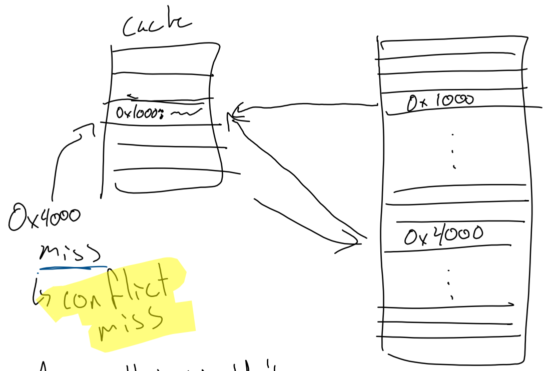
Lecture 20: More caches

Monday, February 25, 2019 10:59 AM

Outline
- Set associativity
- Three C's
- AMAT
- Multi-level hierarchies

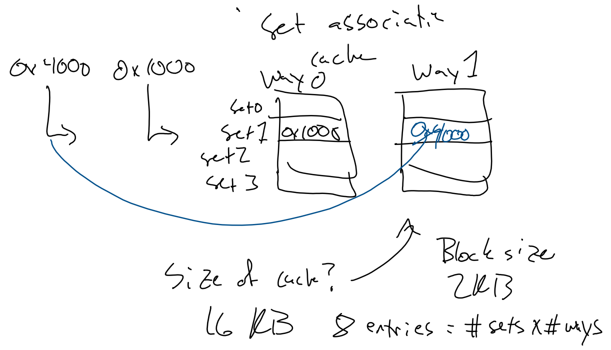
Set associativity → multiple ways for data to be placed

Direct-mapped cache → each address only maps to a single location



miss
↳ conflict miss

A miss that wouldn't happen if we had more associativity



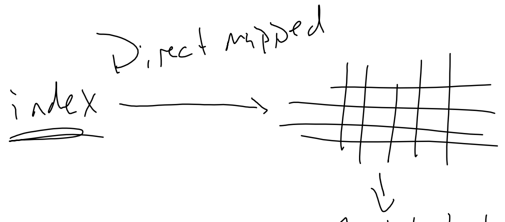
Size of cache? 16 KB
Block size 2KB
8 entries = # sets x # ways

What is better?

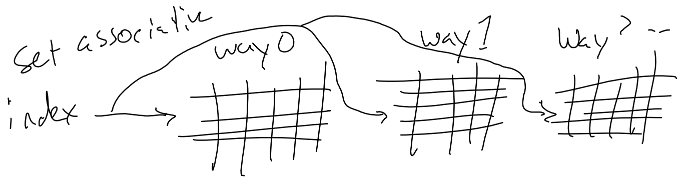
Performance → reduce miss ratio → more associative

fully associative # of ways = lines in cache → any address can map to any location
↳ minimize conflict misses

Downside of associativity is power and area



- 1 block + 1 tag
- compare 1 tag



- # ways (3) blocks + # ways tags
- compare many tags
- usually in parallel

Want high associativity when hit rate is very important → if memory is far away or high miss latency

---

address

accessing SA cache

tag	index	offset

bits whatever is left

32 bit address → 15 bits

direct-mapped # of bits log of # lines → 12 bits

# of bits → log size of cache line

16 B cache line → 4 bits

[illegible]

# of [illegible]
8192 lines → 15[illegible]

Changes for SA?

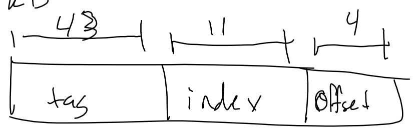
now index bits → log(# of lines in a way or # of sets)
↳ log(# of unique addressable locations in cache)

# of index bits direct-mapped ⇒ log(# of lines) # sets = $\frac{\#\text{ lines}}{1}$
One way

# of index bits for fully associative cache ⇒ ∅ index bits

Example

64 kB cache



What is the associativity?

# sets = 2048 (2^{11}) # of lines = $\frac{64kB}{16B}$ = 4,096

line size = 16 (2^4)

# ways = $\frac{\#\text{ lines}}{\#\text{sets}}$ = $\frac{4096}{2048}$ = 2 → $\boxed{\text{2-way set associative}}$

lines in cache = sets × ways

Equations

Blocks/lines = sets × ways

Sets = $2^{\text{index bits}}$

Size = block size × sets × ways

Offset bits = log(block size)

RISC-V 32 bit
32 bit addresses
2^{32} → 4 GB
most one app can address

Kinds of misses

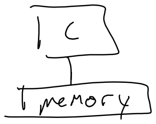
☆ Conflict misses result from low associativity
↳ reduce by incr. set associativity

☆ Cold misses / compulsory misses → first access to an address
↳ reduce by prefetching
↳ generally hard to reduce these misses

☆ Capacity misses → an address was evicted before it was reused
↳ reduce these by making cache bigger
↳ smarter replacement policy

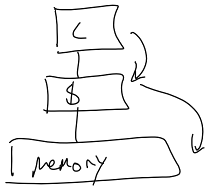
Predict performance for caches
↳ Average memory access time (AMAT)

Memory takes 50 cycles



AMAT = memory time = 50 cycles

Cache → 2 cycle hit time + 95% hit ratio



AMAT = $\underline{0.95 \times 2}$ + $\underbrace{0.05 \times 2 + 0.05 \times 50}_{\text{miss}}$

= 2 + 0.05 × 50

= 2 + 0.05 · 50

= 4.5 cycles

AMAT = $\underline{\text{hit time for cache}}$ + miss ratio x (miss time)

10 cycle hit time and 80% hit ratio for L2

AMAT = 2 + 0.05[10 + 0.2 x 50]

= 3