MEMORY

BASIC CONCEPTS

- The maximum size of the memory that can be used in any computer is determined by the addressing scheme.
  For example,
  - a computer that generates 16-bit addresses is capable of addressing up to 2^{16} = 64K (kilo) memory locations.
  - Machines whose instructions generate 32-bit addresses can utilize a memory that contains up to 2^{32} = 4G (giga) locations, whereas machines with 64-bit addresses can access up to 2^{64} = 16E (exa) $\approx 16 \times 10^{18}$ locations.
- The number of locations represents the size of the address space of the computer. The memory is usually designed to store and retrieve data in word-length quantities.
  - Consider, for example, a byte-addressable computer whose instructions generate 32-bit addresses. When a 32-bit address is sent from the processor to the memory unit, the high order 30 bits determine which word will be accessed. If a byte quantity is specified, the low-order 2 bits of the address specify which byte location is involved.
- The connection between the processor and its memory consists of address, data, and control lines, as shown in Figure A.

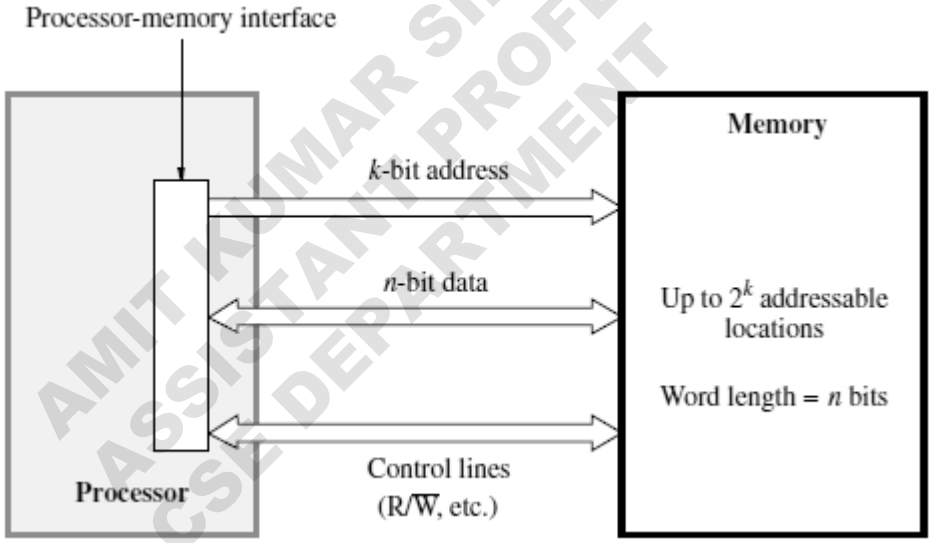


Figure A

The processor uses the address lines to specify the memory location involved in a data transfer operation, and uses the data lines to transfer the data. At the same time, the control lines carry the command indicating a Read or a Write operation and whether a byte or a word is to be transferred. The control lines also provide the necessary timing information and are used by the memory to indicate when it has completed the requested operation.

When the processor-memory interface receives the memory's response, it asserts the MFC signal. This is the processor's internal control signal that indicates that the

requested memory operation has been completed. When asserted, the processor proceeds to the next step in its execution sequence.

- A useful measure of the speed of memory units is the time that elapses between the initiation of an operation to transfer a word of data and the completion of that operation. This is referred to as the *memory access time*.
- Another important measure is the *memory cycle time*, which is the minimum time delay required between the initiation of two successive memory operations.
  For example, the time between two successive Read operations.
  The cycle time is usually slightly longer than the access time, depending on the implementation details of the memory unit.
- A memory unit is called a random-access memory (RAM) if the access time to any location is the same, independent of the location's address. This distinguishes such memory units from serial, or partly serial, access storage devices such as magnetic and optical disks.
  Access time of the latter devices depends on the address or position of the data. The technology for implementing computer memories uses semiconductor integrated circuits.

Key Characteristics of Computer Memory Systems

1. *Location*
   (i) Internal (e.g. processor registers, main memory, cache)
   - Internal memory is often equivalent to main memory. But there are other forms of internal memory.
   - The processor requires its own local memory, in the form of registers. Further, as we shall see, the control unit portion of the processor may also require its own internal memory. Cache is another form of internal memory.

   (ii) External (e.g. optical disks, magnetic disks, tapes)
   - External memory consists of peripheral storage devices, such as disk and tape, that are accessible to the processor via I/O controllers.
2. *Capacity*
   An obvious characteristic of memory is its capacity. For internal memory, this is typically expressed in terms of *bytes* (1 byte = 8 bits) or *words*. Common word lengths are 8, 16, and 32 bits. External memory capacity is typically expressed in terms of bytes.
   (i) Number of words specifies the number of words (8, 16, 32, etc. bits) available in the particular memory device.
   (ii) Number of bytes specifies the number of bytes (8-bits) available in the particular memory device.
3. *Unit of Transfer*
   For main memory, this is the number of bits read out of or written into memory at a time. The unit of transfer need not equal a *word* or an addressable unit. For external memory, data are often transferred in much larger units than a word, and these are referred to as *blocks*.
4. *Access Method*
   (i) Sequential Access
   Memory is organized into units of data, called records. Access must be made in a specific linear sequence. Stored addressing information is used to separate records and assist in the retrieval process. A shared read–write mechanism is used, and this must be moved from its current location to the desired location, passing and rejecting each intermediate

record. Thus, the time to access an arbitrary record is highly variable. Tape units are sequential access

(ii) Direct Access

As with sequential access, direct access involves a shared read–write mechanism. However, individual blocks or records have a unique address based on physical location. Access is accomplished by direct access to reach general vicinity plus sequential searching, counting, or waiting to reach the final location. Again, access time is variable.

Random Access

Each addressable location in memory has a unique, physically wired-in addressing mechanism. The time to access a given location is independent of the sequence of prior accesses and is constant. Thus, any location can be selected at random and directly addressed and accessed. Main memory and some cache systems are random access.

(iii)Associative Access

This is a random access type of memory that enables one to make a comparison of desired bit locations within a word for a specified match, and to do this for all words simultaneously. Thus, a word is retrieved based on a portion of its contents rather than its address. As with ordinary random-access memory, each location has its own addressing mechanism, and retrieval time is constant independent of location or prior access patterns. Cache memories may employ associative access.

5. *Performance*

(i) Access time

For random-access memory, this is the time it takes to perform a read or write operation, that is, the time from the instant that an address is presented to the memory to the instant that data have been stored or made available for use. For non-random-access memory, access time is the time it takes to position the read–write mechanism at the desired location.

(ii) Memory Cycle time

This concept is primarily applied to random-access memory and consists of the access time plus any additional time required before a second access can commence. This additional time may be required for transients to die out on signal lines or to regenerate data if they are read destructively. Note that memory cycle time is concerned with the system bus, not the processor.

(iii)Transfer rate

This is the rate at which data can be transferred into or out of a memory unit. For random-access memory, it is equal to 1/(cycle time). For non-random-access memory, the following relationship holds:

$$T_n = T_A + \frac{n}{R}$$

where

T_n = Average time to read or write n bits

T_A = Average access time

n = Number of bits

R = Transfer rate, in bits per second (bps)

6. *Physical Type*

A variety of physical types of memory have been employed. The most common today are

(i) Semiconductor

(ii) Magnetic

(iii)Optical

7. *Physical Characteristics*

(i) Volatile/nonvolatile

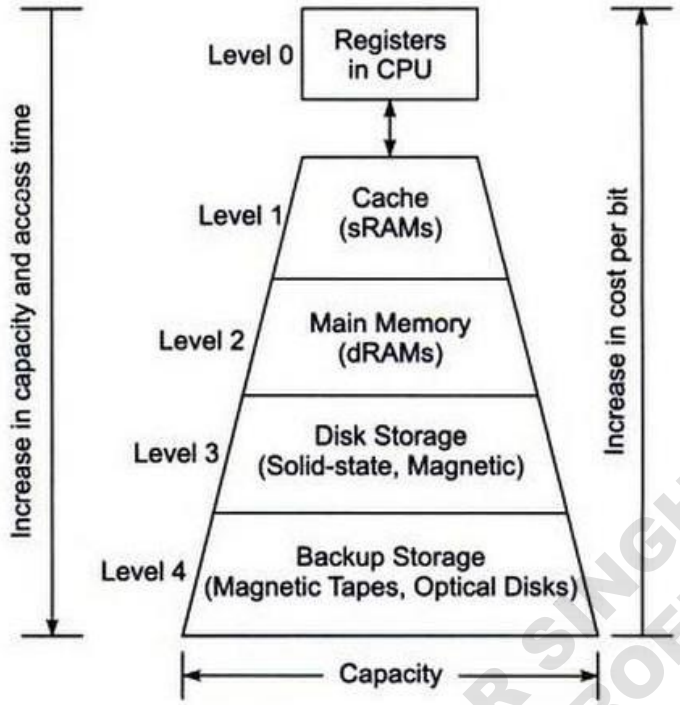
In a volatile memory, information decays naturally or is lost when electrical power is switched off. In a nonvolatile memory, information once recorded remains without deterioration until deliberately changed; no electrical power is needed to retain information. Magnetic-surface memories are nonvolatile. Semiconductor memory (memory on integrated circuits) may be either volatile or nonvolatile.

(ii) Erasable/ nonerasable

If data in the memory can be erased then memory is called as erasable. On the other hand, Nonerasable memory cannot be altered, except by destroying the storage unit. Semiconductor memory of this type is known as read-only memory (ROM). Of necessity, a practical nonerasable memory must also be nonvolatile.

Memory Hierarchy

All of the different types of memory units are employed effectively in a computer system. The entire computer memory can be viewed as the hierarchy depicted in figure.



The fastest access is to data held in processor registers. Therefore, if we consider the registers to be part of the memory hierarchy, then the processor registers are at the top in terms of speed of access. Of course, the registers provide only a minuscule portion of the required memory.

At the next level of the hierarchy is a relatively small amount of memory that can be implemented directly on the processor chip. This memory, called a *processor cache*, holds copies of the instructions and data stored in a much larger memory that is provided externally. There are often two or more levels of cache. A primary cache is always located on the processor chip. This cache is small and its access time is comparable to that of processor registers. The primary cache is referred to as the *level 1* (L1) cache. A larger, and hence somewhat slower, secondary cache is placed between the primary cache and the rest of the memory. It is referred to as the *level 2* (L2) cache. Often, the L2 cache is also housed on the processor chip. Some computers have a *level 3* (L3) cache of even larger size, in addition to the L1 and L2 caches. An L3 cache, also implemented in SRAM technology, may or may not be on the same chip with the processor and the L1 and L2 caches.

The next level in the hierarchy is the *main memory*. This is a large memory implemented using dynamic memory components, typically assembled in memory modules such as DIMMs. The main memory is much larger but significantly slower than cache memories. In a computer with a processor clock of 2 GHz or higher, the access time for the main memory can be as much as 100 times longer than the access time for the L1 cache. Disk devices provide a very large amount of inexpensive memory, and they are widely used as secondary storage in computer systems. They are very slow compared to the main memory. They represent the bottom level in the memory hierarchy.

SEMICONDUCTOR RAM MEMORIES

The basic technology for implementing main memories uses semiconductor integrated circuits. Semiconductor random-access memories (RAMs) are available in a wide range of speeds. Their cycle times range from 100 ns to less than 10 ns.

INTERNAL ORGANIZATION OF MEMORY CHIPS:

Memory cells are usually organized in the form of array, in which each cell is capable of storing one bit of information. Each row of cells constitutes a memory word and all cells of a row are connected to a common line called as word line which is driven by the address decoder on the chip. The cells in each column are connected to Sense / Write circuit by two bit lines. Figure 1 shows the possible arrangements of memory cells.

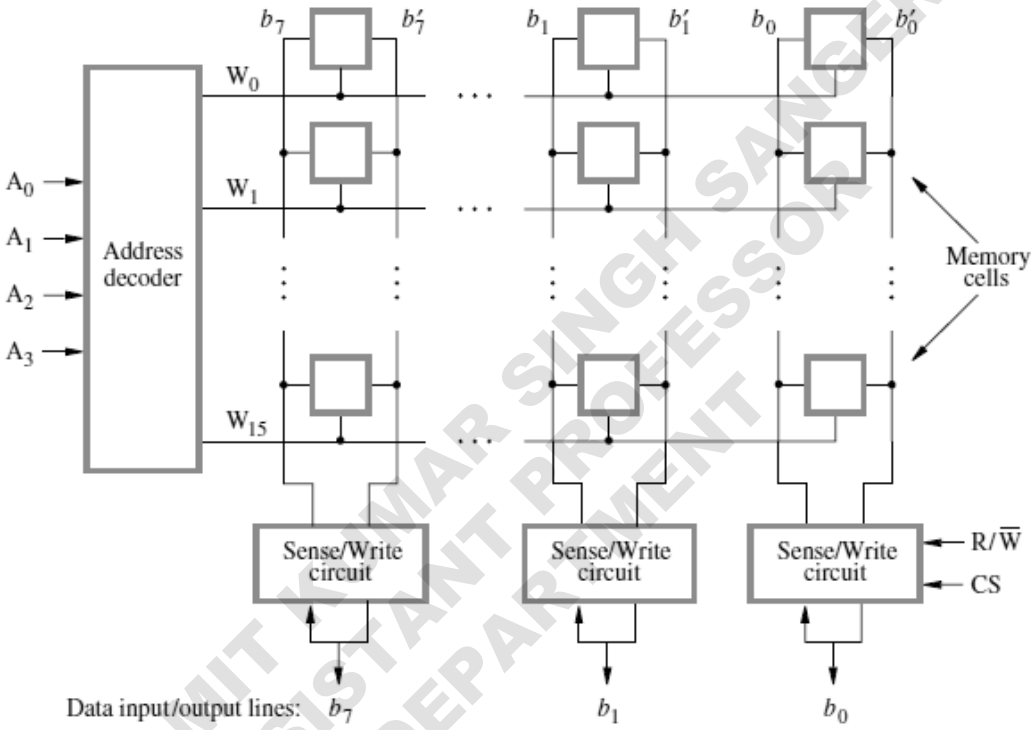


Figure 1 Organization of bit cells in a memory chip.

The Sense / Write circuits are connected to data input or output lines of the chip.

During a Read operation, these circuits sense, or read, the information stored in the cells selected by a word line and place this information on the output data lines.

During a write operation, the sense / write circuit receives input information and stores it in the cells of the selected word. The data input and data output of each sense / write circuits are connected to a single bidirectional data line that can be connected to a data bus of the CPU.

Figure 1 is an example of a very small memory circuit consisting of 16 words of 8 bits each. This is referred to as a 16 × 8 organization.

- The data input and the data output of each Sense/Write circuit are connected to a single bidirectional data line that can be connected to the data lines of a computer.
- Two control lines, R/W and CS, are provided.
  - The R/W (Read/Write) input specifies the required operation, and
  - CS (Chip Select) input selects a given chip in a multichip memory system.

Now consider a slightly larger memory circuit with 1K (1024) memory cells.

- This circuit can be organized as a 128 × 8 memory, requiring a total of 19 external connections.
- Alternatively, the same number of cells can be organized into a 1K×1 format. In this case, a 10-bit address is needed, but there is only one data line, resulting in 15 external connections.
- Figure 2 shows such an organization. The required 10-bit address is divided into two groups of 5 bits each to form the row and column addresses for the cell array.
    - A row address selects a row of 32 cells, all of which are accessed in parallel. But, only one of these cells is connected to the external data line, based on the column address.

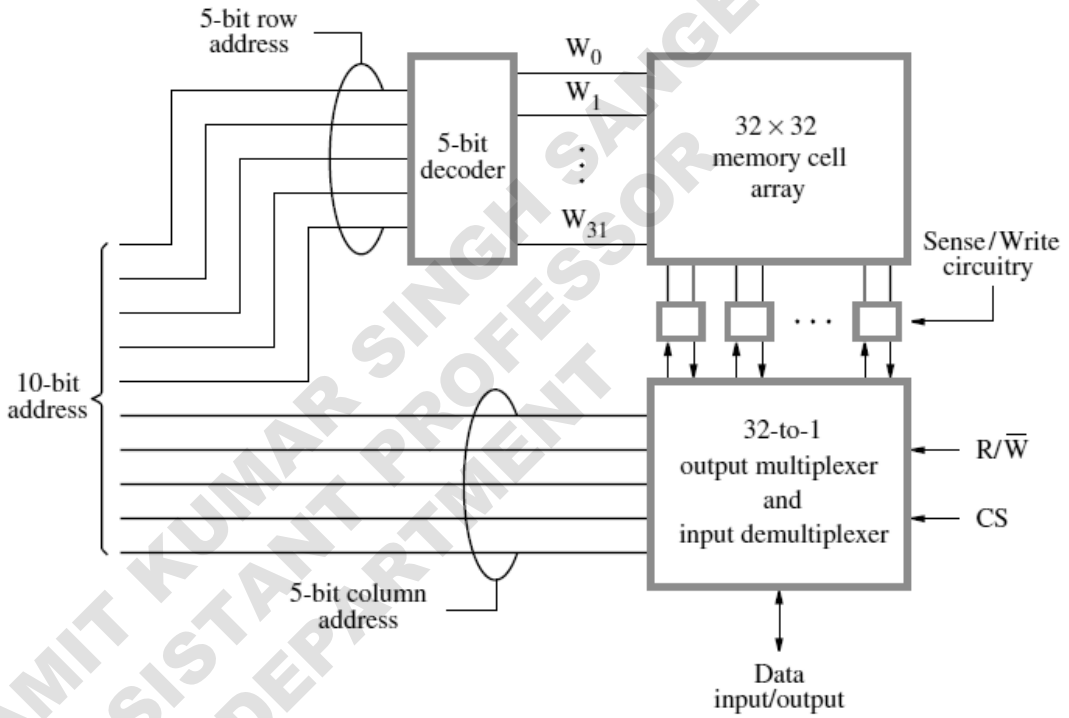


Figure 2 Organization of a 1K × 1 memory chip.

Static RAMs

- Memories that consist of circuits capable of retaining their state as long as power is applied are known as static memories.
- Static random-access memory (SRAM) is a type of semiconductor memory that uses bistable latching circuitry to store each bit. The term *static* differentiates it from *dynamic* RAM (DRAM) which must be periodically refreshed. SRAM exhibits data retention, but is still *volatile* in the conventional sense that data is eventually lost when the memory is not powered.
- Figure 3 shows how a static RAM (SRAM) cell may be implemented.

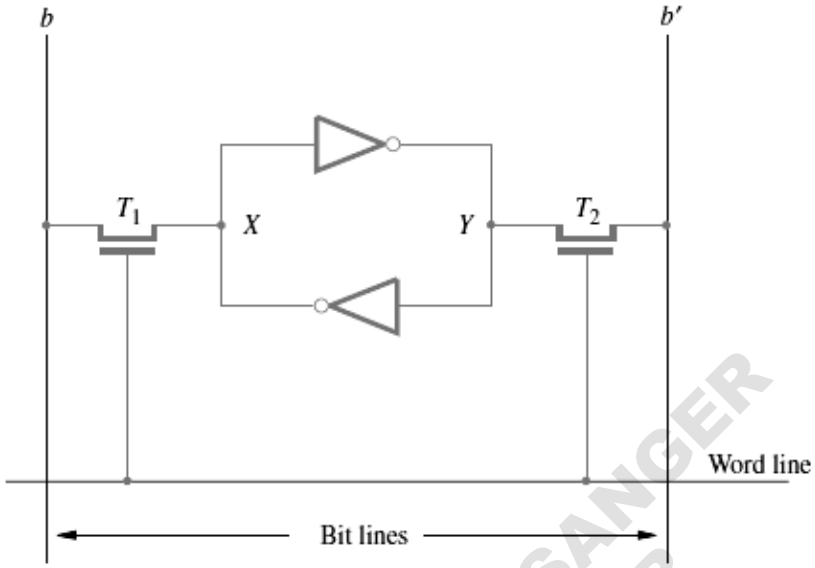


Figure 3 A static RAM cell.

Two inverters are cross-connected to form a latch. The latch is connected to two bit lines by transistors T_1 and T_2. These transistors act as switches that can be opened or closed under control of the word line. When the word line is at ground level, the transistors are turned off and the latch retains its state.

- *Read Operation:* In order to read the state of the SRAM cell, the word line is activated to close switches T_1 and T_2.
  - If the cell is in state 1, the signal on bit line b is high and the signal on bit line b' is low.
  - The opposite is true if the cell is in state 0.
    Thus, b and b' are always complements of each other.

  The Sense/Write circuit at the end of the two bit lines monitors their state and sets the corresponding output accordingly.
- *Write Operation:*
  - The appropriate value on bit line b and its complement on b' are placed and word line is activated. This forces the cell into the corresponding state, which the cell retains when the word line is deactivated.
  - The Sense/Write circuit drives bit lines b and b' to perform write operation, instead of sensing their state.
- *CMOS Cell of Static RAM*

  A CMOS realization of the cell is given in Figure 4. Transistor pairs (T3, T5) and (T4, T6) form the inverters in the latch. The state of the cell is read or written as just explained.

  In state 1, the voltage at point X is maintained high by having transistors T3 and T6 on, while T4 and T5 are off. If T1 and T2 are turned on, bit lines b and b' will have high and low signals, respectively.

  The CMOS requires 5V (in older version) or 3.3.V (in new version) of power supply voltage. The continuous power is needed for the cell to retain its state. If power is interrupted, the cell's contents are lost. When power is restored, the latch settles into a

stable state, but not necessarily the same state the cell was in before the interruption. Hence, SRAMs are said to be volatile memories because their contents are lost when power is interrupted.

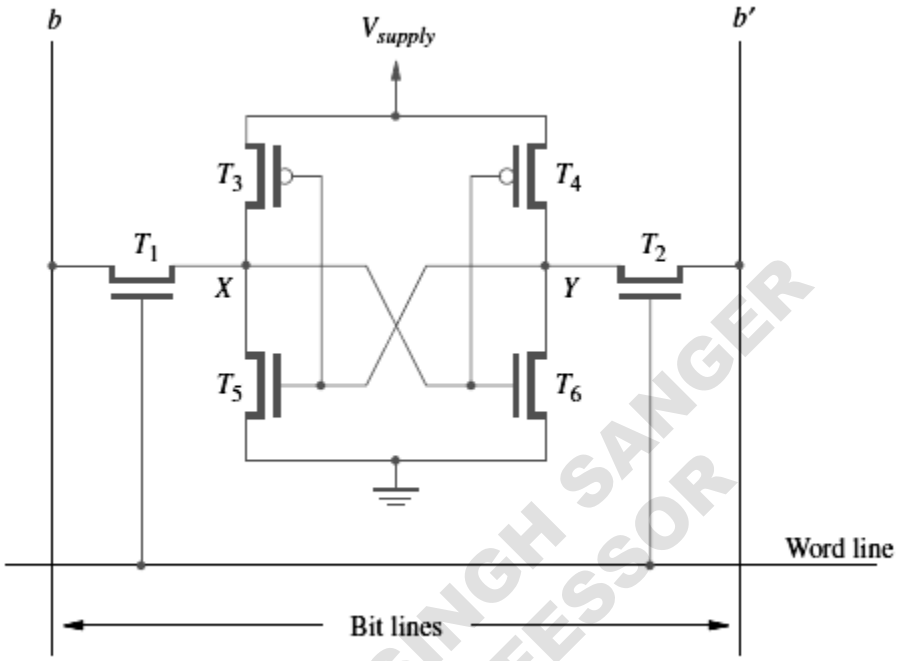


Figure 4 An example of a CMOS memory cell.

A major advantage of CMOS SRAMs is

- Their very low power consumption, because current flows in the cell only when the cell is being accessed.
- Static RAMs can be accessed very quickly. Access times on the order of a few nanoseconds are found in commercially available chips. SRAMs are used in applications where speed is of critical concern.

Dynamic RAMs

- Less expensive and higher density RAMs can be implemented with simpler cells. But, these simpler cells do not retain their state for a long period, unless they are accessed frequently for Read or Write operations. Memories that use such cells are called dynamic RAMs (DRAMs).
- Information is stored in a dynamic memory cell in the form of a charge on a capacitor, but this charge can be maintained for only tens of milliseconds. Since the cell is required to store information for a much longer time, its contents must be periodically *refreshed* by restoring the capacitor charge to its full value.
- This occurs when the contents of the cell are read or when new information is written into it.
- An example of a dynamic memory cell that consists of a capacitor, C, and a transistor, T, is shown in Figure 5.
- To store information in this cell, transistor T is turned on and an appropriate voltage is applied to the bit line. This causes a known amount of charge to be stored in the capacitor.

- After the transistor is turned off, the charge remains stored in the capacitor, but not for long. The capacitor begins to discharge which is caused by the capacitor‘s own leakage resistance.

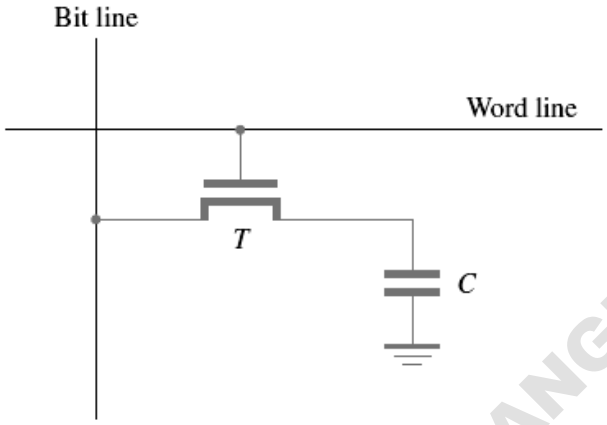


Figure 5 A single-transistor dynamic memory cell.

Hence, the information stored in the cell can be retrieved correctly only if it is read before the charge in the capacitor drops below some threshold value.

- During a Read operation, the transistor in a selected cell is turned on. A sense amplifier connected to the bit line detects whether the charge stored in the capacitor is above or below the threshold value.
    - If the charge is above the threshold, the sense amplifier drives the bit line to the full voltage representing the logic value 1. As a result, the capacitor is recharged to the full charge corresponding to the logic value 1.
    - If the sense amplifier detects that the charge in the capacitor is below the threshold value, it pulls the bit line to ground level to discharge the capacitor fully.

  Thus, reading the contents of a cell automatically refreshes its contents. Since the word line is common to all cells in a row, all cells in a selected row are read and refreshed at the same time.
- For Example; a 256-Megabit DRAM chip, configured as 32M × 8, is shown in Figure 6.
    - The cells are organized in the form of a 16K × 16K array.
    - The 16,384 cells in each row are divided into 2,048 groups of 8, forming 2,048 bytes of data.
    - Therefore,
        - 14 address bits are needed to select a row, and
        - 11 bits are needed to specify a group of 8 bits in the selected row.
        - Total, a 25-bit address is needed to access a byte in this memory.
    - The high-order 14 bits and the low-order 11 bits of the address constitute the row and column addresses of a byte, respectively.
    - To reduce the number of pins needed for external connections, the row and column addresses are multiplexed on 14 pins.

- During a Read or a Write operation, the row address is applied first. It is loaded into the row address latch in response to a signal pulse on an input control line called the Row Address Strobe ($\overline{RAS}$). This causes a Read operation to be initiated, in which all cells in the selected row are read and refreshed.

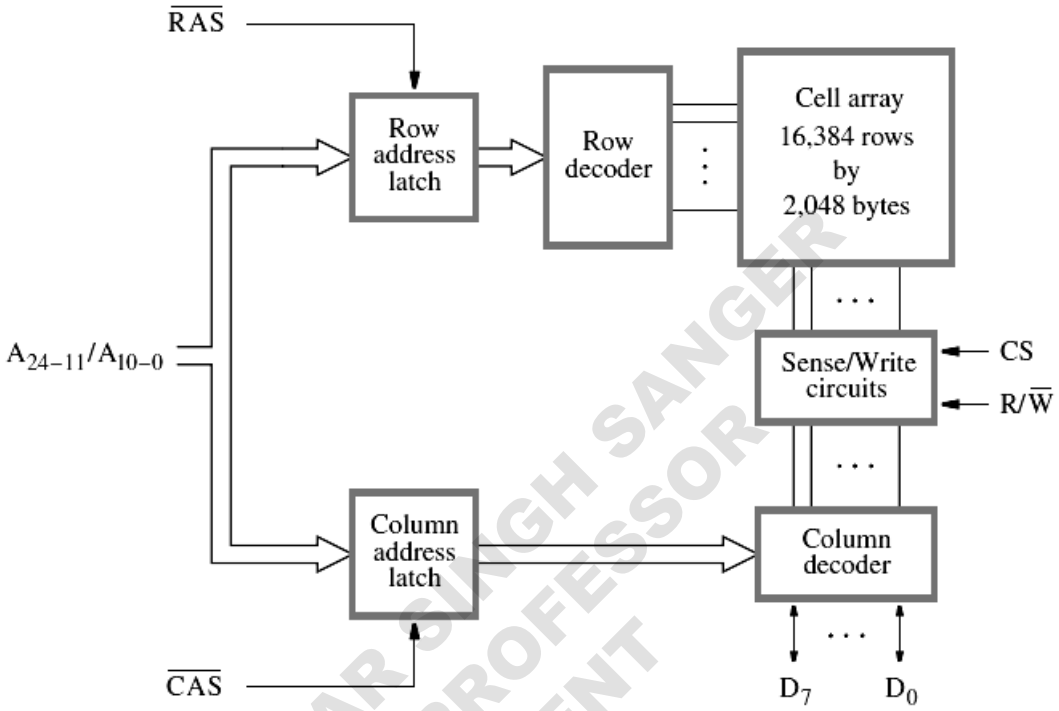


Figure 6 Internal organization of a 32M × 8 dynamic memory chip.

- Shortly after the row address is loaded, the column address is applied to the address pins and loaded into the column address latch under control of a second control line called the Column Address Strobe ($\overline{CAS}$). The information in this latch is decoded and the appropriate group of 8 Sense/Write circuits is selected.
- If the R/W control signal indicates a Read operation, the output values of the selected circuits are transferred to the data lines, D_{7-0}.
- For a Write operation, the information on theD_{7-0} lines is transferred to the selected circuits, and then used to overwrite the contents of the selected cells in the corresponding 8 columns.
- The timing of the operation of the DRAM described above is controlled by the $\overline{RAS}$ and $\overline{CAS}$ signals. These signals are generated by a memory controller circuit external to the chip when the processor issues a Read or a Write command. During a Read operation, the output data are transferred to the processor after a delay equivalent to the memory's access time. Such memories are referred to as *asynchronous DRAMs*. The memory controller is also responsible for refreshing the data stored in the memory chips.

- **Fast Page Mode:**
  A simple addition to the circuit makes it possible to access the other bytes in the same row without having to reselect the row.
  Each sense amplifier also acts as a latch. When a row address is applied, the contents of all cells in the selected row are loaded into the corresponding latches. Then, it is only necessary to apply different column addresses to place the different bytes on the data lines.
  This arrangement leads to a very useful feature. All bytes in the selected row can be transferred in sequential order by applying a consecutive sequence of column addresses under the control of successive $\overline{\text{CAS}}$ signals.

  Thus, a block of data can be transferred at a much faster rate than can be achieved for transfers involving random addresses. The block transfer capability is referred to as the *fast page mode* feature. (A large block of data is often called a page.) The faster rate possible in the fast page mode makes dynamic RAMs particularly well suited to this environment.

Synchronous DRAMs

- In the early 1990s, developments in memory technology resulted in DRAMs whose operation is synchronized with a clock signal. Such memories are known as *synchronous DRAMs (SDRAMs).*
- Their structure is shown in Figure 7.

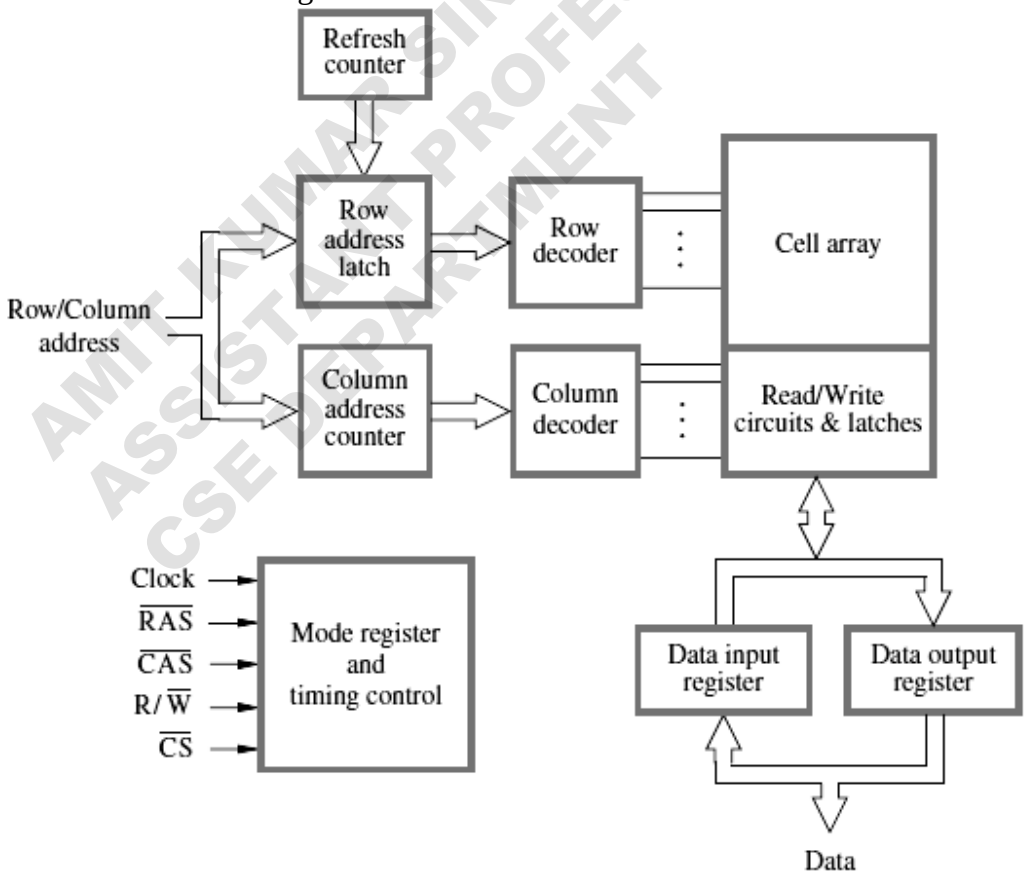


Figure 7 Synchronous DRAM.

- The cell array is the same as in asynchronous DRAMs. The distinguishing feature of an SDRAM is the use of a clock signal. The availability of clock signal makes it possible to integrate control circuitry on the chip that provides many useful features.
- The address and data connections of an SDRAM may be buffered by means of registers.
- Internally, the Sense/Write amplifiers function as latches, as in asynchronous DRAMs.
- A Read operation causes the contents of all cells in the selected row to be loaded into these latches. The data in the latches of the selected column are transferred into the data register, thus becoming available on the data output pins.
- The buffer registers are useful when transferring large blocks of data at very high speed.
- By isolating external connections from the chip's internal circuitry, it becomes possible to start a new access operation while data are being transferred to or from the registers.
- SDRAMs have several different modes of operation, which can be selected by writing control information into a mode register.
  For example, burst operations of different lengths can be specified.
- Figure 8 shows a timing diagram for a typical burst read of length 4.

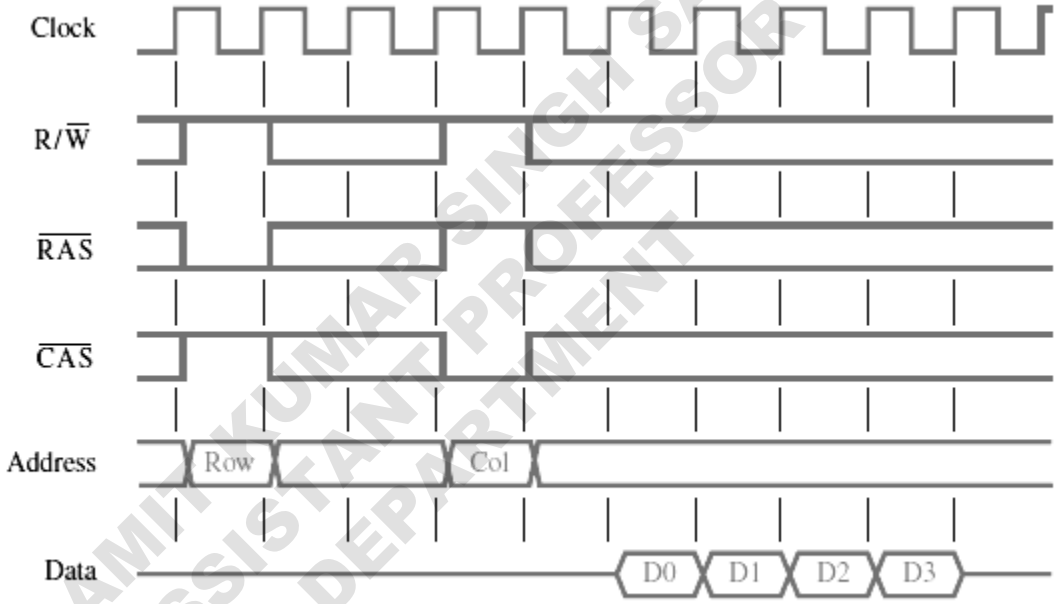


Figure 8 A burst read of length 4 in an SDRAM.

- First, the row address is latched under control of the $\overline{RAS}$ signal. The memory typically takes 2 or 3 clock cycles (we use 2 in the figure for simplicity) to activate the selected row. Then, the column address is latched under control of the $\overline{CAS}$ signal.
- After a delay of one clock cycle, the first set of data bits is placed on the data lines. The SDRAM automatically increments the column address to access the next three sets of bits in the selected row, which are placed on the data lines in the next 3 clock cycles.

- Synchronous DRAM scan deliver data at a very high rate, because all the control signals needed are generated inside the chip.
- The initial commercial SDRAMs in the 1990s were designed for clock speeds of up to 133 MHz. As technology evolved, much faster SDRAM chips were developed. Today's SDRAMs operate with clock speeds that can exceed 1 GHz.

Latency and Bandwidth

Data transfers to and from the main memory often involve blocks of data. The speed of these transfers has a large impact on the performance of a computer system. The memory access time is not sufficient for describing the memory's performance when transferring blocks of data.

- *Latency:* During block transfers, memory *latency* is the amount of time it takes to transfer the first word of a block.
  The time required to transfer a complete block depends also on the rate at which successive words can be transferred and on the size of the block. The time between successive words of a block is much shorter than the time needed to transfer the first word.
- *Bandwidth:* A useful performance measure is the number of bits or bytes that can be transferred in one second. This measure is often referred to as the memory *bandwidth*. It depends on the speed of access to the stored data and on the number of bits that can be accessed in parallel. The rate at which data can be transferred to or from the memory depends on the bandwidth of the system interconnections.

Double-Data-Rate SDRAM

- In the continuous hunt for improved performance, faster versions of SDRAMs have been developed.
- New organizational and operational features make it possible to achieve high data rates during block transfers. The key idea is to take advantage of the fact that a large number of bits are accessed at the same time inside the chip when a row address is applied. Various techniques are used to transfer these bits quickly to the pins of the chip.
  To make the best use of the available clock speed, data are transferred externally on both the rising and falling edges of the clock. For this reason, memories that use this technique are called double-data-rate SDRAMs (DDR SDRAMs).
- Several versions of DDR chips have been developed.
  - The earliest version is known as DDR.
  - Later versions, called DDR2, DDR3, and DDR4, have enhanced capabilities.
  - They offer increased storage capacity, lower power, and faster clock speeds.

Rambus Memory

- The rate of transferring data between the memory and the processor is a function of both the bandwidth of the memory and the bandwidth of its connection to the processor.
- Rambus is a memory technology that achieves a high data transfer rate by providing a high-speed interface between the memory and the processor.
- One way for increasing the bandwidth of this connection is to use a wider data path. However, this requires more space and more pins, increasing system cost.
- The alternative is to use fewer wires with a higher clock speed. This is the approach taken by Rambus.

- The key feature of Rambus technology is the use of a differential-signaling technique to transfer data to and from the memory chips.
  Instead of using signals that have voltage levels of either 0 or V_{Supply} to represent the logical values, the signals consist of much smaller voltage swings around a reference voltage V_{Ref}. The reference Voltage is about 2V and the two logical values are represented by 0.3V swings above and below V_{Ref}. This type of signaling is generally is known as *Differential Signaling.*
- In Rambus technology, signals are transmitted using small voltage swings of 0.1V above and below a reference value. Several versions of this standard have been developed, with clock speeds of up to 800 MHz and data transfer rates of several gigabytes per second.
- Rambus technology competes directly with the DDR SDRAM technology.
- Each has certain advantages and disadvantages. A nontechnical consideration is that the specification of DDR SDRAM is an open standard that can be used free of charge. Rambus, on the other hand, is a proprietary scheme that must be licensed by chip manufacturers.

Structure of Larger Memories

How memory chips may be connected to form a much larger memory.

- **Static Memory Systems**
  - Consider a memory consisting of 2M words of 32 bits each. Figure 9 shows how this memory can be implemented using 512K $\times$ 8 static memory chips.
  - Each column in the figure implements one byte position in a word, with four chips providing 2M bytes.
  - Four columns implement the required 2M $\times$ 32 memory.
  - Each chip has a control input called Chip-select. When this input is set to 1, it enables the chip to accept data from or to place data on its data lines. Only the selected chip places data on the data output line, while all other outputs are electrically disconnected from the data lines.
  - Twenty-one address bits are needed to select a 32-bit word in this memory.
  - The high-order two bits of the address are decoded to determine which of the four rows should be selected.
  - The remaining 19 address bits are used to access specific byte locations inside each chip in the selected row. The R/W inputs of all chips are tied together to provide a common Read/Write control line.

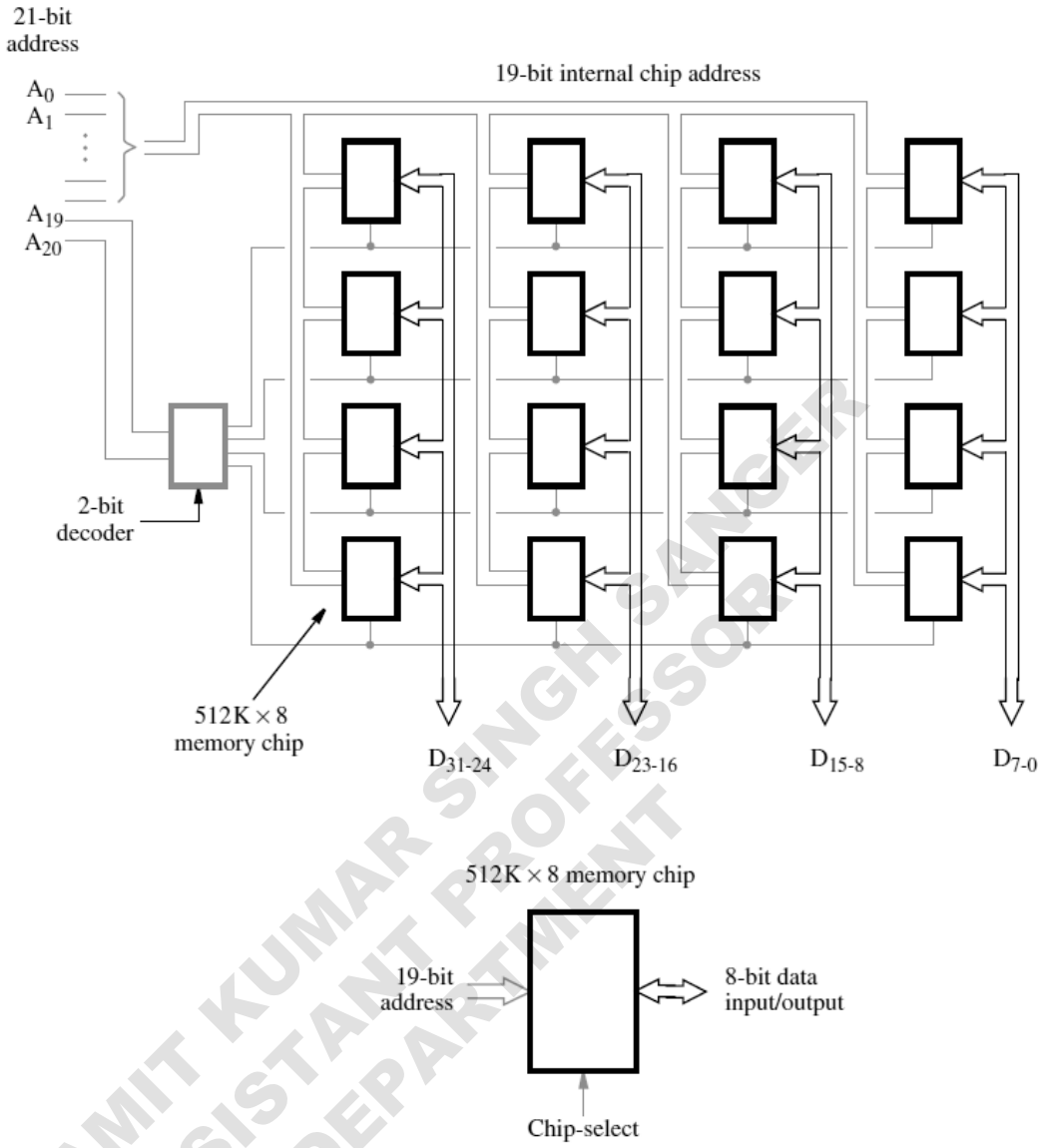


Figure 9 Organization of a 2M × 32 memory module using 512K × 8 static memory chips.

- **Dynamic Memory Systems**
  - A large memory leads to better performance, because more of the programs and data used in processing can be held in the memory, thus reducing the frequency of access to secondary storage.
  - Because of their high bit density and low cost, dynamic RAMs, mostly of the synchronous type, are widely used in the memory units of computers. They are slower than static RAMs, but they use less power and have considerably lower cost per bit. Available chips have capacities as high as 2G bits, and even larger chips are being developed. To reduce the number of memory chips needed in a given computer, a memory chip may be organized to read or write a number of bits in parallel.
  - Chips are manufactured in different organizations, to provide flexibility in designing memory

- systems. For example, a 1-Gbit chip may be organized as 256M × 4, or 128M × 8.
- Packaging considerations have led to the development of assemblies known as memory modules.
- Each such module houses many memory chips, typically in the range 16 to 32, on a small board that plugs into a socket on the computer's motherboard. Memory modules are commonly called *SIMM*s (Single In-line Memory Modules) or *DIMM*s (Dual In-line Memory Modules), depending on the configuration of the pins. Modules of different sizes are designed to use the same socket. For example, 128M × 64, 256M × 64, and 512M × 64 bit DIMMs all use the same 240-pin socket. Thus, total memory capacity is easily expanded by replacing a smaller module with a larger one, using the same socket.

Memory System Consideration

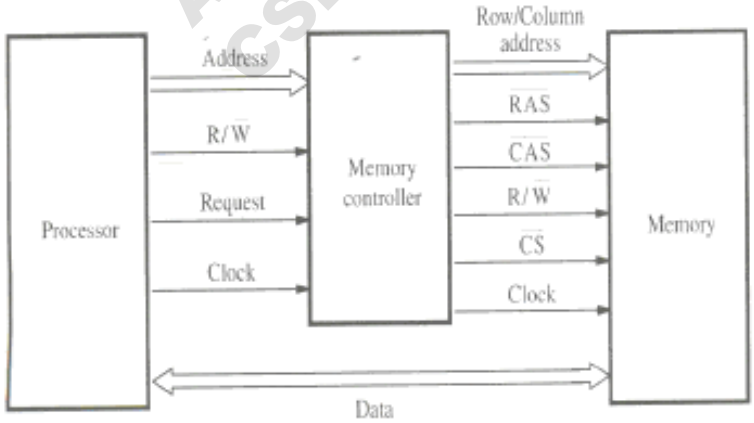
- **Memory Controller**

To reduce the number of pins the multiplexed address lines are used. The address lines are divided into two parts. The high-order address bits, which select a row in the cell array, are provided first and latched into the memory chip under control of the RAS signal. Then, the low-order address bits, which select a column, are provided on the same address pins and latched under control of the CAS signal.

A typical processor issues all bits of an address at the same time, a multiplexer is required. This function is usually performed by a memory controller circuit. The controller accepts a complete address and the R/W signal from the processor, under control of a Request signal which indicates that a memory access operation is needed.

It forwards the R/Wsignals and the row and column portions of the address to the memory and generates the RAS and CAS signals, with the appropriate timing. When a memory includes multiple modules, one of these modules is selected based on the high-order bits of the address. The memory controller decodes these high-order bits and generates the chip-select signal for the appropriate module. Data lines are connected directly between the processor and the memory.

Dynamic RAMs must be refreshed periodically. The circuitry required to initiate refresh cycles is included as part of the internal control circuitry of synchronous DRAMs. However, a control circuit external to the chip is needed to initiate periodic Read cycles to refresh the cells of an asynchronousDRAM.The memory controller provides this capability.



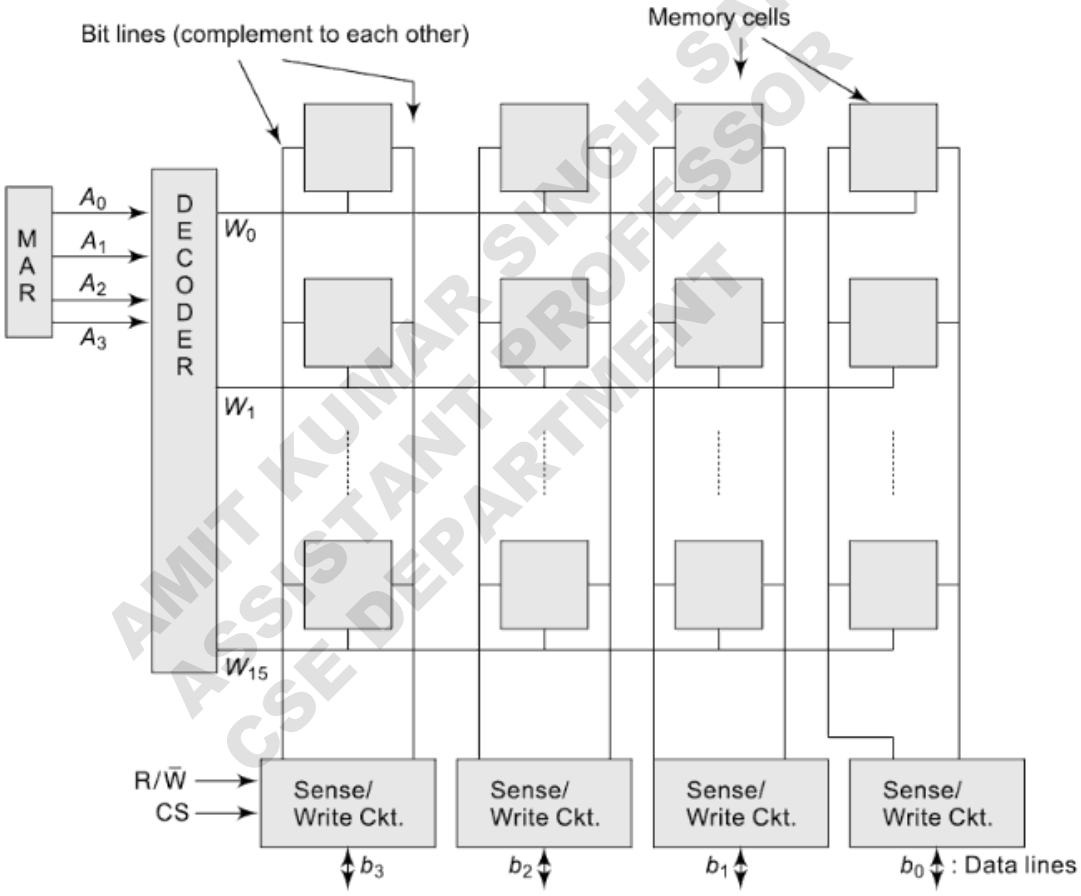
Refresh Overhead

A dynamic RAM cannot respond to read or write requests while an internal refresh operation is taking place. Such requests are delayed until the refresh cycle is completed. However, the time lost to accommodate refresh operations is very small. For example, consider an SDRAM in which each row needs to be refreshed once every 64 ms. Suppose that the minimum time between two row accesses is 50 ns and that refresh operations are arranged such that all rows of the chip are refreshed in 8K (8192) refresh cycles. Thus, it takes $8192 \times 0.050 = 0.41$ ms to refresh all rows. The refresh overhead is $0.41/64 = 0.0064$, which is less than 1 percent of the total time available for accessing the memory.

2D Organization

This is the simplest type of organization. An example of size 16 x 4 memory is shown in figure 1. The cells are organizaed in the form of a two-dimensional array with rows and colums.

Each row refers to a word line. For a 4-bit per word memory, 4 cells are interconnected to a word line. Each column in the array refers to a bit line. The Memory Address Register (MAR) holds the address of the location where read/write operation is executed. For a $w \times b$ memory, MAR has $\log_2 w = n$ bits. Here in this example, $n = 4$. The content of MAR is decoded by an address decoder on the chip to activate each word line. The cells in each column are connected to a sense/write circuit by two bit lines. Two bit lines are complement to each other. The sense/write circuits are activated by the chip select (CS) lines. The sense/write circuits are connected to the data lines of the chip. During a read operation, these circuits sense or read the information stored in the cells selected by a word line and transmit this information to the data lines. During a write operation, the sense/write circuits receive or write input information from the data lines and store it in the selected cells.

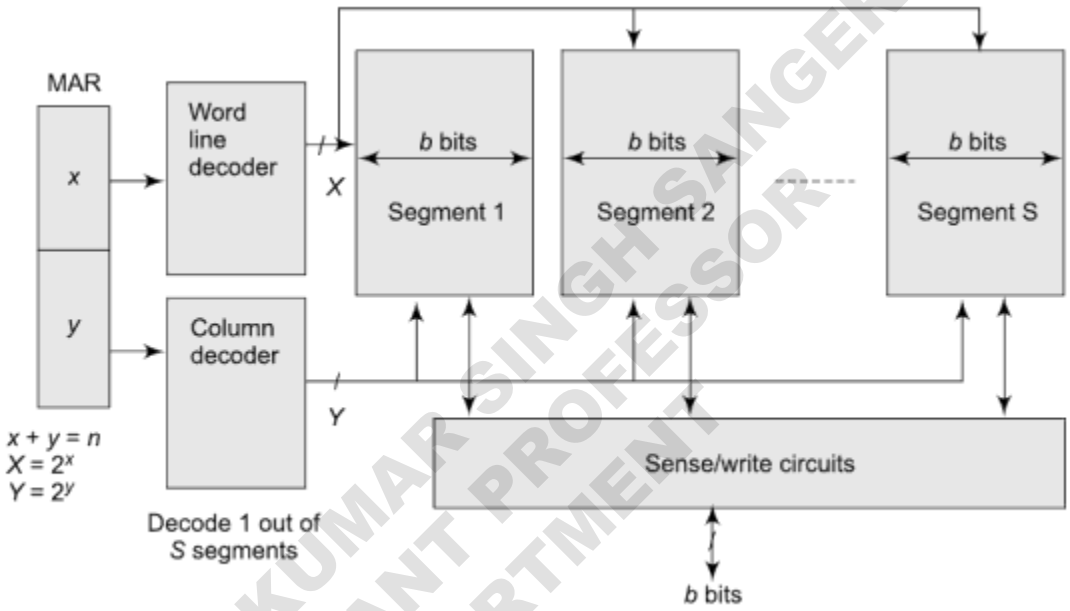


2D organization of a memory chip of size 16 × 4

$2^{1/2}$D Organization To cope with the above difficulties experienceed in 3D organization, the design of $2^{1/2}$D organization has evolved which combines the function of bit lines and Y drives lines.

In 2.5D organization there exists a segment, corresponding to bit plane of 3D organization. The content of MAR is divided into two parts—*x* and *y* number of bits. The number of segments S is equal to 2^y. X = 2^x drive lines are fed into the cell array and *y* number of bits decode one bit line out of S lines fed into a segment of the array. In total, there are Sb number of bit lines for a b bit per word memory.

Thus, for any given address in the MAR, the column decoder decodes *b* out of *Sb* bit lines by using the y bits of the MAR while a particular word line is activated by using the x bits. Thus only the *b* number of bits in the array are accessed by enabling the word line and b number of bit lines simultaneously. A general $2^{1/2}$D memory organization is shown in figure 2.

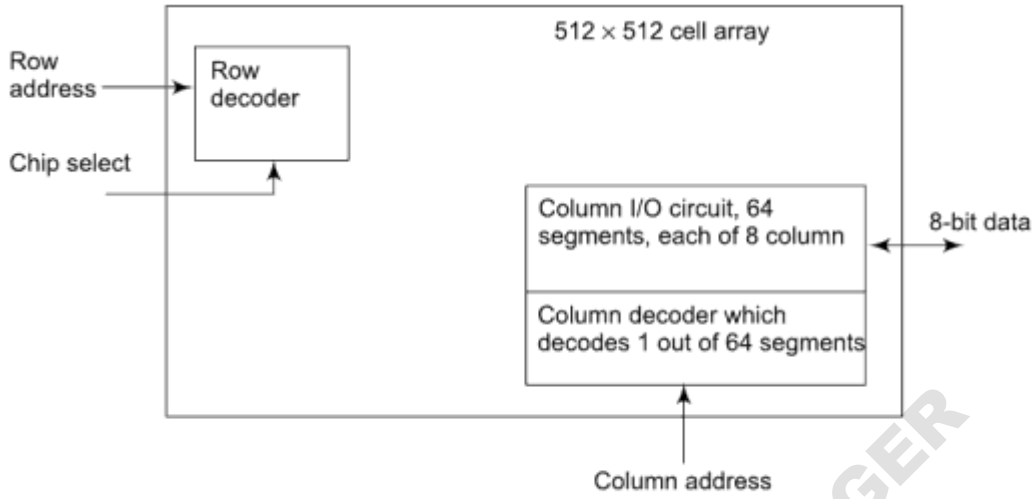


Memory organization

Figure 2

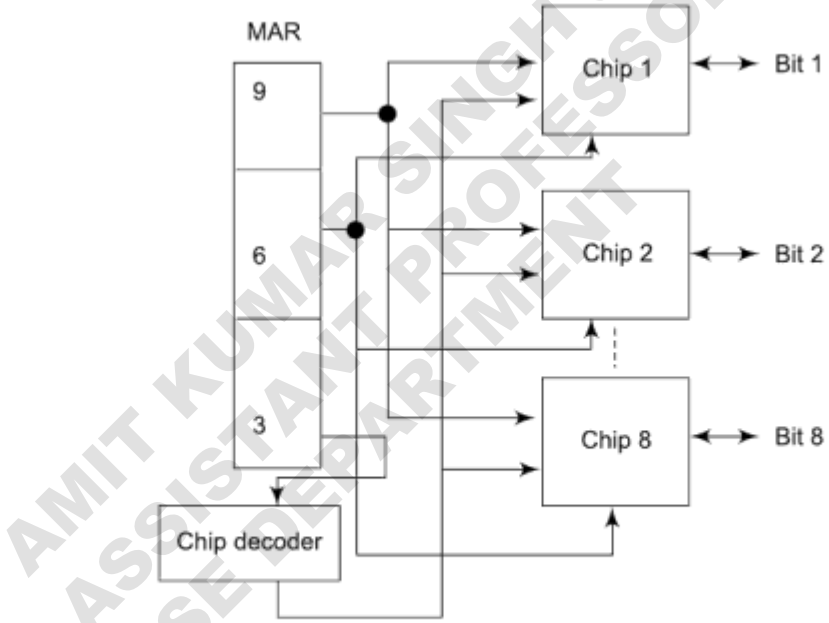
Let us consider an example to realize a 256 x 8 (256K word, 8-bit/word) memory with 512 x 512 cell array in a $2^{1/2}$D organized configuration. The cell array in a chip, as shown in figure 3, can be organized with 512 rows and 64 segments with 8 columns per segment. The chip select is used as decoder enable signal within the chip that realizes 32K x 8-bit memory. Eight such chips can be organized as noted in figure 4, to realize the 256K x 8-bit memory. In this configuration, the MAR is divided into three parts –the 9-bit field is fed as the row address for each chip while the 6-bit field is input as the column address. The remaining three bits of 18-bit MAR are used to select 1 out of 8 chips as noted in figure 4.

Though 2.5D organized memory may need lesser chip decoding logic, it suffers from one drawback. With high density chips, a simple failure, such as external pin connection opening or a failure on one bit can render the entire chip inoperative.



2.5D organized memory chip built with 512 × 512 cell array

Figure 3



(b) Organization of chip array

256K × 8 memory configured out of 2.5D 512 × 512 memory chips

Figure 4

READ-ONLY MEMORIES

Both static and dynamic RAM chips are volatile, which means that they retain information only while power is turned on. There are many applications requiring memory devices that retain the stored information when power is turned off. For example, the need to store a small program in such a memory, to be used to start the bootstrap process of loading the operating system from a hard disk into the main memory.

Many embedded applications require nonvolatile memories to store their software.

Different types of nonvolatile memories have been developed. Generally, their contents can be read in the same way as for their volatile counterparts. But, a special writing process is needed to place the information into a nonvolatile memory. Since its normal operation involves only reading the stored data, a memory of this type is called a *read-only memory* (ROM).

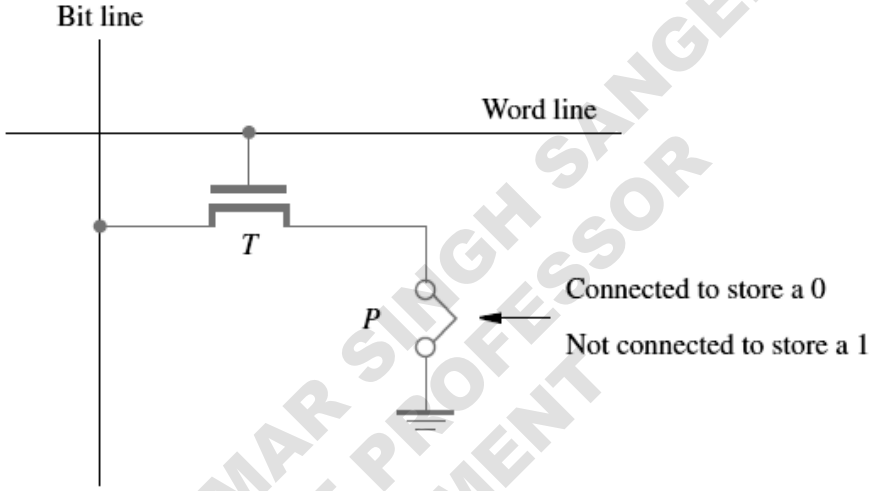


Figure 1 A ROM cell.

ROM

- A memory is called a read-only memory, or ROM, when information can be written into it only once at the time of manufacture.
- Figure 1 shows a possible configuration for a ROM cell. A logic value 0 is stored in the cell if the transistor is connected to ground at point *P*; otherwise, a 1 is stored. The bit line is connected through a resistor to the power supply.
- To read the state of the cell, the word line is activated to close the transistor switch. As a result, the voltage on the bit line drops to near zero if there is a connection between the transistor and ground. If there is no connection to ground, the bit line remains at the high voltage level, indicating a 1. A sense circuit at the end of the bit line generates the proper output value. The state of the connection to ground in each cell is determined when the chip is manufactured, using a mask with a pattern that represents the information to be stored.

PROM

- Some ROM designs allow the data to be loaded by the user, thus providing a *programmable ROM* (PROM).
- Programmability is achieved by inserting a fuse at point *P* in Figure 1.

- Before it is programmed, the memory contains all 0s. The user can insert 1s at the required locations by burning out the fuses at these locations using high-current pulses. Of course, this process is irreversible.
- PROMs provide flexibility and convenience not available with ROMs. The cost of preparing the masks needed for storing a particular information pattern makes ROMs cost effective only in large volumes.
- The alternative technology of PROMs provides a more convenient and considerably less expensive approach, because memory chips can be programmed directly by the user.

EPROM

- Another type of ROM chip provides an even higher level of convenience. It allows the stored data to be erased and new data to be written into it. Such an *erasable*, reprogrammable ROM is usually called an *EPROM*.
- It provides considerable flexibility during the development phase of digital systems. Since EPROMs are capable of retaining stored information for a long time, they can be used in place of ROMs or PROMs while software is being developed. In this way, memory changes and updates can be easily made.
- An EPROM cell has a structure similar to the ROM cell in Figure 1. However, the connection to ground at point *P* is made through a special transistor. The transistor is normally turned off, creating an open switch. It can be turned on by injecting charge into it that becomes trapped inside. Thus, an EPROM cell can be used to construct a memory in the same way as the previously discussed ROM cell.
- Erasure requires dissipating the charge trapped in the transistors that form the memory cells. This can be done by exposing the chip to ultraviolet light, which erases the entire contents of the chip. To make this possible, EPROM chips are mounted in packages that have transparent windows.

EEPROM

- An EPROM must be physically removed from the circuit for reprogramming. Also, the stored information cannot be erased selectively. The entire contents of the chip are erased when exposed to ultraviolet light.
- Another type of erasable PROM can be programmed, erased, and reprogrammed electrically. Such a chip is called an *electrically erasable* PROM, or EEPROM.
- It does not have to be removed for erasure. Moreover, it is possible to erase the cell contents selectively.
- One disadvantage of EEPROMs is that different voltages are needed for erasing, writing, and reading the stored data, which increases circuit complexity. However, this disadvantage is outweighed by the many advantages of EEPROMs. They have replaced EPROMs in practice.

Flash Memory

- An approach similar to EEPROM technology has given rise to *flash memory* devices.
- A flash cell is based on a single transistor controlled by trapped charge, much like an EEPROM cell. Also like an EEPROM, it is possible to read the contents of a single cell. The key difference is that, in a flash device, it is only possible to write an entire block of cells. Prior to writing, the previous contents of the block are erased.
- Flash devices have greater density, which leads to higher capacity and a lower cost per bit.

- They require a single power supply voltage, and consume less power in their operation. The low power consumption of flash memories makes them attractive for use in portable, battery-powered equipment.
- Typical applications include hand-held computers, cell phones, digital cameras, and MP3 music players.
- Larger memory modules consisting of a number of chips are used where needed. There are two popular choices for the implementation of such modules: flash cards and flash drives.
    - **Flash Cards**
        - One way of constructing a larger module is to mount flash chips on a small card. Such flash cards have a standard interface that makes them usable in a variety of products.
        - A card is simply plugged into a conveniently accessible slot. Flash cards with a USB interface are widely used and are commonly known as memory keys. They come in a variety of memory sizes. Larger cards may hold as much as 32 Gbytes.
    - **Flash Drives**
        - Larger flash memory modules have been developed to replace hard disk drives, and hence are called flash drives.
        - They are designed to fully emulate hard disks, to the point that they can be fitted into standard disk drive bays. Though, the storage capacity of flash drives is significantly lower.In contrast, hard disks have capacities exceeding a terabyte. Also, disk drives have a very low cost per bit.
        - The fact that flash drives are solid state electronic devices with no moving parts provides important advantages over disk drives. They have shorter access times, which result in a faster response. They are insensitive to vibration and they have lower power consumption, which makes them attractive for portable, battery-driven applications.

CACHE MEMORY

Cache Memory Concept

- A special very-high speed memory is sometimes used to increase the speed of processing by making active portions of the program and data available to the CPU at a rapid rate. This reduces the average memory access time, 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 shown in figure 1.

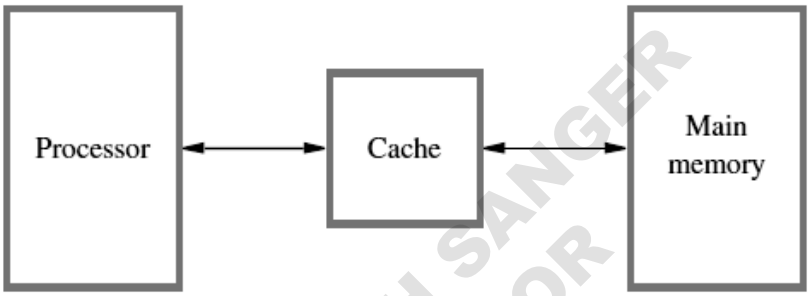


Figure 1

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. Though the cache is only a small fraction of the size of main memory, a large fraction of memory requests will be found in the fast cache memory because of the locality of reference property of programs.
- 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. In this manner, some data are transferred to cache so that future references to memory find the required words in the fast cache memory.

Locality of Reference

- Analysis of a large number of typical programs has shown that the references to memory at any given interval of time tend to be limited within a few localized areas in memory. This phenomenon is known as the property of *locality of reference*.
- The reason for this property may be understood considering that a typical computer program flows in a straight-line fashion with program loops and subroutine calls encountered frequently.
    - When a program loop is executed, the CPU repeatedly refers to the set of instructions in memory that constitute the loop.

- Every time a given subroutine is called, its set of instructions is fetched from memory.

Thus loops and subroutines tend to localize the references to memory for fetching instructions.

- The result of all these observations is the locality of reference property, which states that over a short interval of time, the addresses generated by a typical program refer to a few localized areas of memory repeatedly, while the remainder of memory is accessed relatively infrequently.

  There are three dimensions of the locality property:

  (i) Temporal locality

  (ii) Spatial locality

  (i) Temporal Locality- Recently referenced items (instruction and data) are likely to be referenced in the near future. This is often caused by special program constructs such as iterative loops, process stacks, temporary variables, or subroutines. Once a loop is entered or a subroutine is called a small code segment will be referenced repeatedly many times. Thus temporal locality tends to cluster the access in the recently used areas.

  (ii) Spatial Locality- This refers to the tendency for a process to access items whose addresses are near one another.

  For example, operations on tables or arrays involve accesses of a certain clustered area in the address space. Program segments, such as routines and macros, tend to be stored in the same neighborhood of the memory space.

Performance Consideration of Cache Memory

Hit Rate and Miss Penalty

- The performance of cache memory is 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.

$$\text{Hit Ratio (h)} = \frac{\text{Number of hits}}{\text{Total CPU References to CPU}}$$

$$= \frac{\text{Number of hits}}{\text{Number of hits} + \text{Number of misses}}$$

The hit ration is the probability of getting hits out of some number of memory references made by CPU. So its range is $0 \leq h \leq 1$.

- Consider a system with only one level of cache. In this case, the miss penalty consists almost entirely of the time to access a block of data in the main memory.

  Let *h* be the hit ratio, *M* the miss penalty, and *C* the time to access information in the cache. Thus, the average access time experienced by the processor is

$$t_{avg} = hC + (1 - h)M$$

- In high-performance processors, two levels of caches are normally used, separate L1 caches for instructions and data and a larger L2 cache. These caches are often implemented on the processor chip. In this case, the L1 caches must be very fast, as they determine the memory access time seen by the processor. The L2 cache can be

slower, but it should be much larger than the L1 caches to ensure a high hit rate. Its speed is less critical because it only affects the miss penalty of the L1 caches. A typical computer may have L1 caches with capacities of tens of kilobytes and an L2 cache of hundreds of kilobytes or possibly several megabytes.

Including an L2 cache further reduces the impact of the main memory speed on the performance of a computer. Its effect can be assessed by observing that the average access time of the L2 cache is the miss penalty of either of the L1 caches. For simplicity, we will assume that the hit rates are the same for instructions and data. Thus, the average access time experienced by the processor in such a system is:

$$t_{avg} = h_1C_1 + (1 - h_1)(h_2C_2 + (1 - h_2)M)$$

Where

h_1 is the hit rate in the L1 caches.

h_2 is the hit rate in the L2 cache.

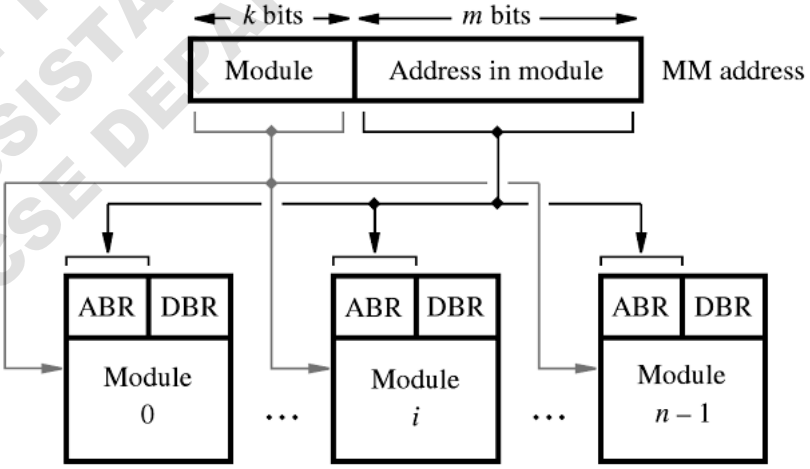
C_1 is the time to access information in the L1 caches.

C_2 is the miss penalty to transfer information from the L2 cache to an L1 cache.

M is the miss penalty to transfer information from the main memory to the L2 cache.

Interleaving

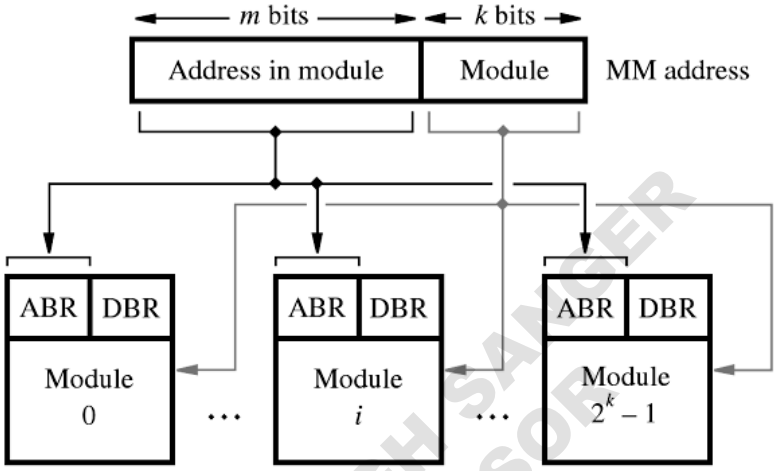
- If the main memory of a computer is structured as a collection of physically separate module memory access operations may be proceed in more than one module at the same time. The aggregate rate of transmission of words to and from the main memory system can be increased.
- Two methods of address layout are indicated in figure. In the first case, memory address generated by the processor is decoded as shown in (a) of the figure. The high-order k bits name one of n modules and the low-order m bits name a particular word in that module. When consecutive locations are accessed, only one module is involved. At the same time, devices with DMA ability may be accessing information in other modules.



(a) Consecutive words in a module

- In the second case, as shown in (b) of the figure, which is called memory interleaving. The low-order k bits of the memory address select a module, and the high order m

bits name a location within the module. Thus, any component of the system that generates requests for access to consecutive memory locations can keep several modules busy at any one time which results in both faster access to a block of data and higher average utilization of the memory system as a whole. To implement the interleaved structure there must be 2^k modules; otherwise there will be gaps of nonexistent locations in the memory address space.



(b) Consecutive words in consecutive modules

Other Enhancements

In addition to the main design issues just discussed, several other possibilities exist for enhancing performance. Three of them are:

(i) Write Buffer
(ii) Prefetching
(iii) Lockup-Free

(i) Write Buffer

- When the write-through protocol is used, each write operation results in writing a new value into the main memory. If the CPU must wait for the memory function to be completed, then the CPU is slowed down by all write requests. Yet the CPU typically does not immediately depend on the result of a write operation, so it is not necessary for the CPU to wait for the write request to be completed.
- To improve performance, a write buffer can be included for temporary storage of write requests. The CPU places each write request into this buffer and continues execution of the next instruction. The write requests stored in the write buffer are sent to the main memory whenever the memory is not responding to read requests.

Note that it is important that the read requests be serviced immediately, because the CPU usually cannot proceed without the data that is to be read from the memory. Hence, these requests are given priority over write requests.

- The write buffer may hold a number of write requests. Thus, it is possible that a subsequent read request may refer to data that are still in the write buffer. To ensure correct operation, the addresses of data to be read from the memory are compared with the addresses of the data in the write buffer. In case of a match, the data in the write buffer are used.

- But the cost is justified by improved performance. A different situation occurs with the write-back protocol. In this case, the write operations are simply performed on the corresponding word in the cache. But consider what happens when a new block of data is to be brought into the cache as a result of a read miss, which replaces an existing block that has some dirty data. The dirty block has to be written into the main memory. If the required write-back is performed first, then the CPU will have to wait longer for the new block to be read into the cache. It is more practical to read the new block first. This can be arranged by providing a fast write buffer for temporary storage of the dirty block that is ejected from the cache while the new block is being read. Afterward, the contents of the buffer are written into the main memory. Thus, the write buffer also works well for the write-back protocol.

(ii) Prefetching

- In the cache mechanism, we assumed that new data are brought into the cache when they are first needed. A read miss occurs, and the desired data are loaded from the main memory. The CPU has to pause until the new data arrive, which is the effect of the miss penalty.
- To avoid stalling the CPU, it is possible to prefetch the data into the cache before they are needed. The simplest way to do this is through software. A special prefetch instruction may be provided in the instruction set of the processor. Executing this instruction causes the addressed data to be loaded into the cache, as in the case of a read miss. However, the processor does not wait for the referenced data. A prefetch instruction is inserted in a program to cause the data to be loaded in the cache by the time they are needed in the program.
- The hope is that prefetching will take place while the CPU is busy executing instructions that do not result in a read miss, thus allowing accesses to the main memory to be overlapped with computation in the CPU.
- Prefetch instructions can be inserted into a program either by the programmer or by the compiler. It is obviously preferable to have the compiler insert these instructions, which can be done with good success for many applications. Note that software prefetching imposes a certain overhead; because inclusion of prefetch instructions increases the length of programs. Moreover, some prefetches may load into the cache data that will not be used by the instruction. This can happen if the prefetched data are ejected from the cache by a read miss involving other data. However, the overall effect of software prefetching on performance is positive, and many processors have machine instructions to support this feature.
- Prefetching can also be done through hardware. This involves adding circuitry that attempts to discover a pattern in memory references, and then prefetches data according to this pattern.

(iii) Lockup-Free

- In cache the software prefetehing scheme does not work well if it interferes significantly with the normal execution of instructions. This is the case if the action of prefetehing stops other accesses to the cache until the prefetch is completed. A cache of this type is said to be locked while it services a miss. We can solve this problem by modifying the basic cache structure to allow the CPU to access the cache while a miss is being serviced. In fact, it is desirable that more than one outstanding miss can be supported. A cache that can support multiple outstanding misses is called lockup-free.

Since it can service only one miss at a time, it must include circuitry that keeps track of all outstanding misses. This may be done with special registers that hold the pertinent information about these misses.

- Lockup-free caches were first used in the early l980s in the Cyber series of computers manufactured by Control Data Company. We have used software prefetching as an obvious motivation for a cache that is not locked by a read miss. A much more important reason is that, in a processor that uses a pipelined organization, which overlaps the execution of several instructions, a read miss caused by one instruction could stall the execution of other instructions. A lockup-free cache reduces the likelihood of such stalling.

CACHE MAPPING

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

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

1. Associative Mapping

- The fastest and most flexible cache organization uses an associative memory. This organization is illustrated in figure 2.
- 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. 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.

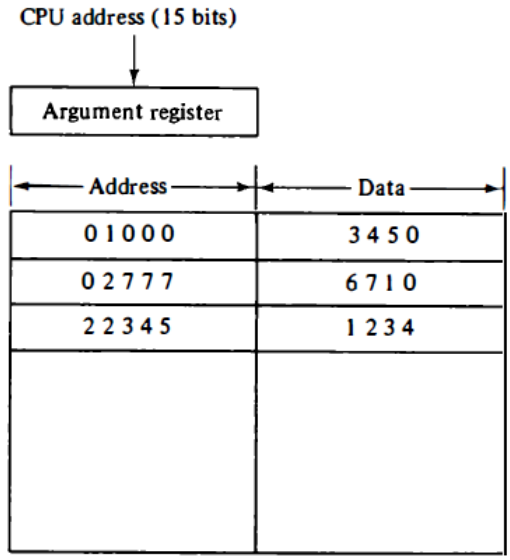


Figure 2: Associative mapping cache (all numbers in octal).

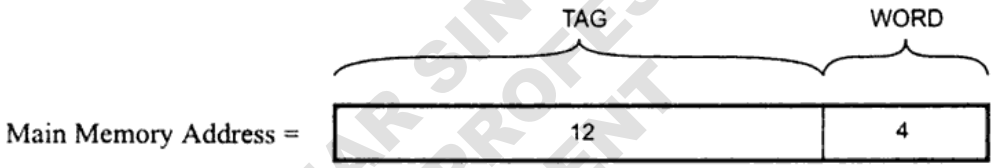
Merits of Associative Mapping: This memory is easy to implement and it is also very fast.
Demerits of Associative Mapping: This memory is expensive as compared to random access memories because of additional storage of addresses with data in cache memory.
Example: Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048 words, and assume that the main memory is addressable by a 16 bit address and it consists of 4k blocks. How many bits are there in each of the TAG, BLOCK, and WORD fields for associative mapping technique.
Solution:

Word Bits : The word length will remain same i.e. 4 bits.

In the associative mapping technique each block in the main memory is identified by the tag bits and an address received from the CPU is compared with the tag bits of each block of the cache to see if the desired block is present. Therefore, this type of technique does not have block bits, but all remaining bits (except word bits) are reserved as tag bits.

Tag Bits : To address each block in the main memory (2^{12} = 4096) 12 bits are required and therefore, there are 12 tag bits.

The main memory address for direct mapping technique is divided as shown below :



2. Direct mapping

In the 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 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.

The internal organization of the words in the cache memory is as shown in figure 3. Each word in cache consists of the data word and its associated tag.

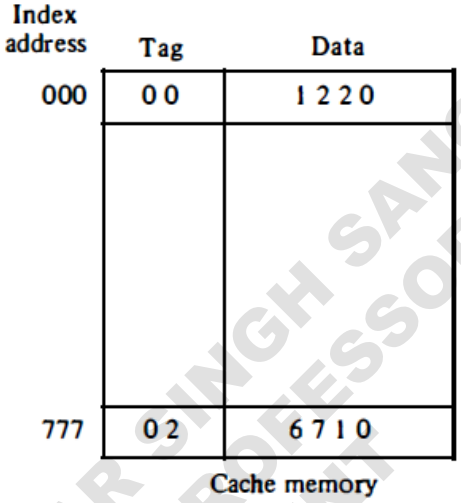


Figure 3

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. 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. However, this possibility is minimized by the fact that such words are relatively far apart in the address range.

For Example; consider the numerical example shown in figure 4.

- 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.

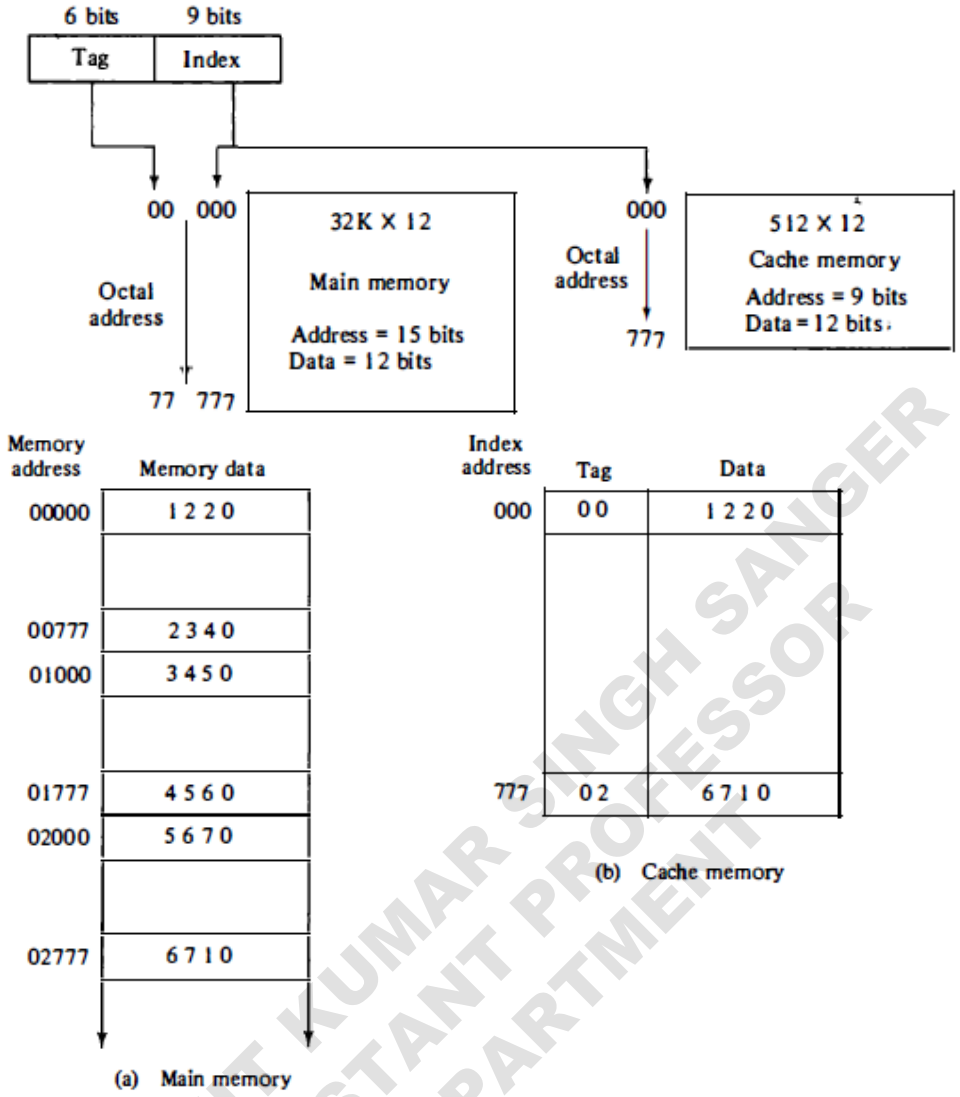


Figure 4: Direct mapping using block size one.

The cache is divided into cache *blocks*, also called cache *lines*. A block contains a set of contiguous address words of same size. We know that data is transferred from main memory to cache memory block.

When the block size in direct mapping cache organization is greater than one word the index field is divided into two parts:

(i) The *Block* field (ii) The *Word* field.

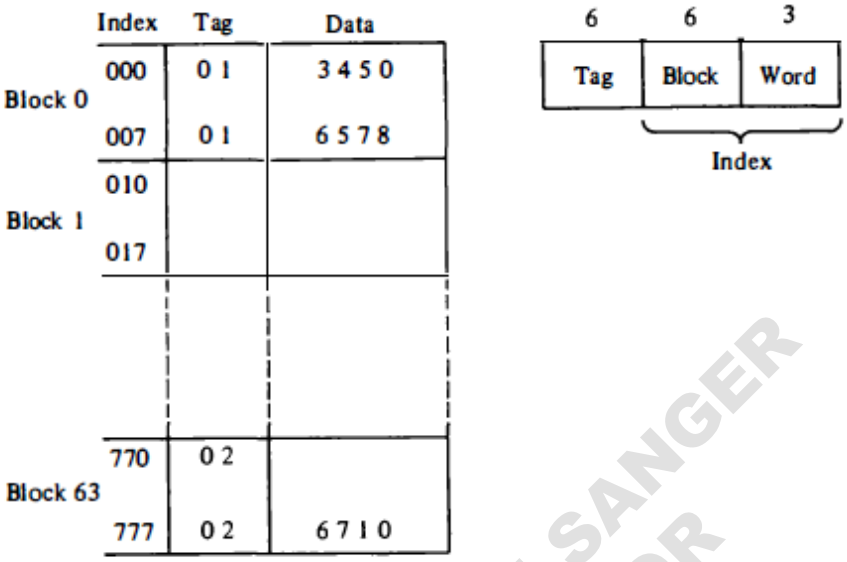
The *Tag* field for all stored words within a block is same. When a miss occurs in the cache, an entire block must be brought into cache memory from main memory.

For Example;

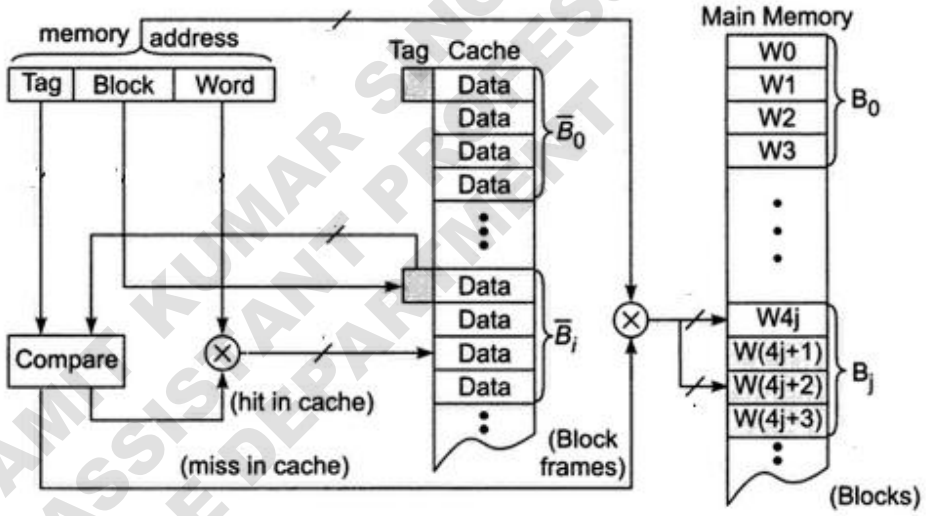
The direct-mapping example using a block size of 8 words.

- 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. 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.



Direct mapping cache with block size of 8 words.



Merits of Direct Mapping

(i) This is simplest type of cache mapping.

(ii) It is less expensive cache relative to the associative cache.

Demerits of Direct Mapping

(i) Hit ratio is not good.

(ii) It needs frequent replacement for data tag value.

Example: Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048 words, and assume that the main memory is addressable by a 16 bit address and it consists of 4k blocks. How many bits are there in each of the TAG, BLOCK, and WORD fields for direct mapping technique.

Solution:

Word Bits : We know that each block consists of 16 words. Therefore, to identify each word we must have (2^4 = 16) four bit reserved for it.

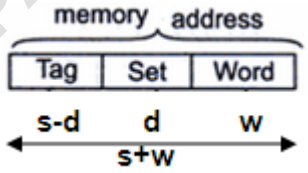
Block Bits : The cache memory consists of 128 blocks and using direct mapped technique block k of the main memory maps onto block k modulo 128 of the cache. It has one to one correspondence and requires unique address for each block. To address 128 block we require (2^7 =128) seven bits.

Tag bits : The remaining 5 (16 – 4 – 7) address bits are tag bits which stores the higher address of the main memory.

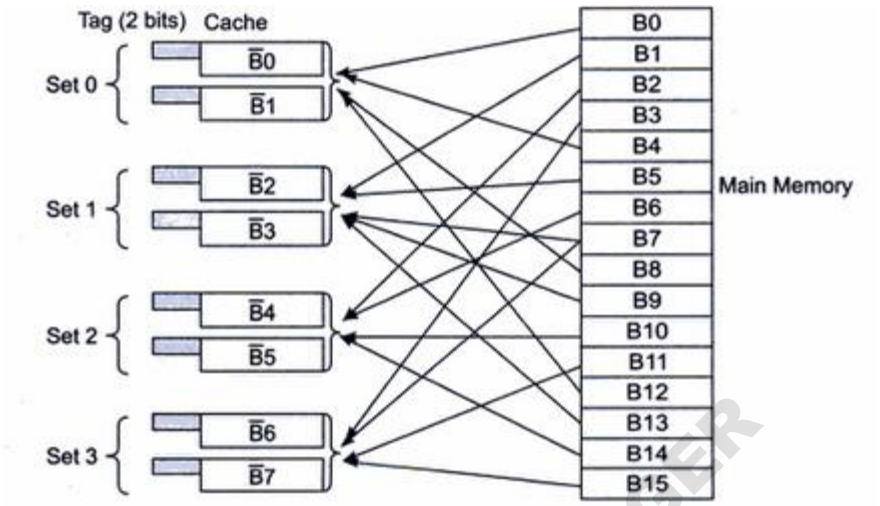
The main memory address for direct mapping technique is divided as shown below :

Set-Associative Mapping

- 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 words under an index of cache is said to form a *set*.
- In general, a set-associative cache of set size k will accommodate k words of main memory in each word of cache and the cache memory is called k-way set associative.
  In other word, in a *k-way* set associative cache, the m cache blocks are divided into $v = m/k$ sets, with *k* blocks per set. Each set is identified by a d bit set number, where $2^d = v$.

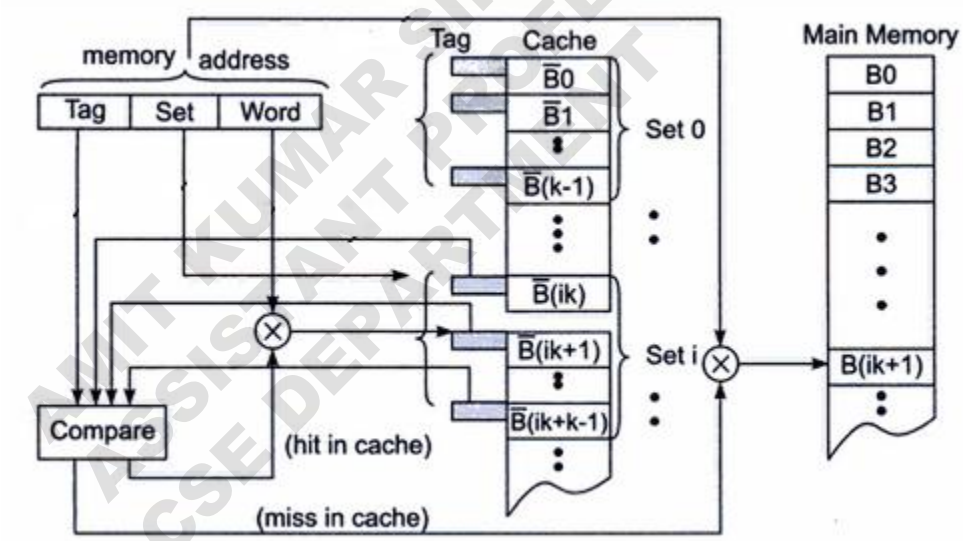


In practice, the set size *k* or associatively is chosen as 2, 4, 8,16 or 64 depending on a tradeoff among block size, cache size and other performance/cost factors.
Fully associative cache mapping can be visualized as having a single set or *m-way* associativity.



Set Associative organization and example of 2-way associative mapping

- 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 tags in the cache to determine if a match occurs. The comparison logic is done by an associative search of the tags in the set similar to an associative memory search: 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. However, an increase in the set size increases the number of bits in words of cache and requires more complex comparison logic.
- 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.

Merit of Set-Associative cache

This cache memory has highest hit ratio compared to other two cache memories.

Demerits of Set-Associative cache

This is the most expensive memory. The cost increases as set size increases.

Example: Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048 words, and assume that the main memory is addressable by a 16 bit address and it consists of 4k blocks. How many bits are there in each of the TAG, BLOCK, and WORD fields for 2-way set associative mapping technique.

Solution

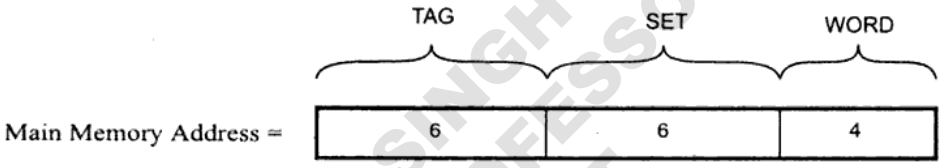
Let us assume that there is a 2-way set associative mapping. Here, cache memory is mapped with two blocks per set. The set field of the address determines which set of the cache might contain the desired block.

Word Bits : The word length will remain same i.e. 4 bits

Set bits : There are 64 sets (128/2). To identify each set (2^6 = 64) six bits are required.

Tag bits : The remaining 6 (16 – 4 – 6) address bits are the tag bits which stores higher address of the main memory.

The main memory address for 2-way set associative mapping technique is divided as shown below :



Replacement Algorithm

In case a miss occurs in cache memory, then a new data from main memory needs to be placed over old data in the selected location of cache memory. In case of direct mapping cache, we have no choice and thus no replacement algorithm is required. The new data has to be stored only in a specified cache location as per the mapping rule for direct mapping cache.

For associative and set associative mapping we need a replacement algorithm since we have multiple choices for locations.

The most common replacement algorithms used are: random replacement, first-in, first-out (FIFO), and least recently used (LRU).

First-in First-out (FIFO) algorithm: This algorithm chooses the word that has been in the cache for a long time. In other words, the words which entered the cache first, gets pushed out first.

Least recently used (LRU) algorithm: This algorithm chooses the word for replacement that has gone the longest time without being referenced. This block is called the *least recently used* (LRU) block, and the technique is called the LRU replacement algorithm.

Cache Writing Methods

An important aspect of cache organization is concerned with memory write requests. When the CPU finds a word in cache during a read operation, the main memory is not involved in the transfer. However, if the operation is a write, there are two ways that the system can proceed.

(i) ***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. This characteristic is important in systems with direct memory access transfers. It ensures that the data residing in main memory are valid at all times.

(ii) ***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 bit called dirty/modified bit so that later when the word is removed from the cache it is copied into main memory.

The reason for the write-back method is that during the time a word resides in the cache, it may be updated several times; however, as long as the word remains in the cache, it does not matter whether the copy in main memory is out of date, since requests from the word are filled from the cache. It is only when the word is displaced from the cache that an accurate copy need be rewritten into main memory. Analytical results indicate that the number of memory writes in a typical program ranges between 10 and 30 percent of the total references to memory.

Cache Types

Caches are distinguished by the kind of information they store.

Instruction cache vs Data cache: Instruction or I- cache stores instructions only, while a data or D-cache stores data only. Separating the stored data in this way recognizes the different access behavior patterns of instructions and data.

For example, program tends to involve few write accesses and they often exhibit more temporal and spatial locality than the data process.

Unified cache vs Split cache:

- A cache that stores both instructions and data is reffered to an unified cache. A split cache, consists of two associated but largely independent units: an I-cache for instructions and a D-cache for data.
- A unified cache is simpler; a split cache makes it possible to access programs and data concurrently.

Caches are also classified by the level they occupy in the memory hierarchy. A level 1 (L1) or primary cache is an efficient way to implement on-chip memory. An additional memory level can be introduced via an off-chip, level 2 (L2) pr secondary cache. The L1 cache is faster and L2 cache increases with the size of main memory, assuming that the size of on chip memory is fixed.

VIRTUAL MEMORY: CONCEPT IMPLEMENTATION

INTRODUCTION

In a memory hierarchy system, programs and data are first stored in auxiliary memory. Portions of a program or data are brought into main memory as they are needed by the CPU.

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.

This is done dynamically, while programs are being executed in the CPU. The translation or mapping is handled automatically by the hardware by means of a mapping table.

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. 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.

An address in main memory is called a location or physical address. The set of such locations is called the memory space.

VIRTUAL MEMORY CONCEPT

- Techniques that automatically move program and data blocks into the physical main memory when they are required for execution is called the Virtual Memory Techniques. The binary address that the processor issues either for instruction or data is called the *virtual / logical address.*
- The virtual address is translated into *physical address by a combination of hardware and software components.* This kind of address translation is done by *MMU (Memory Management Unit)*.
- When the desired data are in the main memory, these data are fetched / accessed immediately. If the data are not in the main memory, The MMU causes the operating system to bring the data into memory from the disk. Transfer of data between disk and main memory is performed using DMA scheme.

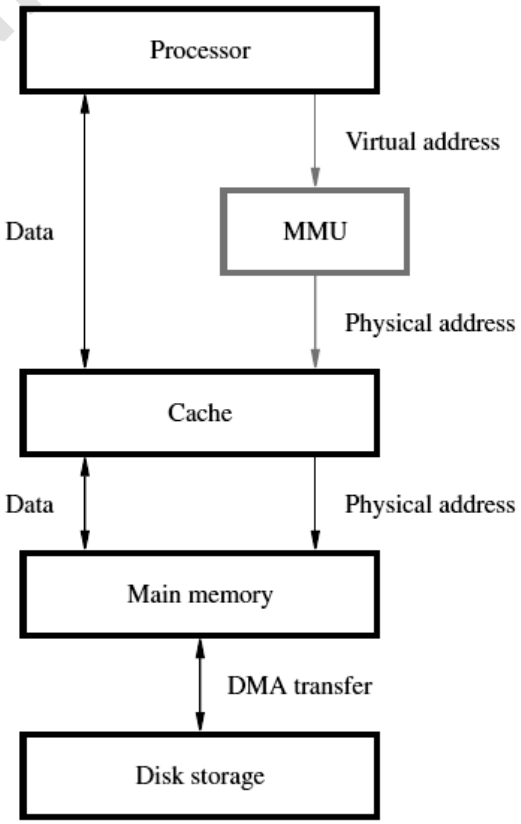


Figure 1

Address Translation

- *Pages:*

  A simple method for translating virtual addresses into physical addresses is to assume that all programs and data are composed of fixed-length units called *pages*, each of which consists of a block of words that occupy contiguous locations in the main memory.
  - Pages commonly range from 2K to 16K bytes in length.
  - They constitute the basic unit of information that is transferred between the main memory and the disk whenever the MMU determines that a transfer is required.
  - Pages should not be too small, because the access time of a magnetic disk is much longer (several milliseconds) than the access time of the main memory. The reason for this is that it takes a considerable amount of time to locate the data on the disk, but once located, the data can be transferred at a rate of several megabytes per second. On the other hand, if pages are too large, it is possible that a substantial portion of a page may not be used, yet this unnecessary data will occupy valuable space in the main memory.
- *Page Table:*

  It contains the information about the main memory address where the page is stored and the current status of the page.

- *Page Frame:*

  An area in the main memory that holds one page is called the page frame.

- *Page Table Base Register:*

  It contains the starting address of the page table.

A virtual-memory address-translation method based on the concept of fixed-length pages is shown schematically in figure 2.

- Each virtual address generated by the processor, whether it is for an instruction fetch or an operand load/store operation, is interpreted as a virtual page number (high-order bits) followed by an offset (low-order bits) that specifies the location of a particular byte (or word) within a page.
- Information about the main memory location of each page is kept in a page table. This information includes the main memory address where the page is stored and the current status of the page.
- An area in the main memory that can hold one page is called a *page frame*.

  The starting address of the page table is kept in a *page table base register*. By adding the *virtual page number* to the contents of this register, the address of the corresponding entry in the page table is obtained.

Virtual Page Number **+** ***Page Table Base register***	} →	***Gives the address of the corresponding in Page Table entry in the page table***

The contents of this location give the starting address of the page if that page currently resides in the main memory.

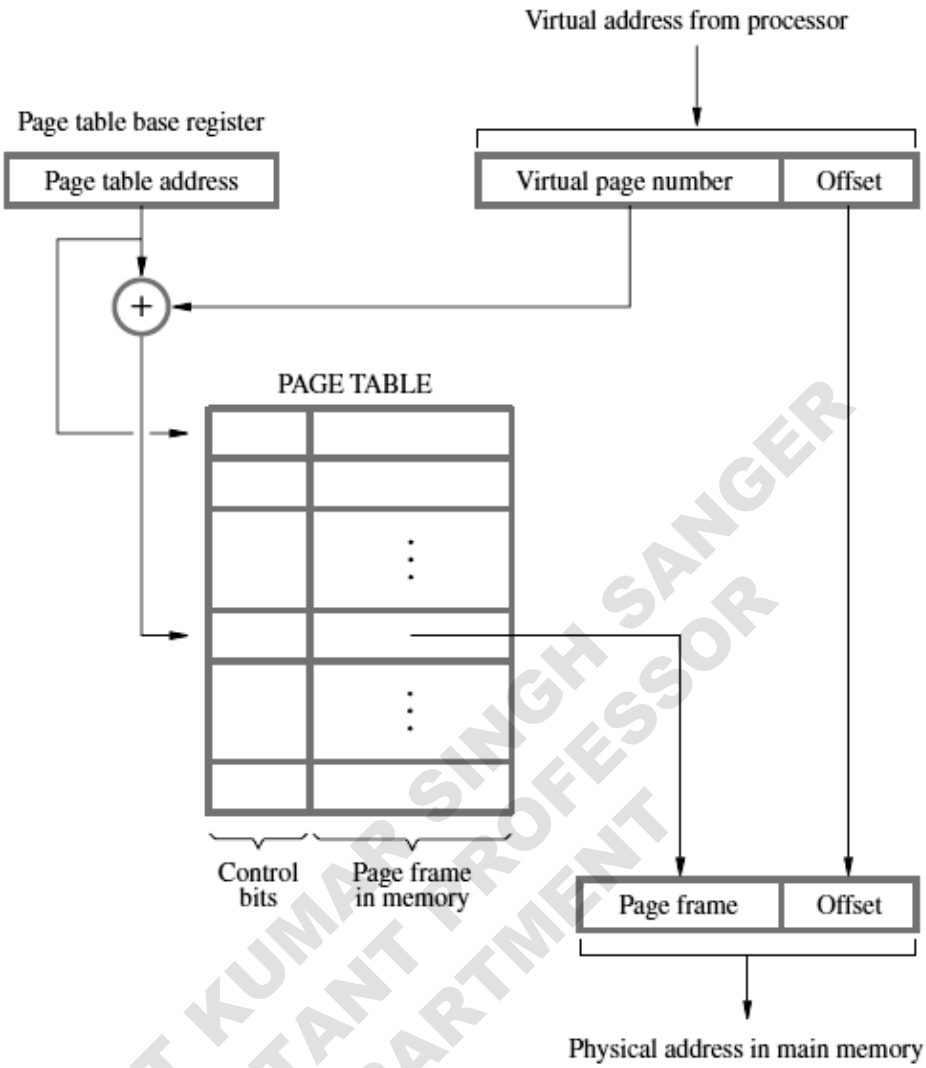


Figure 2

- Each entry in the page table also includes some control bits that describe the status of the page while it is in the main memory.
  - One bit indicates the validity of the page, that is, whether the page is actually loaded in the main memory. It allows the operating system to invalidate the page without actually removing it.
  - Another bit indicates whether the page has been modified during its residency in the memory.
  - Other control bits indicate various restrictions that may be imposed on accessing the page. For example, a program may be given full read and write permission, or it may be restricted to read accesses only.

Translation Lookaside Buffer

- The page table information is used by the MMU for every read and write access.
- Ideally, the page table should be situated within the MMU. Practically it is not possible so a copy of only a small portion of the table is accommodated within the MMU, and the complete table is kept in the main memory.

- The small portion maintained within the MMU consisting the entries corresponding to the most recently accessed pages are stored in a small table called the Translation Lookaside Buffer (TLB).
    - The TLB functions as a cache for the page table in the main memory. Each entry in the TLB includes a copy of the information in the corresponding entry in the page table.
    - In addition, it includes the virtual address of the page, which is needed to search the TLB for a particular page.
    - Figure 3 shows a possible organization of a TLB that uses the associative-mapping technique.
    - Set-associative mapped TLBs are also found in commercial products.

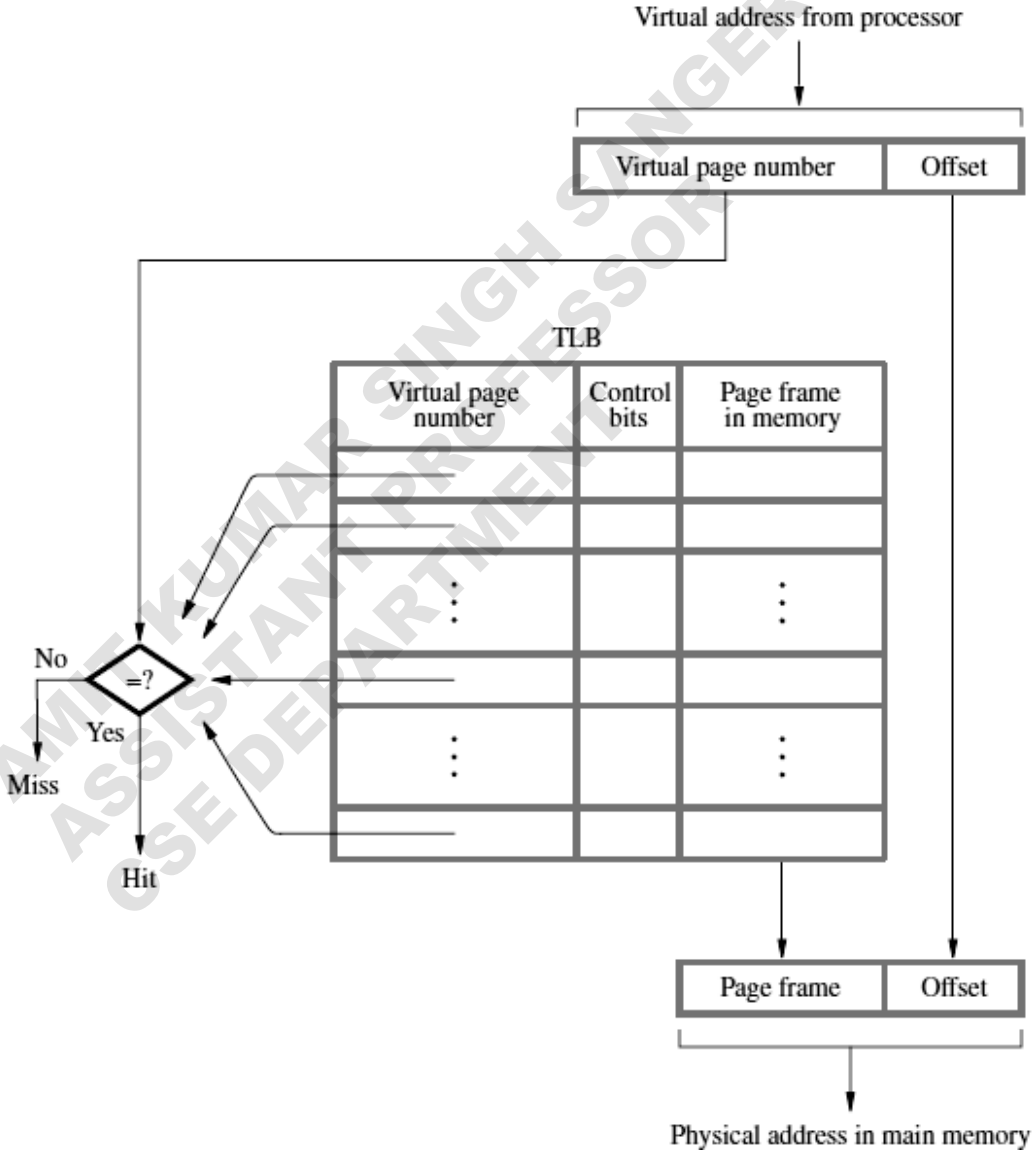


Figure 3

An essential requirement is that the contents of the TLB be coherent with the contents of page tables in the memory.

Address translation proceeds as follows.

- Given a virtual address, the MMU looks in the TLB for the referenced page.
  - If the page table entry for this page is found in the TLB, the physical address is obtained immediately.
  - If there is a miss in the TLB, then the required entry is obtained from the page table in the main memory and the TLB is updated.

It is essential to ensure that the contents of the TLB are always the same as the contents of page tables in the memory. When the operating system changes the contents of a page table, it must simultaneously invalidate the corresponding entries in the TLB.

One of the control bits in the TLB is provided for this purpose. When an entry is invalidated, the TLB acquires the new information from the page table in the memory as part of the MMU's normal response to access misses.

Page Faults

- When a program generates an access request to a page that is not in the main memory, a page fault is said to have occurred. The entire page must be brought from the disk into the memory before access can proceed.
- When it detects a page fault,
  - The MMU asks the operating system to intervene by raising an interrupt.
  - Processing of the program that generated the page fault is interrupted, and control is transferred to the operating system.
  - The operating system copies the requested page from the disk into the main memory. Since this process involves a long delay, the operating system may begin execution of another program whose pages are in the main memory.
  - When page transfer is completed, the execution of the interrupted program is resumed.
- When the MMU raises an interrupt to indicate a page fault, the instruction that requested the memory access may have been partially executed. It is essential to ensure that the interrupted program continues correctly when it resumes execution. There are two options. Either the execution of the interrupted instruction continues from the point of interruption, or the instruction must be restarted.
- If a new page is brought from the disk when the main memory is full, it must replace one of the resident pages. The problem of choosing which page to remove is just as critical and the observation that programs spend most of their time in a few localized areas also applies.
- Two of the most common replacement algorithms used are the first-in, first-out (FIFO) and the least recently used (LRU).
- *The FIFO algorithm* selects for replacement the page that has been in memory the longest time. Each time a page is loaded into memory, its identification number is pushed into a FIFO stack. FIFO will be full whenever memory has no more empty blocks. When a new page must be loaded, the page least recently brought in is removed. The page to be removed is easily determined because its identification number is at the top of the FIFO stack.

  The FIFO replacement policy has the advantage of being easy to implement.

It has the disadvantage that under certain circumstances pages are removed and loaded from memory too frequently.

- *The Least Recently Used* policy is more difficult to implement but has been more attractive on the assumption that the least recently used page is a better candidate for removal than the least recently loaded page as in FIFO.
  The LRU algorithm can be implemented by associating a counter with every page that is in main memory. When a page is referenced, its associated counter is set to zero. At fixed intervals of time, the counters associated with all pages presently in memory are incremented by 1. The least recently used page is the page with the highest count. The counters are often called aging registers, as their count indicates their age, that is, how long ago their associated pages have been referenced.

AUXILIARY MEMORY

The most common auxiliary memory devices used in computer systems are magnetic disks, tapes and optical disks. To understand fully the physical mechanism of auxiliary memory devices one must have knowledge of magnetics, electronics, and electromechanical systems. Although the physical properties of these storage devices can be quite complex, their logical properties can be characterized and compared by a few parameters. The important characteristics of any device are its access mode, access time, transfer rate, capacity, and cost.

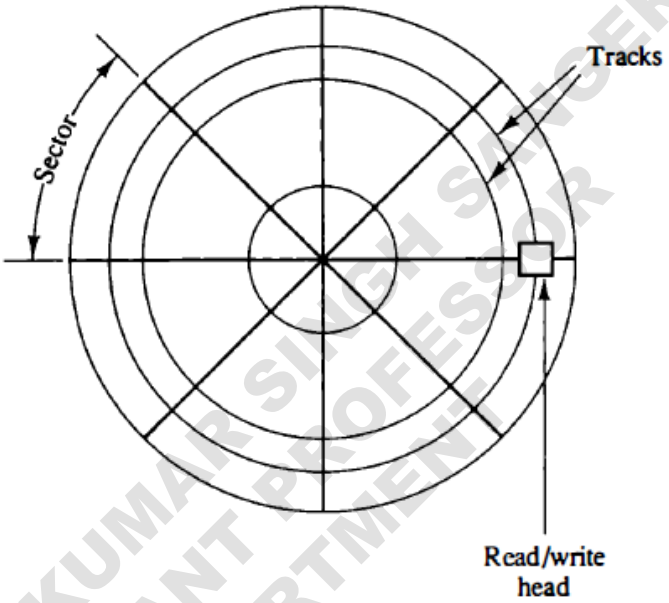
- The average time required to reach a storage location in memory and obtain its contents is called the *access time*.
- In electromechanical devices with moving parts such as disks and tapes, the access time consists of a *seek time* required to position the read-write head to a location and a *transfer time* required to transfer data to or from the device. Because the seek time is usually much longer than the transfer time, auxiliary storage is organized in records or blocks. A record is a specified number of characters or words. Reading or writing is always done on entire records.
- The *transfer rate* is the number of characters or words that the device can transfer per second, after it has been positioned at the beginning of the record.

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. All disks rotate together at high speed and are not stopped or started for access purposes. Bits are stored in the magnetized surface in spots along concentric circles called tracks. The tracks are commonly divided into sections called sectors. In most systems, the minimum quantity of information which can be transferred is a sector. The subdivision of one disk surface into tracks and sectors is shown in Fig. 12-5.

Some units use a single read/write head for each disk surface. In this type of unit, the track address bits are used by a mechanical assembly to move the head into the specified track position before reading or writing. In other disk systems, separate read/write heads are provided for each track in each surface. The address bits can then select a particular track electronically through a decoder circuit. This type of unit is more expensive and is found only in very large computer systems. Permanent timing tracks are used in disks to synchronize the bits and recognize the sectors. A disk system is addressed by address bits that specify the disk number, the disk surface, the sector number and the track within the sector. After the read/write heads are positioned in the specified track, the system has to wait until the rotating disk reaches the specified sector under the read/write head. Information transfer is very fast once the beginning of a sector has been reached. Disks may have multiple heads and simultaneous transfer of bits from several tracks at the same time. A track in a given sector near the circumference is longer than a track near the center of the disk If bits are recorded with equal density, some tracks will contain more recorded

bits than others. To make all the records in a sector of equal length, some disks use a variable recording density with higher density on tracks near the center than on tracks near the circumference. This equalizes the number of bits on all tracks of a given sector. Disks that are permanently attached to the unit assembly and cannot be removed by the occasional user are called hard disks . A disk drive with removable disks is called a floppy disk. The disks used with a floppy disk drive are small removable disks made of plastic coated with magnetic recording material. There are two sizes commonly used, with diameters of 5.25 and 3. 5 inches. The 3.5-inch disks are smaller and can store more data than can the 5.25-inch disks. Floppy disks are extensively used in personal computers as a medium for distributing software to computer users.



Magnetic disk.

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.

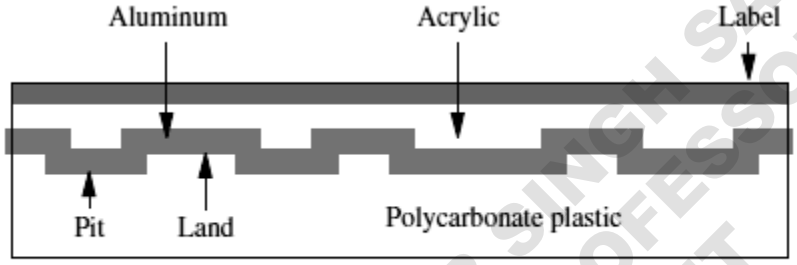
Usually, seven or nine bits are recorded simultaneously to form a character together with a parity bit. Read/write heads are mounted one in each track so that data can be recorded and read as a sequence of characters. Magnetic tape units can be stopped, started to move forward or in reverse, or can be rewound. However, they cannot be started or stopped fast enough between individual characters. For this reason, information is recorded in blocks referred to as records. Gaps of unrecorded tape are inserted between records where the tape can be stopped. The tape starts moving while in a gap and attains its constant speed by the time it reaches the next record. Each record on tape has an identification bit pattern at the beginning and end. By reading the bit

pattern at the beginning, the tape control identifies the record number. By reading the bit pattern at the end of the record, the control recognizes the beginning of a gap. A tape unit is addressed by specifying the record number and the number of characters in the record. Records may be of fixed or variable length.

OPTICAL DISKS

Compact Disk (CD) Technology: The optical technology that is used for CD system is based on laser light source. A laser beam is directed onto the surface of the spinning disk. Physical indentations in the surface are arranged along the tracks of the disk. They reflect the focused beam towards a photo detector, which detects the stored binary patterns.

The laser emits a coherent light beam that is sharply focused on the surface of the disk. Coherent light consists of Synchronized waves that have the same wavelength. If a coherent light beam is combined with another beam of the same kind, and the two beams are in phase, then the result will be brighter beam. But, if a photo detector is used to detect the beams, it will detect a bright spot in the first case and a dark spot in the second case. A cross section of a small portion of a CD shown in figure



Cross-section

The bottom payer is Polycarbonate plastic, which functions as a clear glass base. The surface of this plastic is Programmed to store data by indenting it with pits. The unindented parts are called lands. A thin layer of reflecting aluminum material is placed on top of a programmed disk. The aluminum is then covered by a protective acrylic.

Finally the topmost layer is deposited and stamped with a label the laser source and the Photo detector are positioned below the polycarbonate plastic. The emitted bean travels through this plastic, reflects off the aluminum layer and travels back toward photo detector.

Some important optical disks are listed below

1. ***CD-ROM***

- The CDs used to store computer data are called *CD-ROMs*, because, like semiconductor ROM chips, their contents can only be read.
- Stored data are organized on CD-ROM tracks in the form of blocks called *sectors*. There are several different formats for a sector. The number of sectors per track is variable; there are more sectors on the longer outer tracks.
- CD-ROM drives operate at a number of different rotational speeds. The basic speed, known as 1X, is 75 sectors per second. This provides a data rate of 153,600 bytes/s (150 Kbytes/s), using the Mode 1 format. Higher speed CD-ROM drives are identified in relation to the basic speed. Thus, a 56X CD-ROM has a data transfer rate that is 56 times that of the 1X CD-ROM, or about 6 Mbytes/s. This transfer rate is considerably lower than the transfer rates of magnetic hard disks, which are in the range of tens of megabytes per second.

- Another significant difference in performance is the seek time, which in CD-ROMs may be several hundred milliseconds.
- So, in terms of performance, CD-ROMs are clearly inferior to magnetic disks. Their attraction lies in their small physical size, low cost, and ease of handling as a removable and transportable mass-storage medium. As a result, they are widely used for the distribution of software textbooks, application programs, video games, and so on.

2. CD-RWs (CD-re writables)

- The most flexible CDs are those that can be written multiple times by the user. They are known as CD-RWs (CD-ReWritables).
- The basic structure of CD-RWs is similar to the structure of CD-Rs. Instead of using an organic dye in the recording layer, an alloy of silver, indium, antimony, and tellurium is used.
- The stored data can be erased using the annealing process, which returns the alloy to a uniform crystalline state. A reflective material is placed above the recording layer to reflect the light when the disk is read.
- A CD-RW drive uses three different laser powers.
  - The highest power is used to record the pits.
  - The middle power is used to put the alloy into its crystalline state; it is referred to as the "erase power."
  - The lowest power is used to read the stored information. CD drives designed to read and write CD-RW disks can usually be used with other compact disk media.
- They can read CD-ROMs and can read and write CD-Rs. They are designed to meet the requirements of standard interconnection interfaces, such as SATA and USB.
- CD-RW disks provide low-cost storage media. They are suitable for archival storage of information that may range from databases to photographic images. They can be used for low-volume distribution of information, just like CD-Rs, and for backup purposes. The CD-RW technology has made CD-Rs less relevant because it offers superior capability at only slightly higher cost.

3. DVD technology (Digital Versatile disk)

- The success of CD technology and the continuing quest for greater storage capability has led to the development of DVD (Digital Versatile Disk) technology.
- The first DVD standard was defined in 1996 by a consortium of companies, with the objective of being able to store a full-length movie on one side of a DVD disk.
- The physical size of a DVD disk is the same as that of CDs. The disk is 1.2 mm thick, and it is 120 mm in diameter. Its storage capacity is made much larger than that of CDs by several design changes:
  - Ared-light laser with a wavelength of 635 nm is used instead of the infrared light laser used in CDs, which has a wavelength of 780 nm. The shorter wavelength makes it possible to focus the light to a smaller spot.
  - Pits are smaller, having a minimum length of 0.4 micron.
  - Tracks are placed closer together; the distance between tracks is 0.74 micron.
- Using these improvements leads to a DVD capacity of 4.7 Gbytes.
- Access times for DVD drives are similar to CD drives. However, when the DVD disks rotate at the same speed, the data transfer rates are much higher because of the higher density of pits. Rewritable versions of DVD devices have also been developed, providing large storage capacities.