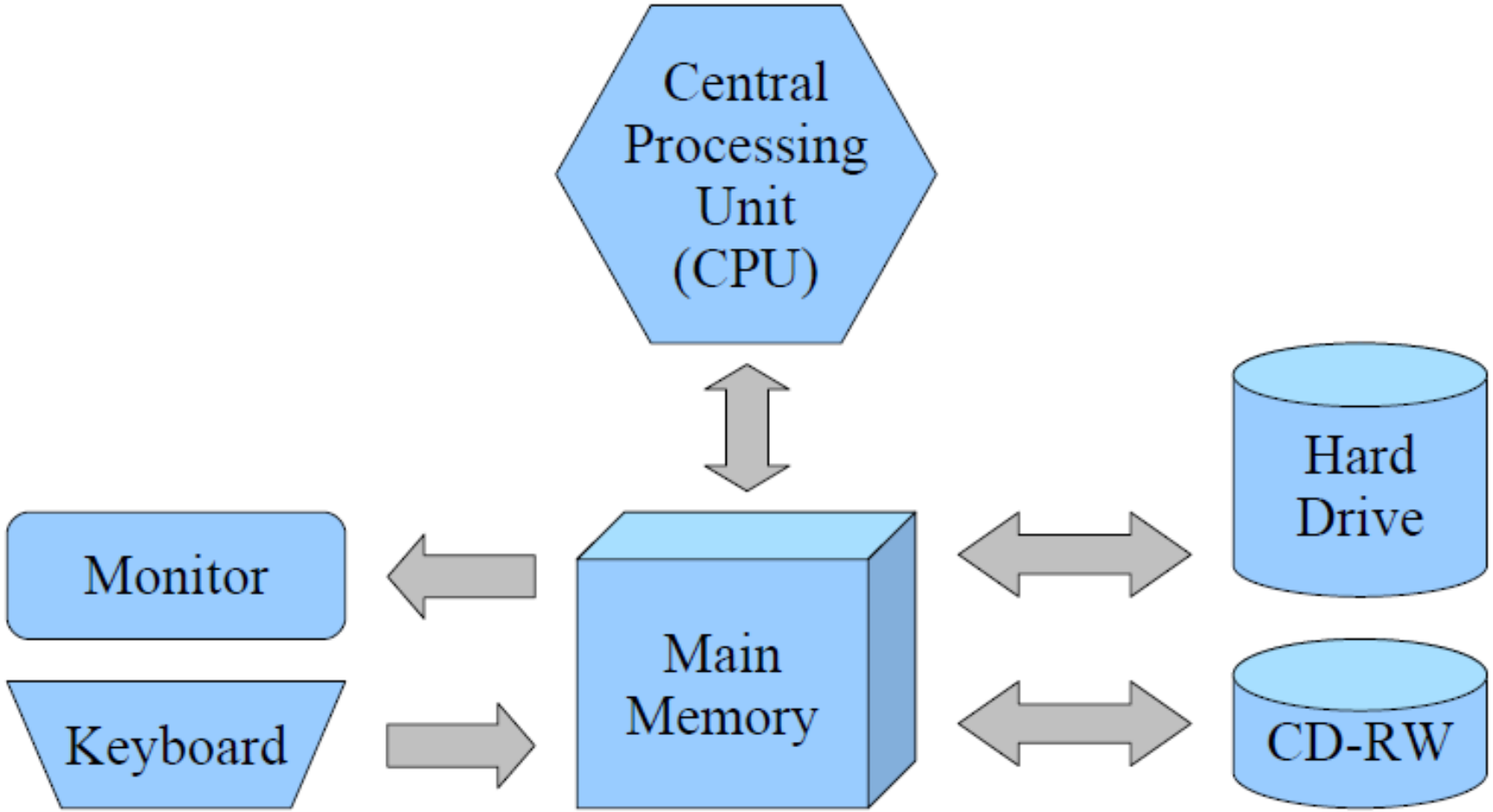
Evaluasi Kinerja dan Optimisasi Sistem Komputer

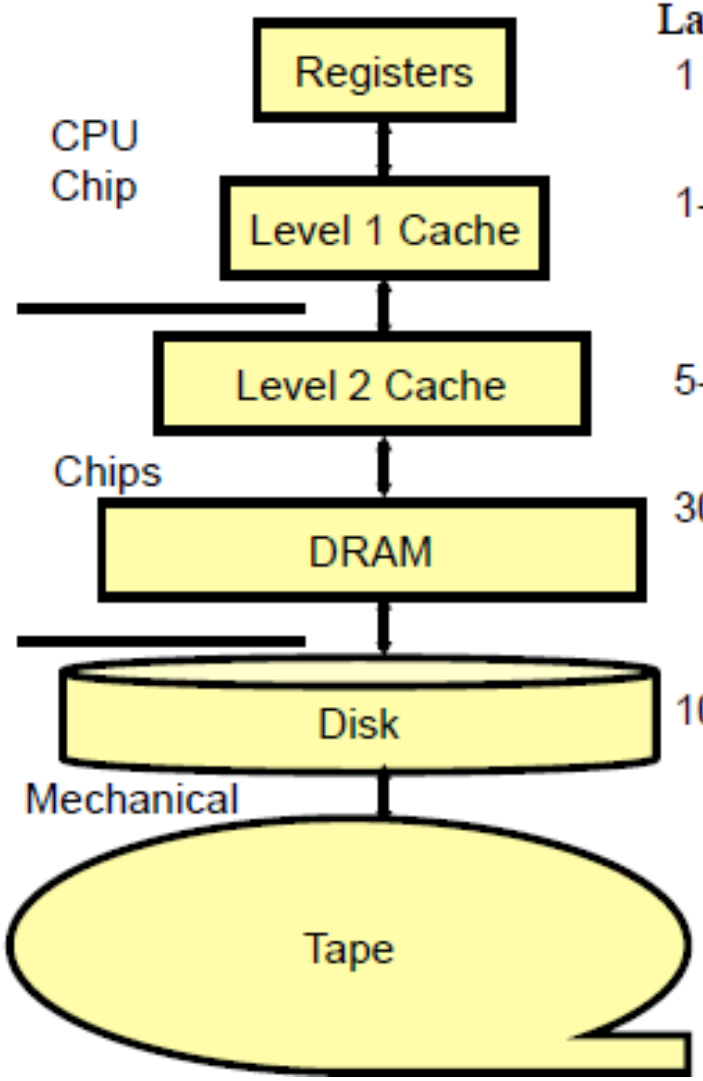
Memory Optimization

Amil Ahmad Ilham

Computer Organization



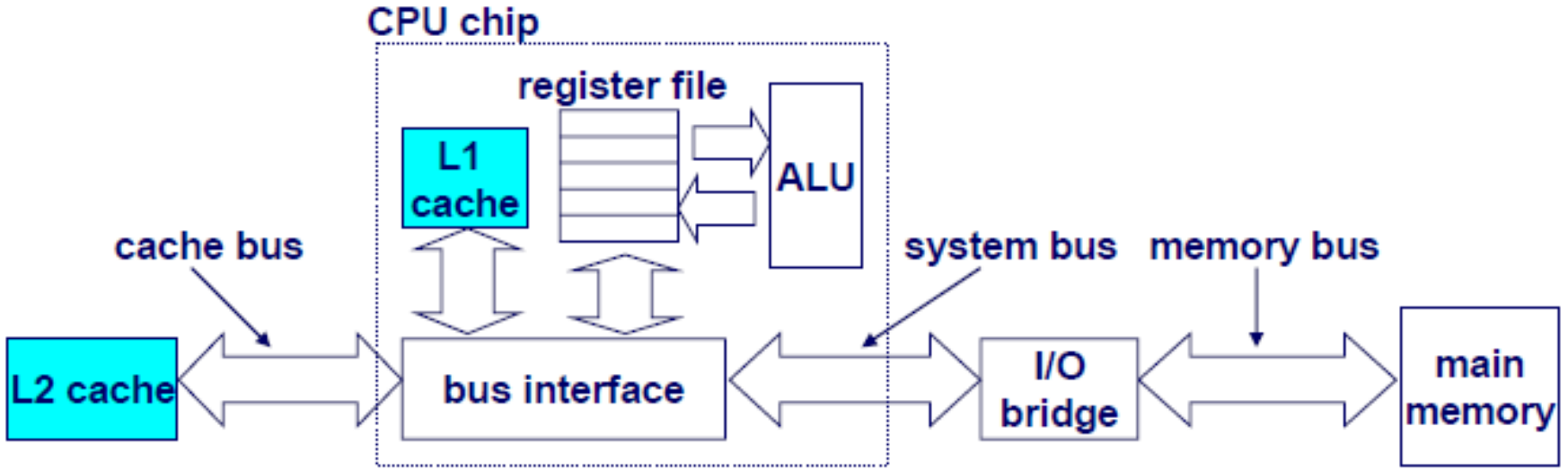
The Memory Hierarchy



	Latency	Bandwidth	
Registers	1 cyc	3-10 words/cycle < 1KB	compiler managed
Level 1 Cache	1-3cy	1-2 words/cycle 32KB -1MB	hardware managed
Level 2 Cache	5-10cy	1 word/cycle 1MB - 4MB	hardware managed
DRAM	30-100cy	0.5 words/cycle 64MB - 4GB	OS managed
Disk	10^6-10^7cy	0.01 words/cycle 4GB+	OS managed

Memory in Computer System

- Cache memories are small, fast SRAM-based memories managed automatically in hardware.
- CPU looks first for data in L1, then in L2, then in main memory.



Cache Terminology

- *Instruction cache* – cache that only holds instructions.
- *Data cache* – cache that only caches data.
- *Unified cache* – cache that holds both instructions and data

Cache Performance Metrics

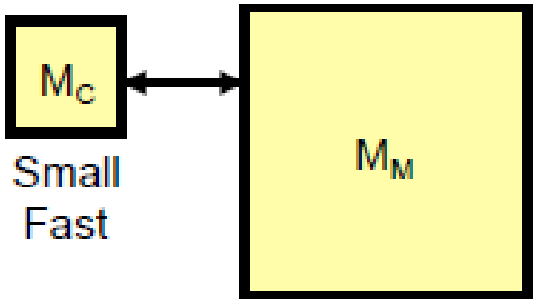
- Cache hit: memory references found in the cache
  - Hit time: time to deliver a line in the cache to the processor (includes time to determine whether the line is in the cache).
- Cache miss: memory references not found in the cache
  - Miss rate: fraction of memory references not found in the cache (misses/references)
- Miss penalty: additional time required because of a miss (cycles to service a miss)

Classification of cache misses (3Cs)

- Classifying misses by causes (3Cs) – Hill and Smith's classification
  - Compulsory—To bring blocks into cache for the first time. Also called cold start misses or first reference misses.
    (Misses in even an Infinite Cache)
  - Conflict—For set associative or direct mapped caches, blocks can be discarded and later retrieved if too many blocks map to its set. Also called collision misses or interference misses.
    (Misses in N-way Associative, Size X Cache)
  - Capacity—Cache is not large enough such that some blocks are discarded and later retrieved.
    (Misses in Fully Associative Size X Cache)

Average Memory Access Time (AMAT)

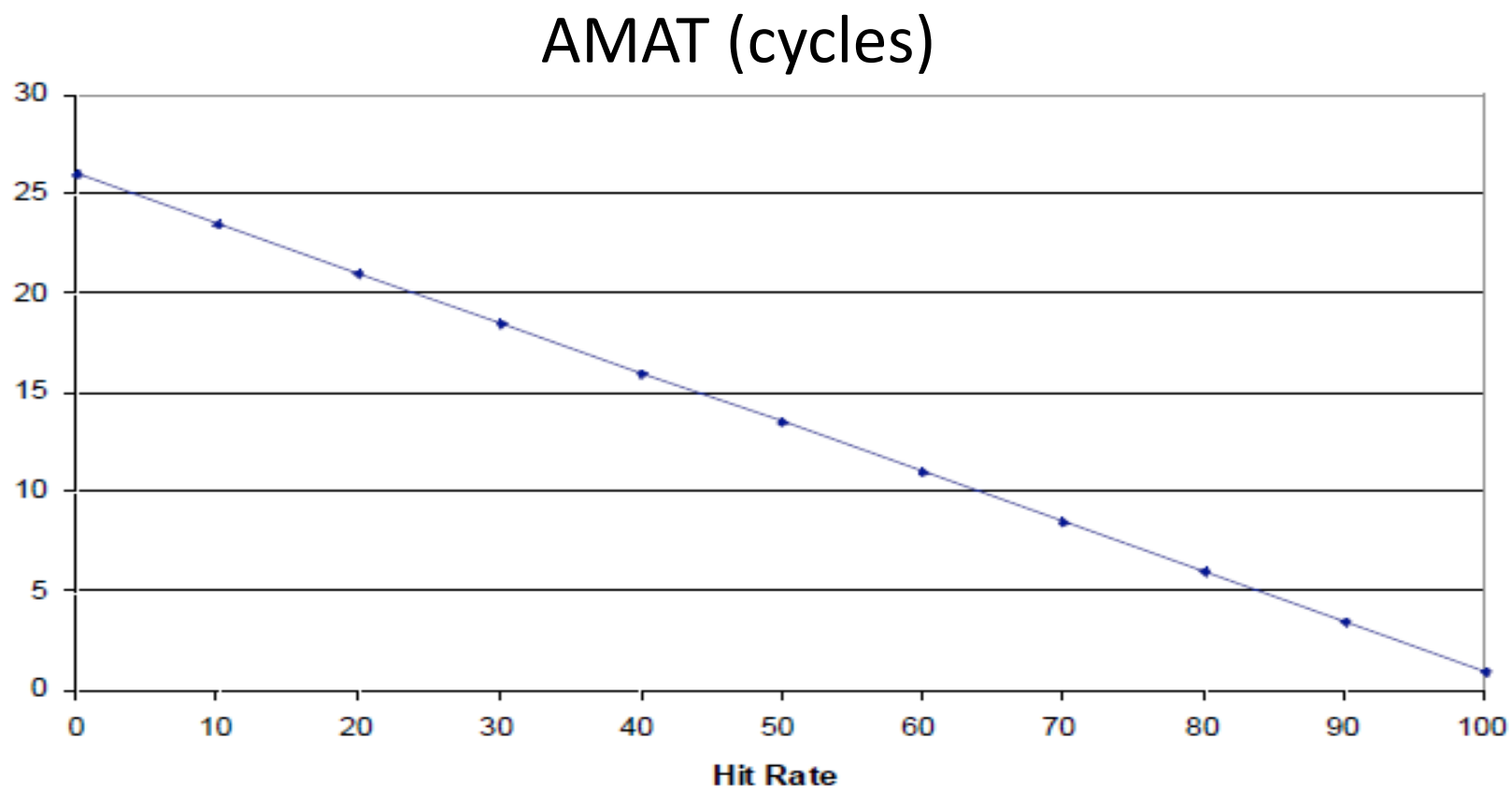
- What is the average memory access time (AMAT) in clock cycles if:
  - 70% of references hit in cache
  - Cache hits take one clock cycle
  - Main memory references take 25 clock cycles



Average access time =
$HitTime_{L1} + MissRate_{L1} * MissPenalty_{L1}$

Impact of hit rate

- Cache hit takes 1 cycle
- Main memory references take 25 cycles



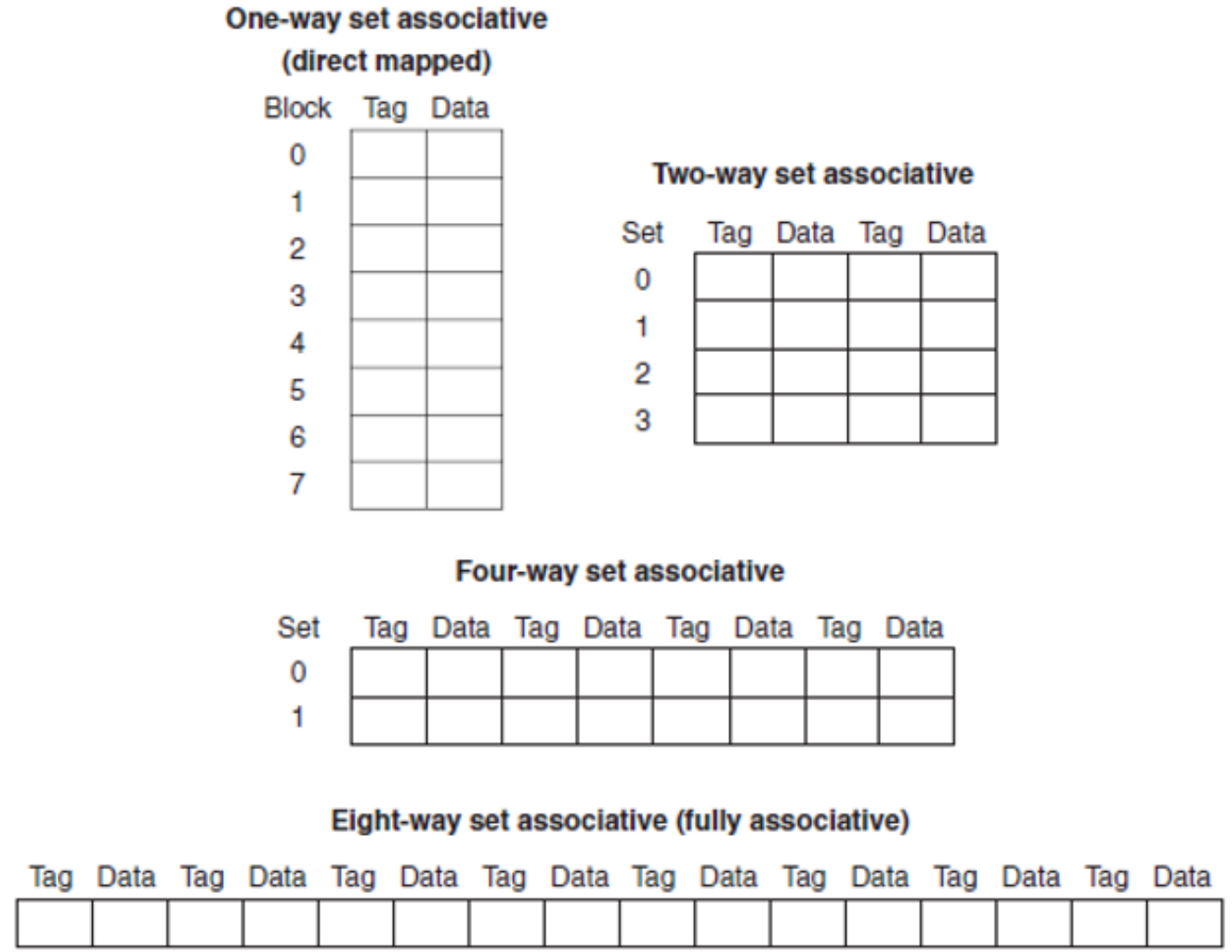
Memory Locality

- The memory access of programs present different patterns and memory hierarchies try to explore the memory locality
- Memory locality is the principle that future memory accesses are ***near*** past accesses
- There are two types of memory locality:
  - **Spatial Locality**: near in space/distance.
    - The concept that likelihood of referencing a resource is higher if a resource near it (close address) was just referenced
  - **Temporal Locality**: near in time
    - The concept that a resource that is referenced at one point in time will be referenced again sometime in the near future
- *Why does code have locality?*
  - repeating loops
  - accessing elements of an array/struct

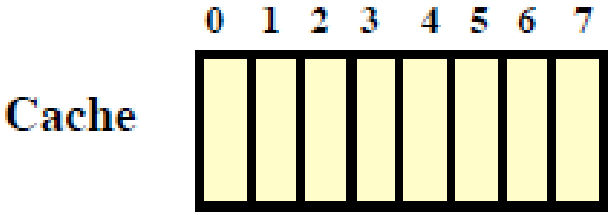
Cache architecture

Cache Architecture

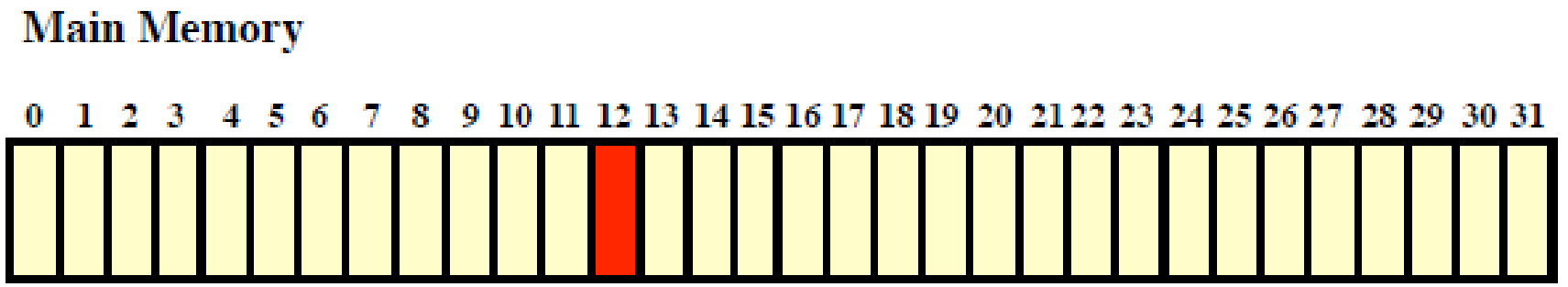
Eight-block cache configured as direct mapped, two-way set associative, four-way set associative, and fully associative



Where does the data go in the cache?



- Word = 4 bytes
- Block = 1 word

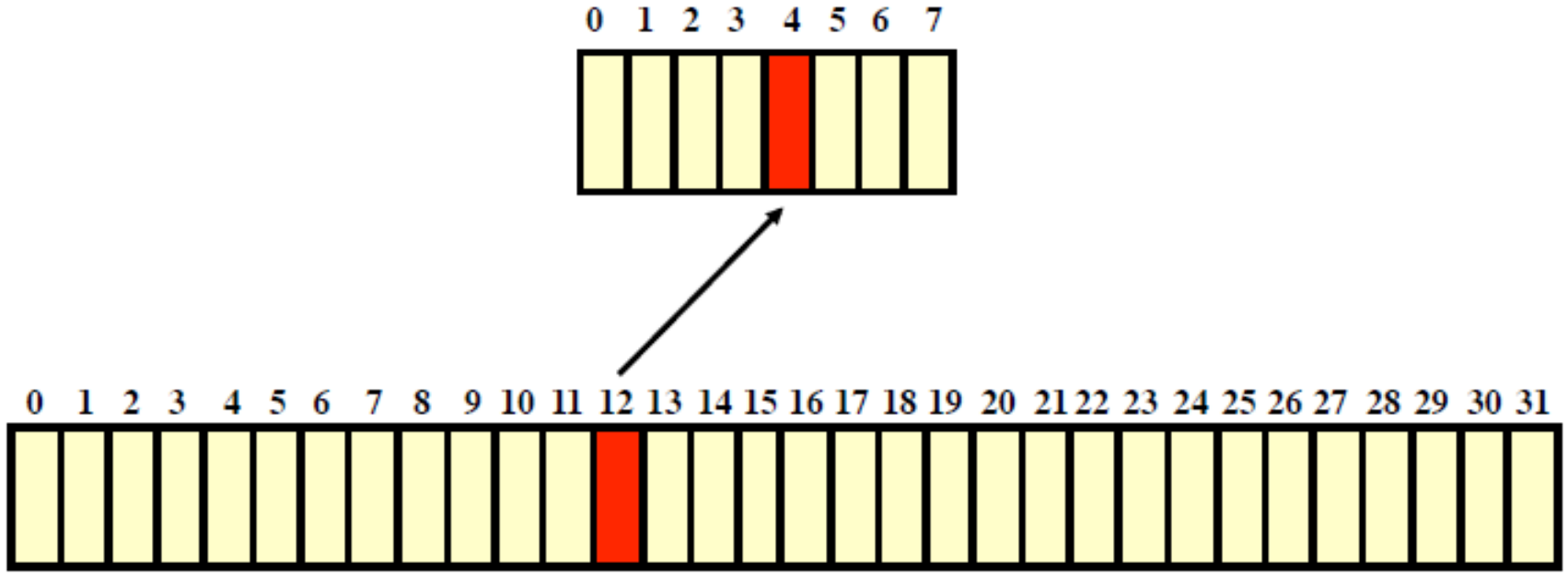


- Where do we put block 12?

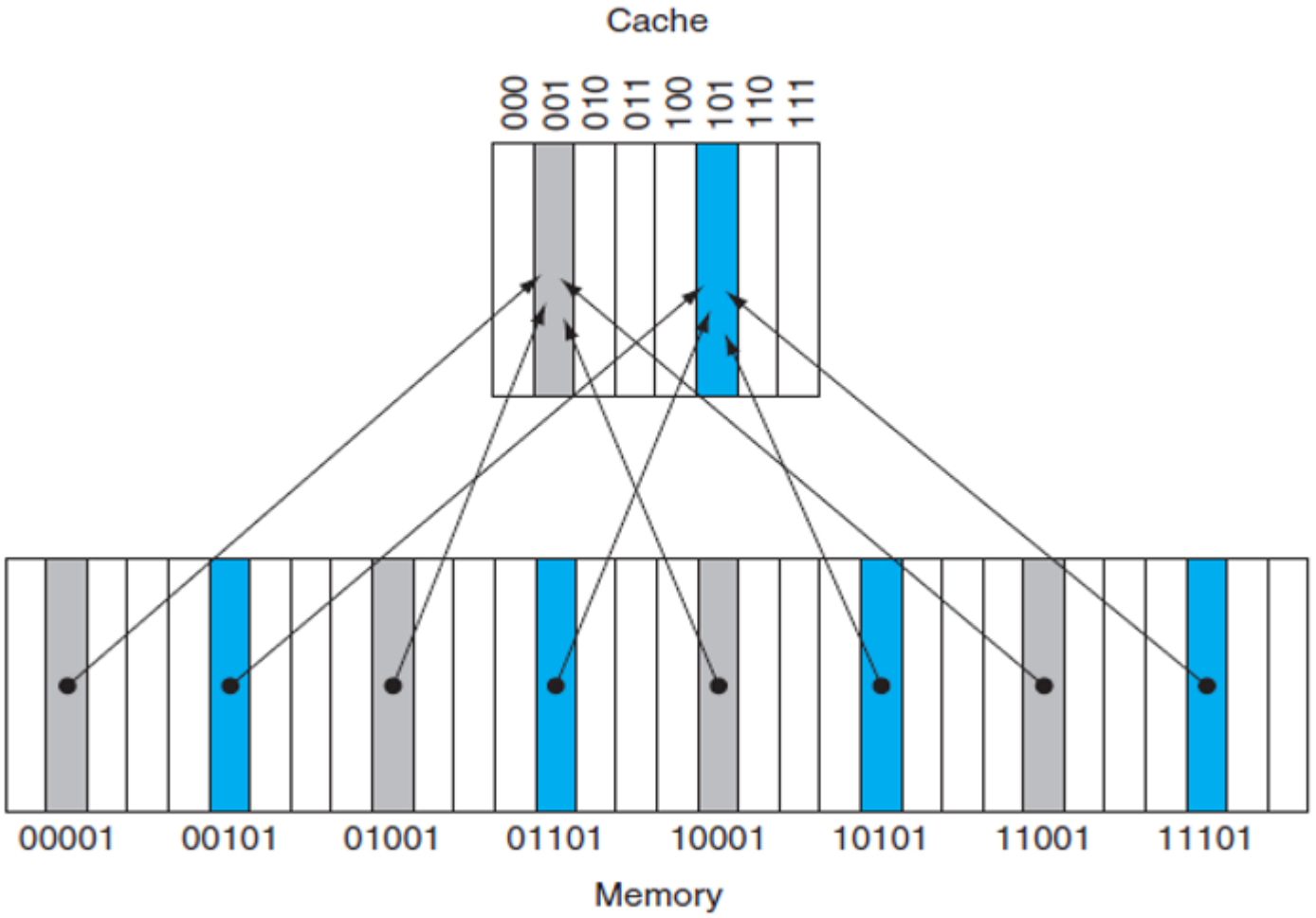
Eight-block cache configured as direct mapped (one-way set associative)

- **Each block mapped to exactly 1 cache location**

Cache location = (block address) MOD (# blocks in cache)



Eight-block direct mapped cache
(Eight-block one-way set associative cache)



Eight-block direct mapped cache (Eight-block one-way set associative cache)

- The initial state of the cache after power-on

Index	V	Tag	Data
000	0		
001	0		
010	0		
011	0		
100	0		
101	0		
110	0		
111	0		

Eight-block direct mapped cache (Eight-block one-way set associative cache)

Misalkan processor membutuhkan data dari RAM pada alamat (dalam desimal): 22,26,22,26,16,3,16,18,26,18. Dengan menggunakan *eight-block direct mapped cache*, Gambarkan tabel yang menunjukkan bagaimana data-data di alamat tersebut dipindahkan dari RAM ke cache.

1. 22 -> 10110
2. 26 -> 11010
3. 22 -> 10110
4. 26 -> 11010
5. 16 -> 10000
6. 3 -> 00011
7. 16 -> 10000
8. 18 -> 10010
9. 26 -> 11010
10. 18 -> 10010

Index	V	Tag	Data
000	0		
001	0		
010	0		
011	0		
100	0		
101	0		
110	0		
111	0		

Eight-block direct mapped cache (Eight-block one-way set associative cache)

- Misalkan processor membutuhkan data dari RAM pada alamat (dalam desimal): 22,26,22,26,16,3,16,18,26,18. Dengan menggunakan *eight-block direct mapped cache*, Gambarkan tabel yang menunjukkan bagaimana data-data di alamat tersebut dipindahkan dari RAM ke cache.

1. 22 -> 10110 : Com M
2. 26 -> 11010 : Com M
3. 22 -> 10110 : HIT
4. 26 -> 11010 : HIT
5. 16 -> 10000 : Com M
6. 3 -> 00011 : Com M
7. 16 -> 10000 : HIT
8. 18 -> 10010 : Com M
9. 26 -> 11010 : Conf M
10. 18 -> 10010: Conf M

Index	V	Tag	Data
000	1	10	M[10000]
001	0		
010	1	~~11~~ ~~10~~ ~~11~~ 10	~~M[11010]~~ ~~M[10010]~~ ~~M[11010]~~ M[10010]
011	1	00	M[00011]
100	0		
101	0		
110	1	10	M[10110]
111	0		