CSE301 – Computer Organization

Lecture 6 – Cache Memory

Characteristics

i. Structural Characteristics

Characteristic	What it Describes	Examples
Location	Where memory resides in the system	CPU registers, cache, main memory, SSD/HDD
Capacity	How much data it can store	8 GB RAM, 1 TB SSD
Unit of transfer	How data moves in/out	Word, block, cache line
Access method	How memory is reached	Sequential, Direct, Random, Associative

Access methods:

1️⃣ ***Sequential Access***

- Data is organized and accessed **in linear order**
- To reach a particular item, all preceding data must be traversed
- **Variable and slow access time**
- Used in: **Magnetic tape**, streaming storage

2️⃣ ***Direct Access***

- Memory is divided into **fixed-size blocks/records**
- The device can move **directly to the vicinity** of the block, then access sequentially within it
- Faster than sequential but **not uniform access time**
- Used in: **HDDs, optical disks**

3️⃣ ***Random Access***

- Each location has a **unique physical address**
- Any location can be accessed **directly and with uniform access time**
- Provides high performance at **higher cost**
- Used in: **RAM, CPU registers**

4️⃣ ***Associative Access (Content-Addressable)***

- Access based on **data content**, not address
- Parallel comparison across all locations → **very fast search**
- Hardware is costly, used for small specialized memories
- Used in: **Cache tag memory, TLB, CAM**

ii. Performance Characteristics

Characteristic	Meaning
Access time	Random: Delay between read request and delivery of data Non-Random: time to position rd/wr mechanism at desired location
Memory cycle time	Time before memory is ready for the next operation
Transfer rate	How fast data is moved once access begins

Transfer rate:

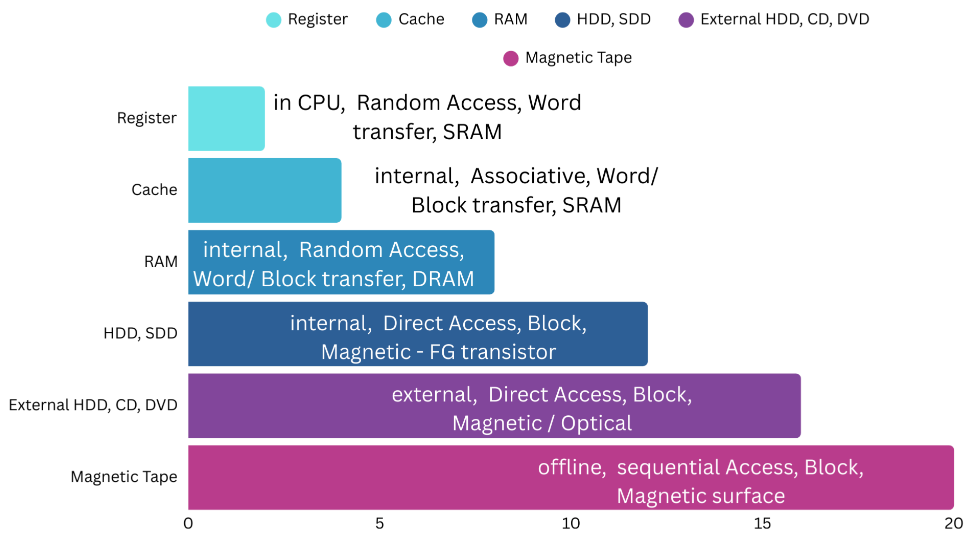
Random:
cycle_time = $T_{access} + T_{recover}$
R = 1/cycle_time

Non-Random:
R = $N/(T_N - T_A)$

where:
N: # of bits
T_N: Av. time to rd/wr N bits
T_A: Av. access time

iii. Physical Characteristics

Category	Description	Examples
Physical Type	The technology used to store bits	Semiconductor memory (RAM/ROM), Magnetic surface memory (disk, tape), Optical / Magneto-optical (CD/DVD)
Physical Characteristics (Volatility)	Behavior of stored data with respect to power	**Volatile:** Data lost without power (e.g., DRAM, SRAM) — **Non-volatile:** Data retained without power (e.g., Flash, ROM, HDD)
Organization	Physical arrangement of bits into larger data units	Bits grouped into **words**, blocks, or pages for access



Memory Hierarchy

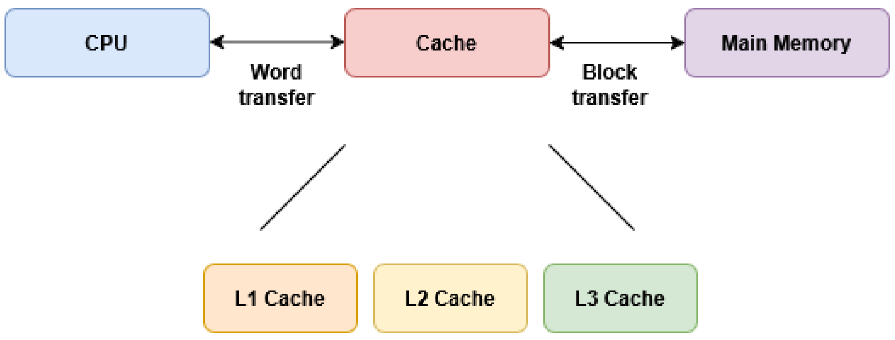
Locality of reference

Locality of reference is the tendency of a program to access a relatively small portion of memory repeatedly over a short period of time.

This is the principle that make us use cache

Cache Memory

- Small amount of fast memory
- Based on SRAM - 1bit ➡️ 6 transistors
- Located between CPU and Main Memory

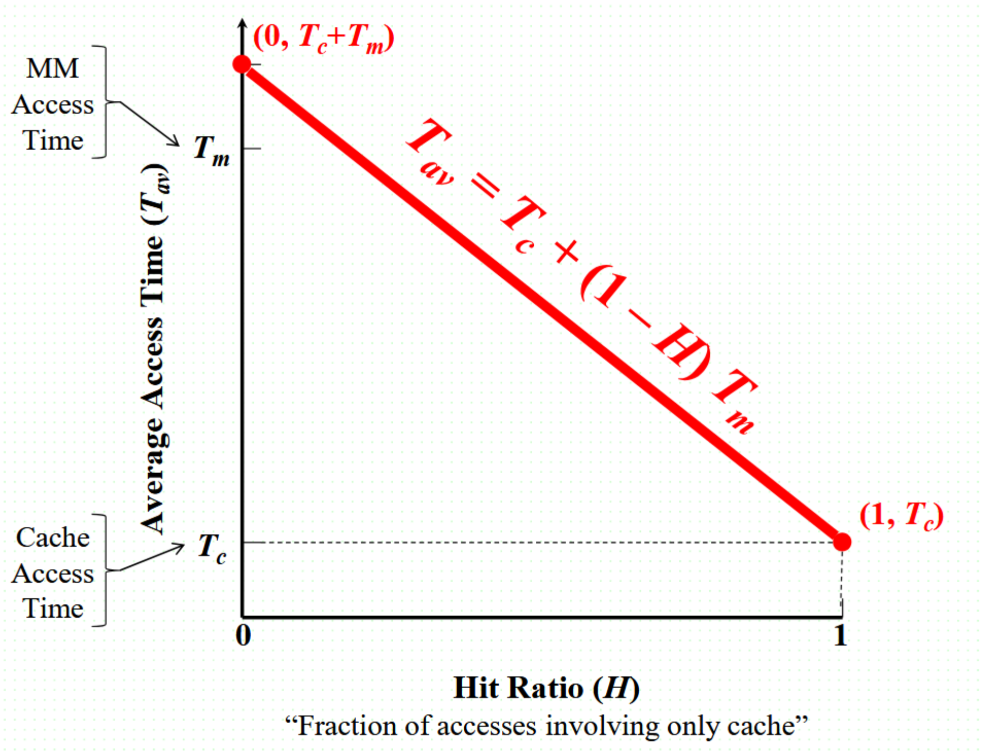


Cache levels

Cache Operation:

CPU requests data → Cache checks:

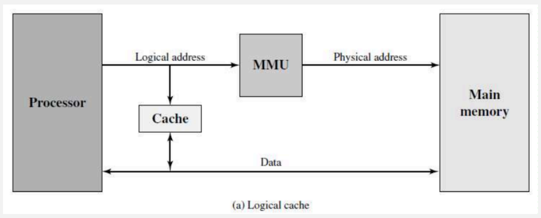
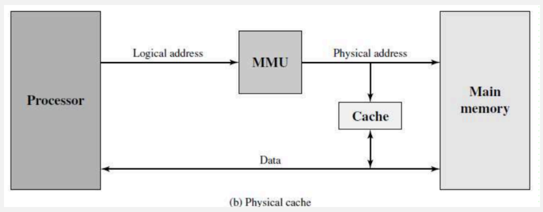
- **Hit:** fast access (`T_cache`)
- **Miss:** slow access from memory + load cache (`T_cache + T_memory`)



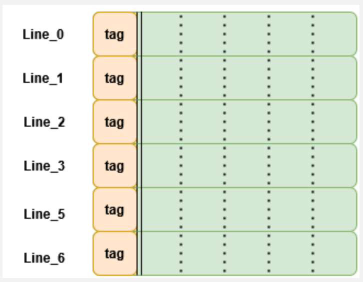
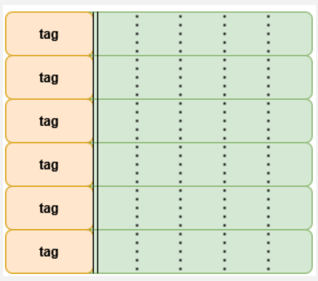
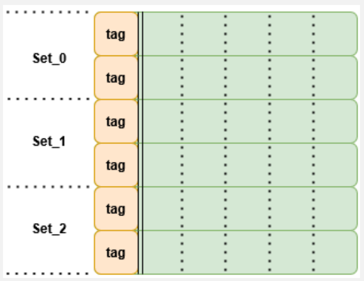
Access time Vs. Hit ratio

Cache Addressing:

	Logical (Virtual) Cache	Physical Cache
Addressing	Uses **virtual addresses**	Uses **physical addresses**
Access Path	CPU accesses cache **before MMU translation**	CPU accesses cache **after address translation**
Speed	Faster access	Slightly slower (translation required)
Context Switch	Must flush cache on each context switch	No flush needed
Application Sharing	Same virtual addresses can be used by different processes	Physical addresses are unique across all memory

Pros & Cons	+ Faster access before MMU - Requires flushing on context switch - Can cause aliasing issues	+ Safe, no flush needed - Slightly slower due to translation
Schematic	 Processor Logical address MMU Physical address Cache Main memory Data (a) Logical cache	 Processor Logical address MMU Physical address Cache Main memory Data (b) Physical cache

Mapping Functions

	Direct Mapping	Associative Mapping	Set Associative Mapping
Schematic			
Address format	Tag (s-r bit) | Line (r bit) | Word (W bit)	Tag (s bit) | Word (W bit)	Tag (s-d bit) | Set (d bit) | Word (W bit)
Memory Size ≡ No. of addressable units	2^{s+w} word	2^{s+w} word	2^{s+w} word
Block size ≡ Line size	2^{w} word	2^{w} word	2^{w} word
No. of blocks in MM	2^{s} block	2^{s} block	2^{s} block
No. of cache lines	2^{r} line	undet.	k * 2^{d} line
Cache size	2^{r} x 2^{w} word	undet.	k x 2^{d} x 2^{w} word
Cache Sets	-	-	2^{d} set
Tag Size	s-r bit	s bit	s-d bit
Pros & Cons	+ Simple Addressing + Fixed Location + Fast Access - Cache miss frequently happen	+ Cache hit rate is much better - Harder to implement comparison circuit - Slow access	-

Replacement Algorithms

- **Direct Mapping:** No replacement needed — each memory block maps to a fixed cache line.
- **Associative & Set-Associative Mapping:** When a replacement is required:

1. **Least Recently Used (LRU):** Replace the block that has not been used for the longest time.
2. **First In First Out (FIFO):** Replace the oldest block in the cache (based on entry time).
3. **Least Frequently Used (LFU):** Replace the block accessed the least number of times (counter-based).
4. **Random:** Replace a randomly chosen block.

Read/Write Policies

Read Policy

- **Read Hit:**
  - Data is read directly from the cache.
- **Read Miss:**
  1. **Read-through:** Load the word from memory into cache and read.
  2. **No Read-through:** Read directly from memory without updating the cache.

Write Policy

- **Write Hit:**
  1. **Write-through:** Write data to both cache and main memory immediately (word-by-word).
  2. **Write-back:** Write data to cache only; update memory later **only when the block is replaced** (block-by-block). Use a **dirty bit** to track changes.
- **Write Miss:**
  - Allocate the block in cache first, then follow the **write-hit policy**.