

---

**IC220 Slide Set #16:  
More More Memory (Hierarchy)  
(Chapter 5)**

1

---

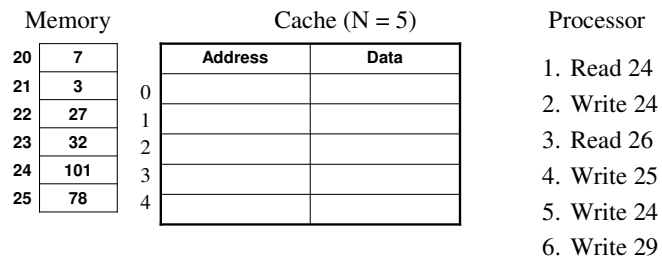
**Improving our Simple Cache**

1. How to handle a write?
2. Efficient Bit Manipulation
3. How to eliminate even more conflicts?
4. Can hierarchy help?

2

---

**Issue #1: What to do on a write?**



3

---

**Comparing Write Strategies**

- Write-through:
- Write-back
- How to improve write-through?

4

## Issue #2: Efficient Bit Manipulation

Given cache with 8 bytes per block,  $N=16$ , what is index of address "153"?

OLD: 
$$\text{Index} = \left\lfloor \frac{\text{ByteAddress}}{\text{BytesPerBlock}} \right\rfloor \bmod N$$

NEW: (assuming dealing with powers-of-2)

a. Express in binary. ( $153_{10} = 99_{16}$ )

b. Grab the right bits!

ByteOffset =

Index =

5

## Example #1: Bit Manipulation

1. Suppose cache has:

- 8 byte blocks
- 256 blocks

Show how to break the following address into the tag, index, & byte offset.

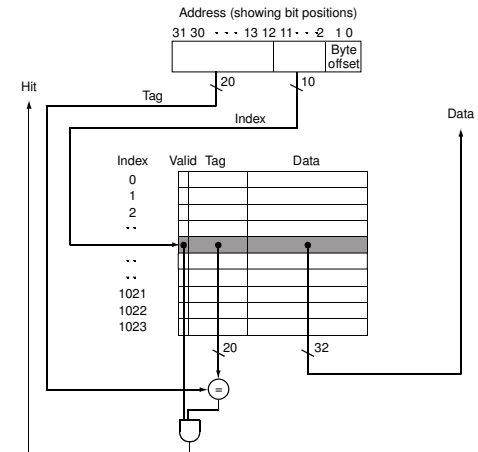
0000 1000 0101 1100 0001 0001 0111 1001

2. Same cache, but now 4-way associative. How does this change things?

0000 1000 0101 1100 0001 0001 0111 1001

7

## Real Cache with Efficient Bit Manipulation



6

## Example #2: Bit Manipulation

Suppose a direct-mapped cache divides addresses as follows:



What is the block size?

The number of blocks?

Total size of the cache?  
(usually refers to size of data only)

8

## Key Rules

EX 7-21...

- How do the # sets and # blocks relate?
- Calculate # index bits from # sets
- One hex 'digit' = 4 bits
  - 0x1234 = 0001 0010 0011 0100

9

## Issue #3: How to eliminate even more conflicts?

- Fully associative cache – cache block can go \_\_\_\_\_ in cache
- Pros
- Cons
- Can view all caches as n-way associative:
  - Direct-mapped, n =
  - 4-way associative, n =
  - Fully associative, n =

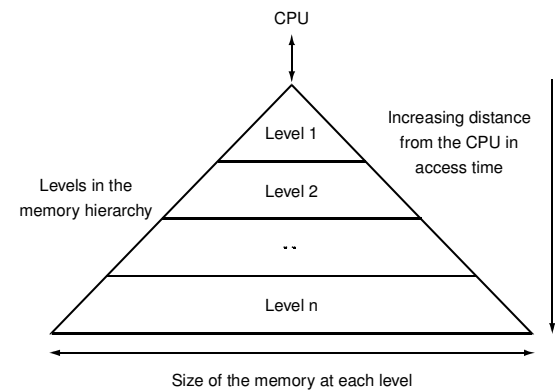
10

## Issue #4: More hierarchy – L2 cache?

- Add a second level cache:
  - often primary cache is on the same chip as the processor
  - use SRAMs to add another cache above primary memory (DRAM)
  - miss penalty goes down if data is in 2nd level cache
- Performance smarts:
  - try and optimize the \_\_\_\_\_ on the 1st level cache
  - try and optimize the \_\_\_\_\_ on the 2nd level cache

11

## Memory Hierarchy



12

