Due date: Thursday 4/19 11:59:59pm EDT.

Turning in the lab: To turn in this lab, commit and push your changes you made to your git repository.

Check-off meeting: After turning in this lab, you are required to go to the lab for a check-off meeting within a week of the lab’s due date. See the course website for lab hours.

Set-Associative Caches

You have been promoted to senior-senior engineer at GATE Co. Once the software engineers found out that the elevator’s microprocessors have been improved, they decided to include a bunch of unnecessary programs to the elevator’s computers. To accomodate these programs, they added some external DRAM and a simple direct-mapped cache. As the newest senior-senior engineer at GATE Co., you are in charge of evaluating this direct-mapped cache and implementing a two-way set-associative cache with an LRU replacement policy to better handle both instructions and data.

1 Direct-Mapped Cache

The direct-mapped cache matches the code presented in 6.S084’s lecture and can be found in the Lab 6 git repo.

2 Set-Associative Cache

2.1 Checking for Correct Behavior

Exercise 1 (40 Points): In TwoWayCache.bsv, implement a 2-way set-associative cache with an LRU replacement policy. Check the functionality of this cache with the Beveren test.

Discussion Question 1 (10 Points): Is it possible for a memory request to hit on each way cache of the two-way set-associative cache? Why or why not?

2.2 Checking for Expected Performance

Exercise 2 (20 points): Add a cache hit counter and a cache miss counter to the direct-mapped cache and the two-way set associative cache. Each counter should be a 32-bit register, and each counter should only be incremented in one rule in your cache. For each counter, you should add a method to the Cache interface to return the value for the counter so it can be inspected from outside the module.

Exercise 3 (10 Points): Fill in the test vectors in FixedReqTest.bsv to produce a test that shows you have a 2-way set associative cache and that the replacement policy is LRU. At the completion of the test, you should print the number of cache hits and misses for each cache, and you should be able to deduce the fact that your cache is 2-way set associative using an LRU replacement policy solely from the hit and miss counts.

Discussion Question 2 (5 Points): What part of your test in FixedReqTest.bsv shows that your cache is 2-way set associative?
Discussion Question 3 (5 Points): What part of your test in FixedReqTest.bsv shows that your cache is implementing an LRU replacement policy?

Discussion Question 4 (5 Points): For the 16 MemReq test in FixedReqTest.bsv what is the maximum possible difference between the number of hits for the direct mapped cache and the number of hits for the set-associative cache? What sequence of MemReqs will produce these results?

Discussion Question 5 (5 Points): In a realistic program, what is the maximum difference in hitrates you can expect between a direct mapped cache and your two-way set-associative cache with LRU replacement policy?