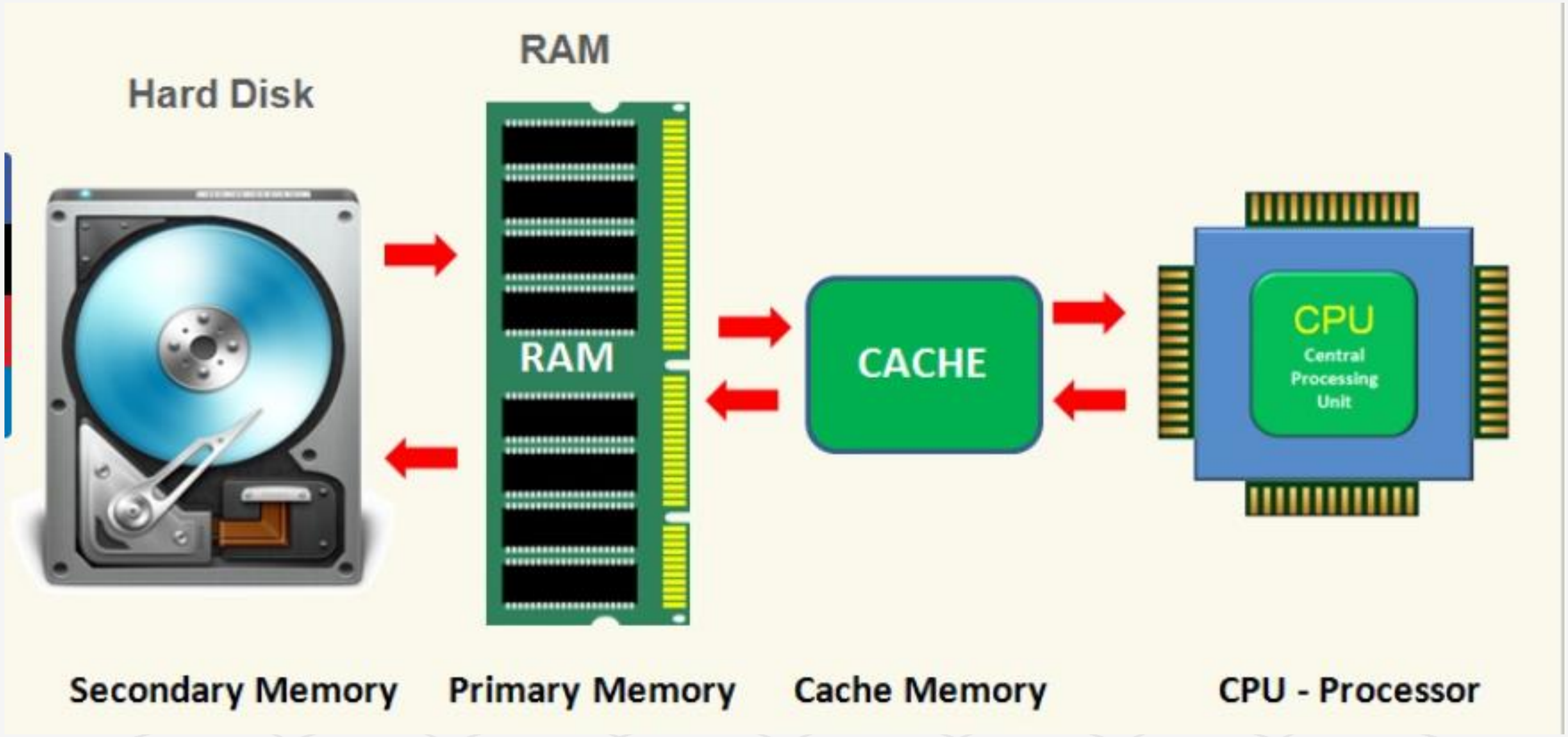
Unit-2: Memory Organization

Prof. Shankar Mali

What is Computer Memory?

Computer memory refers to the hardware or system components used to store data and instructions temporarily or permanently in a computer. It plays a critical role in a computer's functionality by holding the data that the CPU processes or needs to execute tasks. Memory can be classified based on its speed, volatility, and permanence.

Memory Architecture



Memory Architecture

Memory architecture refers to how a computer's memory is organized and how data flows between different types of memory and the processor.

Processor (CPU): The brain of the computer that processes data and instructions.

Cache Memory: A small, super-fast memory located inside or close to the CPU. It stores frequently used data to speed up processing.

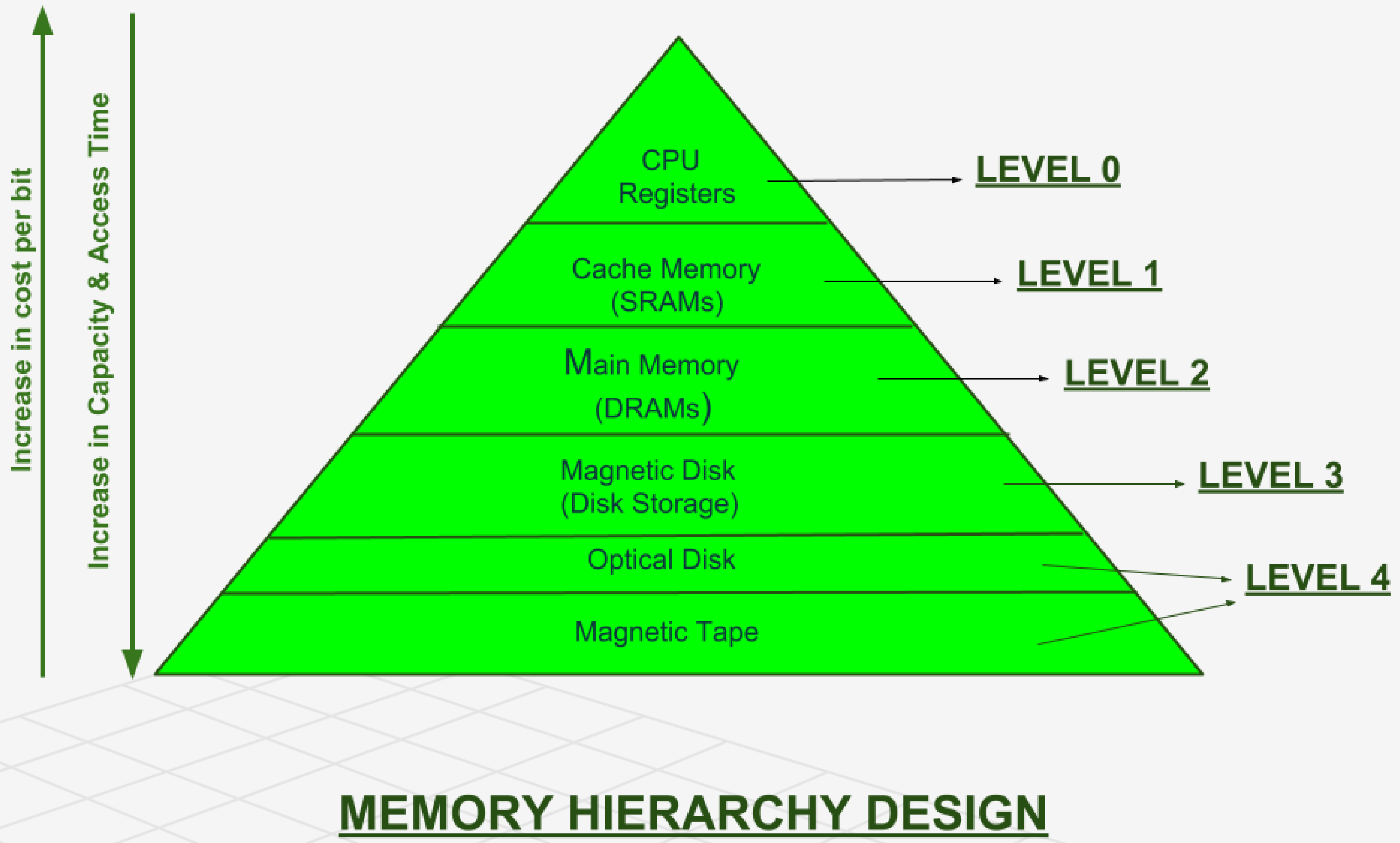
Main Memory (RAM): The computer's primary working memory. It temporarily holds data and instructions the CPU is actively using. It's fast but loses data when the power is off.

Storage (Hard Drive or SSD): Long-term memory where all your files, programs, and data are stored. It's slower than RAM but keeps data even when the computer is off.

Bus System: A pathway that connects the CPU, memory, and storage. It allows data to travel between these components

Memory Hierarchy

The memory hierarchy is the arrangement of different types of memory in a computer system based on speed, size, cost, and proximity to the CPU. It helps ensure efficient data storage and retrieval by balancing performance and cost.



Registers

Location: Inside the CPU.

Function: Temporarily holds data and instructions that the CPU is actively processing.

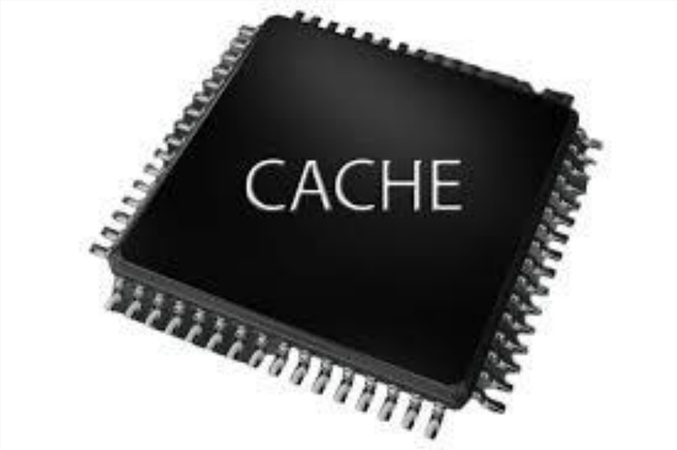
Speed: The fastest type of memory, as it’s directly wired into the CPU.

Capacity: Extremely small (typically 32 or 64 bits per register).

Cost: Very expensive due to high speed and integration with the CPU.

Example: Registers store operands for immediate calculations, like results of arithmetic operations.

Cache Memory



Location: Built into or near the CPU.

Function: Stores frequently accessed data and instructions to avoid fetching them from slower memory.

Speed: Very fast, but slightly slower than registers.

Capacity: Larger than registers but still relatively small.

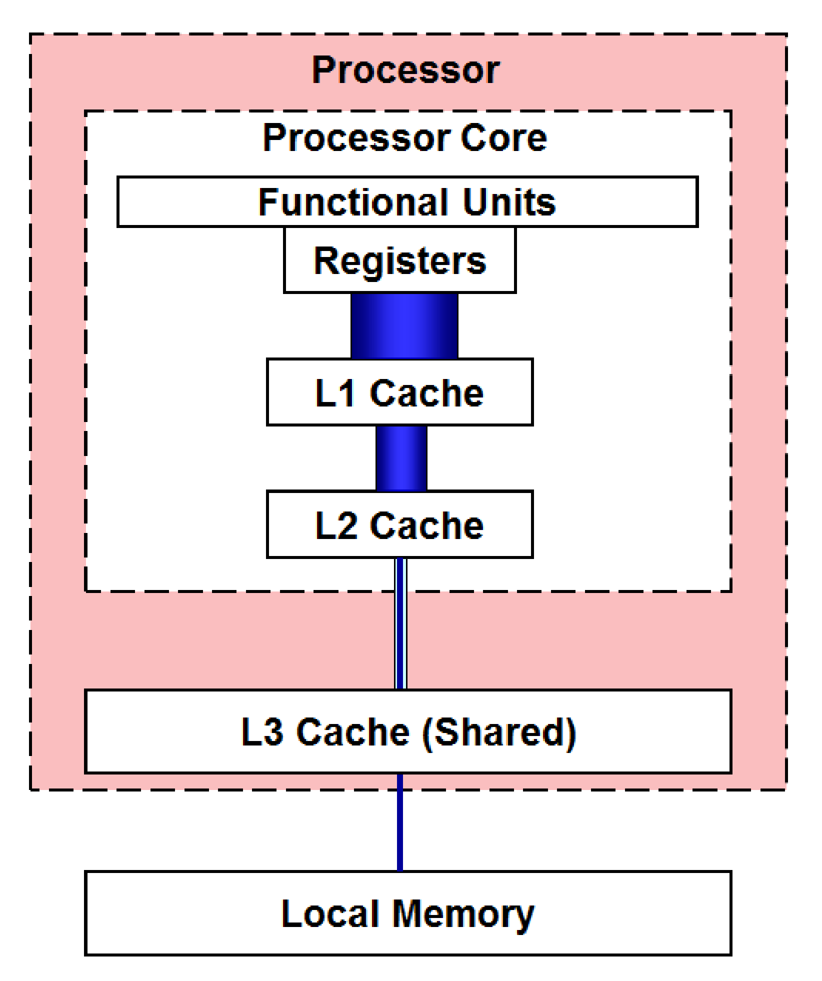
Cost: High, but less expensive than registers.

Cache Memory Levels

L1 Cache: Closest to the CPU, very small (32KB–128KB), and extremely fast.

L2 Cache: Larger (256KB–12MB), shared between cores in some architectures, slower than L1.

L3 Cache: Largest (up to 64MB+), shared among all CPU cores, slower but still faster than RAM.



Main Memory (RAM)

Location: Connected to the motherboard, accessed by the CPU via the memory bus.

Function: Temporarily holds data and programs currently in use.

Speed: Slower than cache but much faster than secondary storage.

Capacity: Typically 8GB–64GB in modern computers.

Cost: Moderate cost per byte, significantly cheaper than cache.

Volatility: Volatile, meaning data is lost when power is off.

Example: When you open an application, its data is loaded into RAM for quick access.

Secondary Storage

Location: Connected to the motherboard via data cables (e.g., SATA).

Function: Provides long-term storage for operating systems, applications, and files.

Speed:

SSD (Solid-State Drive): Faster than traditional hard drives.

HDD (Hard Disk Drive): Slower due to mechanical parts.

Capacity: Typically 256GB–2TB or more.

Cost: Much cheaper than RAM per byte.

Volatility: Non-volatile, retains data even when power is off.

Example: Your operating system and documents are stored here.

Tertiary Storage

Location: Often external to the computer.

Function: Used for backups, archival, and infrequently accessed data.

Speed: Very slow compared to other types of memory.

Capacity: Large, often measured in terabytes or higher.

Cost: Very cheap per byte.

Example: External hard drives, DVDs, Blu-rays, and tape drives.

Cloud Storage

Location: Remote servers accessed via the internet.

Function: Provides off-site storage with virtually unlimited capacity.

Speed: Dependent on internet bandwidth and latency.

Capacity: Scalable, typically in gigabytes or terabytes for individual users.

Cost: Subscription-based, depending on storage size.

Example: Google Drive, Dropbox, and iCloud.

Characteristics of Memory

Speed

Size (Capacity)

Volatility

Cost

Accessibility

Latency

Bandwidth

Power Consumption

Physical Size

Reliability

Write Cycles

Speed

Definition: The rate at which data can be read from or written to memory.

Explanation: Faster memory allows quicker data access, improving overall system performance.

Comparison:

Registers and cache are the fastest.

RAM is slower than cache but faster than secondary storage.

Hard drives, SSDs, and other secondary storage are much slower.

Size (Capacity)

Definition: The amount of data the memory can hold, typically measured in bytes (e.g., KB, MB, GB, TB).

Explanation: Higher capacity memory can store more data, which is essential for running complex programs or handling large datasets.

Comparison:

Registers and cache have the smallest size.

RAM has moderate capacity.

Secondary and tertiary storage offer the largest capacities.

Volatility

Definition: Whether the memory retains data when power is turned off.

Types:

Volatile Memory: Loses data when power is off (e.g., RAM, cache).

Non-Volatile Memory: Retains data even without power (e.g., ROM, SSD, HDD).

Cost

Definition: The cost per byte of memory.

Explanation: Faster and more advanced memory types are more expensive.

Comparison:

Registers and cache are the most expensive.

RAM is moderately priced.

Secondary storage like SSDs and HDDs are cheaper per byte.

Accessibility

Definition: How the CPU accesses the memory.

Types:

Direct Access: Registers, cache, and RAM are directly accessible by the CPU.

Indirect Access: Secondary and tertiary storage require additional steps to access.

Latency

Definition: The time it takes to retrieve data from memory.

Explanation: Lower latency means faster data access.

Comparison:

Registers and cache have the lowest latency.

RAM has moderate latency.

Secondary storage has higher latency due to mechanical or network delays.

Power Consumption

Definition: The amount of power the memory uses during operation.

Comparison:

Registers and cache consume less power due to their smaller size and proximity to the CPU.

RAM consumes more power, especially in large capacities.

SSDs are energy-efficient compared to traditional HDDs.

Bandwidth

Definition: The amount of data that can be transferred in a given amount of time.

Explanation: Higher bandwidth allows more data to be transferred, increasing performance.

Physical Size

Definition: The physical dimensions of the memory component.

Explanation: Smaller devices require more compact memory solutions.

Example:

Registers and cache are microscopic and embedded within the CPU.

RAM modules and storage drives are physically larger.

Reliability

Definition: The likelihood of the memory operating without errors over time.

Explanation: Non-volatile memory like ROM and SSDs is generally more reliable for long-term data storage than volatile memory like RAM.

Memory Characteristics

Characteristic	Registers	Cache	RAM	SSD	HDD
Speed	Fastest	Very Fast	Fast	Moderate	Slow
Size	Smallest	Small	Moderate	Large	Very Large
Volatility	Volatile	Volatile	Volatile	Non-Volatile	Non-Volatile
Cost	Highest	High	Moderate	Moderate	Lowest
Accessibility	Direct	Direct	Direct	Indirect	Indirect

What is RAM?

Random Access Memory, is a type of computer memory that allows data to be read and written randomly, meaning that the computer can access any location in the memory directly rather than having to read the data in a specific order. This makes RAM an essential component of a computer system, as it enables the CPU to access data quickly and efficiently.

RAM is volatile in nature, which means if the power goes off, the stored information is lost. RAM is used to store the data that is currently processed by the CPU. Most of the programs and data that are modifiable are stored in RAM.

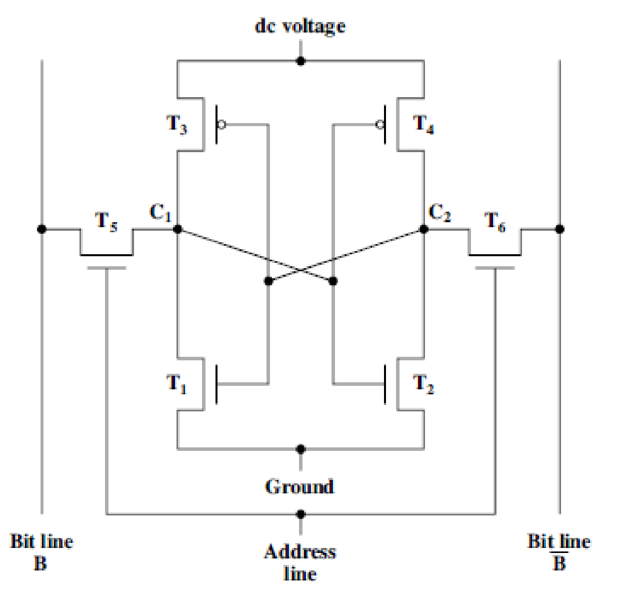
Mainly RAM have 2types

SRAM (Static RAM)

DRAM (Dynamic RAM)

SRAM (Static Random-Access Memory)

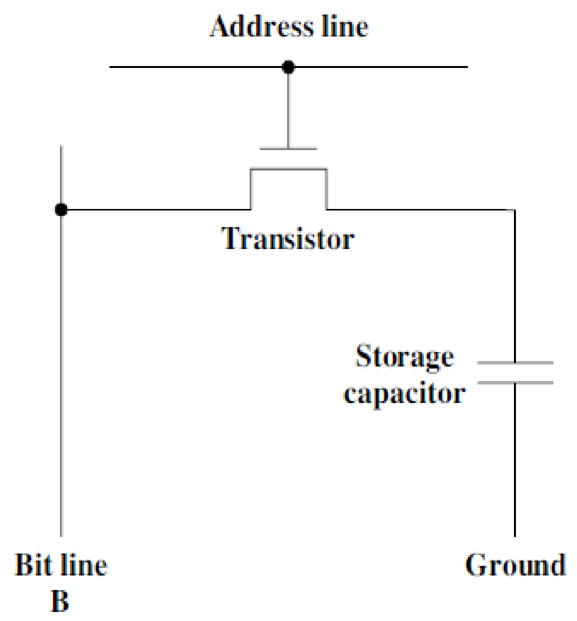
- **Operation**: SRAM uses bistable latching circuitry (flip-flops) to store each bit of data. It does not need to be refreshed as long as power is supplied.
- **Speed**: SRAM is faster than DRAM because it does not require refreshing.
- **Power Consumption**: SRAM consumes less power when idle but more power when active compared to DRAM.
- **Density**: SRAM is less dense, meaning it takes up more space on a chip for the same amount of memory.
- **Cost**: SRAM is more expensive to produce due to its complex structure.
- **Use Cases**: SRAM is typically used in cache memory (e.g., CPU caches) where speed is crucial.



(b) Static RAM (SRAM) cell

DRAM (Dynamic Random-Access Memory)

- **Operation**: DRAM stores each bit of data in a separate capacitor within an integrated circuit. Since capacitors leak charge, the data in DRAM must be refreshed periodically (typically every few milliseconds).
- **Speed**: DRAM is slower than SRAM due to the need for refreshing.
- **Power Consumption**: DRAM consumes more power when idle because of the refresh cycles but less power when active compared to SRAM.
- **Density**: DRAM is more dense, allowing for larger memory capacities in the same physical space.
- **Cost**: DRAM is cheaper to produce, making it suitable for larger memory requirements.
- **Use Cases**: DRAM is commonly used as the main memory in computers (e.g., RAM in PCs and servers) where larger capacities are needed.



(a) Dynamic RAM (DRAM) cell

SRAM Vs DRAM

Feature	SRAM (Static RAM)	DRAM (Dynamic RAM)
Storage Mechanism	Uses flip-flops (transistors) to store data.	Uses capacitors to store data.
Refresh Requirement	No refresh needed (static).	Requires periodic refreshing (dynamic).
Speed	Faster access times (nanoseconds).	Slower due to refresh cycles.
Power Consumption	Lower power when idle, higher when active.	Higher power due to refresh cycles.
Density	Less dense (takes more space for same memory).	More dense (higher capacity in same space).
Cost	More expensive to produce.	Cheaper to produce.
Use Cases	Cache memory (e.g., CPU L1, L2, L3 caches).	Main system memory (e.g., RAM in PCs).
Volatility	Volatile (loses data when power is off).	Volatile (loses data when power is off).

What is DDR? (Double Data Rate Memory)

DDR (Double Data Rate) is a type of **synchronous dynamic RAM (SDRAM)** that can transfer **data twice per clock cycle**—once on the rising edge and once on the falling edge of the clock signal. This **doubles the data transfer rate** compared to older single data rate (SDR) memory. Over time, DDR technology has evolved to provide higher speeds, lower power consumption, and better efficiency:

DDR Type	Speed (MT/s)	Voltage	Bandwidth	Introduction Year
DDR1	200–400 MT/s	2.5V	1.6–3.2 GB/s	2000
DDR2	400–1066 MT/s	1.8V	3.2–8.5 GB/s	2003
DDR3	800–2133 MT/s	1.5V	6.4–17 GB/s	2007
DDR4	1600–3200 MT/s	1.2V	12.8–25.6 GB/s	2014
DDR5	3200–8400 MT/s	1.1V	32–67.2 GB/s	2021

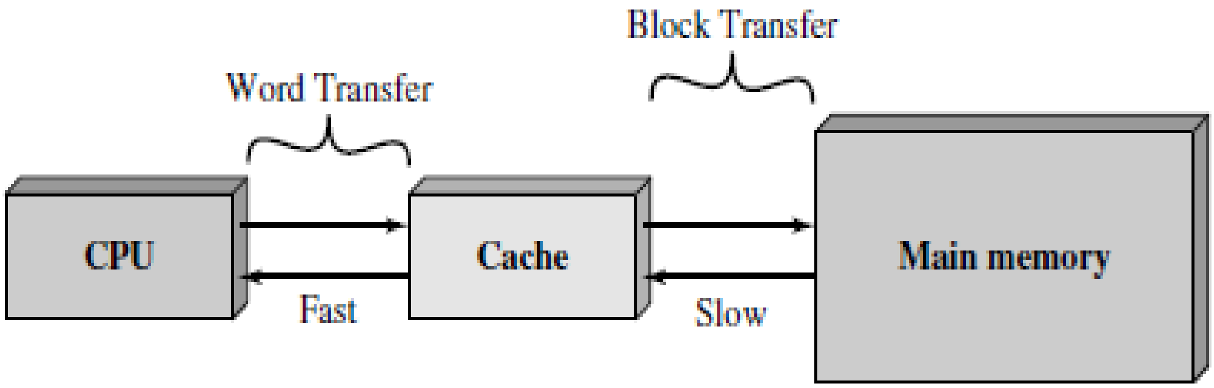
Cache Memory : Cache Memory Principles

Cache memory is intended to give memory speed approaching that of the fastest memories available, and at the same time provide a large memory size at the price of less expensive types of semiconductor memories.

The concept is illustrated in Figure 4.3a.

There is a relatively large and slow main memory together with a smaller, faster cache memory. The cache contains a copy of portions of main memory.

When the processor attempts to read a word of memory, a check is made to



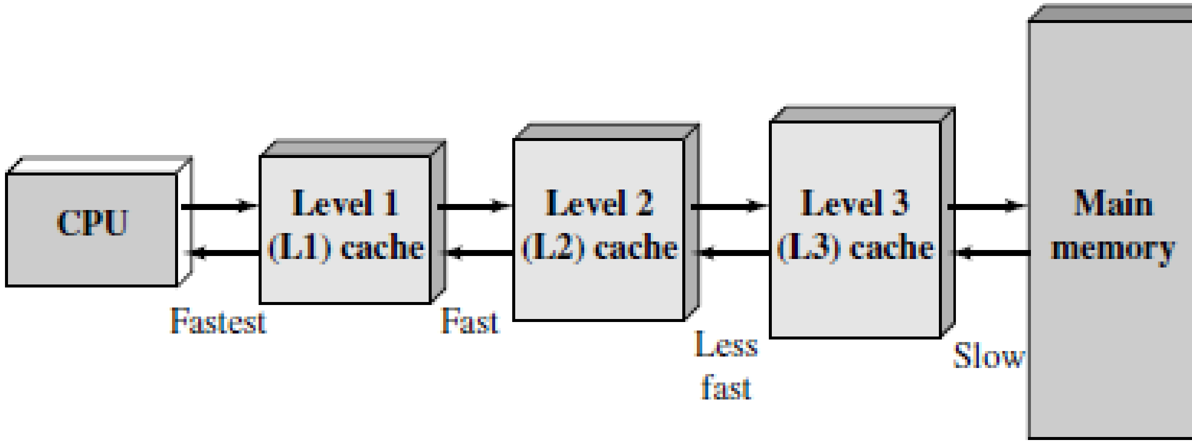
(a) Single cache

Cache Memory

(Cont.) determine if the word is in the cache.

If so, the word is delivered to the processor. If not, a block of main memory, consisting of some fixed number of words, is read into the cache and then the word is delivered to the processor.

Figure 4.3b depicts the use of multiple levels of cache. The L2 cache is slower and typically larger than the L1 cache, and the L3 cache is slower and typically larger than the L2 cache



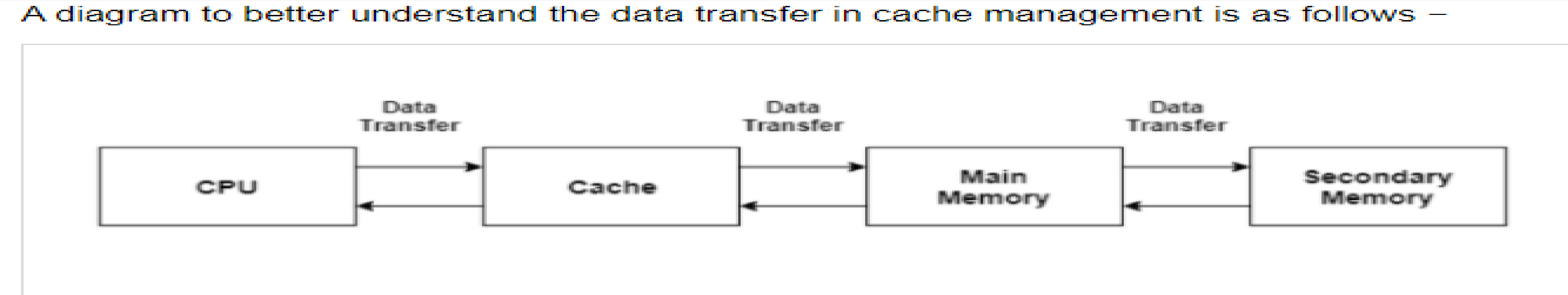
(b) Three-level cache organization

Figure 4.3 Cache and Main Memory

Cache Memory

Cache is a type of memory that is used to increase the speed of data access. Normally, the data required for any process resides in the main memory. However, it is transferred to the cache memory temporarily if it is used frequently enough.

A diagram to better understand the data transfer in cache management is as follows –



A diagram to better understand the data transfer in cache management is as follows –

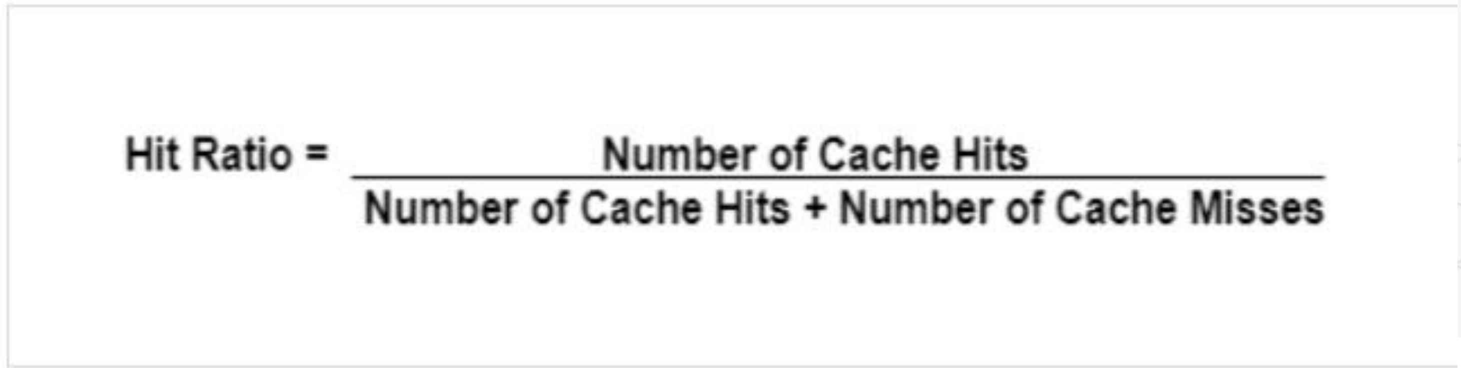
Cache Performance

The cache performance can be explained using the following steps −

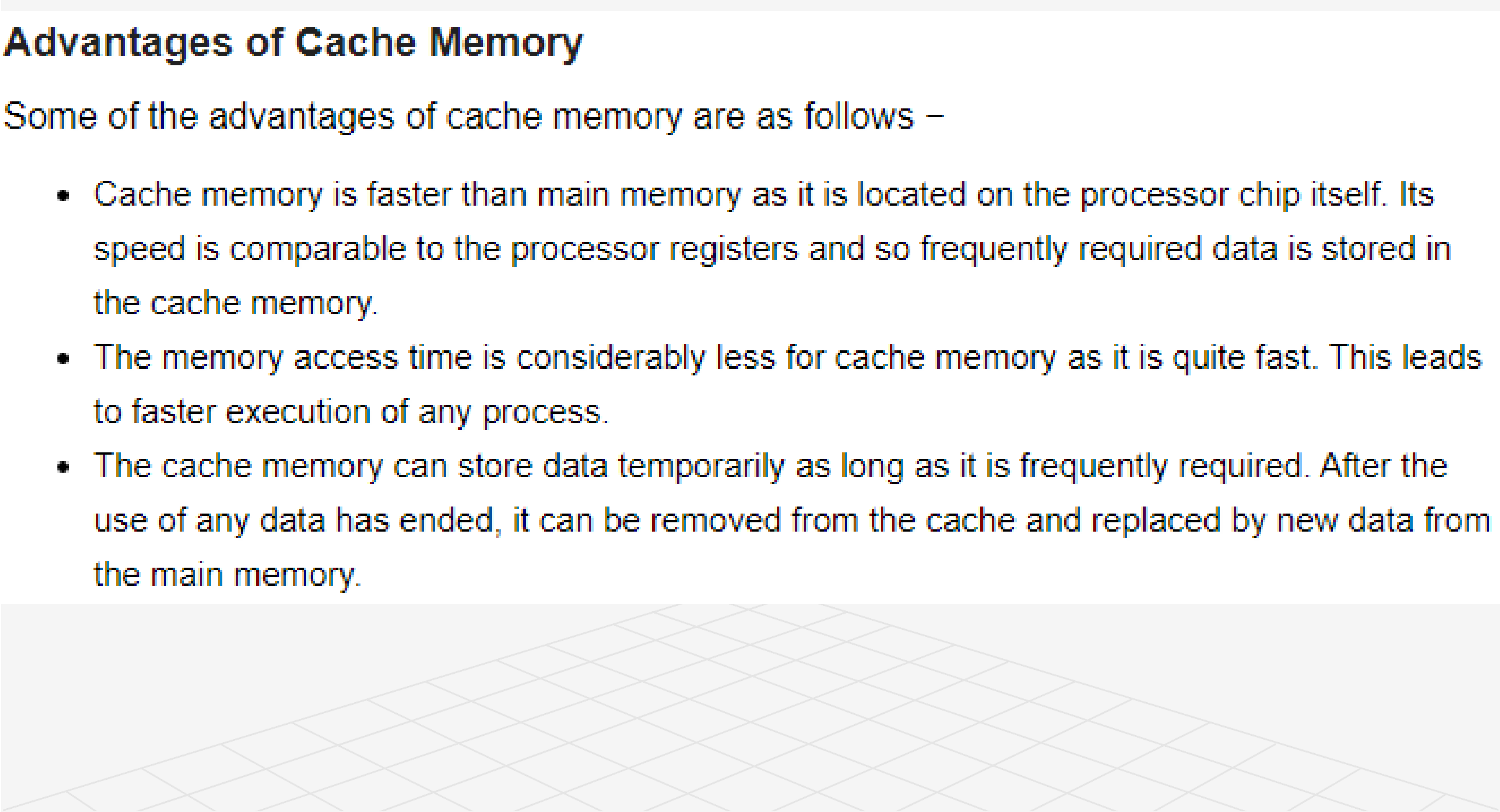
- If a process needs some data, it first searches in the cache memory. If the data is available in the cache, this is termed as a **cache hit** and the data is accessed as required.

- If the data is not in the cache then it is termed as **a cache miss**. Then the data is obtained from the main memory. After that the data is transferred to the cache memory under the assumption that it will be needed again.

- The performance of the cache is measured using the hit ratio. It is the number of cache hits divided by the total cache accesses. The formula for this is:



$$\text{Hit Ratio} = \frac{\text{Number of Cache Hits}}{\text{Number of Cache Hits} + \text{Number of Cache Misses}}$$



Advantages of Cache Memory

Some of the advantages of cache memory are as follows −

- Cache memory is faster than main memory as it is located on the processor chip itself. Its speed is comparable to the processor registers and so frequently required data is stored in the cache memory.
- The memory access time is considerably less for cache memory as it is quite fast. This leads to faster execution of any process.
- The cache memory can store data temporarily as long as it is frequently required. After the use of any data has ended, it can be removed from the cache and replaced by new data from the main memory.

Disadvantages of Cache Memory

Some of the disadvantages of cache memory are as follows –

Since the cache memory is quite fast, it is extremely useful in any computer system. However, it is also quite expensive.

The cache is memory expensive as observed from the previous point. Also, it is located directly on the processor chip. Because of these reasons, it has a limited capacity and is much smaller than main memory.

Read Only Memory (ROM)

ROM (Read-Only Memory) is a type of non-volatile memory that permanently stores data. Unlike RAM, which loses data when the power is off, ROM retains information even after power loss. It is mainly used for firmware, system boot processes, and embedded applications.

Read Only Memory (ROM)

What is it?

ROM is a pre-programmed memory that comes with data already stored at the time of manufacturing.

The data in ROM cannot be modified or erased.

How it Works?

Since ROM is non-volatile, it stores critical data such as the BIOS (Basic Input/Output System), which helps start a computer.

When the system is powered on, the processor reads instructions from ROM to boot up the system.

Uses of ROM

✅ Used in BIOS, embedded systems, calculators, and old gaming consoles.

📌 Example: The firmware in old computers and video game consoles like Nintendo cartridges.

PROM (Programmable Read-Only Memory)

what is it?

PROM is a blank ROM chip that allows the user to write data once using a special device called a PROM programmer.

Once written, the data cannot be changed or erased.

How it Works?

Inside a PROM chip, there are tiny fuses that can be "burned" (blown permanently) when programmed.

The pattern of burned and unburned fuses represents binary data (1s and 0s).

Since the fuses cannot be repaired, data is permanent after programming.

Uses of PROM

✅ Used in firmware, game cartridges, and microcontrollers where data needs to be stored permanently.

📌 Example: Used in old gaming consoles like Atari and arcade machines.

EPROM (Erasable Programmable Read-Only Memory)

What is it?

EPROM is a type of PROM that can be erased and reprogrammed multiple times using ultraviolet (UV) light.

It has a transparent quartz window on top through which UV light can pass to erase the data.

How it Works?

When exposed to UV light for 10-30 minutes, the stored data is erased, making the chip reusable.

After erasing, new data can be written using a special EPROM programmer.

Uses of EPROM

✅ Used in BIOS chips, microcontrollers, and industrial equipment where updates are required.

📌 Example: Early computer BIOS chips were often EPROM-based, allowing manufacturers to update system firmware.

EEPROM (Electrically Erasable Programmable Read-Only Memory)

What is it

EEPROM is an advanced version of EPROM that can be erased and rewritten electrically, without UV light.

It allows byte-level modification, meaning specific sections of data can be changed without erasing the entire memory.

How it Works?

Uses electrical signals to erase and rewrite data.

Unlike EPROM, which needs a UV light and a dedicated erasing process, EEPROM can be reprogrammed while still inside a system.

Uses of EEPROM

✅ Used in modern BIOS chips, smart cards, RFID tags, and microcontrollers for firmware storage and updates.

📌 Example: Used in modern computers' BIOS and USB flash drives (a form of EEPROM-based memory).

EEPROM (Electrically Erasable Programmable Read-Only Memory)

DMA (Direct Memory Access)

DMA (Direct Memory Access) is a technique that allows certain hardware components, like disk controllers, network cards, or GPUs, to transfer data directly to and from the main memory (RAM) without involving the CPU. This improves system performance by freeing the CPU from managing repetitive data transfer tasks.

Why DMA is needed?

DMA (Direct Memory Access)

In a system without DMA, the CPU is responsible for handling all data transfers between memory and I/O devices, which leads to:

- High CPU utilization.
- Slower system performance due to CPU involvement in data transfer.

With DMA, the CPU initiates the transfer but is free to perform other tasks while the data transfer happens independently.

Components of DMA

DMA Controller (DMAC): A dedicated hardware module that manages data transfers.

Memory: The primary location where data is stored.

I/O Device: The source or destination of the data transfer.

Bus System: Includes the data bus, address bus, and control bus to facilitate communication.

How DMA Works (Steps in a DMA Transfer)

1. CPU Initiates Transfer: The CPU sets up the DMA controller by specifying:

   Source and destination addresses.

   Amount of data to transfer.

   Type of transfer (read or write).

2. DMA Controller Takes Over: The DMA controller directly communicates with the memory and I/O device to perform data transfer.

3. Cycle Stealing (If Needed): The DMA controller temporarily takes control of the system bus from the CPU to perform data transfer.

4. DMA Transfer Completion: Once the transfer is complete, the DMA controller sends an interrupt signal to notify the CPU.

Types of DMA transfer

1. Burst Mode (Block Transfer Mode):

   DMA transfers a complete block of data at once.

   CPU is halted during the transfer.

   Suitable for high-speed data transfer.

2. Cycle Stealing Mode:

   DMA controller transfers one data unit at a time.

   CPU execution is only briefly interrupted.

   Used when continuous CPU operation is required.

3. Transparent Mode (Hidden DMA):

   DMA only transfers data when the CPU is not using the bus.

   No CPU performance impact but slower data transfer.

Cycle Stealing in DMA

Definition: A process where the DMA controller momentarily "steals" the bus cycles from the CPU to transfer data.

Effect: CPU performance is slightly affected, but it continues execution with minimal delay.

Example: Used in audio/video processing where real-time data transfer is needed.

Cycle Stealing in DMA

Advantages of DMA

Reduces CPU workload.

Faster data transfer compared to CPU-controlled I/O.

Enables real-time data processing (e.g., audio and video streaming).

Supports multi-device communication efficiently.

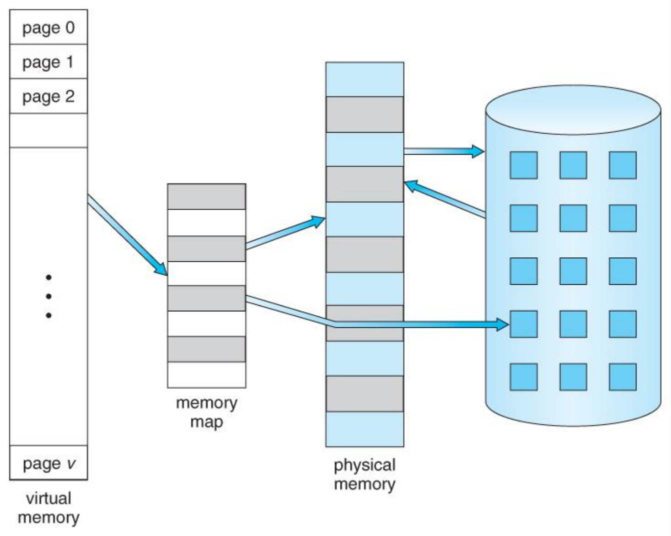
Disadvantages of DMA

Complex hardware (requires a DMA controller).

Potential memory access conflicts between CPU and DMA.

Needs proper bus arbitration to avoid data corruption.

Virtual Memory



Virtual Memory is a technique in modern operating systems that allows a computer to use part of its storage (hard disk or SSD) as if it were RAM. It creates an illusion that the system has more memory than it physically does by managing memory efficiently between RAM and disk space.

Without virtual memory, if a system runs out of RAM, it would either crash or stop running new programs. Virtual memory prevents this by temporarily moving data from RAM to disk storage, allowing processes to run smoothly.

Why we need Virtual Memory?

Limited Physical RAM:

Many applications require more memory than the system's physical RAM can provide.

Virtual memory enables systems to run large applications smoothly.

Efficient Multitasking:

Allows multiple programs to run at the same time by swapping inactive parts of memory to disk.

Memory Isolation & Protection:

Prevents one process from interfering with another’s memory space, enhancing security and stability.

Process Execution Without Loading Entire Program:

Some applications (e.g., large software like Photoshop, databases, or games) don’t need to be fully loaded into RAM at once.

Virtual memory loads only the necessary portions, optimizing memory usage.

How does need Virtual Memory Works?

A. Address Translation

The CPU generates virtual addresses instead of physical addresses.

A hardware unit called the Memory Management Unit (MMU) translates these virtual addresses into physical addresses in RAM or storage.

B. Paging (Most Common Virtual Memory Technique)

Memory is divided into fixed-size pages (in virtual memory) and page frames (in physical RAM).

The Page Table maps virtual pages to physical frames.

If a required page is not in RAM, a page fault occurs, and the system loads it from disk (swap space).

C. Swapping

When RAM is full, the OS moves inactive pages from RAM to a reserved area on disk called swap space.

This process, called paging in (from disk to RAM) and paging out (from RAM to disk), helps manage memory efficiently.

Components of Virtual Memory

RAM (Physical Memory) – Stores active pages and frequently accessed data.

Hard Disk/SSD (Swap Space) – Stores less frequently used data when RAM is full.

Page Table – A mapping system that links virtual memory addresses to physical memory locations.

Memory Management Unit (MMU) – Handles address translation between virtual and physical memory.

CPU & Operating System – Manages virtual memory operations and ensures efficient data access.

Virtual Memory Techniques

Paging

Divides memory into fixed-size blocks (pages).

Reduces external fragmentation but can cause internal fragmentation.

Segmentation

Divides memory into logical units (segments) like functions, arrays, or stacks.

More flexible than paging but leads to external fragmentation

Demand Paging

Loads pages into memory only when they are needed (on-demand).

Reduces memory waste but may cause page faults.

Advantages of Virtual Memory

✅ Allows Running Large Applications: Programs that require more memory than available RAM can still run smoothly.

✅ Enhances Multitasking: Multiple applications can run without exhausting RAM.

✅ Provides Memory Isolation & Protection: Prevents unauthorized access between processes.

✅ Cost-Effective: Allows cheaper systems to run demanding applications without needing expensive RAM upgrades.

✅ Efficient Memory Usage: Optimizes RAM usage by keeping only necessary data in active memory.

Disadvantages of Virtual Memory

❌ Slower Performance Compared to RAM: Since accessing data from disk storage is much slower than accessing RAM.

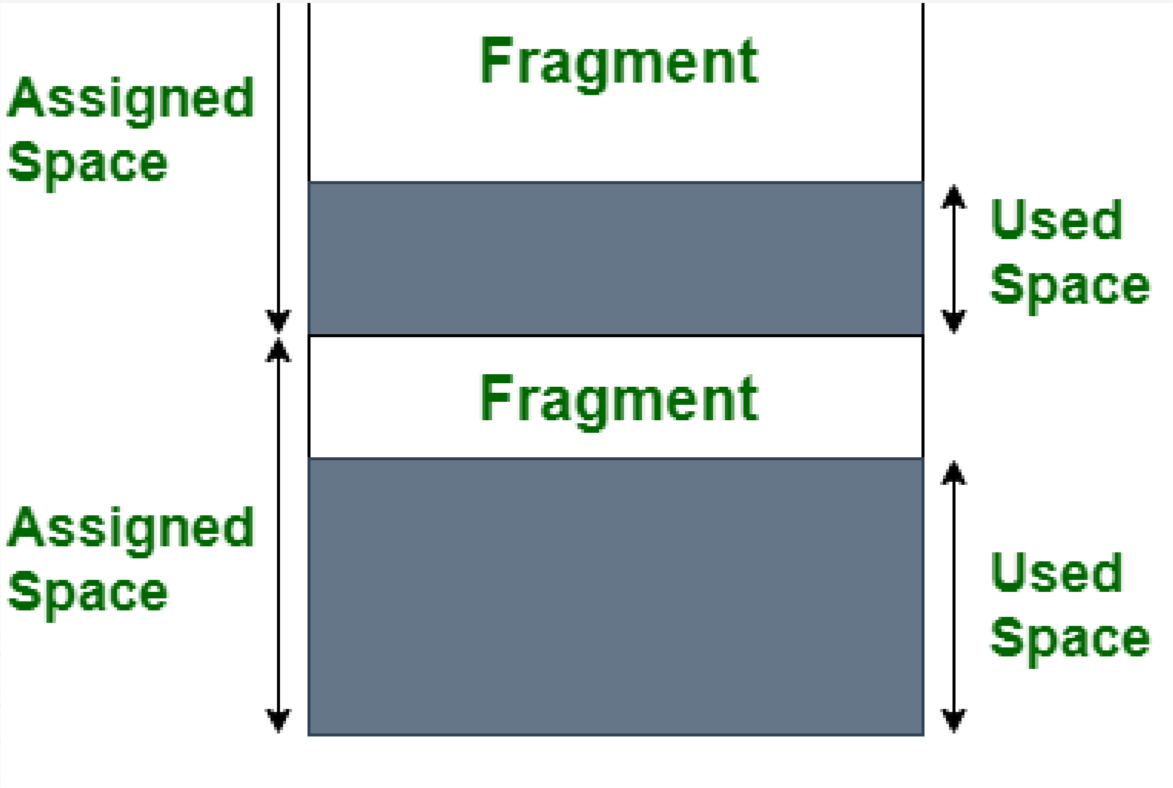
❌ Page Thrashing: When too many pages are swapped between RAM and disk, leading to performance degradation.

❌ Increased Wear on SSDs: Frequent read/write operations to disk can shorten the lifespan of SSDs.

❌ Extra Disk Space Required: Swap files take up storage space, which could be used for other data.

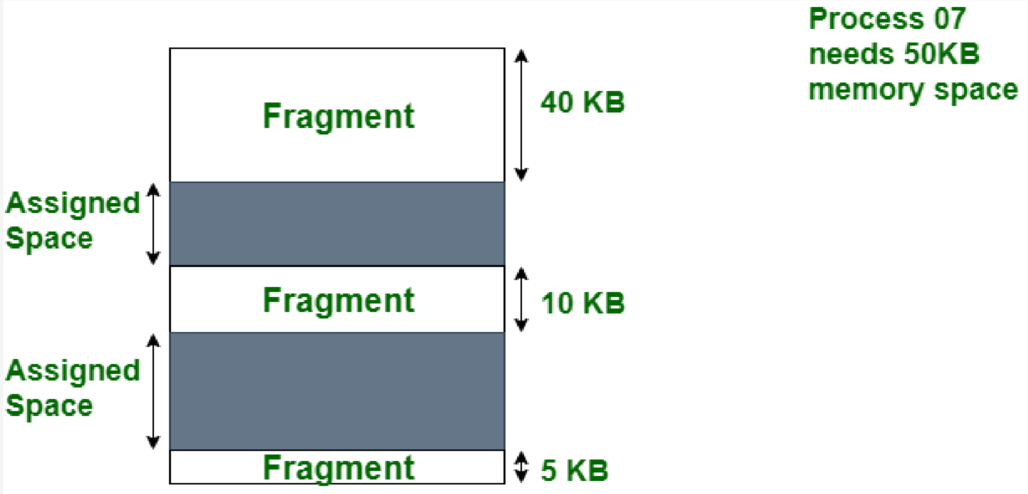
What is Internal Fragmentation?

Internal fragmentation happens when the memory is split into mounted-sized blocks. Whenever a method is requested for the memory, the mounted-sized block is allotted to the method. In the case where the memory allotted to the method is somewhat larger than the memory requested, then the difference between allotted and requested memory is called internal fragmentation. We fixed the sizes of the memory blocks, which has caused this issue. If we use dynamic partitioning to allot space to the process, this issue can be solved.



What is External Fragmentation?

External fragmentation happens when there's a sufficient quantity of area within the memory to satisfy the memory request of a method. However, the process's memory request cannot be fulfilled because the memory offered is in a non-contiguous manner. Whether you apply a first-fit or best-fit memory allocation strategy it'll cause external fragmentation.


Process 07
needs 50KB
memory space
Fragment
40 KB
Assigned Space
Fragment
10 KB
Assigned Space
Fragment
5 KB