

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Comp 120 Computer Organization

Spring 2005

Solutions to Problem Set #9

Problem 1. “Cache Accounting”[6 points each]

- (A) Maximum number of words in cache = 4 words/line * 16 lines = 64 words
Number of bits in address used for selecting line: $\log_2(16) = 4$
Number of bits in address used for tag: $32 - 4 - 2 - 2 = 24$
- (B) It indicates whether the data in that line is valid or not. If V is 1, the data is valid.
Otherwise, the data is invalid.
- (C) In the 32-bit address [31:0], bits [7:4] are used to select row and bits [3:2] are used to select column. For 0x3328C, [7:4] are 1000, and [3:2] is 11. So the data is stored in row 8, column 3. The corresponding tag field is 0x000332.
- (D) For 0x12368 and 0x322F68, the two memory locations are not going to appear in the cache at the same time, because they both require being stored at row 6 and column 2. For 0x2536038 and 0x10F4, there is no such conflict, so they can be both present in the cache at the same time.
- (E) The entire block needs to be fetched which is 4-word.

Problem 2 “Show Me the bits?”[6 points each]

- (A) For both instruction and data cache in L1, they are both 2-way set associative, with each DMCS containing 64 bytes/line * 512line = 256K bits = 32K bytes. We need $\log_2 512 = 9$ bits to select cache line, $\log_2 64 = 6$ bits for block selection, and remaining 17 bits to be tags.

For L2 cache, is 16-way associative. So each DMCS also contains 32bits/word * 16 words/line * 512line = 256K bits = 32K bytes. So the bit division in address is the same with instruction and data in L1.

- (B) For instruction in L1,
Number of bits = 64 bytes * 8 bits/byte * 512 lines * 2 +
512 lines * 17 tags/line * 2 + 512 * 1
= 2^{19}
where 512*1 bits are for replacement.

For data in L1,
Number of bits = 64 bytes * 8 bits/byte * 512 lines * 2 +
512 lines * 17 tags/line * 2 + 512 * 1 + 4*512*2

$$= 2^{19}$$

where 512×1 bits are for replacement, and $4 \times 512 \times 2$ is for dirty bits.

For data in L2,

$$\begin{aligned} \text{Number of bits} &= 64 \text{ bytes} \times 8 \text{ bits/byte} \times 512 \text{ lines} \times 16 + \\ &512 \text{ lines} \times 17 \text{ tags/line} \times 16 + 512 \times 0 + 4 \times 512 \times 16 \\ &= 2^{22} \end{aligned}$$

Here dirty bits become 512×0 because L2 uses RABDOM replacement.

(C) For instruction,

$$0.92 \times 1 + (1-0.92) \times 0.98 \times (1+4) + (1-0.92) \times (1-0.98) \times (1+4+20) = 1.352 \text{ cycles}$$

For data access,

$$0.88 \times 1 + (1-0.88) \times 0.98 \times (1+4) + (1-0.88) \times (1-0.98) \times (1+4+20) = 1.528 \text{ cycles}$$

Overall effective access time,

$$\text{Instruction} \times 100/120 + \text{data access} \times 20/120 = 1.381 \text{ cycles}$$

(D) For instruction,

$$0.92 \times 1 + (1-0.92) \times 0.95 \times (1+4) + (1-0.92) \times (1-0.95) \times (1+4+20) = 1.4 \text{ cycles}$$

For data access,

$$0.88 \times 1 + (1-0.88) \times 0.95 \times (1+4) + (1-0.88) \times (1-0.95) \times (1+4+20) = 1.6 \text{ cycles}$$

Overall effective access time,

$$\text{Instruction} \times 100/120 + \text{data access} \times 20/120 = 1.433 \text{ cycles}$$

(E) Since the instruction access time occupies majority in the average access time, decrease the instruction access time will decrease the overall average access time. Since the total size is fixed, to increase the hit ratio, we need to increase either associability or blocksize.

Problem 3 "Trip Around the Block" [5 points each]

(A) $(1-m) \times 1 + m \times (1+2+3+B) = 1 + 5m + Bm$

(B) 4.

(C) The expression becomes $1 + 5m + Bm/4$, and the answer is 16.

(D) The expression becomes $(1-m) \times 1 + m \times (1+2+9+B/4)$, and the answer is 64.

Problem 4 “I’m Almost LRU” [5 points each]

(A)

| Addr | Line # | Miss? | Replacement bits |
|------|--------|-------|------------------|
| 100 | 2 | M | 1,01 |
| 1000 | 0 | M | 0,11 |
| 101 | 3 | M | 1,10 |
| 102 | 1 | M | 0,00 |
| 100 | 2 | | 1,01 |
| 1001 | 0 | M | 0,11 |
| 101 | 3 | | 1,10 |
| 102 | 1 | | 0,00 |
| 100 | 2 | | 1,01 |
| 1002 | 0 | M | 0,11 |
| 101 | 3 | | 1,10 |
| 102 | 1 | | 0,00 |
| 100 | 2 | | 1,01 |
| 1003 | 0 | M | 0,11 |
| 101 | 3 | | 1,10 |
| 102 | 1 | | 0,00 |

Miss ratio is $\frac{1}{4}$.

(B) Half the lines step by step until there are only two lines in each small group. The first bit is used to indicate which half is most recently used, the second is used to indicate which $\frac{1}{4}$ among the half is used most recently, and so on. The last bit is used to indicate which line is least recently used in the pair of lines. There are totally $\log N$ levels, the number of bits needed is $1 + 2 + 4 + \dots + N/2 = N-1$

(C) If the replacement bits are (0,00), line pair (2, 3) is accessed less recently than pair (0, 1), and 2 is accessed before 3 while 0 is accessed before 1. There are three possible access pattern which are (2, 3, 0, 1), (2, 0, 3, 1), and (0, 2, 3, 1). If the true access order is (0, 2, 3, 1), then the second least recently accessed line 2 is more likely to be replaced than line 0.

(D)

| Addr | Line # | Miss? | Replacement bits |
|------|--------|-------|------------------|
| 100 | 2 | M | 1,01 |
| 101 | 0 | M | 0,11 |
| 102 | 3 | M | 1,10 |
| 103 | 1 | M | 0,00 |
| 101 | 0 | | 0,11 |
| 102 | 3 | | 1,10 |
| 104 | 1 | M | 0,00 |

When the last miss happens, line 2 is least recently used, but in Lori’s algorithm, the second least used line 1 will be replaced.