20. Cache Organization: Set-associative caches & 3C's of caches

EECS 370 – Introduction to Computer Organization – Winter 2023

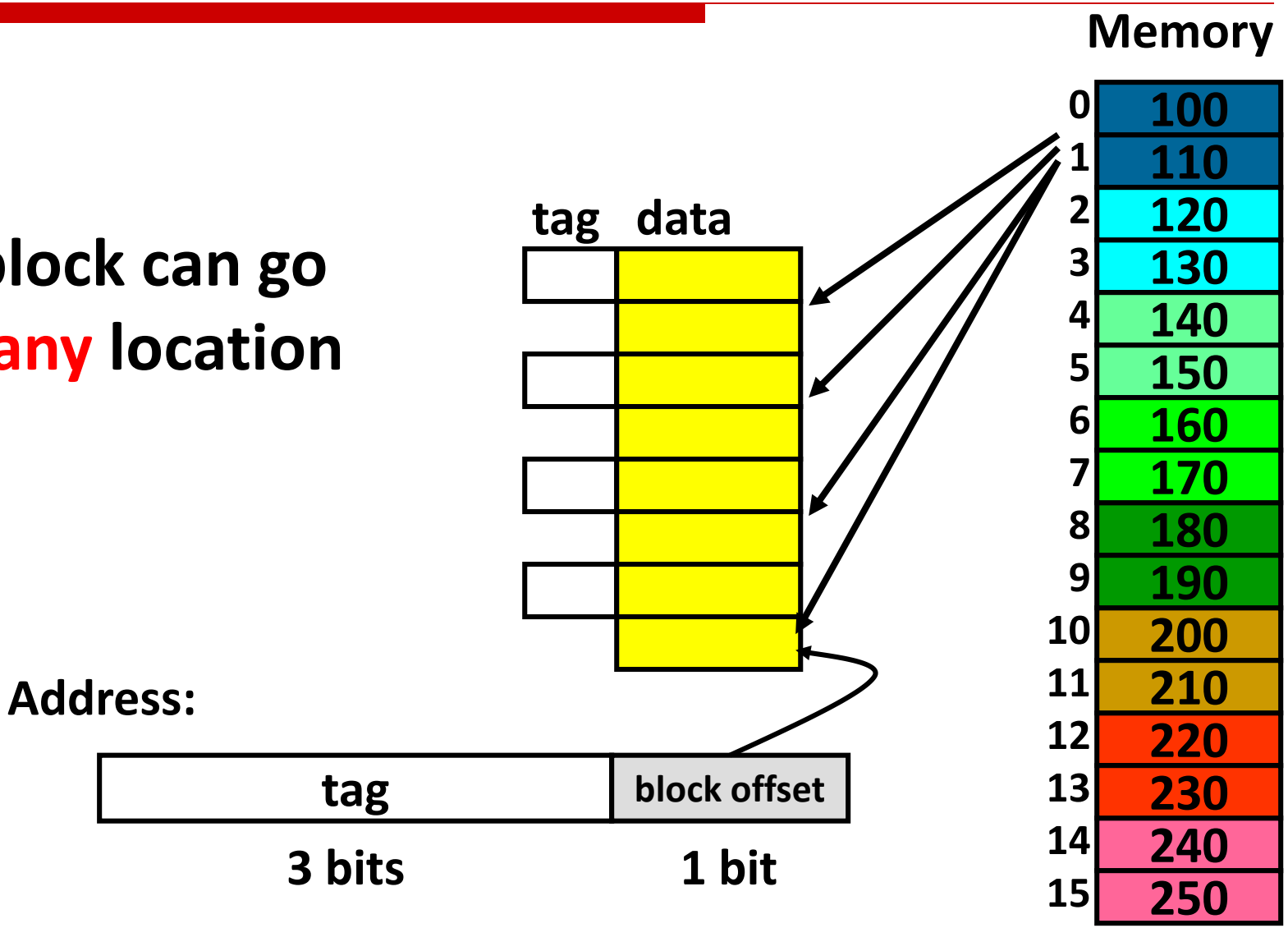
EECS Department
University of Michigan in Ann Arbor, USA

Announcements and checking in

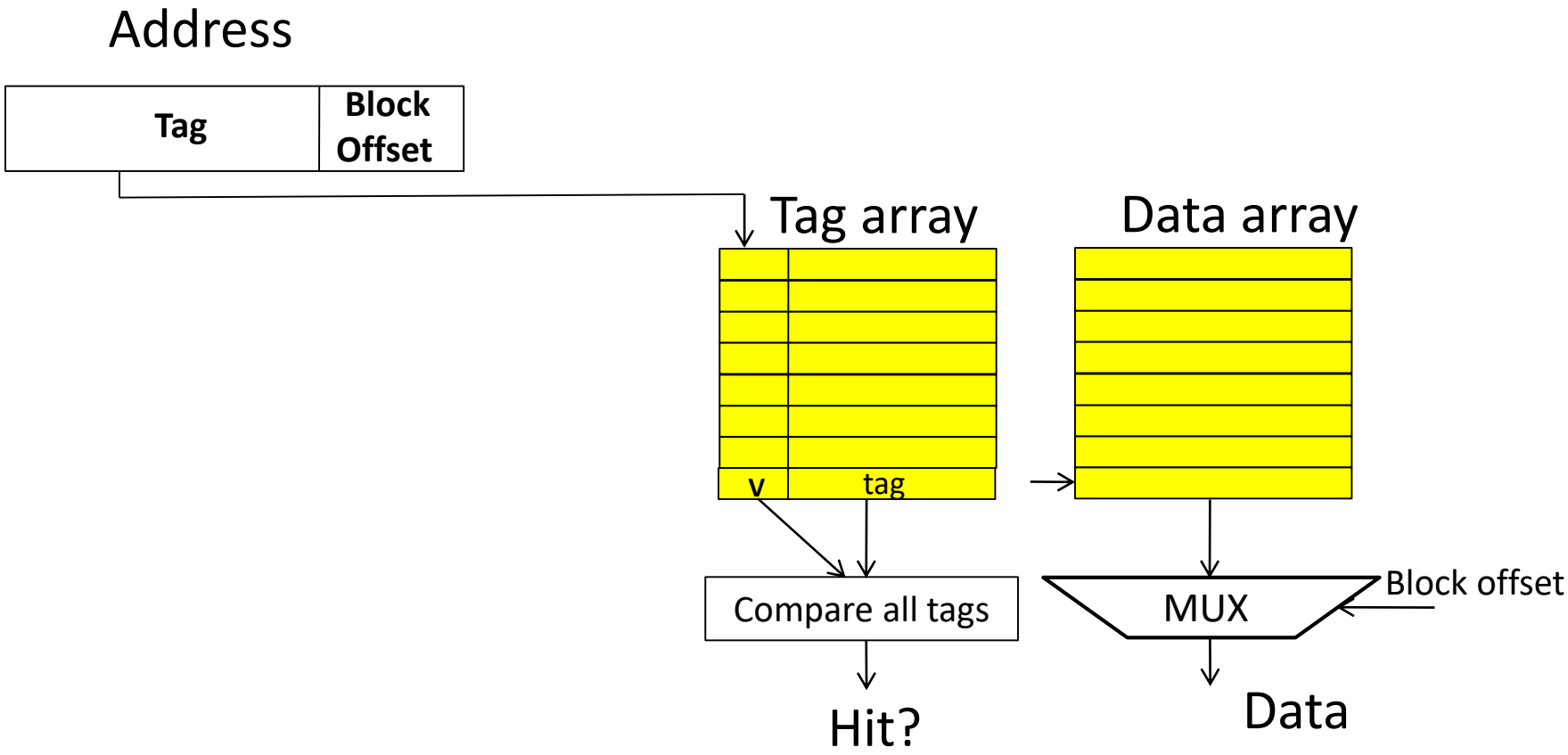
- Project 3 is due today
- Homework 5 is due Monday April 3rd

Review: Fully-associative caches

A block can go to any location

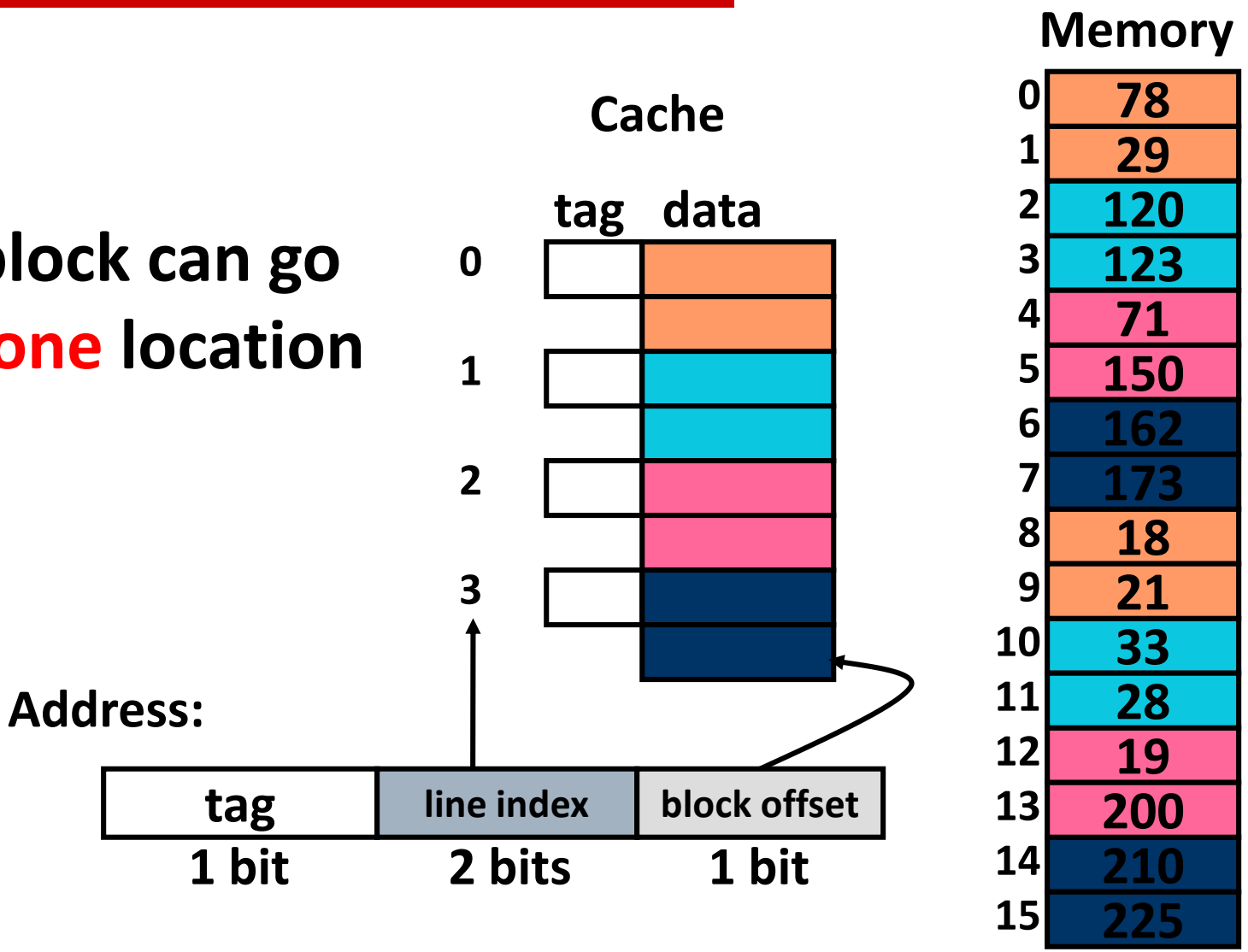


Fully-associative cache: Placement & Access

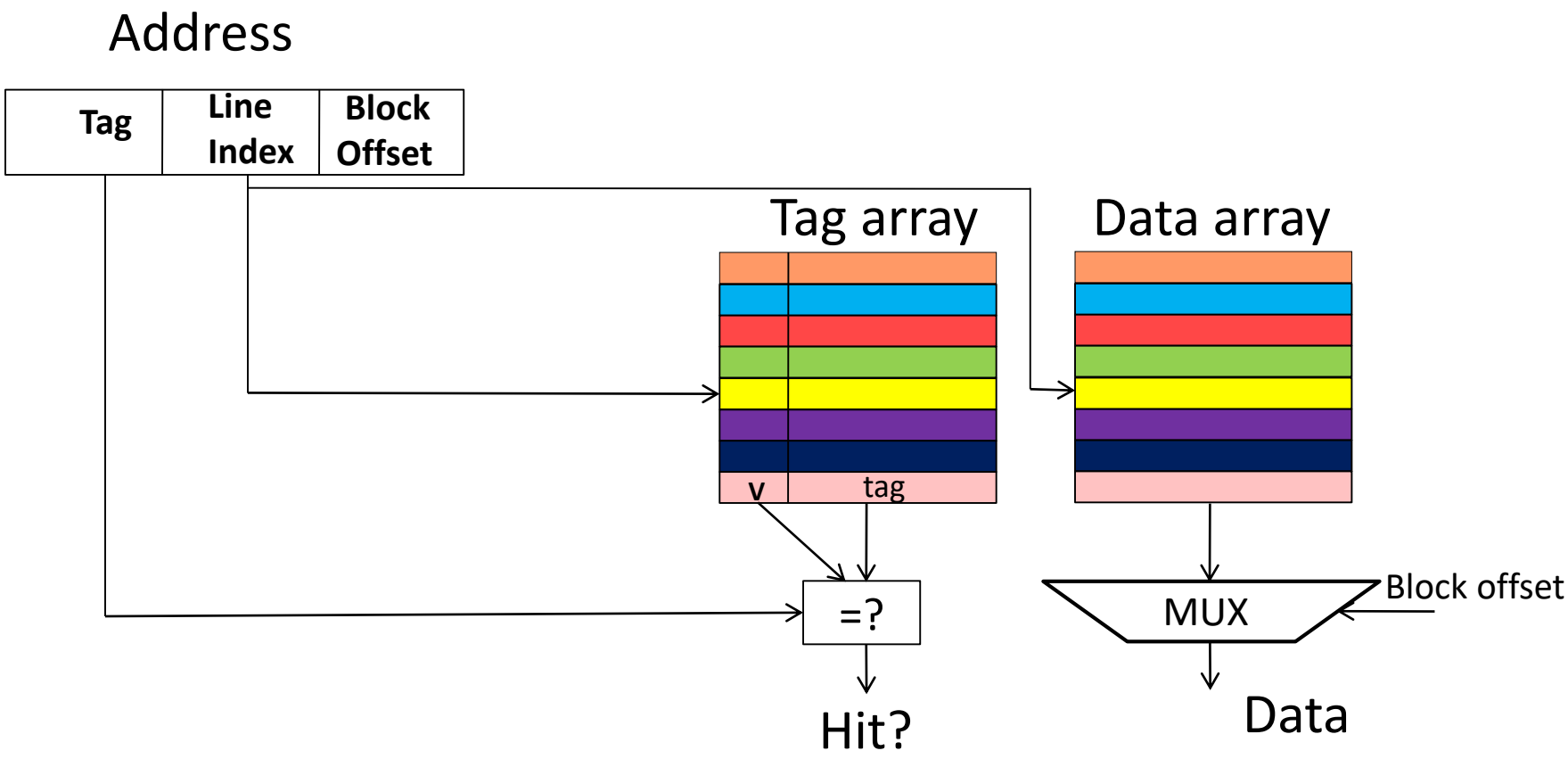


Review: Direct-mapped caches

A block can go to one location



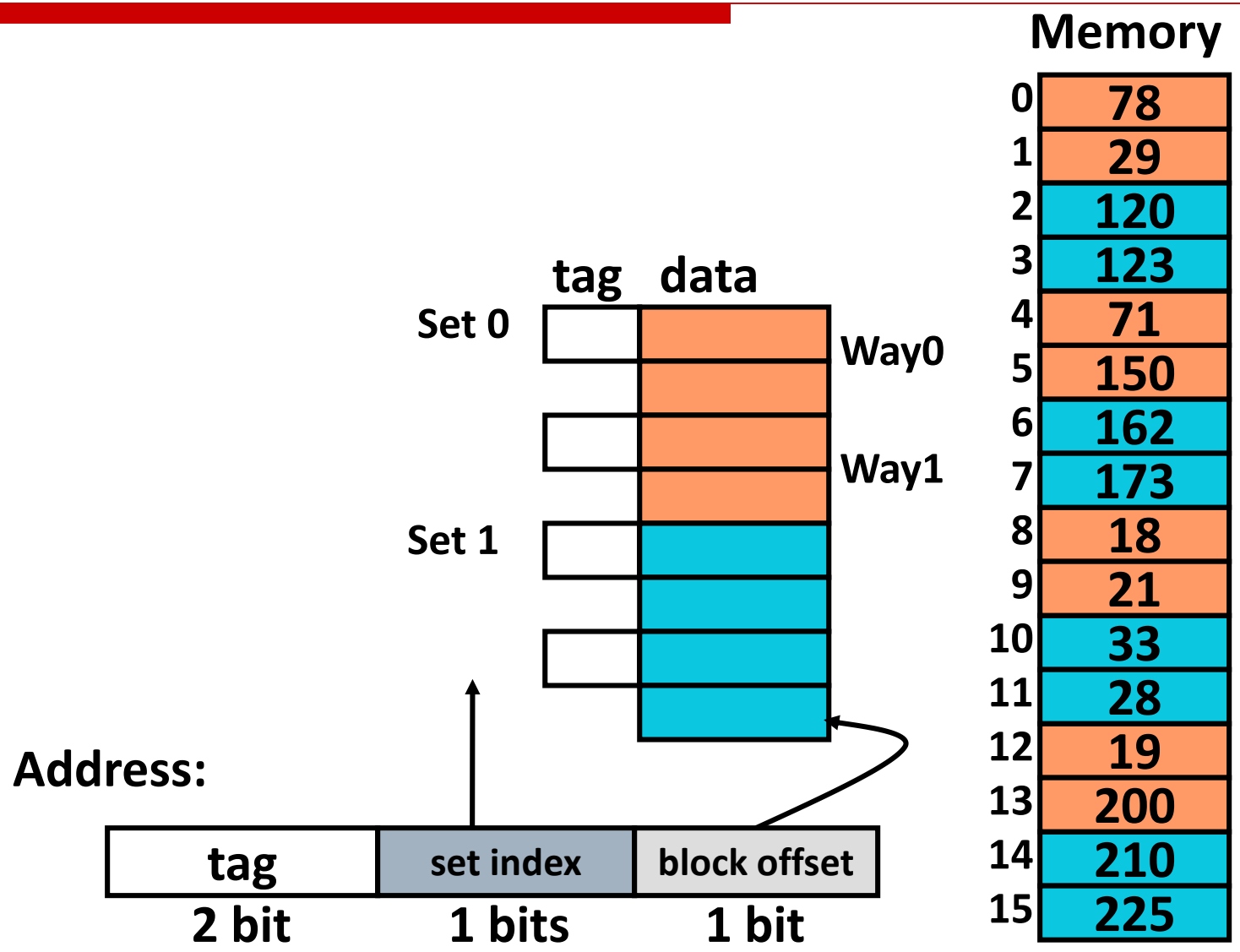
Direct-mapped cache: Placement & Access



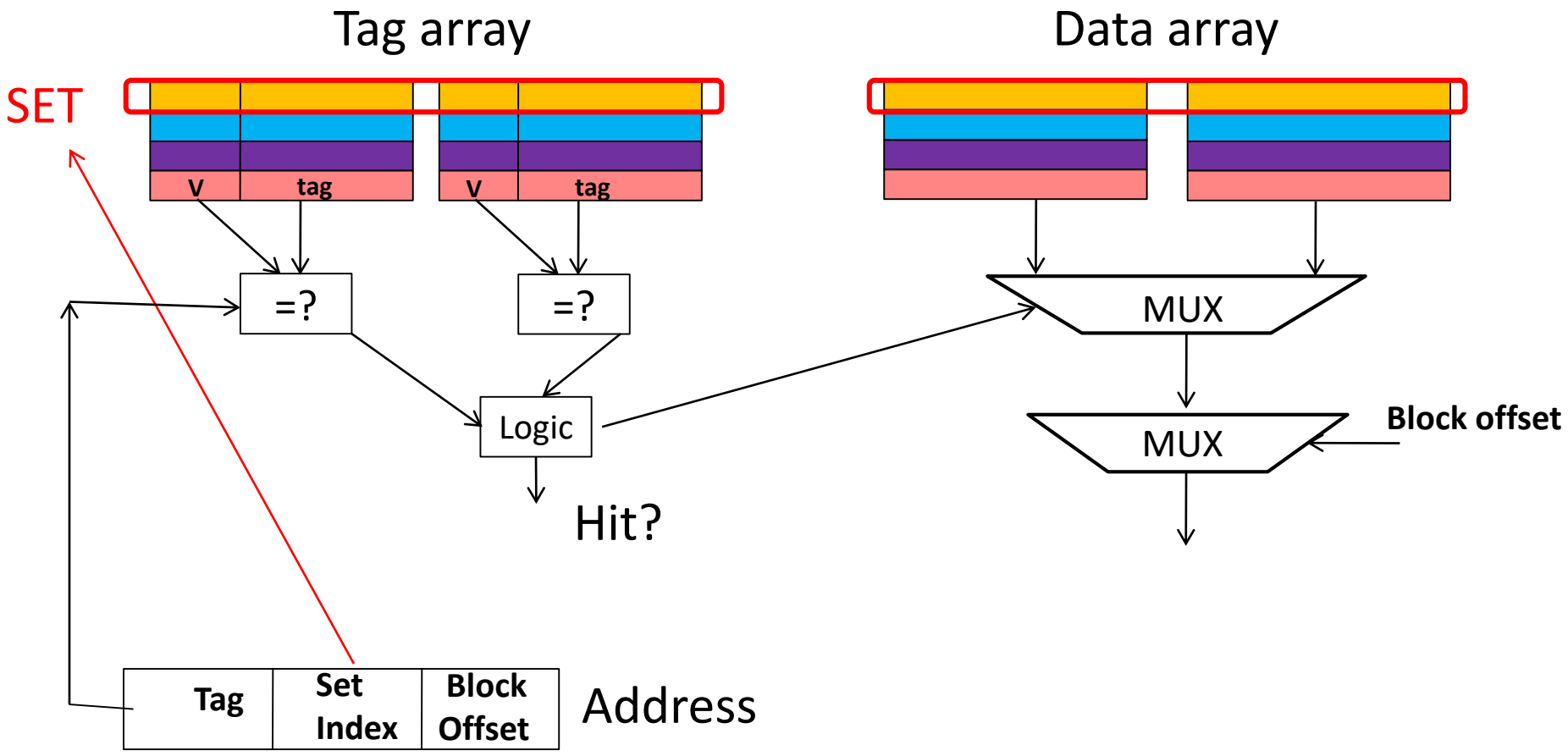
Get the advantage of both…

- Set associative caches:
  - Partition memory into regions
    - like direct mapped but fewer partitions
  - Associate a region to a set of cache lines
    - Check tags for all lines in a set to determine a HIT
- Treat each line in a set like a small fully associative cache
  - LRU (or LRU-like) policy generally used

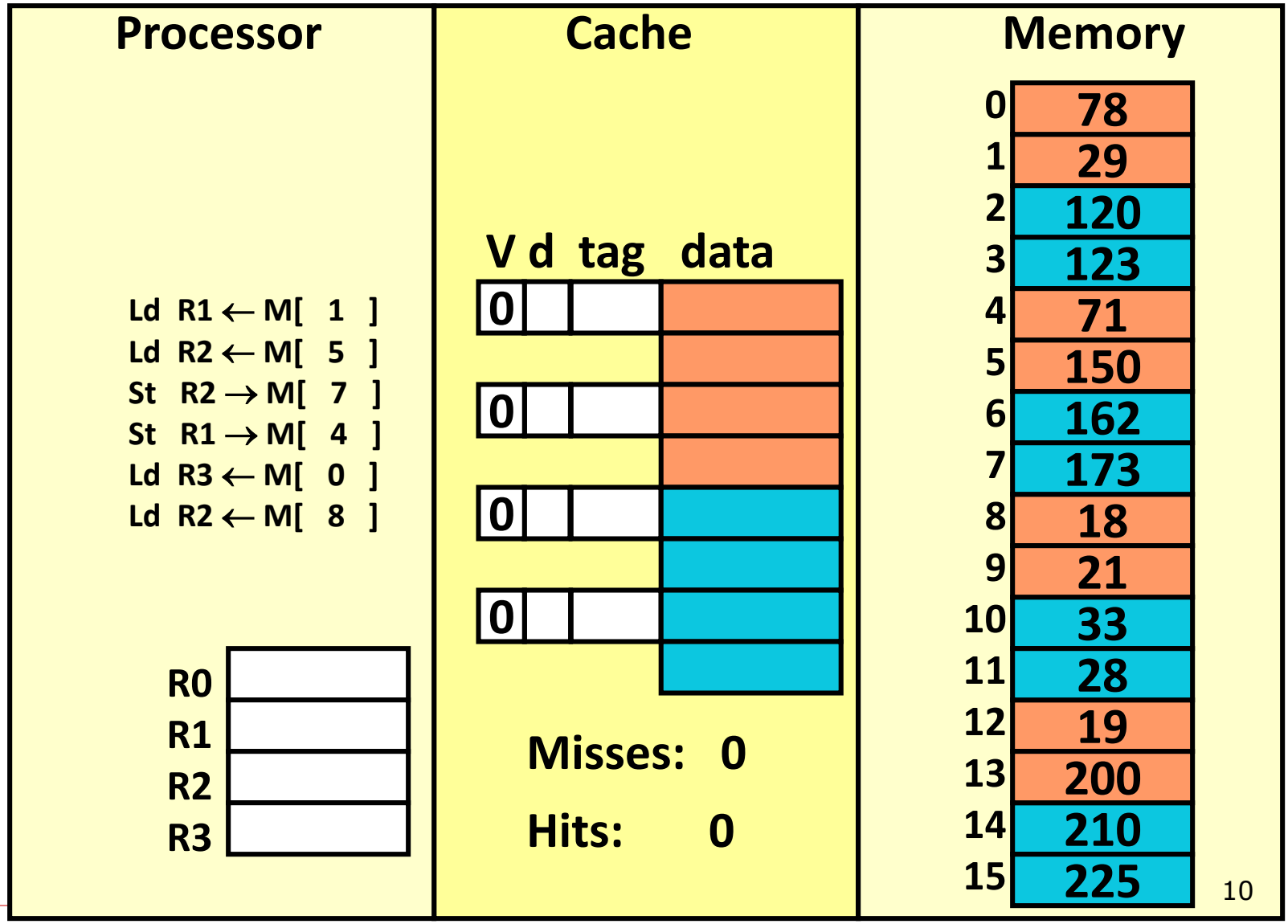
Set-associative cache



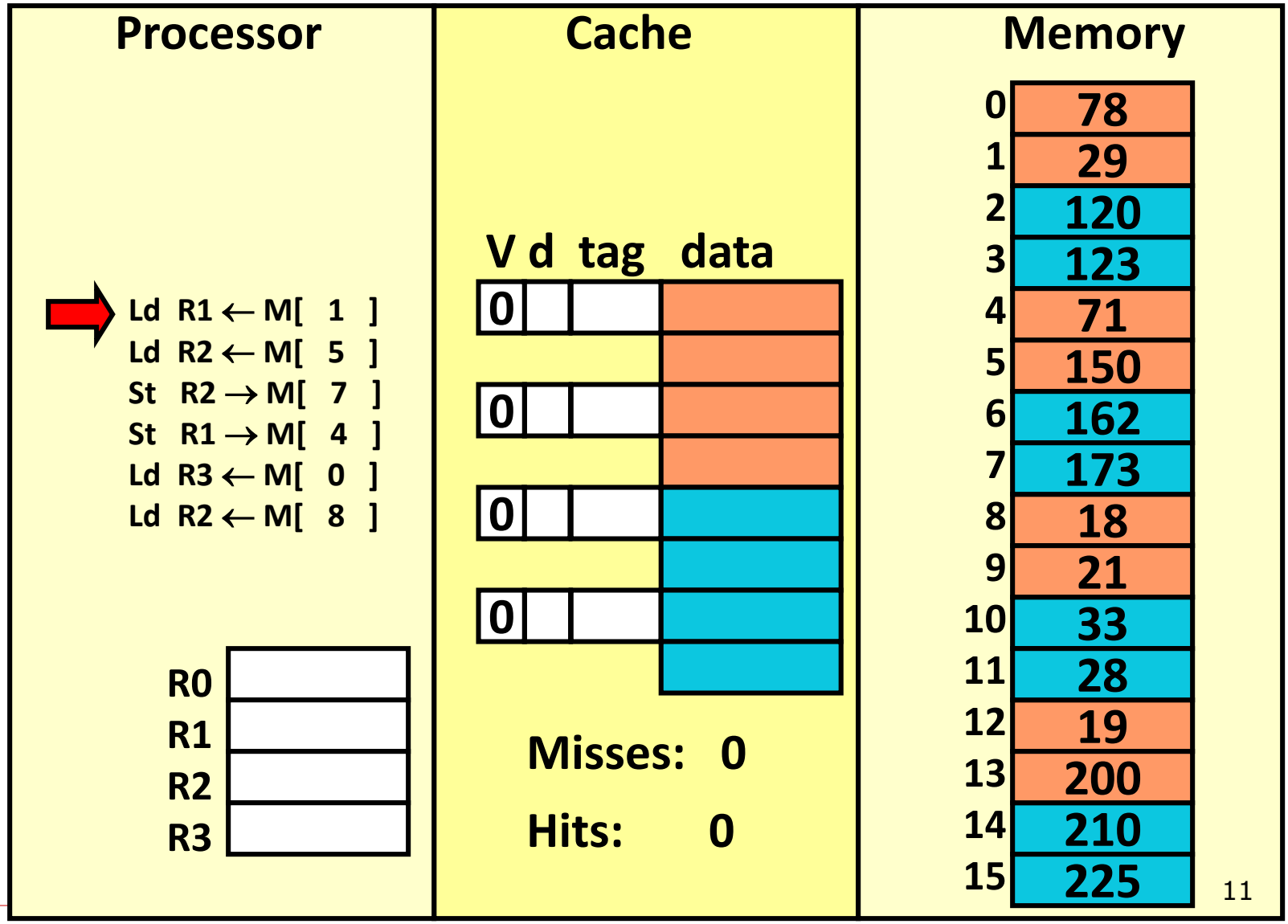
Set-associative cache: Placement & Access



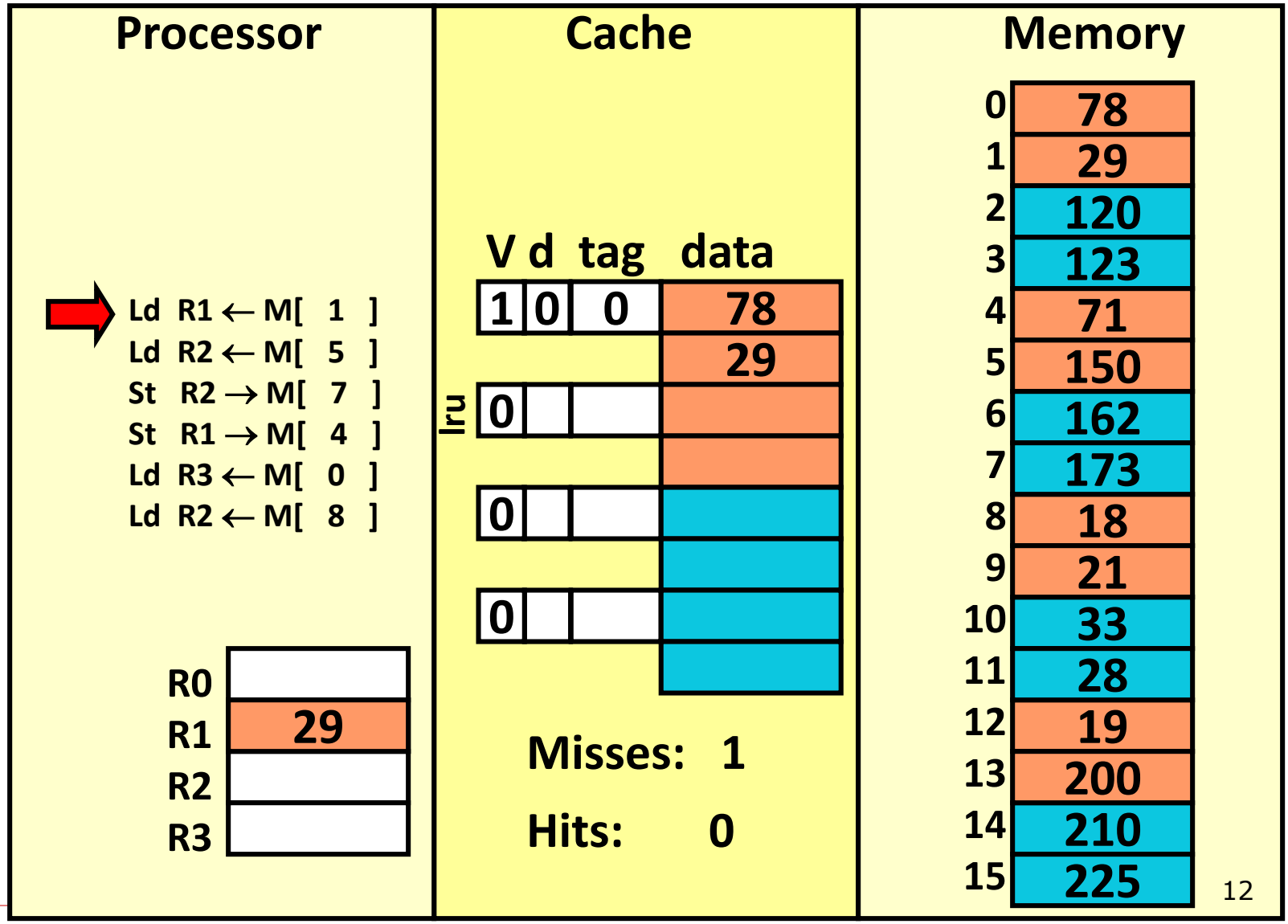
Set-associative cache example (Write-back, write allocate)



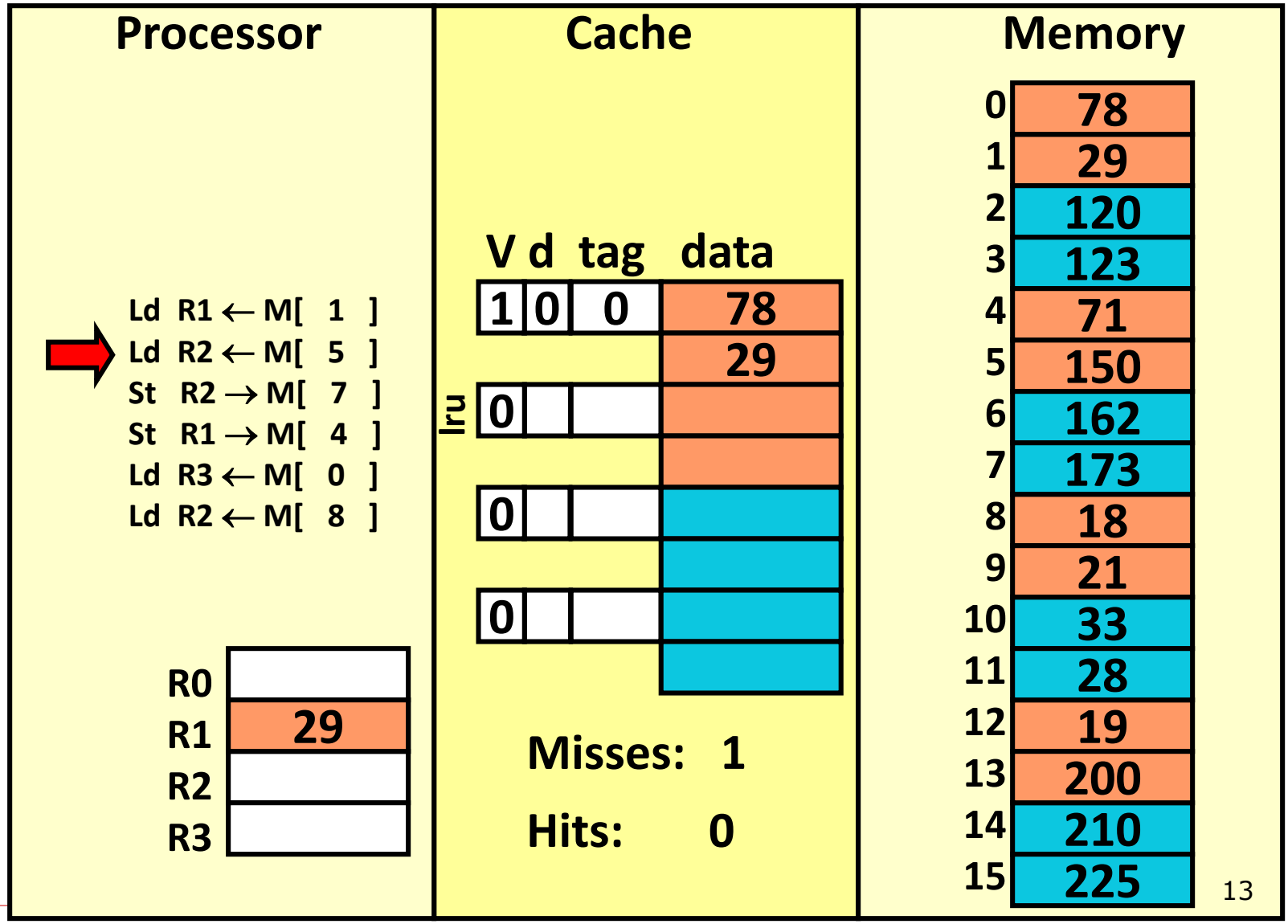
Set-associative cache (REF 1)



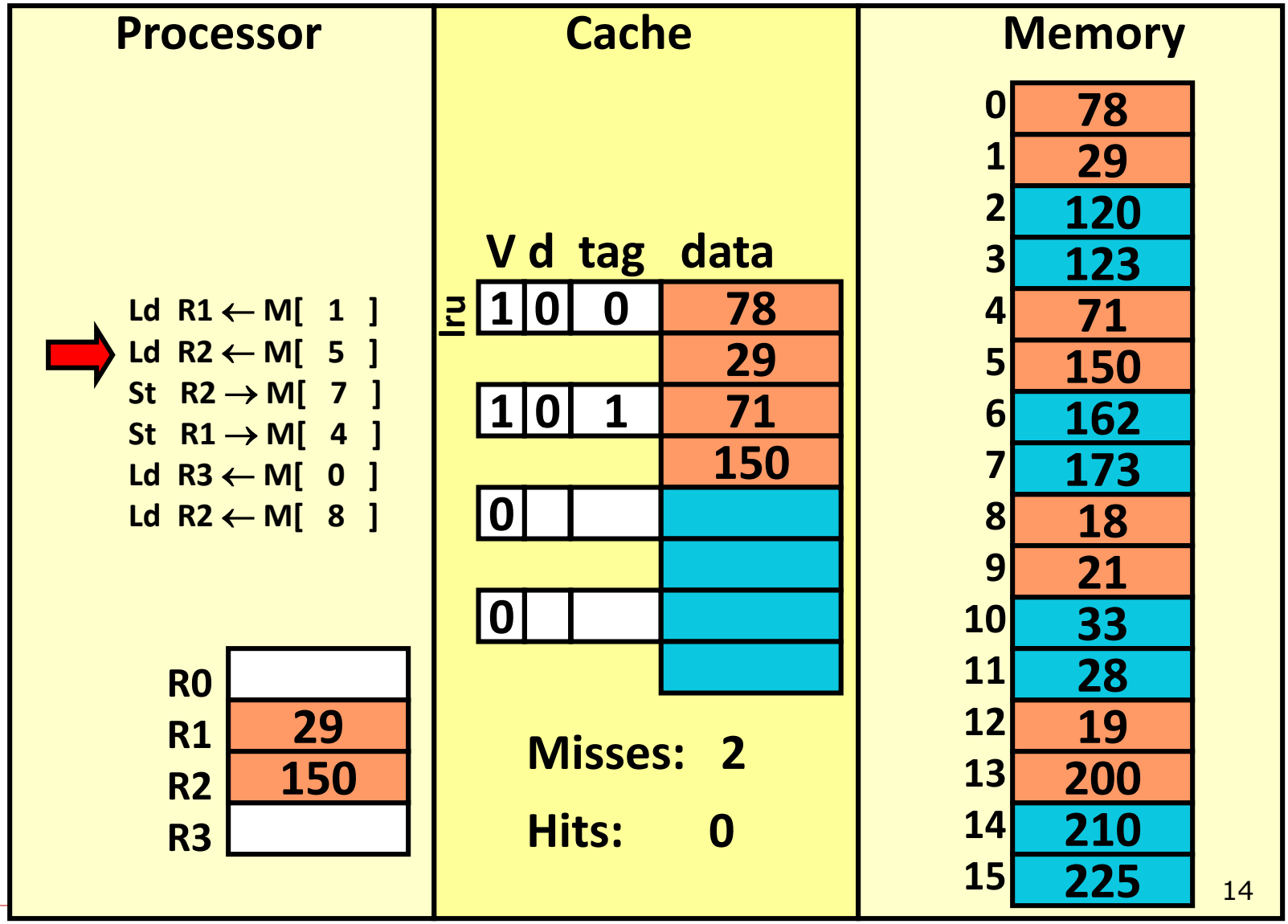
Set-associative cache (REF 1)



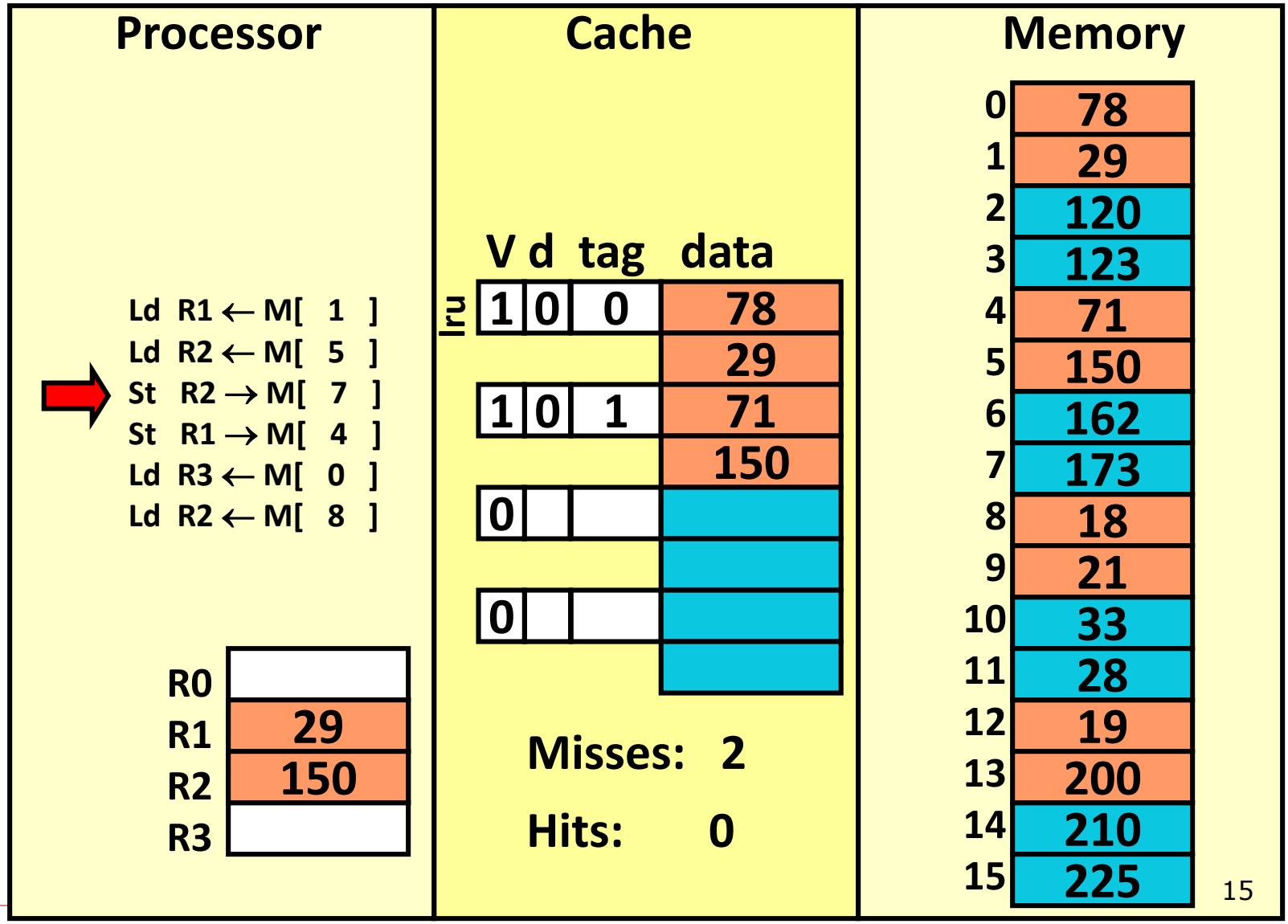
Set-associative cache (REF 2)



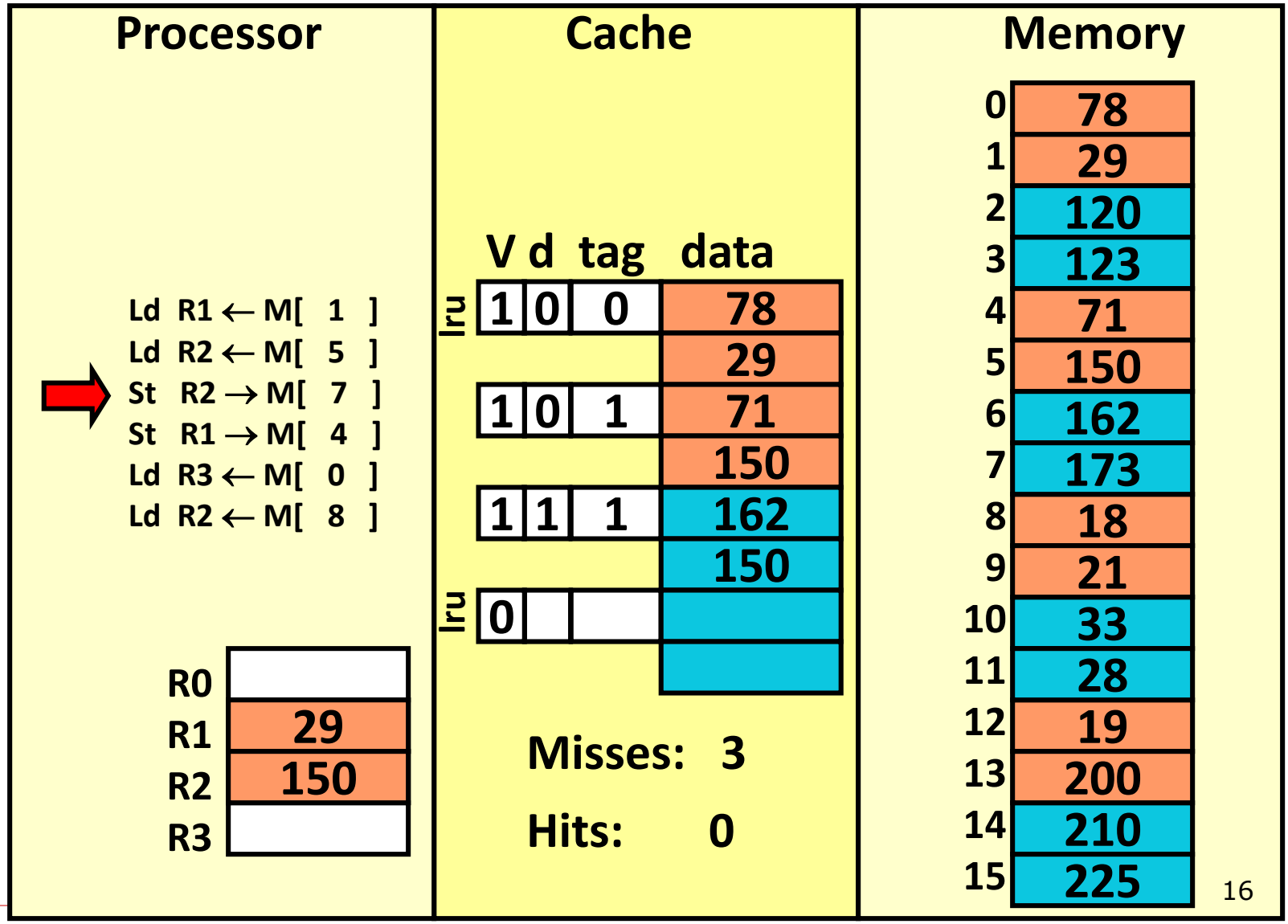
Set-associative cache (REF 2)



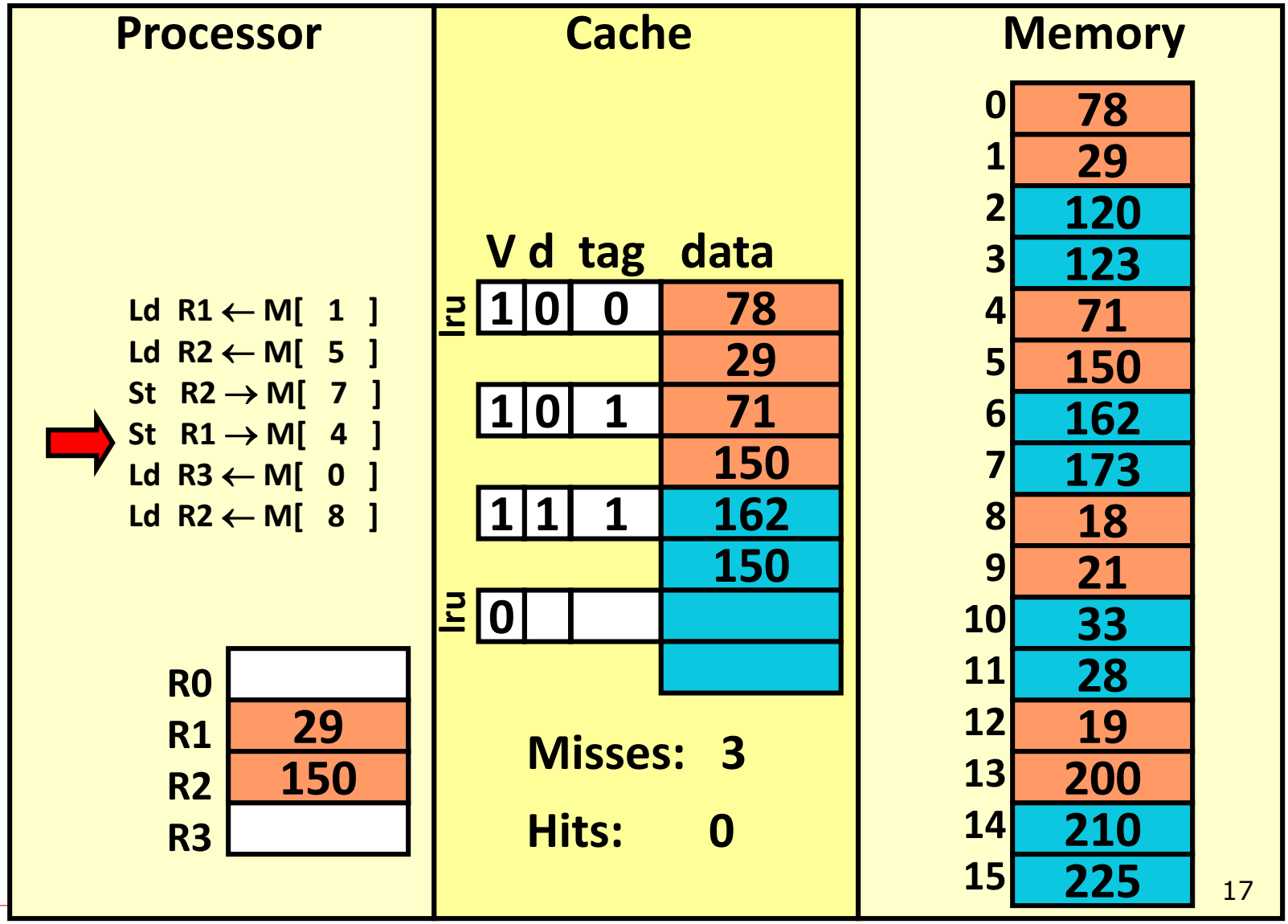
Set-associative cache (REF 3)



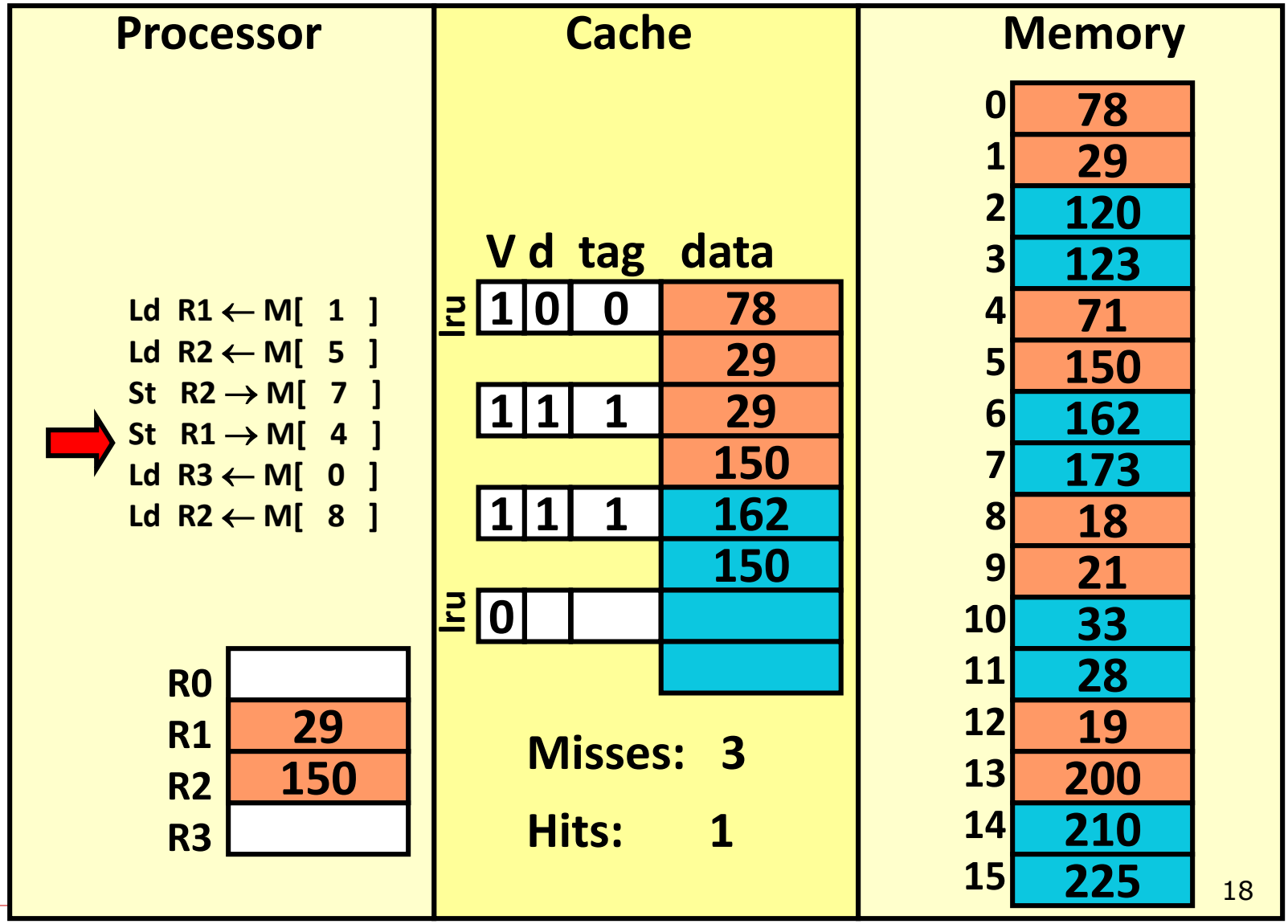
Set-associative cache (REF 3)



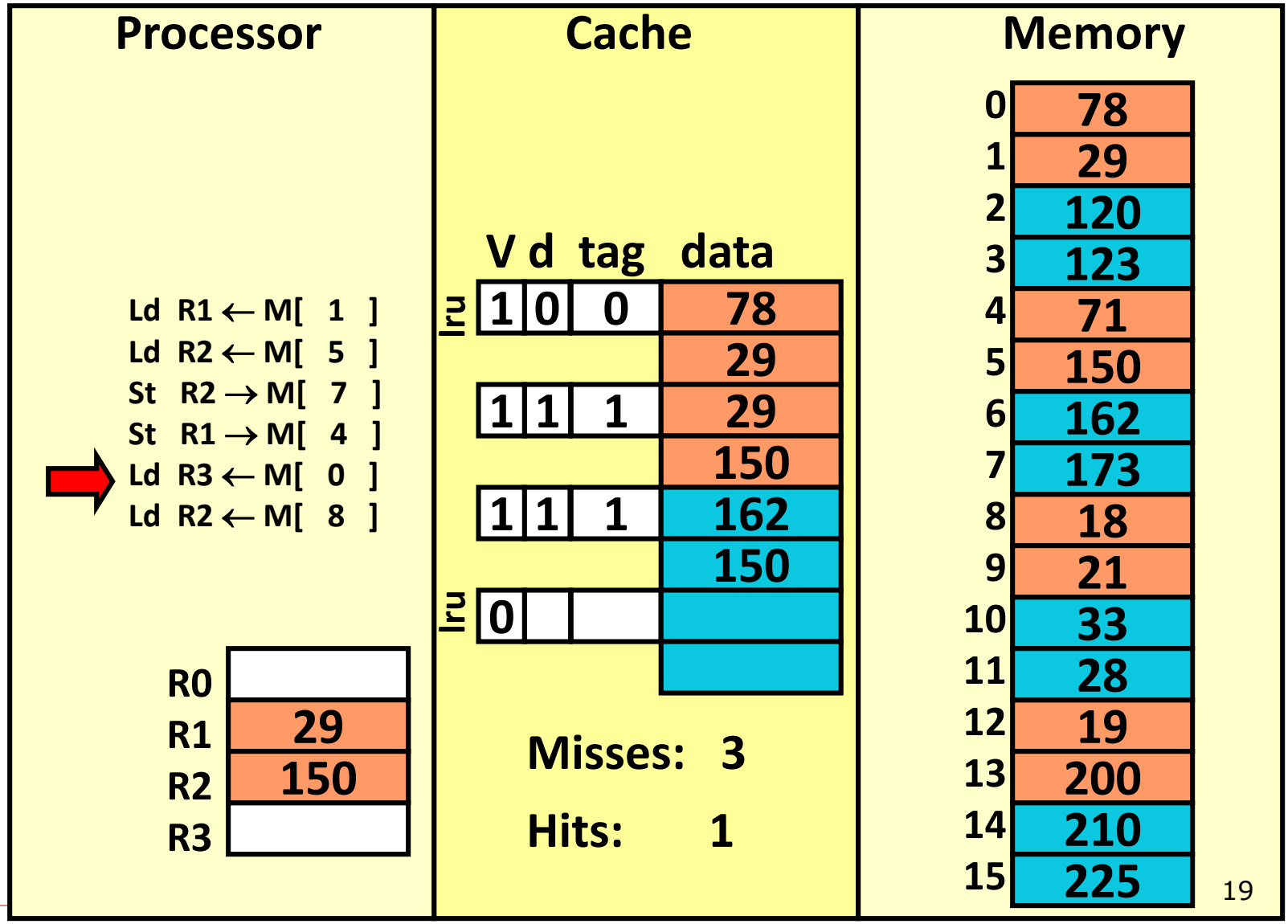
Set-associative cache (REF 4)



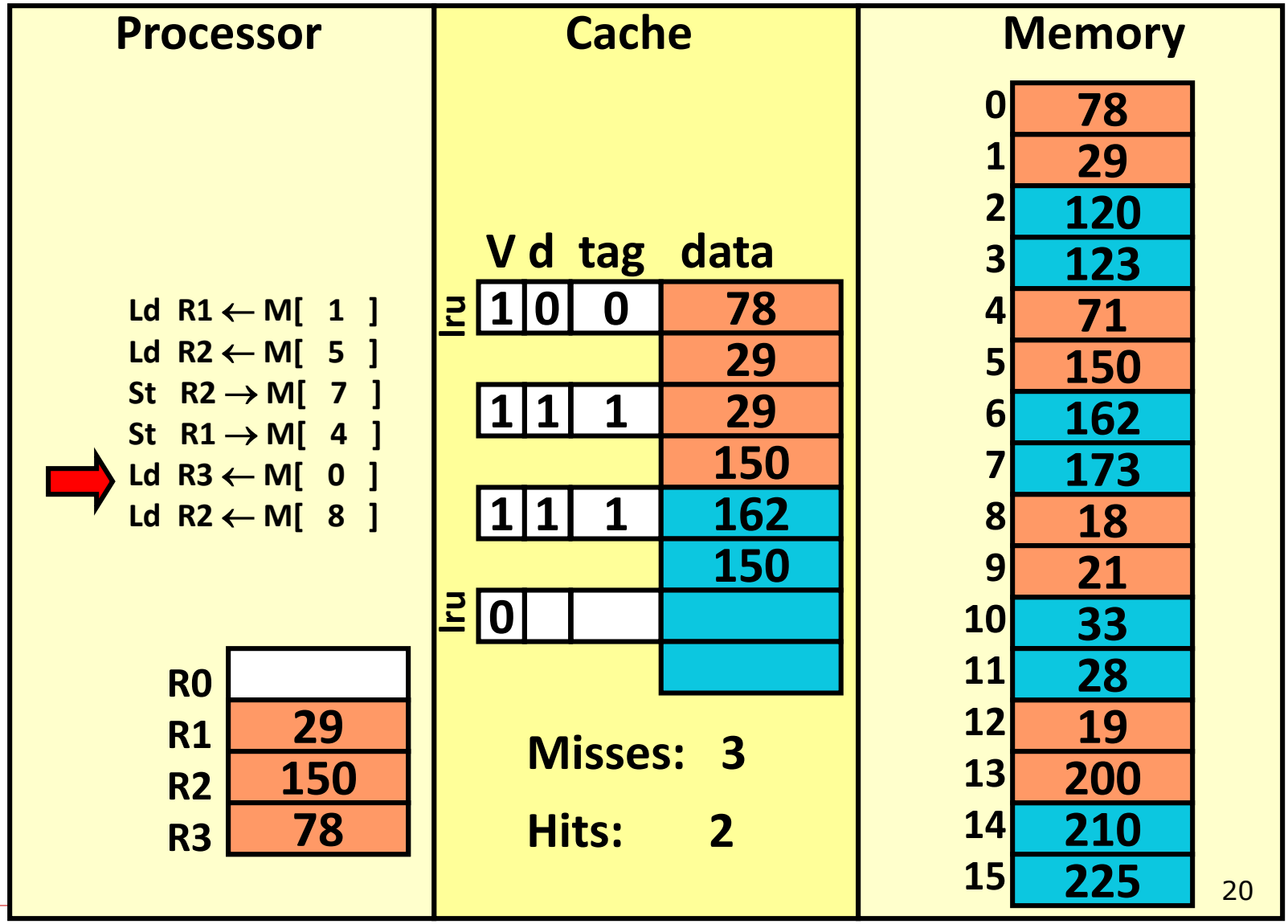
Set-associative cache (REF 4)



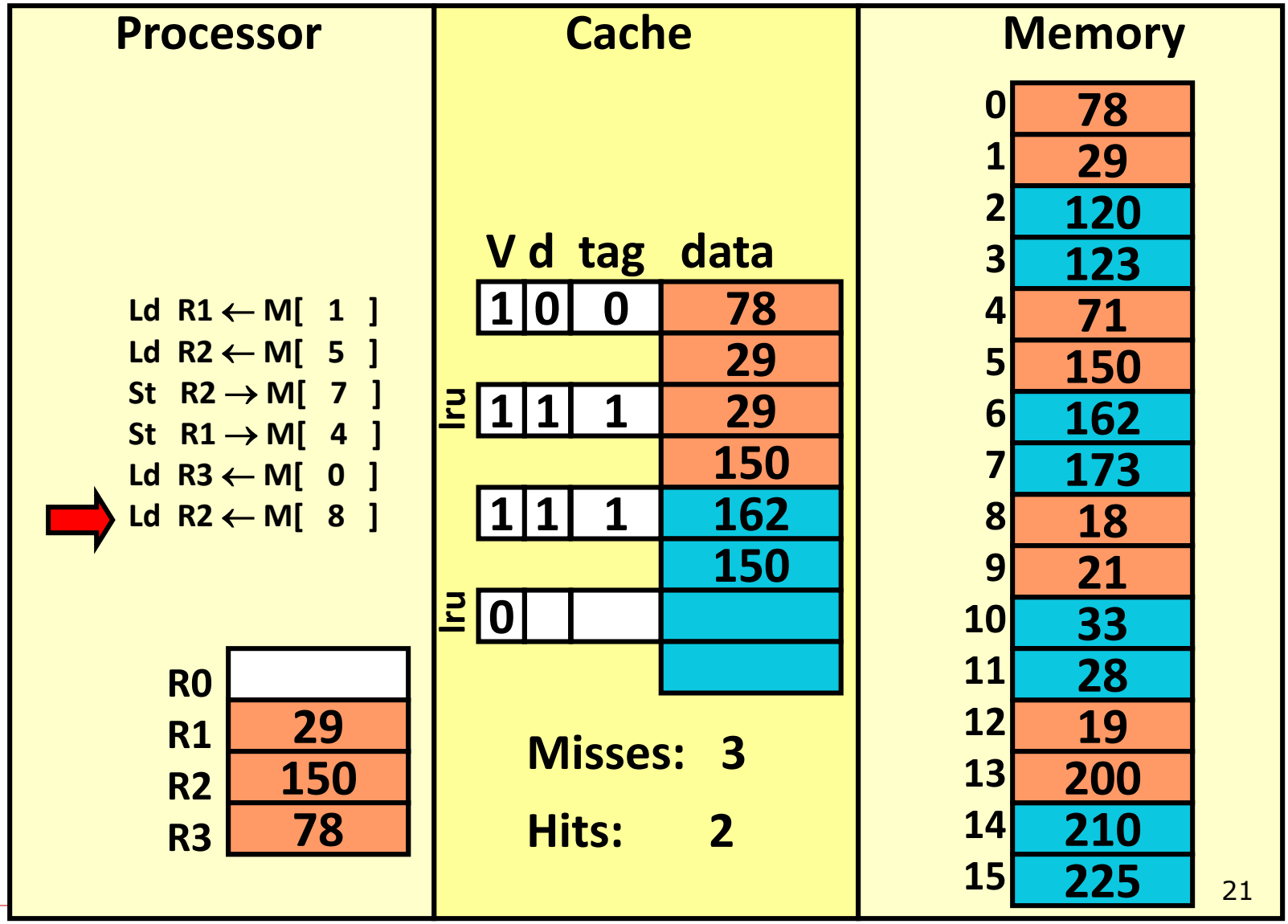
Set-associative cache (REF 5)



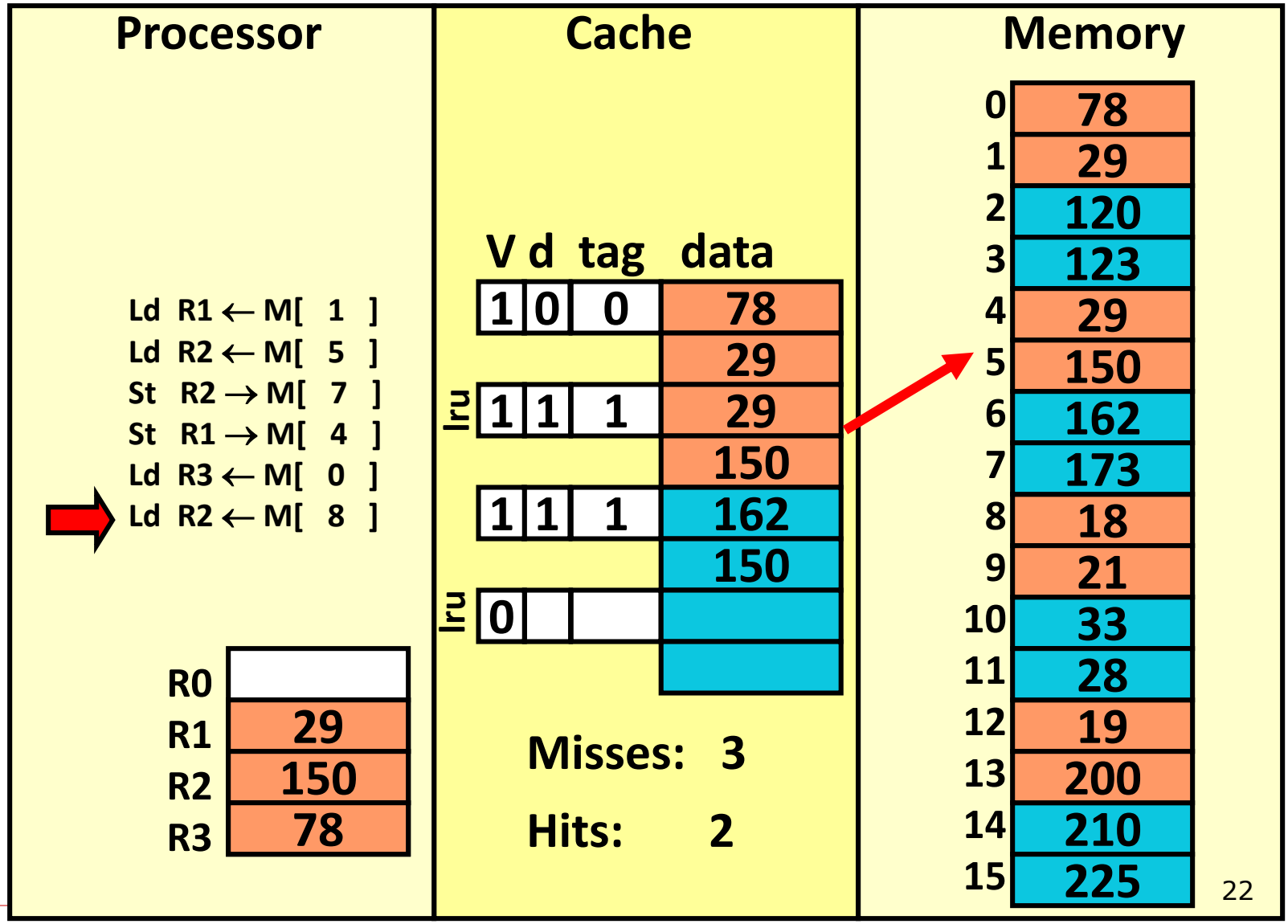
Set-associative cache (REF 5)



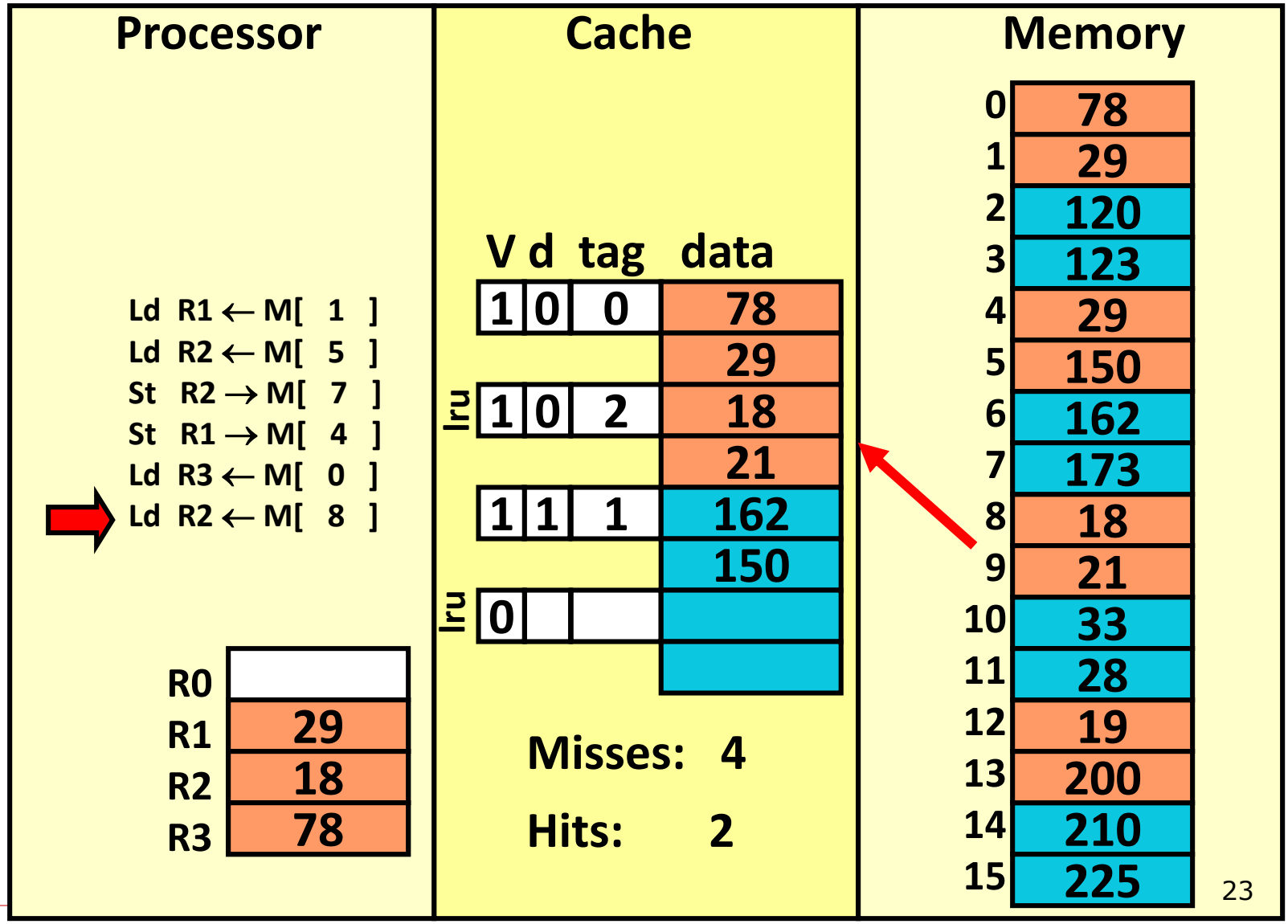
Set-associative cache (REF 6)



Set-associative cache (REF 6)



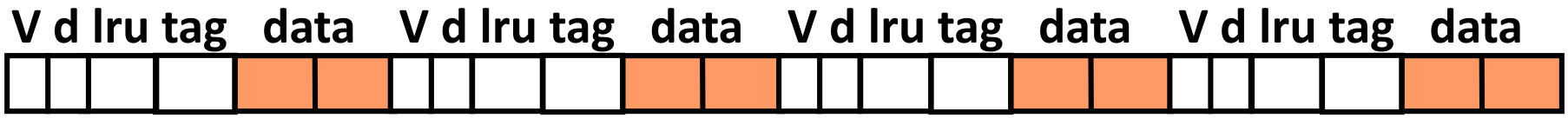
Set-associative cache (REF 6)



Cache Organization Comparison

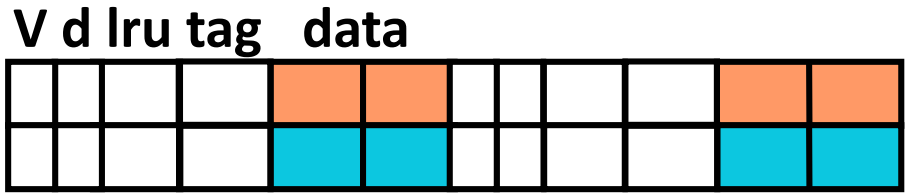
Block size = 2 bytes, total cache size = 8 bytes for all caches

1. Fully associative (4-way associative)



2. Direct mapped

3. 2-way associative



Class Problem

For a 32-bit address and 16KB cache with 64-byte blocks, show the breakdown of the address for the following cache configuration:

A) fully associative cache

B) 4-way set associative cache

C) Direct-mapped cache

Class Problem (Solution)

For a 32-bit address and 16KB cache with 64-byte blocks, show the breakdown of the address for the following cache configuration:

A) fully associative cache

Block Offset = 6 bits
Tag = 32 - 6 = 26 bits

B) 4-way set associative cache

Block Offset = 6 bits
#sets = #lines / ways = 64
Set Index = 6 bits
Tag = 32 - 6 - 6 = 20 bits

C) Direct-mapped cache

Block Offset = 6 bits
#lines = 256 Line Index = 8 bits
Tag = 32 – 6 – 8 = 18 bits

Questions to ask

- Can block size be not power of 2?
- Can number of sets be not power of 2?
- Can number of ways be not power of 2?
  - Can we have 3-way set associative cache?

Reasons for cache misses a.k.a.

The 3C's of Cache Misses

- First reference to an address
  - **Compulsory** miss
    - Also sometimes called a **"cold start"** miss
    - First reference to any block will always miss
- Cache is too small to hold all the data
  - **Capacity** miss
    - Would have had a hit with a large enough cache
- Replaced it from a busy set
  - **Conflict** miss
    - Would have had a hit with a fully associative cache

Classifying Cache Misses

- Can you classify all cache misses?
  - **Compulsory miss?**
  - **Capacity miss?**
  - **Conflict miss?**
- Yes! (with a cache simulator)
  - Simulate with a cache of unlimited size (cache size = memory size) – Any misses must be **compulsory misses**
  - Simulate again with a fully associative cache of the intended size - Any new misses must be **capacity misses**
  - Simulate a third time, with the actual intended cache - Any new misses must be **conflict misses**

Fixing cache misses

- **Compulsory** misses
  - First reference to a address
  - No way to completely avoid these
  - Reduce by **increasing block size**
  - This reduces the total number of blocks
- **Capacity** misses
  - Would have a hit with a large enough cache
  - Reduce by **building a bigger cache**
- **Conflict** misses
  - Would have had a hit with a fully associative cache
  - Cache does not have enough associativity
  - Reduce by **increasing associativity**

3 C's Sample Problem

Consider a cache with the following configuraion: write-allocate, total size is 64 bytes, block size is 16 bytes, and 2-way associative. The memory address size is 16 bits and byte-addressable. The replacement policy is LRU. The cache is empty at the start.

For the following memory accesses, indicate whether the reference is a hit or miss, and the type of a miss (compulsory, conflict, capacity)

3 C's Practice Problem – Infinite cache

Address

0x00

0x14

0x27

0x08

0x38

0x4A

0x18

0x27

0x0F

0x40

3 C's Practice Problem – Fully associative

Address

0x00

0x14

0x27

0x08

0x38

0x4A

0x18

0x27

0x0F

0x40

3 C's Practice Problem – Set Associative

Address

0x00

0x14

0x27

0x08

0x38

0x4A

0x18

0x27

0x0F

0x40

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00				
0x14				
0x27				
0x08				
0x38				
0x4A				
0x18				
0x27				
0x0F				
0x40				

block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00	M	M	M	
0x14	M	M	M	
0x27	M	M	M	
0x08	H	H	H	
0x38				
0x4A				
0x18				
0x27				
0x0F				
0x40				

block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00	M	M	M	Compulsory
0x14	M	M	M	Compulsory
0x27	M	M	M	Compulsory
0x08	H	H	H	---
0x38				
0x4A				
0x18				
0x27				
0x0F				
0x40				

block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00	M	M	M	Compulsory
0x14	M	M	M	Compulsory
0x27	M	M	M	Compulsory
0x08	H	H	H	---
0x38	M	M	M	
0x4A	M	M	M	
0x18	H	M	H	
0x27	H	M	M	
0x0F				
0x40				

block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00	M	M	M	Compulsory
0x14	M	M	M	Compulsory
0x27	M	M	M	Compulsory
0x08	H	H	H	---
0x38	M	M	M	Compulsory
0x4A	M	M	M	Compulsory
0x18	H	M	H	---
0x27	H	M	M	Capacity
0x0F				
0x40				

block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00	M	M	M	Compulsory
0x14	M	M	M	Compulsory
0x27	M	M	M	Compulsory
0x08	H	H	H	---
0x38	M	M	M	Compulsory
0x4A	M	M	M	Compulsory
0x18	H	M	H	---
0x27	H	M	M	Capacity
0x0F	H	M	M	
0x40	H	H	M	

block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

3 C's Practice Problem – 3 C's

Address	Infinite	FA	SA	3Cs
0x00	M	M	M	Compulsory
0x14	M	M	M	Compulsory
0x27	M	M	M	Compulsory
0x08	H	H	H	---
0x38	M	M	M	Compulsory
0x4A	M	M	M	Compulsory
0x18	H	M	H	---
0x27	H	M	M	Capacity
0x0F	H	M	M	Capacity
0x40	H	H	M	Conflict

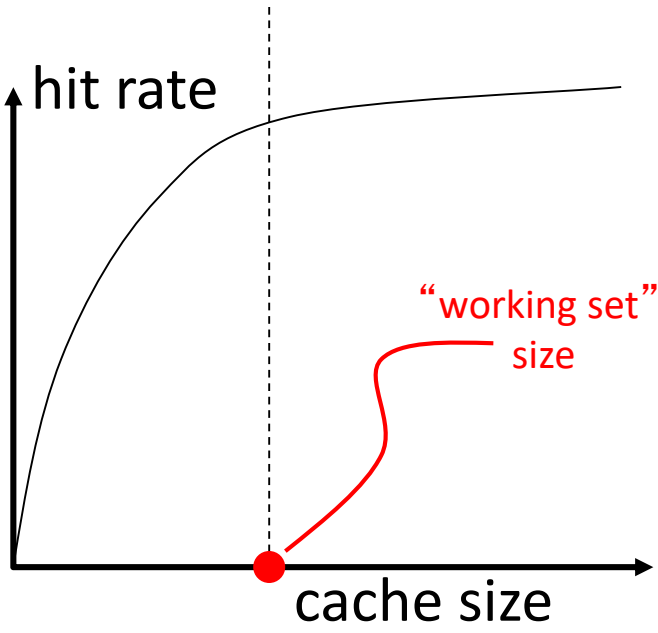
block size is 16 bytes, FA and SA have 4 blocks, SA is 2-way.

Cache Parameters vs. Miss Rate

- Cache Size
- Block Size
- Associativity
- Replacement policy

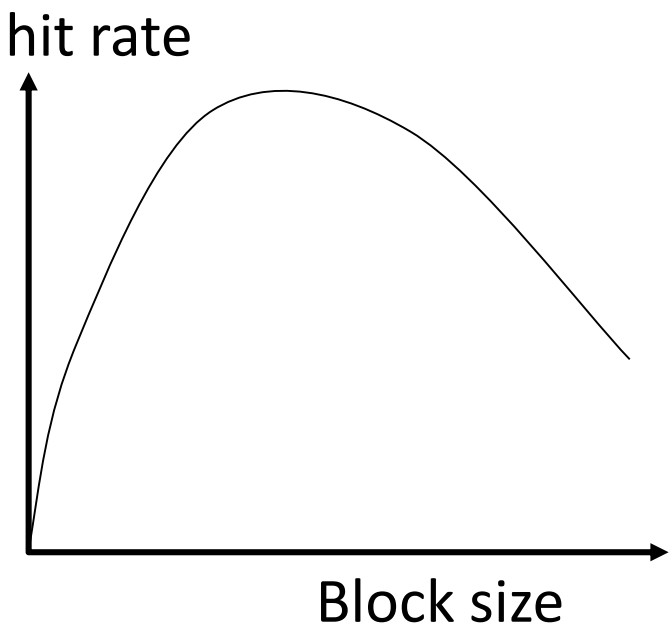
Cache Size

- Cache size in the total data (not including tag) capacity
  - bigger can exploit temporal locality better
  - not ALWAYS better
- Too large a cache adversely affects hit & miss latency
  - smaller is faster => bigger is slower
  - access time may degrade critical path
- Too small a cache
  - doesn't exploit temporal locality well
  - useful data replaced often
- Working set: the whole set of data executing application references
  - **Within a time interval**



Block size (also called Line size)

- Block size is the data that is associated with an address tag
  - Sub-blocking: A block divided into multiple pieces (each with V bit)
    - Can improve "write" performance
- Too small blocks
  - don't exploit spatial locality well
  - have larger tag overhead
- Too large blocks
  - too few total # of blocks
    - likely-useless data transferred
    - Extra bandwidth/energy consumed



Associativity

- How many blocks can map to the same index (or set)?
- Larger associativity
  - lower miss rate, less variation among programs
  - diminishing returns
- Smaller associativity
  - lower cost
  - faster hit time
    - Especially important for L1 caches
- Power of 2 associativity?