Fall 2013: COMP 7300 Advanced Computer Architecture

Test 2 (100 pts)

Grading policy:
¼ Credit for correct answer
¾ Credit for wellwritten and solid justification/facts/arguments. **Show your work.**

For Test 2, we assume a CPU with a 32-bit address bus.

>>>>>> **Last Question (Last Page) is easy** <<<<<<<<<<

A) Cache (Questions) Justify your answers

1) We consider a direct mapped 16 KB cache with 512 byte blocks. The CPU generates address 25,000.

   a. **(4 points)** What is the block address of Block ***B*** containing address 25,000?

B.A = Bloc Address = $\lfloor \text{Address} / \text{Block Size} \rfloor = 48$

$$\text{Address} = \overbrace{\underbrace{1}\,\underbrace{10\ 000}}^{\text{B.A}} \quad \text{OR} \quad \overbrace{1\ 1010\ 1000}^{\text{offset}}$$

   b. **(4 points)** What is Block B's tag?

tag = 1   index = Block Number = 16

   c. **(4 points)** Where will Block ***B*** be stored in the cache?

Block Number = Block Address % Number of Sets

= 48 % 32

= 16

2) We consider a 4-way associative 16 KB cache with 512 byte blocks. The CPU generates address 25,000.

   a. **(1 point)** What is the block address of Block ***B*** containing address 25,000?

Same as above

   b. **(4 points)** Where will Block ***B*** be stored in the cache? 1

$$\text{Number of sets} = \frac{\text{Cache Size (in Blocks)}}{n \text{ (from n-way)}}$$

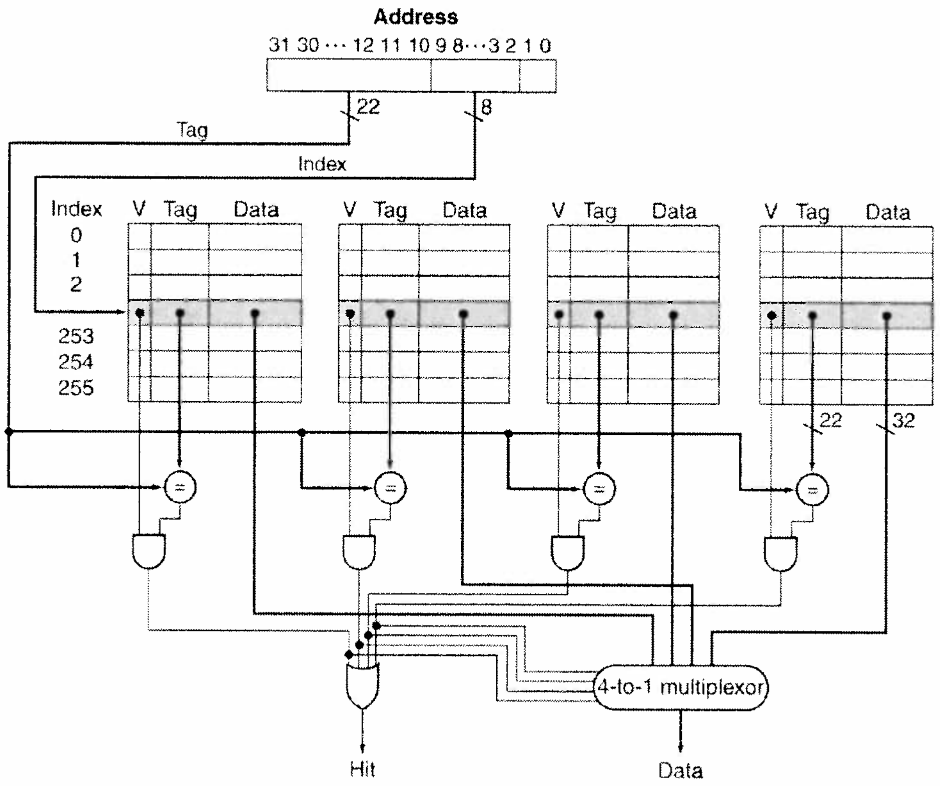
$$= \frac{32}{4} = 8 \text{ sets}$$

Block Number = 48 % 8 = 0

3) **(9 points)** Compare a direct mapped cache versus a fully associative cache in terms of hit time, energy cost, and miss rate. For each metric (hit time, energy, and miss rate), briefly justify why one outperforms (if applicable) the other.

Refer text book

4) Consider the cache on the figure below. Answer the following questions based on the figure below



a. **(2 points)** Is this a direct mapped or an associative cache? Explain

Associative because the index points to four blocks (4 blocks per set as shown)

b. **(6 points)** If associative, is it 2 way, 4 way.... or x-way associative? Explain

4-way because 4 blocks per set

c. **(4 points)** If associative, how many sets does this cache have? Explain

256 because the index has 8 bits

$2^8 = 256$

5) **Exercise 1:** We assume that the data cache size is 16 KB (data) and the block size is 4 KB. We assume that M[i][j] is adjacent to M[i][j+1] in the memory and that the cache is fully associative using LRU replacing. Consider the following code:

```
int A[32][256];
int i,j;
for (j = 0; j < 256; j++){
        for (i = 0; i < 32; i++)
              A[i][j] = 2 * A[i]j];
}
```

a) **(12 points)** We assume that the code is in a separate instruction cache and the variables i and j are in registers. How many compulsory, capacity, and conflict misses occur for the above code?

[no conflict miss because fully associative]

each block contains 4 lines

cache can accomodate 16 lines

Matrix has $\frac{32 \times 256 \times 4}{4K} = 8$ blocks

$\Rightarrow$ 8 compulsory misses

After A[3][0], each reference will trigger a capacity miss [except last 4 references]

$\Rightarrow$ number of capacity misses $= 255 \times 8$

b) **(16 points)** Would a loop interchange decrease the number of misses? If yes, How many compulsory, capacity, and conflict misses occur for the above code with loop interchange?

Yes: when referencing a line, we process it completely before discarding it.

$\Rightarrow$ 8 compulsory miss (cannot avoid these)

0 capacity miss [at no time is a block discarded while needed data.

0 conflict (see above)

6) The block size is 128 bytes. The cache size is 32 KB. Assume a fully associative cache with LRU replacement. We consider the following algorithm to achieve an in-place transposition of a matrix:

```
int i, j; // in registers
char A[256][256];
for (j = 0; j < 256; j++){
      for (i = j+1; i < 256; i++) {
            // swap A[i][j] and A[j][i]
            A[i][j] = A[i][j] ^ A[j][i];
            A[j][i] = A[i][j] ^ A[j][i];
            A[i][j] = A[i][j] ^ A[j][i];
      }
}
```

a) **(8 points)** We assume that the code is in a separate instruction cache and the variables i and j are in registers. How many compulsory misses occur for the above code?

Number of compulsory misses

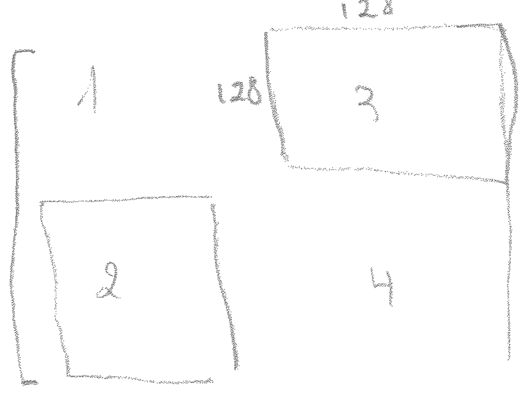
= Number of blocks in matrix

$$= \frac{256 \times 256}{128} = \frac{2^8 \times 2^8}{2^7} = \frac{2^{16}}{2^7} = 2^9 = 512$$

b) **(4 points)** Would a loop interchange decrease the number of misses? If yes, How many compulsory misses occur for the above code with loop interchange?

No. We access A[i][j] and the transpose A[j][i] in the same statement $\Rightarrow$ when order is ideal for A[j][i] it will not be for A[i][j] and vice versa.

c) **(10 points)** What should be the minimal size of the cache to take advantage of a blocked execution?



we need access to both 2 and 3 quadrant.

$\Rightarrow$ we need

$2 \times 128 \times 128$

$= 2 \times 2^7 \times 2^7$

$= 2^{15} = 32\text{ KB}$

B) Virtual Memory

(12 points) We consider a 32 bit address bus and a 512 KB physical memory. Page size is 8 KB. Assuming a validity bit and a dirty bit, what is the size of the page table? **Show the different steps to determine the page table size.**

Page Table size = number of entries x size of entry

number of entries = number of pages

$= \frac{\text{Addressing Space}}{\text{Page Size}} = \frac{2^{32}}{2^{13}} = 2^{19}$

size of entry = 1 V bit + 1 D bit + number of bits in frames

Number of frames = $\frac{\text{Size Physical Space}}{\text{Frame Size}}$

$= \frac{512\text{ KB}}{8\text{ KB}} = \frac{2^{19}}{2^{13}} = 2^{6}$

6

then size of entry = 1 + 1 + 6 = 8.

$\Longrightarrow$ Pagetable Size = $2^{19} \times 8$

$= 2^{19}$ Bytes