UNIVERSITY OF GUJRAT

(UOG)

ASSIGNMENT NO: 01

Course Title: Parallel & Distributed Computing

Course Code: CS-424

Submitted To:

Professor Maryam

Submitted By:

Muzamil Fatima

Reg No.22171915-001

Date of Submission:

October 10,2025

Memory Hierarchy

Definition

The **Memory Hierarchy** in computer architecture is the arrangement of different types of memory in a system based on **speed, cost, and size**.

The faster the memory → the smaller and more expensive it is.
The slower the memory → the larger and cheaper it is.

To balance performance and cost, computers use **a hierarchy of memories**, from fastest to slowest.

Purpose / Need of Memory Hierarchy

1. **Speed Difference**: CPU is much faster than main memory. The hierarchy helps bridge the gap between CPU speed and memory speed.
2. **Cost Efficiency**: Fast memory like cache is expensive. Storing everything in it would be costly, so we combine it with cheaper memory types.
3. **Efficient Access**: Frequently used data is kept in faster memory (cache), and less used data stays in slower memory (like hard disk).
4. **Better Performance**: Reduces CPU waiting time by keeping important data close to it.

Structure of Memory Hierarchy

Level	Memory Type	Speed	Size	Cost per Bit	Example
1	**Registers**	Fastest	Very Small	Very High	CPU registers
2	**Cache Memory**	Very Fast	Small	High	L1, L2, L3 Cache
3	**Main Memory (RAM)**	Fast	Medium	Medium	DDR4/DDR5 RAM
4	**Secondary Memory**	Slow	Large	Low	Hard Disk (HDD), SSD

Level	Memory Type	Speed	Size	Cost per Bit	Example
5	**Tertiary / Backup Storage**	Slowest	Very Large	Very Low	Optical Disks, Magnetic Tapes, Cloud

◈ 1. Registers

- Located **inside the CPU**.
- Used to hold **temporary data** and **instructions** currently being executed.
- Access time: **1 nanosecond or less**.
- Example:
  - Accumulator Register (stores arithmetic results)
  - Instruction Register (holds current instruction)

Example:
When CPU performs addition `A + B`, it loads A and B into **registers**, performs operation, and stores result again in a register.

◈ 2. Cache Memory

- Small, high-speed memory between **CPU and Main Memory**.
- Stores **frequently accessed data** to reduce access time.
- Divided into **L1, L2, and L3** caches.
  - **L1 Cache**: Closest to CPU, fastest, smallest.
  - **L2 Cache**: Slightly larger and slower.
  - **L3 Cache**: Shared among CPU cores, larger but slower.
- Access time: **2–10 nanoseconds**.

Example:
If CPU needs data X, it first checks cache.
If found → *Cache Hit*.
If not found → *Cache Miss*, so data is fetched from RAM and stored in cache for next use.

◈ 3. Main Memory (Primary Memory / RAM)

- Stores data and programs **currently in use**.
- Volatile — contents are lost when power is off.
- Access time: **50–100 nanoseconds**.

Example:
When you open MS Word, the program loads from hard disk → into RAM → for CPU execution.

◈ 4. Secondary Memory

- Used for **long-term storage** of data and programs.
- Non-volatile (data remains even when power is off).
- Slower but cheaper and has large capacity.

Example:
Files, videos, or applications are stored permanently on your hard drive until deleted.

◈ 5. Tertiary / Backup Storage

- Used for **archiving** and **backup**.
- Very high capacity but slow access.

Example:
Government or bank databases backup old records on tapes or cloud servers for long-term storage.

◈ Pyramid Representation (Text Diagram)

```
          +--------------------+
          |     Registers      |   ← Fastest, Smallest, Most Expensive
          +--------------------+
          |    Cache Memory    |
          +--------------------+
          |    Main Memory     |
          +--------------------+
          | Secondary Memory   |
          +--------------------+
          |  Tertiary Memory   |   ← Slowest, Largest, Cheapest
          +--------------------+
```

◈ Relationship Between Levels

- **Speed decreases** as we move down.
- **Capacity increases** as we move down.
- **Cost per bit decreases** as we move down.

👉 **Formula Relation (conceptually):**

```
Speed:   Register > Cache > Main Memory > Secondary > Tertiary
Cost:    Register > Cache > Main Memory > Secondary > Tertiary
Size:    Register < Cache < Main Memory < Secondary < Tertiary
```

◈ Example (Real-Life Analogy)

Think of it like a **student preparing for exams**:

- **Registers** → Things you instantly recall (formulas in your brain).
- **Cache** → Notes you keep open on the desk for quick reference.
- **Main Memory (RAM)** → Books you are currently reading.
- **Secondary Memory** → Books on your shelf.
- **Tertiary Memory** → Old notes stored in the storeroom or cloud.

◈ Advantages of Memory Hierarchy

1. Increases overall **speed and efficiency** of computer system.
2. Reduces **CPU idle time**.
3. Provides **balance between cost and performance**.
4. Enables **fast access to frequently used data**.

◈ Conclusion

The **Memory Hierarchy** is a structured approach to organize memory so that the computer performs efficiently while keeping cost low.
By storing frequently accessed data in faster memory (like cache and registers) and less used data in slower memory (like hard disk), systems achieve a perfect balance between **speed**, **cost**, and **capacity**.