CS61C Spring Homework 5

TA: Donggyu Kim, Nolan Lum

Due Sunday, April 5th, 2015 @ 23:59:59

- Update 3/26 9:50 PM: Added a new direction for the cache miss rate in Problem 2.b.
- Update 3/30 7:10 PM: Fixed a typo in hw.txt. Please use a new template for your submission
- Update 3/30 11:40 AM: Any change for the caches does not change the clock time and frequency(2GHz), but may change their clock cycles. This assumption does not affect your solution unless you worry about the change of the clock time for Problem 1.
- Update 4/3 12:15 PM: Please check the Piazza post about the cache miss types before submitting.
- Update 4/4 3:10 PM: Cache miss rates should be given as a fraction or three significant digits.

Goals

This assignment will cover caches and floating point numbers.

Setup

You should writeup your solutions in `hw5.txt`. **Please use the provided format in `hw5.txt`.** File names are case-sensitive and the submission program will not accept your submission if your file names differ at all from those specified. Detailed submission instructions are given at the bottom of this page. Failure to follow these will result in loss of credit.

Exercises

Problem 1: Cache Block Size, Associativity, and AMAT (15 pts)

Suppose we have 16 bit byte-addresses and the clock frequency is 2GHz. We have the cache parameters as follows:

- Cache size: 4KiB
- Block size: 1 word (4 bytes)
- Cache hit time: 2 cycles
- Cache miss time: 100 cycles

Now suppose we access the following addresses, in order:

```
0x0000, 0x0004, 0x0008, 0x000c, 0x1000, 0x1004, 0x1008, 0x100c, 0x0000, 0x0004, 0x0008, 0x000c
```

a. First, we start with a direct-mapped cache.

1. What are the widths of the tag, set index, and block offset fields?
2. Is each memory access a cache hit or a miss? Also, identify the cache miss type if it's a miss, and fill in the following table.

Address	Cache Hit / Miss?	Miss Type(Compulsory, Capacity, or Conflict)
0x0000		
0x0004		
0x0008		
0x000c		
0x1000		
0x1004		
0x1008		
0x100c		
0x0000		
0x0004		
0x0008		
0x000c		

3. Calculate the miss rate and the AMAT in ns. How much does it decrease or increase the memory access time compared to having no cache?

b. Next, we increase the block size to 2 words. The overall cache size remains unchanged.

1. Repeat (a.1) for a cache with a block size of 2 words.
2. Repeat (a.2) for a cache with a block size of 2 words.
3. Calculate the miss rate. Note that we have different hit or miss times (or both) with the increased block size. Choose the valid one among the following timing parameters, and calculate the AMAT in ns. How much does it decrease or increase the memory access time compared to having no cache?
   - Cache hit time: 1, 2, 3 cycles
   - Cache miss time: 90, 100, 110 cycles

c. Lastly, we increase the associativity of the cache in (b) whose block size is two words. The overall cache size remains unchanged.

1. Repeat (a.1) for a two-way set associative cache.

2. Repeat (a.2) for a two-way set associative cache.
3. Repeat (b.3) for a two-way set associative cache.

Problem 2: Cache Friendly Programming (5 pts)

The following C program is run (with no optimization) on a processor with a direct-mapped data cache with a size of 1KiB and a block size of 16 bytes.

```
int i, j, array[256*256];

/* ... */

for (i = 0 ; i < 255 ; i++) {
  for (j = 0 ; j < 256 ; j++) {
    array[256*j] += array[256*j + i + 1];
  }
}
```

Assume `sizeof(int) == 4` and `array == 0x4000`.

a. Calculate the miss rate.

b. Rewrite the code to improve the performance (don't use cache blocking). What is the miss rate in this case? Give a fraction or three significant digits for the percentage.

c. Will a write-back cache be helpful with the code in (b)? Explain briefly, in one or two sentences.

Problem 3: Floating Point Numbers (10 pts)

If you're starting this homework early and we haven't covered this yet don't worry. We'll be going over floating point representation in Thursday's lecture. For the following questions, we will be referring to the IEEE 32-bit floating point representation ***except*** with a 6 bit exponent (bias of 2^6/2 - 1 = 31) and a denorm implicit exponent of -30.

a. Convert -42.375 to floating point format. Write your answer in hexadecimal.
b. Convert the floating point number 0xD3510000, which follows the 6 bit exponent representation as described above, to decimal. Please specify infinities as `+inf` or `-inf`, and not a number as `NaN`.
c. What's the smallest non-infinite positive integer (an integer has nothing to the right of the decimal) it CANNOT represent? Leave your answer in decimal (ex: 12).
d. What's the smallest positive value it can represent that is not a denorm? Leave your answer as a power of 2 (ex: 2^x).
e. What's the smallest positive value it can represent? Leave your answer as a power of 2 (ex: 2^x).

Submission

There are **two** steps required to submit hw5.txt. Failure to perform both steps will result in loss of credit:

1. First, you must submit using the standard unix submit program on the instructional servers. To do so, follow these instructions after logging into your cs61c-XX class account:

   ```
   $ mkdir ~/files_for_submit
   $ cd ~/files_for_submit
   $ mkdir hw5
   $ cd hw5
   $ cp [Your hw5 text file location] hw5.txt              # replace the braces with the location of your hw5.txt file
   $ submit hw5
   ```

   Once you type `submit hw5`, follow the prompts generated by the submission system. It will tell you when your submission has been successful and you can confirm this by looking at the output of `glookup -t`.

2. Additionally, you must submit hw5.txt to your GitHub repository. To do so, follow these instructions after logging into your cs61c-XX class account:

   ```
   $ cd ~/work                                            # this is the location of your git repo on your class account
   $ mkdir hw5
   $ cd hw5
   $ cp [Your hw5 text file location] hw5.txt             # replace the braces with the location of your hw5.txt file
   $ cd ..
   $ git add hw5/hw5.txt
   $ git commit -m "Homework 5 submission"
   $ git tag "hw5"                                        # The tag MUST be "hw5". Failure to do so will result in loss of credit.
   $ git push origin master --tags                        # Note the "--tags" at the end. This pushes tags to github
   ```