CACHE ARCHITECTURE

Mahdi Nazm Bojnordi

Assistant Professor

School of Computing

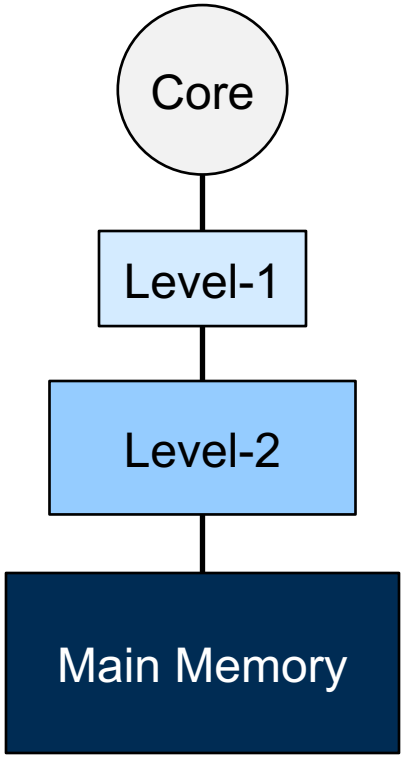
University of Utah

Overview

- Announcement
  - No homework assignments till Thu ☺
- Cache Architecture
  - Addressing and lookup
- Cache Optimizations
  - Techniques to improve miss rate
  - Replacement policies
  - Write policies

Recall: Cache Performance

- Bridging the processor-memory performance gap



Main memory access time: 300 cycles

Two level cache

- L1: 2 cycles hit time; 60% hit rate
- L2: 20 cycles hit time; 70% hit rate

What is the average mem access time?

$$AMAT = t_{h1} + r_{m1}\, t_{p1}$$

$$t_{p1} = t_{h2} + r_{m2}\, t_{p2}$$

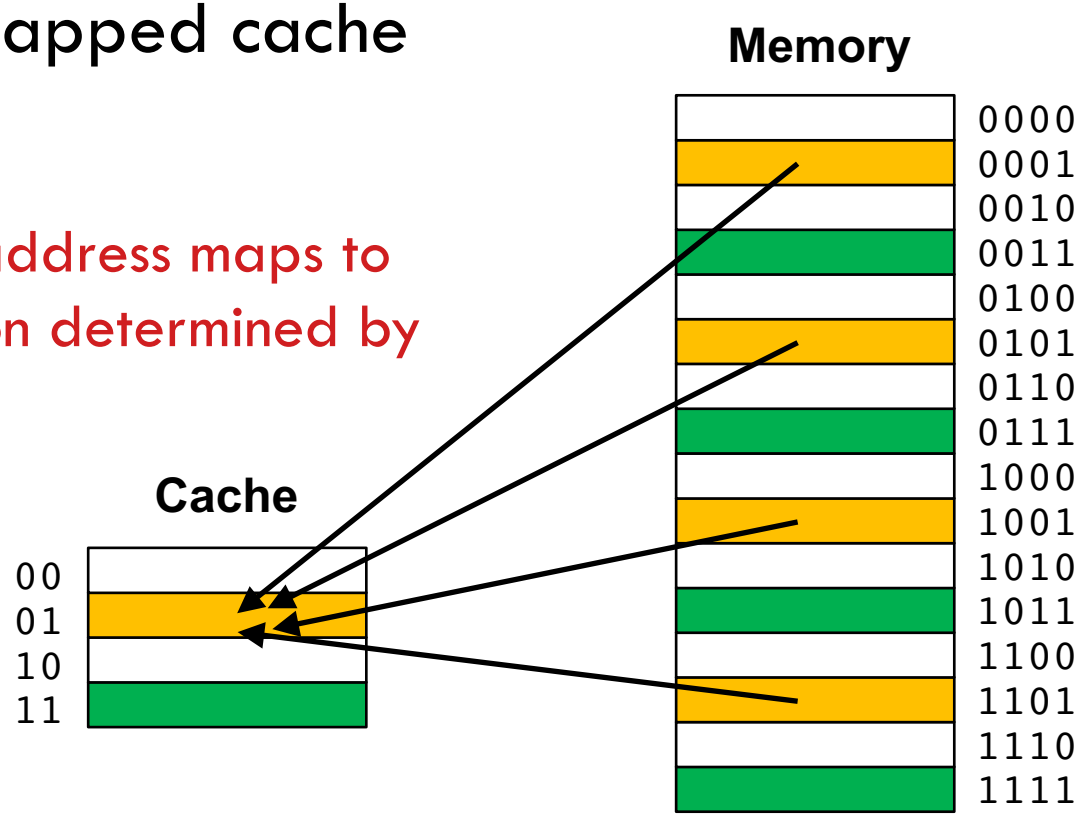
$$AMAT = 46$$

Cache Addressing

- Instead of specifying cache address we specify main memory address
- Simplest: direct-mapped cache

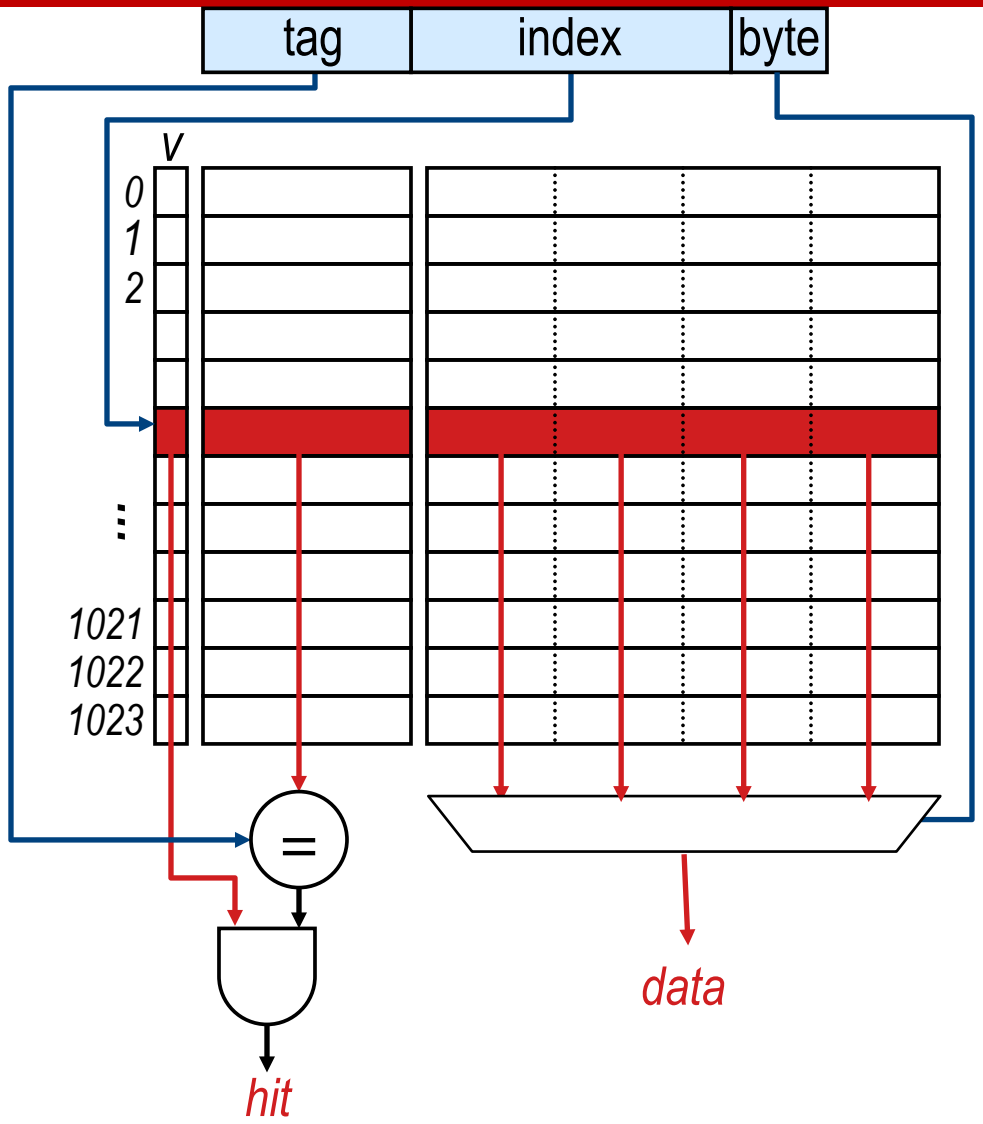
Note: each memory address maps to a single cache location determined by modulo hashing

How to exactly specify which blocks are in the cache?



Direct-Mapped Lookup

- Byte offset: to select the requested byte
- Tag: to maintain the address
- Valid flag (v): whether content is meaningful
- Data and tag are always accessed



Example Problem

- Find the size of tag, index, and offset bits for an 8MB, direct-mapped L3 cache with 64B cache blocks. Assume that the processor can address up to 4GB of main memory.

Example Problem

- Find the size of tag, index, and offset bits for an 8MB, direct-mapped L3 cache with 64B cache blocks. Assume that the processor can address up to 4GB of main memory.
- $4GB = 2^{32}$ B → address bits = 32
- $64B = 2^{6}$ B → byte offset bits = 6
- $8MB/64B = 2^{17}$ → index bits = 17
- tag bits = 32 – 6 – 17 = 9

Cache Optimizations

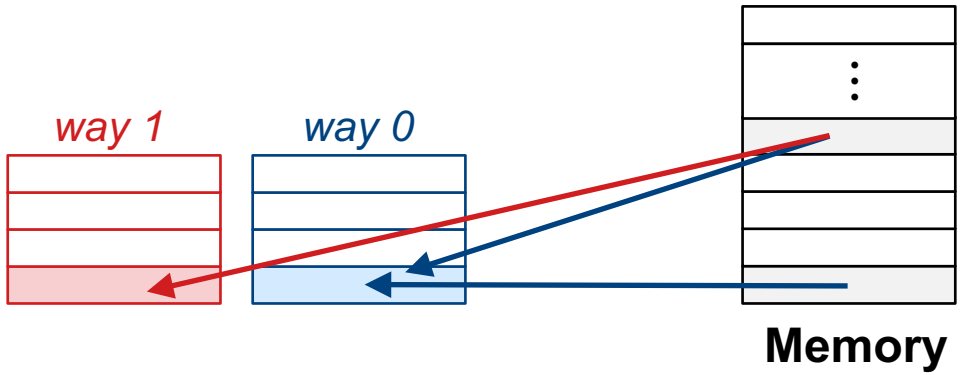
- How to improve cache performance

$$AMAT = t_h + r_m\, t_p$$

- Reduce hit time (t_h)
  - Memory technology, critical access path
- Improve hit rate ($1 - r_m$)
  - Size, associativity, placement/replacement policies
- Reduce miss penalty (t_p)
  - Multi level caches, data prefetching

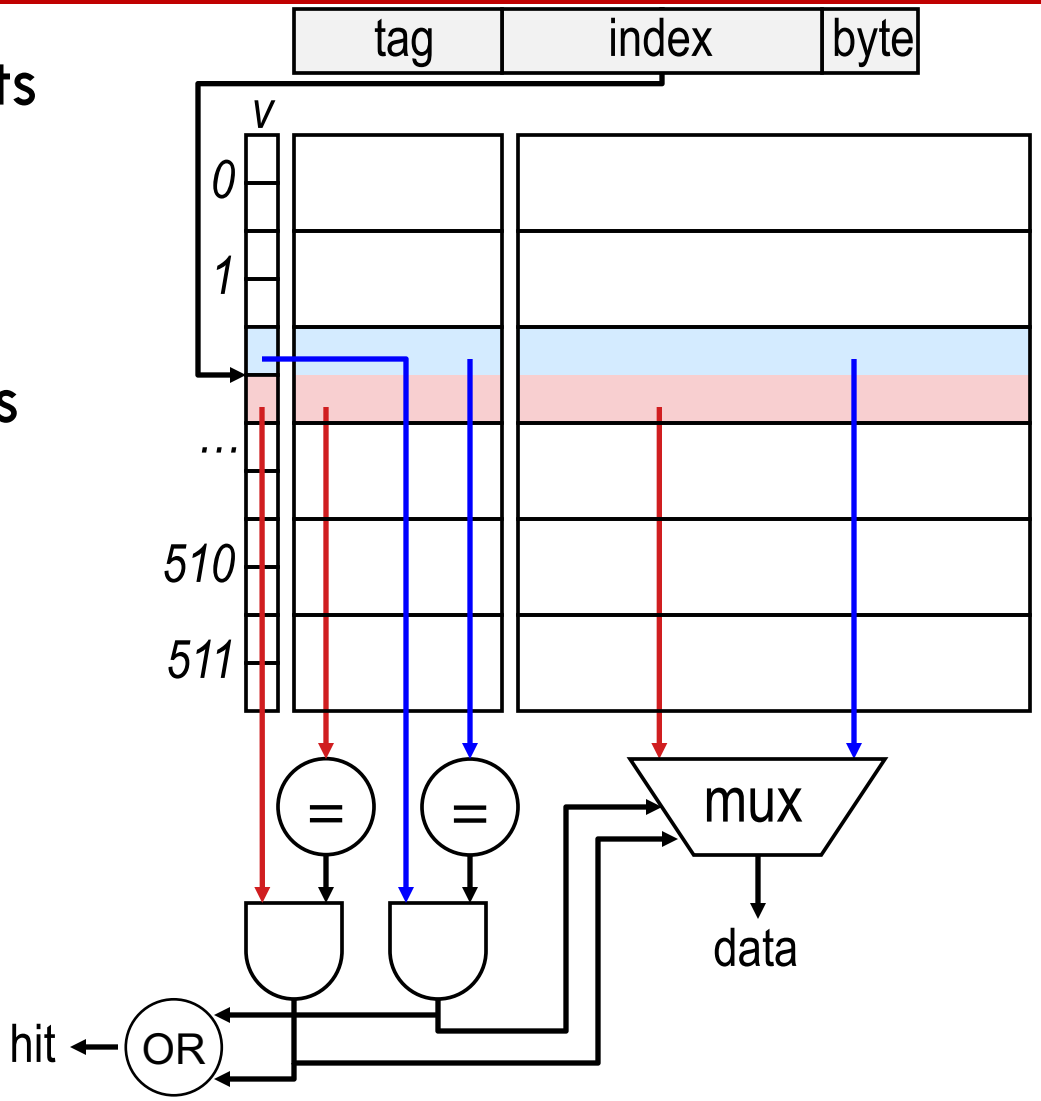
Set Associative Caches

- Improve cache hit rate by allowing a memory location to be placed in more than one cache block
  - N-way set associative cache
  - Fully associative
- For fixed capacity, higher associativity typically leads to higher hit rates
  - more places to simultaneously map cache lines
  - 8-way SA close to FA in practice



n-Way Set Associative Lookup

- Index into cache sets
- Multiple tag comparisons
- Multiple data reads
- Special cases
  - Direct mapped
    - Single block sets
  - Fully associative
    - Single set cache



Example Problem

- Find the size of tag, index, and offset bits for an 4MB, 4-way set associative cache with 32B cache blocks. Assume that the processor can address up to 4GB of main memory.

Example Problem

- Find the size of tag, index, and offset bits for an 4MB, 4-way set associative cache with 32B cache blocks. Assume that the processor can address up to 4GB of main memory.
- $4GB = 2^{32}$ B → address bits = 32
- $32B = 2^{5}$ B → byte offset bits = 5
- $4MB/(4x32B) = 2^{15}$ → index bits = 15
- tag bits = 32 – 5 – 15 = 12

Cache Miss Classifications

- Start by measuring miss rate with an ideal cache
  - 1. ideal is fully associative and infinite capacity
  - 2. then reduce capacity to size of interest
  - 3. then reduce associativity to degree of interest

1. Cold (compulsory)	2. Capacity	3. Conflict
❑ Cold start: first access to block	❑ Cache is smaller than the program data	❑ Set size is smaller than mapped mem. locations
❑ How to improve	❑ How to improve	❑ How to improve
○ Large blocks	○ Large cache	○ Large cache
○ prefetching		○ More assoc.