Cache Memories

Model of Memory Hierarchy

Increasing Speed

SRAM

Physical Memory

DRAM

Virtual Memory

Reg File

L1 Data cache

L1 Inst cache

L2 Cache

Main Memory

DISK

Increasing Capacity

- A memory system has a cache, a main memory, and a virtual memory. If the hit rate in the cache is 98% and the hit rate in the main memory is 99%, what is the average memory access time if it takes 2 cycles to access the cache, 150 cycles to fetch a line from main memory, and 100,000 cycles to access the virtual memory?

Four Central Questions in Designing a Cache

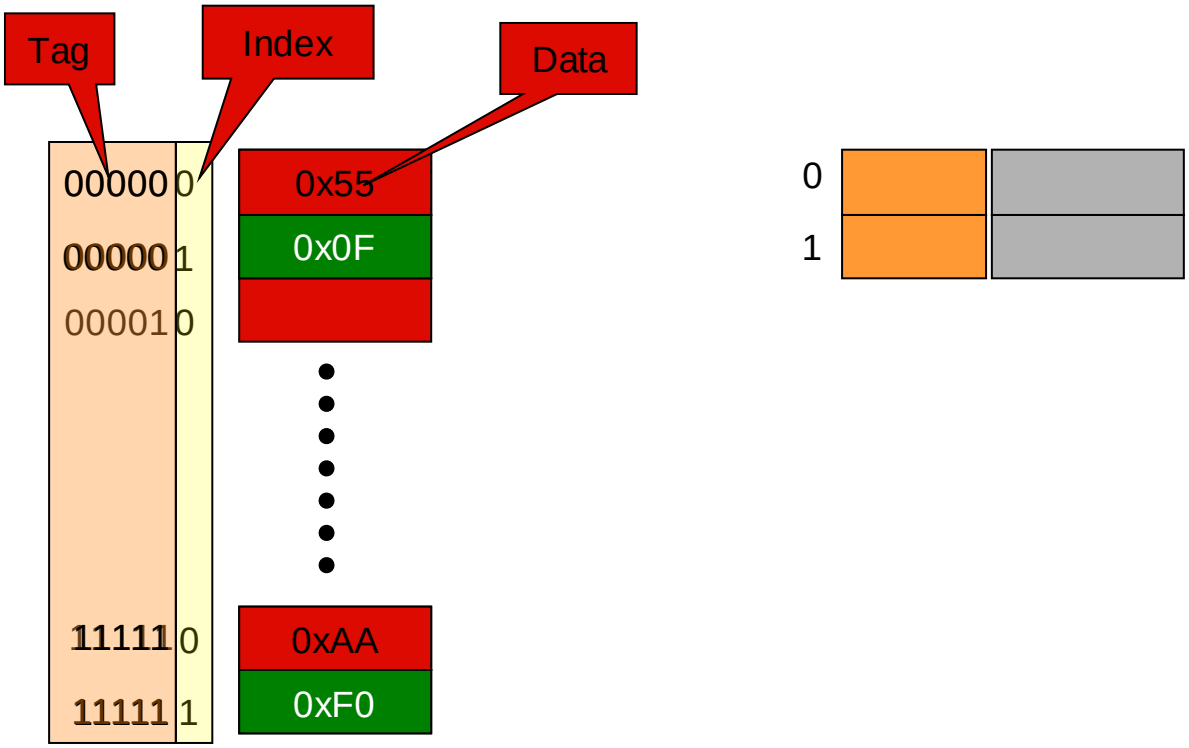
- P-I-R-W:
  - ✓ placement: where can a block of memory go?
  - ✓ identification: how do i find a block of memory?
  - ✓ replacement: how do i make space for new blocks?
  - ✓ write policy: how do i propagate changes?
- need to consider these for all levels of the memory hierarchy
  - ✓ L1/L2/L3 caches now
- main memory, disks have similar issues, addressed later

Types of Caches

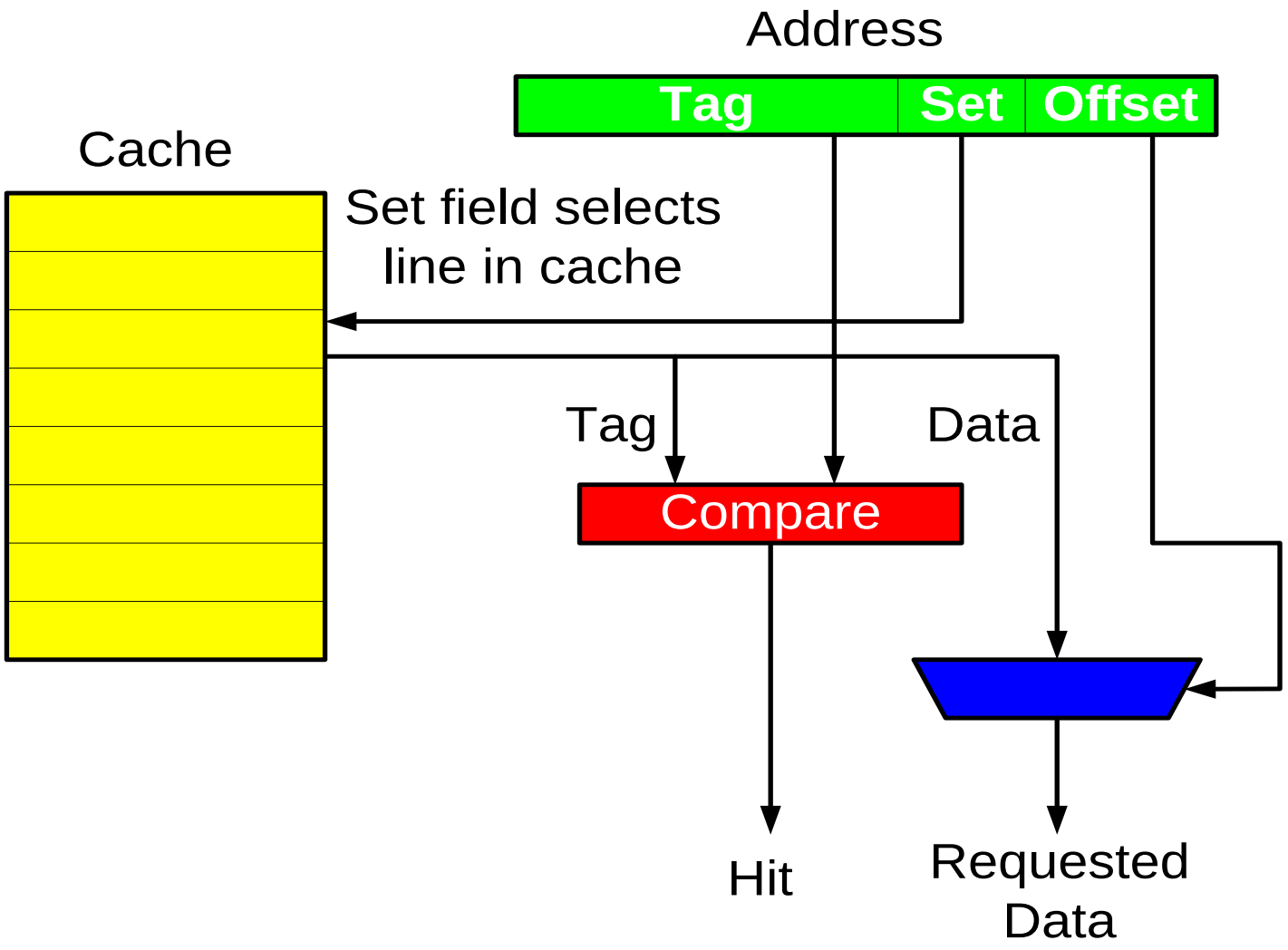
type of cache	mapping of data from memory to cache	complexity of searching the cache
direct mapped (DM)	a memory value can be placed at **a single corresponding location** in the cache	fast indexing mechanism
set-associative (SA)	a memory value can be placed in **any of a set of locations** in the cache	slightly more involved search mechanism
fully-associative (FA)	a memory value can be placed in **any location** in the cache	extensive hardware resources required to search (CAM)

Direct Mapping

- Direct mapping:
  - A memory value can only be placed at a single corresponding location in the cache

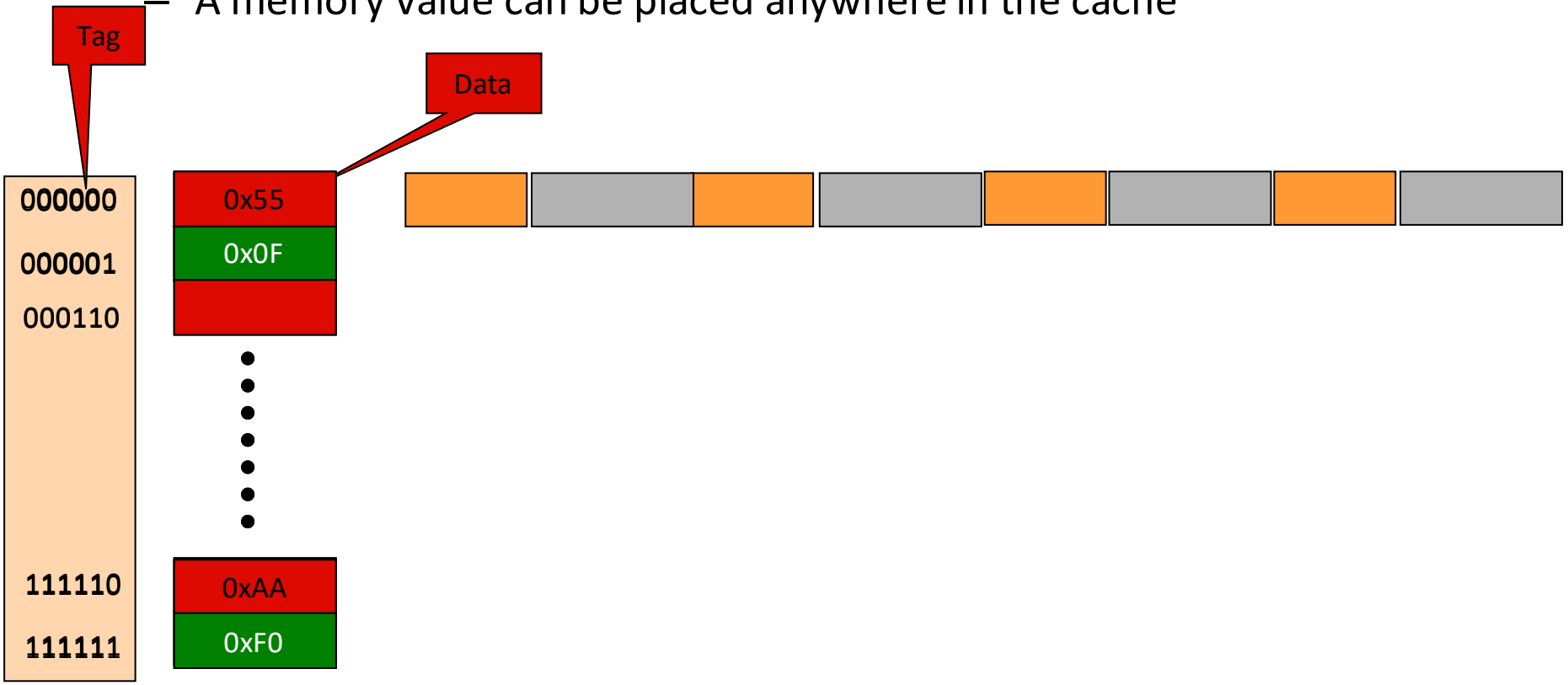


Direct-Mapped:
One cache location for each address

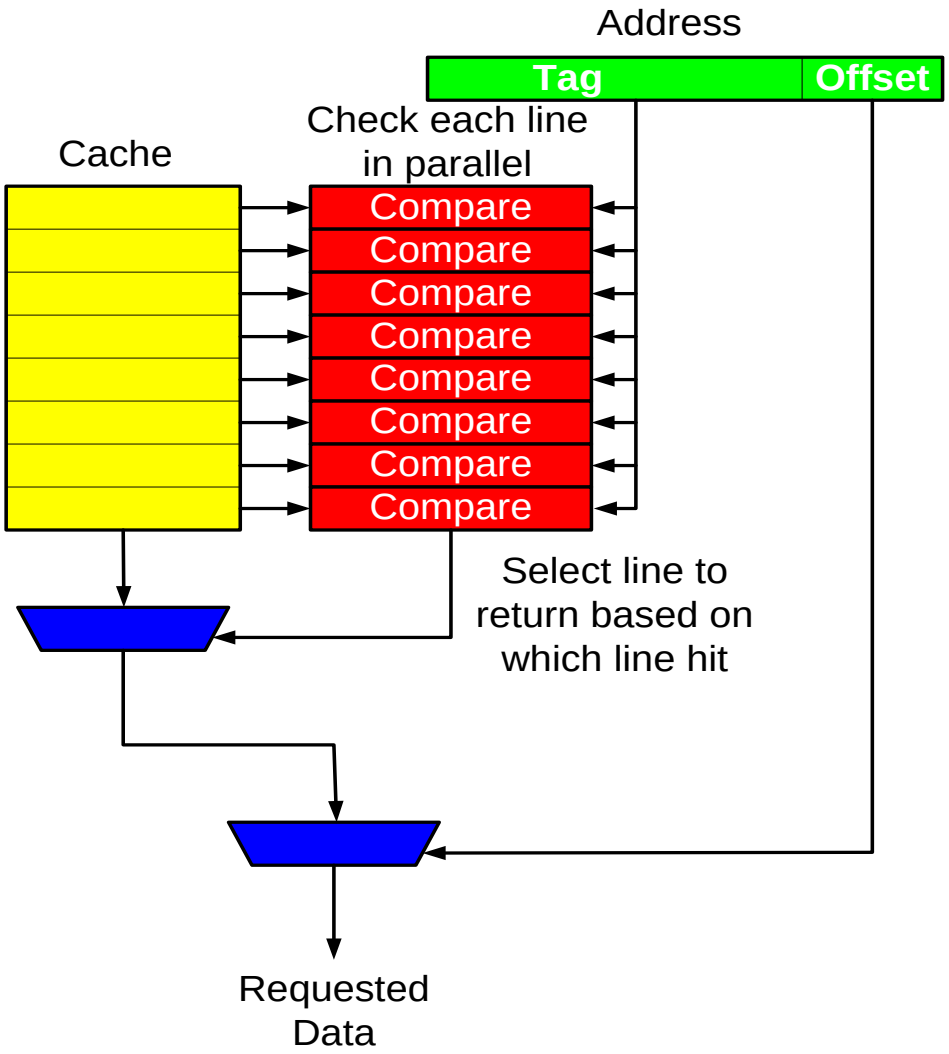


Fully Associative Mapping

- Fully-associative mapping:
  - A memory value can be placed anywhere in the cache

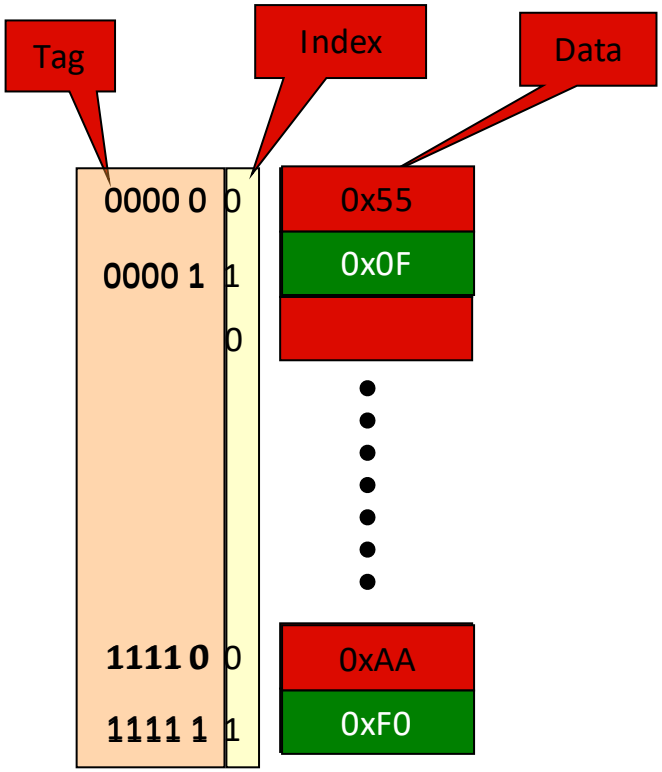


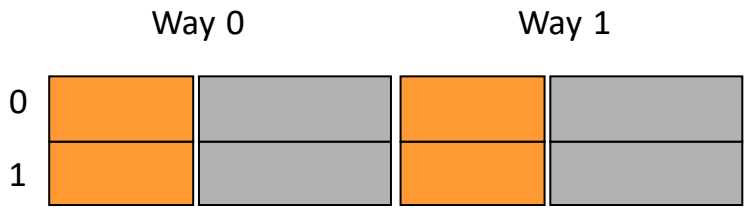
Fully-Associative: Anything Can Go Anywhere



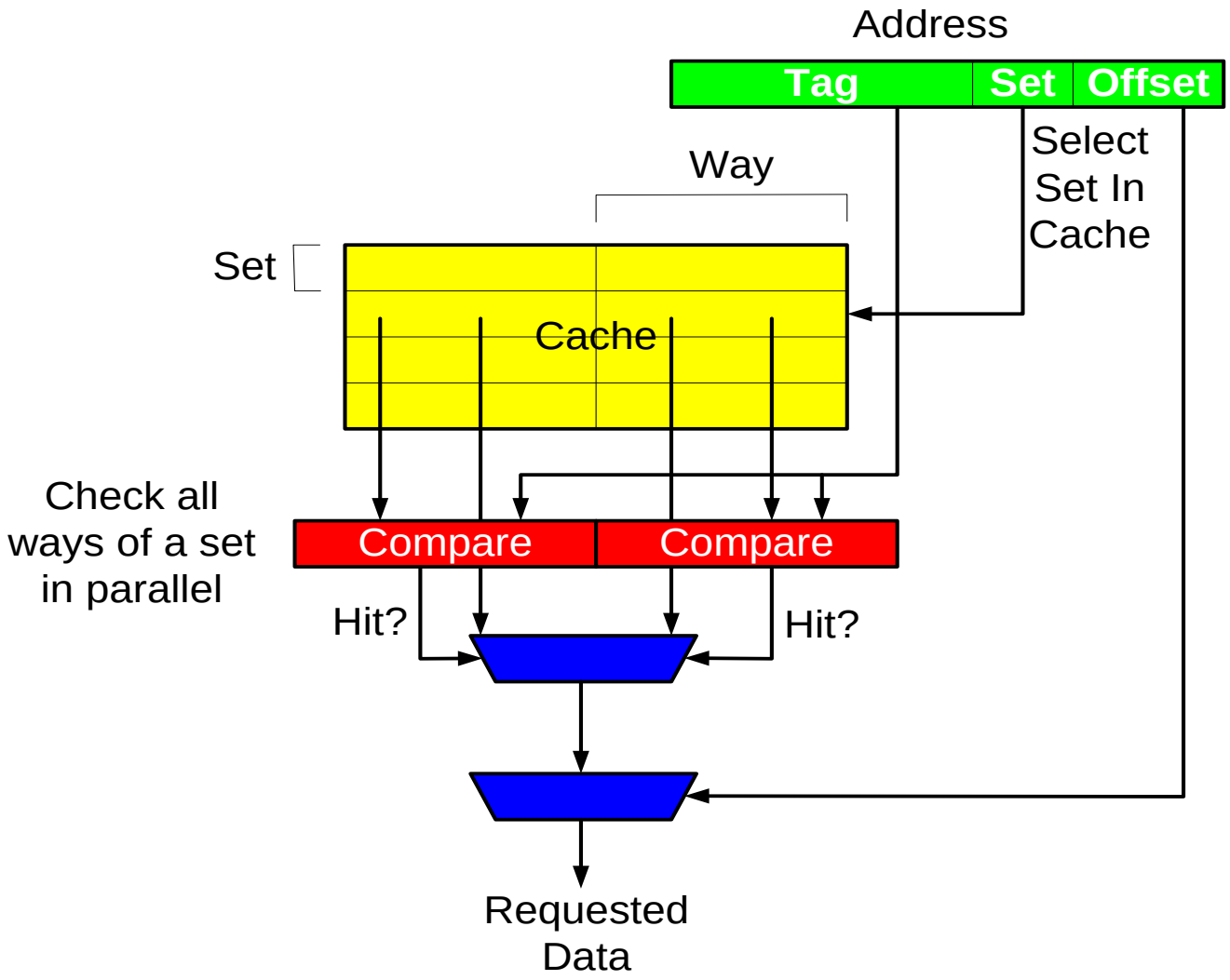
Set Associative Mapping (2-Way)

- Set-associative mapping:
  - A memory value can be placed in any location of a set in the cache



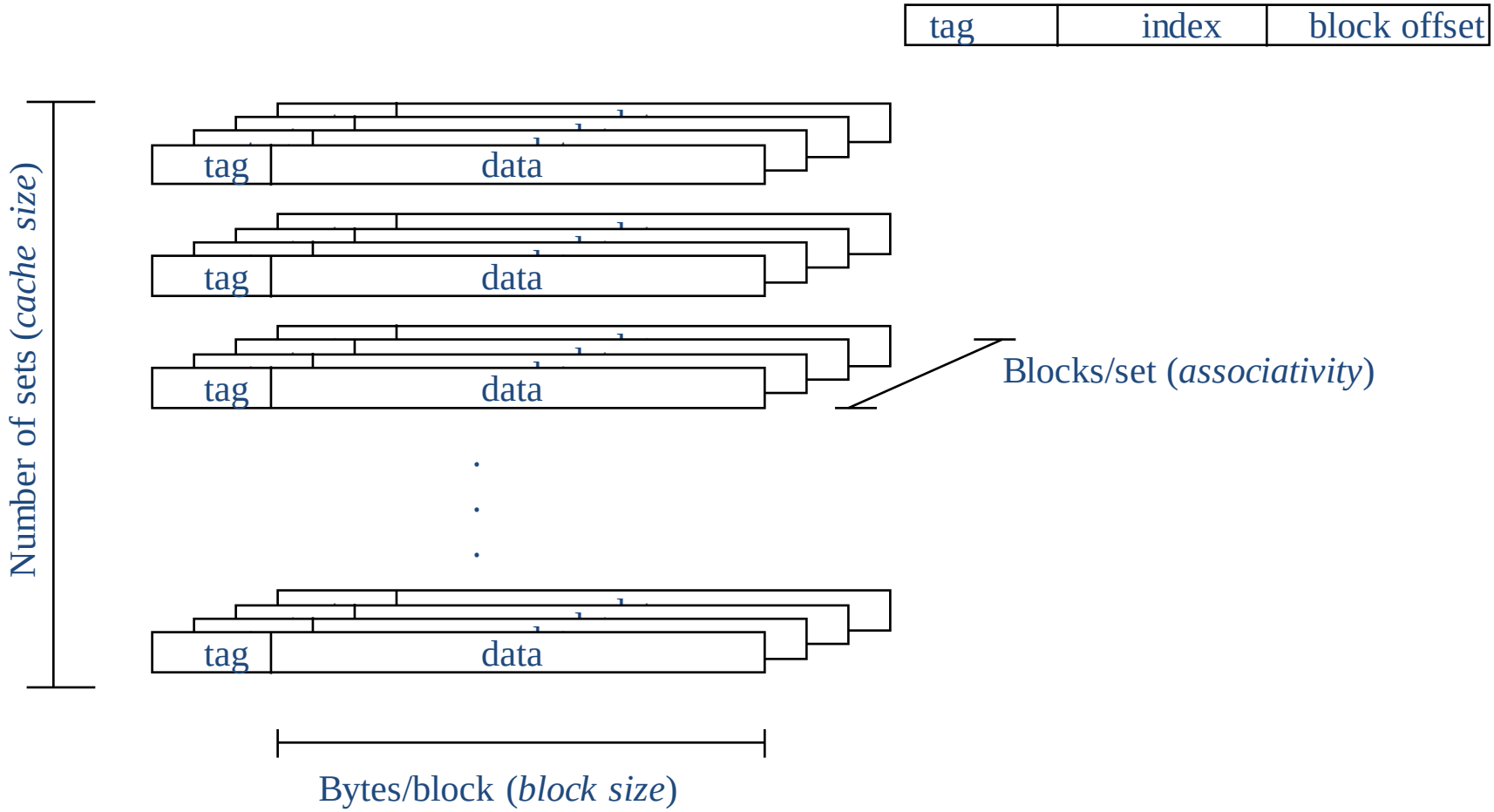


Compromise: Set-Associative Caches

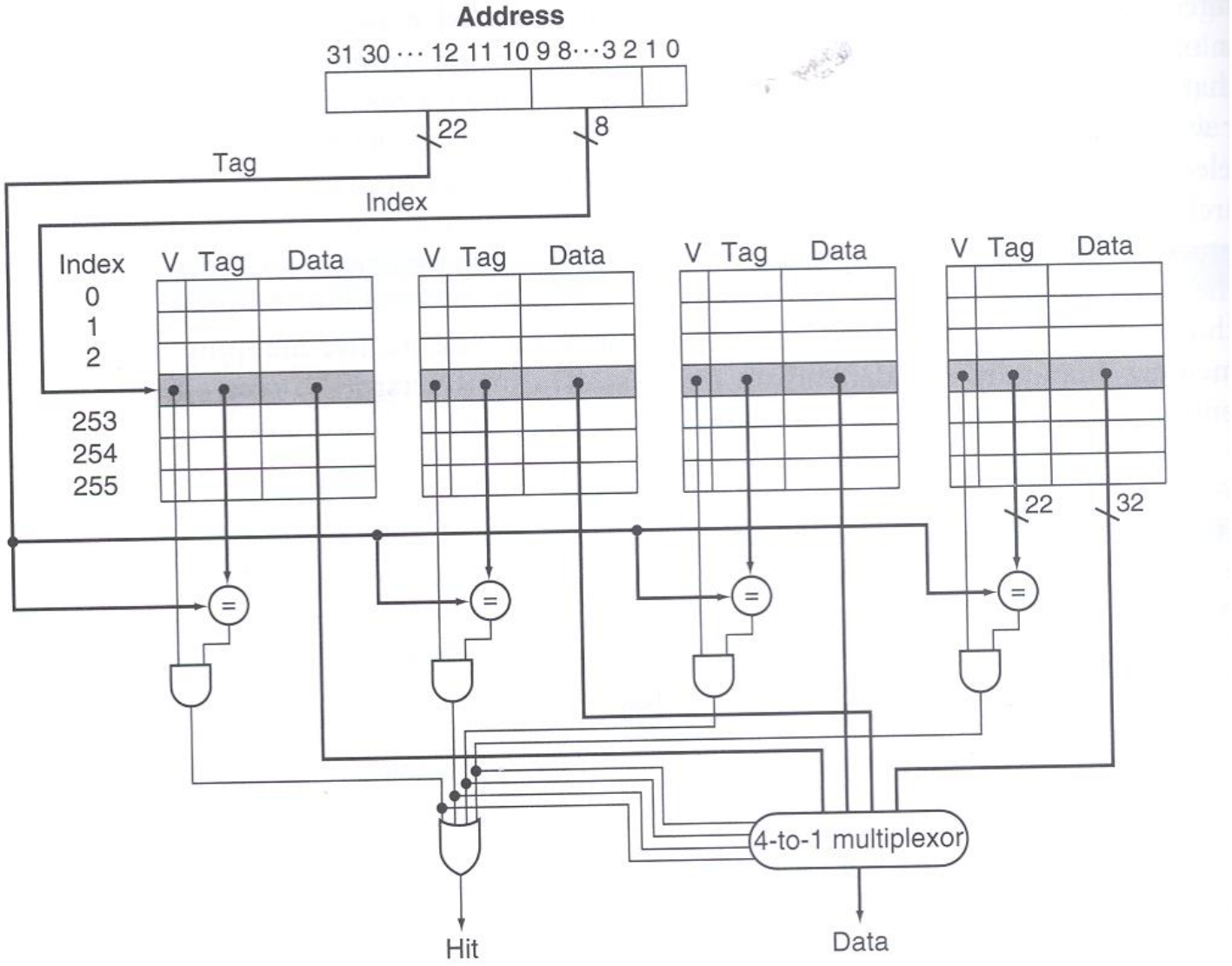


Cache Organization -- Recap

- A typical cache has three dimensions

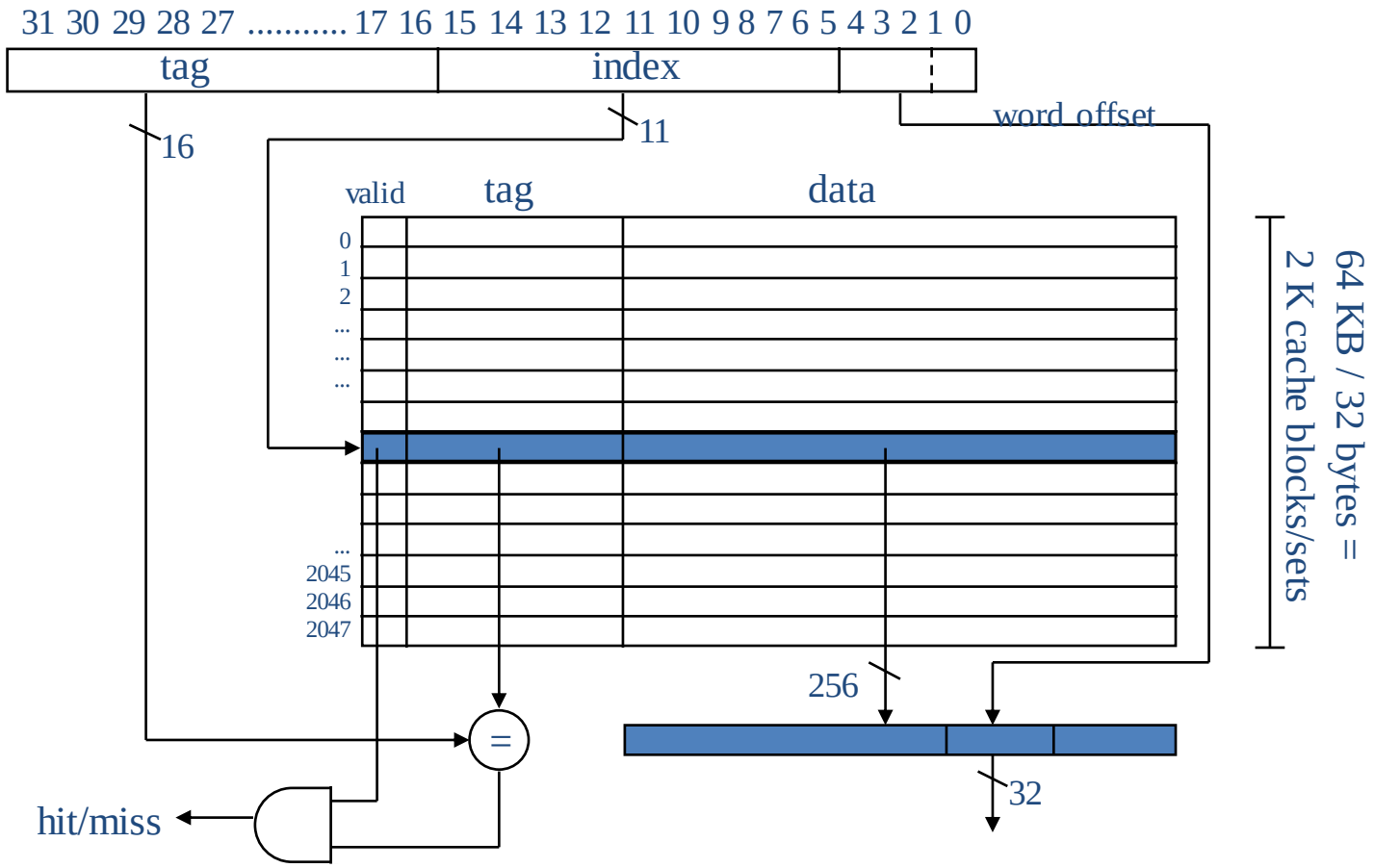


A 4-way set associative cache



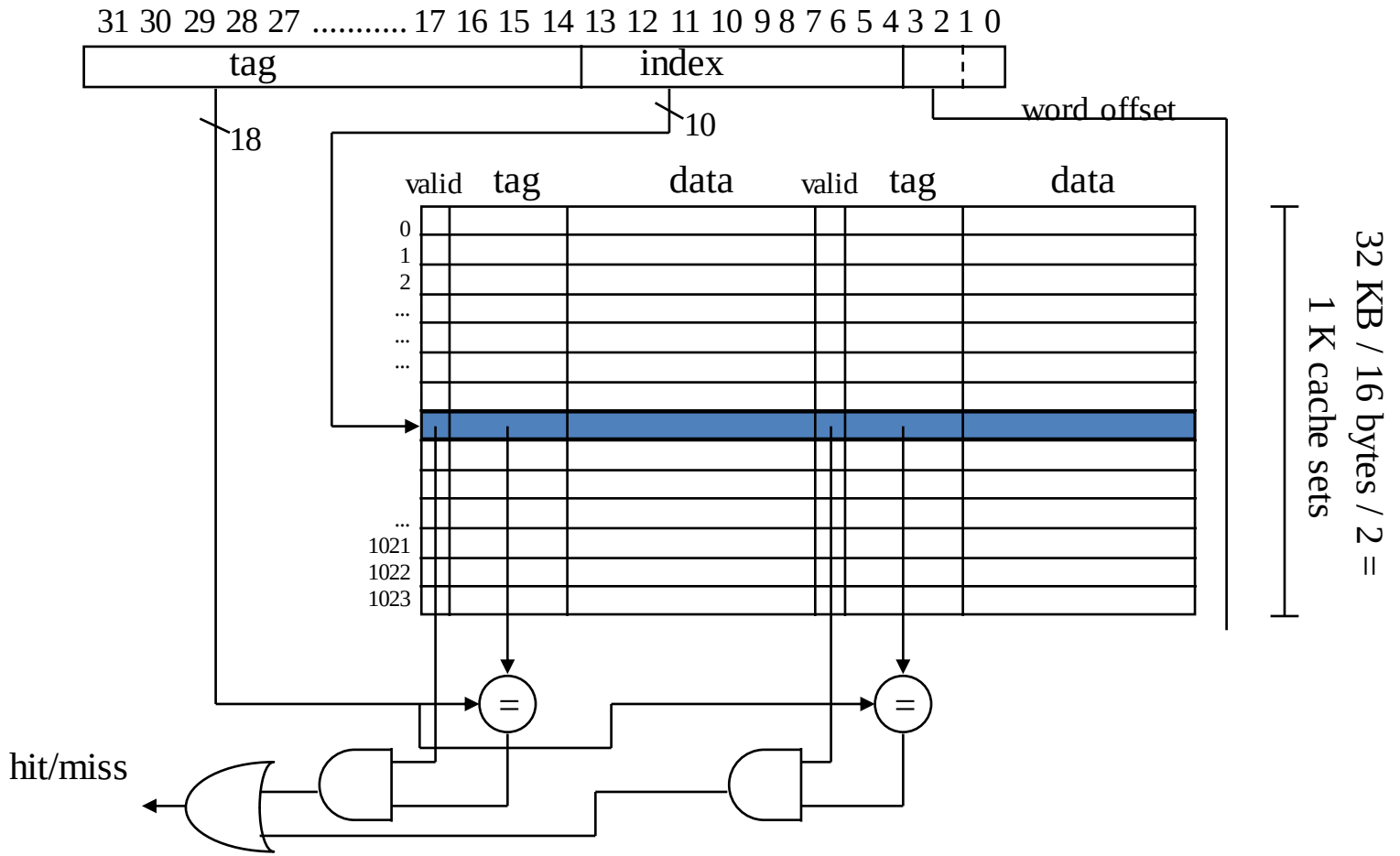
Putting it all together

64 KB cache, direct-mapped, 32-byte cache block



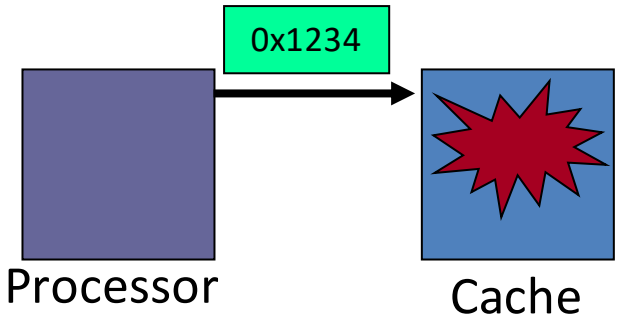
A set associative cache

32 KB cache, 2-way set-associative, 16-byte blocks



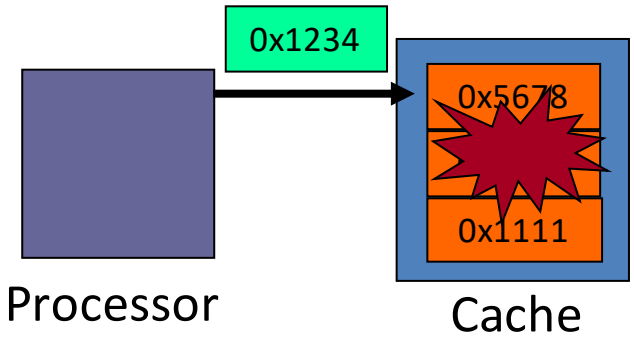
Three Cs (Cache Miss Terms)

- Compulsory Misses:
  - Cold start misses (caches do not have valid data at the start of the program)



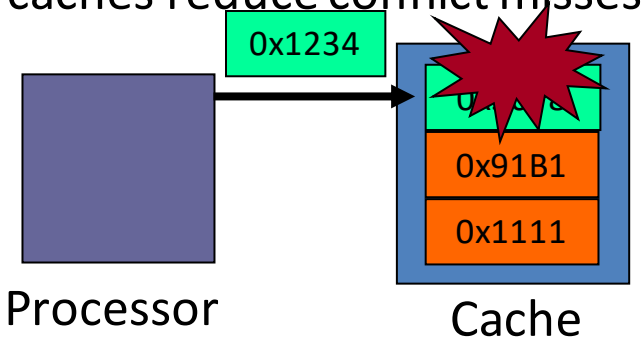
Three Cs (Cache Miss Terms)

- Capacity Misses:
  - Increase cache size



Three Cs (Cache Miss Terms)

- Conflict Misses:
  - Increase cache size and/or associativity.
  - Associative caches reduce conflict misses



Cache Parameters

Cache size = Number of sets * block size * associativity

128 blocks, 32-byte blocks, direct mapped, size = ?

128 KB cache, 64-byte blocks, 512 sets, associativity = ?

Sample Problem

A cache has a capacity of 32 KB and 256-byte lines. On a machine with a 32-bit virtual address space, how many bits long are the tag, set, and offset fields for

- A direct-mapped implementation?
- A four-way set-associative implementation?
- A fully-associative implementation?