CSE338 PS – Chapter 5 Problems

Lokman ALTIN
lokman.altin@marmara.edu.tr

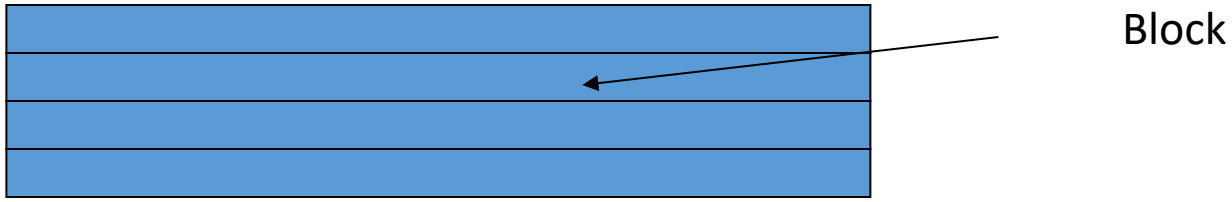
Multilevel Cache Example

- Given
  - CPU base CPI = 1, clock rate = 4GHz
  - Miss rate/instruction = 2%
  - Main memory access time = 100ns
- With just primary cache
  - Miss penalty = 100ns/0.25ns = 400 cycles
  - Effective CPI = 1 + 0.02 × 400 = 9

Example (cont.)

- Now add L-2 cache
  - Access time = 5ns
  - Global miss rate to main memory = 0.5%
- Primary miss with L-2 hit
  - Penalty = 5ns/0.25ns = 20 cycles
- Primary miss with L-2 miss
  - Extra penalty = 500 cycles ← ! 400 cycles
- CPI = 1 + 0.02 × 20 + 0.005 × 400 = 3.4
- Performance ratio = 9/3.4 = 2.6

Cache Architecture



The Block Allocation Problem

- 3 possible solutions:
  - Data from each address ***A*** will go to a fixed block.
    - *Direct Mapped Cache*
  - Data from each address ***A*** will go to any block.
    - *Fully associative cache*
  - Data from address ***A*** will go to a fix ***set*** of blocks.
    - Data may be put into any block within a set.
    - *Set associative cache*.

Direct Mapped Cache

- 1-word block size
- Cache size 8-words

000	
001	
010	
011	
100	
101	
110	
111	

A cache block can only go in one spot in the cache. It makes a cache block very easy to find, but it's not very flexible about where to put the blocks.

2-way Set Associative Cache

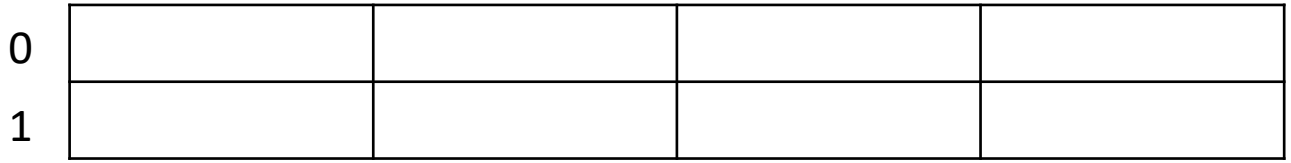
- 1-word block size
- Cache size 8-words

00		
01		
10		
11		

This cache is made up of sets that can fit two blocks each. The index is now used to find the set, and the tag helps find the block within the set.

4-way Set Associative Cache

- 1-word block size
- Cache size 8-words



Each set here fits four blocks, so there are fewer sets. As such, fewer index bits are needed.

Fully Associative Cache

- 1-word block size
- Cache size 8-words

No index is needed, since a cache block can go anywhere in the cache. Every tag must be compared when finding a block in the cache, but block placement is very flexible

Problem 1.

Size of Tags versus Set Associativity

Increasing associativity requires more comparators and more tag bits per cache block. Assuming a cache of 4K blocks, a 4-word block size, and a 32-bit address, find the total number of sets and the total number of tag bits for caches that are direct mapped, two-way and four-way set associative, and fully associative.

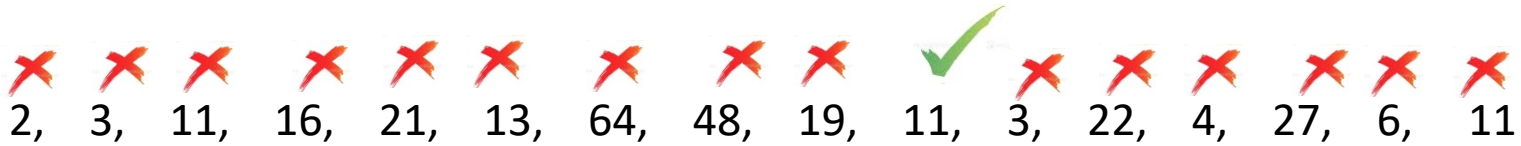
- 16 (2^4) bytes per block → 4 bits of offset
- 32 – 4 = 28 bits used for index and tag
- Direct Mapped Cache has the same number of sets as blocks
  - $2^2 * 2^{10} = 2^{12}$ → 12 bits of index
  - 32 – 12 – 4 = 16 bits of tag
  - 16 * (4K) = 64K tag bits

Problem 1 (cont.)

- Each degree of associativity decreases the number of set by a factor of 2,
  - This decreases index bits by 1.
  - This also increases tag bits by 1.
- 2-way set associative cache
  - There are 2K sets → $2^1 * 2^{10} = 2^{11}$ → 11 bits of index
  - 32 – 11 – 4 = 17 bits of tag
  - 17 * 2 * (2K) = 68K tag bits
- 4-way set associative cache
  - There are 1K sets → $2^0 * 2^{10} = 2^{10}$ → 10 bits of index
  - 32 – 10 – 4 = 18 bits of tag
  - 18 * 4 * (1K) = 72K tag bits
- Fully associative cache
  - There is only 1 set with 4K blocks
  - 32 – 4 = 28 bits of tag
  - 28 * (4K) * 1 = 112K tag bits

Problem 2.

Using the series of references given below, show the hits and misses and final cache contents for a two-way set-associative cache with one-word blocks and a total size of 16 words. Assume LRU replacement.



Blk_Addr = (Word address) mod N
N is the number of **sets!**

0	~~16~~ **48**	**64**
1		
2	**2**	
3	~~3~~ ~~19~~ ~~3~~ **11**	~~11~~ **27**
4	**4**	
5	**21**	**13**
6	**22**	**6**
7		

Hit # = 1
Miss # = 15

Hit Ratio = 1 /16
Miss Ratio = 15/16

Problem 3.

Using the series of references given below, show the hits and misses and final cache contents for a fully associative cache with one-word blocks and a total size of 16 words. Assume LRU replacement.

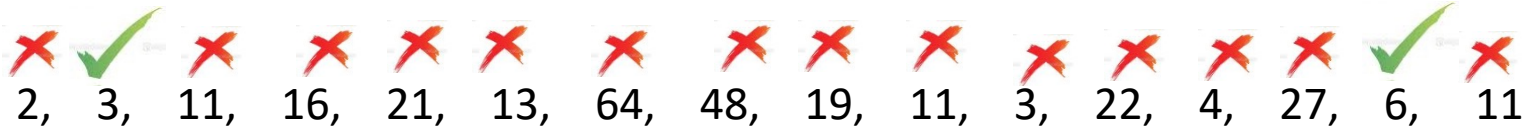
2	3	11	16	21	13	64	48	19	22	4	27	6			
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Hit # = 3
Miss # = 13

Hit Ratio = 3 /16
Miss Ratio = 13/16

Problem 4.

Using the series of references given below, show the hits and misses and final cache contents for a fully associative cache with four-word blocks and a total size of 16 words. Assume LRU replacement.



~~0,1,2,3~~ 12,13,14,15 ~~8,9,10,11~~ 24,25,26,27	~~8,9,10,11~~ 64,65,66,67 ~~0,1,2,3~~ 8,9,10,11	~~16,17,18,19~~ ~~48,49,50,51~~ 20,21,22,23	~~20,21,22,23~~ ~~16,17,18,19~~ 4,5,6,7

Hit # = 2
Hit Ratio = 2 /16

Miss # = 14
Miss Ratio = 14/16

Problem 5

Assume that cache size is 1024 KB and the size of the physical memory is 16 GB. Assume that the block size of 32 words, a 8-way set associative cache, and a 32-bit address (Assume that we consider byte addressing)

a. How many bits are needed for the following?

TAG	INDEX	OFFSET (TOTAL)
15 bits	10 bits	7 bits

- 32 x 4 bytes = 128 = 2^7 bytes (7 bits offset)
- 1024 KB / (128 * 8) = 2^{20} / (2^7 * 2^3) = 2^{10} blocks (10 bits index)
- Tag = 32 – (10+7) = 15 bits

Problem 5 (cont.)

b. Rewrite the above fields for each of the following changes?

b.1. Decrease the cache size from 1024 KB → 512 KB

No change in offset.

Index decreases by 1 index → Tag increments by 1 bit

512 KB / 128 = 2^{19} / 2^{7} = 2^{12} blocks

# of sets = 2^{12} / 8 way = 2^{9} (9 bits index)

b.2. Decrease the block size from 32 words → 16 words

Offset decrements by 1 bit

Index increments by 1 bit

16 x 4 bytes = 64= 2^{6} bytes (6 bits offset)
1024 KB / 64 = 2^{20} / 2^{6} = 2^{14} blocks
# of sets = 2^{14} / 8 way = 2^{11} (11 bits index)

b.3. Change 8-way set associativity to full associativity

32x 4 bytes = 128= 2^{7} bytes (7 bits offset)

	TAG	INDEX	OFFSET (TOTAL)
b1	16 bits	9 bits	7 bits
b2	15 bits	11 bits	6 bits
b3	25 bits	0 bits	7 bits

Problem 5 (cont.)

Assume that cache size is 1024 KB and the size of the physical memory is 16 GB. Assume that the block size of 32 words, a 8-way set associative cache, and a 32-bit address (Assume that we consider byte addressing)

c. What is the total size of the cache given in part A, including valid, tag and data field?

Each entry = Data bits + Tag bits + valid bit

= (32 x 4 x 8) + 15 + 1

= 1040 bits

Total Size = 1040 x $(2^{10} \times 8) = 1040 \times 2^{13}$

8 - way

Problem 6

Suppose you have the following direct mapped cache system: cache size = 4 blocks, 1 block = 4 words, cache hit time = 30 ns, time to read a block from main memory to cache = 150 ns.

6,	17,	25,	19,	7,	1,	11,	24,	8,	13,	18,	10

Blk_Addr = *floor*(word_address/words_per_block) mod N
N is the total number of blocks in the cache

a. Show the hits and misses for the given reference string of word addresses. Show the final content of the cache.

00	**~~16,17,18,19~~**	**~~0,1,2,3~~**	**16,17,18,19**	
01	**4,5,6,7**			
10	**~~24,25,26,27~~**	**~~8,9,10,11~~**	**~~24,25,26,27~~**	**8,9,10,11**
11	**12, 13, 14, 15**			

# of hits = 3
# of misses = 9

Problem 6 (cont.)

Suppose you have the following direct mapped cache system: cache size = 4 blocks, 1 block = 4 words, cache hit time = 30 ns, time to read a block from main memory to cache = 150 ns.

b. Compute the total access time for instruction fetches for the above program.

- Total time = total time for accessing cache + time for accessing memory
  (Hit / Miss) (Miss Penalty)

  = (9 + 3) x 30 + (9 x 150)

  = 360 + 1350

  = 1710 ns.

Problem 6 (cont.)

c. Assume that we have another 16-words cache, which is a two-way set associative cache with two word blocks. Assume LRU replacement. Show the hits and misses for the given reference string of word addresses. Show the final content of the cache.

6,	17,	25,	19,	7,	1,	11,	24,	8,	13,	18,	10

Blk_Addr = *floor*(word_address/words_per_block) mod N
Here *N* is the number of **sets**, NOT NUMBER OF BLOCKS!

00	~~**16,17**~~ ~~**0,1**~~ **8,9**	**24,25**
01	**18,19**	**10,11**
10	**12,13**	
11	**6,7**	

# of hits = 4
# of misses = 8