16.482 / 16.561: Computer Architecture and Design

Spring 2014

Lecture 8: Key Questions
April 10, 2014

1. How are blocks identified in the cache? Show how an address can be broken down and identify the role of each field in a cache access.

2. Example: Given the following:

   - 32-bit address
   - 32 KB direct-mapped cache
   - 64-byte blocks

What are the sizes for the tag, index, and block offset fields?

3. Example: Say we have the following cache organization
   - Direct-mapped
   - 4-byte blocks
   - 32 B cache
   - 6-bit memory addresses

Given the addresses 3, 4, 6, 11, 37, and 43, answer the following questions:

a. Which addresses belong to the exact same block (i.e., both tag and index are equal)?

b. Which addresses map to the same cache line, but are part of different blocks (i.e., index numbers are equal, but tags are different?

Hint: Must convert all addresses into binary, then determine the appropriate values for each field.

4. Explain what it means for a block to be evicted from the cache. When an eviction is required in a set-associative cache, how do we choose which block to evict?

5. Describe the two different cache write policies. Which one is more commonly used, and why?

6. Example: Given the final cache and memory state shown (after walking through the first five instructions in the direct-mapped cache access example):

MEMORY:

Address		Address	
0	78	8	18
1	29	9	21
2	120	10	33
3	123	11	28
4	18	12	19
5	150	13	200
6	162	14	210
7	173	15	225

CACHE:

V	D	Tag	Data	
1	0	1	18	21
0	0	0	0	0
1	1	1	19	29
0	0	0	0	0

Determine the new cache state, as well as any modified registers and/or memory blocks, after each access listed below.

Access	Modified register	Cache state					Modified mem. block
		V	D	Tag	Data		
`lb $t1,3($zero)`							
`sb $t0,2($zero)`							
`lb $t0,11($zero)`							

7. Example: Given a 4-way set associative cache, and five blocks (A, B, C, D, E) that all map to the same set, determine which block is evicted on the access to block E in each of the access sequences below if we use LRU replacement:

a. A, B, C, D, E

b. A, B, C, D, B, C, A, D, A, C, D, B, A, E

c. A, B, C, D, C, B, A, C, A, C, B, E

8. Example: Given the final set-associative cache and memory state shown (after walking through the first five instructions in the set-associative cache example):

MEMORY:

Address		Address	
0	78	8	18
1	29	9	21
2	120	10	33
3	123	11	28
4	18	12	19
5	150	13	200
6	162	14	210
7	173	15	225

CACHE:

V	D	MRU	Tag	Data	
1	0	1	10	18	21
1	1	0	11	19	29
0	0	0	00	0	0
0	0	0	00	0	0

Determine the new cache state, as well as any modified registers and/or memory blocks, after each access listed below.

Access	Modified register	Cache state						Modified mem. block
		V	D	MRU	Tag	Data		
`lb $t1,3($zero)`								
`lb $t0,11($zero)`								
`sb $t0, 2($zero)`								