EECS 470 Winter 2024
Homework 4

Due to lecture changes, this is due Thursday 3/28. That's a week later than the class schedule says. Quiz will be on 4/2, HW5 still expected to be due on 4/16 but we'll see.

This is an individual assignment; all of the work should be your own. Assignments that are difficult to read will lose at least 50% of the possible points and we may not grade them at all. If you use references other than the text and class notes, be sure to cite them! Problems 1 and 6 saw minor changes/clarifications on 3/18. Those changes are in red.

---

1. Consider the following access pattern: A, B, C, B, A. Assume that A, B, and C are memory addresses each of which are in a different block of memory. Further, assume A, B and C are generated in a uniformly random way **and that a "true" LRU replacement algorithm is used. Further, assume that any given block has an equal chance of being placed in either "way" if both blocks are invalid.** *To receive credit, you must show your work*. **[11 points]**

    a. What is the stack distance of the second access to "A"? **[1]**
    b. What is the probability that the second instance of "A" will be a hit if:
        1. The cache has 2 lines and is fully-associative. **[1]**.
        2. The cache has 4 lines and is fully-associative. **[1]**
        3. The cache has 8 lines and is direct-mapped. **[1]**
        4. The cache has 4 lines and is two-way associative. **[2]**
        5. The cache has 4 lines and is a directed-mapped hash-cache. **[1]**
        6. The cache has 4 lines and is 2-way skew-associative. **[4]**

2. Given a virtually indexed, physically-tagged cache that is four-way associative and has 16-byte blocks: **[6 points]**
    a. What is the largest *total size* the cache could have if pages were 4KB in size? **[2]**
    b. What is the largest *number of sets* it could have in that case? **[2]**
    c. How would a and b change if the cache were still four-way associative but the block size was 64 bytes? **[2]**

3. Consider two caches, both with 4 lines total and fully associative. Cache A uses true LRU replacement and cache B uses the pseudo-LRU replacement algorithm described in class. Both are initially empty (all lines invalid). The cache will always allocate an invalid block before evicting a valid one. **[15 points]**
    a. Provide the shortest reference pattern (fewest accesses) where the true-LRU cache will get one more hit than the pseudo-LRU cache. **[7]**
    b. Provide the shortest reference pattern (fewest accesses) where the pseudo-LRU cache will get one more hit than the true-LRU cache. **[8]**

4. **[TEAM PROBLEM]** Your entire team should each turn in the same answer to this question.
    Write a "long" (hundreds of instructions *executed*, it may be shortish in terms of instructions in the program) RISC-V testbench that would help you figure out:
    a. If your X-way superscalar processor met the goal of getting an IPC greater than X-1.
    b. If your processor met the goal of executing back-to-back dependent instructions without a stall between them.

    In each case provide not only the testbench, but explain how looking at the CPI would help you know if your design was meeting the stated goal (note: "a" is a required project goal, "b" is not.) **[10 points]**

5. Answer the following questions about memory scheduling **[10 points]**
    a. Define the terms “memory disambiguation” and “store-to-load forwarding”. Explain how an LSQ helps with memory disambiguation **[2]**
    b. (Déjà vu) Define the terms “spill”, “fill”, and “register pressure” in the context of register usage. **[2]**
    c. Using the terms defined in part b, explain one reason we may see a lot of store-to-load forwarding in the LSQ. **[2]**
    d. Consider a unified LSQ with the head holding instruction “A”, the next entry holding “B”, etc. until you get to the tail which holds instruction “F”. For this question assume all memory access are 4 bytes and are all aligned. **[4]**
        1. If D is a load, describe under what circumstances you can be certain that D will need to get its data from memory?
        2. If D is a load, describe under what circumstances you can be certain that D can get its data from a store in the LSQ.

6. Consider the following pseudo-assembly program on the right. You have two nearly-ideal out-of-order processors other than as noted here. Both take 1 cycle to execute adds and stores. Both take 4 cycles to execute loads and multiplies. Processor A can “magically” resolve memory dependencies (likely via speculation). Processor B stalls loads until all older stores have committed. **[10 points]**
    a. Draw the effective data dependency graphs for the given code as seen by each of processor A and processor B. **[6]**
    b. Compute the execution time of A and of B. You are to ignore pipeline issues (e.g. fetch, commit) and just worry about the data flow. **[4]**

```
A:  R1=R1+1
B:  R2=R1*R4
C:  R3=R3+R2
D:  MEM[R3+100]=R1
E:  R3=MEM[R5+100]
F:  R4=R3*R5
G:  R5=R4*R6
```

7. In class it was mentioned that virtual caches can lead to various types of aliasing. There are two types of problems. In the literature, one is called the "synonym problem" and another is called the "homonym problem". Clearly define those terms in your own words and give an example of each. As always, be sure to cite your sources (hint: Wikipedia has pretty good coverage)! **[8 points]**

8. Consider a processor which, when running a given application performs 2 billion loads and 1 billion stores per second. Assume the following is true: **[10 points]**
    - The processor’s multi-level cache system gets a 97% hit rate on both loads and stores.
    - Cache lines are 32-bytes in size
    - There is no prefetching and the instruction cache never misses.
    - 5% of all lines evicted from the last level of the cache are dirty.
    - The cache is write-back and write-allocate.
    - There are no coherence misses.
    - All loads and stores are to 4-byte values.

    a. What is the read bandwidth (bytes/second) on the bus? Show your work. **[5]**
    b. What is the write bandwidth (bytes/second) on the bus? Show your work. **[5]**

9. Consider a case of having 3 processors using a snoopy MESI protocol where the memories can snarf data. All three have a 2-line direct-mapped cache with each line consisting of 16 bytes. The caches begin with all lines marked as invalid. Fill in the following tables indicating
   - If the processor gets a hit or a miss in its cache. *<u>If the data isn't in a state that is needed to avoid a bus transaction, it's a miss</u>*.
   - What bus transaction(s) (if any) the processor performs (BRL, BWL, BRIL, BIL)
   - If a HIT or HITM (or nothing) occurs on the bus during snoop.
   - For misses only, indicate if the miss is compulsory, capacity, conflict, or coherence. A coherence miss is one where there would have been a hit were it not for the actions of another processor.
   - For "state after action" indicate the state of that local processor line after the read/write

   In the event more than one bus transaction occurs due to a given memory read/write indicate the response for each bus transaction. Finally, indicate the state of the processor after all of these memory operations have completed. The operations occur in the order shown and the cache starts empty. **[20 points]**

Processor	**Address**	**Read/ Write**	**Hit/ Miss**	**Bus transaction(s)**	**HIT/ HITM**	**state after transaction**	**"4C" miss type (if any)**
1	0x120	Read					
1	0x120	Write					
1	0x100	Read					
1	0x120	Write					
2	0x120	Read					
2	0x110	Write					
1	0x110	Read					
3	0x100	Write					
1	0x100	Read					
1	0x120	Read					
3	0x130	Read					
3	0x100	Read					
1	0x100	Write					

Final state:

	<u>Proc 1</u>	
	Tag	**State**
Set 0		
Set 1		

	<u>Proc 2</u>	
	Tag	**State**
Set 0		
Set 1		

	<u>Proc 3</u>	
	Tag	**State**
Set 0		
Set 1		