EC7551: COMPUTER ARCHITECTURE AND ORGANIZATION

UNIT IV: MEMORY ORGANIZATION

Presented By,

Dr. V. SATHIESH KUMAR

Assistant Professor

Department of Electronics Engineering

MIT-Anna University

SYLLABUS:

UNIT IV MEMORY ORGANIZATION

Random Access Memories, Serial - Access Memories, RAM Interfaces, Magnetic Surface Recording, Optical Memories, multilevel memories, Cache & Virtual Memory, Memory Allocation, Associative Memory.

TEXT BOOKS:

1. John P.Hayes, Computer architecture and Organization‘, Tata McGraw-Hill, Third edition, 1998.
2. V.Carl Hamacher, Zvonko G. Varanesic and Safat G. Zaky, — Computer Organization—, V edition, McGraw-Hill Inc, 1996.

Presentation Slides:

www.sathieshkumar.com/tutorials

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

- A CPU should have rapid, uninterrupted access to the external memories (program and data is stored) so that the CPU can operate at or near its maximum speed.
- Single memories that operate at speeds comparable to processor speeds are expensive.
- Stored information can be distributed, over various memory units that have very different performance and cost.

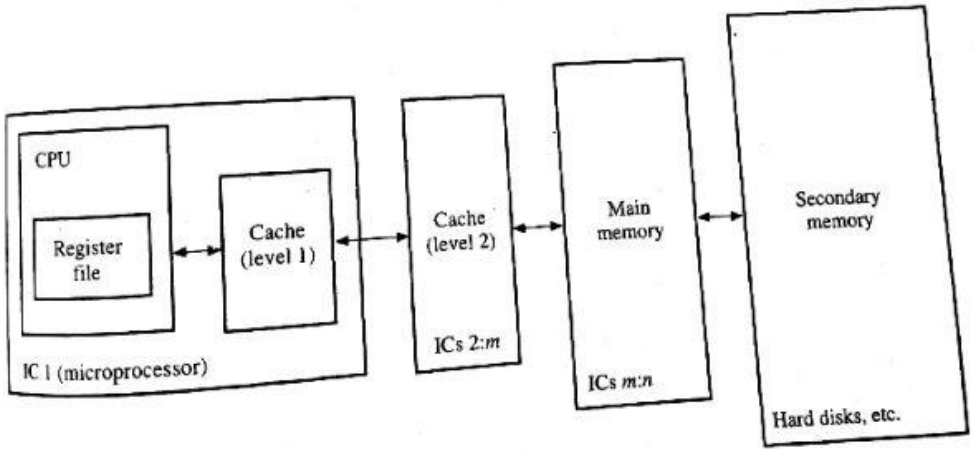


Figure 6.1
Conceptual organization of a multilevel memory system in a computer.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

MEMORY TYPES:

1. **CPU Registers :** High-speed registers in the CPU serve as the working memory for temporary storage of instructions and data.

- Usually form a general-purpose register file for storing data as it is processed.
- A capacity of 32 data words is typical of a register file, and each register can be accessed, that is, read from or written into, within a single clock cycle (a few nanoseconds).

2. **Main (primary) memory :** large, fairly fast external memory stores programs and data that are in active use.

- Storage locations in main memory are addressed directly by the CPU's load and store instructions.
- Access is slower because of main memory's large capacity is typically between 1 and 2^{10} megabyte.
- Access times of five or more clock cycles are usual.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

3. Secondary memory : much larger in capacity but also much slower than main memory.

- It stores system programs, large data files, and the like that are not continually required by the CPU.
- It also acts as an overflow memory when the capacity of the main memory is exceeded.
- Is accessed via input/output programs that transfer information between main and secondary memory.
- Eg: Magnetic hard disk and CD-ROMs – relatively slow electromechanical access mechanisms.
- Capacities in terms of gigabytes, while access times are measured in milliseconds.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

4. Cache : several levels of memory is possible, which is positioned logically between the CPU registers and main memory, but with an access time of one to three cycles, the cache is much faster than main memory because some or all of it can reside on the same IC as the CPU.

- Caches are essential components of high-performance computers that aim to make CPI ≤ 1.
- Unlike the three other memory types, caches are normally transparent to the programmer.
- Together, a computer's caches and main memory implement the external memory M addressed directly by CPU instructions.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Performance and cost :

- Variety of memory devices that employ various electronic, magnetic and optical technologies are available.
- Offers many cost/performance trade-offs.
- The price include not only the cost of the information storage medium itself but also the cost of the peripheral equipment (access circuitry) needed to operate the memory.
- Let C be the price in dollars of a complete memory system with S bits of storage capacity.
- The cost c of the memory is given by,

$$c = C/S \text{ dollars/bit}$$

- A basic performance measure is the average time to read a fixed amount of information, for instance, one word, from the memory – **read access time** or **access time (t_A).**
- Write access time is not always equal to read access time.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

- The access time depends on the physical nature of the storage medium and on the access mechanism used.
- Clearly, low cost and short access time are desirable memory characteristics.
- Memory units with fast access are expensive, while low-cost memories are slow.

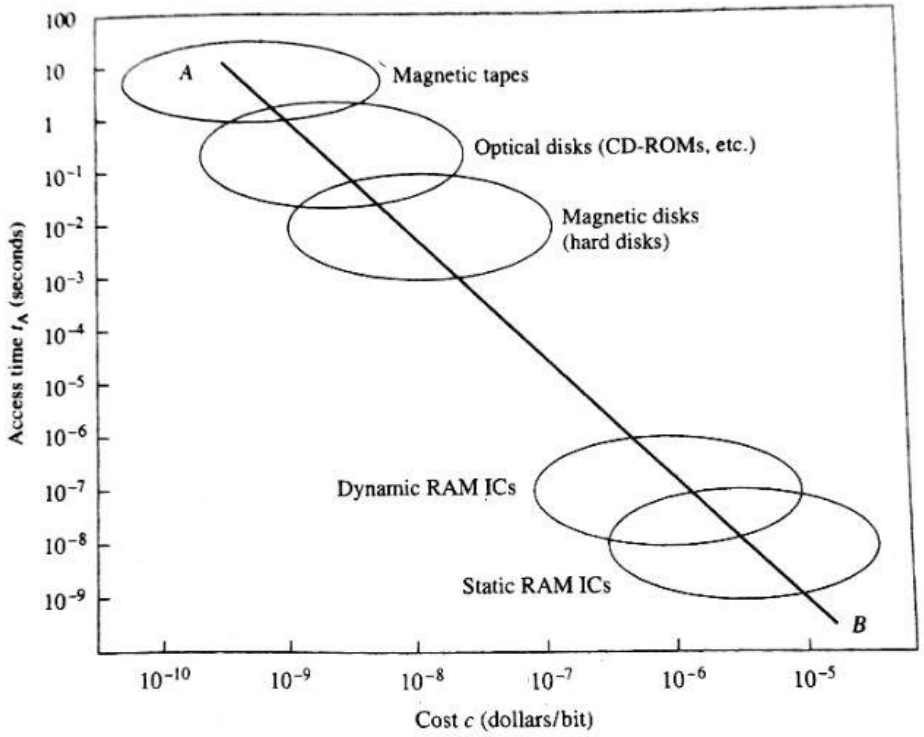


Figure 6.2
Access time versus cost for representative memory technologies.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Access modes:

- A fundamental characteristic of a memory is the order or sequence in which information can be accessed.
- If storage locations can be accessed in any order and access time is independent of the location being accessed, the memory is termed a **random-access memory (RAM).**
- Memories whose storage locations can be accessed only in a certain predetermined sequence are called **serial-access memories**. Eg: Magnetic disks and tapes, as well as, optical memories like CD-ROMs, employ serial-access methods.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Access modes:

- Each storage location in a RAM can be accessed independently of the other locations.
- A separate access mechanism, or **read-write** "head", for every location.

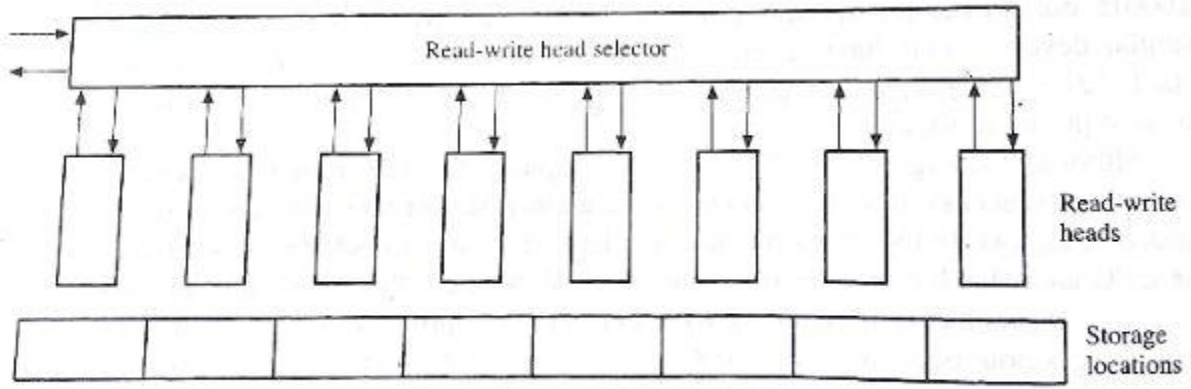


Figure 6.3
Conceptual model of a random-access memory.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Access modes:

- In serial memories, the access mechanism is shared by the storage locations and must be assigned to different locations at different times by moving the stored information, the read-write head or both.
- Many serial-access memories operate by continually moving the storage locations around a closed path or track.

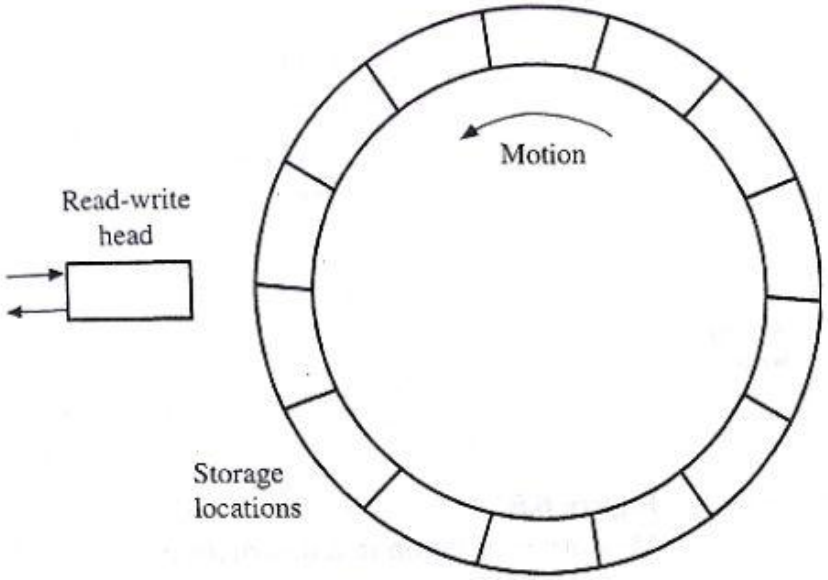


Figure 6.4 Conceptual model of a serial-access memory.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Access modes:

- In serial-access memories, a particular location can be accessed only when it passes the fixed read-write head.
- Hence the time to access a particular location depends on its position relative to the read-write head when the memory receives an access request.
- Since every location has its own access mechanism, random-access memories tend to be more costly than the serial type.
- Serial access memories are slower compared to RAM.
- Memory devices such as magnetic hard disks and CD-ROMs contain many rotating storage tracks.
- If each track has its own read-write head, the tracks can be accessed randomly, but access within each track is serial – access mode is **semirandom**.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Memory retention:

- The method of writing information into a memory can be permanent or irreversible.
- **ROM (read only memories)** – non-erasable storage device. – widely used to store control programs such as microprograms.
- **Compact disk (CD) ROMs** – class of non-erasable secondary memory device – employ an optical (laser) read-write mechanism.
- A standard (12 cm diameter) CD-ROM has a capacity of about 600 MB – used to store large program and data files.
- Semiconductor ROMs whose contents can be changed off-line with some difficulty are called **programmable read-only memories (PROMs)**.
- Programmable CDs are referred to as CD-recordable (CD-R) disks.
- Memories in which reading or writing can be done with impunity on-line are called **read-write memories** - temporary storage purposes - **RAM**

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Memory retention:

- In some technologies, the stored information is lost over a period of time unless corrective action is taken.
- Three characteristics of memories that destroy information in this way are

1. **destructive readout (DRO)** - The method of reading the memory destroys the stored information

- Memories in which reading does not affect the stored data have **nondestructive readout (NDRO)**
- In DRO memories, each read operation must be followed by a write operation that restores the memory's original state.
- This restoration is carried out automatically using a buffer register.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Memory retention:

- The read transfers the word at the addressed (shaded) location to the buffer register where it is available to external devices.
- The contents of the buffer are automatically written back into the original location.

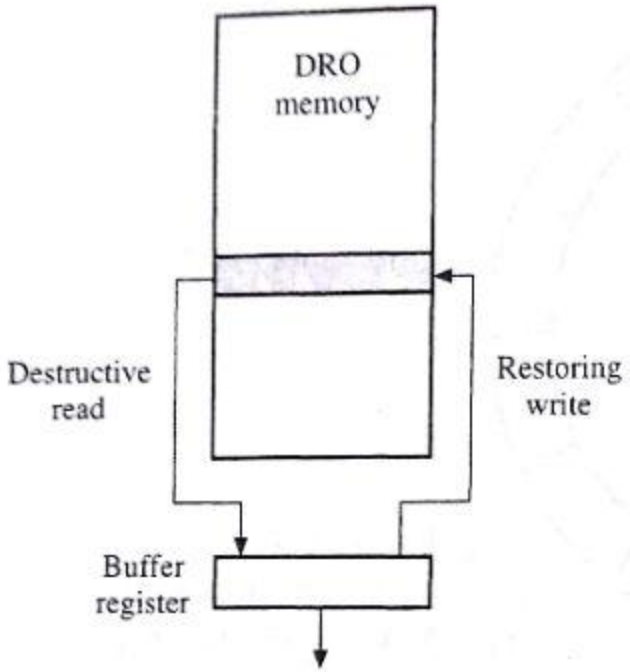


Figure 6.5
Memory restoration in a destructive readout (DRO) memory.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Memory retention:

2. **dynamic storage** – Certain memory devices have the property that a stored 1 tends to become a 0 or vice-versa.

- In some IC memories, an electric charge in a capacitor represents a stored 1; the absence of a stored charge represents a 0.
- Overtime, a stored charge tends to leak away, causing a loss of information unless the charge is restored by a process called **refreshing.**
- Memories that require periodic refreshing are called **dynamic memories**, as opposed to **static memories**, which requires no refreshing.
- Most memories that employ magnetic or optical storage techniques are static.
- Main memories are usually built from dynamic ICs referred to as dynamic RAMS (DRAMs).
- ICs which implement static memories referred to as static RAMS (SRAMs).

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Memory retention:

- SRAM tend to be faster, but have lower access time than DRAMs.
- Cost per bit of SRAMs is higher compared to DRAMs.
- SRAMs are often used to build caches.
- A dynamic memory is refreshed in much the same way that data is restored in a DRO memory.
- The content of every location are sent periodically to buffer registers and then returned in amplified form to their original locations.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Memory retention:

3. Volatility : A memory is volatile if the loss of power destroys the stored information.

- Information can be stored indefinitely in a volatile memory by providing battery backup or other means to maintain a continuous supply of power.
- Most IC memories are volatile, while most magnetic and optical memories are nonvolatile.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Other Characteristics:

- The maximum amount of information that can be transferred to or from the memory per unit time is the **data-transfer rate** or **bandwidth b_M** and is measured in bits or words per second.
- Some DRO and dynamic memories cannot initiate a new access until a restore or refresh operation has been carried out.
- Therefore, the minimum time that must elapse between the start of two consecutive access operations can be greater than t_A. This elapsed time is called the **cycle time t_M** of the memory and represents the time needed to complete a read or write operation.
- **Reliability** – which is measured by the mean time before failure (MTBF)
- In general, memories with no moving parts have much higher reliability than memories such as magnetic disks, which involve considerable mechanical motion.
- Error detecting and error-correcting codes can increase the reliability of the memory.

MEMORY TECHNOLOGY : DEVICE CHARACTERISTICS

Technology	Primary storage medium	Access mode	Alterability	Permanence	Typical access time t_A
Bipolar semiconductor	Electronic	Random	Read/write	NDRO, volatile	10 ns
Metal oxide semiconductor (MOS)	Electronic	Random	Read/write	DRO or NDRO, volatile	50 ns
Magnetic (hard) disk	Magnetic	Semirandom	Read/write	NDRO, nonvolatile	10 ms
Magneto-optical disk	Optical	Semirandom	Read/write	NDRO, nonvolatile	50 ms
Compact disk ROM	Optical	Semirandom	Read only	NDRO, nonvolatile	100 ms
Magnetic tape cartridge	Magnetic	Serial	Read/write	NDRO, nonvolatile	1 s

Figure 6.6
Characteristics of some common memory technologies.

RANDOM ACCESS MEMORIES

➢Each storage location can be accessed independently with fixed access and cycle times that are independent of the position of the accessed location.

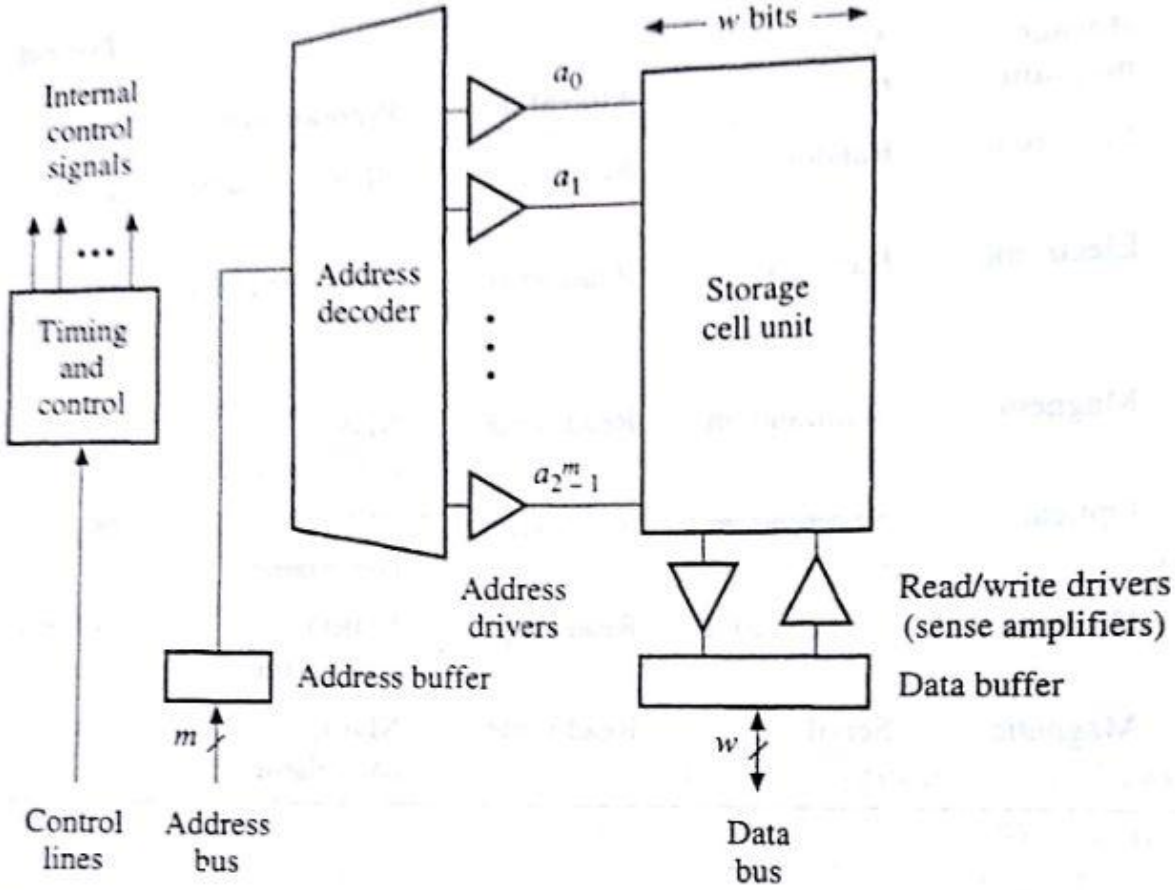


Figure 6.7
One-dimensional (1-D) random-access memory unit.

RANDOM ACCESS MEMORIES

- Storage unit composed of a large number (2^m) of addressable locations, each of which stores a w-bit word.
- Individual bits are not directly addressable unless w=1.
- Referred to as 2^m x w- bit or 2^m- word memory.
- The RAM operates as follows:

1. The address of the target location to be accessed is transferred via the address bus to the RAM's address buffer.
2. The address is then processed by the address decoder, which selects the required location in the storage cell unit.
3. A control line indicates the type of access to be performed.

- If a read operation (load) is requested, the contents of the address location are transferred from the storage cell unit to the data buffer and from there to the data bus.

RANDOM ACCESS MEMORIES

- If a write (store) is requested, the word to be stored is transferred from the data bus to the selected location in the storage unit.
- Single, bidirectional bus is usually preferred.
- The storage unit is made up of many identical 1-bit memory cells and their interconnections.
- The drivers, decoders and control circuits form the **access circuitry** of the RAM and can have significant impact on the total size and cost of the memory.
- Above specified RAM is one-dimensional (1-D)

RANDOM ACCESS MEMORIES

➢**Two-dimensional (2-D)** or **row-column RAM** scheme is shown below,

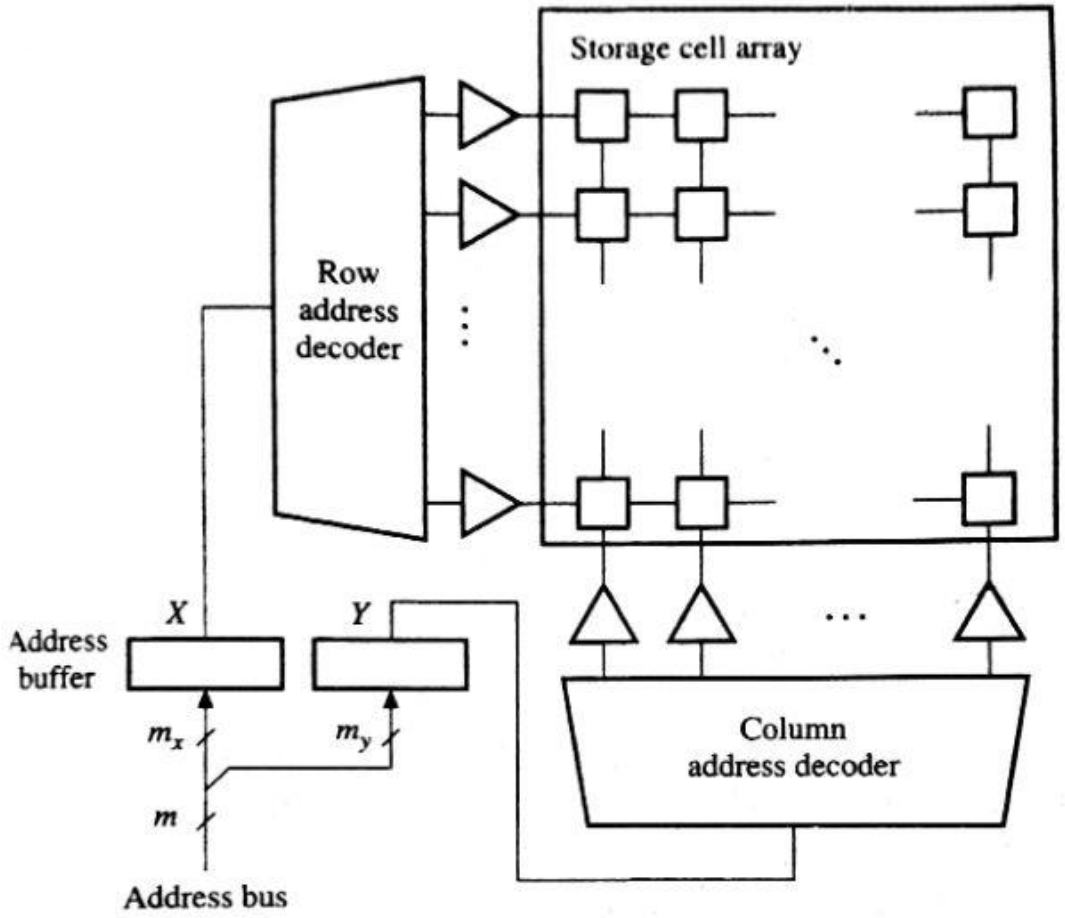


Figure 6.8
Two-dimensional (2-D) RAM addressing scheme.

RANDOM ACCESS MEMORIES

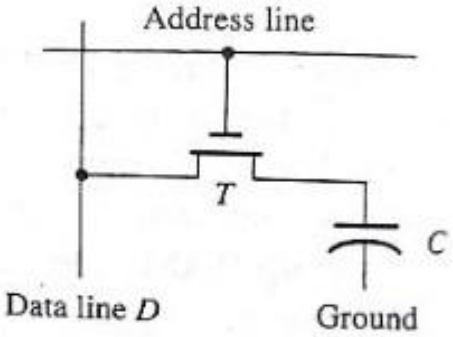
- For simplicity, the data and control circuits are omitted.
- M-bit address word is divided into two parts, X and Y, consisting of m_x and m_y bits, respectively.
- The cells are arranged in a rectangular array of $N_x \leq 2^{mx}$ rows and $N_y \leq 2^{my}$ columns, so the total number of cells is $N = N_x N_y$.
- A cell is selected by the coincidence of signals applied to its X and Y address lines.
- The 2D organization requires much less access circuitry than a 1-D organization for the same storage capacity.

RANDOM ACCESS MEMORIES : SEMICONDUCTOR RAMs

- Storage cells are small transistor circuits – used for high speed CPU registers.
- Single chip RAMs – size – ranging from a few hundred bits to 1 GB or more.
- Both bipolar and MOS transistor circuits are used in RAMs.
- Semiconductor memories fall into two categories – **SRAMs** and **DRAMs** – whose data-retention methods are static and dynamic, respectively.
- SRAMs consists of memory cells that resemble the flip-flops used in processor design.
- SRAM cells differ from flip-flops primarily in the methods used to address the cells and transfer data to and from them.
- Multifunction lines minimize storage-cell complexity and the number of cell connections, thereby facilitating the manufacture of very large 2-D arrays of storage cells.

RANDOM ACCESS MEMORIES : SEMICONDUCTOR RAMs

- In DRAM cell the 1 and 0 states correspond to the presence or absence of a stored charge in a capacitor controlled by a transistor switching circuit.
- DRAM cell can be constructed using a single transistor, whereas a static cell requires up to six transistors, higher storage density is achieved with DRAMs.
- The charge stored in a DRAM cell tends to decay with time, and the cell must be periodically refreshed.
- Both SRAMs and DRAMs are volatile, that is, the stored information is lost when the power source is removed.



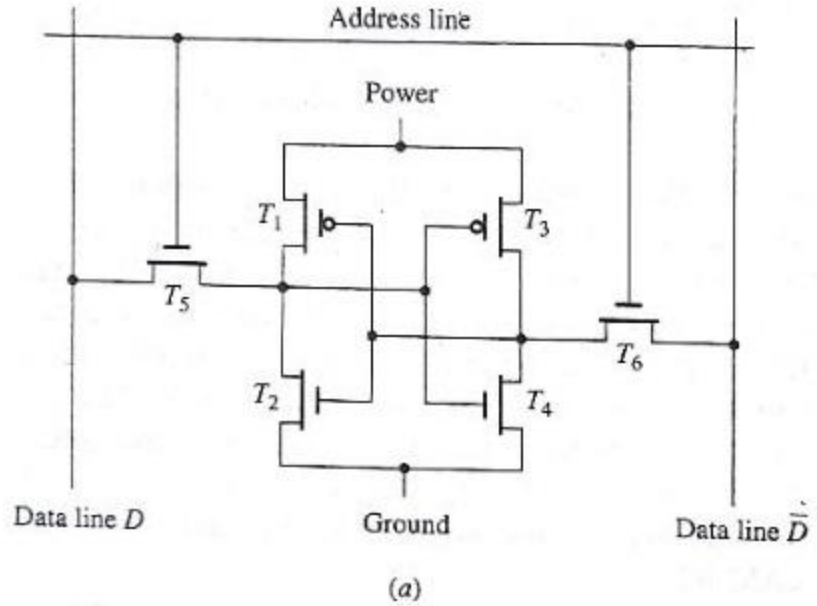
(b)

RANDOM ACCESS MEMORIES : SEMICONDUCTOR RAMs

- In DRAM the readout process is destructive, the data being read out is amplified and subsequently written back to the cell; this process may be combined with the periodic refreshing operation required by dynamic memories.
- The advantages of this DRAM cell are its small size, which means that ICs with very high cell density can be measured, and its low power consumption.

RANDOM ACCESS MEMORIES : SEMICONDUCTOR RAMs

- The six-transistor SRAM cell superficially resembles a flip-flop.
- A signal applied to the address line (also called the **word line**) by the address decoder selects the cell for either the read or write operation.
- The two data lines (also called **bit lines**) are used to transfer the stored data and its complement between the cell and the data drivers.



(a)

Figure 6.9
(a) Static and (b) dynamic RAM cells in MOS technology.

RANDOM ACCESS MEMORIES : RAM DESIGN

➢A RAM IC typically contains all required access circuitry, including address decoders, drivers and control circuits.

➢ 2^m x w-bit RAM IC

➢ **WE** is the write-enable line; a memory write (read) operation takes place if WE=1(0).

➢A second control line, the **chip-select line CS**, triggers a memory operation.

➢ Bidirectional data bus D is directly wired to all addressable storage locations, and so it requires a third control line, **output enable OE**.

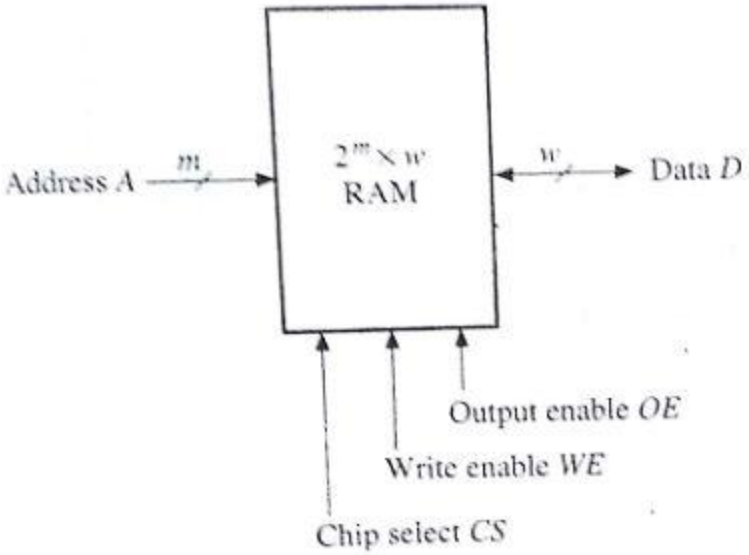


Figure 6.10
A RAM IC showing its major external connections.

RANDOM ACCESS MEMORIES : RAM DESIGN

- In write (input) operations this line is deactivated (OE=0), allowing D to act as an input bus to all storage locations.
- Of course, only the addressed location actually stores the word received on D.
- In read (output) operations, OE must be activated (OE=1) so that only the addressed memory location transfers its data to D.
- Given that N x w-bit RAM ICs denoted $M_{N,w}$ are available, design an N' x w'-bit RAM, where N'>N and/or w'>w.
- A general approach is to construct a p x q array of the $M_{N,w}$ ICs, where p=[N'/N], q=[w'/w], and [x] denotes the smallest integer greater than or equal to x.

RANDOM ACCESS MEMORIES : RAM DESIGN

- Consider a task of designing an N x 4w-bit RAM using N x w-bit ICs.
- Four ICs are needed to quadruple the word size in this way, since p=1 and q=4.

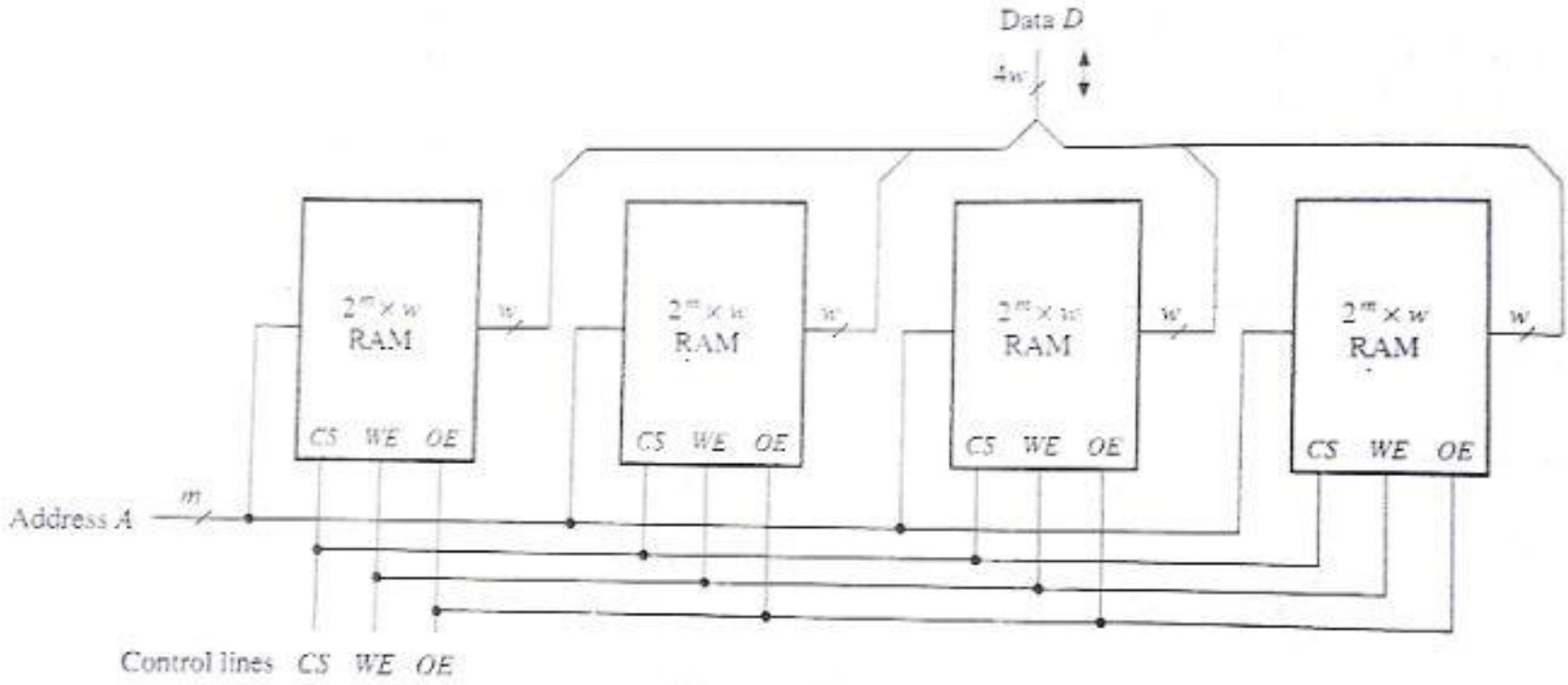


Figure 6.11
Increasing the word size of a RAM by a factor of four.

RANDOM ACCESS MEMORIES : RAM DESIGN

- All the address and control lines are connected in exactly the same way to each IC.
- Their w-bit data buses are concatenated to form a single 4w-bit bus.
- Now suppose we want to increase the number of stored words by a factor of four.
- This time p=4 and q=1, again we need four RAM ICs.
- The number of addresses has quadrupled, hence two lines are added to the address bus.
- Further more, a one-out-of-four address decoder is introduced to decode the extra address bits.
- The original m address lines are connected to the m-bit address bus of every RAM IC; the new lines are the main inputs to the decoder.
- Each decoder output line is connected to the CS inputs of the RAMs in the same row, ensuring that the row has a unique address.
- The output buses of all RAM ICs in the same column are designed so they can be wired together without additional logic.

RANDOM ACCESS MEMORIES : RAM DESIGN

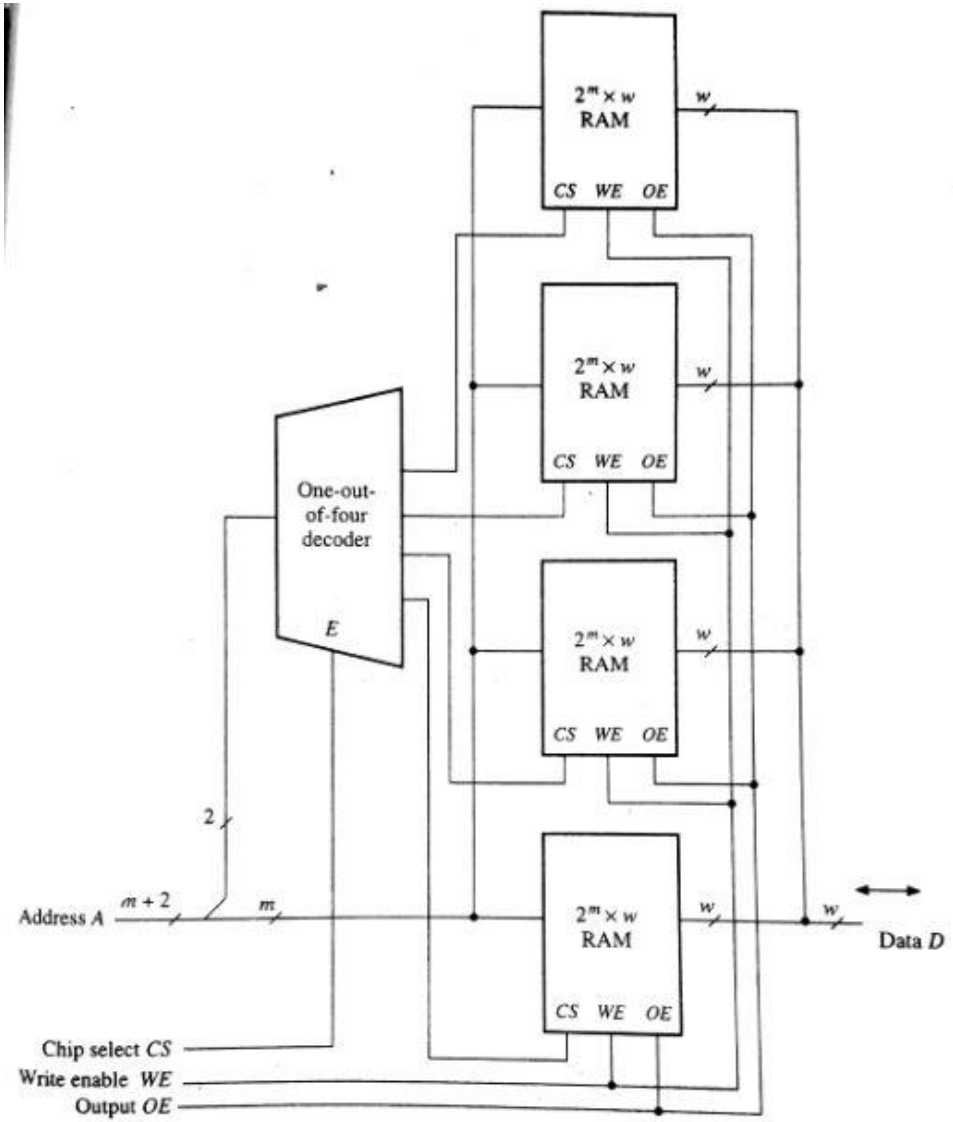


Figure 6.12
Increasing the number of words stored in a RAM by a factor of four.

RAM: OTHER SEMICONDUCTOR MEMORIES

- **Read-only memories (ROMs)** cannot have their contents rewritten once they are installed in a system, that is, on-line.
- A ROM has essentially the same internal organization and external interface as a RAM, but without the latter's writing ability.
- However, ROMs have the advantage of being nonvolatile, so they are widely used to store permanent code at the instruction and microinstruction level.
- **Programmable ROMs (PROMs)** can be programmed repeatedly, which requires their contents to be erased in bulk off-line and then replaced via a special writing process referred to as "programming".
- **Flash memory** offers the same nonvolatality as a PROM, but it can be programmed and erased on-line.
- The programming can be done a bit at a time, but erasure is done in large blocks – a "flash erase" process. Flash memories are suitable for writable control stores and as replacement for secondary memories.

RAM: FAST RAM INTERFACES

- Suppose a particular RAM technology must supply a faster external processor with individually addressable n-bit words.
- There are two basic ways to increase the data-transfer rate across its external interface by a factor of S,

1. Use a bigger memory word :

- Design the RAM with an internal memory word size of w=Sn bits.
- This size permits Sn bits to be accessed as a unit in one memory cycle time T_M.
- Need fast circuits inside the RAM.
- In case of read operation, can access an Sn-bit word, break into S parts, and output them to the processor, all within the period T_M.
- During write operation, these circuits must accept up to S n-bit words from the processor, assemble them into an nS-bit word, and store the result, again within the period T_M.

RAM: FAST RAM INTERFACES

2. Access more than one word at a time

- Partition the RAM into S separate banks M_0, M_1, …, M_{S-1}, each covering part of the memory address space and each provided with its own addressing circuitry.
- Then it is possible to carry out S independent accesses simultaneously in one memory clock period TM.
- Need fast circuits inside the RAM unit to assemble and disassemble the words being accessed.
- The special interface circuits also add substantially to the overall cost of the memory system.

SERIAL ACCESS MEMORIES

- The data in a serial-access memory must be accessed in a predetermined order via read-write circuitry that is shared by different storage locations.
- Large serial memories typically store information in a fixed set of tracks, each consisting of a sequence of 1-bit storage cells.
- A track has one or more access points at which a read-write "head " can transfer information to or from the track.
- A stored item is accessed by moving either the stored information or the read-write heads or both.
- Main application : Secondary computer memories because of their low cost per bit and relatively long access times.

SERIAL ACCESS MEMORIES : ACCESS METHODS

- Long access time is due to several factors:

1. The read-write head positioning time
2. The relatively slow speed at which the tracks move.
3. The fact that data transfer to and from the memory is serial rather than parallel.

- Serial memories such as magnetic hard disks – each track has one or more fixed read-write heads and those whose read-write heads are shared among different tracks.
- In memories that share read-write heads, the need to move the heads between tracks introduces a delay.
- The average time to move a head from one track to another is the **seek time t_S** of the memory.
- Once the head is in position, the desired cell may be in the wrong part of the moving storage track.

SERIAL ACCESS MEMORIES : ACCESS METHODS

- Some time is required for this cell to reach the read-write head so that data transfer can begin.
- The average time for this movement to take place is the **latency t_L** of the memory.
- In memories where information rotates around a closed track t_L is called the **rotational latency**.
- Each storage cell in a track stores a single bit.
- A w-bit word may be stored in two different ways.
- It can consist of w consecutive bits along a single track.
- Alternatively, w tracks may be used to store the word, with each track storing a different bit.
- BY synchronizing the w tracks and providing a separate read-write head for each track, all w bits can be accessed simultaneously.
- In either case it is inefficient to read or write just one word per serial access, since the seek time and the rotational latency consume so much time.

SERIAL ACCESS MEMORIES : ACCESS METHODS

- Words are therefore grouped into larger units called **blocks**.
- All the words in a block are stored in consecutive locations so that the time to access an entire block includes only one seek and one latency time.
- Once the read-write head is positioned at the start of the requested word or block, data is transferred at a rate that depends on two factors:

1. The speed of the stored information relative to the read-write head
2. Storage density along the track

- The speed at which data can be transferred continuously to or from the track under these circumstances is the **data-transfer rate**.
- If a track has a storage density of T bits/cm and moves at a velocity of V cm/s past the read-write head, then data-transfer rate is TV bits/s.

SERIAL ACCESS MEMORIES : ACCESS METHODS

- The time t_B needed to access a block of data in a serial-access memory can be estimated as follows.
- Assume that the memory has closed, rotating storage tracks.
- Let each track have a fixed (average) capacity of N words and rotate at r revolutions per second.
- Let n be the number of words per block.
- The data-transfer rate of the memory is then rN words/s.
- Once the read-write head is positioned at the start of the desired block, its data can be transferred in approximately n/(rN) seconds.
- The average latency is 1/(2r) seconds, which is the time needed for half a revolution.
- If t_S is the average seek time, then an appropriate formula for t_B is

$$t_B = t_S + \frac{1}{2r} + \frac{n}{rN}$$

SERIAL ACCESS MEMORIES : MEMORY ORGANIZATION

➢Assume that each word is stored along a single track and that each access results in the transfer of a block of words.

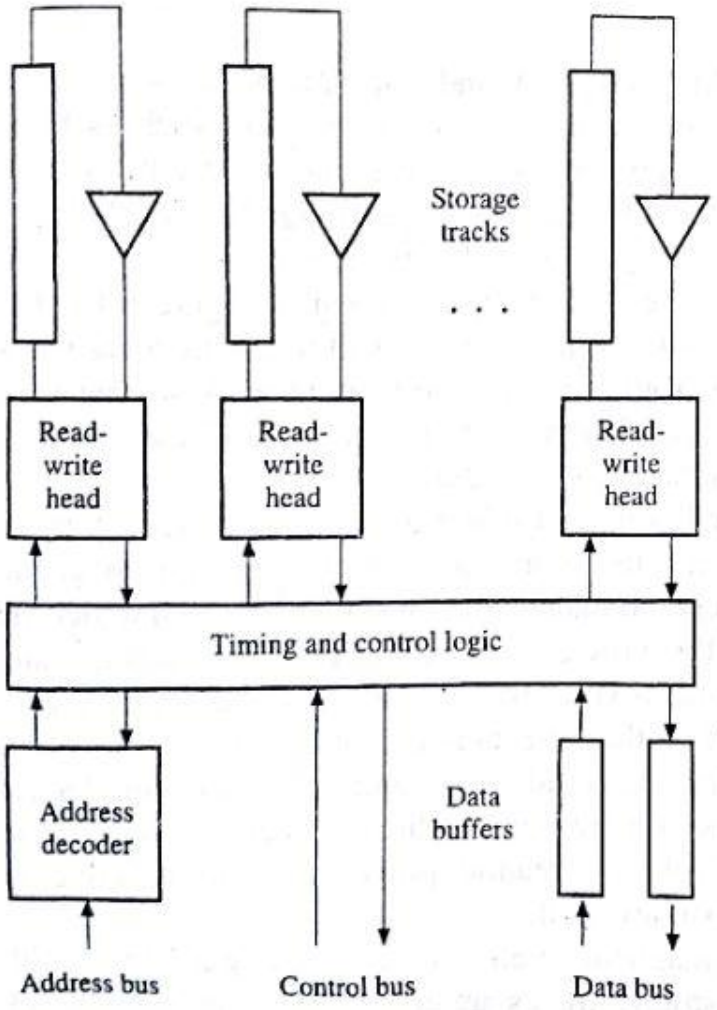


Figure 6.15
Organization of a serial-access memory unit.

SERIAL ACCESS MEMORIES : MEMORY ORGANIZATION

- The address of the data to be accessed is applied to the address decoder, whose output determines the track to be used (the track address) and the location of the desired block of information within the track (the block address).
- The track address determines the particular read-write head to be selected.
- Then if necessary, the selected head is moved into position to transfer data to or from the target track.
- The desired block cannot be accessed until it coincides with the selected head.
- To determine when this condition occurs, a track position indicator generates the address of the block that is currently passing the read-write head.
- The generated address is compared with the block address produced by the address decoder.
- When they match, the selected head is enabled and the data transfer between the storage track and the memory data buffer registers begins. The read-write head is disabled when a complete block of information has been transferred.

SERIAL ACCESS MEMORIES : MEMORY ORGANIZATION

- In many memories the read-write heads or the storage locations are moved through space by electromechanical devices such as electric motors, in order to perform an access.
- Eg : Magnetic disk and tape units, optical memories (CD-ROMs)

SERIAL ACCESS MEMORIES : MAGENTIC SURFACE RECORDING

- Magnetic disk and tape memories store information on the surface of tracks coated with a magnetic medium such as ferric oxide.
- Each cell of a track has two stable magnetic states that represent logical 0 and 1.
- These magnetic states are defined by the direction or magnitude of the cell's magnetic flux in the cell.
- Electric current alters and sense the magnetic states, for example, via an inductive read-write head.
- The read and write signals pass through coils around a ring of soft magnetic material.
- A very narrow gap separates the ring from a cell on the storage track so that their respective magnetic fields can interact.
- This interaction permits information transfer between the read-write head and the storage medium.

SERIAL ACCESS MEMORIES : MAGENTIC SURFACE RECORDING

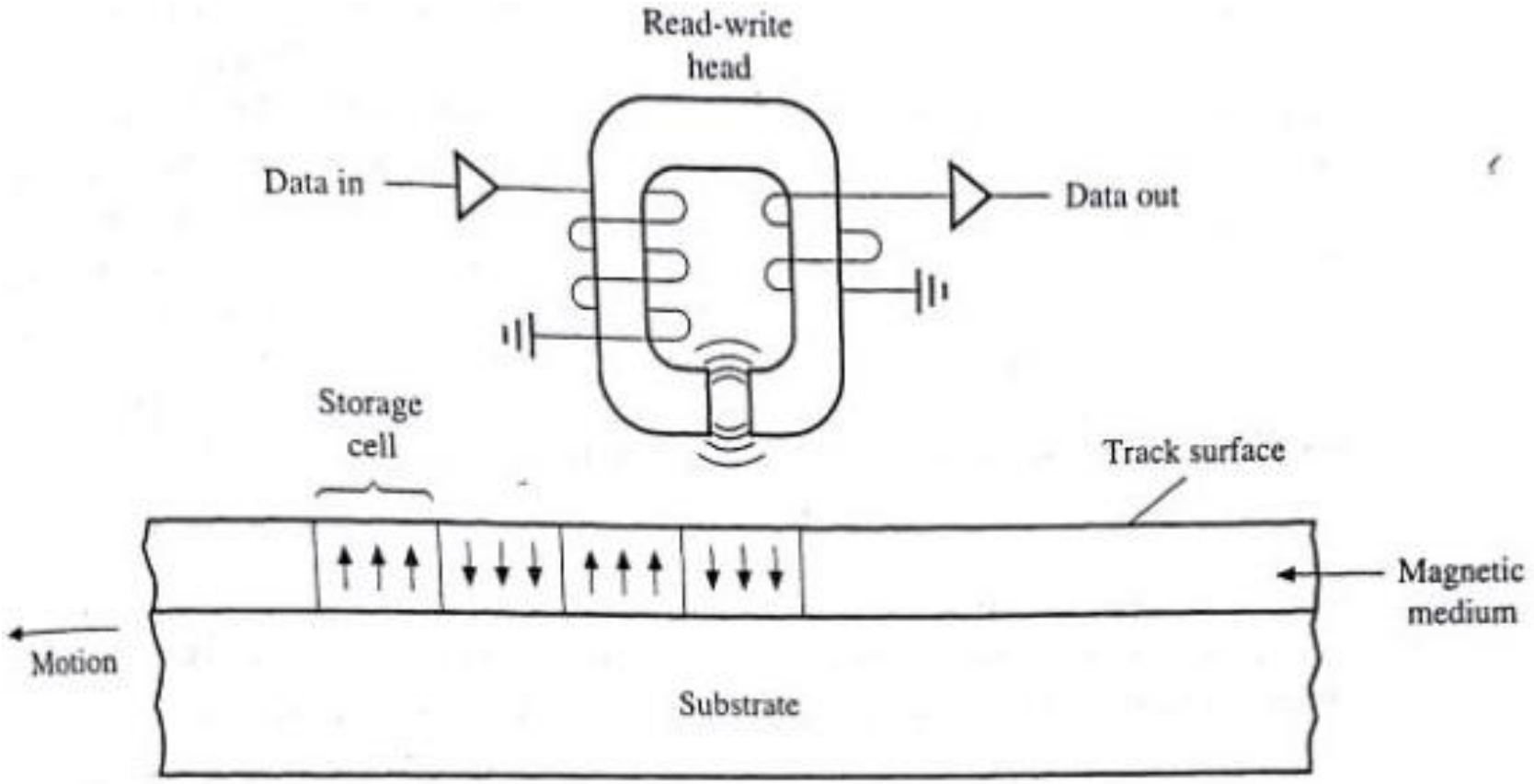


Figure 6.16
Magnetic-surface recording mechanism.

SERIAL ACCESS MEMORIES : MAGENTIC SURFACE RECORDING

- To **write data**, the addressed cell is moved under the read-write gap.
- A pulse of current is then transmitted through the write coil, which alters the magnetic field at the ring gap; this in turn alters the magnetization state of the cell under the gap.
- The direction or magnitude of the write current determines the resulting state.
- To **read a cell,** it is moved past the read-write head, causing the magnetic field of the cell to induce a magnetic field in the core material of the read-write head.
- Since the cell is in motion, this magnetic field varies and so induces an electric voltage pulse in the read coil.
- This voltage pulse, which is then fed to a sense amplifier, identifies the state of the cell.
- The readout process is nondestructive; in addition, magnetic surface storage is nonvolatile.
- In disk memories the tracks form concentric circles on the surface of a plastic or metal disk.

SERIAL ACCESS MEMORIES : MAGENTIC DISK MEMORIES

- In tape memories the tracks form parallel lines on the surface of a long, narrow plastic tape.
- A **magnetic-disk** unit employs storage media consisting of thin disks with a coating of magnetic material on which data can be recorded.
- One or both surfaces of a disk contain thousands of recording tracks arranged in concentric circles.
- Several disks can be attached to a common spindle; the four disks provide up to eight recording surfaces.
- During operation of the memory, the disks are rotated at a constant speed by a **disk drive unit**.
- Each recording surface is supplied with at least one read-write head.
- The read-write heads can be connected to form a read-write arm, so that all heads move in unison.

SERIAL ACCESS MEMORIES : MAGENTIC DISK MEMORIES

- This arm moves back and forth to select a particular set of tracks for reading or writing.
- The recording surface is divided into **sectors** so that the part of a track within a sector stores a fixed amount of information corresponding to the memory unit's block size.
- Memory control is simplified if all tracks store the same amount of data, in which case the track density (bits stored per cm) on the outer tracks is less than the maximum possible.
- Example : Floppy Disk, Hard Disk

SERIAL ACCESS MEMORIES : MAGENTIC SURFACE RECORDING

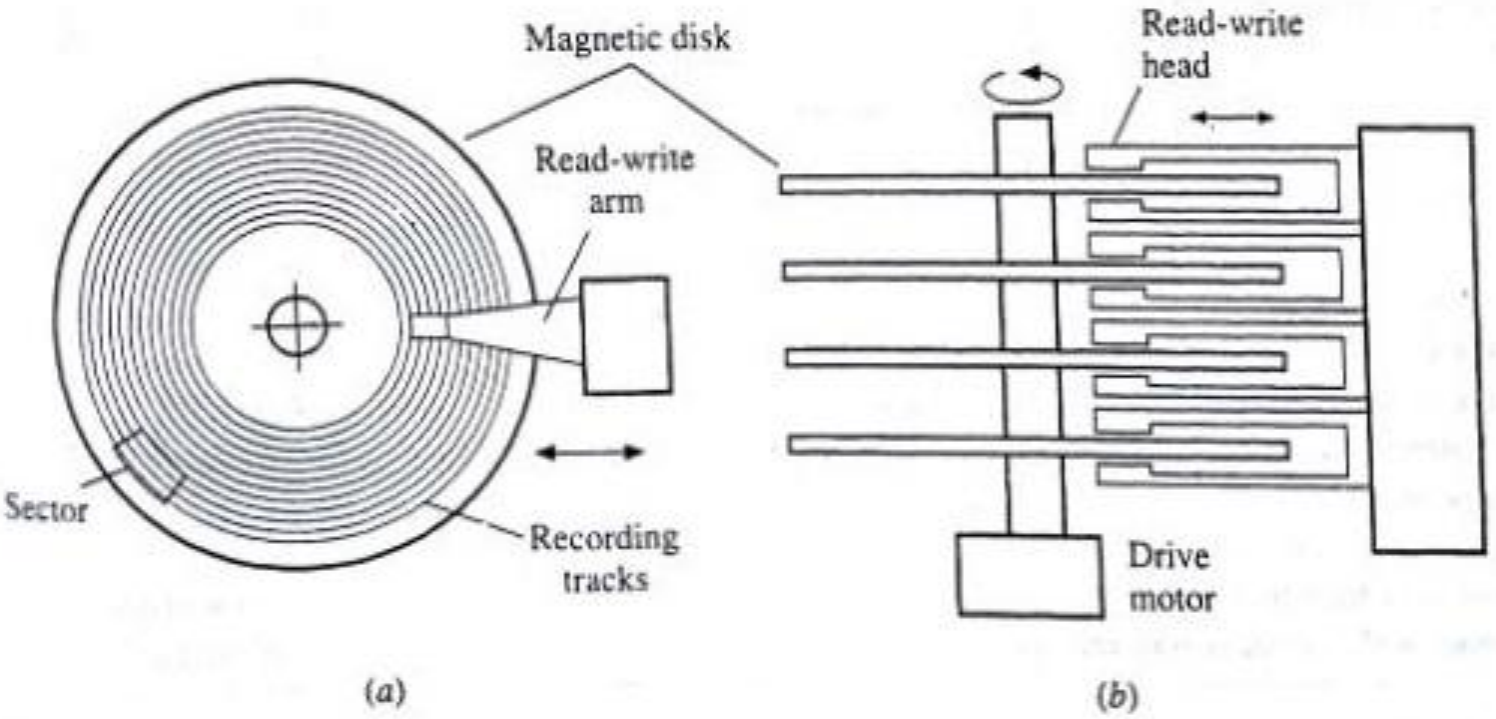


Figure 6.17
(a) Top view and (b) side view of a magnetic-disk drive unit.

SERIAL ACCESS MEMORIES : MAGENTIC TAPE MEMORIES

- Use – to provide backup storage for a computer system in the event of its hard disk subsystem.
- It stores binary digital information.
- The storage medium has as its substrate a flexible plastic tape, usually packaged in a small cassette or cartridge.
- Standard memory of the data-cartridge type containing a magnetic tape, which is 0.25 in (6.35 mm) wide and about 200 m long.
- Data is stored on a tape in parallel, longitudinal tracks.
- Tapes may have several hundred tracks.
- Data transfer takes place when the tape is moving at constant velocity relative to a read-write head; hence the maximum data transfer rate depends on the storage density along the tape and the tape's speed.
- The time to scan or rewind an entire tape is about a minute.

SERIAL ACCESS MEMORIES : MAGENTIC TAPE MEMORIES

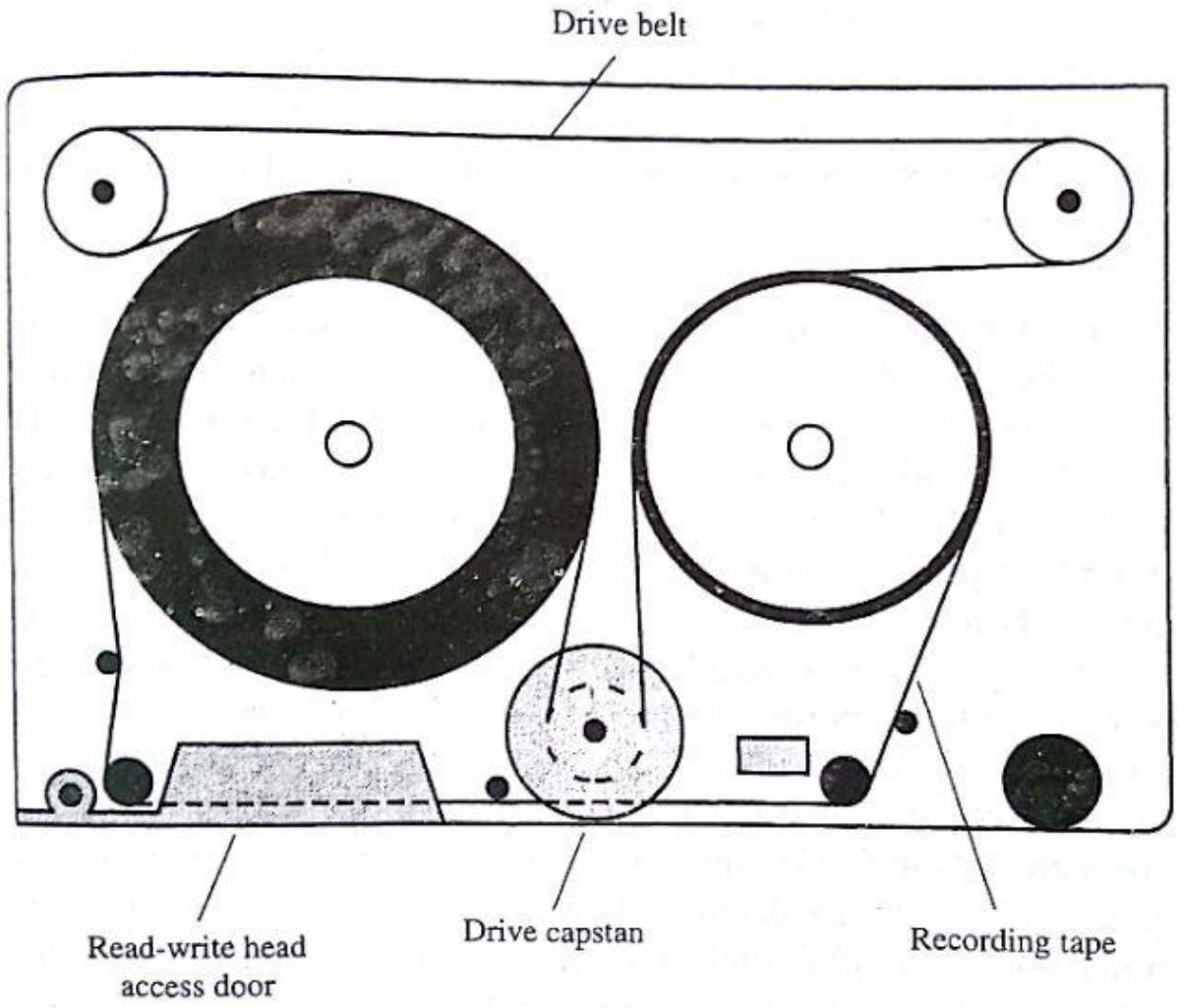


Figure 6.19
A magnetic tape cartridge.

SERIAL ACCESS MEMORIES : MAGENTIC TAPE MEMORIES

- Information stored on magnetic tapes is organized into blocks, usually of fixed length.
- A relatively large gap is inserted at the end of each block to permit the tape to start and stop between blocks.
- If the block length is bl and the interblock gap length is gl, then the **tape's (space) utilization u** is measured by,

$$u = \frac{bl}{bl + gl}$$

- Because of the interblock gaps and the time needed to start and stop the tape between accesses, the **effective data-transfer rate d_{eff}** seen by the user is less than the maximum rate d.
- Let t_D denote the time to scan a data block, let t_G be the time to scan an interblock gap, and t_{SS} be the time to start and stop the tape. Then,

$$d_{eff} = \frac{t_D d}{t_D + t_G + t_{ss}}$$

SERIAL ACCESS MEMORIES : MAGENTIC TAPE MEMORIES

➢If the block and gap sizes in bytes are bs and gs, respectively, then t_D=bs/d and t_G=gs/d, so this equation becomes,

$$d_{eff} = \frac{bs \bullet d}{bs + gs + t_{ss} \bullet d}$$

and the effective block access time t_B is $1/d_{eff}$.

SERIAL ACCESS MEMORIES : OPTICAL MEMORIES

- Optical disks employ optical or light-based techniques for data storage.
- Store binary information in concentric tracks on an electromechanically rotated disk.
- The information is read or written optically, however, with a laser replacing the read-write arm of a magnetic disk drive.
- Offer high storage capacities, but their access rates are generally less than those of magnetic disks.
- Read-only optical memories (CD-ROMs) are well developed, but low-cost read-write memories have proven difficult to build.
- CD-ROMs size is of 12 cm and can be mass produced at very low cost per disk by injection molding.
- Binary data is stored in the form of 0.1 μm wide **pits** and **lands** (nonpitted areas) in circular tracks on a plastic substrate.

SERIAL ACCESS MEMORIES : OPTICAL MEMORIES

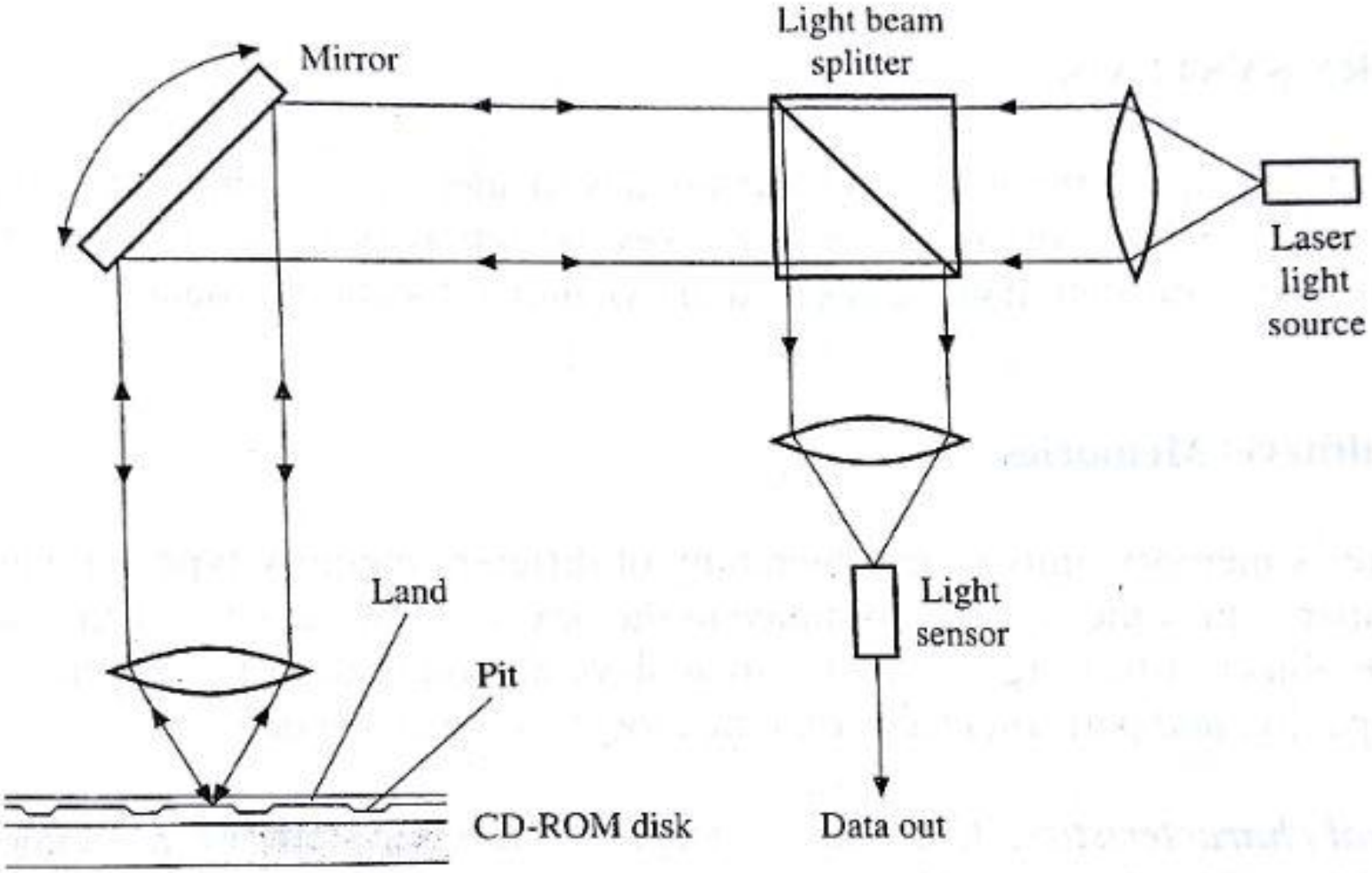


Figure 6.20
Optical readout mechanism for a CD-ROM.

SERIAL ACCESS MEMORIES : OPTICAL MEMORIES

- A laser beam scans the tracks and is reflected differently by the pits and lands.
- A mirror and lens system forms a read arm that can move back and forth across the tracks.
- The mirror can also be tilted slightly to provide fine tracking adjustments,
- The reflected light from the laser is picked up by a sensor and decoded to extract the stored information, which is then converted to electronic form for further processing.
- A standard 12cm CD-ROM has a capacity of around 600 MB, which is enough to store some 240,000 pages of printed text – a large encyclopedia, or instance.
- Access time is about 100 ms, and data is transferred from the disk at a rate of 3.6 MB/s.
- CD-recordable (CD-R)
- CD-rewritable (CD-RW) – They employ a laser to create (burn)pits on the surface of blank disks.

SERIAL ACCESS MEMORIES : OPTICAL MEMORIES

- **Digital video disk (DVD)** introduced in both read-only and read-write forms.
- With two recording surfaces and one or two storage layers per surface, a DVD can have a capacity as high as 16 GB.
- A **magneto-optical disk memory** uses rotating disks that store information in magnetic form but are accessed by a laser beam similar to that in a CD-ROM drive.
- It has a magnetizable surface coating whose direction of magnetization can be polarized (up or down corresponding to 0 or 1).
- A cell is **read** by bouncing a laser beam off it.
- The beam's angle of polarization is affected by the cell's magnetization direction, a phenomenon known as the **Kerr effect**.
- The slight change in the polarization angle of the reflected laser beam is sensed and decoded by the read mechanism.

SERIAL ACCESS MEMORIES : OPTICAL MEMORIES

- **Writing** is accomplished by using the laser beam to briefly heat a chosen cell above a specific temperature (**the Curie temperature** of the magnetic medium), at which point the cell's magnetic coercivity becomes zero, making the cell sensitive to external magnetic fields.
- An electromagnetic coil placed below the rotating disk then supplies a magnetic field of the required direction.
- The heated cell captures the magnetic field's direction which is retained after the cell cools below its Curie temperature.

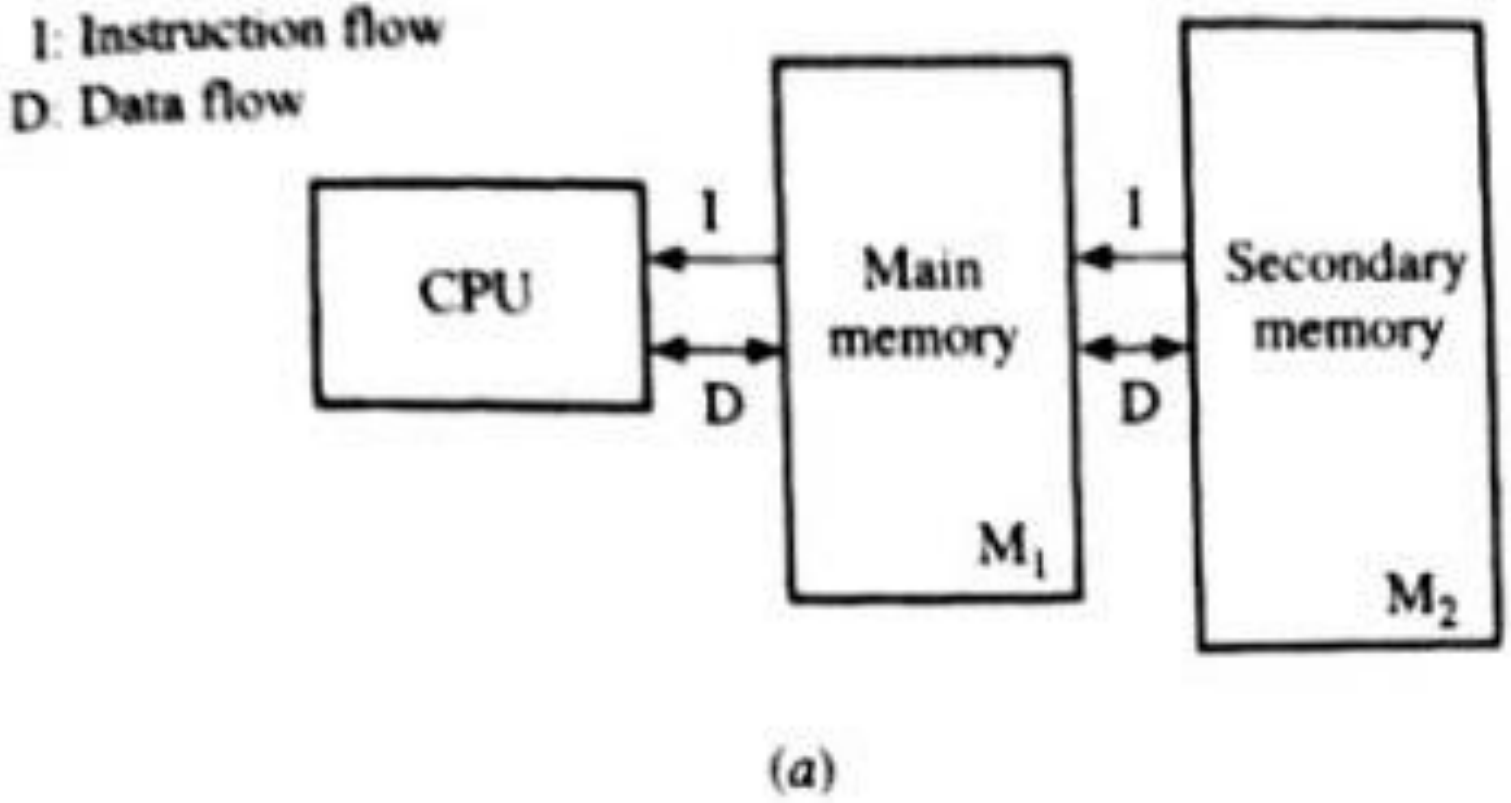
MEMORY SYSTEMS : MULTILEVEL MEMORIES

- Objective of the memory hierarchy is to achieve a good trade-off between cost, storage capacity and performance.
- Consider a general n-level system of n memory types (M_1, M_2,…,M_n).
- Semiconductor SRAMs for cache memory, semiconductor DRAMs for main memory and magnetic disk units for secondary memory.
- Figure 6.21b add a cache of a type called a **split cache**, since it has separate areas for storing instructions (the I-cache) and data cache (the D-cache).
- The following relations normally hold between adjacent memory levels M_i and M_{i+1} in a memory hierarchy,

Cost per bit	$c_i > c_{i+1}$
Access time	$t_{Ai} < t_{Ai+1}$
Storage capacity	$S_i < S_{i+1}$

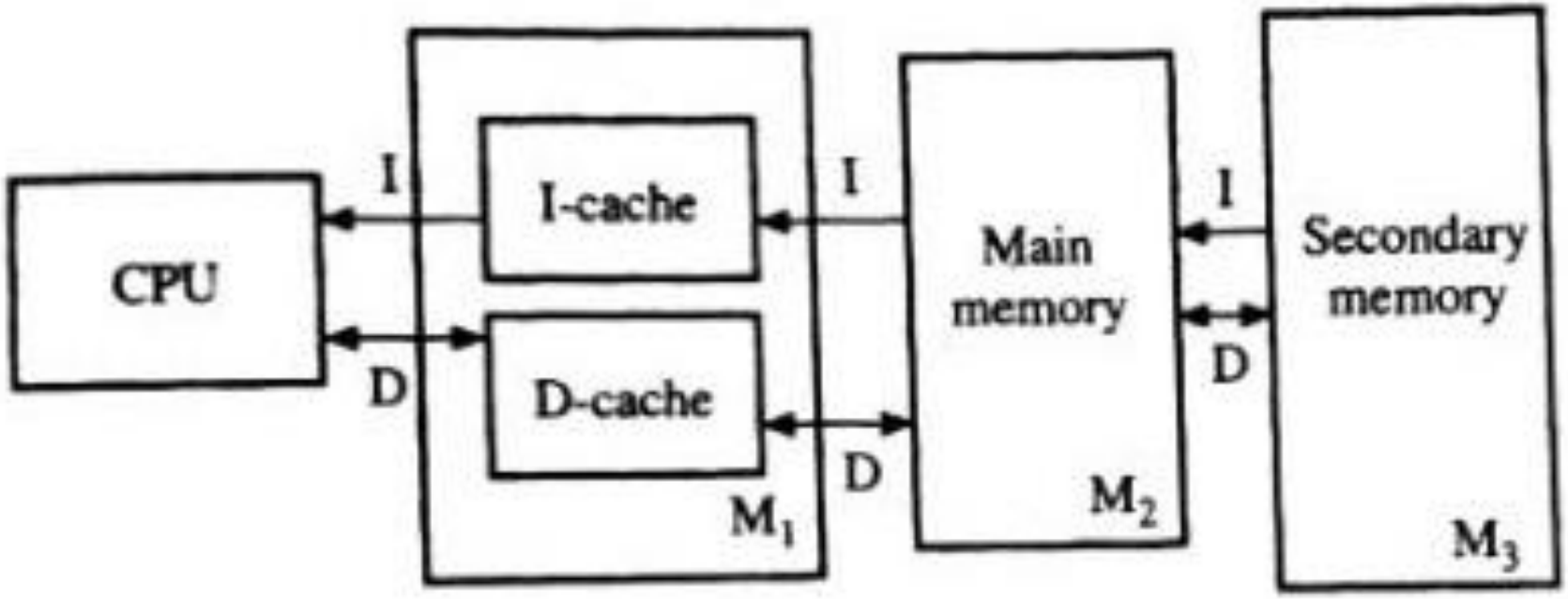
- The differences in cost, access time and capacity between M_i and M_{i+1} can be several orders of magnitude.

MEMORY SYSTEMS : MULTILEVEL MEMORIES



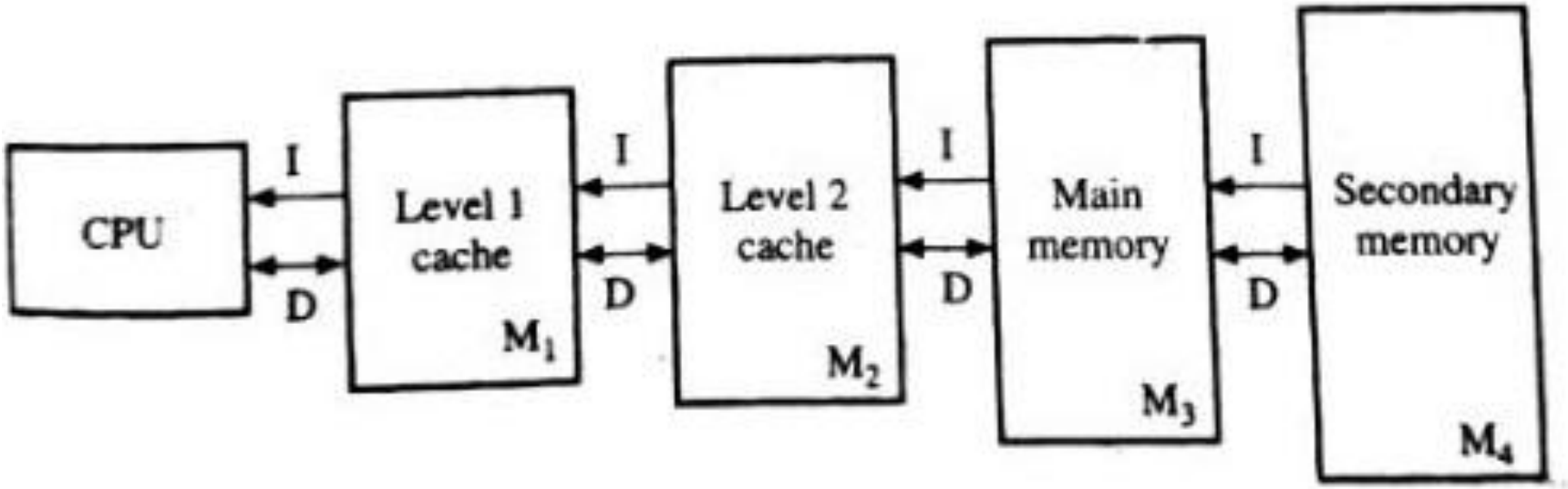
(a)

MEMORY SYSTEMS : MULTILEVEL MEMORIES



(b)

MEMORY SYSTEMS : MULTILEVEL MEMORIES



(c)

Figure 6.21
Common memory hierarchies with (*a*) two, (*b*) three, and (*c*) four levels.

MEMORY SYSTEMS : MULTILEVEL MEMORIES

- The CPU and other processors can communicate directly with M_1 only, M_1 can communicate with M_2, and so on.
- Consequently, for the CPU to read information held in some memory level M_i requires a sequence of I data transfers for the form,

$$M_{i-1} := M_i;\ M_{i-2} := M_{i-1};\ \ldots,\ M_1 := M_2;\ \text{CPU} := M_1$$

- In general, all the information stored in M_i at any time is also stored in M_{i+1}, but not vice versa

MULTILEVEL MEMORIES : CACHE AND VIRTUAL MEMORY

- The cache and main memory resemble a single memory M to the software being executed.
- Main and secondary memory form another distinct two-level sub hierarchy.
- This interaction is managed by the operating system.
- The term **virtual memory** is applied when the main and secondary memories appear to a user program like a single, large, and directly addressable memory.
- Three reasons for using virtual memory are,

1. To free user programs from the need to carry out storage allocation and to permit efficient sharing of the available memory space among different users.
2. To make programs independent of the configuration and capacity of the physical memory present for their execution.
3. To achieve the very low access time and cost per bit that are possible with a memory hierarchy.

MULTILEVEL MEMORIES : CACHE AND VIRTUAL MEMORY

- A memory system is addressed by a set of **logical** or **virtual addresses** derived from identifiers explicitly or implicitly specified in an object program.
- A set of **physical** or **real addresses R** identifies the fixed physical storage locations in each memory unit M_i.
- An efficient and flexible mechanism to implement address mappings of the form f:V->R is the key to successful design of a multilevel memory.

MULTILEVEL MEMORIES : LOCALITY OF REFERENCE

- The predictability of memory addresses depends on the characteristic of computer programs called **locality of reference**.
- Suppose a request is made for a one-word instruction I stored at address A, but this address is currently assigned to $M_i \neq M_1$.
- The instruction most likely to be required next by the CPU is the one immediately following I whose address is A+1.
- Instead of simply transferring I to M_1 when it is referenced, it is more efficient to transfer a block of consecutive words containing I.
- It is done by subdividing the information stored in M_i into pages, each containing S_{pi} of consecutive words.
- Information is then transferred one page or S_{pi} words at a time between M_i and M_{i-1}
- This method is called **spatial locality of reference**.

MULTILEVEL MEMORIES : LOCALITY OF REFERENCE

- The second factor in locality of reference is the presence of loops in programs.
- Instructions in a loop, even when they are far apart in spatial terms, are executed repeatedly, resulting in a high frequency of reference to their addresses –**Temporal locality of reference**.
- When a loop is being executed, it is desirable to store the entire loop in M_1 if possible.

MULTILEVEL MEMORIES : ADDRESS TRANSLATION

- A central issue in managing (M_1, M_2) or any multilevel memory, is to make it appear to its user like a single, fast memory of high capacity.
- Memory management system needs to perform the following task,

1. Translation of memory addresses from the virtual addresses encountered in program execution to the real addresses that identify physical storage locations.
2. Dynamic (re)allocation or swapping of information among the different memory levels so that stored items reside in the fastest level before they are needed.

- The set of abstract locations that a program Q can reference is Q's **virtual address space V**.
- To execute Q on a particular computer, its virtual addresses must be mapped onto the real address space R, defined by the addressable (external) memory M that is physically present in the computer.
- This process is called **address translation** or **address mapping**.

MULTILEVEL MEMORIES : ADDRESS TRANSLATION

- The **real address space R** is a linear sequence of numbers 0, 1, 2, …, n-1 corresponding to the addressable word locations in M.
- It is convenient to identify M with main memory, while noting that R is usually distributed over several levels of memory hierarchy, including the cache and the level labeled main memory.
- Address translation can be viewed abstractly as a function f : V -> R.
- Address assignment and translation is carried out at various stages in the life of the program, specifically,

1. By the programmer while writing the program
2. By the compiler during program compilation
3. By the loader at initial program-load time
4. By run-time memory management hardware and/or software.

- In modern computers, programmers normally deal only with virtual address.

MULTILEVEL MEMORIES : ADDRESS TRANSLATION

- If the program is sufficiently simple, the compiler can completely map virtual addresses to real addresses.
- Address translation can also be completed when the program is first loaded for execution. **– static translation**.
- Since the real address space of the program is fixed for the duration of execution.
- It is often desirable to vary the virtual space of a program dynamically during execution – **dynamic translation**.

ADDRESS TRANSLATION : BASE ADDRESSING

- An executable program comprises a set of instruction and data blocks each of which is a sequence of words to be stored in consecutive memory locations during execution.
- A word W within a block has its own **effective address A_{eff}**, which the CPU must know to access W.
- W is also specified by the address C, called the **base address**, of the block that contain it, along with W's **relative address** or **displacement address D** (**offset or index**) within the block.

$$A_{eff} = B+D$$

- Often the address is designed so that B supplies the high order bits of A_{eff} while D supplies the lower order bits,

$$A_{eff} = B.D$$

- A_{eff} is formed simply by concatenating B and D, a process that does not significantly increase the time for address generation.

ADDRESS TRANSLATION : BASE ADDRESSING

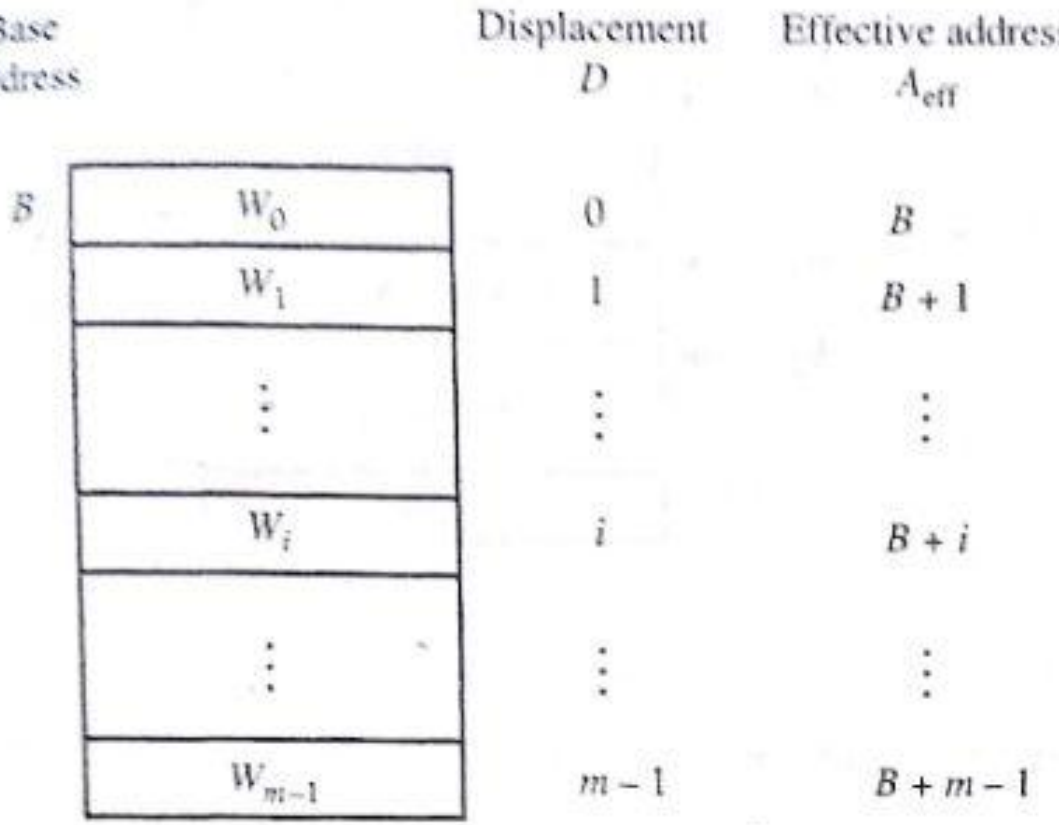


Figure 6.24
Block of m words with (base) address B.

ADDRESS TRANSLATION : BASE ADDRESSING

- A simple way to implement static and dynamic address mapping is to put base addresses in a **memory map** or **memory address table** controlled by the memory management system.
- The table can be stored in memory, CPU registers or in both.
- Blocks are easily relocated in memory by manipulating their base addresses.
- Suppose two blocks are allocated to main memory M.
- It is desired to load a third block K_3 into M; however, a contiguous empty space, or "hole", of sufficient size is unavailable.
- A solution to this problem is to move block K_2, by assigning it a new base address B_2' and reloading it into memory.
- This creates a gap into which block K_3 can be loaded by assigning to it an appropriate base address.
- The block can be permitted to read from certain locations, but writing outside its assigned area must be prevented – specifying the highest address L_i (**limit address**).

ADDRESS TRANSLATION : BASE ADDRESSING

- The base address B_i and the limit address L_i are stored in the memory map.
- Every real address A_r generated by the block is compared to B_i and L_i; the memory access is completed if and only if the condition,

$$B_i \leq A_r \leq L_i$$

Is satisfied.

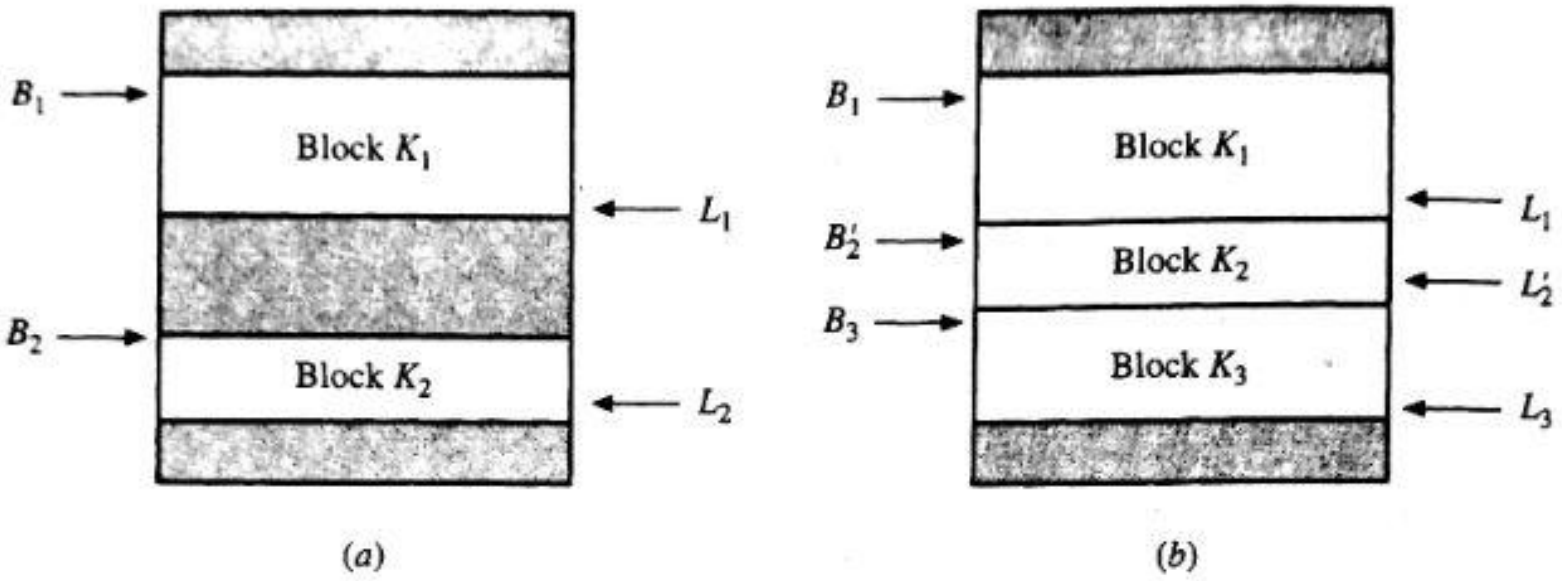


Figure 6.25
Relocation of blocks in memory using base and limit addresses.

ADDRESS TRANSLATION : TRANSLATION LOOK-ASIDE BUFFER

- Address translation using translation look-aside buffer.

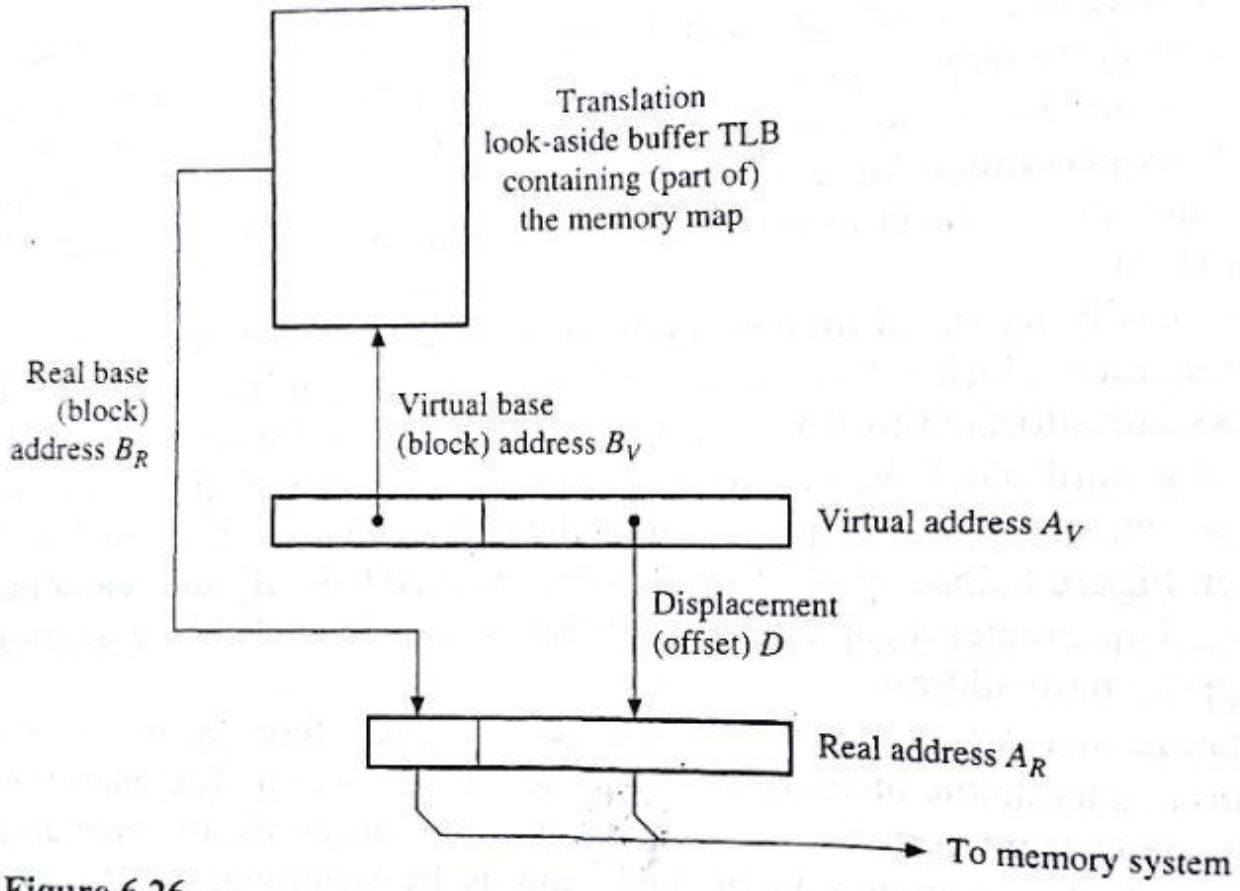


Figure 6.26
Structure of a dynamic address-translation system.

ADDRESS TRANSLATION : TRANSLATION LOOK-ASIDE BUFFER

- The input address A_v is a virtual address consisting of a (virtual) base address B_v concatenated with a displacement D.
- A_v contains an effective address computed in accordance with some program-defined addressing mode (direct, indirect, indexed and so on.) for the memory item being accessed.
- The real address $B_R=f(B_v)$ assigned to B_v is stored in a memory map somewhere in the memory system.
- To speed up the mapping process, part (or occasionally all) of the memory map is placed in a small high-speed memory in the CPU called **translation look-aside buffer (TLB)**.
- The TLB's input is thus the base address part B_v of A_v; its output is the corresponding real base address B_R. This address is then concatenated with D part of A_v to obtain the full physical address A_R.

ADDRESS TRANSLATION : TRANSLATION LOOK-ASIDE BUFFER

- If the virtual address B_v is not currently assigned to the TLB, then the part of the memory map that contain B_v is first transferred from the external memory into the TLB.
- Hence the TLB itself forms a cache like level within a multilevel address storage system for memory maps.
- For this reason, the TLB is sometimes referred to as an **address cache**.

MEMORY ALLOCATION

- Placement of blocks of information in a memory system is called Memory allocation.
- **Replacement policy:** method of selecting the part of the memory (M_1) in which an incoming block K is to be placed.
- Successful memory allocation methods results in high hit ratio and a lower average access time.
- Information needed for allocation within a two-level hierarchy (M_1, M_2):
  - **Occupied space list for M_1:** Each entry of this list specifies a block name, the (base) address of the region it occupies, and, if variable, the block size.
  - **Available space list for M_1:** each entry of the list specifies the address of an unoccupied region and if necessary, its size.
  - **Directory of M_2:** this list specifies the units that contain the directories for all the blocks associated with the current programs.

MEMORY ALLOCATION

- **Nonpreemptive allocation:**
  - Suppose a block K_i of n_i words is to be transferred from M_2 to M_1.
  - If none of the available blocks already occupying M_1 can be overwritten or moved by K_i then it is necessary to find or create an available region n_i or words to accommodate K_i.

Widely used algorithms for nonpreemptive allocation of variable-sized blocks-unpaged segments,

- **First fit:** This method scans the memory map sequentially until an available region R_j of n_i or more words is found, where n_i is the size of the incoming block K_i. It then allocates K_i to R_j.
- **Best fit:** it requires searching the memory map completely and assigning K_i to a region of $n_j \geq n_i$ words such that $n_j - n_i$ is minimized.

MEMORY ALLOCATION

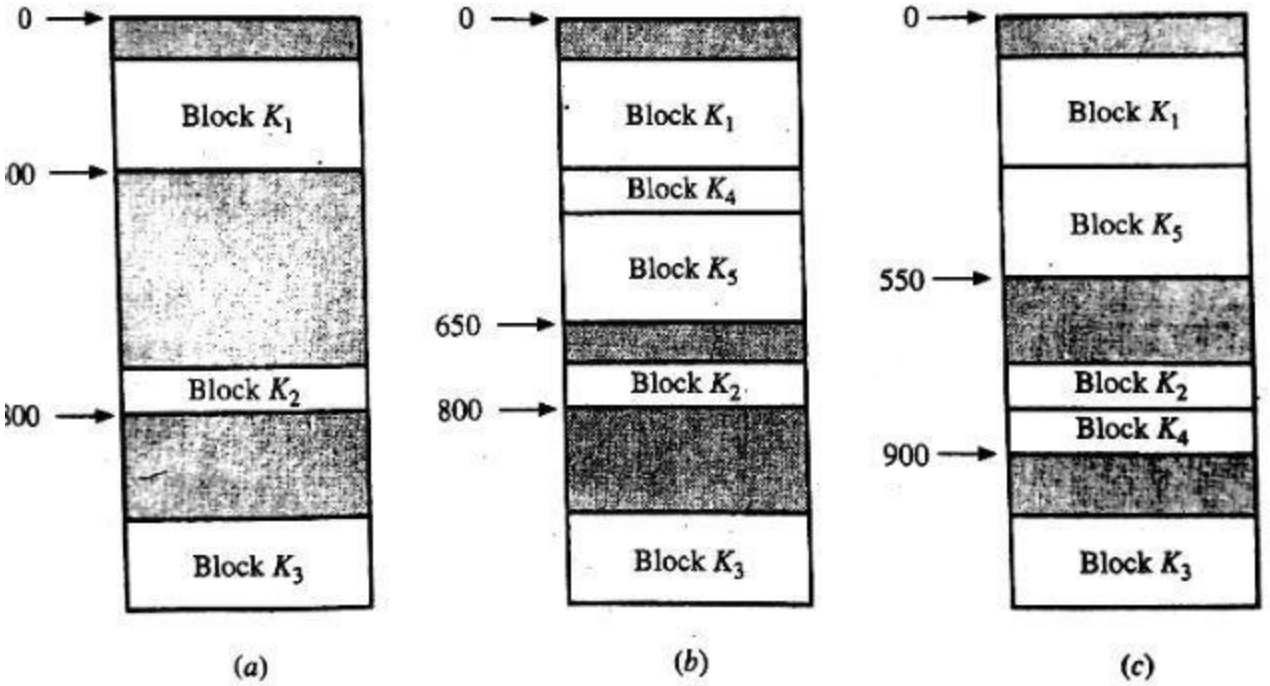


Figure 6.34
(*a*) Initial memory state; (*b*) allocation of K_4 and K_5 by first fit; (*c*) allocation of K_4 and K_5 by best fit.

Region address	Size (words)
0	50
300	400
800	200

MEMORY ALLOCATION

- **Preemptive allocation:**

- Much more efficient use of the available memory space is possible if the occupied space can be reallocated to make room for incoming blocks.
- The blocks already in M_1 can ne relocated within M_1 to create a gap large enough for the incoming block.
- One or more occupied regions can be made available by deallocating the blocks they contain. This method requires a rule- a replacement policy- for selecting blocks to be deallocated and replaced.

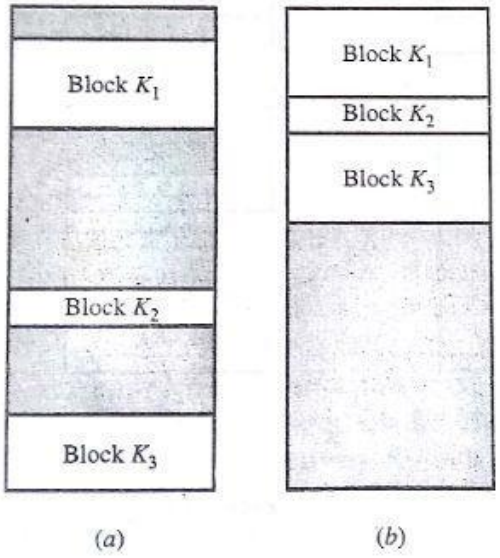


Figure 6.35
Memory allocation (*a*) before and (*b*) after compaction.

MEMORY ALLOCATION

- **Replacement policies:**
- This involves preempting a region R occupied by block K and allocating it to an incoming block K'.
- Main goal in choosing a replacement policy is to maximize the hit ratio of the faster memory M_1 or equivalently minimize the number of times a referenced block is not in M_1 a condition called a memory fault or miss.
- Hit ratio tend increase if the time intervals between successive memory faults are maximized.
- Optimal replacement strategy.
- First-in-first-out (FIFO).
- Least recently used (LRU).

MEMORY ALLOCATION

Consider a paging system in which M_1 has a capacity of 3pages. The execution of a program Q requires reference to 5 distinct pages P_i, where i=1,2,3,4,5, and I is the page address. The page address stream formed by executing Q is

2 3 2 1 5 2 4 5 3 2 5 2

Which means that the first page reference is P_2 the second is P_3 and so on.

MEMORY ALLOCATION

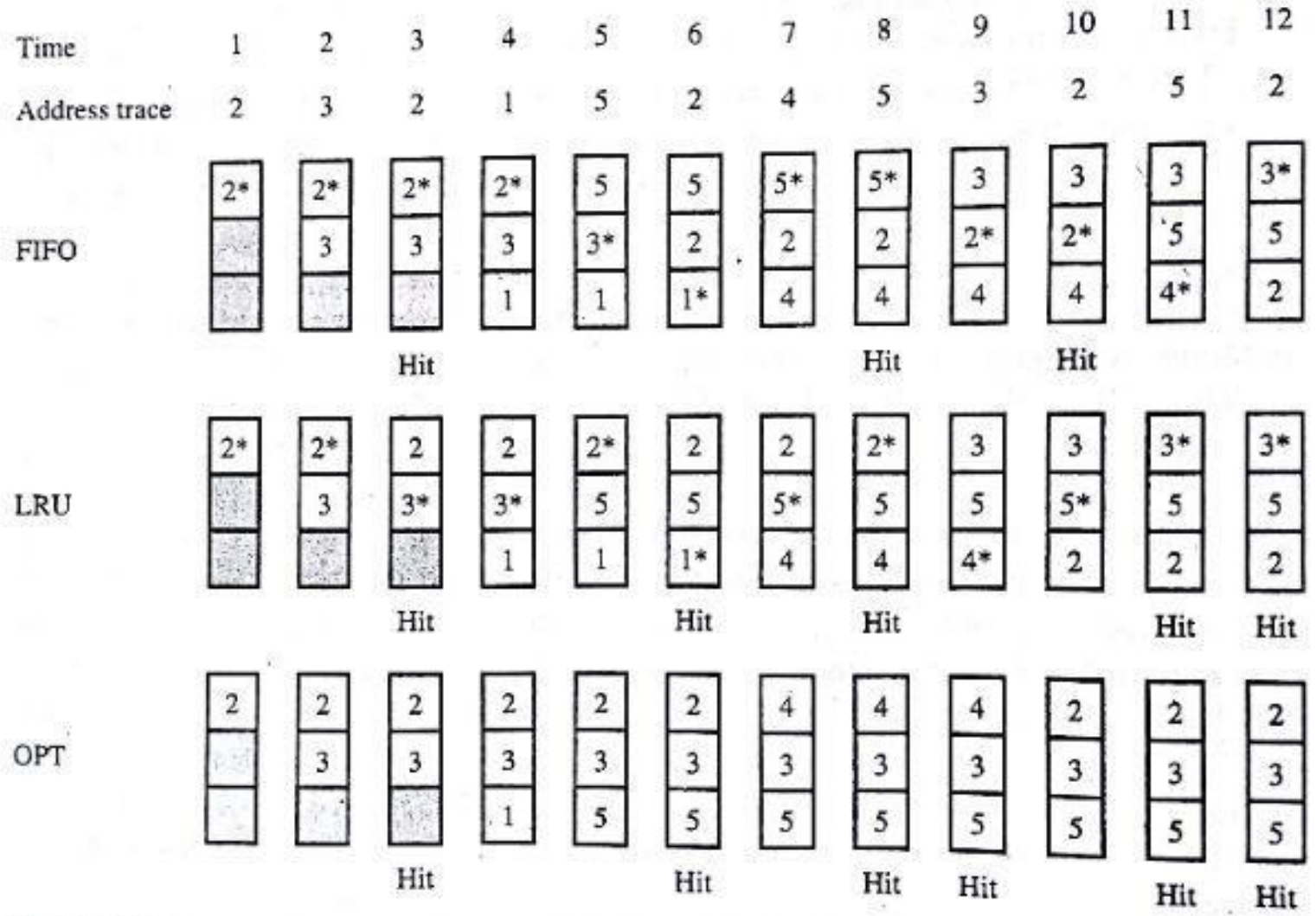


Figure 6.36
Action of three replacement policies on a common address trace.

MEMORY ALLOCATION

* Denotes the next block to be selected for replacement.

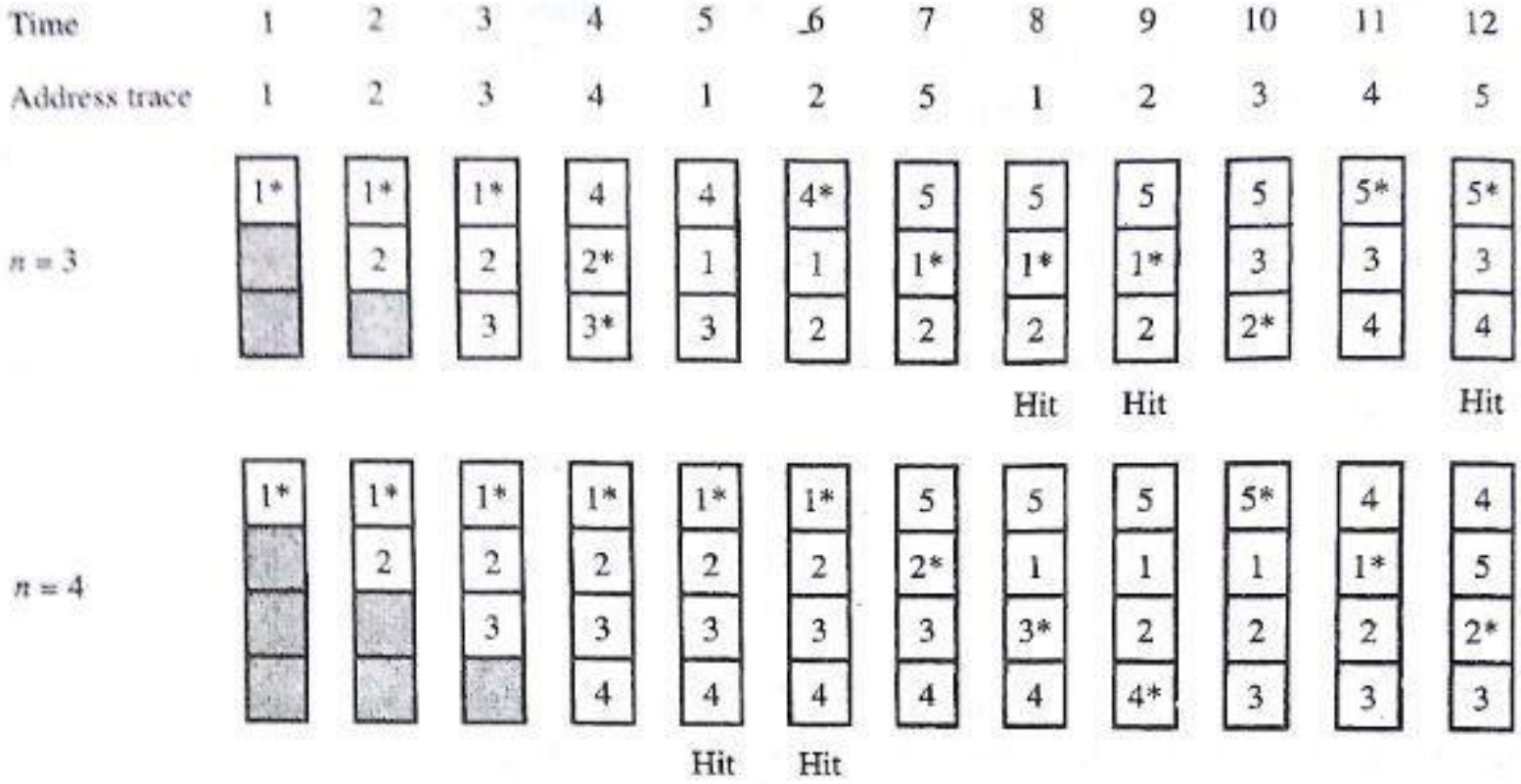


Figure 6.37
FIFO replacement with two different memory capacities.

MEMORY ALLOCATION

Determination of hit ratios with LRU replacement

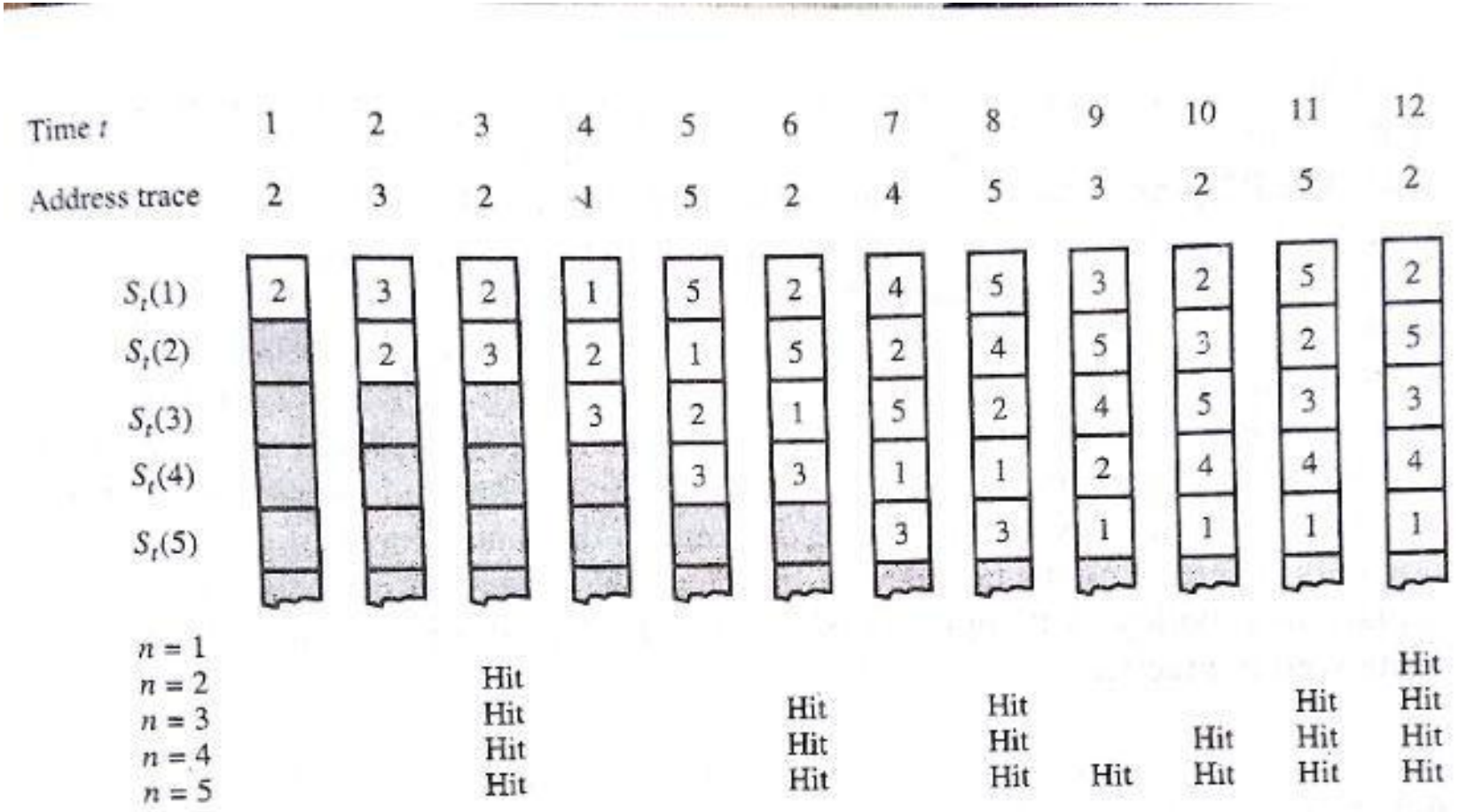


Figure 6.38
Stack processing of a page address trace using LRU.

CACHES

- It refers to the fast intermediate memory within a large memory system.
- It serves as a buffer between a CPU and its main memory.

Two-level hierarchy (M_{i-1}, M_i)	Cache–main memory (M_1, M_2)	Main–secondary memory (M_2, M_3)
Typical access time ratios $t_{A_i}/t_{A_{i-1}}$	5/1	1000/1
Memory management system	Mainly implemented by hardware	Mainly implemented by software
Typical page size	8 B	4 KB
Access of processor to second level M_i	Processor has direct access to M_2	All access to M_3 is via M_2

Figure 6.39
Major differences between cache–main and main–secondary-memory hierarchies.

CACHES

Cache Organization:

- Memory words are stored in a cache data memory and are grouped into small pages called cache blocks or lines.
- Contents of the cache memory block are the copies of the content of the main memory block.
- The collection of tag addresses currently assigned to the cache, which can be noncontiguous, is stored in a special memory, the cache tag memory or directory.

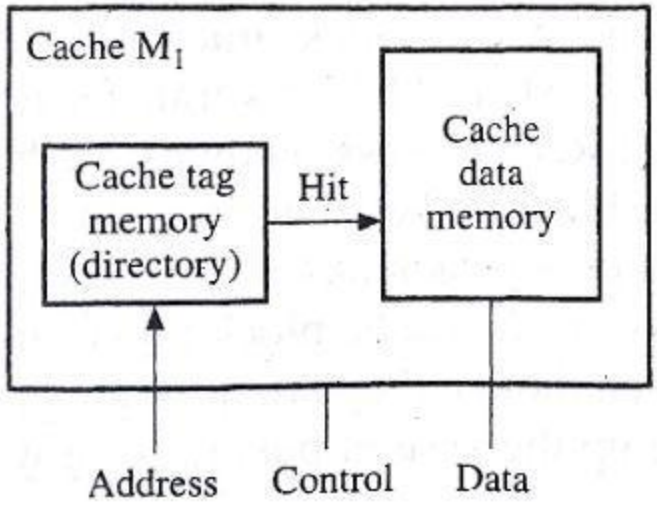


Figure 6.40
Basic structure of a cache.

CACHES

Two system organization for caches;

Look aside:

1.CPU initiates a memory access by placing a address A_i on the memory address bus at the start of read (load) or write (store) cycle.

2.Cache M_1 compares A_i to the tag address currently residing in its tag memory.

3.If match is found in M_1 cache hit occurs.

4.The access completed by a read or write operation, main memory M_2 is not involved.

5.If cache miss, a block (line) of data B_j that includes the target address A_i is transferred from M_2 to M_1.

Look through:

➢CPU communicates with the cache via a separate (local) bus that is isolated from the main system bus.

CACHES

➢CPU does not send all memory request to main memory it does so only after a cache miss.

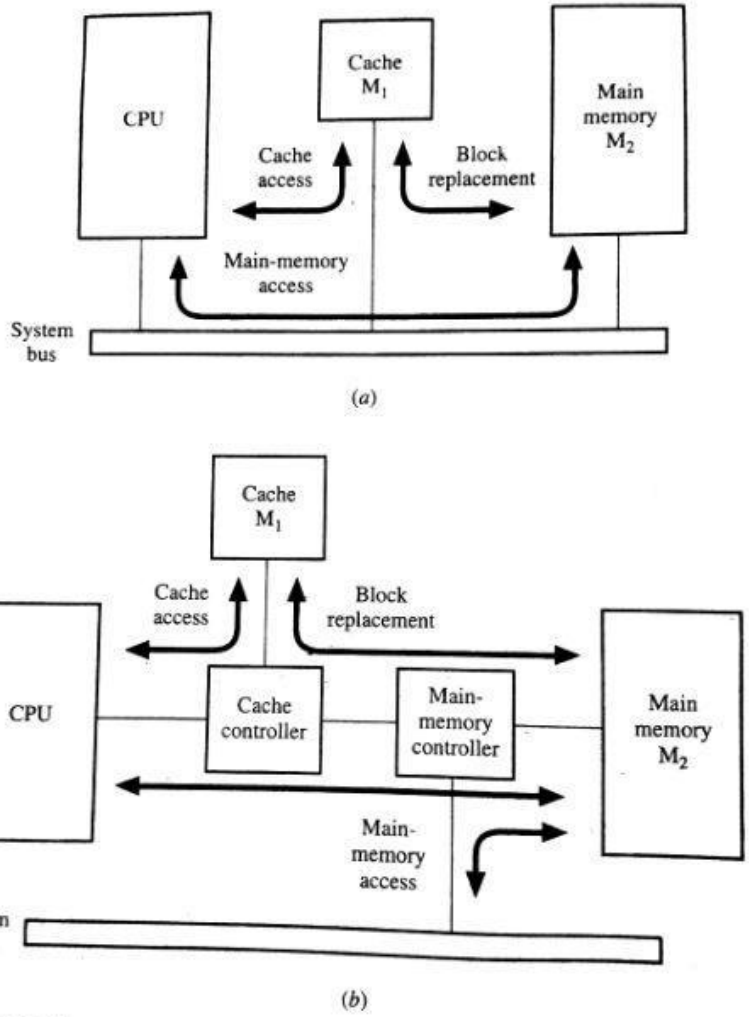


Figure 6.41
Two system organizations for caches: (*a*) look-aside and (*b*) look-through.

CACHES

Cache operation:

➢Memory address 12 bit, 10 higher order bits=tag or block address and 2 lower order bits defines the displacement address within the block.

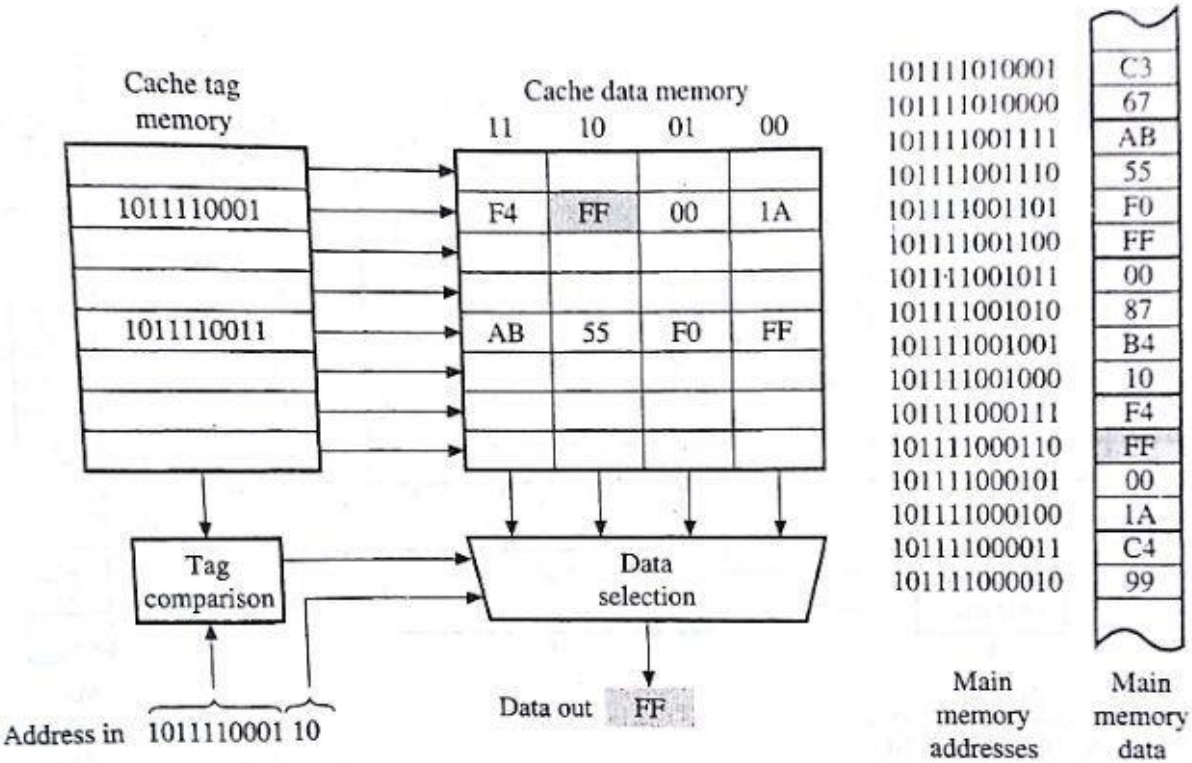


Figure 5.42
Cache execution of a read operation.

CACHES

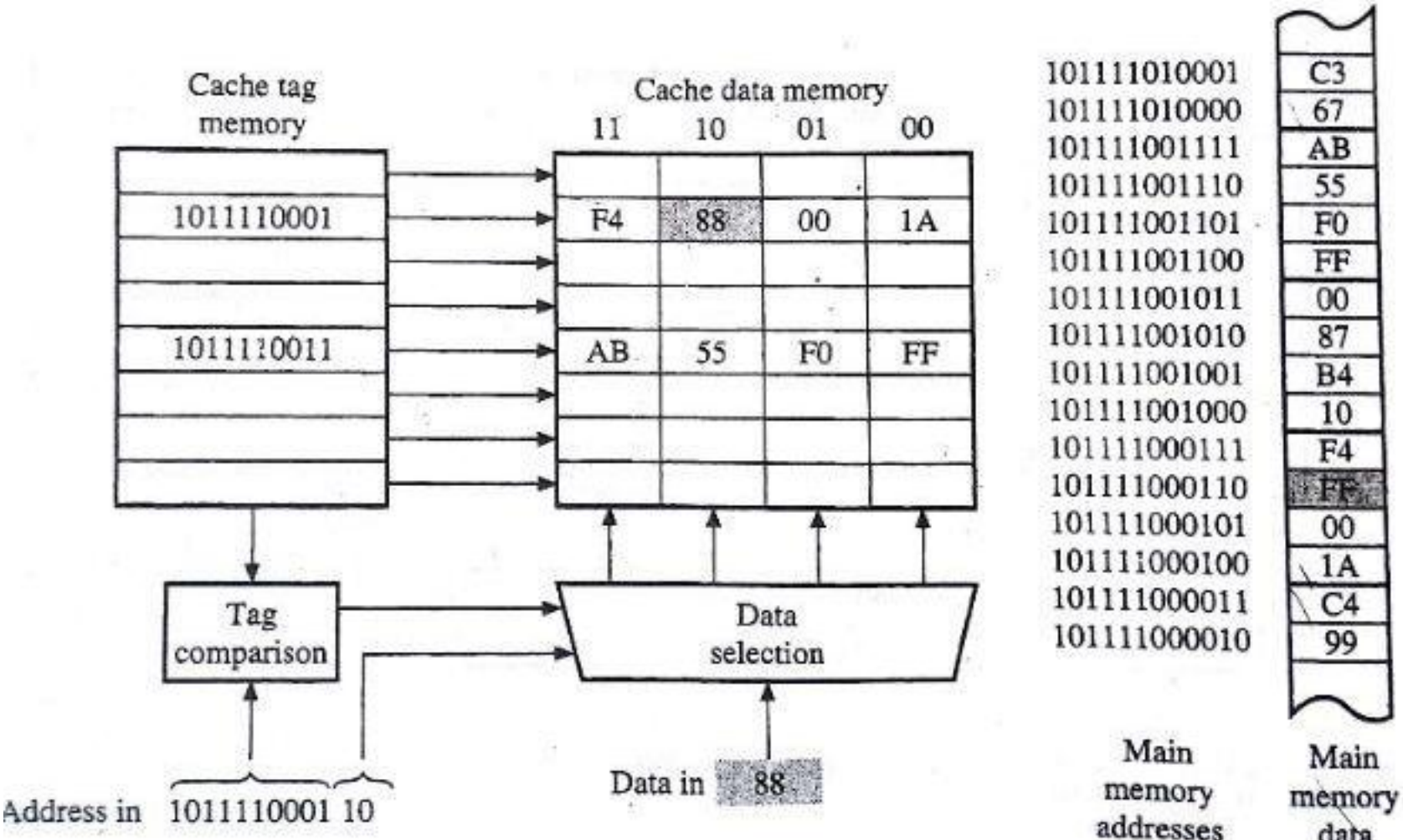


Figure 6.43
Cache execution of a write operation.

ADDRESS MAPPING

- Possible methods for specifying where memory blocks are placed in the cache.
- Consider a cache consisting of 128 blocks of 16 words each, for a total of 2048 (2K) words.
- Assume main memory is addressable by a 16-bit address.
- Total number of addressable words = 2^{16} = 65536 words (64 K).
- No of blocks of 16 words each in main memory = 65536/16 = 4096 blocks.
- For simplicity, we will assume that consecutive addresses refer to consecutive words.
- Three types of Address Mapping techniques:

1. Direct Mapping
2. Associative Mapping
3. Set-Associative Mapping

ADDRESS MAPPING – DIRECT MAPPING

- Block j of the main memory maps onto block j modulo 128 of the cache.
- One of the main memory blocks 0, 128, 256,… is loaded in the cache, it is stored in cache block 0.
- Blocks 1, 129, 257, … are stored in cache block 1, and so on.
- Disadvantage : 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.

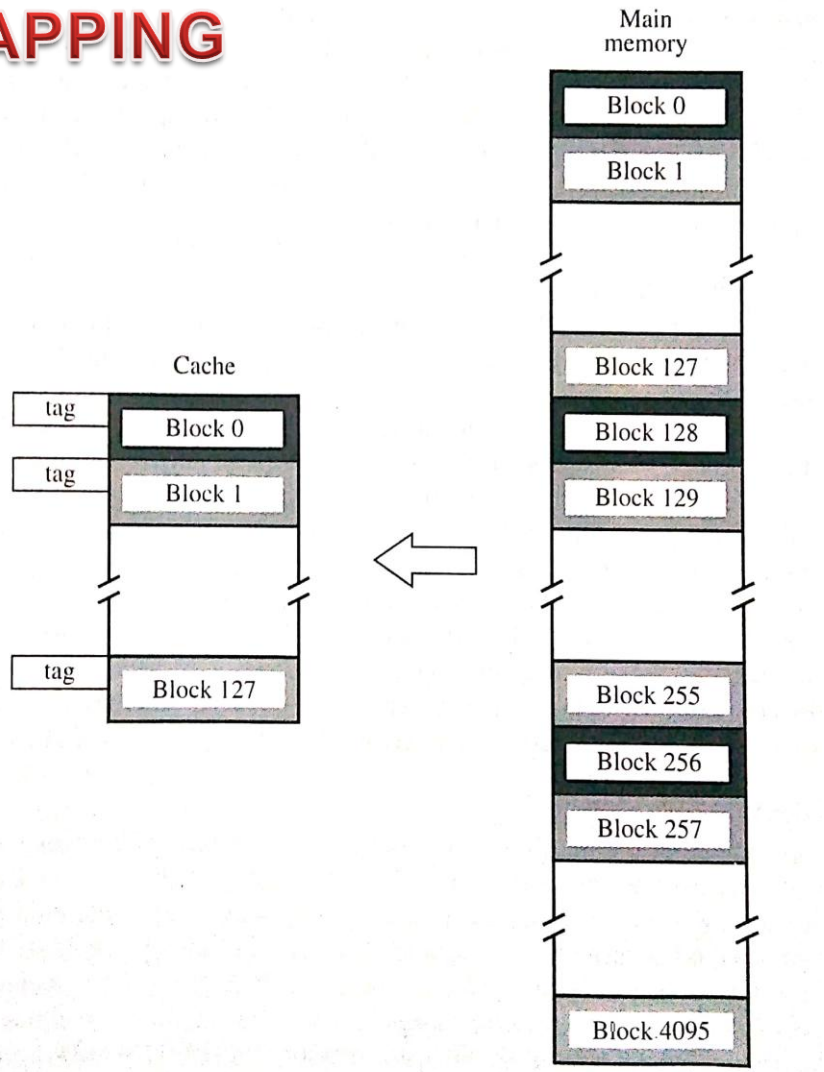


Figure 5.15 Direct-mapped cache.

ADDRESS MAPPING – DIRECT MAPPING

- Placement of a block in the cache is determined from the memory address.

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.
- They identify which of 32 blocks (4096/128) that are mapped into this cache position are currently resident in the cache.

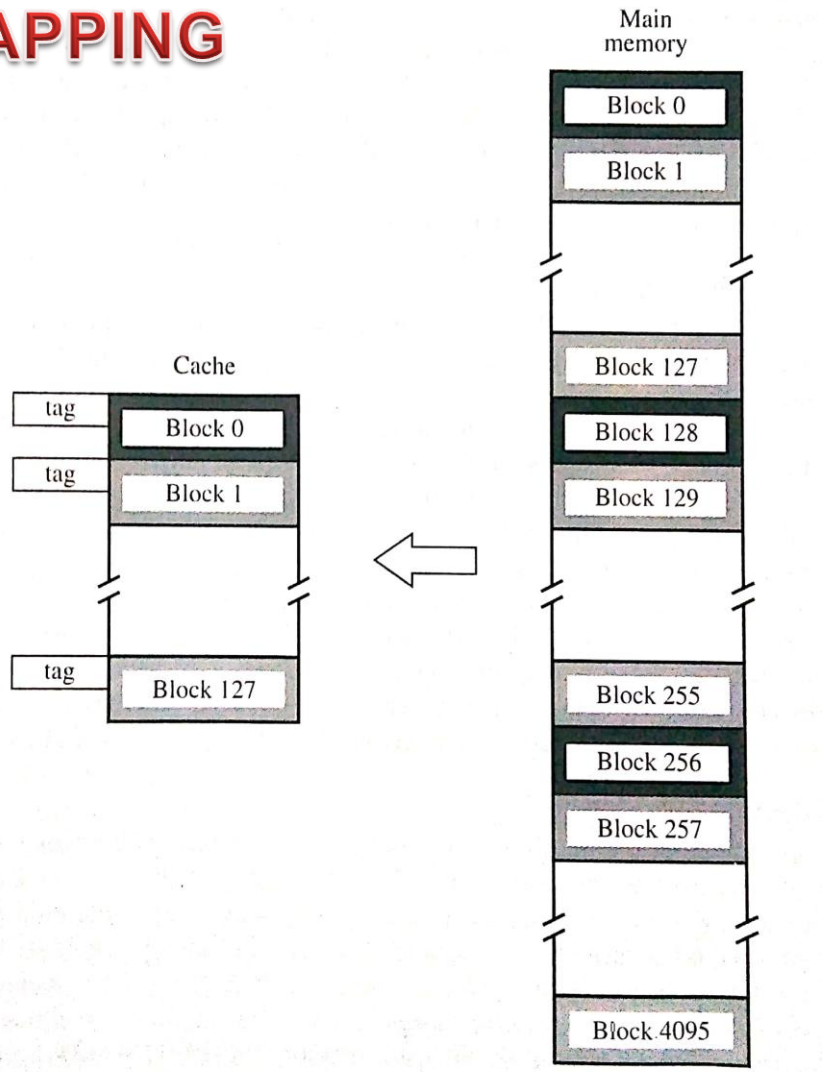


Figure 5.15 Direct-mapped cache.

ADDRESS MAPPING – DIRECT MAPPING

- As execution proceeds, the 7-bit cache block field of each address generated by the processor points to a particular block location in the cache.
- The high-order 5 bits of the address are compared with the tag bits associated with that cache location.
- If they match, then the desired word is in that block of the cache.
- If there is no match, then the block containing the required word must first be read from the main memory and loaded into the cache.
- Advantage: Easy to implement, but it is not flexible.

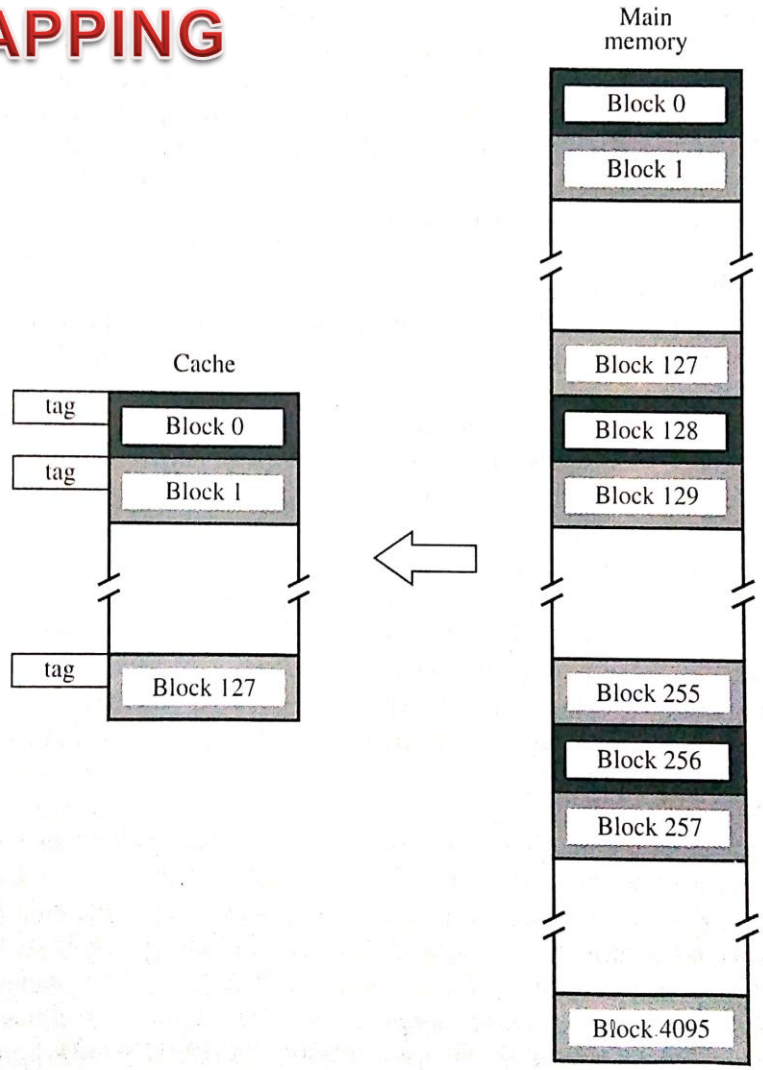


Figure 5.15 Direct-mapped cache.

ADDRESS MAPPING – ASSOCIATIVE MAPPING

- Flexible mapping method.
- Main memory block can be placed into any cache block location.
- 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.
- It gives complete freedom in choosing the cache location in which to place the memory block.

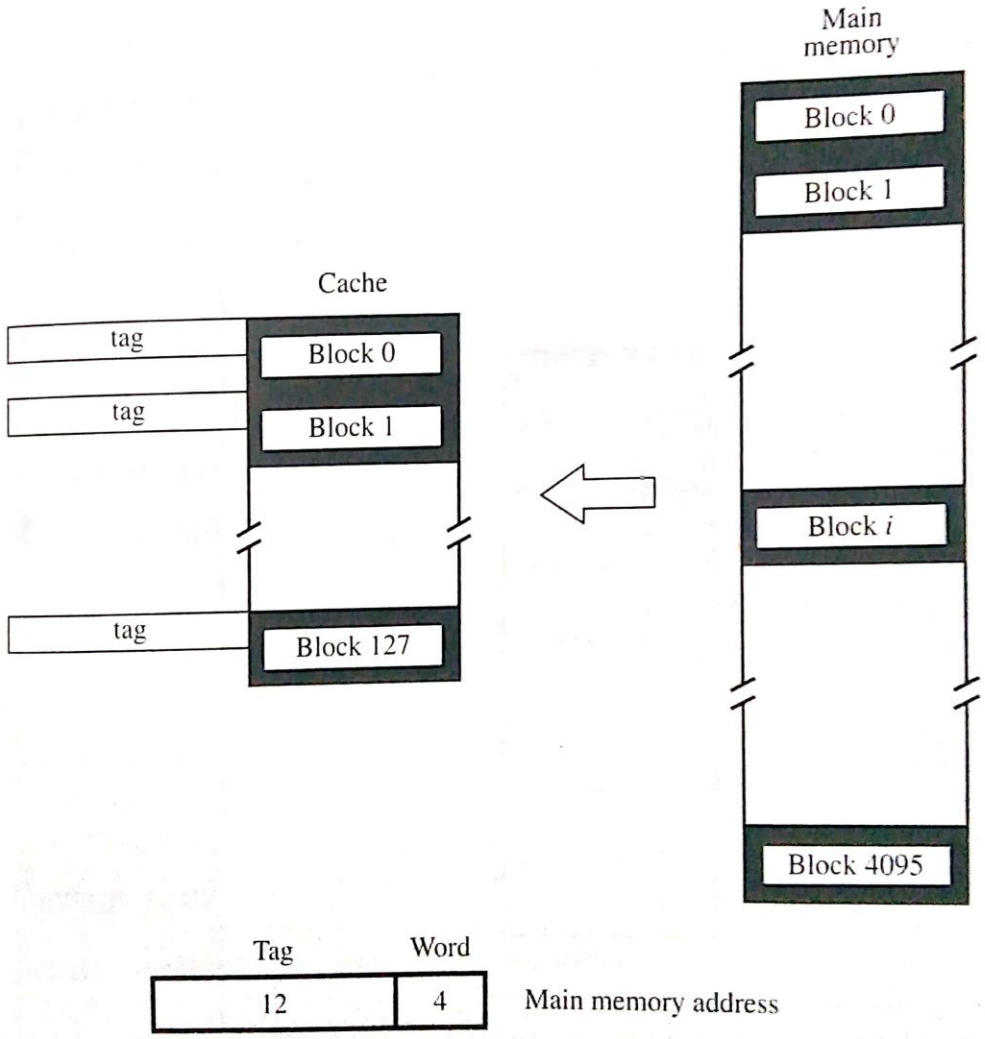


Figure 5.16 Associative-mapped cache.

ADDRESS MAPPING – ASSOCIATIVE MAPPING

- A new block that has to be brought into the cache has to replace (eject) an existing block only if the cache is full.
- Disadvantage : The cost of an associative cache is higher than the cost of a direct-mapped cache because of the need to search all 128 tag pattern to determine whether a given block is in the cache.
- A search of this kind is called an associative search.

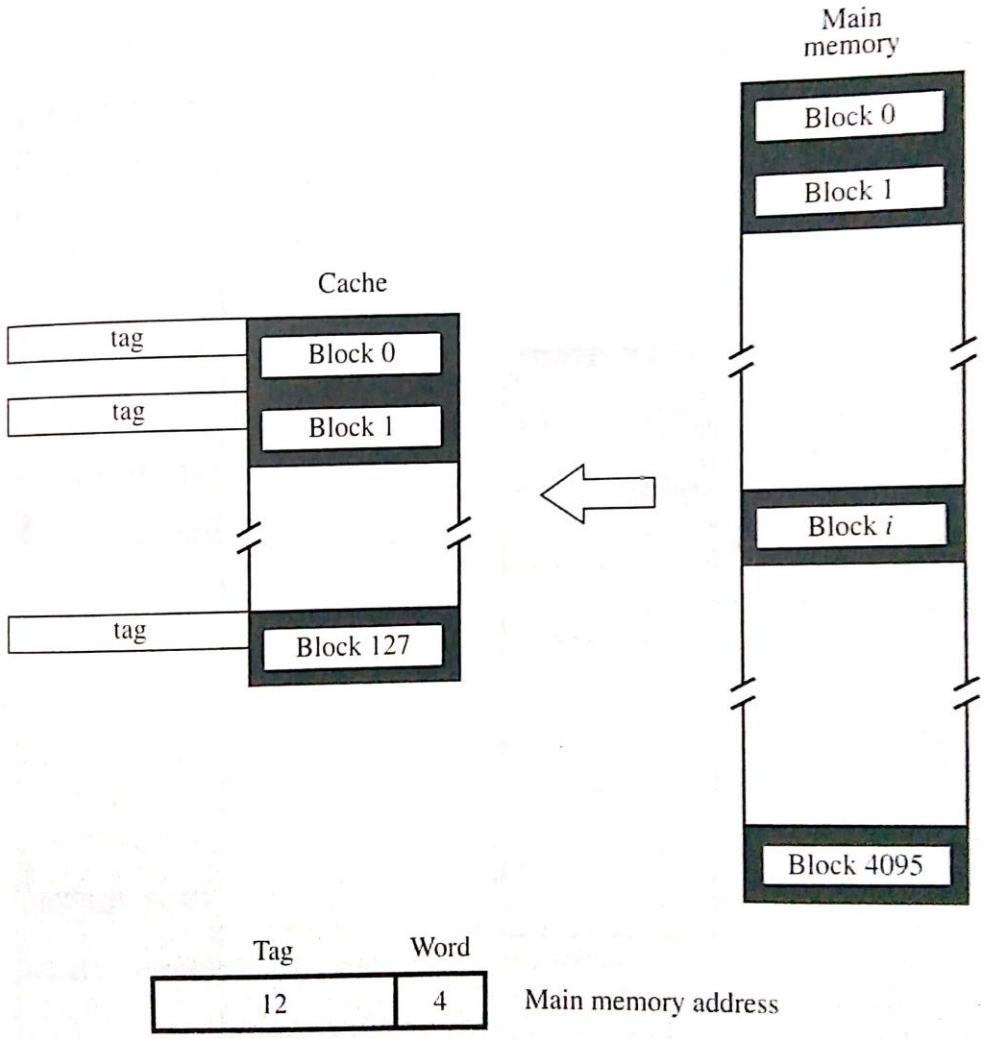


Figure 5.16 Associative-mapped cache.

ADDRESS MAPPING – SET ASSOCIATIVE MAPPING

- A combination of the direct and associative mapping techniques can be used.
- 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.

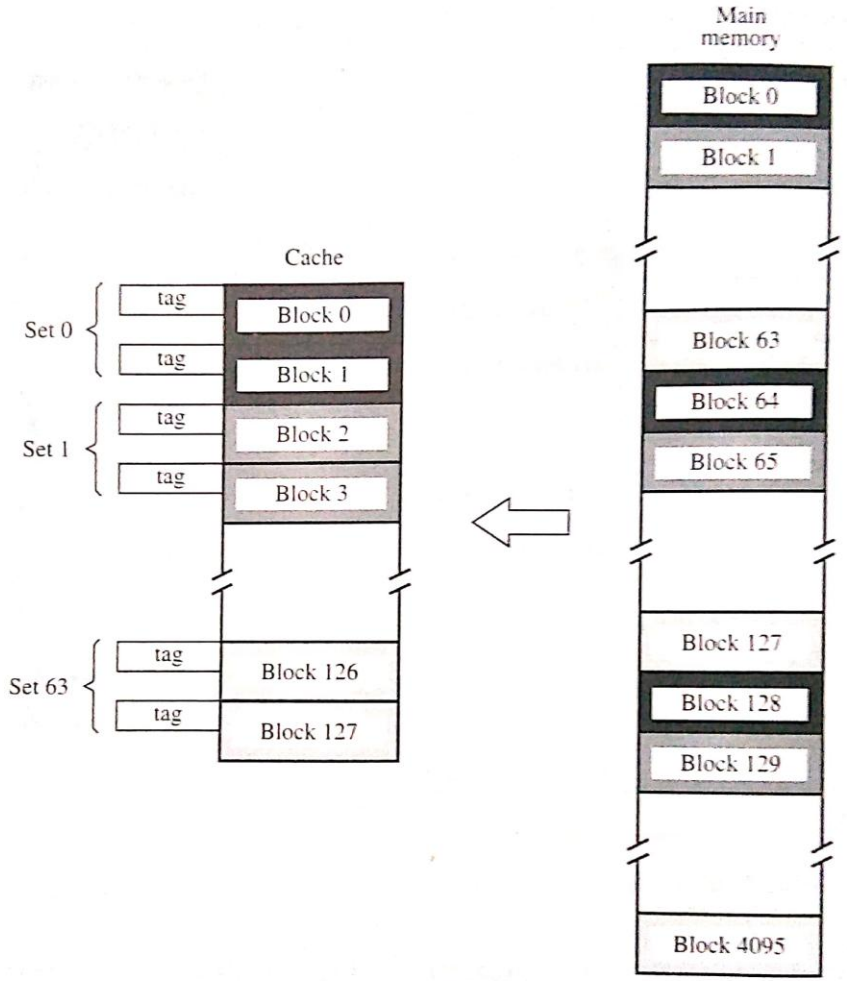


Figure 5.17 Set-associative-mapped cache with two blocks per set.

ADDRESS MAPPING – SET ASSOCIATIVE MAPPING

- For a cache with two blocks per set, memory blocks 0, 64, 128,…, 4032 map into cache set 0, and they can occupy either of the two block positions within the set.
- Having 64 set 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 number of blocks per set is a parameter that can be selected to suit the requirements of a particular computer.

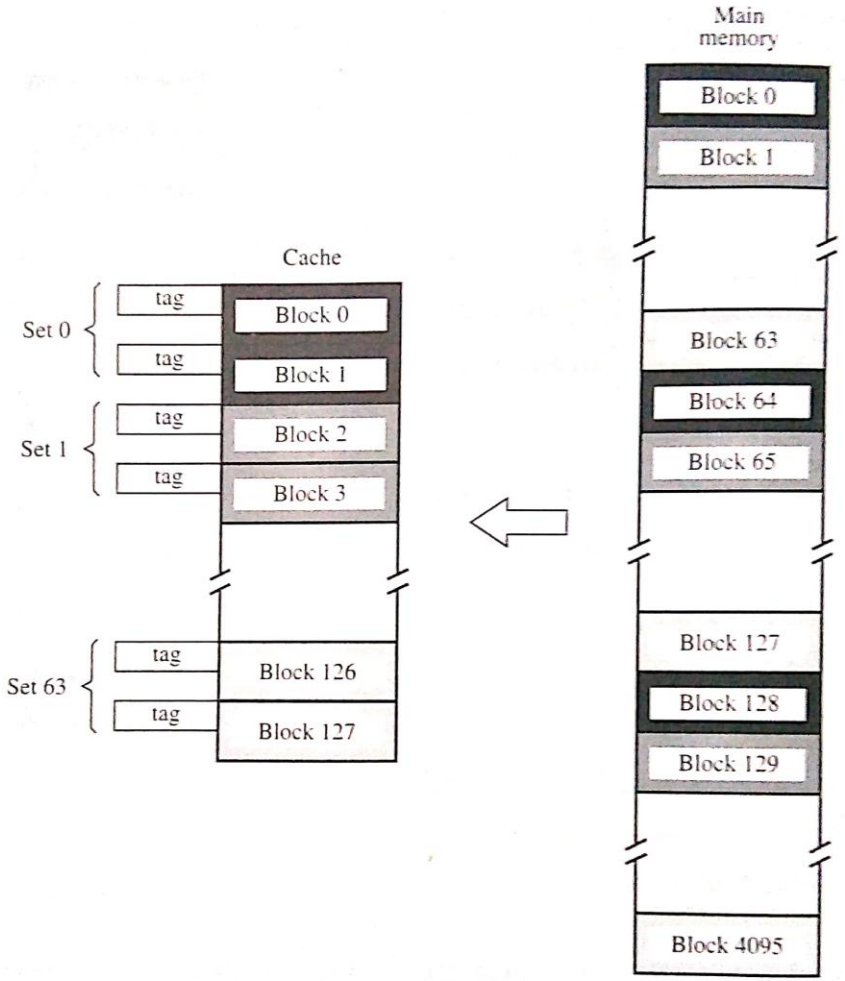


Figure 5.17 Set-associative-mapped cache with two blocks per set.

ADDRESS MAPPING – SET ASSOCIATIVE MAPPING

- For the main memory and cache sizes of four blocks per set can be accommodated by a 5-bit set field, eight blocks per set by a 4-bit set field, and so on.
- 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.

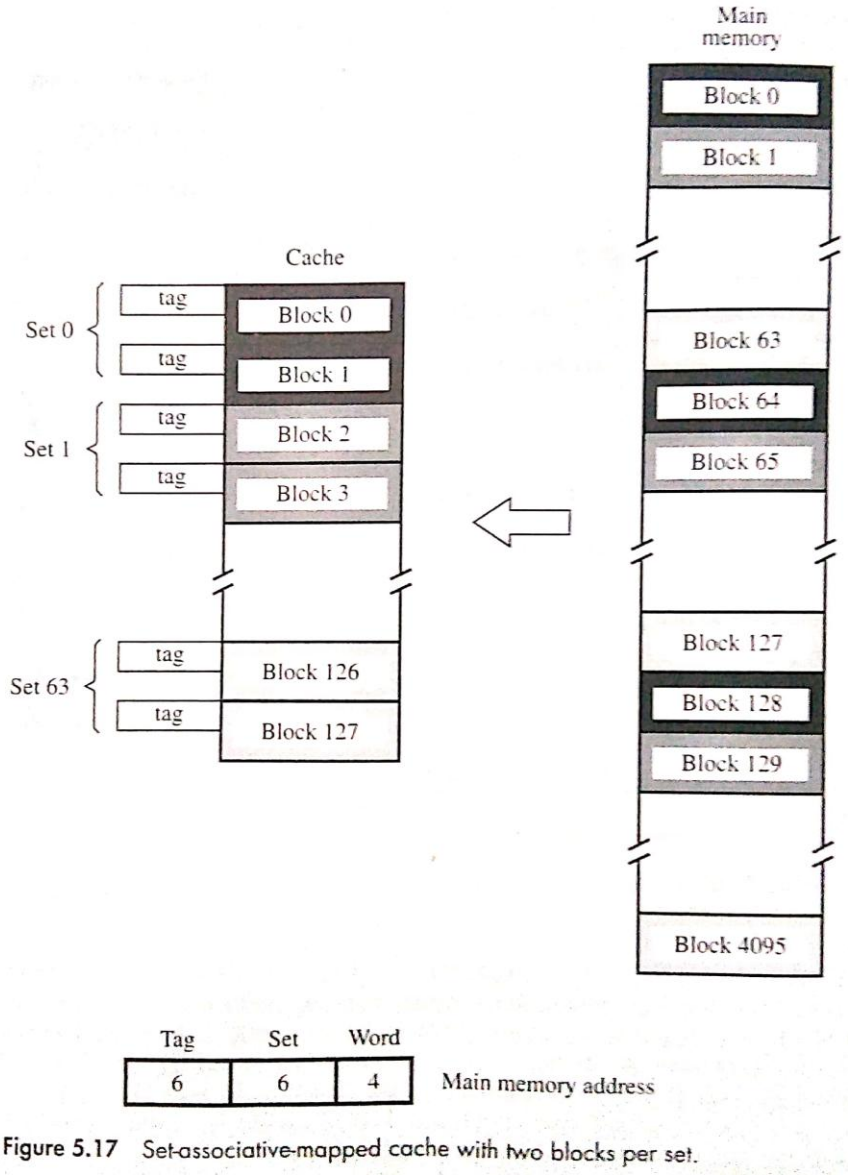


Figure 5.17 Set-associative-mapped cache with two blocks per set.

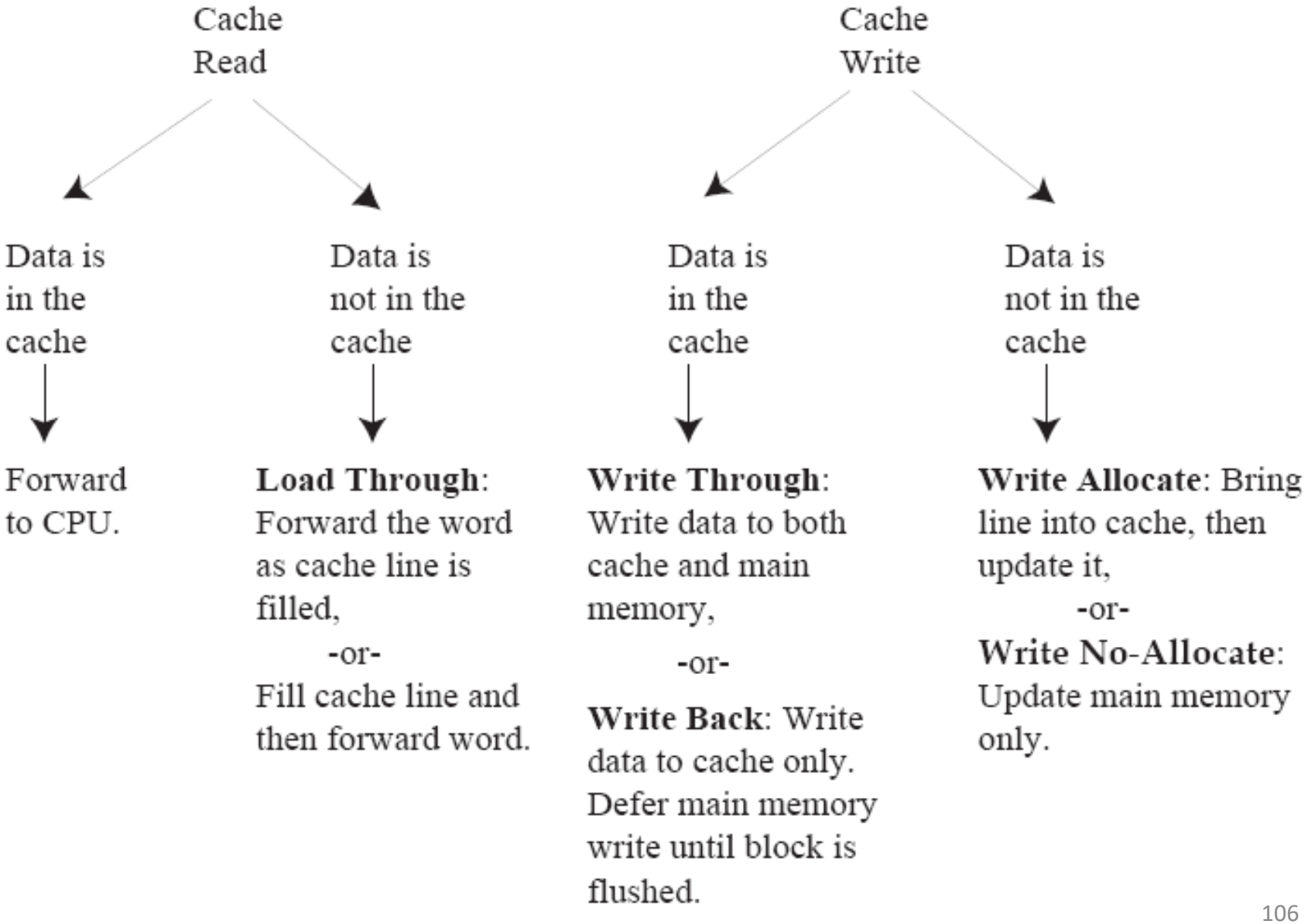
ADDRESS MAPPING – SET ASSOCIATIVE MAPPING

- One more control bit, called the valid bit, must be provided for each block.
- This bit indicates whether the block contains valid data.
- Dirty bit indicates whether the block has been modified during its cache residency, is needed only in systems that do not use the write-through method.
- **Write through:** The information is written to both the block in the cache and to the block in the lower-level memory.
- **Write back:** The information is written only to the block in the cache. The modified cache block is written to main memory only when it is replaced.
- The valid bits are all set to 0 when power is initially applied to the system or when the main memory is loaded with new programs and data from the disk.
- Transfers from the disk to the main memory are carried out by a DMA mechanism.
- Normally, they bypass the cache for both cost and performance reasons.

ADDRESS MAPPING – SET ASSOCIATIVE MAPPING

- The valid bit of a particular block is set to 1 the first time this block is loaded from the main memory.
- Whenever a main memory block is updated by a source that bypasses the cache, a check is made to determine whether the block being loaded is currently in the cache.
- If it is, its valid bit is cleared to 0. This ensures that stale data will not exist in the cache.

CACHE READ AND WRITE POLICIES



VIRTUAL MEMORIES

- In most modern computer systems, the physical main memory is not as large as the address space spanned by an address issued by 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 ranges from a few hundred megabytes to 1 G bytes.
- When a program does not completely fit into the main memory, the part of it is not currently being executed are stored on secondary storage devices, such as magnetic disks.
- Of course, all parts of a program that are eventually executed are first brought into the main memory.
- When a new segment of a program is to be moved into a full memory, it must replace another segment already in the memory.

VIRTUAL MEMORIES

- In most modern computer systems, the operating system moves programs and data automatically between the main memory and secondary storage.
- Thus the application programmer does not need to be aware of limitations imposed by the available main memory.
- Techniques that automatically move program and data blocks into the physical main memory when they are required for execution are called **virtual-memory techniques**.
- Programs, and hence the processor, reference an instruction and data space that is independent of the available physical main memory space.
- The binary addresses that 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 components.

VIRTUAL MEMORIES

- 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.
- On the other hand, if the references address is not in the main memory, its content must be brought into a suitable location in the memory before they can be used.
- A special hardware unit, called the **Memory Management Unit (MMU)** translates virtual addresses into physical addresses.
- When the desired data (or instructions) are in the main memory, these data's are fetched as described as that of cache mechanism.
- If the data are not in the main memory, the MMU causes the operating system to bring the data into the main memory from the disk.
- Transfer of data between the disk and the main memory is performed using the DMA scheme.

VIRTUAL MEMORIES

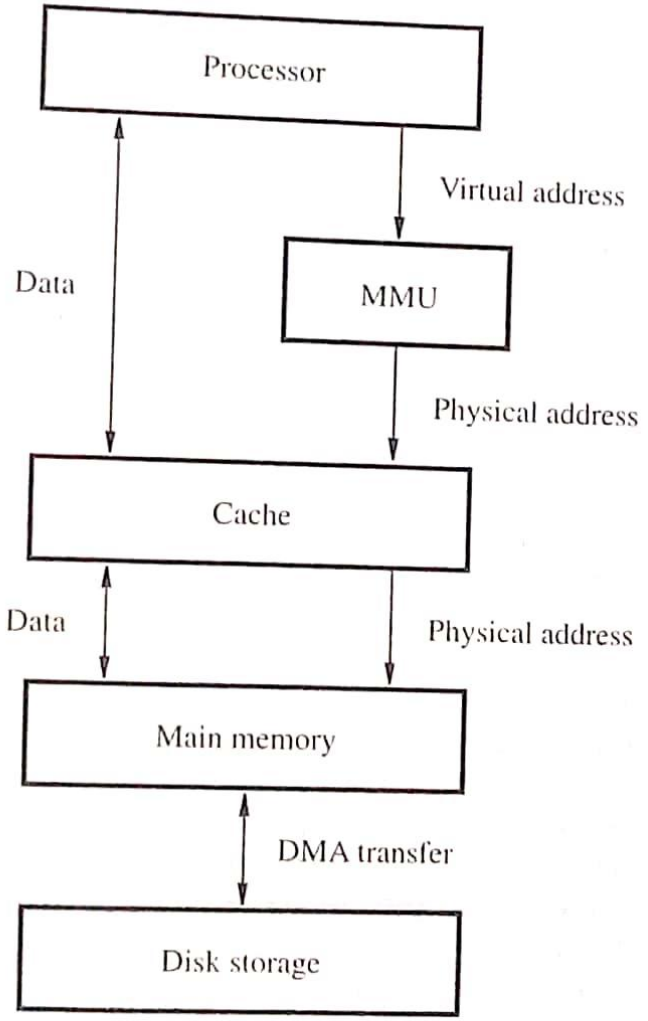


Figure 5.26 Virtual memory organization.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

- 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 moved between the main memory and the disk whenever the translation mechanism determines that a move is required.
- Pages should not be too small, because the access time of a magnetic disk is much longer (several ms) than the access time of the main memory..
- If pages are too large it is possible that a substantial portion of the page may not be used, yet this unnecessary data will occupy valuable space in the main memory.
- The virtual-memory mechanism bridges the size and speed gaps between the main memory and secondary storage and is usually implemented in part by software technique.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

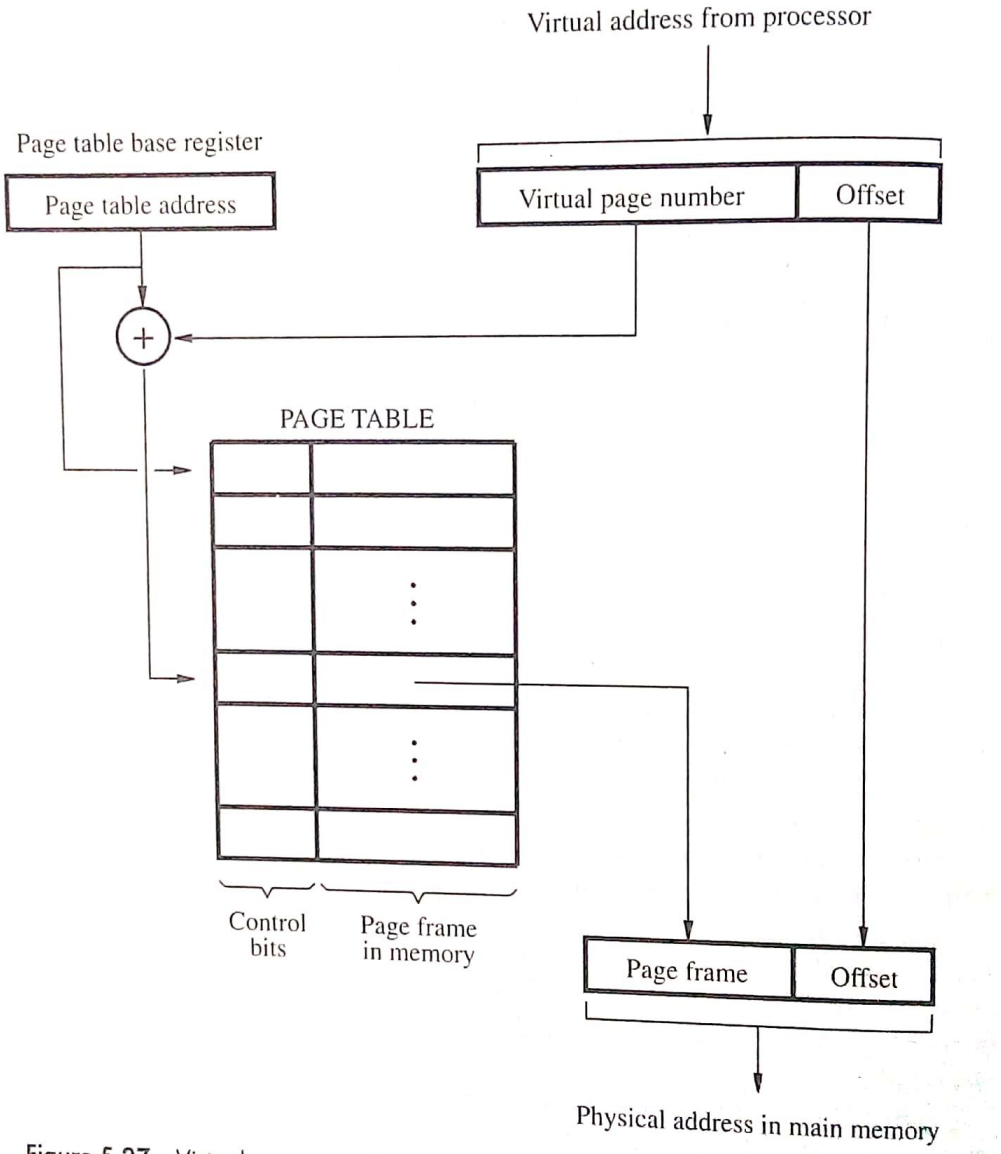


Figure 5.27 Virtual-memory address translation.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

- Each virtual address generated by the processor, whether it is for an instruction fetch or an operand fetch/store operation, it is interpreted as a virtual page number (high-order bits) followed by an offset (lower-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.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

- Each entry in the page table also includes some control bits that describe the status of the page white it is in the main memory.
- One bit indicates the validity of the page – 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.
- Other control bits indicate restrictions that may be imposed on accessing the page. For example, a program may be given full read and write permissions, or it may be restricted to read access only.
- Page table is kept in the main memory.
- However a copy of a small portion of the page table can be accommodated within the MMU. This information is used by the MMU for every read and write access.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

- This portion consists of the page table entries that correspond to the most recently accessed pages.
- A small cache, usually called the **translation lookaside buffer (TLB)** is incorporated into the MMU for the purpose.
- In addition to the information that constitutes a page table entry, the TLB must also include the virtual address of the entry.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

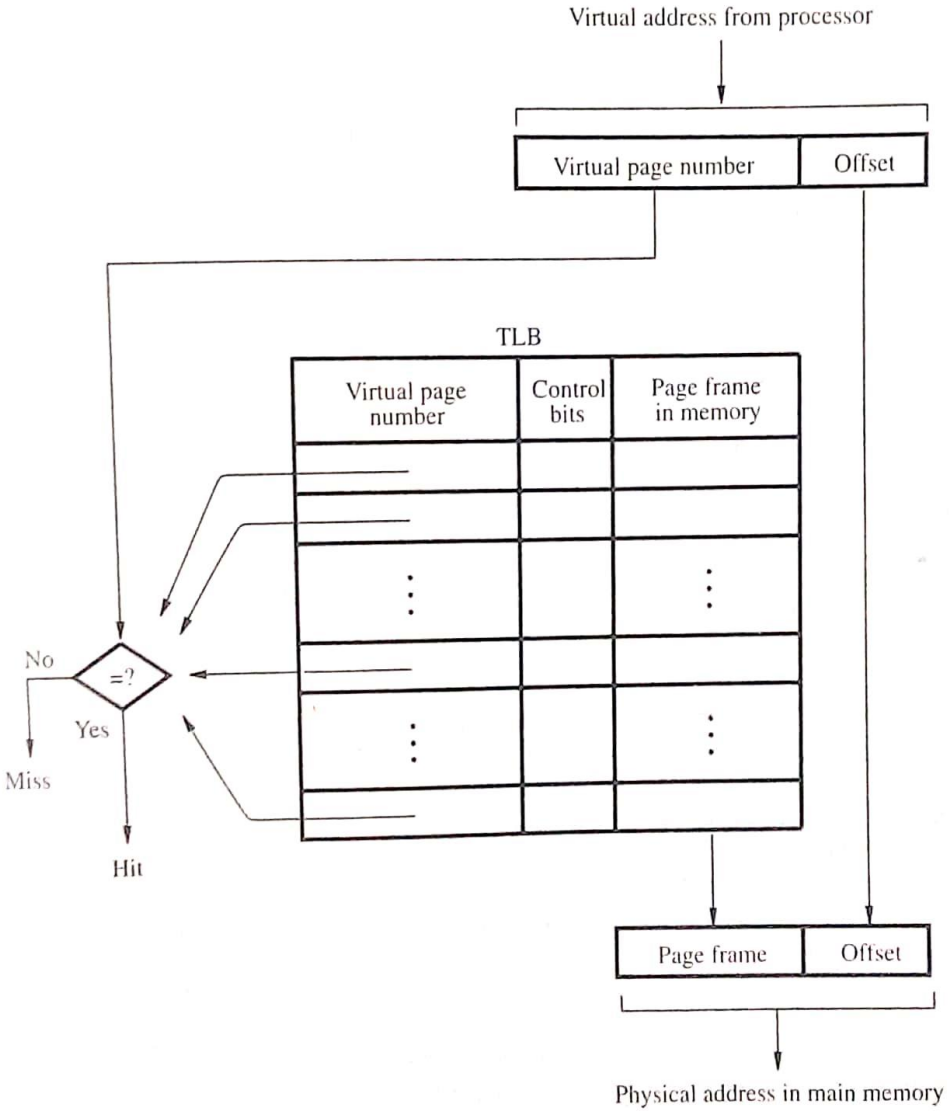


Figure 5.28 Use of an associative-mapped TLB.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

- An essential requirement is that the contents of the TLB be coherent with the contents of page tables in the memory.
- When the operating system changes the contents of the page tables, it must simultaneously invalidate the corresponding entries in the TLB. One of the control bits in the TLB is provided for this purpose.
- 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.
- 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 whole page must be brought from the disk into the memory before access can proceed.

VIRTUAL MEMORIES – ADDRESS TRANSLATION

- The address translation process in the MMU requires some time to perform, mostly dependent on the time needed to look up entries in the TLB.
- Because of locality of reference, it is likely that many successive translations involve addresses on the same page. This is particularly evident in fetching instructions.
- Thus we can reduce the average translation time by including one or more special registers that retain the virtual page number and the physical page frame of the most recently performed translations.
- The information in these registers can be accessed more quickly than the TLB.

SUMMARY

- Memory organization
- Random access memories
- Serial access memories
- Cache memory and its access mechanism
- Memory allocation strategies
- Virtual memory