Microprocessor System Design (ECE-585)

Fall 2022

Project by:

(Team 14)

Adithya Rajagopal

Jahnvi Mehta

Nishka Sathisha

Swaroop Nellur

Table of Contents

Objective:

Simulation of last level cache (LLC) for a new processor that can be used with up to three other processor in a shared memory configuration.

Design Specification:

The cache has a total capacity of 16MB, uses 64-byte lines, and is 8-way set associative. It employs a write allocate policy and uses the MESI protocol to ensure cache coherence. The replacement policy is implemented with a pseudo-LRU scheme.

Project Specifications:

Code is parameterized but tested and simulated only for given specification.

Global Parameters	Default Values	Description
WAYS	8	8 way associative
WAYS_REP	3	Ways can be represented using these bits
CAPACITY	‘h1000000	16MB total capacity
LINE_SIZE	‘h40	64B line size
INDEX	‘d15	15 bits of index to represent sets
BYTE	‘h6	Byte offset
TAG	‘hB	Number of tag bits
NUM_OF_LINES	‘h40000	Total number of lines
NUM_OF_SETS	‘h80000	Total number of sets

Calculation:

1. INDEX = $\log_2$((CAPACITY/(LINE_SIZE/WAYS)))

   = 16MB/(64/8)

   = 2^{15} Number of indices.

2. BYTE: $\log_2(64)$ = 6 bits to represent the offset.
3. (We are using 32 because of unsigned integer)

   TAG = (32 – INDEX – BYTE) = 12 bits for tag.

4. NUM_OF_LINES = 2^{INDEX} * WAYS

   = 2^{15+3} = 2^{18}

5. NUM_OF_SETS = 2^{INDEX} = 2^{15}

Port description:

InputPorts	Description
clk	clock to the design
rstb	reset as logic low
[31:0] address	memory address
[3:0] n	n is the trace mode
valid	valid pulse

Output Ports	Description
[15:0] hit_cntr	Counter to count the total number of hits
[15:0] miss_cntr	Counter to count the total number of miss
bus_func_out	Bus function
l2tol1msg_out	Messages from l2 to l1
[1:0] C	HIT, HITM or NOHIT
[32767:0] sets	To represent the sets

Block Diagram of DUT

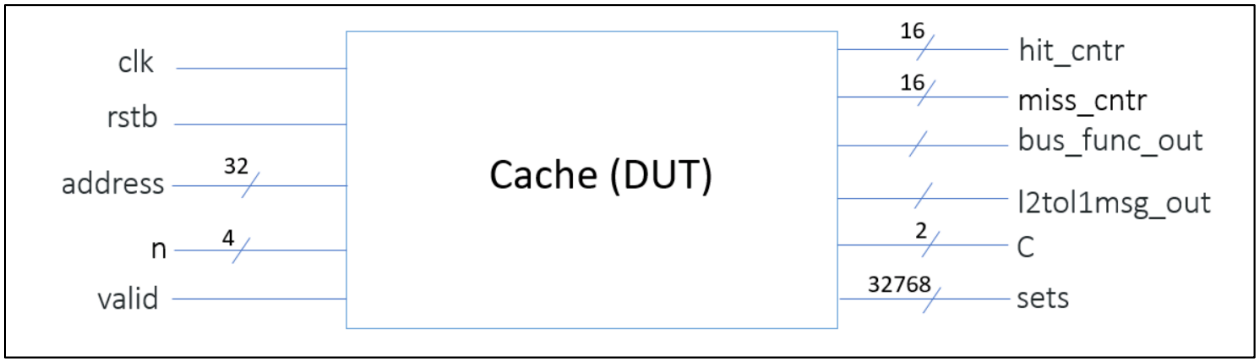

Figure 1.1

Address Decoding:

Since, 32 Bit Address

TAG (11 bits)	**INDEX (15 bits)**	**BYTE OFFSET (6 bits)**

Assumptions made apart from specification:

MESI States:

```
M = 2'b11,              // Modified
E = 2'b10,              // Exclusive
S = 2'b01,              // Shared
I = 2'b00               // Invalidate // reset state hence assigned the value 00
```

Trace file modes:

```
READ_REQ_L1_D     = 4'd0,  // read request from L1 data cache
WRITE_REQ_L1_D    = 4'd1,  // write request from L1 data cache
READ_REQ_L1_I     = 4'd2,  // read request from L1 instruction cache
SNOOP_INVALID_CMD = 4'd3,  // snooped invalidate command
SNOOP_READ_REQ    = 4'd4,  // snooped read request
SNOOP_WRITE_REQ   = 4'd5,  // snooped write request
SNOOP_READ_WITH_M = 4'd6,  // snooped read with intent to modify request
CLR_CACHE_RST     = 4'd8,  // clear the cache and reset all state
PRINT_CONTENTS    = 4'd9   // print contents and state of each valid cache line
                             (doesn't end simulation!)
```

Bus functions:

```
READ       = 3'd1,       /* Bus Read */
INVALIDATE = 3'd2,       /* Bus Invalidate */
RWIM       = 3'd3,       /* Bus Read With Intent to Modify */
NULL       = 3'd4,      /* No Bus output*/
WRITE      = 3'd5       /* Bus Flush (DRAM Write) */
```

Table 1.1

Trace Modes	Work	Description
0	Read request from L1 data cache	Read request from L1 Data cache considered as **PrRd**
1	Write request from L1 data cache	Write request from L1 Data Cache considered as **PrWr**
2	Read request from L1 instruction cache	Read request from L1 instruction cache considered as **PrRd**
3	Snooped invalidate command	Considered as Snoop Bus upgrade **BusUpgr**
4	snooped read request	Considered as **BusRd**
5	Snooped write request	Considered as **BusRdX**
6	Snooped read with intent to modify request	Considered as **BusRdX**

8	Clear the cache and reset all state	Will be driving the reset of cache to 0 (logic low)
9	Print contents and state of each valid cache line	Prints every line which is not in invalid state: Does not display PLRU bits.

Design decisions that has been made:

1. Write from L1 will be considered as processor write.
2. L1 is not part of snoop network, So no MESI in L1.
3. Data is not used in any of the trace file.
4. Byte offset need not be stored in the array as we are always dealing with lines.
5. C in MESI FSM is modelled in RTL and not in Validation under the module "Cache_get_snoop_function".
6. Flush in MESI FSM is considered as Bus Operation Write.
7. When HITM is present, Processor is not modelling the delay where Flush will happen and after which CPU2 will read the bus.
8. Not modelling Outgoing HIT, HITM or NOHIT.
9. Always assumes that L1 has the data to be evicted and L2 will send the EVICTLINE command, regardless.
10. Snoop operation on a line will make the line recently used.
11. Assuming a random clock delay of 100 cycles between successive operations.
12. Snoop on a line is also considered for hit and miss(counters). But for read and write counters snoop is not considered.
13. Upon (n==8) design is clearing the hit and miss counters as well.
14. Design instantiates one MESI FSM logic which will update MESI bits of all lines
    a. In first cycle for PLRU & fetching the MESI bits from the line
    b. In Second cycle MESI FSM logic updates the state.
    c. In third cycle updated MESI states of the line will be returned back to cache.

Defined Modules and Sub Modules:

Table 1.2

Module Name	Description
Cache	top module
Cache_struct	contains the structure of code
Cache_defines	defines local parameter for the code
Cache_opr_ctrl	generates pulse counter for the code to work in serial fashion
Cache_get_PLRU	contains the code for PLRU tree for determining the way
Cache_mesi_fsm	contains the code for mesi fsm for each state
Cache_replacement_algorithm	contains the code for replacement, i.e., when a memory address needs to be directed to a way to occupy. In our case it is PLRU
Cache_update_PLRU	updates the PLRU bits
Cache_read_hit	contains the code for when a hit should happen
Cache_get_snoop_function	Modelling of HIT, HITM & NOHIT
Cache_mesi_fsmtb	testbench for Cache_mesi_fsm.sv
tb_getLRU	testbench for Cache_get_PLRU.sv
tb_updateLRU	testbench for Cache_update_PLRU.sv
Cache_TB	testbench for Cache.sv

CHAPTER 2 - TEST PLAN

Block diagram:

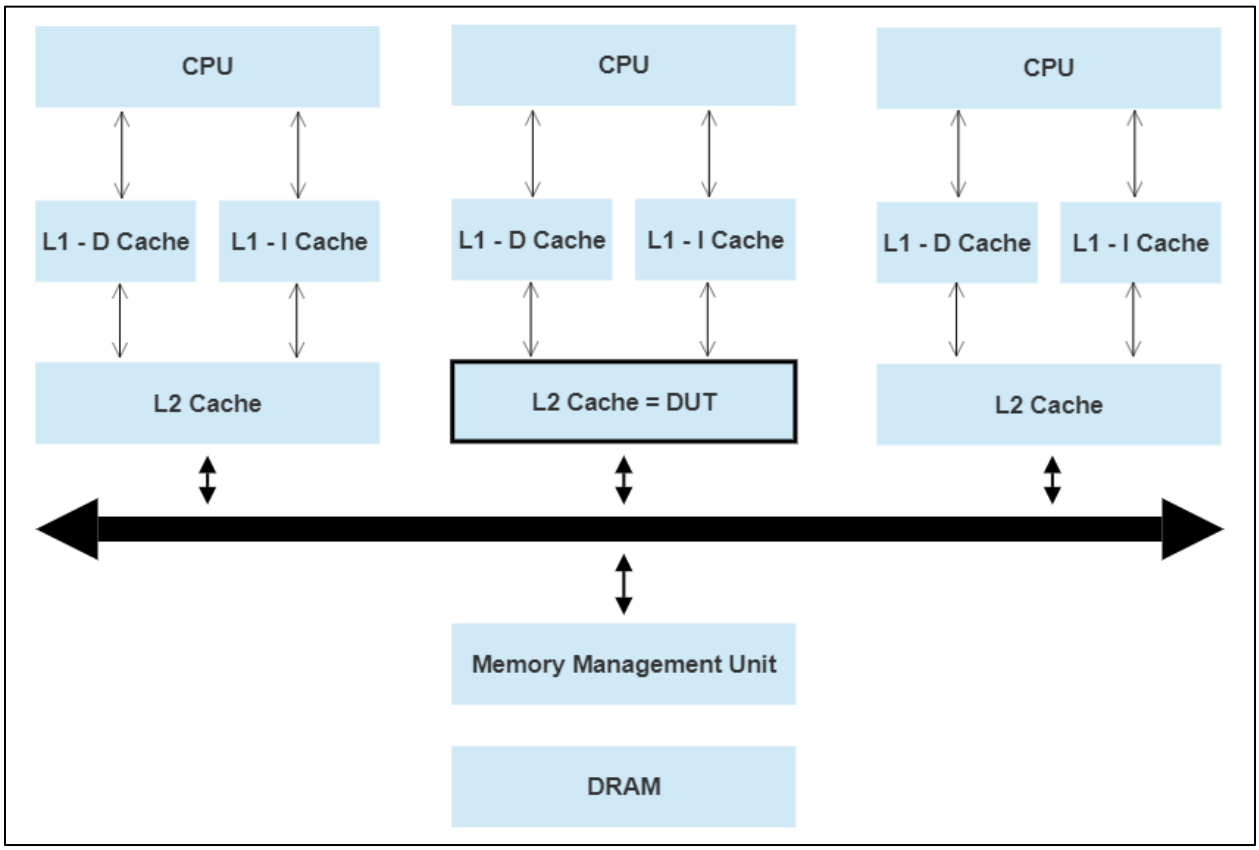

Figure 2.1

Test Plan:

1. **Basic memory test:** This test, tests that cache is storing tag bit or not for the same address.
    a. Write to an address.
    b. Read from the same address.

c. It will be a hit for read since we are reading from the same address it was last written to.
d. Repeat this for multiple iterations.

Hit/miss test: This test will write and read to a random address for multiple iterations.

Output expected:

- First Miss
- Then Hit
- a and b will happen the number of times we write and read.

This test will also do an eviction in back but our test is targeted for testing HIT and Miss so we don't test LRU in this test.

2. PLRU test: This test will stress the cache lines/sets by making sure all the decisive statements are exercised during the read and write process.

   Test 1:

   a. Write to same set with different tag in random order.
   b. At the 9^{th} Write/Read, check if you are getting the Least Recently Used line evicted.
   c. Also check if eviction is occurring in all lines after 9^{th} Write/Read. Use for loop to do it.
   d. Repeat a, b, c for different sets.

   Test 2:

   a. Send writes in order.
   b. Initiate random read in between.
   c. Expect the eviction.
   d. Repeat a, b, c for different sets.

3. MESI FSM test case:

   This will test the following:

   a. Cache is empty, L1 will be requesting data from L2 and if it is a miss in L2 too, a BusRd signal is initiated to the bus. Upon obtaining data from DRAM, L2 will change its state from Invalidate to Exclusive. Upon a PRWr to L1 (Which we won't be Modelling), If L1 is evicting the same

modified line, there will be a PrWr to L2 and L1 should write back to L2 the evicted line and change its state from Exclusive to Modified.

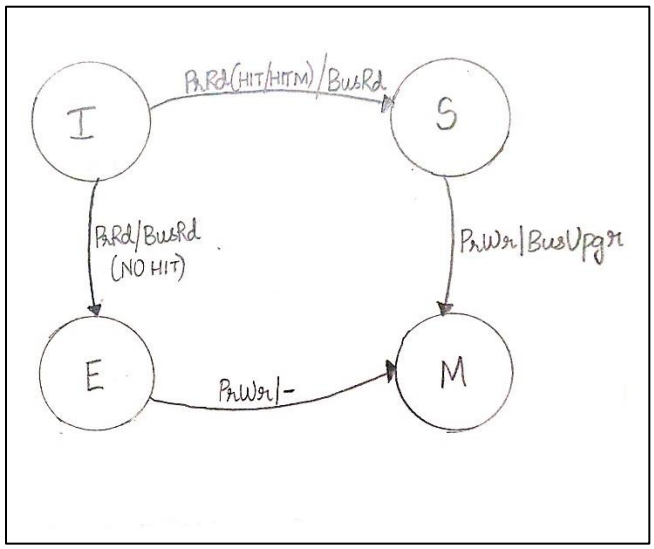

Figure 2.2: MESI Test Case 1

b. If the Cache is Empty, it will be in Invalid state. If a PrWr is initiated, L1 will request L2, L2 will do a DRAM read. After receiving, L2 will give to L1. And then Processor will write to L1. L1 will send to L2 which will in turn change its state from Invalid to Modified.

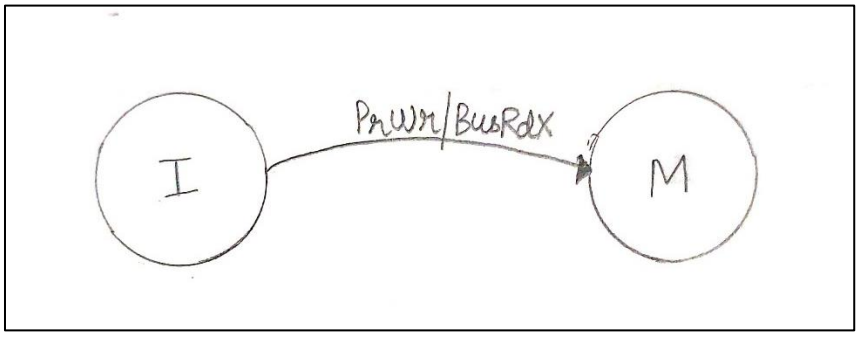

Figure 2.3: Test Case 2

c. When dealing with two processors, if CPU 1 L2 encounters a PrRd, its state changes from Invalid to Exclusive upon receiving the requested data. A snoop request from CPU 2 L2 for same address will result in CPU 1 L2 Cache change its state from Exclusive to Shared, and asserts C. If there is a BusUpgr/BusRdX from CPU 2 L2, CPU 1 L2 will change its state from Shared to Invalid. PrRd to same address to CPU 1 L2, Cache state

changes from Invalid to Shared. If CPU 1 L2 initiates PrWr to the same address, state changes from Shared to Modified, BusUpgr will be asserted high which will make data in L2 of CPU2 as invalid. If CPU 3 L2 initiates a BusRdX, L2 of CPU 1 will go from Modified to Invalid and Flush will be asserted to high therefore DRAM write operation will occur.

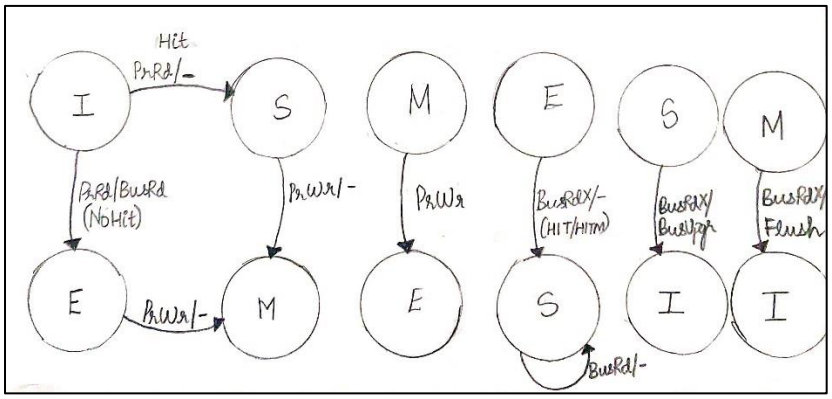

Figure 2.4: Test Case 3

d. When dealing with two processors, if a PrRd is sent to CPU 1 L2, L2 will go from Invalid to Exclusive. CPU 2 L2 will also do a PrRd from the same address as CPU 1 L2. So, Both CPU1 L2 and CPU2 L2 will go to Shared state. Now, there is PrWr in CPU 2 L2, it asserts BusRdX and therefore CPU1 L2 will go to Invalid from Shared and the data in CPU 1 L2 will be stale.

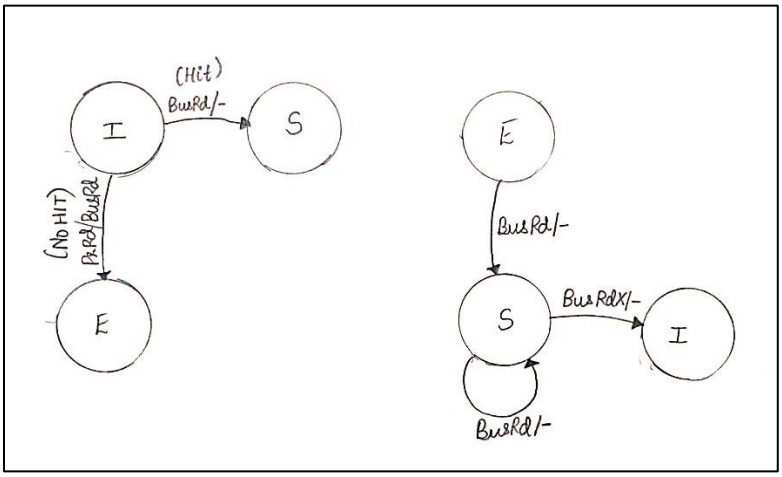

Figure 2.5: Test Case 4

e. When dealing with two processors, a PrWr will be initiated to CPU 1 L2, L2 will go from Invalid to Modified. CPU 2 L2 will do a PrRd from the

same address as CPU 1 L2. So, CPU 1 L2 will go from Modified to Shared after asserting Flush.

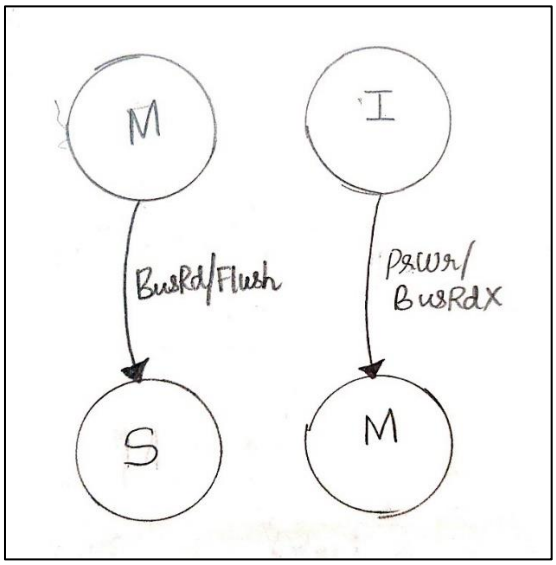

Figure 2.6: Test Case 5

4. Write allocate test/write back:
    a. We write to an empty cache line. Since its empty, it will be a miss. This will lead to a DRAM read request by the Cache.
    b. The cache line is placed to the cache depending on the PLRU bits and the cache will change its MESI state from “Invalidate” to “Modified”. BusRdX should be asserted high upon this transition.
    c. Repeat a,b for different sets

5. Inclusivity test
    a. On a miss to L1 and a hit to L2 /On a miss to L1, hit to L2 but dirty/Modified:
    Read/Write request will be initiated. If it is in L2, we will reply by sending the required Cache line. L2 should be written with the address from where the cache line should be sent to L1. On a miss to L1 and a miss to L2, DRAM request will be initiated from L2. DRAM will reply to L2 with data which is asked for and then L2 will reply to L1 by forwarding the same data that arrives at L2. L1 and L2 should make space for the new incoming data from the DRAM.
    b. On a snopped Upgr hit to L2:

L2 should tell L1 invalidate the cache line and also invalidate its own copy (**Design does not implement a single bit in L2 to indicate "valid" in L1). If L1 does not have the copy which needs to be invalidated, it does not do anything.

c. On a snooped ReadX hit to modified line in L2:
   L2 requests L1 for the same address. If there are any dirty bits set in L1, then L2 will flush the updated data to DRAM and then invalidate both L1 and L2 copy.
d. L1 is having modified data L2 is having non-modified data; Here Assumption is being violated (L1 does not have MESI bits which means if L1 is modified L2 will also be in modified state). So, this cannot be tested.
e. Repeat a,b,c for different sets

6. Clear cache test:
   a. First, make all the Cache states as Invalid
   b. Write to several addresses in the cache
   c. When n is assigned 8, Cache should reset
   d. After c, when we perform a Read/Write to the same addresses, miss counter will be updated to 1 and a DRAM read request will be initiated.

Test Results:

Basic memory test for Hit/Miss:

(Used a task to compare the output results obtained from DUT and manually calculated results)

```
# trace file is opened
# silent Mode is used
# Valid = 1, n = 1, address = 00000000, Hit = 0, Miss = 0
# Valid = 0, n = 1, address = 00000000, Hit = 0, Miss = 0
# Output matches with the expected. H=0, M=1, IDX=0
# Valid = 1, n = 0, address = 00000000, Hit = 0, Miss = 1
# Valid = 0, n = 0, address = 00000000, Hit = 0, Miss = 1
# Output matches with the expected. H=1, M=1, IDX=1
# Valid = 1, n = 1, address = 00000001, Hit = 1, Miss = 1
# Valid = 0, n = 1, address = 00000001, Hit = 1, Miss = 1
# Output matches with the expected. H=2, M=1, IDX=2
# Valid = 1, n = 0, address = 00000001, Hit = 2, Miss = 1
# Valid = 0, n = 0, address = 00000001, Hit = 2, Miss = 1
```

```
# Output matches with the expected. H=3, M=1, IDX=3
# Valid = 1, n = 1, address = 00000002, Hit = 3, Miss = 1
# Valid = 0, n = 1, address = 00000002, Hit = 3, Miss = 1
# Output matches with the expected. H=4, M=1, IDX=4
# Valid = 1, n = 0, address = 00000002, Hit = 4, Miss = 1
# Valid = 0, n = 0, address = 00000002, Hit = 4, Miss = 1
# Output matches with the expected. H=5, M=1, IDX=5
# Valid = 1, n = 1, address = 00000003, Hit = 5, Miss = 1
# Valid = 0, n = 1, address = 00000003, Hit = 5, Miss = 1
# Output matches with the expected. H=6, M=1, IDX=6
# Valid = 1, n = 0, address = 00000003, Hit = 6, Miss = 1
# Valid = 0, n = 0, address = 00000003, Hit = 6, Miss = 1
# Output matches with the expected. H=7, M=1, IDX=7
# Valid = 1, n = 1, address = 00000004, Hit = 7, Miss = 1
# Valid = 0, n = 1, address = 00000004, Hit = 7, Miss = 1
# Output matches with the expected. H=8, M=1, IDX=8
# Valid = 1, n = 0, address = 00000004, Hit = 8, Miss = 1
# Valid = 0, n = 0, address = 00000004, Hit = 8, Miss = 1
# Output matches with the expected. H=9, M=1, IDX=9
# Valid = 1, n = 1, address = 00000005, Hit = 9, Miss = 1
# Valid = 0, n = 1, address = 00000005, Hit = 9, Miss = 1
# Output matches with the expected. H=10, M=1, IDX=10
# Valid = 1, n = 0, address = 00000005, Hit = 10, Miss = 1
# Valid = 0, n = 0, address = 00000005, Hit = 10, Miss = 1
# Output matches with the expected. H=11, M=1, IDX=11
# Valid = 1, n = 1, address = 00000006, Hit = 11, Miss = 1
# Valid = 0, n = 1, address = 00000006, Hit = 11, Miss = 1
# Output matches with the expected. H=12, M=1, IDX=12
# Valid = 1, n = 0, address = 00000006, Hit = 12, Miss = 1
# Valid = 0, n = 0, address = 00000006, Hit = 12, Miss = 1
# Output matches with the expected. H=13, M=1, IDX=13
# Valid = 1, n = 1, address = 00000007, Hit = 13, Miss = 1
# Valid = 0, n = 1, address = 00000007, Hit = 13, Miss = 1
# Output matches with the expected. H=14, M=1, IDX=14
# Valid = 1, n = 0, address = 00000007, Hit = 14, Miss = 1
# Valid = 0, n = 0, address = 00000007, Hit = 14, Miss = 1
# Output matches with the expected. H=15, M=1, IDX=15
# Valid = 1, n = 1, address = 00000008, Hit = 15, Miss = 1
# Valid = 0, n = 1, address = 00000008, Hit = 15, Miss = 1
# Output matches with the expected. H=16, M=1, IDX=16
# Valid = 1, n = 0, address = 00000008, Hit = 16, Miss = 1
# Valid = 0, n = 0, address = 00000008, Hit = 16, Miss = 1
# Output matches with the expected. H=17, M=1, IDX=17
# Valid = 1, n = 1, address = 00000009, Hit = 17, Miss = 1
# Valid = 0, n = 1, address = 00000009, Hit = 17, Miss = 1
# Output matches with the expected. H=18, M=1, IDX=18
# Valid = 1, n = 0, address = 00000009, Hit = 18, Miss = 1
# Valid = 0, n = 0, address = 00000009, Hit = 18, Miss = 1
# Output matches with the expected. H=19, M=1, IDX=19
# Valid = 1, n = 1, address = 0000000a, Hit = 19, Miss = 1
# Valid = 0, n = 1, address = 0000000a, Hit = 19, Miss = 1
# Output matches with the expected. H=20, M=1, IDX=20
# Valid = 1, n = 0, address = 0000000a, Hit = 20, Miss = 1
# Valid = 0, n = 0, address = 0000000a, Hit = 20, Miss = 1
# Output matches with the expected. H=21, M=1, IDX=21
# Valid = 1, n = 1, address = 0000000b, Hit = 21, Miss = 1
# Valid = 0, n = 1, address = 0000000b, Hit = 21, Miss = 1
```

```
# Output matches with the expected. H=22, M=1, IDX=22
# Valid = 1, n = 0, address = 0000000b, Hit = 22, Miss = 1
# Valid = 0, n = 0, address = 0000000b, Hit = 22, Miss = 1
# Output matches with the expected. H=23, M=1, IDX=23
# Valid = 1, n = 1, address = 0000000c, Hit = 23, Miss = 1
# Valid = 0, n = 1, address = 0000000c, Hit = 23, Miss = 1
# Output matches with the expected. H=24, M=1, IDX=24
# Valid = 1, n = 0, address = 0000000c, Hit = 24, Miss = 1
# Valid = 0, n = 0, address = 0000000c, Hit = 24, Miss = 1
# Output matches with the expected. H=25, M=1, IDX=25
# Valid = 1, n = 1, address = 0000000d, Hit = 25, Miss = 1
# Valid = 0, n = 1, address = 0000000d, Hit = 25, Miss = 1
# Output matches with the expected. H=26, M=1, IDX=26
# Valid = 1, n = 0, address = 0000000d, Hit = 26, Miss = 1
# Valid = 0, n = 0, address = 0000000d, Hit = 26, Miss = 1
# Output matches with the expected. H=27, M=1, IDX=27
# Valid = 1, n = 1, address = 0000000e, Hit = 27, Miss = 1
# Valid = 0, n = 1, address = 0000000e, Hit = 27, Miss = 1
# Output matches with the expected. H=28, M=1, IDX=28
# Valid = 1, n = 0, address = 0000000e, Hit = 28, Miss = 1
# Valid = 0, n = 0, address = 0000000e, Hit = 28, Miss = 1
# Output matches with the expected. H=29, M=1, IDX=29
# Valid = 1, n = 1, address = 0000000f, Hit = 29, Miss = 1
# Valid = 0, n = 1, address = 0000000f, Hit = 29, Miss = 1
# Output matches with the expected. H=30, M=1, IDX=30
# Valid = 1, n = 0, address = 0000000f, Hit = 30, Miss = 1
# Valid = 0, n = 0, address = 0000000f, Hit = 30, Miss = 1
# Output matches with the expected. H=31, M=1, IDX=31
# Valid = 1, n = 1, address = 23000001, Hit = 31, Miss = 1
# Valid = 0, n = 1, address = 23000001, Hit = 31, Miss = 1
# Output matches with the expected. H=31, M=2, IDX=32
# Valid = 1, n = 1, address = a5000001, Hit = 31, Miss = 2
# Valid = 0, n = 1, address = a5000001, Hit = 31, Miss = 2
# Output matches with the expected. H=31, M=3, IDX=33
# Valid = 1, n = 1, address = 5a000001, Hit = 31, Miss = 3
# Valid = 0, n = 1, address = 5a000001, Hit = 31, Miss = 3
# Output matches with the expected. H=31, M=4, IDX=34
```

PLRU_test1:

(Used a task to compare the output results obtained from DUT and manually calculated results)

```
# trace file is opened
# silent Mode is used
# Valid = 1, n = 1, address = ce46aa23, Hit = 0, Miss = 0
# Valid = 0, n = 1, address = ce46aa23, Hit = 0, Miss = 0
# Way= 7
# Output matches with the expected.
# Valid = 1, n = 1, address = ab46aa21, Hit = 0, Miss = 1
# Valid = 0, n = 1, address = ab46aa21, Hit = 0, Miss = 1
# Way= 3
# Output matches with the expected.
# Valid = 1, n = 0, address = ce46aa23, Hit = 0, Miss = 2
# Valid = 0, n = 0, address = ce46aa23, Hit = 0, Miss = 2
# Way= 7
```

```
# Output matches with the expected.
# Valid = 1, n = 1, address = 1246aa27, Hit = 1, Miss = 2
# Valid = 0, n = 1, address = 1246aa27, Hit = 1, Miss = 2
# Way= 1
# Output matches with the expected.
# Valid = 1, n = 1, address = 1a46aa29, Hit = 1, Miss = 3
# Valid = 0, n = 1, address = 1a46aa29, Hit = 1, Miss = 3
# Way= 5
# Output matches with the expected.
# Valid = 1, n = 1, address = c246aa22, Hit = 1, Miss = 4
# Valid = 0, n = 1, address = c246aa22, Hit = 1, Miss = 4
# Way= 2
# Output matches with the expected.
# Valid = 1, n = 1, address = ef46aa25, Hit = 1, Miss = 5
# Valid = 0, n = 1, address = ef46aa25, Hit = 1, Miss = 5
# Way= 6
# Output matches with the expected.
# Valid = 1, n = 1, address = 5646aa24, Hit = 1, Miss = 6
# Valid = 0, n = 1, address = 5646aa24, Hit = 1, Miss = 6
# Way= 0
# Output matches with the expected.
# Valid = 1, n = 1, address = 7a46aa23, Hit = 1, Miss = 7
# Valid = 0, n = 1, address = 7a46aa23, Hit = 1, Miss = 7
# Way= 4
# Output matches with the expected.
# Valid = 1, n = 1, address = 9c46aa2e, Hit = 1, Miss = 8
# Valid = 0, n = 1, address = 9c46aa2e, Hit = 1, Miss = 8
# Way= 3
# Output matches with the expected.
# Valid = 1, n = 1, address = 5246aa26, Hit = 1, Miss = 9
# Valid = 0, n = 1, address = 5246aa26, Hit = 1, Miss = 9
# Way= 7
# Output matches with the expected.
# Valid = 1, n = 1, address = 1f46aa2a, Hit = 1, Miss = 10
# Valid = 0, n = 1, address = 1f46aa2a, Hit = 1, Miss = 10
# Way= 1
# Output matches with the expected.
# Valid = 1, n = 1, address = 2346aa2c, Hit = 1, Miss = 11
# Valid = 0, n = 1, address = 2346aa2c, Hit = 1, Miss = 11
# Way= 5
# Output matches with the expected.
# Valid = 1, n = 1, address = 4a46aa2b, Hit = 1, Miss = 12
# Valid = 0, n = 1, address = 4a46aa2b, Hit = 1, Miss = 12
# Way= 2
# Output matches with the expected.
# Valid = 1, n = 1, address = 3746aa24, Hit = 1, Miss = 13
# Valid = 0, n = 1, address = 3746aa24, Hit = 1, Miss = 13
# Way= 6
# Output matches with the expected.
# Valid = 1, n = 1, address = 8b46aa21, Hit = 1, Miss = 14
# Valid = 0, n = 1, address = 8b46aa21, Hit = 1, Miss = 14
# Way= 0
# Output matches with the expected.
# Valid = 1, n = 1, address = 6746aa2f, Hit = 1, Miss = 15
# Valid = 0, n = 1, address = 6746aa2f, Hit = 1, Miss = 15
# Way= 4
# Output matches with the expected.
```

mesi_testcase_2 :

```
# Our snooping bus operation :Address = f4dfb293, Bus operation = READ
# Address = f4dfb293, Message = SENDLINE
# Our snooping bus operation :Address = f4dfb293, Bus operation = NULL
# Address = f4dfb293, Message = NULLMsg
# other processor's snooping bus operation :Address = f4dfb293, Bus operation
= SNOOP_READ_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = f4dfb293, Bus operation = NULL
# Address = f4dfb293, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = f4dfb293, Bus operation
= SNOOP_WRITE_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = ba0d79bb, Bus operation = READ
# Address = ba0d79bb, Message = SENDLINE
# Our snooping bus operation :Address = ba0d79bb, Bus operation = NULL
# Address = ba0d79bb, Message = NULLMsg
# other processor's snooping bus operation :Address = ba0d79bb, Bus operation
= SNOOP_READ_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = ba0d79bb, Bus operation = NULL
# Address = ba0d79bb, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = ba0d79bb, Bus operation
= SNOOP_WRITE_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = e42747ca, Bus operation = READ
# Address = e42747ca, Message = SENDLINE
# Our snooping bus operation :Address = e42747ca, Bus operation = NULL
# Address = e42747ca, Message = NULLMsg
# other processor's snooping bus operation :Address = e42747ca, Bus operation
= SNOOP_READ_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = e42747ca, Bus operation = NULL
# Address = e42747ca, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = e42747ca, Bus operation
= SNOOP_WRITE_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = 6ae81e55, Bus operation = READ
# Address = 6ae81e55, Message = SENDLINE
# Our snooping bus operation :Address = 6ae81e55, Bus operation = NULL
# Address = 6ae81e55, Message = NULLMsg
# other processor's snooping bus operation :Address = 6ae81e55, Bus operation
= SNOOP_READ_REQ, snoopresult=HITM
# Our snooping bus operation :Address = 6ae81e55, Bus operation = NULL
# Address = 6ae81e55, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 6ae81e55, Bus operation
= SNOOP_WRITE_REQ, snoopresult=HITM
# Our snooping bus operation :Address = 8844f600, Bus operation = READ
# Address = 8844f600, Message = SENDLINE
# Our snooping bus operation :Address = 8844f600, Bus operation = NULL
# Address = 8844f600, Message = NULLMsg
# other processor's snooping bus operation :Address = 8844f600, Bus operation
= SNOOP_READ_REQ, snoopresult=HIT
# Our snooping bus operation :Address = 8844f600, Bus operation = NULL
# Address = 8844f600, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 8844f600, Bus operation
= SNOOP_WRITE_REQ, snoopresult=HIT
```

```
# Our snooping bus operation :Address = 18a5aa70, Bus operation = READ
# Address = 18a5aa70, Message = SENDLINE
# Our snooping bus operation :Address = 18a5aa70, Bus operation = NULL
# Address = 18a5aa70, Message = NULLMsg
# other processor's snooping bus operation :Address = 18a5aa70, Bus operation
= SNOOP_READ_REQ, snoopresult=HIT
# Our snooping bus operation :Address = 18a5aa70, Bus operation = NULL
# Address = 18a5aa70, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 18a5aa70, Bus operation
= SNOOP_WRITE_REQ, snoopresult=HIT
# Our snooping bus operation :Address = a4dbe159, Bus operation = READ
# Address = a4dbe159, Message = SENDLINE
# Our snooping bus operation :Address = a4dbe159, Bus operation = NULL
# Address = a4dbe159, Message = NULLMsg
# other processor's snooping bus operation :Address = a4dbe159, Bus operation
= SNOOP_READ_REQ, snoopresult=HITM
# Our snooping bus operation :Address = a4dbe159, Bus operation = NULL
# Address = a4dbe159, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = a4dbe159, Bus operation
= SNOOP_WRITE_REQ, snoopresult=HITM
# Our snooping bus operation :Address = 007aae73, Bus operation = READ
# Address = 007aae73, Message = SENDLINE
# Our snooping bus operation :Address = 007aae73, Bus operation = NULL
# Address = 007aae73, Message = NULLMsg
# other processor's snooping bus operation :Address = 007aae73, Bus operation
= SNOOP_READ_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = 007aae73, Bus operation = NULL
# Address = 007aae73, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 007aae73, Bus operation
= SNOOP_WRITE_REQ, snoopresult=NOHIT
# Number of cache reads =     8
# Number of cache writes =     0
# Number of cache hits =    16
# Number of cache misses =     8
# Cache hit ratio : hit_cntr/(hit_cntr+miss_cntr) =    16/   24
```

write_allocate_test :

```
# Our snooping bus operation :Address = 00000000, Bus operation = RWIM
# Address = 00000000, Message = SENDLINE
# Our snooping bus operation :Address = 00000000, Bus operation = NULL
# Address = 00000000, Message = SENDLINE
# Our snooping bus operation :Address = 00000040, Bus operation = RWIM
# Address = 00000040, Message = SENDLINE
# Our snooping bus operation :Address = 00000040, Bus operation = NULL
# Address = 00000040, Message = SENDLINE
# Our snooping bus operation :Address = 00000080, Bus operation = RWIM
# Address = 00000080, Message = SENDLINE
# Our snooping bus operation :Address = 00000080, Bus operation = NULL
# Address = 00000080, Message = SENDLINE
# Our snooping bus operation :Address = 00000020, Bus operation = NULL
# Address = 00000020, Message = SENDLINE
# Our snooping bus operation :Address = 00000020, Bus operation = NULL
# Address = 00000020, Message = SENDLINE
# Our snooping bus operation :Address = 00000040, Bus operation = NULL
```

```
# Address = 00000040, Message = SENDLINE
# Our snooping bus operation :Address = 00000040, Bus operation = NULL
# Address = 00000040, Message = SENDLINE
# Our snooping bus operation :Address = 00000200, Bus operation = RWIM
# Address = 00000200, Message = SENDLINE
# Our snooping bus operation :Address = 00000200, Bus operation = NULL
# Address = 00000200, Message = SENDLINE
# Our snooping bus operation :Address = 00000400, Bus operation = RWIM
# Address = 00000400, Message = SENDLINE
# Our snooping bus operation :Address = 00000400, Bus operation = NULL
# Address = 00000400, Message = SENDLINE
# Number of cache reads =     7
# Number of cache writes =     7
# Number of cache hits =     9
# Number of cache misses =     5
# Cache hit ratio : hit_cntr/(hit_cntr+miss_cntr) =     9/   14
```

inclusivity_testcase_2 :

```
# Our snooping bus operation :Address = 4008e0c0, Bus operation = READ
# Address = 4008e0c0, Message = SENDLINE
# Our snooping bus operation :Address = 4008e0c0, Bus operation = NULL
# Address = 4008e0c0, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 4008e0c0, Bus operation
= SNOOP_WRITE_REQ, snoopresult=HIT
# Our snooping bus operation :Address = 10010011, Bus operation = READ
# Address = 10010011, Message = SENDLINE
# Our snooping bus operation :Address = 10010011, Bus operation = NULL
# Address = 10010011, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 10010011, Bus operation
= SNOOP_READ_WITH_M, snoopresult=HITM
# Our snooping bus operation :Address = 400e1000, Bus operation = READ
# Address = 400e1000, Message = SENDLINE
# Our snooping bus operation :Address = 400e1000, Bus operation = NULL
# Address = 400e1000, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 400e1000, Bus operation
= SNOOP_INVALID_CMD, snoopresult=HIT
# Our snooping bus operation :Address = 10009e12, Bus operation = READ
# Address = 10009e12, Message = SENDLINE
# Our snooping bus operation :Address = 10009e12, Bus operation = NULL
# Address = 10009e12, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 10009e12, Bus operation
= SNOOP_WRITE_REQ, snoopresult=NOHIT
# Our snooping bus operation :Address = 20037811, Bus operation = READ
# Address = 20037811, Message = SENDLINE
# Our snooping bus operation :Address = 20037811, Bus operation = NULL
# Address = 20037811, Message = INVALIDATELINE
# other processor's snooping bus operation :Address = 20037811, Bus operation
= SNOOP_READ_WITH_M, snoopresult=HITM
# Our snooping bus operation :Address = 000afffa, Bus operation = READ
# Address = 000afffa, Message = SENDLINE
# Our snooping bus operation :Address = 000afffa, Bus operation = READ
# Address = 000afffa, Message = SENDLINE
# other processor's snooping bus operation :Address = 000afffa, Bus operation
= SNOOP_INVALID_CMD, snoopresult=NOHIT
```

```
# Number of cache reads =     6
# Number of cache writes =     0
# Number of cache hits =     6
# Number of cache misses =     6
# Cache hit ratio : hit_cntr/(hit_cntr+miss_cntr) =     6/   12
```

clear_cache_testcase:

```
# Our snooping bus operation :Address = 0101667f, Bus operation = RWIM
# Address = 0101667f, Message = SENDLINE
# Our snooping bus operation :Address = 2fe200a1, Bus operation = RWIM
# Address = 2fe200a1, Message = SENDLINE
# Our snooping bus operation :Address = 4710a001, Bus operation = RWIM
# Address = 4710a001, Message = SENDLINE
# Our snooping bus operation :Address = 4310ce06, Bus operation = RWIM
# Address = 4310ce06, Message = SENDLINE
# Our snooping bus operation :Address = 53a9d101, Bus operation = RWIM
# Address = 53a9d101, Message = SENDLINE
# Our snooping bus operation :Address = 19edc900, Bus operation = RWIM
# Address = 19edc900, Message = SENDLINE
# Clearing all cache
#
# Our snooping bus operation :Address = 0101667f, Bus operation = READ
# Address = 0101667f, Message = SENDLINE
# Printing valid contents
#
# Set=       1433 has valid tag=    8 in way=          7
# Our snooping bus operation :Address = 2fe200a1, Bus operation = READ
# Address = 2fe200a1, Message = SENDLINE
# Our snooping bus operation :Address = 4710a001, Bus operation = READ
# Address = 4710a001, Message = SENDLINE
# Our snooping bus operation :Address = 4310ce06, Bus operation = READ
# Address = 4310ce06, Message = SENDLINE
# Our snooping bus operation :Address = 53a9d101, Bus operation = READ
# Address = 53a9d101, Message = SENDLINE
# Our snooping bus operation :Address = 19edc900, Bus operation = READ
# Address = 19edc900, Message = SENDLINE
# Number of cache reads =     6
# Number of cache writes =     6
# Number of cache hits =     0
# Number of cache misses =     6
# Cache hit ratio : hit_cntr/(hit_cntr+miss_cntr) =     0/    6
```