ECE485

Cash Controller Design Project

Tachchai Buraparatana

Jinho Park

Anthony Romano

Hoa Quach

Project Report

1. Objective

Design a 4-way set associative L2 Cache for a 32-bit processor and implement the design using verilog HDL.

Simulate random, LRU and Pseudo-LRU replacement policies and measure the performance of each policies

1. Requirement Specification

|  |  |
| --- | --- |
|  | Requirements |
| Cache | Line Size MUST BE greater than 4 bytes (32 bits) |
| MUST implement write-back and write allocate policies |
| MUST implement all of the following replacement policies   1. Random 2. LRU 3. Pseudo-LRU |
| Testbench | Must read memory access from text file |
| Must Keep track of following key parameters   1. Total number of memory references 2. Number of reads 3. Number of writes 4. Number of cache hits 5. Number of cache misses 6. Hit ratio |
| Should emulate burst mode in memory |

1. Assumptions Made
   1. All memory references are world-aligned and the processor addresses items by word.
   2. Full range of 32 bit address space is accessed.
   3. Write-through cache. All the memory access from L1 goes through L2.
   4. Burst mode is ensured to be completed without any interrupt.
2. Block Diagram



1. Module Description

|  |  |
| --- | --- |
| Module Name | L1Cache |
| Input Ports | stall, debug |
| Output Ports | addr[31:0], addrstb, we |
| Bidirectional Ports | data[31:0] |
| Description | Emulates the L1 cache side interface by reading in memory references from the trace file and make read/write request to L2 cache. |
| Interface mechanism with L2Cache | This module outputs 32 bit memory address to addr[31:0] and assert (we=0) or de-assert (we=1) the active low we signal according to the type of the memory reference. When valid address, we, and data (in case of memory write request) are available on the bus, it toggles addrstb, which triggers L2 cache to latch in the address, we and data.  L2 cache assert active high stall (stall = 1) when L2 is processing request from L1 cache. L1 cache stalls until stall is de-asserted by L2. |

|  |  |
| --- | --- |
| Module Name | L2Cache |
| Input Ports | Stb  we\_L1, addrstb\_L1, addr\_L1 |
| Output Ports | stall we\_MEM, addr\_MEM, addrstb\_MEM |
| Bidirectional Ports | data\_L1[31:0], data\_MEM[63:0] |
| Description | Processes cache request from L1Cache and manage L2 cache.  Employs one of the replacement policies.  Keep track of test statistics. / Interface the Main memory |

|  |  |
| --- | --- |
| Module Name | MainMemory |
| Input Ports | we, addrstb, addr[31:0] |
| Output Ports | stb |
| Bidirectional Ports | data[63:0] |
| Description | Emulates the main memory side interface by processing memory access from L2 cache.  Supports burst read. |
| Interface mechanism with L2Cache | When addrstb is toggled, this module latches the address and we values from addr[31:0] and we. L2 stalls until stb is toggled. MainMemory toggles stb as it bursts out each chunk of data to data[63:0] bus or toggles stb when process is done. |

|  |  |
| --- | --- |
| Module Name | L2CacheTestBench |
| Input Ports | N/A |
| Output Ports | N/A |
| Description | Instantiate L1Cache, L2Cache, and MainMemory modules.  Provide interface to debug and rep port of L2 cache. |
| Interface mechanism with L2Cache | Assert debug to display debug flags in the codes.  Assign appropriate values of replacement policy to rep[1:0] to enable a replacement policy. |

1. L2 cache Design Decisions



* 1. Cache line size is 64 bytes because the graph above from class lecture slides show that the cache line size of 64 byte gives the best performance and current processors use cache size of 32 – 128 bytes.
  2. Main memory’s burst length is eight to support cache line fill of 64-byte-long line over 64 bit data bus.
  3. The number of lines is 64 to ensure number of replacement reasonably high. When the number of liens are higher than 64, replacement does not happen often enough to yield data for comparison. When the numbers are too small, cache miss rate becomes too high. (provides opportunity to test replacement policies).
  4. Cache Size

Following is the size of cache used for the simulations. These sizes include valid, dirty and tag bits as well as data bits.

* + 1. Random Replacement
    2. Pseudo-LRU
    3. LRU

L2 Cache Module Algorithm



1. Testing
   1. Cache Initiation
      1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 2 | 2 | 4 | 2 | N/A |

* + 1. Description:

Initiate cache\_data array so that the 4 **MSB’s** of each word represents the **index** and the **4 LSB’s** represent the **word number** in the line.

* + 1. Result:

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: 00000000

SIM: Word: 1: Content: 00000001

SIM: Word: 2: Content: 00000002

SIM: Word: 3: Content: 00000003

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: 00000000

SIM: Word: 1: Content: 00000001

SIM: Word: 2: Content: 00000002

SIM: Word: 3: Content: 00000003

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

* 1. Cache Line Fill
     1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 2 | 2 | 4 | 2 | N/A |

* + 1. Description

Fill cache **line 0** by causing a cache miss on line 0 with address 0xFFFC0000.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word  [5:4] | | | | | 00 | | |
| 16383 | | | | | | | | | | | | | | 0 | | | | | | | | | | | | 0 | | | | | 0 | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | |

Dram output increment the address by 4(0x00000004) after each burst, so that each item stored represent the starting address of the 32 bit word for debugging purpose.

* + 1. Expected Result of cache line fill:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Way** | **Index** | **Word** | **Content** | **Note** |
| 0 | 0 | 0 | FFFF0000 | Newly filled line after cache miss |
|  |  | 1 | FFFF0004 |
|  |  | 2 | FFFF0008 |
|  |  | 3 | FFFF000C |
|  | 1 | 0 | 10000000 |  |
|  |  | 1 | 10000001 |
|  |  | 2 | 10000002 |
|  |  | 3 | 10000003 |

* + 1. Test Vectors

2 fffc0000

* + 1. Result:

sim> run

SIM: =================================================

SIM: command:2 address:fffc0000

SIM: Tag: 1048512 Index: 0 Word 0

SIM: L2 MISS

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 1048512, Index: 0

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: 00000000

SIM: Word: 1: Content: 00000001

SIM: Word: 2: Content: 00000002

SIM: Word: 3: Content: 00000003

SIM: Index: 1

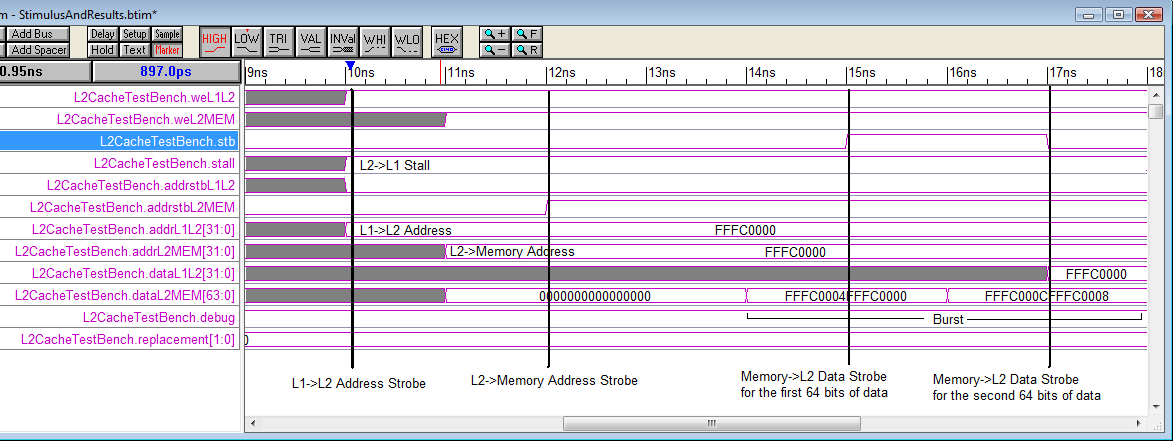
SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

* + 1. Timing Diagram for burst of 2



* 1. Cache Look Up (Hit)
     1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 2 | 2 | 4 | 2 | N/A |

* + 1. Setup:

Run test b. Cache Line Fill

* + 1. Description

After fill the cache line as in b. with address FFFC0000 (Tag = 16383, Index = 0, Word = 0). Access FFFC0004, FFFC0008, FFFC000C (Tag = 16383, Index =1, Word = 1~3) and see if each addresses that match the tag and index gives cache hit.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Way** | **Index** | **Word** | **Content** | **Note** |
| 0 | 0 | 0 | FFFF0000 |  |
|  |  | 1 | FFFF0004 | Words to be accessed |
|  |  | 2 | FFFF0008 |
|  |  | 3 | FFFF000C |
|  | 1 | 0 | 10000000 |  |
|  |  | 1 | 10000001 |
|  |  | 2 | 10000002 |
|  |  | 3 | 10000003 |

* + 1. Expected Result

Three consecutive hits on way 0 index 0

* + 1. Test Vectors

2 fffc0000

2 fffc0004

2 fffc0008

2 fffc000c

* + 1. Result

SIM: =================================================

SIM: command:2 address:fffc0004

SIM: Tag: 16383 Index: 0 **Word 1**

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

**SIM: L2 HIT**

SIM: =================================================

SIM: command:2 address:fffc0008

SIM: Tag: 16383 Index: 0 **Word 2**

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

**SIM: L2 HIT**

SIM: =================================================

SIM: command:2 address:fffc000c

SIM: Tag: 16383 Index: 0 **Word 3**

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

**SIM: L2 HIT**

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: 00000000

SIM: Word: 1: Content: 00000001

SIM: Word: 2: Content: 00000002

SIM: Word: 3: Content: 00000003

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

* 1. Cache Look Up (Miss)
     1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 2 | 2 | 4 | 2 | N/A |

* + 1. Description:

After fill the cache line as in b. with address FFFC0000 (Tag = 16383, **Index = 0**, Word = 0). Access FFFC0040 (Tag = 16383, **Index =1,** Word = 0) and see if it gives miss and result in cache line fill in line 1.

* + 1. Expected Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Way** | **Index** | **Word** | **Content** | **Note** |
| 0 | 0 | 0 | FFFF0000 |  |
|  |  | 1 | FFFF0004 |
|  |  | 2 | FFFF0008 |
|  |  | 3 | FFFF000C |
|  | 1 | 0 | FFFC0040 | Newly filled line after cache miss |
|  |  | 1 | FFFC0044 |
|  |  | 2 | FFFC0048 |
|  |  | 3 | FFFC004C |

* + 1. Test Vectors:

2 fffc0000

2 fffc0040

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word  [5:4] | | | | | 00 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 4 | | | | | 0 | | | | | |
| 16383 | | | | | | | | | | | | | | **1** | | | | | | | | | | | | | 0 | | | | | | 0 | |

* + 1. Result:

SIM: =================================================

SIM: command:2 address:fffc0040

SIM: Tag: 16383 Index: 1 Word 0

SIM: **L2 MISS**

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 16383, Index: **1**

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: Index: 1

SIM: Word: 0: Content: fffc0040

SIM: Word: 1: Content: fffc0044

SIM: Word: 2: Content: fffc0048

SIM: Word: 3: Content: fffc004c

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: 00000000

SIM: Word: 1: Content: 00000001

SIM: Word: 2: Content: 00000002

SIM: Word: 3: Content: 00000003

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

* 1. Replacement - Random
     1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 2 | 2 | 4 | 2 | **Random** |

* + 1. Description

Fill Index 0 of way 0 and way 1 and cause read miss on the same line.

Check if replacement occurs on either one of the way.

* + 1. Expected Result:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Way** | **Index** | **Word** | **Content** | **Note** |
| 0 | 0 | 0 | FFFC0000 | Index 0 of line 0 or line 1 should be replaced with FFF40000~FFF4000C |
|  |  | 1 | FFFC0004 |
|  |  | 2 | FFFC0008 |
|  |  | 3 | FFFC000C |
|  | 1 | 0 | 10000000 |  |
|  |  | 1 | 10000001 |
|  |  | 2 | 10000002 |
|  |  | 3 | 10000003 |
| 1 | 0 | 0 | FFF80000 |  |
|  |  | 1 | FFF80004 |
|  |  | 2 | FFF80008 |
|  |  | 3 | FFF8000C |
|  | 1 | 0 | 10000000 |  |
|  |  | 1 | 10000001 |
|  |  | 2 | 10000002 |
|  |  | 3 | 10000003 |

* + 1. Test Vectors:

FFFC0000 (Tag = 16383, Index = 0, Word = 0)

FFF80000 (Tag = 16382, Index = 0, Word = 0)

FFF40000 (Tag = 16381, Index = 0, Word = 0)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word  [5:4] | | | | | 00 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| **16383** | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 8 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| **16382** | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 4 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| **16381** | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |

* + 1. Result:

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fff**c**0000

SIM: Word: 1: Content: fff**c**0004

SIM: Word: 2: Content: fff**c**0008

SIM: Word: 3: Content: fff**c**000c

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: fff80000

SIM: Word: 1: Content: fff80004

SIM: Word: 2: Content: fff80008

SIM: Word: 3: Content: fff8000c

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

SIM: =================================================

SIM: command:2 address:fff40000

SIM: Tag: 16381 Index: 0 Word 0

**SIM: L2 MISS**

**SIM: Task: Replacement\_Way\_Lookup\_Random**

**SIM: Replace Way: 0**

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 16381, Index: 0

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

**SIM: Word: 0: Content: fff40000**

**SIM: Word: 1: Content: fff40004**

**SIM: Word: 2: Content: fff40008**

**SIM: Word: 3: Content: fff4000c**

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: fff80000

SIM: Word: 1: Content: fff80004

SIM: Word: 2: Content: fff80008

SIM: Word: 3: Content: fff8000c

SIM: Index: 1

SIM: Word: 0: Content: 10000000

SIM: Word: 1: Content: 10000001

SIM: Word: 2: Content: 10000002

SIM: Word: 3: Content: 10000003

* 1. PLRU
     1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 4 | **1** | 4 | 2 | **PLRU** |

* + 1. Description
       1. fill Index 0 of way 0 - way 3 (PLRU = 000 way 0)
       2. access index 0 of way 0 (PLRU = 110 way 2)
       3. access index 0 of way 3 (PLRU = 010 way 1)
       4. cause read miss on index 0 and see if **way 1** is replaced.
    2. Test Vectors

2 fffc0000

2 fff80000

2 fff40000

2 fff00000

2 fffc0000

2 fff00000

2 ffec0000

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word  [5:4] | | | | | 00 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16383 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 8 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16382 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 4 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16381 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16380 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | E | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16379 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |

* + 1. Expected Result

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Way** | **Index** | **Word** | **Content** | **Note** | **After Repacement** |
| 0 | 0 | 0 | FFFC0000 |  | FFFC0000 |
|  |  | 1 | FFFC0004 | FFFC0004 |
|  |  | 2 | FFFC0008 | FFFC0008 |
|  |  | 3 | FFFC000C | FFFC000C |
| 1 | 0 | 0 | FFF80000 | PLRU = **010** when replacement occurs, so line 1 should be evicted | **FFEC0000** |
|  |  | 1 | FFF80004 | **FFEC0004** |
|  |  | 2 | FFF80008 | **FFEC0008** |
|  |  | 3 | FFF8000C | **FFEC000C** |
| 2 | 0 | 0 | FFF40000 |  | FFF40000 |
|  |  | 1 | FFF40004 | FFF40004 |
|  |  | 2 | FFF40008 | FFF40008 |
|  |  | 3 | FFF4000C | FFF4000C |
| 3 | 0 | 0 | FFF00000 |  | FFF00000 |
|  |  | 1 | FFF00004 | FFF00004 |
|  |  | 2 | FFF00008 | FFF00008 |
|  |  | 3 | FFF0000C | FFF0000C |

* + 1. Result:

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: fff80000

SIM: Word: 1: Content: fff80004

SIM: Word: 2: Content: fff80008

SIM: Word: 3: Content: fff8000c

SIM: ----------------------------------------------

SIM: Way: 2

SIM: Index: 0

SIM: Word: 0: Content: fff40000

SIM: Word: 1: Content: fff40004

SIM: Word: 2: Content: fff40008

SIM: Word: 3: Content: fff4000c

SIM: ----------------------------------------------

SIM: Way: 3

SIM: Index: 0

SIM: Word: 0: Content: fff00000

SIM: Word: 1: Content: fff00004

SIM: Word: 2: Content: fff00008

SIM: Word: 3: Content: fff0000c

SIM: Data from L2: fff00000

SIM: =================================================

SIM: command:2 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

**SIM: Matching Way: 0**

SIM: L2 HIT

**SIM: cache\_plru[line=0]: 110**

SIM: =================================================

SIM: command:2 address:fff00000

SIM: Tag: 16380 Index: 0 Word 0

SIM: Task: Look\_For\_Match

**SIM: Matching Way: 3**

SIM: L2 HIT

**SIM: cache\_plru[line=0]: 010**

SIM: =================================================

SIM: command:2 address:ffec0000

SIM: Tag: 16379 Index: 0 Word 0

SIM: L2 MISS

**SIM: Task: Replacement\_Way\_Lookup\_PLRU**

**SIM: Replace Way: 1**

SIM: Task : Cache\_Line\_Fill

SIM: Way: 1, Tag: 16379, Index: 0

SIM: cache\_plru[line=0]: 100

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: ffec0000

SIM: Word: 1: Content: ffec0004

SIM: Word: 2: Content: ffec0008

SIM: Word: 3: Content: ffec000c

SIM: ----------------------------------------------

SIM: Way: 2

SIM: Index: 0

SIM: Word: 0: Content: fff40000

SIM: Word: 1: Content: fff40004

SIM: Word: 2: Content: fff40008

SIM: Word: 3: Content: fff4000c

SIM: ----------------------------------------------

SIM: Way: 3

SIM: Index: 0

SIM: Word: 0: Content: fff00000

SIM: Word: 1: Content: fff00004

SIM: Word: 2: Content: fff00008

SIM: Word: 3: Content: fff0000c

* 1. LRU
     1. Cache Parameters:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **# of ways** | **# of lines** | **# of words** | **Burst length** | **Replacement** |
| 4 | **1** | 4 | 2 | **PLRU** |

* + 1. Description
       1. After fill Index 0 of way 0 - way 3 (LRU bits: way 3> way 2> way 1>way0(=LRU))
       2. access index 0 of way 0 (LRU bits: way0>way3>way2>way1(=LRU))
       3. access index 0 of way 1 (LRU bits: way1>way0>way3>way2(=LRU))
       4. cause read miss on index 0 and see if **way 2** is replaced.
    2. Test Vectors

2 fffc0000

2 fff80000

2 fff40000

2 fff00000

2 fffc0000

2 fff80000

2 ffec0000

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word  [5:4] | | | | | 00 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16383 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 8 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16382 | | | | | | | | | | | | | | **1** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 4 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16381 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16380 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | E | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16379 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |

* + 1. Expected Result

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Way** | **Index** | **Word** | **Content** | **Note** | **After Repacement** |
| 0 | 0 | 0 | FFFC0000 | LRU | FFFC0000 |
|  |  | 1 | FFFC0004 | FFFC0004 |
|  |  | 2 | FFFC0008 | FFFC0008 |
|  |  | 3 | FFFC000C | FFFC000C |
| 1 | 0 | 0 | FFF80000 |  | **FFEC0000** |
|  |  | 1 | FFF80004 | **FFEC0004** |
|  |  | 2 | FFF80008 | **FFEC0008** |
|  |  | 3 | FFF8000C | **FFEC000C** |
| 2 | 0 | 0 | FFF40000 | LRU = 0 for way2 when replacement occurs, so line 2 should be evicted | FFF40000 |
|  |  | 1 | FFF40004 | FFF40004 |
|  |  | 2 | FFF40008 | FFF40008 |
|  |  | 3 | FFF4000C | FFF4000C |
| 3 | 0 | 0 | FFF00000 |  | FFF00000 |
|  |  | 1 | FFF00004 | FFF00004 |
|  |  | 2 | FFF00008 | FFF00008 |
|  |  | 3 | FFF0000C | FFF0000C |

* + 1. RESULT:**PASS**

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: fff80000

SIM: Word: 1: Content: fff80004

SIM: Word: 2: Content: fff80008

SIM: Word: 3: Content: fff8000c

SIM: ----------------------------------------------

SIM: Way: 2

SIM: Index: 0

SIM: Word: 0: Content: fff40000

SIM: Word: 1: Content: fff40004

SIM: Word: 2: Content: fff40008

SIM: Word: 3: Content: fff4000c

SIM: ----------------------------------------------

SIM: Way: 3

SIM: Index: 0

SIM: Word: 0: Content: fff00000

SIM: Word: 1: Content: fff00004

SIM: Word: 2: Content: fff00008

SIM: Word: 3: Content: fff0000c

SIM: =================================================

SIM: command:2 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

**SIM: Matching Way: 0**

SIM: L2 HIT

**SIM: cache\_lru[way=0][line=0]: 3**

SIM: cache\_lru[way=1][line=0]: 0

SIM: cache\_lru[way=2][line=0]: 1

SIM: cache\_lru[way=3][line=0]: 2

SIM: =================================================

SIM: command:2 address:fff80000

SIM: Tag: 16382 Index: 0 Word 0

SIM: Task: Look\_For\_Match

**SIM: Matching Way: 1**

SIM: L2 HIT

SIM: cache\_lru[way=0][line=0]: 2

**SIM: cache\_lru[way=1][line=0]: 3**

**SIM: cache\_lru[way=2][line=0]: 0**

SIM: cache\_lru[way=3][line=0]: 1

SIM: =================================================

SIM: command:2 address:ffec0000

SIM: Tag: 16379 Index: 0 Word 0

SIM: L2 MISS

**SIM: Task: Replacement\_Way\_Lookup\_LRU**

**SIM: Replace Way: 2**

SIM: Task : Cache\_Line\_Fill

SIM: Way: 2, Tag: 16379, Index: 0

SIM: cache\_lru[way=0][line=0]: 1

SIM: cache\_lru[way=1][line=0]: 2

SIM: cache\_lru[way=2][line=0]: 3

SIM: cache\_lru[way=3][line=0]: 0

SIM: ----------------------------------------------

SIM: Way: 0

SIM: Index: 0

SIM: Word: 0: Content: fffc0000

SIM: Word: 1: Content: fffc0004

SIM: Word: 2: Content: fffc0008

SIM: Word: 3: Content: fffc000c

SIM: ----------------------------------------------

SIM: Way: 1

SIM: Index: 0

SIM: Word: 0: Content: fff80000

SIM: Word: 1: Content: fff80004

SIM: Word: 2: Content: fff80008

SIM: Word: 3: Content: fff8000c

SIM: ----------------------------------------------

SIM: Way: 2

SIM: Index: 0

SIM: Word: 0: Content: ffec0000

SIM: Word: 1: Content: ffec0004

SIM: Word: 2: Content: ffec0008

SIM: Word: 3: Content: ffec000c

SIM: ----------------------------------------------

SIM: Way: 3

SIM: Index: 0

SIM: Word: 0: Content: fff00000

SIM: Word: 1: Content: fff00004

SIM: Word: 2: Content: fff00008

SIM: Word: 3: Content: fff0000c

1. Result & Conclusion

Following are statistics from the simulations of three different replacement policies using the trace file cc1.din.

* 1. Random

SIM: +++++++STATISTIC+++++++

SIM: L2 Read:......916972

SIM: L2 Write:.....83030

SIM: Replacement:..RANDOM

SIM: Hit:..........972163

SIM: Miss:.........27839

SIM: Hit Ratio:....97.2161%

SIM: +++++++++++++++++++++++

* 1. Pseudo-LRU

SIM: +++++++STATISTIC+++++++

SIM: L2 Read:......916972

SIM: L2 Write:.....83030

SIM: Replacement:..PLRU

SIM: Hit:..........975188

SIM: Miss:.........24814

SIM: Hit Ratio:....97.5186%

SIM: +++++++++++++++++++++++

* 1. LRU

SIM: +++++++STATISTIC+++++++

SIM: L2 Read:......916972

SIM: L2 Write:.....83030

SIM: Replacement:..LRU

SIM: Hit:..........975828

SIM: Miss:.........24174

SIM: Hit Ratio:....97.5826%

SIM: +++++++++++++++++++++++

Above result shows that a cache with relatively small size of 17Kbyte, 64 lines of 64-byte lines 4-way set associative cache gives fairly high hit ratio of 97%. Assuming that the provided trace file resembles the memory access pattern in computing in general, it can be concluded that an L2 cache increases performance

The result of comparison among three different replacement policies is as expected. LRU give the greatest hit rate and random gives the lowest. It also shows that, due to the relatively low miss rate, replacement policy does not play significant role in determining the cache hit rate because the low miss rate implies that replacement occurs rarely. It becomes more apparent when the line size is greater than 64. For example, when the line size is 1024 lines, simulation results in only compulsory misses and there is no difference among random, pseudo-LRU and LRU algorithms. The result also suggests to choose Pseudo-LRU over LRU algorithm because it uses less memory spaces for the replacement information, 37% in the example, and simpler algorithm at the cost of 0.07% lower hit rate, which is a reasonable trade-off.

1. Extra Credit A - MESI Protocol Support
   1. Assumptions
      1. HIT/HITM never asserted
   2. Normal behavior
      1. Read Miss -> Cache Line Fill -> Exclusive
      2. Read Hit: -> No state transition
      3. Write Miss/Write Hit: transition to Modified (**assert Invalidate All**)
   3. Snooped Read
      1. Invalid : Do Nothing
      2. Shared: Do Nothing
      3. Modified: stall -> write back -> continue -> shared
      4. Exclusive: -> Shared
   4. Snooped Write
      1. Invalid: Do Nothing
      2. Shared: -> Invalid
      3. Modified: stall -> write back -> continue -> Invalid
      4. Exclusive: -> Invalid
   5. Bit Assignment
      1. 00 – Invalid
      2. 01 – Exclusive
      3. 10 – Shared
      4. 11 – Modified
   6. Added Ports
      1. Output: inv (invalidate all) – 1 bit signal to invalidate other processor’s lines when going to modified state
      2. Input: snoop, (snoop result) – 2 bit signal indicates what is snooped
         1. 0X – No Snoop
         2. 10 – Snoop Read
         3. 11 – Snoop Write
   7. State Transition Test
      1. From Invalid to Exclusive
         1. Fill Index 0 of way 0 (FFFC0000) (Tag = 16383, Index = 0, Word = 0)
         2. Verify State = Exclusive
         3. Result

SIM: command:2 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

**SIM: Task : Cache\_Line\_Fill**

SIM: Way: 0, Tag: 16383, Line: 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

SIM: Data from L2: fffc0000

* + 1. From Exclusive to Shared
       1. Snoop read on FFFC0000
       2. Verify State = Shared
       3. Test Vectors

2 fffc0000

3 fffc0000

* + - 1. Result

===========================================================

SIM: command:3 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

**SIM: Task: Look\_For\_Match**

**SIM: Matching Way: 0**

**SIM: Snooped Read**

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 2 = SHARED**

* + 1. From Exclusive to Invalid
       1. Snoop write/invalidate on FFFC0000
       2. Verify State = Modified
       3. Test Vectors

2 fffc0000

4 fffc0000

* + - 1. Result

===========================================================

SIM: command:4 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

**SIM: Matching Way: 0**

**SIM: Snooped Write**

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 0 = INVALID**

SIM: ===========================================================

* + 1. From Exclusive to Modified
       1. Cache Write on FFFC0000
       2. Verify State = Invalid
       3. Test Vectors

2 fffc0000

1 fffc0000

* + - 1. Result

SIM: ===========================================================

SIM: command:1 address:fffc0000

SIM: Data from L1: 0000000a

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: L2 HIT

SIM: L1 Write

SIM: Data from L1: 0000000a

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 3 = MODIFIED**

SIM: ===========================================================

* + 1. From Modified to Shared
       1. Write Back!
       2. Snoop read on FFFC0000
       3. Verify State = Shared
       4. Test Vectors

1 fffc0000

3 fffc0000

* + - 1. Results

SIM: command:1 address:fffc0000

SIM: Data from L1: 0000000a

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

**SIM: Task : Cache\_Line\_Fill**

SIM: Way: 0, Tag: 16383, Line: 0

SIM: L1 Write

SIM: Data from L1: 0000000a

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 3 = MODIFIED**

SIM: ===========================================================

SIM: command:3 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: Snooped Read

**SIM: Write Back!**

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 2 = SHARED**

SIM: ===========================================================

* + 1. From Modified to Invalid
       1. Write Back!
       2. Snoop write on FFFC0000
       3. Verify State = Invalid
       4. Test Vectors

1 fffc0000

4 fffc0000

* + - 1. Results

SIM: command:1 address:fffc0000

SIM: Data from L1: 0000000a

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

**SIM: Task : Cache\_Line\_Fill**

SIM: Way: 0, Tag: 16383, Line: 0

SIM: L1 Write

SIM: Data from L1: 0000000a

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 3 = MODIFIED**

SIM: ===========================================================

SIM: command:4 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: Snooped Write

**SIM: Write Back!**

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 0 = INVALID**

SIM: ===========================================================

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word  [5:4] | | | | | 00 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16383 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |

* + 1. Snooped read on non-matching line
       1. Expected: **no state transition**
       2. Test vectors

0 fffc0000

3 fff80000

* + - 1. Results

SIM: command:0 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 16383, Line: 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

SIM: Data from L2: fffc0000

SIM: ===========================================================

SIM: command:3 address:fff80000

SIM: Tag: 16382 Index: 0 Word 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

SIM: ===========================================================

* + 1. Snooped write on non-matching line
       1. Expected: **no state transition**
       2. Test vectors

0 fffc0000

4 fff80000

* + - 1. Results

SIM: command:0 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 16383, Line: 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

SIM: Data from L2: fffc0000

SIM: ===========================================================

SIM: command:4 address:fff80000

SIM: Tag: 16382 Index: 0 Word 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

SIM: ===========================================================

* + 1. Mixed
       1. Read Miss → Snooped Read → Write Hit → Snooped Write
       2. Expected: **Exclusive→ Shared→ Modified → Invalid**
       3. Result:

SIM: command:0 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 16383, Line: 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

SIM: Data from L2: fffc0000

SIM: ===========================================================

SIM: command:3 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: Snooped Read

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 2 = SHARED**

SIM: ===========================================================

SIM: command:1 address:fffc0000

SIM: Data from L1: 0000000a

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: L2 HIT

SIM: L1 Write

SIM: Data from L1: 0000000a

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 3 = MODIFIED**

SIM: ===========================================================

SIM: command:4 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: Snooped Write

SIM: Write Back!

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 0 = INVALID**

SIM: ===========================================================

SIM: Tag: 16383 Index: 0 Word 0

SIM: L2 MISS

SIM: Task : Cache\_Line\_Fill

SIM: Way: 0, Tag: 16383, Line: 0

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 1 = EXCLUSIVE**

===========================================================SIM: command:3 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: Snooped Read

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 2 = SHARED**

SIM: ===========================================================

SIM: command:1 address:fffc0000

SIM: Data from L1: 0000000a

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: L2 HIT

SIM: L1 Write

SIM: Data from L1: 0000000a

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 3 = MODIFIED**

SIM: ===========================================================

SIM: command:4 address:fffc0000

SIM: Tag: 16383 Index: 0 Word 0

SIM: Task: Look\_For\_Match

SIM: Matching Way: 0

SIM: Snooped Write

SIM: Write Back!

SIM: MESI State

SIM: Way: 0, Index: 2 Word

**SIM: MESI: 0 = INVALID**

SIM: ===========================================================

1. Extra Credit B – Hit rate competition
   1. Fully Associative Cache design
   2. Testing
      1. Cache Parameters:

|  |  |  |
| --- | --- | --- |
| **# of lines** | **# of words** | **Burst length** |
| 4 | 4 | 4 |

* + 1. Initialization
       1. Initialize cache so that each word represent the position of the word
       2. Result

SIM: Index: 0

SIM: Word 0 : 00000000

SIM: Word 1 : 00000001

SIM: Word 2 : 00000002

SIM: Word 3 : 00000003

SIM: Index: 1

SIM: Word 0 : 00000000

SIM: Word 1 : 00000001

SIM: Word 2 : 00000002

SIM: Word 3 : 00000003

SIM: Index: 2

SIM: Word 0 : 00000000

SIM: Word 1 : 00000001

SIM: Word 2 : 00000002

SIM: Word 3 : 00000003

SIM: Index: 3

SIM: Word 0 : 00000000

SIM: Word 1 : 00000001

SIM: Word 2 : 00000002

SIM: Word 3 : 00000003

* + 1. Fill Cache Line
       1. Cause cache misses with following addresses (FFFC0000, FFF80000, FFF40000) and see if three different lines are filled.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tag[31:18] | | | | | | | | | | | | | | Index[17:6] | | | | | | | | | | | | Word | | | | | 00 | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16383 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 8 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16382 | | | | | | | | | | | | | | **1** | | | | | | | | | | | | | 0 | | | | | | 0 | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  | |  |
| F | | | | F | | | | F | | | | 4 | | | | 0 | | | | 0 | | | | 0 | | | | | 0 | | | | | |
| 16381 | | | | | | | | | | | | | | **0** | | | | | | | | | | | | | 0 | | | | | | 0 | |

* + - 1. Test Vectors

0 fffc0000

0 fff80000

0 fff40000

* + - 1. Result

SIM: index: 0

SIM: Word: 0 Content: fffc0000

SIM: Word: 1 Content: fffc0004

SIM: Word: 2 Content: fffc0008

SIM: Word: 3 Content: fffc000c

SIM: index: 1

SIM: Word: 0 Content: fff80000

SIM: Word: 1 Content: fff80004

SIM: Word: 2 Content: fff80008

SIM: Word: 3 Content: fff8000c

SIM: index: 2

SIM: Word: 0 Content: fff40000

SIM: Word: 1 Content: fff40004

SIM: Word: 2 Content: fff40008

SIM: Word: 3 Content: fff4000c

SIM: index: 3

SIM: Word: 0 Content: 00000000

SIM: Word: 1 Content: 00000001

SIM: Word: 2 Content: 00000002

SIM: Word: 3 Content: 00000003

* + 1. Cache Hit
       1. Three lines are filled by FFFC0000, FFF80000, FFF40000
       2. Access FFFC0000, FFF80000, FFF40000 and see if it gives hit
       3. Test Vectors

0 fffc0000

0 fff80000

0 fff40000

0 fffc0000

0 fff80000

0 fff40000

* + - 1. Results

SIM: command:0 address:fffc0000

SIM: Tag: 67104768 Word 0

**SIM: Index: 0**

**SIM: L2 HIT**

SIM: L1 Read

SIM: L2 write data: fffc0000

SIM: Tag: 67104768 Index: 0 Word 0

SIM: Data from L2: fffc0000

SIM: ===========================================================

SIM: command:0 address:fff80000

SIM: Tag: 67100672 Word 0

**SIM: Index: 1**

**SIM: L2 HIT**

SIM: L1 Read

SIM: L2 write data: fff80000

SIM: Tag: 67100672 Index: 1 Word 0

SIM: Data from L2: fff80000

SIM: ===========================================================

SIM: command:0 address:fff40000

SIM: Tag: 67096576 Word 0

**SIM: Index: 2**

**SIM: L2 HIT**

SIM: L1 Read

SIM: L2 write data: fff40000

SIM: Tag: 67096576 Index: 2 Word 0

SIM: Data from L2: fff40000

* + 1. Cache Miss
       1. Cause cache miss by accessing fff0000
       2. Test vectors

0 fffc0000

0 fff80000

0 fff40000

0 fff00000

* + - 1. Results

SIM: command:0 address:fff00000

SIM: Tag: 67092480 Word 0

SIM: Index: 0

**SIM: L2 MISS**

SIM: Word: 0 Content: fff00000

SIM: L1 Read

SIM: L2 write data: fff00000

SIM: Tag: 67092480 Index: 3 Word 0

SIM: index: 0

SIM: Word: 0 Content: fffc0000

SIM: Word: 1 Content: fffc0004

SIM: Word: 2 Content: fffc0008

SIM: Word: 3 Content: fffc000c

SIM: index: 1

SIM: Word: 0 Content: fff80000

SIM: Word: 1 Content: fff80004

SIM: Word: 2 Content: fff80008

SIM: Word: 3 Content: fff8000c

SIM: index: 2

SIM: Word: 0 Content: fff40000

SIM: Word: 1 Content: fff40004

SIM: Word: 2 Content: fff40008

SIM: Word: 3 Content: fff4000c

**SIM: index: 3**

**SIM: Word: 0 Content: fff00000**

**SIM: Word: 1 Content: fff00004**

**SIM: Word: 2 Content: fff00008**

**SIM: Word: 3 Content: fff0000c**

* + 1. Random replacement of line
       1. Cause two miss when all four lines are filled and see if random lines are replaced when **capacity miss occurs**
       2. Test vectors

0 fffc0000

0 fff80000

0 fff40000

0 fff00000

0 ffdc0000

0 fffc0000

* + - 1. Results

SIM: command:0 address:ffdc0000

SIM: Tag: 67072000 Word 0

**SIM: Index: 0**

**SIM: L2 MISS**

SIM: Word: 0 Content: ffdc0000

SIM: L1 Read

SIM: L2 write data: ffdc0000

SIM: Tag: 67072000 Index: 0 Word 0

SIM: index: 0

SIM: Word: 0 Content: ffdc0000

SIM: Word: 1 Content: ffdc0004

SIM: Word: 2 Content: ffdc0008

SIM: Word: 3 Content: ffdc000c

SIM: index: 1

SIM: Word: 0 Content: fff80000

SIM: Word: 1 Content: fff80004

SIM: Word: 2 Content: fff80008

SIM: Word: 3 Content: fff8000c

SIM: index: 2

SIM: Word: 0 Content: fff40000

SIM: Word: 1 Content: fff40004

SIM: Word: 2 Content: fff40008

SIM: Word: 3 Content: fff4000c

SIM: index: 3

SIM: Word: 0 Content: fff00000

SIM: Word: 1 Content: fff00004

SIM: Word: 2 Content: fff00008

SIM: Word: 3 Content: fff0000c

SIM: Data from L2: ffdc0000

SIM: ===========================================================

SIM: command:0 address:fffc0000

SIM: Tag: 67104768 Word 0

SIM: Index: 0

SIM: L2 MISS

SIM: Word: 0 Content: fffc0000

SIM: L1 Read

SIM: L2 write data: fffc0000

**SIM: Tag: 67104768 Index: 1 Word 0**

SIM: index: 0

SIM: Word: 0 Content: ffdc0000

SIM: Word: 1 Content: ffdc0004

SIM: Word: 2 Content: ffdc0008

SIM: Word: 3 Content: ffdc000c

**SIM: index: 1**

**SIM: Word: 0 Content: fffc0000**

**SIM: Word: 1 Content: fffc0004**

**SIM: Word: 2 Content: fffc0008**

**SIM: Word: 3 Content: fffc000c**

SIM: index: 2

SIM: Word: 0 Content: fff40000

SIM: Word: 1 Content: fff40004

SIM: Word: 2 Content: fff40008

SIM: Word: 3 Content: fff4000c

SIM: index: 3

SIM: Word: 0 Content: fff00000

SIM: Word: 1 Content: fff00004

SIM: Word: 2 Content: fff00008

SIM: Word: 3 Content: fff0000c

SIM: Data from L2: fffc0000

* 1. Performance Comparison

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Burst Length | Line Size (words) | # of Lines | Line Size (byte) | Hit | Misses | Total | Hit% | Miss% |
| 1024 | 2048 | 255 | 8192 | 999901 | 101 | 1000002 | 99.989900 | 0.010100 |
| 2048 | 4096 | 127 | 16384 | 999949 | 53 | 1000002 | 99.994700 | 0.005300 |
| 4096 | 8192 | 63 | 32768 | 999974 | 28 | 1000002 | 99.997200 | 0.002800 |
| 8192 | 16384 | 31 | 65536 | 999987 | 15 | 1000002 | 99.998500 | 0.001500 |
| 16384 | 32768 | 15 | 131072 | 999993 | 9 | 1000002 | 99.999100 | 0.000900 |
| 32768 | 65536 | 7 | 262144 | 999996 | 6 | 1000002 | 99.999400 | 0.000600 |

In the high-hit rate cache design, the fully associative cache with line size of 128Kbyte (32K words) gave the best performance for the trace file provided.

Since there are only 6 lines being utilized, the cache can have 6 lines only

Line Size =

Tag Bit width = 14 bits

Number of lines = 6 lines

Hit Rate = **99.9994 %**