ECE585
Hayden Galante
Alex Maso
Elliot Lamb
Jacob Louie

Team 6: Test Plan

Address Boundary Test

- Index boundaries: We used test vectors with the same indexes but differing 6 LSB to test the boundaries of each index. Every address with the same index but ending in anything from 0x0 to 0x3F, shares a set. When testing we used four addresses with the same index and different byte select bits one set to 0, one set to the max and two we chose in the middle. We completed this with PrRd commands but it could have been done with other commands

- Index boundaries: Made a test case where the max value for index bits is 0x7FFF (15 bits of all 1's) and a minimum value of 0x0. Four more test cases with predetermined addresses with index bits of different values were also tested.

- Tag boundaries: Made a test case (assuming 32 bit addresses) where the max tag value is 0x7FF (11 bits of all 1's) and a minimum value of 0x0. Four more test cases with predetermined addresses with tag bits of different values were also tested.

- Snooped bits boundaries: Using the 2 least significant bits we confirmed that an address ending in 00 produced a HIT, 01 produced a HITM, 10 produced NOHIT and 11 made a NOTHIT.

- Clear Cache: We made a test that filled a few cache lines and ran command 9 to show that it filled correctly. Then run command 8 followed by command 9 to confirm whether the cache was cleared correctly between commands.

PLRU Test

- Fill correct order (simple): Test by using the same index value for each address and use command 9 between every address. This tells us the way that's being filled with each address. Compare this sequence with the expected fill sequence.

- Evict correct order (simple): Fill the set of the PLRU by using the same index for 8 addresses. Then started to evict one at a time with command 9 in between each eviction to confirm the correct eviction sequence compared to expected.

State Machine Testing

- Modified: Write to any address ending in 0(LSB) will put the line into our cache in the Modified state. Bus Read should change the state to Shared and a Bus RWIM should change the state to invalid. Testbench prints tags in state M at index 1 and 2 then one tag in state S at index 1 upon proper implementation.

ECE585
Hayden Galante
Alex Maso
Elliot Lamb
Jacob Louie

- Exclusive: Read to any address ending in 2 will result in the line being written to our cache in the Exclusive state. A read request should not cause the state to change and a write should change the state to Modified. A snooped read request should change the state to Shared. A snooped RWIM should change the state to invalid. Testbench prints four tags at way 0, 2,4 and 6 in state E and then tags at way 0, 2 and 4 in state E, S, and M.

- Shared: Read ending in 00 will put a line into our cache in Shared state. A read request or a snooped bus read to the same address should not result in a state change. A write request from L1 should change the state to Modified and a snooped RWIM should invalidate. Testbench prints 4 tags at way 0, 2, 4 and 6 all in state S followed by 3 tags at way 0, 2 and 4 in states S, M and S.

- Invalid: We initialize all of the ways as invalid. Read request to a new address should set line to the shared. Snoop results returning HIT (address ending in 00) should put the line in the Shared state and snoop returning NOHIT (address ending in 10 or 11) should put it in the Exclusive state . A write request should set the state of the new line to Modified. Testbench prints tags in state S, E and M at index 1, 2 and 3 respectively.

Miscellany

- Verified that all of the “bookkeeping” functions worked; that the Hit Ratio avoided a divide-by-zero condition on an empty cache, reporting was not overly verbose and correctly reported only valid cache lines.
- Generated several very large random and pseudo-random test files to simulate “real world” performance over the entire address range.