ECE M116C - CS M151B Computer Architecture Systems - UCLA

# Homework 4

**Deadline:** Saturday, Nov. 12, 11:55 PM

Fall 2022

(Upload it to Gradescope.)

Q1. Assume that we have a 2-way set associative cache. Assume that the addresses are 10 bits. Assume that the SA cache has 2 rows/sets (and 2 ways/lines per set). Assume that each cache block is 8 bytes, and each way/line is 32 bytes.

Answer the following questions:

a. What is the size of C, B, and S?

**Blocks per way(or Line) = 32/8 = 4**

**2 Ways(or Lines) per Set and 2 Sets in the Cache = 4*2*2 = 16**

**C = Log(Blocks per Cache) = log 16 = 4**

**B = Log(Blocks per Line) = log 4 = 2**

**S = Log(Lines per Set) = log 2 = 1**

b. What is the cache's block offset, index, and tag size?

**Block per CacheLine/Way = 32/8 = 4**

**So offset = 2 bits.**

**2 sets so Index bits = 1**

**Tag = 10 - 2 - 1 = 7 bits.**

c. Complete the following table for the cache. Assume that the cache is filled already except for two lines. Use LRU replacement policy where (1) means the most recently used.

| HEX_addr | 10-bit Binary | tag in hex | index | offset |
|---|---|---|---|---|
| 111 | 0100010001 | 22 | 0 | 01 |
| 135 | 0100110101 | 26 | 1 | 01 |
| 304 | 1100000100 | 60 | 1 | 00 |
| 305 | 1100000101 | 60 | 1 | 01 |
| 110 | 0100010000 | 22 | 0 | 00 |

| SA **LRU** | Addresses and tags are shown in HEX. Compute the tag for each Way (W0, W1, …). | | | | |
|---|---|---|---|---|---|
| | Set 0 | | Set 1 | | |
| Address | W0 | W1 | W0 | W1 | Hit/Miss |
| 110 | 00 (0) | 22 (1) | 00 (0) | 00 (1) | M |
| 136 | 00 (0) | 22 (1) | 26 (1) | 00 (0) | M |
| 121 | 24 (1) | 22 (0) | 26 (1) | 00 (0) | M |
| 305 | 24 (1) | 22 (0) | 26 (0) | 60 (1) | M |
| 111 | 24(0) | 22(1) | 26(0) | 60(1) | H |
| 135 | 24(0) | 22(1) | 26(1) | 60(0) | H |
| 304 | 24(0) | 22(1) | 26(0) | 60(1) | H |
| 305 | 24(0) | 22(1) | 26(0) | 60(1) | H |
| 110 | 24(0) | 22(1) | 26(0) | 60(1) | H |

Q2. For any of the techniques below, comment on each column. For each cell put *Increase* (I), *Decrease* (D), or *Unchanged* (U).

| | Hit Time | Miss Rate | Miss Penalty | Compulsory Miss | Conflict Miss | Capacity Miss |
|---|---|---|---|---|---|---|
| **Prefetching**[1] | **U** | **D** | **U** | **D** | **U** | **U** |
| **Double the associativity (capacity and line size constant)**[2] | **I** | **D** | **U** | **U** | **D** | **U** |
| **Bigger block size**[3] | **I** | **D** | **I** | **U** | **D** | **D** |

| | | | | | | |
|---|---|---|---|---|---|---|
| **More cache levels[4]** | **U** | **U** | **D** | **U** | **U** | **U** |
| **Victim cache[5]** | **I** | **D** | **U** | **U** | **D** | **U** |

**Notes:**

**[1] Prefetching will speculatively fetch data in cache before core requests it, and therefore it can help with Compulsory misses, leading to reducing miss rate.**

**[2] Having more ways in a set helps decreasing conflict miss. Hit time will increase, if the cache does not have a parallel way lookup.**

**[3] Assuming that a bigger block size increases the size of the cache. Otherwise, unchanged.**

**[4] Assuming we are talking about hit time of only L1 cache, having an L2 cache will help decreasing Miss Penalty**

**[5] High traffic Set's existing data will be evicted to Victim caches (rather than main memory or Lower level caches), therefore decreasing the Miss rate and conflict misses.**