instruction pipeline Characteristics of multiprocessors, inter connection structures, inter processor arbitration, inter processor Communication and synchronization |

# MODULE –IV

# MEMORY ORGANIZATION

# Contents

**Memory organization:**

- **Memory hierarchy**
- **Main memory**
- **Auxiliary memory**
- **Associative memory**
- **Cache memory**
- **Virtual memory**

- **Semiconductor RAMs**

✓ **Internal Organization**

✓ **Static Memories**

✓ **Dynamic RAMs**

i. **Synchronous and Asynchronous DRAMs**

ii. **Structure of larger memories**

✓ **Read-only Memories**

✓ **Cache Memories – Mapping functions**

- **Nonvolatile Solid-State Memory Technologies**
- **Solid state drives (Volatile memory technologies)**

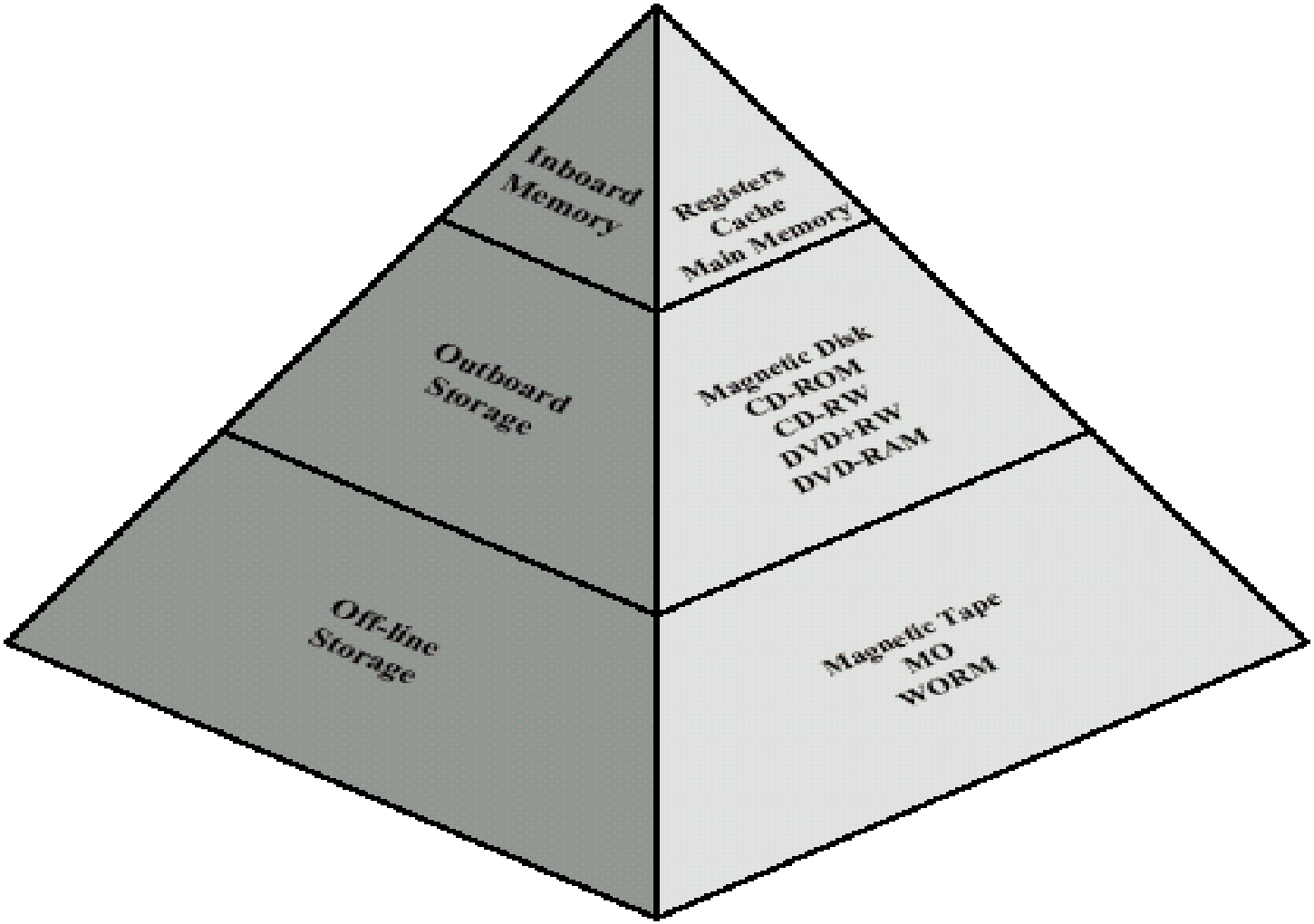
# Course Outcomes

| | |
|---|---|
| CO 1 | Describe Memory hierarchy, main memory and auxiliary memory. |
| CO 2 | Classify the associative memory, cache memory and virtual memory. |
| CO 3 | Describe the Input or output Interface and asynchronous data transfer. |
| CO 4 | Classify the different modes of transfer in Memory organization. |
| CO 5 | Explore the different priority interrupts and direct memory access. |

# Memory Hierarchy

- At the bottom of the hierarchy are the relatively slow magnetic tapes used to store removable files. Next are the magnetic disks used as backup storage. The main memory occupies a central position by being able to communicate directly with the CPU and with auxiliary memory devices through an I/O processor.
- When programs not residing in main memory are needed by the CPU, they are brought in from auxiliary memory. Programs not currently needed in main memory are transferred into auxiliary memory to provide space for currently used programs and data.
- A special very-high speed memory called a cache sometimes used to increase the speed of processing by making current programs and data available to the CPU at a rapid rate.

# Memory Hierarchy



# Memory Hierarchy

- Registers
  - In CPU
- Internal or Main memory
  - May include one or more levels of cache
  - “RAM”
- External memory
  - Backing store

# Memory Hierarchy

- The cache memory is employed in computer systems to compensate for the speed differential between main memory access time and processor logic. CPU logic is usually faster than main memory access time, with the result that processing speed is limited primarily by the speed of main memory.

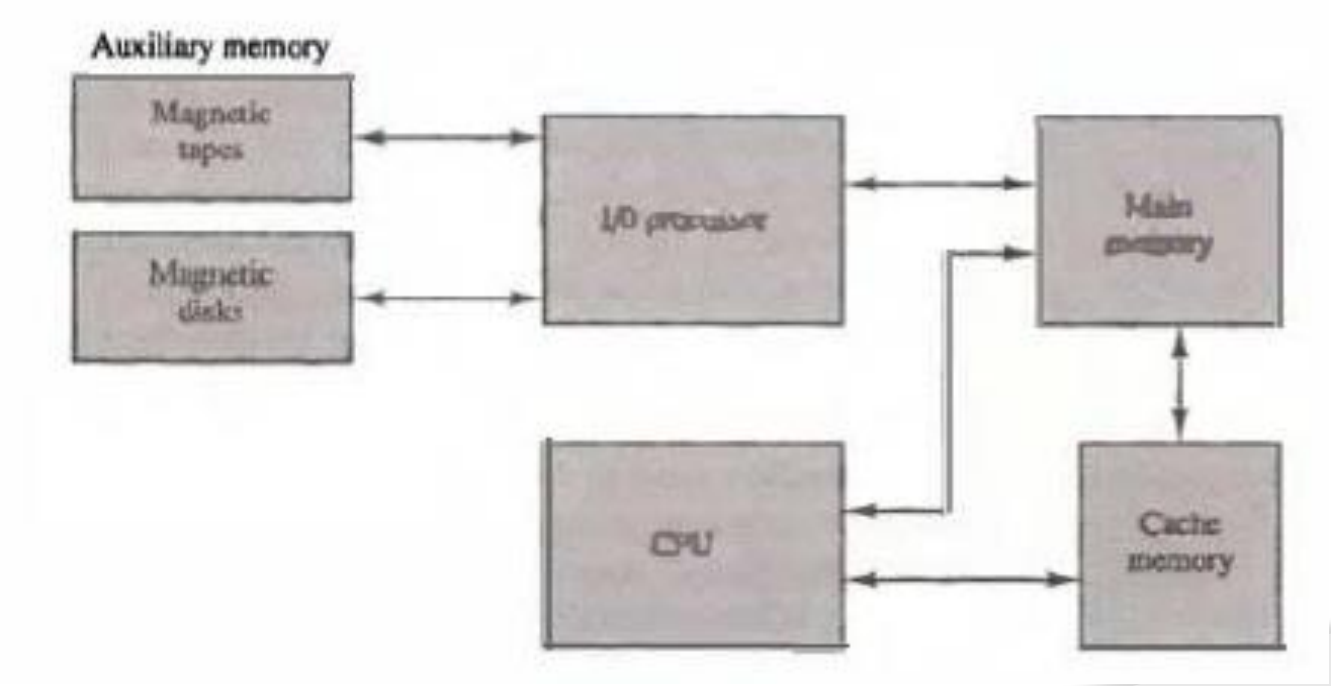



**Figure :**Memory hierarchy in a computer system.

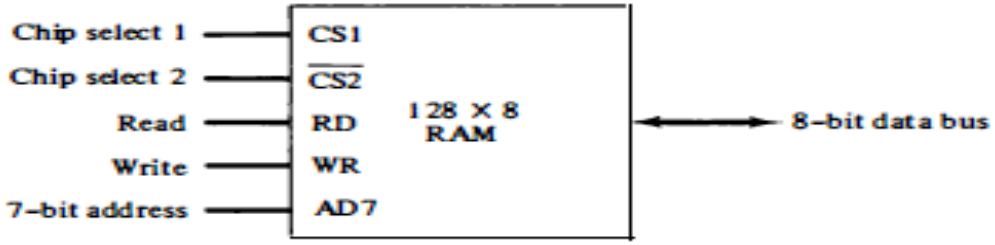
# Main Memory

- The main memory is the central storage unit in a computer system. It is a relatively large and fast memory used to store programs and data during the computer operation. The principal technology used for the main memory is based on semiconductor integrated circuits. Integrated circuit RAM chips are available in two possible operating modes, static and dynamic.
- The static RAM consists essentially of internal flip-flops that store the binary information. The stored information remains valid as long as power is applied to the unit. The dynamic RAM stores the binary information in the form of electric charges that are applied to capacitors.
- The bootstrap loader is a program whose function is to start the computer software operating when power is turned on. Since RAM is volatile, its contents are destroyed when power is turned off. The contents of ROM remain unchanged after power is turned off and on again. The startup of a computer consists of turning the power on and starting the execution of an initial program.

# Main Memory

## RAM and ROM Chips:

- A RAM chip is better suited for communication with the CPU if it has one or more control inputs that select the chip only when needed. Another common feature is a bidirectional data bus that allows the transfer of data either from memory to CPU during a read operation, or from CPU to memory during a write operation. A bidirectional bus can be constructed with three-state buffers.




(a) Block diagram

| CS1 | CS2 | RD | WR | Memory function | State of data bus |
|---|---|---|---|---|---|
| 0 | 0 | × | × | Inhibit | High-impedance |
| 0 | 1 | × | × | Inhibit | High-impedance |
| 1 | 0 | 0 | 0 | Inhibit | High-impedance |
| 1 | 0 | 0 | 1 | Write | Input data to RAM |
| 1 | 0 | 1 | × | Read | Output data from RAM |
| 1 | 1 | × | × | Inhibit | High-impedance |

(b) Function table

Figure: Typical RAM chip.

# Main Memory

- The block diagram of a RAM chip is shown in Fig.The capacity of the memory is 128 words of eight bits (one byte) per word. This requires a 7-bit address and an 8-bit bidirectional data bus. The read and write inputs specify the memory operation and the two chips select (CS) control inputs are for enabling the chip only when it is selected by the microprocessor.
- The read and write inputs are sometimes combined into one line labeled R/W. When the chip is selected, the two binary states in this line specify the two operations or read or write.
- The function table listed in Fig.(b) specifies the operation of the RAM chip. The unit is in operation only when CSI = 1 and CS2 = 0. The bar on top of the second select variable indicates that this input in enabled when it is equal to 0. If the chip select inputs are not enabled, or if they are enabled but the read but the read or write inputs are not enabled, the memory is inhibited and its data bus is in a high-impedance state. When SC1 = 1 and CS2 = 0, the memory can be placed in a write or read mode. When the WR input is enabled, the memory stores a byte from the data bus into a location specified by the address input lines.

# Main Memory

- When the RD input is enabled, the content of the selected byte is placed into the data bus. The RD and WR signals control the memory operation as well as the bus buffers associated with the bidirectional data bus.

**Memory Address Map:**

- a ROM can only read, the data bus can only be in an output mode. The block diagram of a ROM chip is shown in Fig. For the same-size chip, it is possible to have more bits of ROM occupy less space than in RAM. For this reason, the diagram specifies a 512-byte ROM, while the RAM has only 128 bytes.
- The nine address lines in the ROM chip specify any one of the 512 bytes stored in it. The two chip select inputs must be CS1 = 1 and CS2 = 0 for the unit to operate. Otherwise, the data bus is in a high-impedance state. There is no need for a read or write control because the unit can only read.

# Main Memory

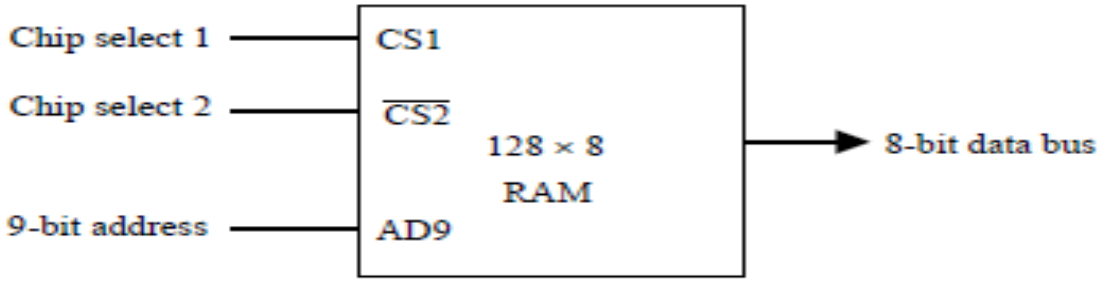



Figure: Typical ROM chip.

- 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

| 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 |

TABLE: Memory Address Map for Microprocomputer

# Main Memory

**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.
- This configuration gives a memory capacity of 512 bytes of RAM and 512 bytes of ROM. It implements the memory map of Table 12-1. 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.

# Main Memory

- 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 .

# Main Memory

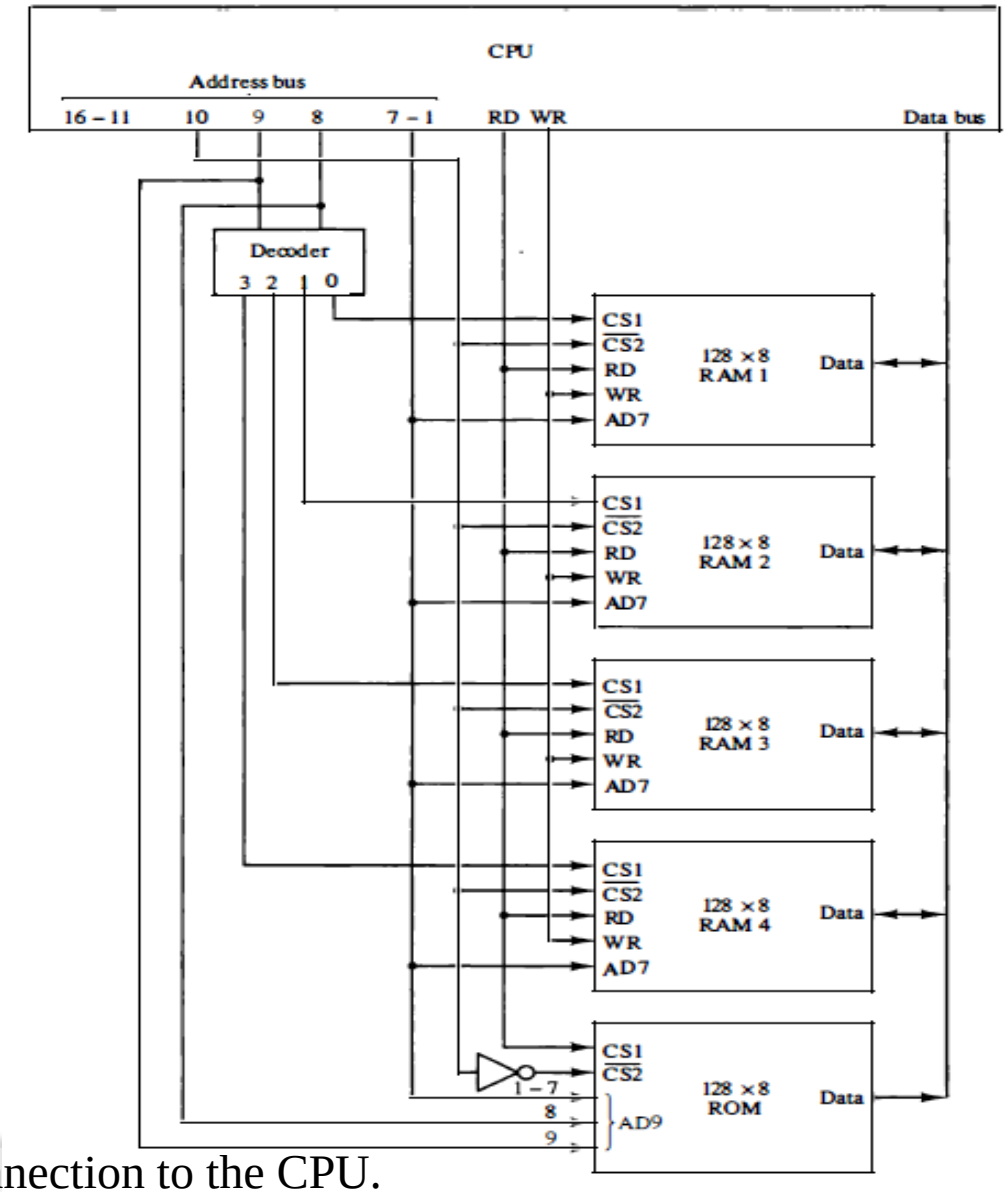



Figure: Memory connection to the CPU.

# Auxiliary Memory

- The most common auxiliary memory devices used in computer systems are magnetic disks and tapes.

**Magnetic Disks:**

- A magnetic disk is a circular plate constructed of metal or plastic coated with magnetized material. Often 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 the magnetized surface in spots along concentric circles called tracks. The tracks are commonly divided into sections called sectors.

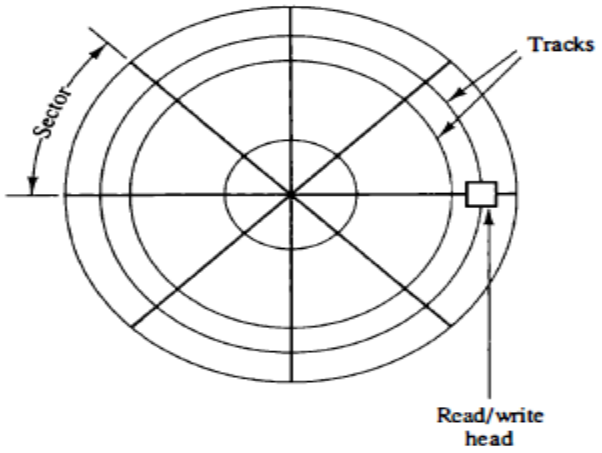



Figure: Magnetic disk.

# Auxiliary Memory

## Magnetic Tape:

- A magnetic tape transport consists of the electrical, 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 the tape along several tracks.
- Usually, seven or nine bits are recorded simultaneously to form a character together with a parity bit. Read/write heads are mounted one in each track so that data can be recorded and read as a sequence of characters.

# Associative Memory

- A memory unit accessed by content is called an associative memory or content addressable memory (CAM). An associative memory is more expensive than a random access memory because each cell must have storage capability as well as logic circuits for matching its content with an external argument. For this reason, associative memories are used in applications where the search time is very critical and must be very short.
- **Hardware Organization:**
- It consists of a memory array and logic for m words with 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 word. 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 the matching process, those bits in the match register that have been set indicate the fact that their corresponding words have been matched.

# Associative Memory

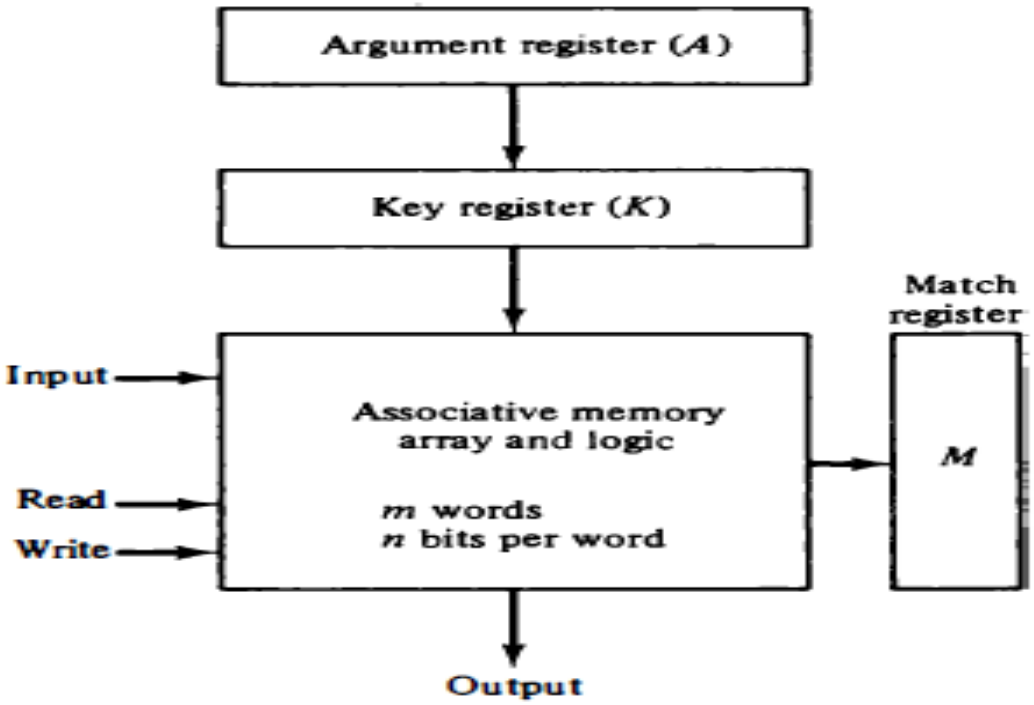



Figure: Block diagram of associative memory.

- 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 l' s. Otherwise, only those bits in the argument that have l's in their corresponding position of the key register are compared.

# Associative Memory

- a numerical example, suppose that the argument register A and the key register K have the bit configuration shown below. Only the three leftmost bits of A are compared with memory words because K has 1's in these positions.

| | | |
|---|---|---|
| *A* | 101 111100 | |
| *K* | 111 000000 | |
| Word 1 | 100 111100 | no match |
| Word 2 | 101 000001 | match |

- Word 2 matches the unmasked argument field because the three leftmost bits of the argument and the word are equal.
- The relation between the memory array and external registers in an associative memory is shown in Fig.
- The cells in the 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.

# Associative Memory

- Thus cell Cij is the cell for bit j in word i. A bit Ai in the argument register is compared with all the bits in column j of the array provided that Ki = 1. This is done for all columns j = 1, 2, . . . , n.
- If a match occurs between all the unmasked bits of the argument and the bits in word i, the corresponding bit M1 in the match register is set to 1. If one or more unmasked bits of the argument and the word do not match, M1 is cleared to 0.

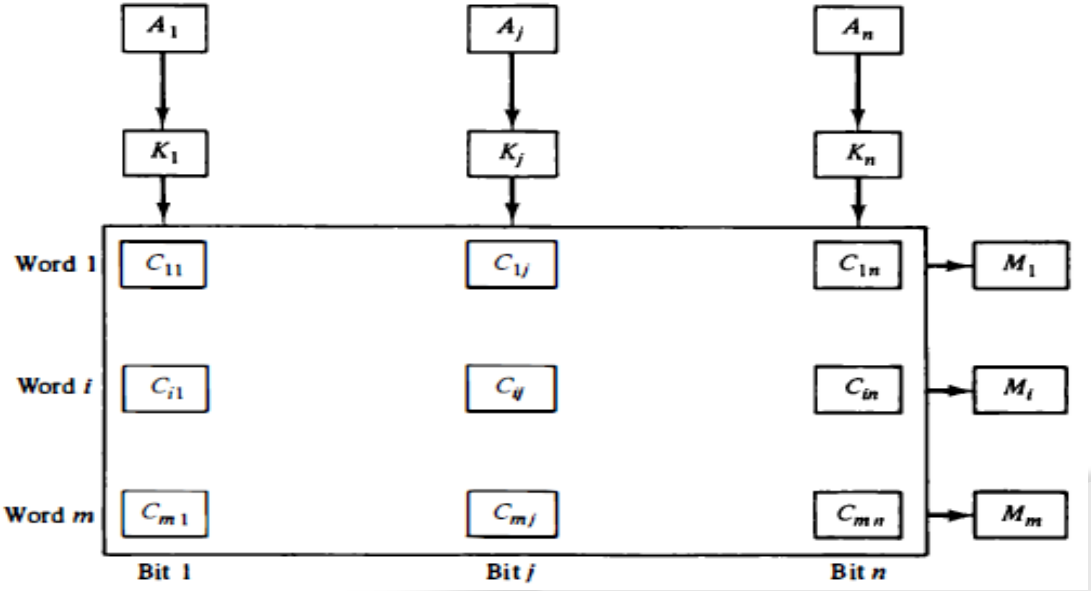



Figure: Associative memory of m word, n cells per word.

# Associative Memory

- The internal organization of a typical cell Cij is shown in Fig. It consists of a flip-flop storage element Fi 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 Mi.

**Match Logic**

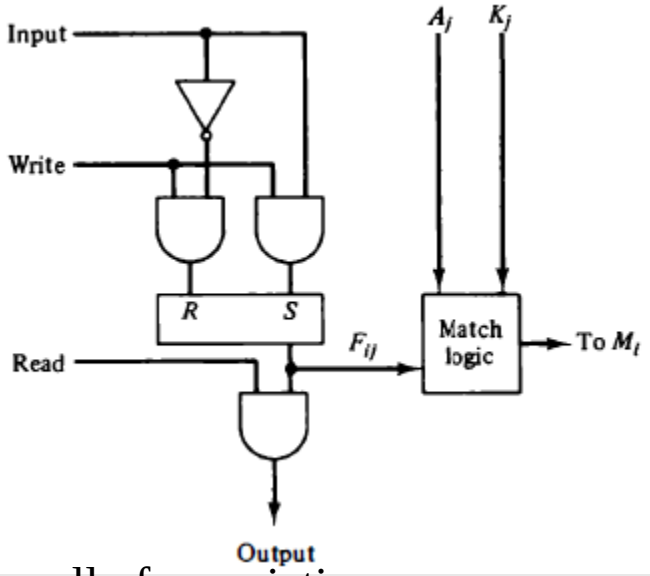



Figure: One cell of associative memory.

# Associative Memory

- 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 Aj = Fij for j = 1, 2, . . . , 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 xj = 1 if the pair of bits in position j are equal; otherwise, xj = 0. For a word i to be equal to the argument in A we must have all xi variables equal to 1. This is the condition for setting the corresponding match bit M, 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.

# Cache Memory

- 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. It is placed between the CPU and main memory as illustrated in Fig.
- The cache memory access time is less than the access time of main memory by a factor of 5 to 10. The cache is the fastest component in the memory hierarchy and approaches the speed of CPU components.
- The fundamental idea of cache organization is that by keeping the most frequently accessed instructions and data in the fast cache memory, the average memory access time will approach the access time of the cache.

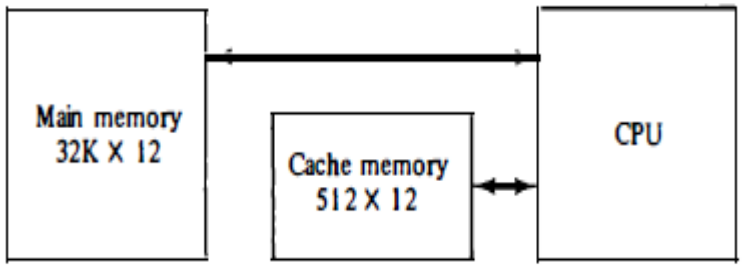



Figure: Example of cache memory.

# Cache Memory

- The basic operation of the cache 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 containing the one just accessed is then transferred from main memory to cache memory.
- 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 hit . If the word is not found in 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.
- For example, a computer with cache access time of 100 ns, a main memory access time of 1000 ns, and a hit ratio of 0.9 produces an average access time of 200 ns. This is a considerable improvement over a similar computer without a cache memory, whose access time is 1000 ns.

# Cache Memory

- The transformation of data from main memory to cache memory is referred to as a mapping process. Three types of mapping procedures are of practical interest when considering the organization of cache memory:

1. Associative mapping
2. Direct mapping
3. Set-associative mapping

**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 diagram shows three words presently stored in the cache.
- 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.

# Cache Memory

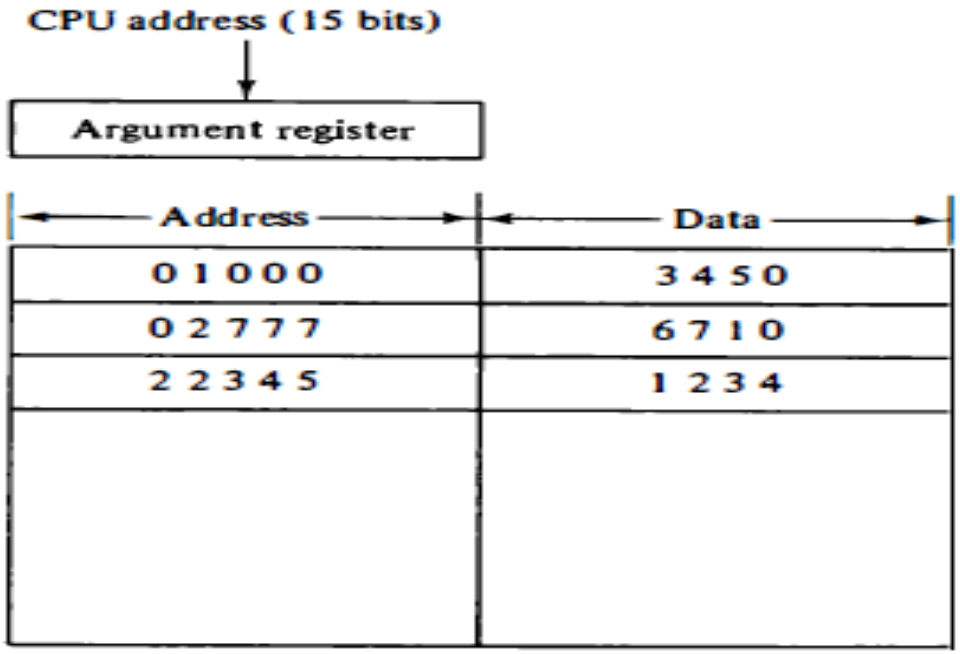



Figure: Associative mapping cache (all numbers in octal).

- 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.

# Cache Memory

**Direct Mapping:**

- Associative memories are expensive compared to random-access memories because of the added logic associated with each cell. 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.
- 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.

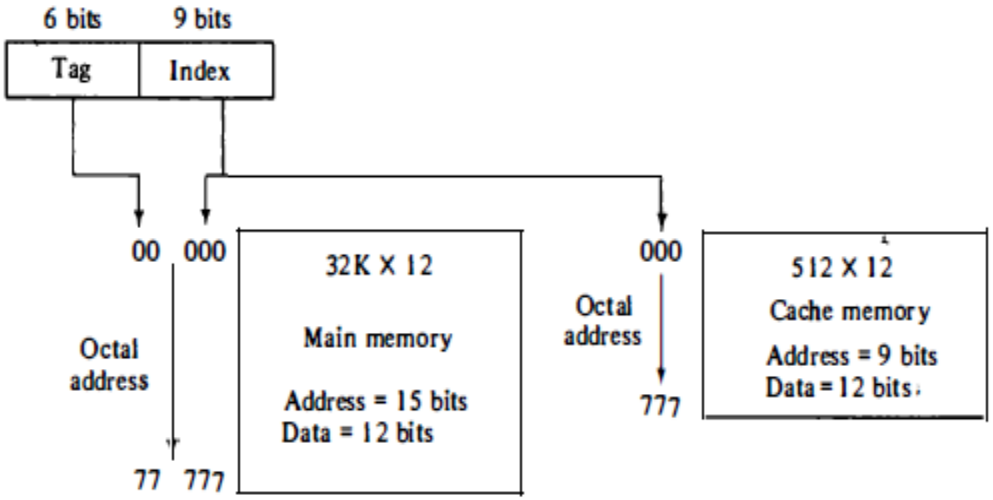



Figure: Addressing relationships between main and cache memories.

# Cache Memory

- In the general case, there are 2k words in cache memory and 2n words 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.

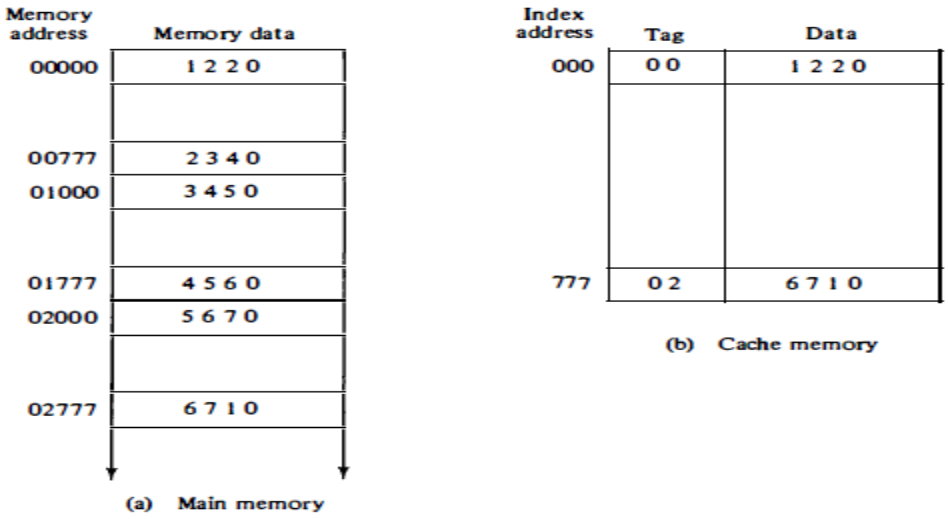



Figure: Direct mapping cache organization.

# Cache Memory

- 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 the hit ratio can drop considerably if two or more words whose addresses have the same index but different tags are accessed repeatedly.
- 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.

# Cache Memory

- 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 x 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.

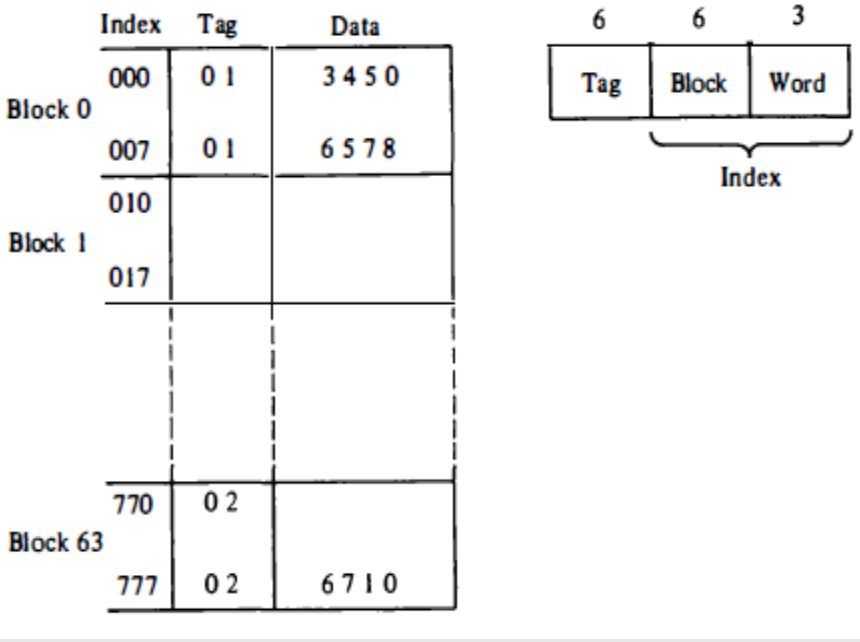
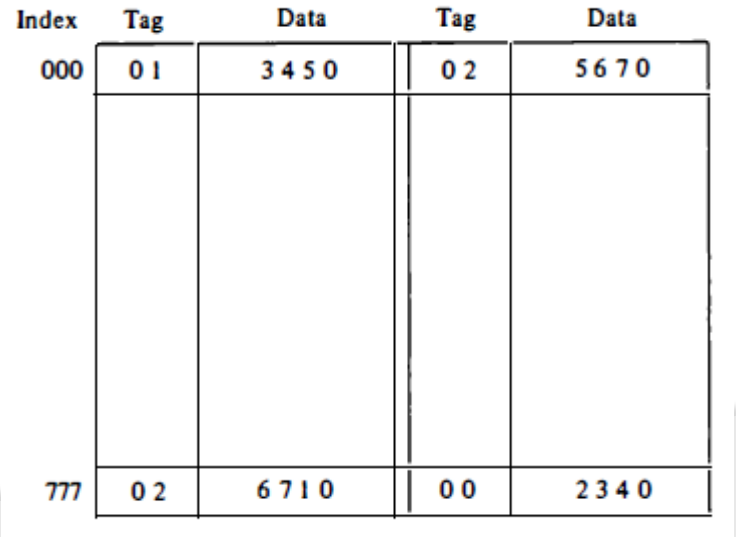



**Figure:** Direct mapping cache with block size of 8 words.

# Cache Memory

## Set-Associative Mapping:

- It was mentioned previously that 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.
- A third type of cache organization, called set-associative mapping, 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.



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

**Figure:** Two-way set associative mapping cache.

# Cache Memory

- The octal numbers listed in Fig. are with reference to the main memory contents illustrated in Fig.(a). 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 the CPU generates a memory request, the index value of the address is used to access the cache. The tag field of the CPU address is then compared with both tags in the cache to determine if a match occurs.
- thus the name "set-associative." The hit ratio will improve as the set size increases because more words with the same index but different tags can reside in cache.

# Cache Memory

**write-through:**

- The simplest and most commonly used procedure is to update main memory with every memory write operation, with cache memory being updated in parallel if it contains the word at the specified address. This is called the write-through method. This method has the advantage that main memory always contains the same data as the cache.

**write-back:**

- The second procedure is called the write-back method. In this method only the cache location is updated during a write operation. The location is then marked by a flag so that later when the word is removed from the cache it is copied into main memory.

# Virtual Memory

- Virtual memory is a concept used in some large computer systems that permit the user to construct programs as though a large memory space were available, equal to the totality of auxiliary memory. Each address that is referenced by the CPU goes through an address mapping from the so-called virtual address to a physical address in main memory.
- Virtual memory is used to give programmers the illusion that they have a very large memory at their disposal, even though the computer actually has a relatively small main memory. A virtual memory system provides a mechanism for translating program-generated addresses into correct main memory locations.

# Virtual Memory

**Address Space And Memory Space**

- An address used by a programmer will be called a virtual address, and the set of such addresses the address space. An address in main memory is called a location or physical address. The set of such locations is called the memory space.
- Thus the address space is the set of addresses generated by programs as they reference instructions and data; the memory space consists of the actual main memory locations directly addressable for processing.
- As an illustration, consider a computer with a main-memory capacity of 32K words (K = 1024). Fifteen bits are needed to specify a physical address in memory since 32K = 215.
- Suppose that the computer has available auxiliary memory for storing 220 = 1024K words. Denoting the address space by N and the memory space by M, we then have for this example N = 1024K and M = 32K.

# Virtual Memory

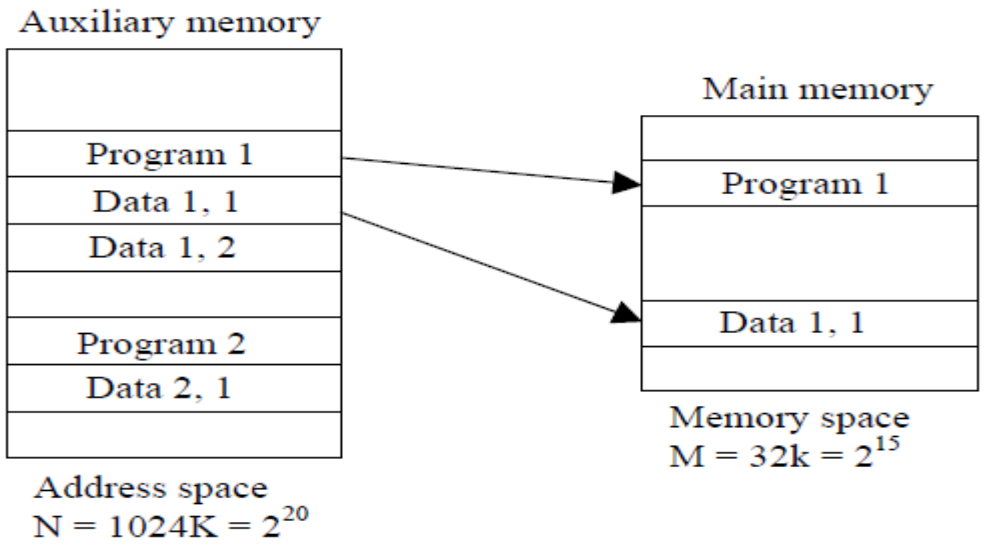



Figure : Relation between address and memory space in a virtual memory system.

- In a virtual memory system, programmers are told that they have the total address space at their disposal. Moreover, the address field of the instruction code has a sufficient number of bits to specify all virtual addresses.
- In our example, the address field of an instruction code will consist of 20 bits but physical memory addresses must be specified with only 15 bits. Thus CPU will reference instructions and data with a 20-bit address.

# Virtual Memory

- To map a virtual address of 20 bits to a physical address of 15 bits. The mapping is a dynamic operation, which means that every address is translated immediately as a word is referenced by CPU. The mapping table may be stored in a separate memory as shown in Fig.in main memory.

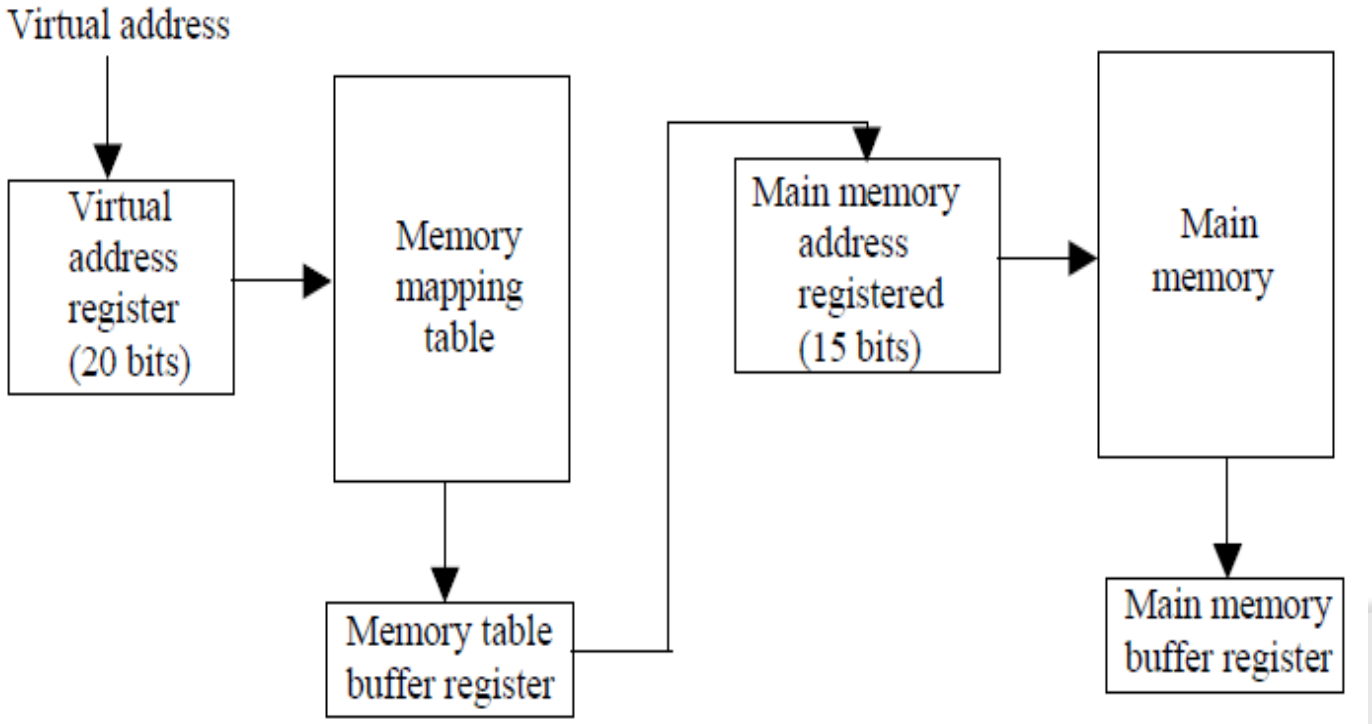



Figure: Memory table for mapping a virtual address.

# Virtual Memory

**Address Mapping Using Pages**

- Consider a computer with an address space of 8K and a memory space of 4K. If we split each into groups of 1K words we obtain eight pages and four blocks as shown in Fig. At any given time, up to four pages of address space may reside in main memory in any one of the four blocks.
- The mapping from address space to memory space is facilitated if each virtual address is considered to be represented by two numbers: a page number address and a line within the page. In a computer with 2^p words per page, p bits are used to specify a line address and the remaining high-order bits of the virtual address specify the page number.
- In the example of Fig. a virtual address has 13 bits. Since each page consists of 2^10 = 1024 words, the high-order three bits of a virtual address will specify one of the eight pages and the low-order 10 bits give the line address within the page. Note that the line address in address space and memory space is the same; the only mapping required is from a page number to a block number.

# Virtual Memory

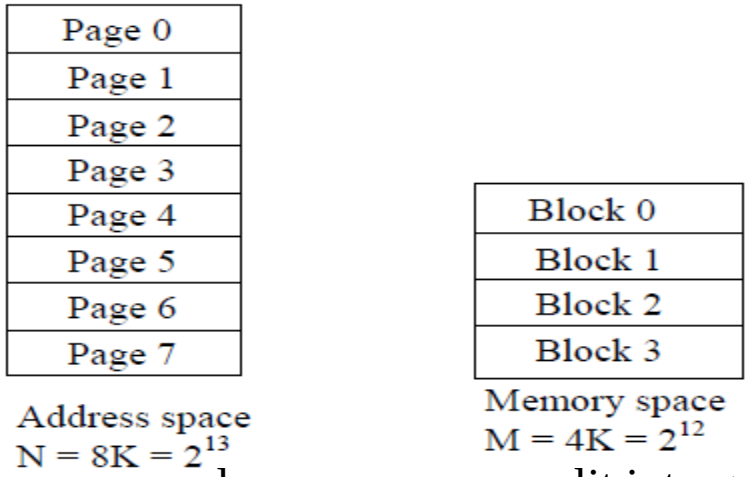



Figure: Address space and memory space split into groups of 1K words.

- The organization of the memory mapping table in a paged system is shown in Fig. The memory-page table consists of eight words, one for each page. The address in the page table denotes the page number and the content of the word gives the block number where that page is stored in main memory. The table shows that pages 1, 2, 5 and 6 are now available in main memory in blocks 3, 0, 1, and 2, respectively.
- A presence bit in each location indicates whether the page has been transferred from auxiliary memory into main memory. A 0 in the presence bit indicates that this page is not available in main memory. The CPU references a word in memory with a virtual address of 13 bits. The three high-order bits of the virtual address specify a page number and also an address for the memory-page table.

# Virtual Memory

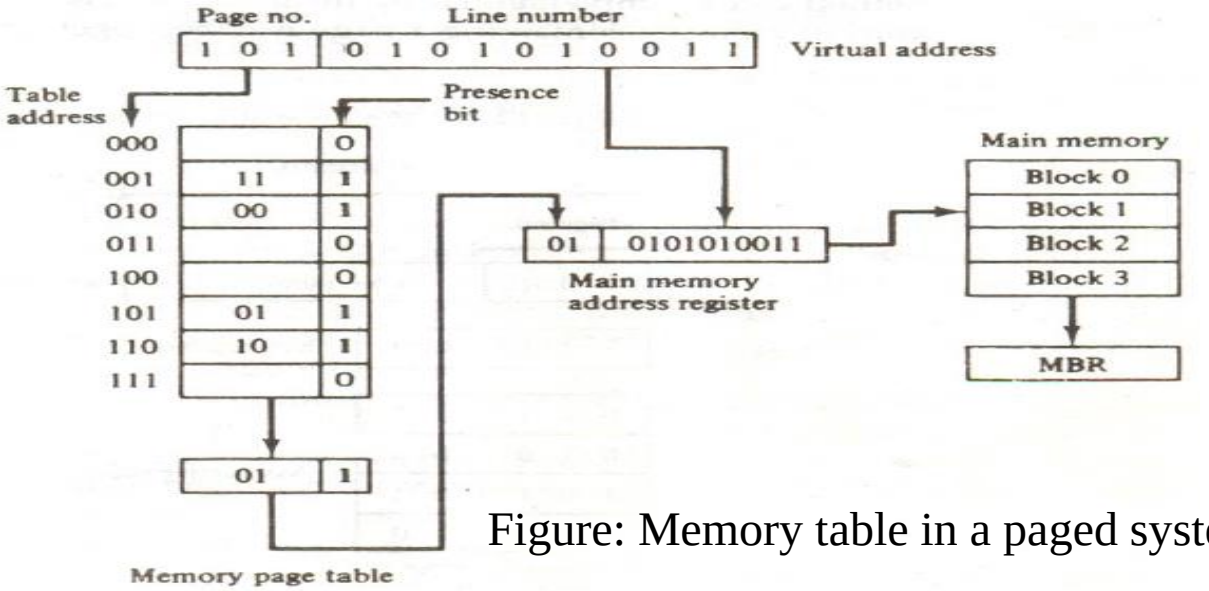



Figure: Memory table in a paged system.

- page table at the page number address is read out into the memory table buffer register. If the presence bit is a 1, the block number thus read is transferred to the two high-order bits of the main memory address register. The line number from the virtual address is transferred into the 10 low order bits of the memory address register.
- A read signal to main memory transfers the content of the word to the main memory buffer register ready to be used by the CPU. If the presence bit in the word read from the page table is 0, it signifies that the content of the word referenced by the virtual address does not reside in main memory.

# Virtual Memory

## Associative Memory Page Table

The page field in each word is compared with the page number in the virtual address. If a match occurs, the word is read from memory and its corresponding block number is extracted.

Consider again the case of eight pages and four blocks as in the example of Fig. We replace the random access memory-page table with an associative memory of four words as shown in Fig. Each entry in the associative memory array consists of two fields. The first three bits specify a field fro storing the page number. The last two bits constitute a field for storing the block number. The virtual address is placed in the argument register. The page number bits in the argument are compared with all page numbers in the page field of the associative memory. If the page number is found, the 5-bit word is read out from memory. The corresponding block number, being in the same word, is transferred to the main memory address register. If no match occurs, a call to the operating system is generated to bring the required page from auxiliary memory.

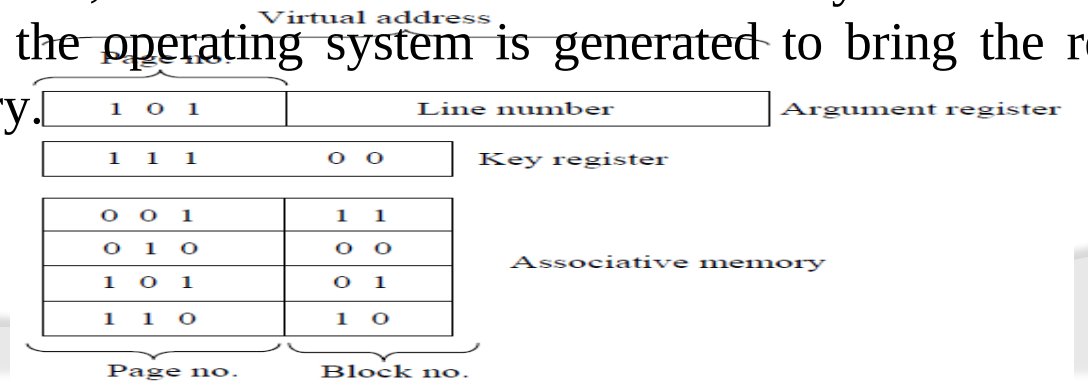



Figure: An associative memory page table.

- Semiconductor RAMs

✓ Internal Organization

✓ Static Memories

✓ Dynamic RAMs

i. Synchronous and Asynchronous DRAMs

ii. Structure of larger memories

✓ Read-only Memories

✓ Cache Memories – Mapping functions

- Nonvolatile Solid-State Memory Technologies
- Solid state drives (Volatile memory technologies)

# Semiconductor Memory

- Semiconductor memory is a type of digital memory technology that uses semiconductors, such as silicon, to store and retrieve digital data.
- It is commonly used in electronic devices, such as computers, smartphones, and other digital devices, as a primary or secondary storage medium.
- The memory implemented using the semiconductor chips is semiconductor memory.

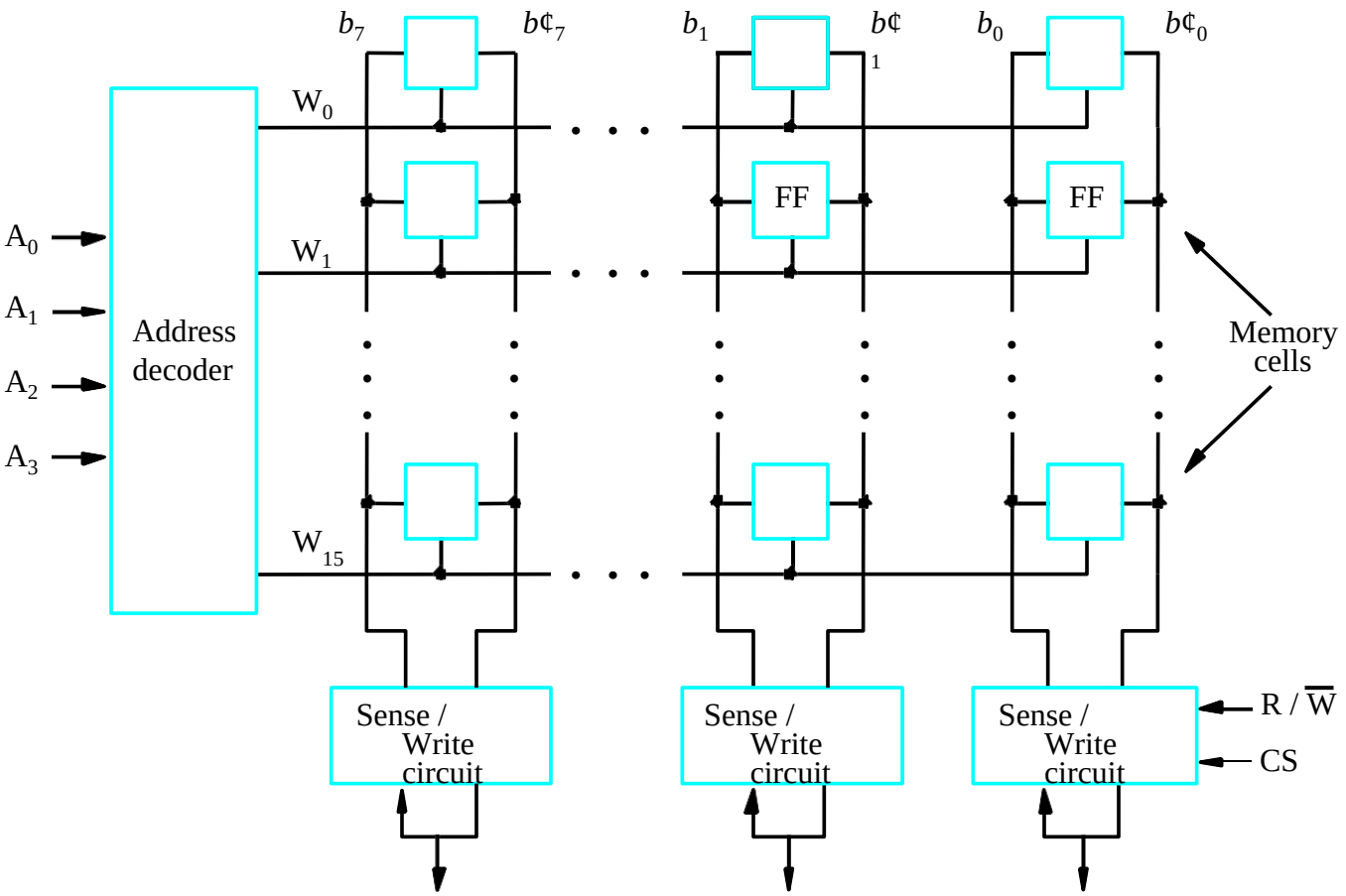
# Semiconductor Memory

## Internal Organization

**Internal organization of memory chips**

- Each memory cell can hold one bit of information.
- Memory cells are organized in the form of an array.
- One row is one memory word.
- All cells of a row are connected to a common line, known as the "word line".
- Wo r d line is connected to the address decoder.
- Sense/write circuits are connected to the data input/output lines of the memory chip.

# Semiconductor Memory




**Internal organization of memory chips**

# Semiconductor Memory

Semiconductor memory can be divided into two main categories: **Volatile and Non- volatile memory**.

- **Volatile memory**, such as dynamic random-access memory (**DRAM**) and static random- access memory (**SRAM**), requires power to retain stored data.
- **Non-volatile memory**, such as read-only memory (**ROM**), flash memory, and Electrically Erasable Programmable Read-Only Memory (**EEPROM**), can retain data even when power is turned off.

# Semiconductor Memory

- Semiconductor memory has several advantages over other types of memory, including faster access times, higher data transfer rates, and lower power consumption.
- These benefits make it an ideal storage medium for a wide range of electronic devices and applications.
- The semiconductor main memory subsystem includes a critical component of a computer system that stores and retrieves data and instructions needed by the processor to perform tasks.

# Semiconductor Memory

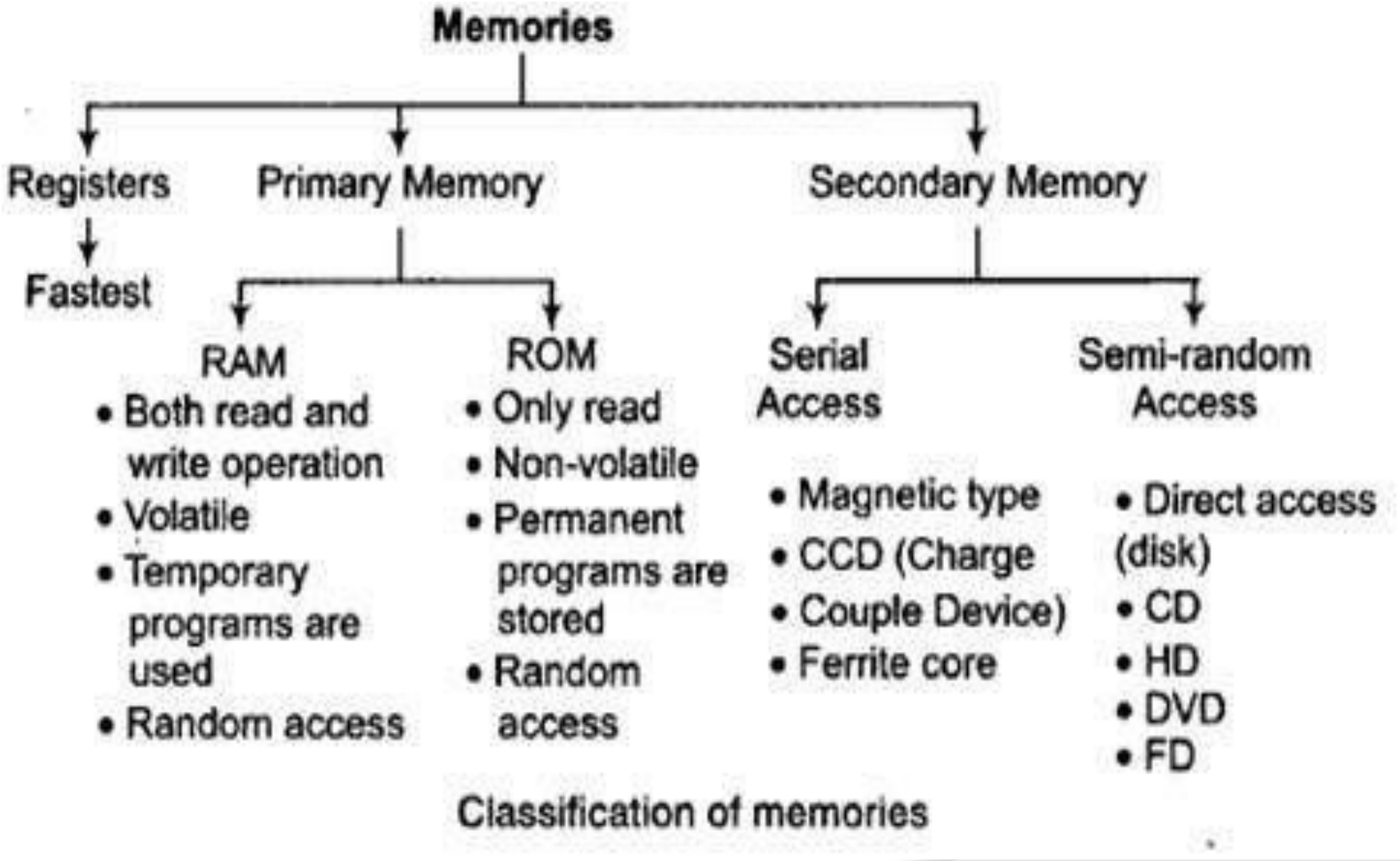
## Types of Semiconductor Memory

- Semiconductor memory is a type of electronic memory that uses semiconductor devices such as transistors to store digital information.
- Semiconductor memory is widely used in modern electronic devices such as computers, mobile phones, and digital cameras.

**Here are the main types of semiconductor memory:**

- Registers
  - In CPU
- Internal or Main memory
  - May include one or more levels of cache
  - “RAM”
- External memory
  - Backing store

# Semiconductor Memory




Classification of memories

# Semiconductor Memory

## 1. Registers and their Types

Registers are memories located within the Central Processing Unit (CPU). Various types of registers are available within the CPU. Registers are small but the CPU can access them quickly. Some of the registers available in the system are given below.

- **Instruction Register**
- **ALU I/O registers**
- **Status Register**
- **Stack pointer register**
- **The program counter, etc.**

# Semiconductor Memory

## 2. Random Access Memory (RAM):

RAM is a type of volatile memory that can be read and written to. RAM is used as temporary storage for data and program instructions when the computer is running.

RAM is further divided into two types:

- **Static RAM (SRAM)**: SRAM is a type of RAM that uses flip-flops to store each bit of data. SRAM is faster and more expensive than DRAM.
- **Dynamic RAM (DRAM)**: DRAM is a type of RAM that uses capacitors to store each bit of data. DRAM is slower and less expensive than SRAM.

# Semiconductor Memory

### 3. Read-Only Memory (ROM):

ROM is a type of non-volatile memory that can only be read, not written to. ROM is used to store program instructions and data that are not intended to be modified. ROM is further divided into two types:

- **Mask ROM:** Mask ROM is a type of ROM that is programmed during the manufacturing process. Once programmed, the contents of the ROM cannot be changed.
- **Programmable ROM (PROM):** PROM is a type of ROM that can be programmed once by the user. Once programmed, the contents of the PROM cannot be changed.

- **Electrically Erasable Programmable Read-Only Memory (EEPROM)**: EEPROM is a type of non-volatile memory that can be programmed and erased electrically. EEPROM is used to store data that needs to be modified occasionally, such as the BIOS settings on a computer motherboard.
- **Flash Memory:** Flash memory is a type of non-volatile memory that can be electrically erased and reprogrammed. Flash memory is used in a wide range of electronic devices such as USB drives, digital cameras, and mobile phones. Flash memory is further divided into two types:

1. NOR Flash: NOR gate flash is used for executing code directly from the memory. It has a slower write speed but a faster read speed.
2. NAND Flash: NAND gate flash is used for data storage. It has a faster write speed but a slower read speed.

# Semiconductor Memory

## Advantages of Static RAM Over Dynamic RAM

**Static RAM and Dynamic RAM** both are types of Read Access Memory. It can be used for the purpose of data storage. Here few differences between SRAM and DRAM are discussed below.

- The access time of SRAM is less and thus these memories are faster memories.
- As SRAM consists of flip-flops thus, refreshing is not required.
- Less number of memory cells are required in SRAM for a unit area.

# Synchronous and Asynchronous DRAM

DRAM requires constant refreshes to retain data. **Synchronous DRAM and Asynchronous DRAM** are two types of DRAM.

The **key difference** between **Synchronous and Asynchronous DRAM** is that the

**Synchronous DRAM** uses the system clock to coordinate the memory access **while Asynchronous DRAM** does not use the system clock to coordinate the memory access.

## What is Synchronous DRAM?

In Synchronous DRAM, the system clock coordinates or synchronizes the memory accessing.

Therefore, the CPU knows the timing or the exact number of cycles in which the data will be available from the RAM to the input, output bus. It increases memory read and write speed.

Overall, the Synchronous DRAM is faster in speed and operates efficiently than the normal DRAM.

# Synchronous and Asynchronous DRAM

**What is Asynchronous DRAM?**

- The first personal computers used asynchronous DRAM. It is an older version of DRAM.
- In asynchronous DRAM, the system clock does not coordinate or synchronizes the memory accessing.
- When accessing the memory, the value appears on the input, output bus after a certain period. Therefore, it has some latency that minimizes the speed.
- Usually, asynchronous RAM works in low-speed memory systems but not appropriate for modern high-speed memory systems.
- At present, the manufacturing of asynchronous RAM is quite low.
- **Today, synchronous DRAM is used instead of the asynchronous DRAM.**

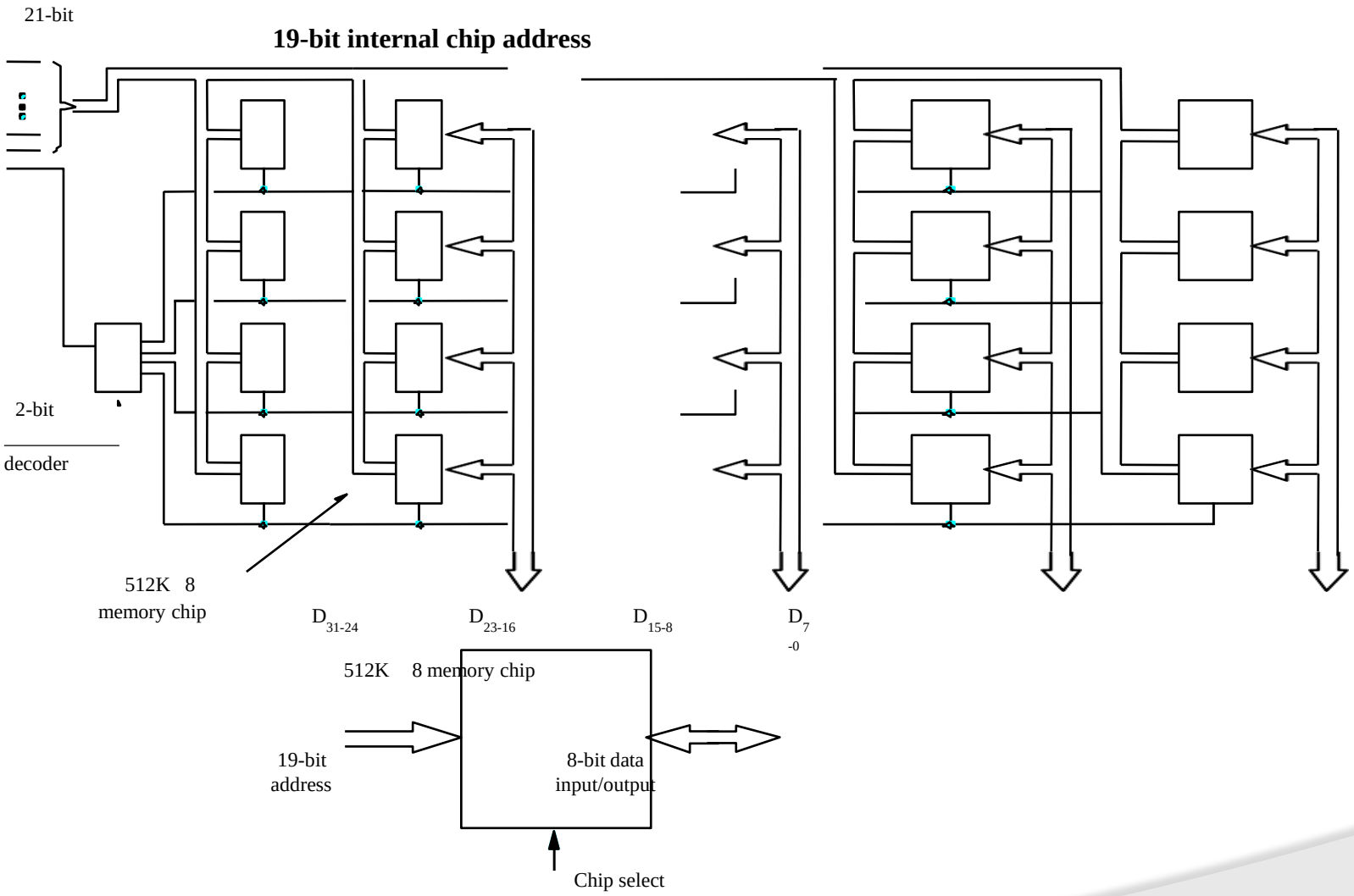
# Synchronous and Asynchronous DRAM

**What is the Difference Between Synchronous and Asynchronous DRAM?**

- Synchronous DRAM uses a system clock to coordinate memory accessing while
- Asynchronous DRAM does not use a system clock to synchronize or coordinate memory accessing.
- Synchronous DRAM is faster and efficient then asynchronous DRAM.
- Furthermore, synchronous DRAM provides high performance and better control than the asynchronous DRAM.
- Modern high-speed PCs uses synchronous DRAM while older low-speed PCs used asynchronous DRAM.

# Structure of Larger Memories

# Structure of Larger Memories




**Organization of a 2M * 32 memory module using 512 K * 8 static chips**

- Implement a memory unit of 2M words of 32 bits each.
- Use 512x8 static memory chips
- Each column consists of 4 chips.
- Each chip implements one byte position.
- A chip is selected by setting its chip select control line to 1.
- Selected chip places its data on he data output line, outputs of other chips are in high impedance state.
- 21 bits to address a 32-bit word.
- High order 19 bits are needed to select the row and lower order 2 bits for select column by activating the
- Four Chip Select signals.
- 19 bits are used to access specific byte locations inside the selected chip.

# Structure of Larger Memories

## Memory System Considerations

The choice of a RAM chip for a given application depends on several factors:

**Cost, speed, power, size…**

SRAMs are faster, more expensive, smaller.

DRAMs are slower, cheaper, larger.

# Read-Only Memories

ROM is used for storing programs that are **PERMENTLY resident** in the computer and for tables of constants that do not change in value once the production of the computer is completed

The **ROM portion of main memory** is needed for storing an **initial program called bootstrap loader,**

ROM

PROM: programmable ROM

EPROM: erasable, reprogrammable ROM

EEPROM: can be programmed and erased electrically

**Flash Memory**

Difference: only possible to write an entire block of cells ,but read the contents of a single cell

Low power consumption

Use in portable equipment

Flash cards

Flash drives

# Cache Memories – Mapping Functions

**Cache Memory**


Figure: Use of a Cache Memory.

- Processor issues a Read request, a block of words is transferred from the main memory to the cache, one word at a time. Subsequent references to the data in this block of words are found in the cache.
- At any given time, only some blocks in the main memory are held in the cache. Which blocks in the main memory are in the cache is determined by a “mapping function.
- When the cache is full, and a block of words needs to be transferred
- from the main memory, some block of words in the cache must be replaced. This is determined by a “replacement algorithm”.

# Cache Memories – Mapping Functions

- 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 cache memory
- The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component.
- When CPU needs to access memory, the cache is examined

1. Locality of reference

✓ temporal

✓ spatial

2. Cache block – cache line

✓ A set of contiguous address locations of some size

# Cache Memories – Mapping Functions

**Principle of Locality or locality of reference:**

- Program accesses a relatively small portion of the address space at any instant of time.
- Temporal locality and spatial locality.

**Temporal locality or locality in time:**

- Keep most recently accessed data items closer to the processor. Spatial locality (locality in space):
- Move blocks consisting of contiguous words to 'upper' levels.
- **Cache Hit:** data appears in some block

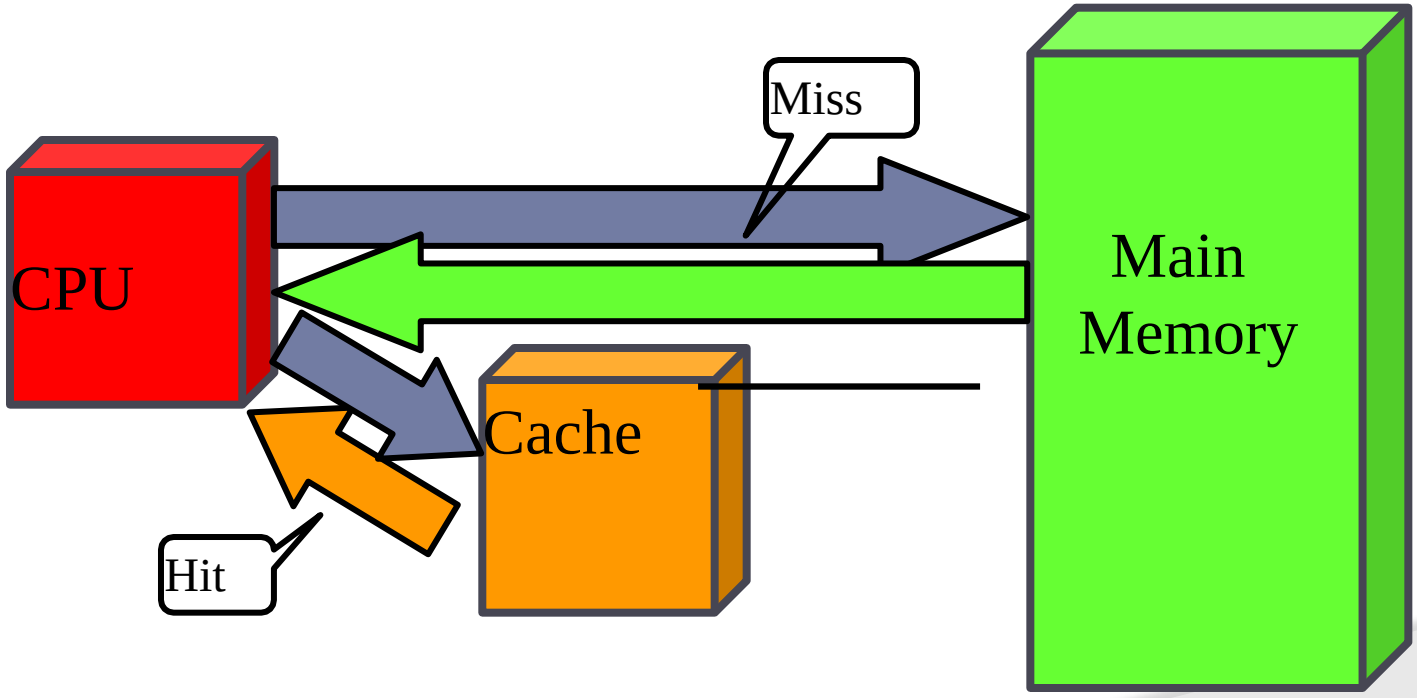
  Hit rate: the fraction of memory access found in the upper level.
- **Cache Miss:** data is not found, and needs to be retrieved from a block.

**The success rate in accessing information at various levels of the memory hierarchy – hit rate / miss rate.**

- A **miss causes extra time** needed to bring the desired information into the cache.
- Hit rate can be **improved by increasing block size**, while considering the cache size .

# Cache Memories – Mapping Functions

- The basic characteristic of cache memory is its fast access time.
- Therefore, very little or no time must be wasted when searching the words in the cache.



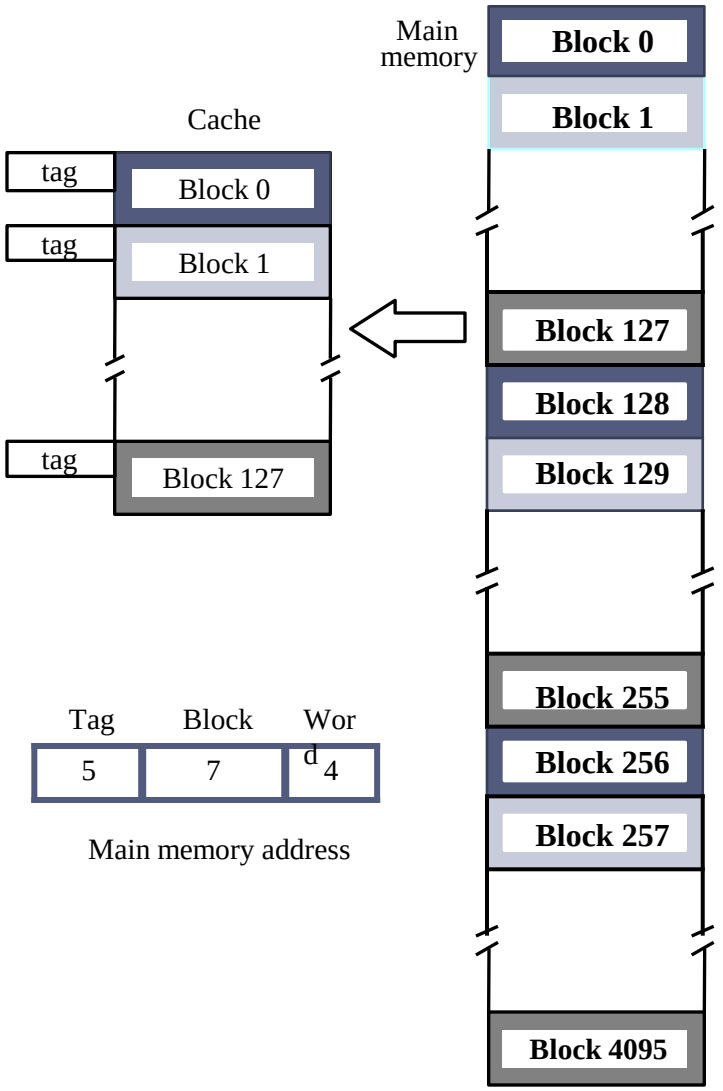
# Cache Memories – Mapping Functions

- The transformation of data from main memory to cache memory is referred to as a mapping process, there are three types of mapping:

1. **Associative mapping**
2. **Direct mapping**
3. **Set-associative mapping**

- In general case, there are 2^k words in cache memory and 2^n words in main memory .
- The n bit memory address is divided into two fields: k-bits for the index and n-k bits for the tag field.

# Cache Memories – Mapping Functions




## Direct mapping

- Block j of the main memory maps to j modulo 128 of the cache. 0 maps to 0, 129 maps to 1.
- More than one memory block is mapped onto the same position in the cache.
- May lead to contention for cache blocks even if the cache is not full.
- Resolve the contention by allowing new block to replace the old block, leading to a trivial replacement algorithm.
- Memory address is divided into three fields:

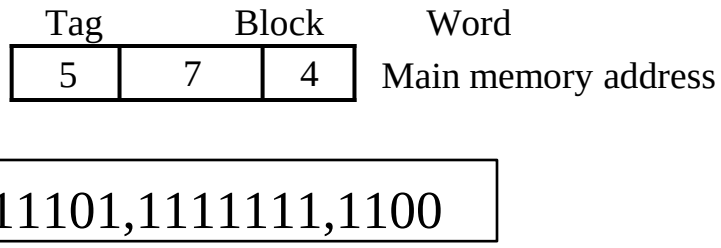
✓ Low order 4 bits determine one of the 16 words in a block.

✓ When a new block is brought into the cache, the next 7 bits determine which cache block this new block is placed in.

✓ High order 5 bits determine which of the possible 32 blocks is currently present in the cache. These are tag bits.

- Simple to implement but not very flexible.

# Cache Memories – Mapping Functions

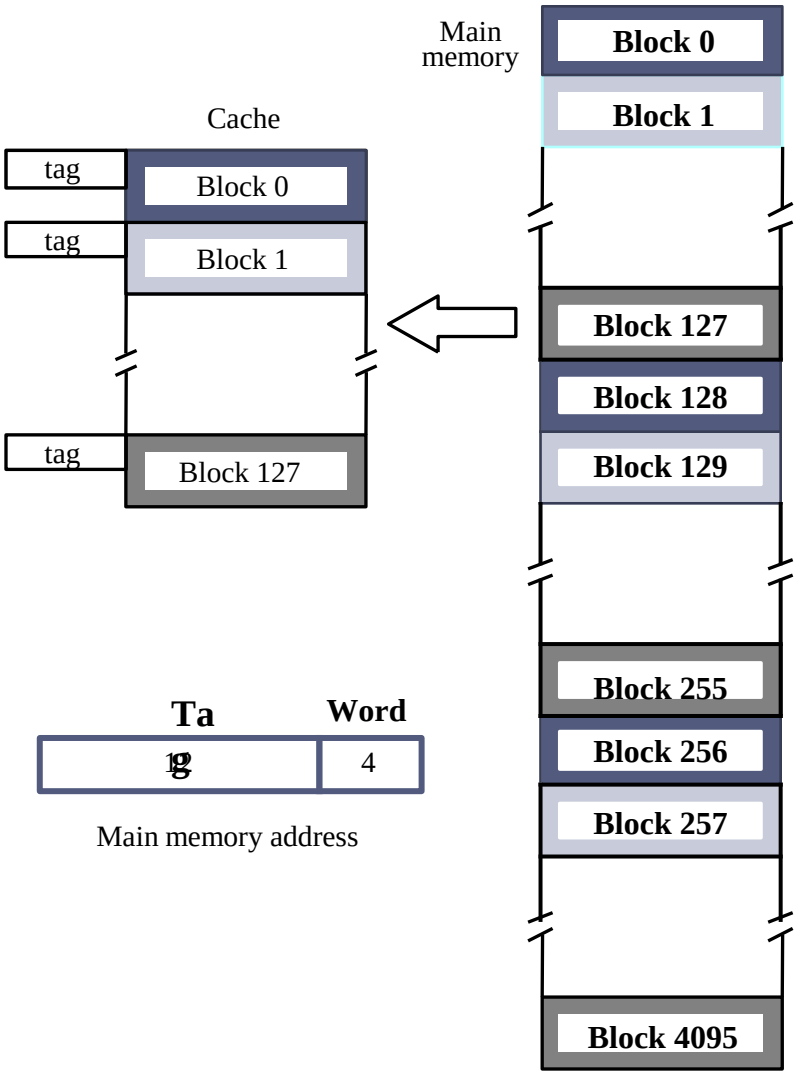



Tag: 11101

Block: 1111111=127, in the 127$^{th}$ block of the cache

W o r d:1100=12, the 12$^{th}$ word of the 127$^{th}$ block in the cache

# Cache Memories – Mapping Functions




## Associative mapping

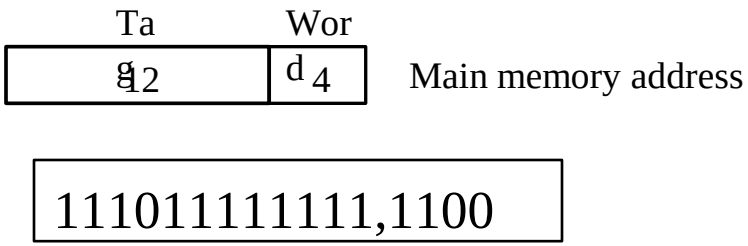
- Main memory block can be placed into any cache position.
- Memory address is divided into two fields:

- Low order 4 bits identify the word within a block.

-high order 12 tag bits Identify which of the 4096 blocks that are resident in the cache 4096=212.

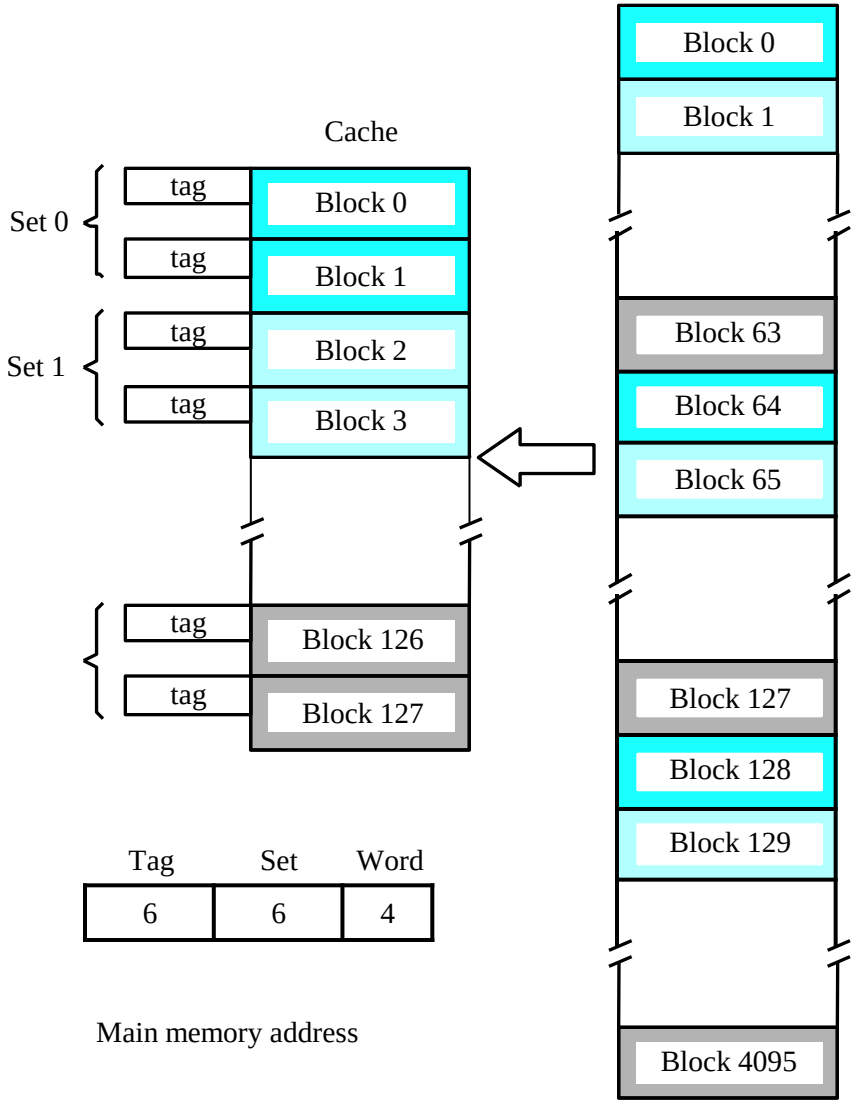
- **Flexible, and uses cache efficiently.**
- Replacement algorithms can be used to replace an existing block in the cache when the cache is full.

# Cache Memories – Mapping Functions




- Tag: 111011111111
- Wo r d:1100=12, the 12th word of a block in the cache

# Cache Memories – Mapping Functions




## Set-Associative Mapping

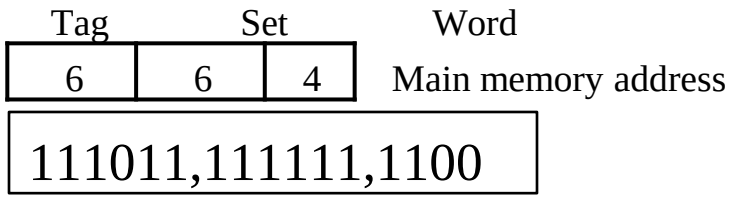
- Blocks of cache are grouped into sets.
- Mapping function allows a block of the main memory to reside in any block of a specific set.
- Divide the cache into 64 sets, with two blocks per set.
- Memory address is divided into three fields:

✓ 6 bit Set field determines the set number.

✓ High order 6 bit fields are compared to the tag fields of the two blocks in a set.

- Number of blocks per set is a design parameter.

# Cache Memories – Mapping Functions




Tag: 111011

Set: 111111=63, in the 63$^{th}$ set of the cache

W o r d:1100=12, the 12$^{th}$ word of the 63th set in the cache

# Non volatile solid-state memory technologies

Nonvolatile solid-state memory technologies retain data even when power is turned off. Here are some key types:

- **Flash Memory:** The most common nonvolatile memory, used widely in USB drives, SSDs, and SD cards. It uses floating-gate transistors to store charges.

**Example: NAND Flash, commonly found in SSDs and USB drives.**

- **Magneto Resistive Random-Access Memory (MRAM):** Uses magnetic storage elements to store data, offering high speed and endurance with nonvolatility.

**Example: Ever spin MRAM, often used in automotive and industrial applications.**

# Non volatile solid-state memory technologies

- Phase-Change Memory (PCM): Operates by changing the phase of a material (typically chalcogenide) between amorphous and crystalline states.

Example: Intel's 3D XPoint technology, marketed as Optane.

- Resistive RAM (ReRAM): Uses a metal oxide layer where resistance changes due to an applied voltage, making it highly scalable.

Example: Panasonic ReRAM, used in IoT and embedded systems.

- Ferroelectric RAM (FeRAM): Stores data using a ferroelectric layer, allowing low power consumption and high-speed writes.

Example: Texas Instruments FeRAM, often used in smart cards and medical devices.

Each technology has unique advantages in terms of speed, durability, and power efficiency, with applications tailored to these strengths.

# Solid State Drives

- Solid-state drive (SSD) is a solid-state storage device that uses integrated circuit assemblies as memory to store data. SSD is also known as a solid-state disk although SSDs do not have physical disks.
- There are no moving mechanical components in SSD. This makes them different from conventional electromechanical drives such as Hard Disk Drives (HDDs) or floppy disks, which contain movable read/write heads and spinning disks.
- SSDs are typically more resistant to physical shock, run silently, and have quicker access time, and lower latency compared to electromechanical devices.
- It is a type of non-volatile memory that retains data even when power is lost. SSDs may be constructed from random-access memory (RAM) for applications requiring fast access but not necessarily data persistence after power loss.
- Batteries can be employed as integrated power sources in such devices to retain data for a certain amount of time after external power is lost.