**컴퓨터구조 과제**

**201524404 강민진**

**4.3 For the hexadecimal main memory addresses 111111, 666666, BBBBBB, show the following information, in hexadecimal format:**

**a. Tag, Line, and Word values for a direct-mapped cache, using the format of Figure 4.10**

**(8, 14, 2)**

111111 -> 0001/0001 00/0100/0100/0100 01 -> Tag : 11, Line : 0444, Word : 1

666666 -> 0110/0110 01/1001/1001/1001 10 -> Tag : 66, Line : 1999, Word : 2

BBBBBB -> 1011/1011 10/1110/1110/1110 11 -> Tag : BB, Line : 2EEE, Word : 3

**b. Tag and Word values for an associative cache, using the format of Figure 4.12**

**(22, 0, 2)**

111111 -> 00/0100/0100/0100/0100/0100 01 -> Tag : 044444, Word : 1

666666 -> 01/1001/1001/1001/1001/1001 10 -> Tag : 199999, Word : 2

BBBBBB -> 10/1110/1110/1110/1110/1110 11 -> Tag : 2EEEEE, Word : 3

**c. Tag, Set, and Word values for a two-way set-associative cache, using the format of Figure 4.15**

**(9, 13, 2)**

111111 -> 0/0010/0010 0/0100/0100/0100 01 -> Tag : 022, Line : 0444, Word : 1

666666 -> 0/1100/1100 1/1001/1001/1001 10 -> Tag : 0CC, Line : 1999, Word : 2

BBBBBB -> 1/0111/0111 0/1110/1110/1110 11 -> Tag : 177, Line : 0EEE, Word : 3

**4.15 Consider the following code:**

**for(i = 0; i < 20; i++)**

**for(j = 0; j < 10; j++)**

**a[i] = a[i]\*j**

**a. Give one example of the spatial locality in the code. (공간)**

for loop(i) 외부 for 루프는 a[0] -> a[1] 이렇게 순차적으로 메모리에 접근할 확률이 높은 것을 이용하여 성능향상에 기여하였다.

**b. Give one example of the temporal locality in the code. (시간)**

for loop(j) 내부 for 루프는 여러 번 사용되어서 temporal locality 를 이용하였다.(최근에 사용된 명령이나 데이터 다시 사용될 확률이 높다는 것을 이용.)

**4.12. Consider a computer with the following characteristics: total of 1MB of main memory word size of 1 byte; block size of 16bytes; and cache size of 64kB.**

**a. For the main memory addresses of F0010, 01234, a nd CABBE, give the corresponding tag, cache line address, and word offsets for a direct-mapped cache.**

F0010 = 1111 0000/0000/0001 0000 -> Tag : F, Line : 001, Word : 0

01234 = 0000 0001/0010/0011 0100 -> Tag : 0, Line : 123, Word : 4

CABBE = 1100 1010/1011/1011 1110 -> Tag : C, Line : ABB, Word : E

**b. Give any two main memory addresses with different tags that map to the same cache slot for a direct-mapped cache.**

0FFFF, 1FFF1

**c. For the main memory addresses of F0010 and CABBE, give the corresponding tag and offset values for a fully-associative cache.**

F0010 : Word : 0h, Tag : F001h

CABBE : Word : Eh, Tag : CABBh

**d. For the main memory addresses of F0010 and CABBE, give the corresponding tag, cache set, and offset values for a two-way set-associative cache.**

(4, 11, 5) word, cache set, tag

F0010 : 1/1110 000/0000/0001 0000 -> Word : 0, Cache Set : 001, Tag : 1E

CABBE : 1/1001 010/1011/1011 1110 -> Word : E, Cache Set : 2BB, Tag : 19

**4.22 A computer has a cache , main memory, and a disk used for virtual memory. If a referenced word is in the cache, 20 ns are required to access it. If it is in main memory but not in the cache, 60 ns are needed to load it in the cache, and then the reference is started again. If the word is not in main memory, 12ms are required to fetch the word from disk, following by 60 ns to copy it to the cache, and then the reference is started again. The cache hit ratio is 0.9 and the main memory hit ratio is 0.6 What is the average time in nanoseconds required to access a referenced word on this system?**

Cache access time : 20 ns

cache hit ratio : 0.9 <-> cache miss ratio : 0.1

memory access time : 80(20 + 60) ns

memory hit ratio : 0.6 <-> memory miss ratio : 0.4

disk access time = 12ms = 12000000ns

average access time : (0.9 \* 20ns) + (0.1 \* 80ns) = 26ns (reference가 메모리 혹은 캐시에 있다고 가정하면.)

average access time: 0.9\*20ns + 0.1\*(0.6\*80ns + 0.4\* (12000000ns + 80ns)) (디스크에 갈 가능성을 고려하면)