Group 3 Project Report

Test Plan

- Major functions
    - For everything, we test them exhaustively for an address size of 16, so we still have a large number of cases, but it is something that is able to be simulated in a reasonable amount of time.
        i. Check that we can parse the instruction into the tag, index and byte select; Address bits = 32; Number of byte select bits = 6; Number of index bits = 15; Number of tag bits = 11; To test we will grab specific bits of the address that we calculated and compare it with the returned values.
        ii. Check that the miss turns into a hit when using the same instruction (with different byte selects)
        iii. Check that the LRU updates correctly; 0 - Move Right, 1 - Move Left;
        We can manually check the values for a smaller number of test cases to make sure out checking algorithm works, and then we can increase the associativity to the actual size
        iv. Check that we can select the correct one to evict;
        0 - Move Left, 1- Move Right;
        We can manually check the values for a smaller number of test cases to make sure out checking algorithm works, and then we can increase the associativity to the actual size
        v. Check that the MESI updates correctly;
        For the MESI protocol, we will have:
        00 = invalid
        01 = exclusive
        10 = shared
        11 = modified
        and we will test all possible bus operations, such as hit with intent to modify, hit, etc.;this will output values like common to alert the other caches to be set to shared.
        vi. Enter differing instructions to test the miss and hits, and make sure it all works together.

Design Specification

- The cache.sv file contains the inputs and outputs of the cache module. In addition, this file takes in the command, tag array, output, bus operation, snoop result, and the communication between L1 and L2.

Source Modules for the Cache

- Below are the .sv files we used for the cache.sv file:
    - MESI_testcase.sv; address_parse.sv; address_parse_tb.sv; block_select.sv; block_select_tb.sv; eviction_lru.sv; eviction_lru_tb.sv; filegenerator.py; hit_miss.sv; hit_miss_tb.sv; parameters.sv; tracefile.txt; tracefileLRU.txt; update_LRU.sv; update_LRU_tb.sv

Associated Modules Used in Validation of the Cache (if any)

- The cachetest1.sv file is the testbench used in validation of the cache.sv file.

Assumptions

- L1 cache and its interface: It is using a physical address instead of a virtual address. If it was a virtual address, then we would have to convert it to a physical address to ask and retrieve from DRAM. L1 cache is a write allocate.
- Processor and its interface: Address size is 32-bit. It is using a physical address instead of a virtual address. If it was a virtual address, then we would have to convert it to a physical address to ask and retrieve from DRAM.
- Memory subsystem and its interface: They have a common line, which will alert other caches to make the MESI shared.

Design Decisions

- All the parameters are used so that if the specifications such as instruction size or associativity change, we can change it easily without modifying all the code.
- We also decided to change from an array for the tag array to a struct as it enables us to access the tag, PLRU bits and MESI bits with more ease, as well as not needing to modify an equation to access everything.