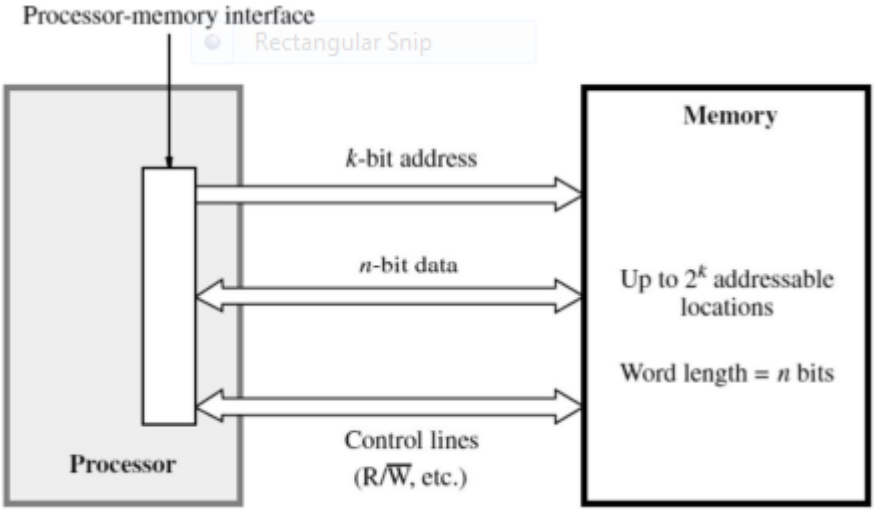
UNIT III

THE MEMORY SYSTEM

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 264 = 16E (exa) ≈ 16 × 1018 locations.



The connection between the processor and its memory consists of address, data, and control lines, as shown in Figure. 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 shown in Figure. 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.

Cache and Virtual Memory:

The processor of a computer can usually process instructions and data faster than they can be fetched from the main memory. Hence, the memory access time is the bottleneck in the system. One way to reduce the memory access time is to use a cache memory. This is a small, fast memory inserted between the larger, slower main memory and the processor. It holds the currently active portions of a program and their data. Virtual memory is another important concept related to memory organization. With this technique, only the active portions of a program are stored in the main memory, and the remainder is stored on the much larger secondary storage device. Sections of the program are transferred back and forth between the main memory and the secondary storage device in a manner that is transparent to the application program. As a result, the application program sees a memory that is much larger than the computer's physical main memory.

Block Transfers:

Data are always transferred in contiguous blocks involving tens, hundreds, or thousands of words. Data transfers between the main memory and high-speed devices such as a graphic display or an Ethernet interface also involve large blocks of data. Hence, a critical parameter for the performance of the main memory is its ability to read or write blocks of data at high speed.

SEMICONDUCTOR RAM MEMORIES:

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 an array, in which each cell is capable of storing one bit of information. A possible organization is illustrated in Figure. Each row of cells constitutes a memory word, and all cells of a row are connected to a common line referred to as the word line, which is driven by the address decoder on the chip. The cells in each column are connected to a

Sense/Write circuit by two bit lines, and the Sense/Write circuits are connected to the data input/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 circuits receive input data and store them in the cells of the selected word.

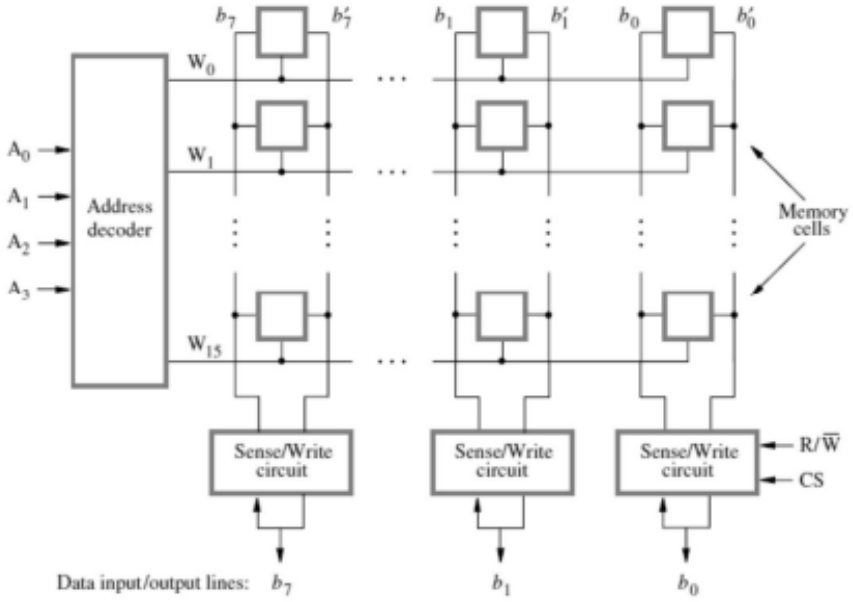
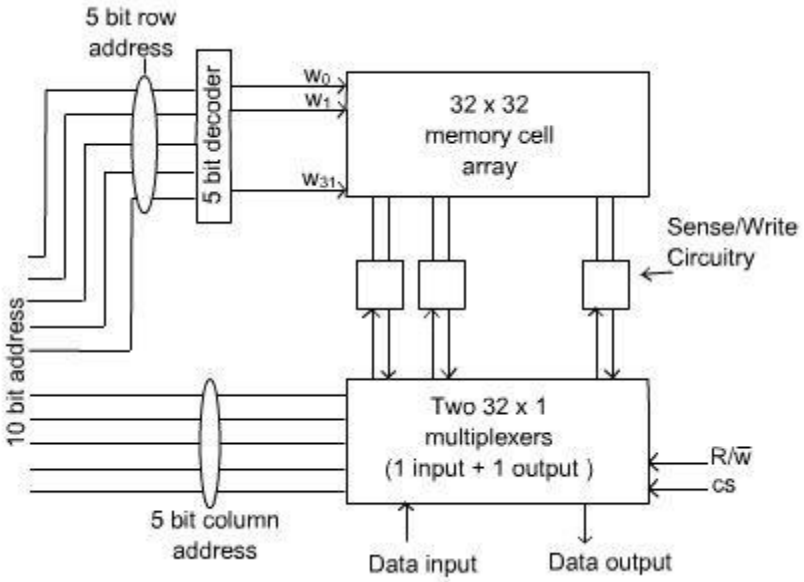


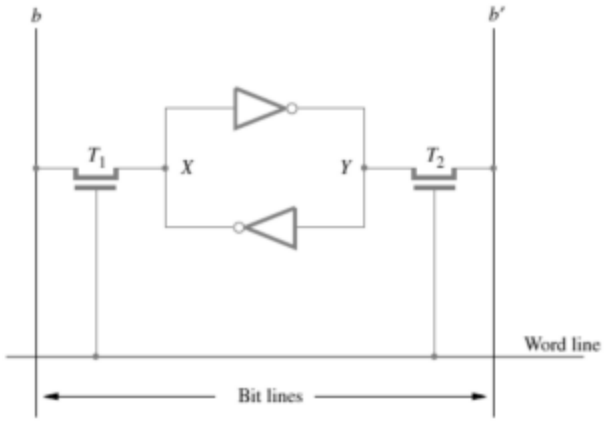
Figure is an example of a very small memory circuit consisting of 16 words of 8 bits each. This is referred to as a 16 × 8 organization. The data input and the data output of each Sense/Write circuit are connected to a single bidirectional data line that can be connected to the data lines of a computer. Two control lines, R/W and CS, are provided. The R/W (Read/Write) input specifies the required operation, and the CS (Chip Select) input selects a given chip in a multichip memory system.

The memory circuit in Figure stores 128 bits and requires 14 external connections for address, data, and control lines. It also needs two lines for power supply and ground connections. Consider now a slightly larger memory circuit, one that has 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.



STATIC MEMORIES;

Memories that consist of circuits capable of retaining their state as long as power is applied are known as static memories. Figure illustrates how a static RAM (SRAM) cell may be implemented. Two inverters are cross-connected to form a latch. The latch is connected to two bit lines by transistors T1 and T2. 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. For example, if the logic value at point X is 1 and at point Y is 0, this state is maintained as long as the signal on the word line is at ground level. Assume that this state represents the value 1.



Read Operation

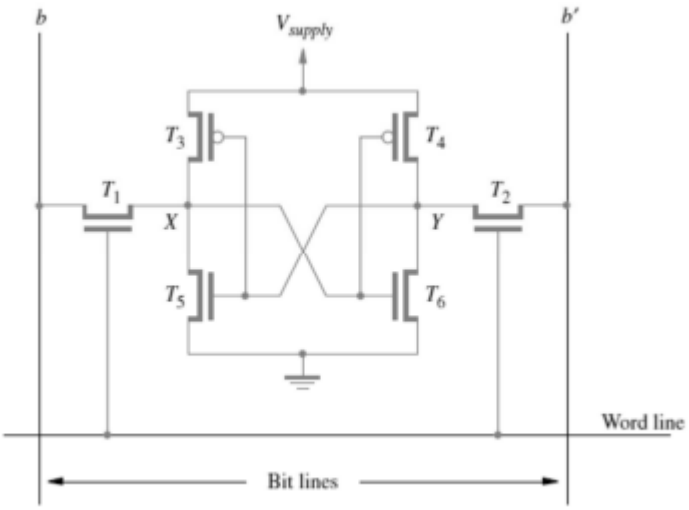
In order to read the state of the SRAM cell, the word line is activated to close switches T1 and T2. If the cell is in state 1, the signal on bit line b is high and the signal on bit line b' is low. The

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: During a Write operation, the Sense/Write circuit drives bit lines b and b', instead of sensing their state. It places the appropriate value on bit line b and its complement on b' and activates the word line. This forces the cell into the corresponding state, which the cell retains when the word line is deactivated.

CMOS Cell:

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

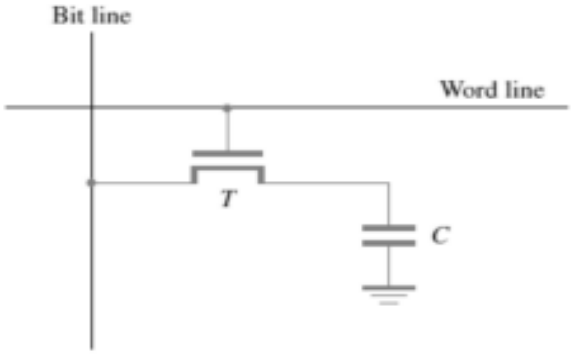


Continuous power is needed for the cell to retain its state. If power is interrupted, the cell's contents are lost. When power is restored, the latch settles into a stable state, but not necessarily the same state the cell was in before the interruption. Hence, SRAMs are said to be volatile memories because their contents are lost when power is interrupted. 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. Otherwise, T1, T2, and one transistor in each inverter are turned off, ensuring that there is no continuous electrical path between Vsupply and ground.

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:

Static RAMs are fast, but their cells require several transistors. 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. 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.

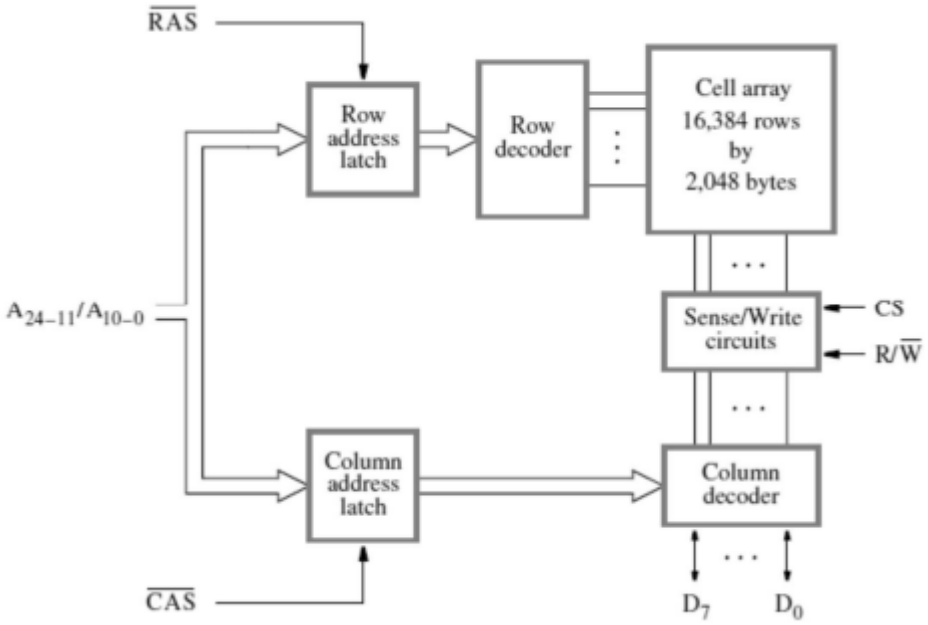


After the transistor is turned off, the charge remains stored in the capacitor, but not for long. The capacitor begins to discharge. Hence, the information stored in the cell can be retrieved correctly only if it is read before the charge in the capacitor drops below some threshold value. During a Read 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.

ASYNCHRONOUS DRAM'S:

A256-Megabit DRAM chip, configured as 32M × 8, is shown in Figure. 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 another 11 bits are needed to specify a group of 8 bits in the selected row. In total, a 25-bit address is needed to access a byte in this memory. The high-order 14 bits and the lower 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 (RAS). This causes a Read operation to be initiated, in which all cells in the selected row are read and refreshed.



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 (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, D7−0. For a Write operation, the information on theD7−0 lines is transferred to the selected circuits, and then used to overwrite the contents of the selected cells in the corresponding 8 columns. We should note that in commercial DRAM chips, the

RAS and CAS control signals are active when low. Hence, addresses are latched when these signals change from high to low. The signals are shown in diagrams as RAS and CAS to indicate this fact. The timing of the operation of the DRAM described above is controlled by the RAS and 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, as we describe later.

Fast Page Mode:

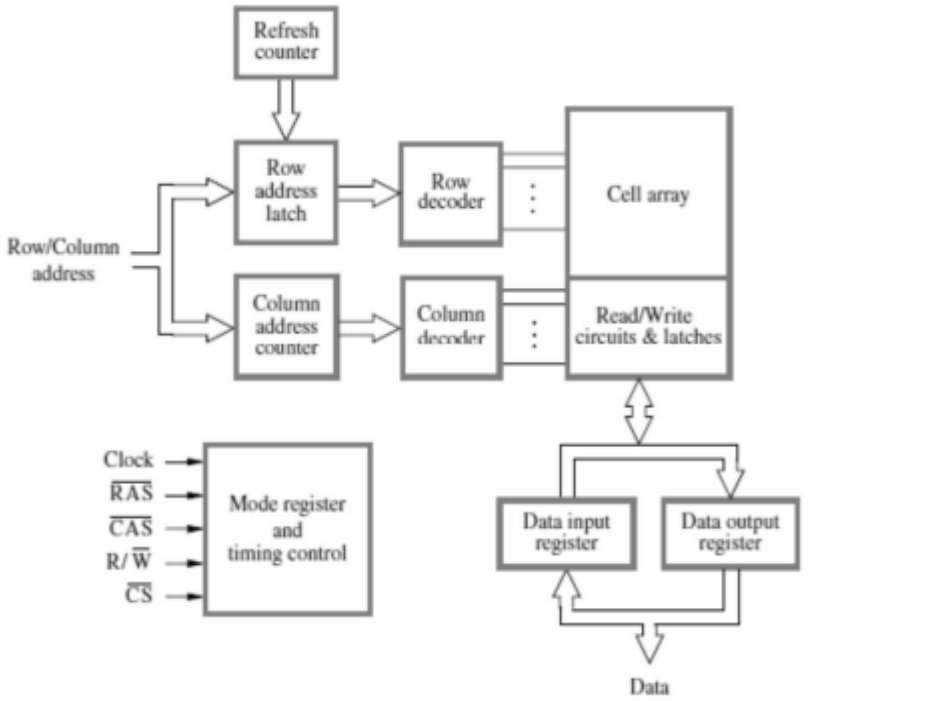
When the DRAM in Figure is accessed, the contents of all 16,384 cells in the selected row are sensed, but only 8 bits are placed on the data lines, D7−0. This byte is selected by the column address, bitsA10−0. A simple addition to the circuit makes it possible to access the other bytes in the same row 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 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.)

SYNCHRONOUS DRAMS:

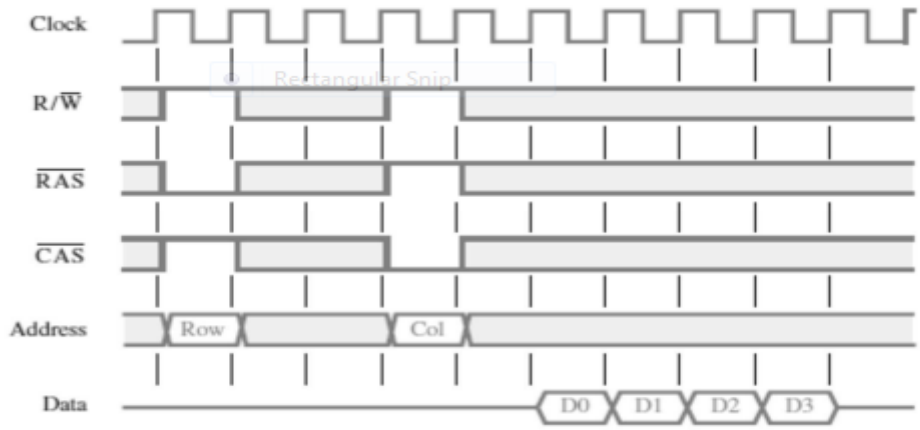
In the early 1990s, developments in memory technology resulted in DRAMs whose operation is synchronized with a clock signal. Such memories are known as synchronous DRAMs (SDRAMs). Their structure is shown in Figure. The cell array is the same as in asynchronous DRAMs. The distinguishing feature of an SDRAM is the use of a clock signal, the availability of which makes it possible to incorporate control circuitry on the chip that provides many useful features. For example, SDRAMs have built-in refresh circuitry, with a refresh counter to provide the addresses of the rows to be selected for refreshing. 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. It is not necessary to provide externally-generated pulses on the CAS line to select successive columns. The necessary control signals are generated internally using a column counter and the clock signal. New data are placed on the data lines at the rising edge of each clock pulse.

Figure shows a timing diagram for a typical burst read of length 4. First, the row address is latched under control of the RAS signal. The memory typically takes 5 or 6 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 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.



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. For instance, in the timing diagram in Figure, the access cycle begins with the assertion of the RAS signal. The first word of data is transferred five clock cycles later. Thus, the latency is five clock cycles. If the clock rate is 500 MHz, then the latency is 10 ns. The remaining three words are transferred in consecutive clock cycles; at the rate of one word every 2 ns. The example above illustrates that we need a parameter other than memory latency to describe the memory's performance during block transfers. 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.

Double-Data-Rate SDRAM:

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. For example, DDR2 and DDR3 can operate at clock frequencies of 400 and 800 MHz, respectively. Therefore, they transfer data using the effective clock speeds of 800 and 1600 MHz, respectively.

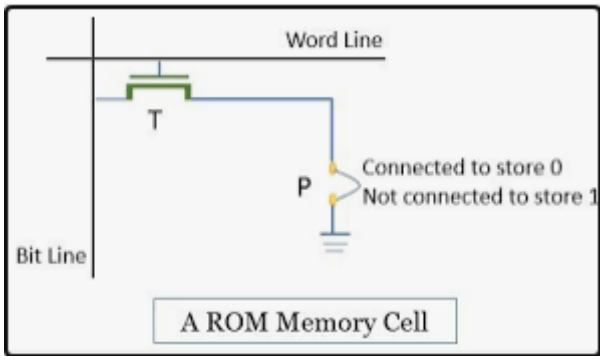
ROM:

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. The embedded applications are another important example. Many embedded applications do not use a hard disk and require non-volatile memories to store their software. Different types of non-volatile memories have been developed. Generally, their contents can be read in the same way as for their volatile counterparts discussed above. But, a special writing process is needed to place the information into a non-volatile memory. Since its normal operation involves only reading the stored data, a memory of this type is called a read-only memory (ROM).

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 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.



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. 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. 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.

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. A flash memory is used in digital cameras to store picture data. In MP3 players, flash memories store the data that represent sound. Single flash chips may not provide sufficient storage capacity for the applications. 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:

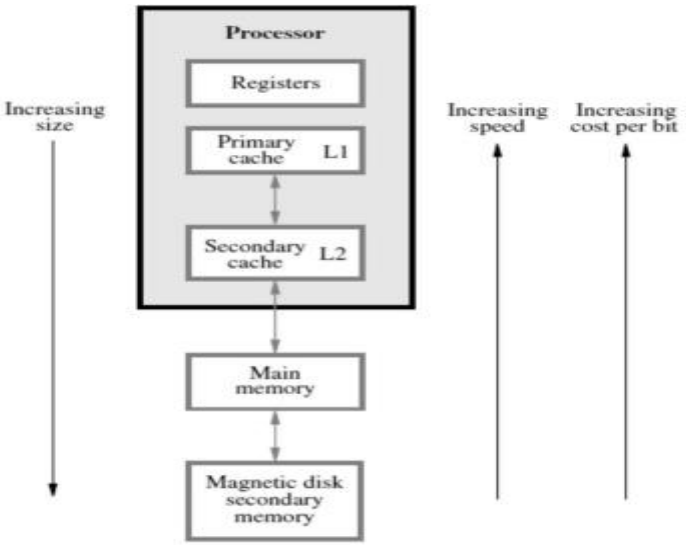
One way of constructing a larger module is to mount flash chips on a small card. Such flash cards have a standard interface that makes them usable in a variety of products. A card is simply plugged into a conveniently accessible slot. Flash cards with a USB interface are widely used and are commonly known as memory keys. They come in a variety of memory sizes. Larger cards may hold as much as 32 G bytes. A minute of music can be stored in about 1 M byte of memory, using the MP3 encoding format. Hence, a 32-Gbyte flash card can store approximately 500 hours of music.

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. However, the storage capacity of flash drives is significantly lower. Currently, the capacity of flash drives is on the order of 64 to 128 G bytes. 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.

SPEED, SIZE AND COST:

An ideal memory would be fast, large, and inexpensive. It is clear that a very fast memory can be implemented using static RAM chips. But, these chips are not suitable for implementing large memories, because their basic cells are larger and consume more power than dynamic RAM cells. Although dynamic memory units with gigabyte capacities can be implemented at a reasonable cost, the affordable size is still small compared to the demands of large programs with voluminous data. A solution is provided by using secondary storage, mainly magnetic disks, to provide the required

memory space. Disks are available at a reasonable cost, and they are used extensively in computer systems. However, they are much slower than semiconductor memory units.



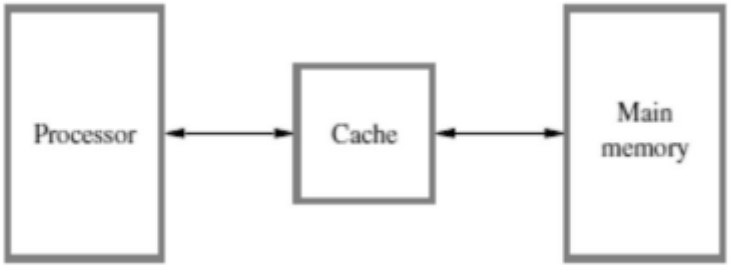
Characteristics	SRAM	DRAM	Magnetis Disk
Speed	Very Fast	Slower	Much slower than DRAM
Size	Large	Small	Small
Cost	Expensive	Less Expensive	Low price

In summary, a very large amount of cost-effective storage can be provided by magnetic disks, and a large and considerably faster, yet affordable, main memory can be built with dynamic RAM technology. This leaves the more expensive and much faster static RAM technology to be used in smaller units where speed is of the essence, such as in cache memories. huge amount of cost effective storage can be provided by magnetic disk; The main memory can be built with DRAM which leaves SRAM‟s to be used in smaller units where speed is of essence

Memory	Speed	Size	Cost
Registers	Very high	Lower	Very Lower
Primary cache	High	Lower	Low
Secondary cache	Low	Low	Low
Main memory	Lower than Seconadry cache	High	High
Secondary Memory	Very low	Very High	Very High

CACHE MEMORIES:

The cache is a small and very fast memory, interposed between the processor and the main memory. Its purpose is to make the main memory appear to the processor to be much faster than it actually is. The effectiveness of this approach is based on a property of computer programs called locality of reference. Analysis of programs shows that most of their execution time is spent in routines in which many instructions are executed repeatedly. These instructions may constitute a simple loop, nested loops, or a few procedures that repeatedly call each other. The actual detailed pattern of instruction sequencing is not important—the point is that many instructions in localized areas of the program are executed repeatedly during some time period. This behaviour manifests itself in two ways: temporal and spatial. The first means that a recently executed instruction is likely to be executed again very soon. The spatial aspect means that instructions close to a recently executed instruction are also likely to be executed soon.



Conceptually, operation of a cache memory is very simple. The memory control circuitry is designed to take advantage of the property of locality of reference. Temporal locality suggests that whenever an information item, instruction or data, is first needed, this item should be brought into the cache, because it is likely to be needed again soon. Spatial locality suggests that instead of fetching just one item from the main memory to the cache, it is useful to fetch several items that are located at adjacent addresses as well. The term cache block refers to a set of contiguous address locations of some size. Another term that is often used to refer to a cache block is a cache line. Consider the arrangement in Figure. When the processor issues a Read request, the contents of a block of memory words containing the location specified are transferred into the cache. Subsequently, when the program references any of the locations in this block, the desired contents are read directly from the cache. Usually, the cache memory can store a reasonable number of blocks at any given time, but this number is small compared to the total number of blocks in the main memory. The correspondence between the main memory blocks and those in the cache is specified by a mapping function. When the cache is full and a memory word (instruction or data) that is not in the cache is referenced, the cache control hardware must decide which block should be

removed to create space for the new block that contains the referenced word. The collection of rules for making this decision constitutes the cache's replacement algorithm

Cache Hits:

The processor does not need to know explicitly about the existence of the cache. It simply issues Read and Write requests using addresses that refer to locations in the memory. The cache control circuitry determines whether the requested word currently exists in the cache. If it does, the Read or Write operation is performed on the appropriate cache location. In this case, a read or write hit is said to have occurred. The main memory is not involved when there is a cache hit in a Read operation. For a Write operation, the system can proceed in one of two ways. In the first technique, called the write-through protocol, both the cache location and the main memory location are updated. The second technique is to update only the cache location and to mark the block containing it with an associated flag bit, often called the dirty or modified bit. The main memory location of the word is updated later, when the block containing this marked word is removed from the cache to make room for a new block. This technique is known as the write-back, or copy-back, protocol.

Cache Misses:

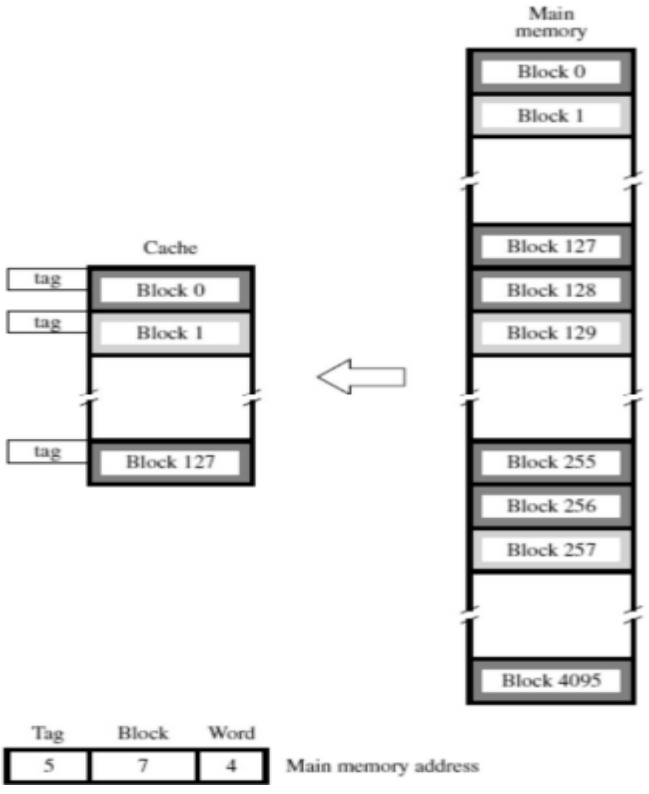
A Read operation for a word that is not in the cache constitutes a Read miss. It causes the block of words containing the requested word to be copied from the main memory into the cache. After the entire block is loaded into the cache, the particular word requested is forwarded to the processor. Alternatively, this word may be sent to the processor as soon as it is read from the main memory. The latter approach, which is called load-through, or early restart, reduces the processor's waiting time somewhat, at the expense of more complex circuitry. When a Write miss occurs in a computer that uses the write-through protocol, the information is written directly into the main memory. For the write-back protocol, the block containing the addressed word is first brought into the cache, and then the desired word in the cache is overwritten with the new information.

MAPPING FUNCTIONS:

There are several possible methods for determining where memory blocks are placed in the cache. It is instructive to describe these methods using a specific small example. Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048 (2K) words, and assume that the main memory is addressable by a 16-bit address. The main memory has 64K words, which we will view as 4K

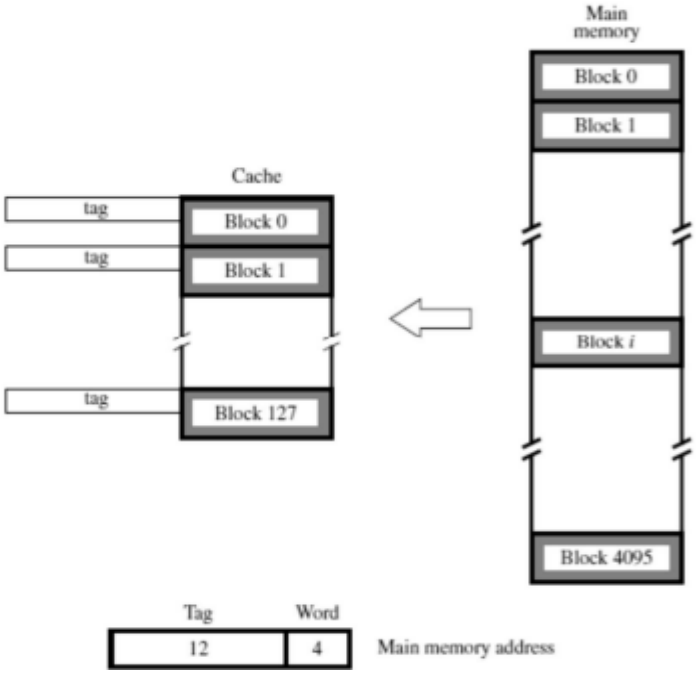
blocks of 16 words each. For simplicity, we have assumed that consecutive addresses refer to consecutive words.

Direct Mapping: The simplest way to determine cache locations in which to store memory blocks is the direct-mapping technique. In this technique, block j of the main memory maps onto block j modulo 128 of the cache, as depicted in Figure. Thus, whenever one of the main memory blocks 0, 128, 256, . . . is loaded into the cache, it is stored in cache block 0. Blocks 1, 129, 257, . . . are stored in cache block 1, and so on. Since more than one memory block is mapped onto a given cache block position, contention may arise for that position even when the cache is not full. For example, instructions of a program may start in block 1 and continue in block 129, possibly after a branch. As this program is executed, both of these blocks must be transferred to the block-1 position in the cache. Contention is resolved by allowing the new block to overwrite the currently resident block. Placement of a block in the cache is determined by its memory address. The memory address can be divided into three fields, as shown in Figure. The low-order 4 bits select one of 16 words in a block. When a new block enters the cache, the 7-bit cache block field determines the cache position in which this block must be stored. The high-order 5 bits of the memory address of the block are stored in 5 tag bits associated with its location in the cache. The tag bits identify which of the 32 main memory blocks mapped into this cache position is currently resident in the cache. The direct-mapping technique is easy to implement, but it is not very flexible



Associative Mapping:

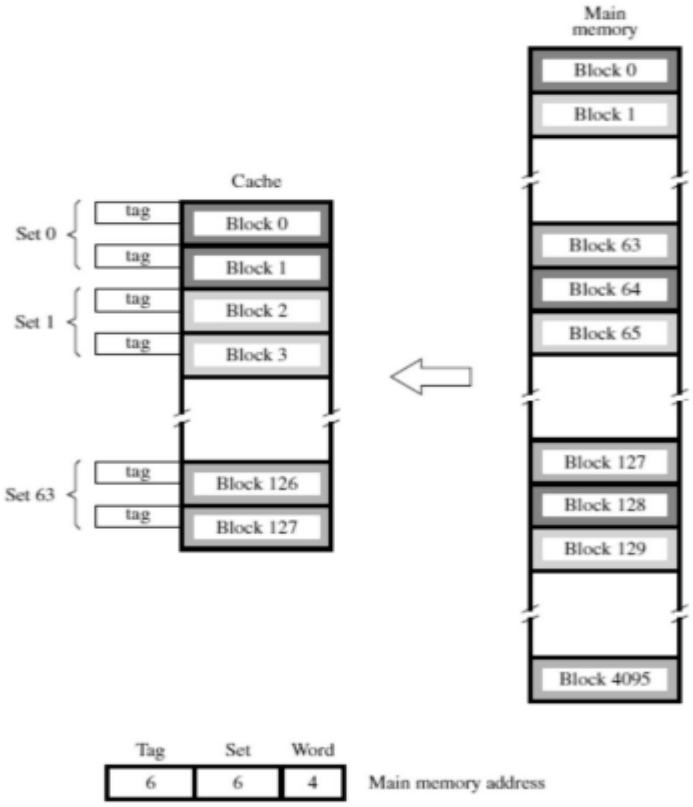
Figure shows the most flexible mapping method, in which a main memory block can be placed into any cache block position. In this case, 12 tag bits are required to identify a memory block when it is resident in the cache. The tag bits of an address received from the processor are compared to the tag bits of each block of the cache to see if the desired block is present. This is called the associative-mapping technique. When a new block is brought into the cache, it replaces (ejects) an existing block only if the cache is full. In this case, we need an algorithm to select the block to be replaced. Many replacement algorithms are possible. The complexity of an associative cache is higher than that of a direct-mapped cache.



Set-Associative Mapping:

Another approach is to use a combination of the direct- and associative-mapping techniques. The blocks of the cache are grouped into sets, and the mapping allows a block of the main memory to reside in any block of a specific set. Hence, the contention problem of the direct method is eased by having a few choices for block placement. At the same time, the hardware cost is reduced by decreasing the size of the associative search. An example of this set-associative-mapping technique is shown in Figure for a cache with two blocks per set. In this case, memory blocks 0, 64, 128, . . . , 4032 map into cache set 0, and they can occupy either of the two block positions within this set. Having 64 sets means that the 6-bit set field of the address determines which set of the cache might contain the desired block. The tag field of the address must then be associatively compared to the

tags of the two blocks of the set to check if the desired block is present. The extreme condition of 128 blocks per set requires no set bits and corresponds to the fully- associative technique, with 12 tag bits. The other extreme of one block per set is the direct mapping method. A cache that has k blocks per set is referred to as a k-way set-associative cache.



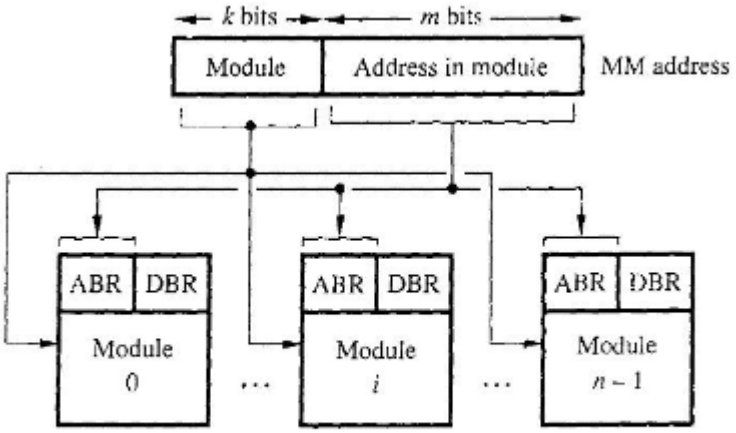
PERFORMANCE CONSIDERATIONS:

IMPROVING CACHE PERFORMANCE:

Two key factors in the commercial success of a computer are performance and cost; A common measure of success is the price/performance ratio. Performance depends on how fast machine instructions can be brought into the processor and how fast they can be executed. The main purpose of this hierarchy is to create a memory that the processor sees as having a short access time and a large capacity. When a cache is used, the processor is able to access instructions and data more quickly when the data from the referenced memory locations are in the cache. Therefore, the extent to which caches improve performance is dependent on how frequently the requested instructions and data are found in the cache.

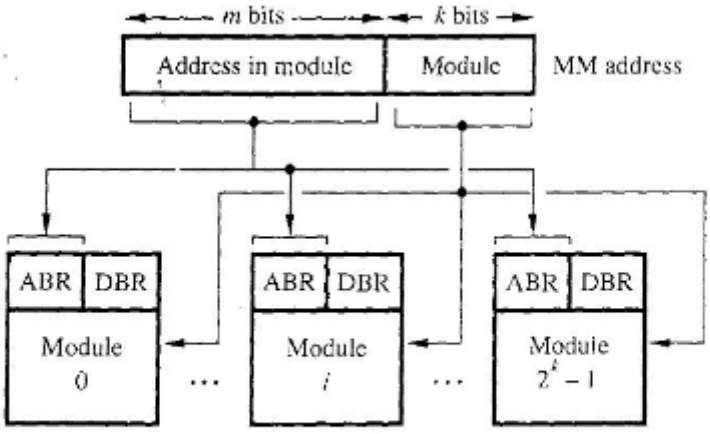
INTERLEAVING:

Main memory of the computer is structured as a collection of physically separate modules. each with its own address buffer register(ABR) and data buffer register(DBR).



(a) Consecutive words in a module

In the first case the memory address generated by the processor is decoded as shown in the fig. The higher order k bits name on one of n modules, and the lower order m bits name a particular word in the module



(b) Consecutive words in consecutive modules

The second and more effective way to address the modules is shown in the fig. It is called memory interleaving. The lower order k bits of the memory address select a module. The higher order m bits name a location within that module.

Hit Rate and Miss Penalty:

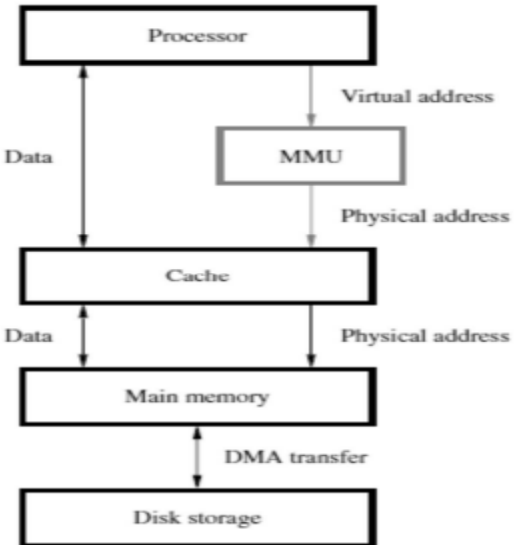
A successful access to data in a cache is called a hit. The number of hits stated as a fraction of all attempted accesses is called the hit rate, and the miss rate is the number of misses stated as a fraction of attempted accesses. Ideally, the entire memory hierarchy would appear to the processor as a single memory unit that has the access time of the cache on the processor chip and the size of the magnetic disk. How close we get to this ideal depends largely on the hit rate at different levels of the hierarchy. High hit rates well over 0.9 are essential for high performance computers.

Performance is adversely affected by the actions that need to be taken when a miss occurs. A performance penalty is incurred because of the extra time needed to bring a block of data from a slower unit in the memory hierarchy to a faster unit. During that period, the processor is stalled waiting for instructions or data. The waiting time depends on the details of the operation of the cache. We refer to the total access time seen by the processor when a miss occurs as the miss penalty.

VIRTUAL MEMORY;

In most modern computer systems, the physical main memory is not as large as the address space of the processor. For example, a processor that issues 32-bit addresses has an addressable space of 4G bytes. The size of the main memory in a typical computer with a 32- bit processor may range from 1G to 4G bytes. If a program does not completely fit into the main memory, the parts of it not currently being executed are stored on a secondary storage device, typically a magnetic disk. As these parts are needed for execution, they must first be brought into the main memory, possibly replacing other parts that are already in the memory. These actions are performed automatically by the operating system, using a scheme known as virtual memory. Under a virtual memory system, programs, and hence the processor, reference instructions and data in an address space that is independent of the available physical main memory space. The binary addresses that the processor issues for either instructions or data are called virtual or logical addresses. These addresses are translated into physical addresses by a combination of hardware and software actions. If a virtual address refers to a part of the program or data space that is currently in the physical memory, then the contents of the appropriate location in the main memory are accessed immediately. Otherwise, the contents of the referenced address must be brought into a suitable location in the memory before they can be used. Figure shows a typical organization that implements virtual memory. A special hardware unit, called the Memory Management Unit (MMU), keeps track of which parts of the virtual address space are in the physical memory. When the desired data or instructions are in the main memory, the MMU translates the virtual address into the corresponding physical address. If

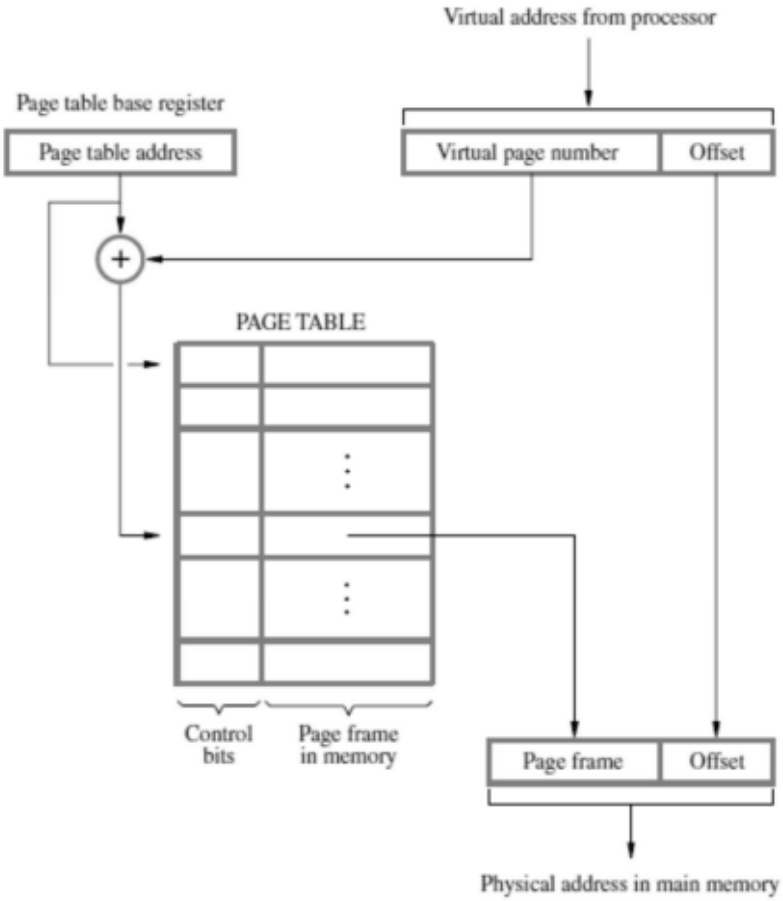
the data are not in the main memory, the MMU causes the operating system to transfer the data from the disk to the memory. Such transfers are performed using the DMA scheme.



Address Translation:

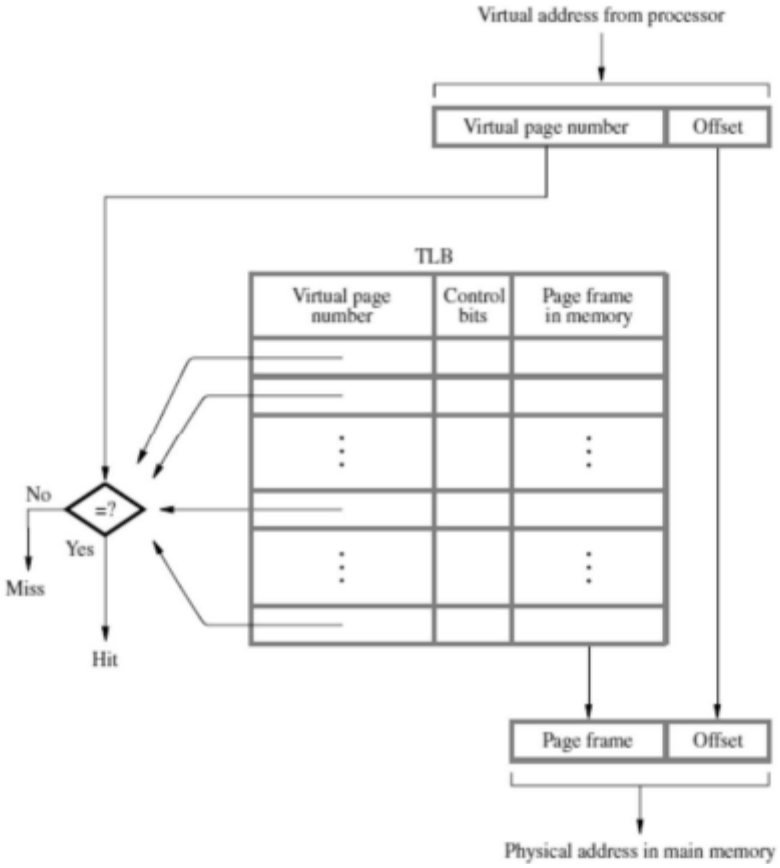
A simple method for translating virtual addresses into physical addresses is to assume that all programs and data are composed of fixed-length units called pages, each of which consists of a block of words that occupy contiguous locations in the main memory. Pages commonly range from 2K to 16K bytes 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.



A virtual-memory address-translation method based on the concept of fixed-length pages is shown schematically in Figure. 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. The contents of this location give the starting address of the page if that page currently resides in the main memory. 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. Another bit indicates whether the page has been modified during its residency in the memory.

Translation Look aside 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. Unfortunately, the page table may be rather large. Since the MMU is normally implemented as part of the processor chip, it is impossible to include the complete table within the MMU. Instead, 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 portion maintained within the MMU consists of the entries corresponding to the most recently accessed pages. They are stored in a small table, usually called the Translation Look aside 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 shows a possible organization of a TLB that uses the associative mapping technique. 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.



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 exception (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.

MEMORY MANAGEMENT REQUIREMENTS:

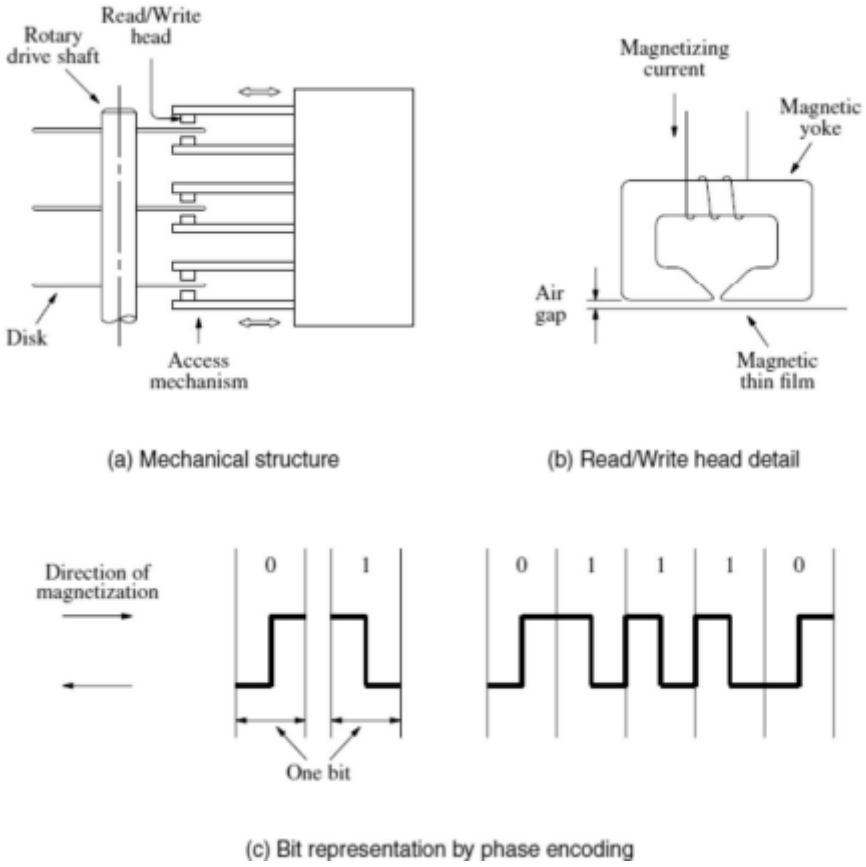
In our discussion of virtual-memory concepts, we have tacitly assumed that only one large program is being executed. If the entire program does not fit into the available physical memory, parts of it (pages) are moved from the disk into the main memory when they are to be executed. Although we have alluded to software routines that are needed to manage this movement of program segments, we have not been specific about the details. Memory management routines are part of the operating system of the computer. It is convenient to assemble the operating system routines into a virtual address space, called the system space, that is separate from the virtual space in which user application programs reside. The latter space is called the user space. In fact, there may be a number of user spaces, one for each user. This is arranged by providing a separate page table for each user program. The MMU uses a page table base register to determine the address of the table to be used in the translation process. Hence, by changing the contents of this register, the operating system can switch from one space to another. The physical main memory is thus shared by the active pages of the system space and several user spaces. However, only the pages that belong to one of these spaces are accessible at any given time. In any computer system in which independent user programs coexist in the main memory, the notion of protection must be addressed. No program should be allowed to destroy either the data or instructions of other programs in the memory. The needed protection can be provided in several ways. Let us first consider the most basic form of protection. Most processors can operate in one of two modes, the supervisor mode and the user

mode. The processor is usually placed in the supervisor mode when operating system routines are being executed and in the user mode to execute user programs. In the user mode, some machine instructions cannot be executed. These are privileged instructions. They include instructions that modify the page table base register, which can only be executed while the processor is in the supervisor mode. Since a user program is executed in the user mode, it is prevented from accessing the page tables of other users or of the system space. It is sometimes desirable for one application program to have access to certain pages belonging to another program. The operating system can arrange this by causing these pages to appear in both spaces. The shared pages will therefore have entries in two different page tables. The control bits in each table entry can be set to control the access privileges granted to each program. For example, one program may be allowed to read and write a given page, while the other program may be given only read access.

ASSOCIATIVE MEMORIES & SECONDARY STORAGE DEVICES:

The semiconductor memories cannot be used to provide all of the storage capability needed in computers. Their main limitation is the cost per bit of stored information. The large storage requirements of most computer systems are economically realized in the form of magnetic and optical disks, which are usually, referred to as secondary storage devices.

Magnetic Hard Disks: The storage medium in a magnetic-disk system consists of one or more disk platters mounted on a common spindle. A thin magnetic film is deposited on each platter, usually on both sides. The assembly is placed in a drive that causes it to rotate at a constant speed. The magnetized surfaces move in close proximity to read/write heads, as shown in Figure a. Data are stored on concentric tracks, and the read/write heads move radically to access different tracks.

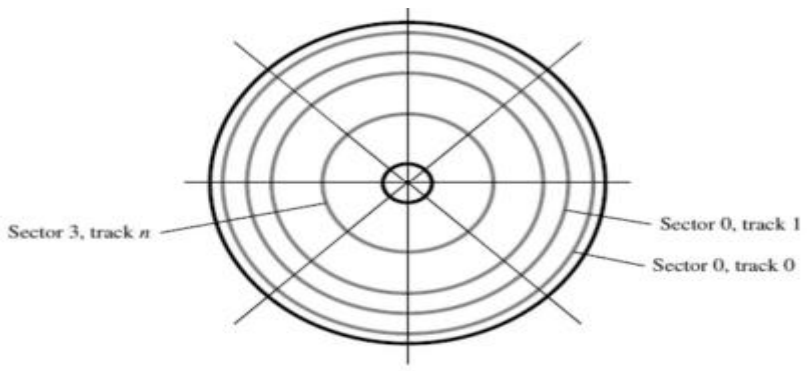


Each read/write head consists of a magnetic yoke and a magnetizing coil, as indicated in Figure b. Digital information can be stored on the magnetic film by applying current pulses of suitable polarity to the magnetizing coil. This causes the magnetization of the film in the area immediately underneath the head to switch to a direction parallel to the applied field. The same head can be used for reading the stored information. In this case, changes in the magnetic field in the vicinity of the head caused by the movement of the film relative to the yoke induce a voltage in the coil, which now serves as a sense coil. The polarity of this voltage is monitored by the control circuitry to determine the state of magnetization of the film. Only changes in the magnetic field under the head can be sensed during the Read operation. Therefore, if the binary states 0 and 1 are represented by two opposite states of magnetization, a voltage is induced in the head only at 0-to-1 and at 1-to-0 transitions in the bit stream. A long string of 0s or 1s causes an induced voltage only at the beginning and end of the string. Therefore, to determine the number of consecutive 0s or 1s stored, a clock must provide information for synchronization. The modern approach is to combine the clocking information with the data. Several different techniques have been developed for such encoding. One simple scheme, depicted in Figure c, is known as phase encoding or Manchester encoding. In this scheme, changes in magnetization occur for each data bit, as shown in the figure. Clocking information is provided by the change in magnetization at the midpoint of each bit period.

The drawback of Manchester encoding is its poor bit-storage density. The space required to represent each bit must be large enough to accommodate two changes in magnetization. We use the Manchester encoding example to illustrate how a self-clocking scheme may be implemented, because it is easy to understand. The flexible spring connection between the head and its arm mounting permits the head to fly at the desired distance away from the surface in spite of any small variations in the flatness of the surface. In most modern disk units, the disks and the read/write heads are placed in a sealed, air-filtered enclosure. This approach is known as Winchester technology. Thus, Winchester disks have a larger capacity for a given physical size compared to unsealed units. Another advantage of Winchester technology is that data integrity tends to be greater in sealed units, where the storage medium is not exposed to contaminating elements. The read/write heads of a disk system are movable. There is one head per surface. All heads are mounted on a comb-like arm that can move radically across the stack of disks to provide access to individual tracks, as shown in Figure a. To read or write data on a given track, the read/write heads must first be positioned over that track. The disk system consists of three key parts. One part is the assembly of disk platters, which is usually referred to as the disk. The second part comprises the electromechanical mechanism that spins the disk and moves the read/write heads; it is called the disk drive. The third part is the disk controller, which is the electronic circuitry that controls the operation of the system. The disk controller may be implemented as a separate module, or it may be incorporated into the enclosure that contains the entire disk system.

Organization and Accessing of Data on a Disk:

The organization of data on a disk is illustrated in Figure. Each surface is divided into concentric tracks, and each track is divided into sectors. The set of corresponding tracks on all surfaces of a stack of disks forms a logical cylinder. All tracks of a cylinder can be accessed without moving the read/write heads. Data are accessed by specifying the surface number, the track number, and the sector number. Read and Write operations always start at sector boundaries.



Data bits are stored serially on each track. Each sector may contain 512 or more bytes. The data are preceded by a sector header that contains identification (addressing) information used to find the desired sector on the selected track. Following the data, there are additional bits that constitute an error-correcting code (ECC). The ECC bits are used to detect and correct errors that may have occurred in writing or reading the data bytes. There is a small inter-sector gap that enables the disk control circuitry to distinguish easily between two consecutive sectors. An unformatted disk has no information on its tracks. The formatting process writes markers that divide the disk into tracks and sectors. During this process, the disk controller may discover some sectors or even whole tracks that are defective. The disk controller keeps a record of such defects and excludes them from use. The formatting information comprises sector headers, ECC bits, and inter-sector gaps. The capacity of a formatted disk, after accounting for the formatting information overhead, is the proper indicator of the disk's storage capability. After formatting, the disk is divided into logical partitions. Figure indicates that each track has the same number of sectors, which means that all tracks have the same storage capacity. The first, called the seek time, is the time required to move the read/write head to the proper track. This time depends on the initial position of the head relative to the track specified in the address. Average values are in the 5- to 8-ms range. The second component is the rotational delay, also called latency time, which is the time taken to reach the addressed sector after the read/write head is positioned over the correct track. On average, this is the time for half a rotation of the disk. The sum of these two delays is called the disk access time.

Data Buffer/Cache:

A disk drive is connected to the rest of a computer system using some standard interconnection scheme, such as SCSI or SATA. The interconnection hardware is usually capable of transferring data at much higher rates than the rate at which data can be read from disk tracks. An efficient way to deal with the possible differences in transfer rates is to include a data buffer in the disk unit. The buffer is a semiconductor memory, capable of storing a few megabytes of data. The requested data are transferred between the disk tracks and the buffer at a rate dependent on the rotational speed of the disk. Transfers between the data buffer and the main memory can then take place at the maximum rate allowed by the interconnection between them. The data buffer in the disk controller can also be used to provide a caching mechanism for the disk. When a Read request arrives at the disk, the controller can first check to see if the desired data are already available in the buffer. If so, the data are transferred to the memory in microseconds instead of milliseconds. Otherwise, the data are read from a disk track in the usual way, stored in the buffer, then transferred to the memory. Because of locality of reference, a subsequent request is likely to refer to data that sequentially

follow the data specified in the current request. In anticipation of future requests, the disk controller may read more data than needed and place them into the buffer. When used as a cache, the buffer is typically large enough to store entire tracks of data. So, a possible strategy is to begin transferring the contents of the track into the data buffer as soon as the read/write head is positioned over the desired track.

Disk Controller:

Operation of a disk drive is controlled by a disk controller circuit, which also provides an interface between the disk drive and the rest of the computer system. One disk controller may be used to control more than one drive. A disk controller that communicates directly with the processor contains a number of registers that can be read and written by the operating system. Thus, communication between the OS and the disk controller is achieved in the same manner as with any I/O interface. The disk controller uses the DMA scheme to transfer data between the disk and the main memory. Actually, these transfers are from/to the data buffer, which is implemented as a part of the disk controller module. The OS initiates the transfers by issuing Read and Write requests, which entail loading the controller's registers with the necessary addressing and control information. Typically, this information includes: Main memory address—The address of the first main memory location of the block of words involved in the transfer. Disk address—The location of the sector containing the beginning of the desired block of words. Word count—The number of words in the block to be transferred. The disk address issued by the OS is a logical address. The corresponding physical address on the disk may be different. For example, bad sectors may be detected when the disk is formatted. The disk controller keeps track of such sectors and maintains the mapping between logical and physical addresses. On the disk drive side, the controller's major functions are:

Seek—Causes the disk drive to move the read/write head from its current position to the desired track.

Read—Initiates a Read operation, starting at the address specified in the disk address register. Data read serially from the disk are assembled into words and placed into the data buffer for transfer to the main memory. The number of words is determined by the word count register.

Write—Transfers data to the disk, using a control method similar to that for Read operations.

Error checking—Computes the error correcting code (ECC) value for the data read from a given sector and compares it with the corresponding ECC value read from the disk. In the case of a

mismatch, it corrects the error if possible; otherwise, it raises an interrupt to inform the OS that an error has occurred. During a Write operation, the controller computes the ECC value for the data to be written and stores this value on the disk.

Floppy Disks:

The disks discussed above are known as hard or rigid disk units. Floppy disks are smaller, simpler, and cheaper disk units that consist of a flexible, removable, plastic diskette coated with magnetic material. The diskette is enclosed in a plastic jacket, which has an opening where the read/write head can be positioned. A hole in the centre of the diskette allows a spindle mechanism in the disk drive to position and rotate the diskette. The main feature of floppy disks is their low cost and shipping convenience. However, they have much smaller storage capacities, longer access times, and higher failure rates than hard disks. In recent years, they have largely been replaced by CDs, DVDs, and flash cards as portable storage media. RAID Disk Arrays Processor speeds have increased dramatically. At the same time, access times to disk drives are still on the order of milliseconds, because of the limitations of the mechanical motion involved. One way to reduce access time is to use multiple disks operating in parallel. They called it RAID, for Redundant Array of Inexpensive Disks. (Since all disks are now inexpensive, the acronym was later reinterpreted as Redundant Array of Independent Disks.) Using multiple disks also makes it possible to improve the reliability of the overall system. Different configurations were proposed, and many more have been developed since. The basic configuration, known as RAID 0, is simple. A single large file is stored in several separate disk units by dividing the file into a number of smaller pieces and storing these pieces on different disks. This is called data striping. When the file is accessed for a Read operation, all disks access their portions of the data in parallel. Various RAID configurations form a hierarchy, with each level in the hierarchy providing additional features. For example, RAID 1 is intended to provide better reliability by storing identical copies of the data on two disks rather than just one. The two disks are said to be mirrors of each other. If one disk drive fails, all Read and Write operations are directed to its mirror drive. Other levels of the hierarchy achieve increased reliability through various parity-checking schemes, without requiring a full duplication of disks. Some also have error recovery capability.

Optical Disks: Storage devices can also be implemented using optical means. The familiar compact disk (CD), used in audio systems, was the first practical application of this technology. Soon after, the optical technology was adapted to the computer environment to provide a high capacity read-only storage medium known as a CD-ROM. The first generation of CDs was developed in the mid-

1980s by the Sony and Philips companies. The technology exploited the possibility of using a digital representation for analog sound signals. To provide high-quality sound recording and reproduction, 16-bit samples of the analog signal are taken at a rate of 44,100 samples per second. Initially, CDs were designed to hold up to 75 minutes, requiring a total of about 3 × 109 bits (3 gigabits) of storage. Since then, higher-capacity devices have been developed.

CD Technology;

The optical technology that is used for CD systems makes use of the fact that laser light can be focused on a very small spot. A laser beam is directed onto a spinning disk, with tiny indentations arranged to form a long spiral track on its surface. The indentations reflect the focused beam toward 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, the result is a brighter beam. But, if the waves of the two beams are 180 degrees out of phase, they cancel each other. Thus, a photo detector can be used to detect the beams. It will see a bright spot in the first case and a dark spot in the second case. Across-section of a small portion of a CD is shown in Figure a. The bottom layer is made of transparent polycarbonate plastic, which serves as a clear glass base. The surface of this plastic is programmed to store data by indenting it with pits. The unintended parts are called lands. A thin layer of reflecting aluminium material is placed on top of a programmed disk. The aluminium is then covered by a protective acrylic. Finally, the topmost layer is deposited and stamped with a label. The total thickness of the disk is 1.2 mm, almost all of it contributed by the polycarbonate plastic. The other layers are very thin. The laser source and the photo detector are positioned below the polycarbonate plastic. The emitted beam travels through the plastic layer, reflects off the aluminium layer, and travels back toward the photo detector.

CD-Rewritable: The most flexible CDs are those that can be written multiple times by the user. They are known as CD-RWs (CD-Rewritable's). 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. This alloy has interesting and useful behaviour when it is heated and cooled. If it is heated above its melting point (500 degrees C) and then cooled down, it goes into an amorphous state in which it absorbs light. But, if it is heated only to about 200 degrees C and this temperature is maintained for an extended period, a process known as annealing takes place, which leaves the alloy in a crystalline state that allows light to pass through. If the crystalline

state represents land area, pits can be created by heating selected spots past the melting point. 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. DVD Technology 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: • A red-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.

Magnetic Tape Systems:

Magnetic tapes are suited for off-line storage of large amounts of data. They are typically used for backup purposes and for archival storage. Magnetic-tape recording uses the same principle as magnetic disks. The main difference is that the magnetic film is deposited on a very thin 0.5- or 0.25-inch wide plastic tape. Seven or nine bits (corresponding to one character) are recorded in parallel across the width of the tape, perpendicular to the direction of motion. A separate read/write head is provided for each bit position on the tape, so that all bits of a character can be read or written in parallel. One of the character bits is used as a parity bit. Data on the tape are organized in the form of records separated by gaps. Tape motion is stopped only when a record gap is underneath the read/write heads. The record gaps are long enough to allow the tape to attain its normal speed before the beginning of the next record is reached. If a coding scheme such as that is used for recording data on the tape, record gaps are identified as areas where there is no change in magnetization. This allows record gaps to be detected independently of the recorded data. To help users organize large amounts of data, a group of related records is called a file. The beginning of a file is identified by a file mark. The file mark is a special single- or multiple-character record, usually preceded by a gap longer than the inter-record gap. The first record following a file mark can be used as a header or identifier for the file. This allows the user to search a tape containing a large number of files for a particular file.