课程名称： 计算机体系结构 实验类型： 综合
实验项目名称：Topic 4. Pipelined CPU with Cache
学生姓名：郭永军 专业： 计算机科学与技术 学号： 3200102126
同组学生姓名：无 指导老师： 翁恺
实验地点： 曹西301 实验日期： 2022 年 12 月 1 日

一、 实验目的和要求

Understand the principle of Cache Management Unit (CMU) and State Machine of CMU
Master the design methods of CMU and Integrate it to the CPU.
Master verification methods of CMU and compare the performance of CPU when it has cache or not.

二、 实验内容和原理

Experiment Tasks

1. Design of Cache Management Unit and integrate it to CPU.
2. Observe and Analyze the Waveform of Simulation.
3. Compare the performance of CPU when it has cache or not.

CMU (Cache Management Unit) Design

三、 实验过程和数据记录

主要实验任务是完成`cache.v`中的填空，具体功能与设计说明放在注释部分。

`cache.v`

```
module cmu (
        // CPU side
        input clk,
        input rst,
        input [31:0] addr_rw,
        input en_r,
        input en_w,
        input [2:0] u_b_h_w,
        input [31:0] data_w,
        output [31:0] data_r,
        output stall,

        // mem side
        output reg mem_cs_o = 0,
        output reg mem_we_o = 0,
        output reg [31:0] mem_addr_o = 0,
        input [31:0] mem_data_i,
        output [31:0] mem_data_o,
        input mem_ack_i,

        // debug info
        output [2:0] cmu_state,
```

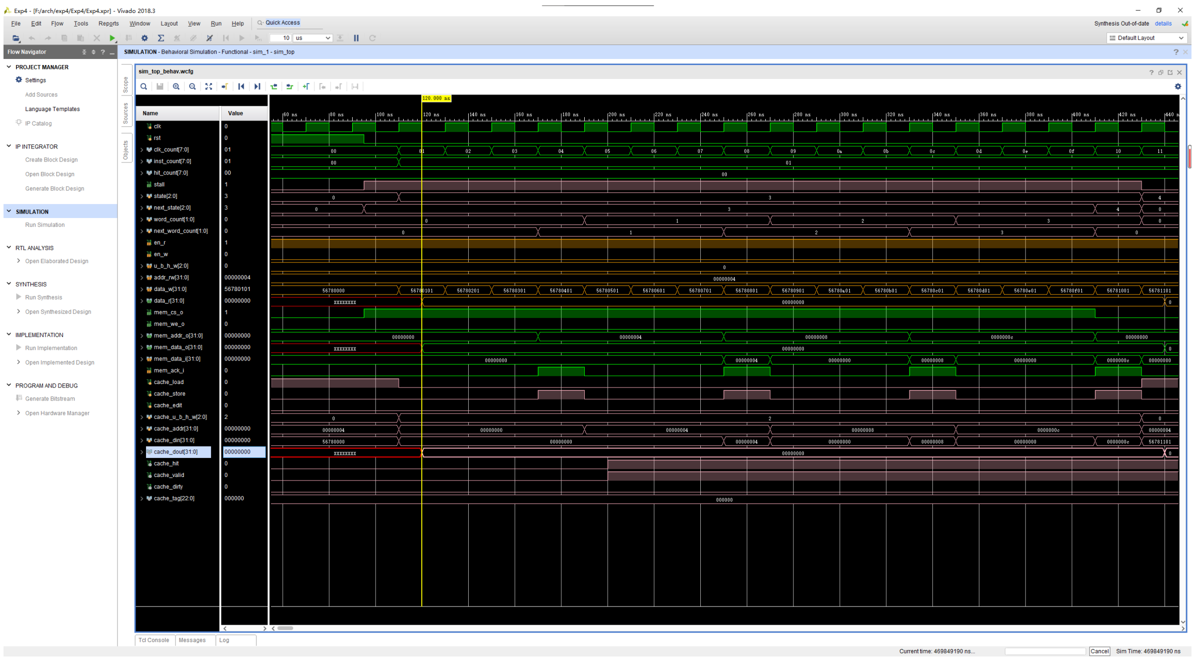
```verilog
        output cmu_hit
    );

    /* debug signal */
    assign cmu_hit = cache_hit;

    `include "addr_define.vh"

    reg [ADDR_BITS-1:0] cache_addr = 0;
    reg cache_load = 0;
    reg cache_store = 0;
    reg cache_edit = 0;
    reg [2:0] cache_u_b_h_w = 0;
    reg [WORD_BITS-1:0] cache_din = 0;
    wire cache_hit;
    wire [WORD_BITS-1:0] cache_dout;
    wire cache_valid;
    wire cache_dirty;
    wire [TAG_BITS-1:0] cache_tag;

    cache CACHE (
        .clk(~clk),
        .rst(rst),
        .addr(cache_addr),
        .load(cache_load),
        .store(cache_store),
        .edit(cache_edit),
        .invalid(1'b0),
        .u_b_h_w(cache_u_b_h_w),
        .din(cache_din),
        .hit(cache_hit),
        .dout(cache_dout),
        .valid(cache_valid),
        .dirty(cache_dirty),
        .tag(cache_tag)
    );

    localparam
        S_IDLE = 0,
        S_PRE_BACK = 1,
        S_BACK = 2,
        S_FILL = 3,
        S_WAIT = 4;

    reg [2:0]state = 0;
    reg [2:0]next_state = 0;
    reg [ELEMENT_WORDS_WIDTH-1:0]word_count = 0;
    reg [ELEMENT_WORDS_WIDTH-1:0]next_word_count = 0;
    assign cmu_state = state;

    always @ (posedge clk) begin
        if (rst) begin
            state <= S_IDLE;
            word_count <= 2'b00;
        end
        else begin
            state <= next_state;
            word_count <= next_word_count;
        end
    end

    // state ctrl
    always @ (*) begin
        if (rst) begin
            next_state = S_IDLE;
            next_word_count = 2'b00;
        end
        else begin
            case (state)
                S_IDLE: begin
                    if (en_r || en_w) begin
                        if (cache_hit)
                            next_state = S_IDLE;
                        else if (cache_valid && cache_dirty)
                            next_state = S_PRE_BACK;
                        else
                            next_state = S_FILL;
                    end
                    next_word_count = 2'b00;
                end

                S_PRE_BACK: begin
                    next_state = S_BACK;
                    next_word_count = 2'b00;
                end

                S_BACK: begin
                    if (mem_ack_i && word_count == {ELEMENT_WORDS_WIDTH{1'b1}})    // 2'b11 in default case
                        next_state = S_FILL;
                    else
                        next_state = S_BACK;

                    if (mem_ack_i)
                        next_word_count = word_count + 1;
                    else
                        next_word_count = word_count;
                end

                S_FILL: begin
                    if (mem_ack_i && word_count == {ELEMENT_WORDS_WIDTH{1'b1}})
                        next_state = S_WAIT;
                    else
                        next_state = S_FILL;

                    if (mem_ack_i)
                        next_word_count = word_count + 1;
                    else
                        next_word_count = word_count;
                end

                S_WAIT: begin
                    next_state = S_IDLE;
                    next_word_count = 2'b00;
                end
                endcase
            end
        end

        // cache ctrl
        always @ (*) begin
            case(state)
                S_IDLE, S_WAIT: begin
                    cache_addr = addr_rw;
                    cache_load = en_r;
                    cache_edit = en_w;
                    cache_store = 1'b0;
                    cache_u_b_h_w = u_b_h_w;
                    cache_din = data_w;
                end
                S_BACK, S_PRE_BACK: begin
                    cache_addr = {addr_rw[ADDR_BITS-1:BLOCK_WIDTH], next_word_count, {ELEMENT_WORDS_WIDTH{1'b0}}};
                    cache_load = 1'b0;
                    cache_edit = 1'b0;
                    cache_store = 1'b0;
                    cache_u_b_h_w = 3'b010;
                    cache_din = 32'b0;
                end
                S_FILL: begin
                    cache_addr = {addr_rw[ADDR_BITS-1:BLOCK_WIDTH], word_count, {ELEMENT_WORDS_WIDTH{1'b0}}};
                    cache_load = 1'b0;
                    cache_edit = 1'b0;
                    cache_store = mem_ack_i;
                    cache_u_b_h_w = 3'b010;
                    cache_din = mem_data_i;
                end
            endcase
        end
        assign data_r = cache_dout;

        // mem ctrl
        always @ (*) begin
            case (next_state)
                S_IDLE, S_PRE_BACK, S_WAIT: begin
                    mem_cs_o = 1'b0;
                    mem_we_o = 1'b0;
                    mem_addr_o = 32'b0;
                end

                S_BACK: begin
                    mem_cs_o = 1'b1;
                    mem_we_o = 1'b1;
                    mem_addr_o = {cache_tag, addr_rw[ADDR_BITS-TAG_BITS-1:BLOCK_WIDTH], next_word_count, {ELEMENT_WORDS_WIDTH{1'b0}}};
                end

                S_FILL: begin
                    mem_cs_o = 1'b1;
                    mem_we_o = 1'b0;
                    mem_addr_o = {addr_rw[ADDR_BITS-1:BLOCK_WIDTH], next_word_count, {ELEMENT_WORDS_WIDTH{1'b0}}};
                end
            endcase
        end
        assign mem_data_o = cache_dout;

        assign stall = (next_state != S_IDLE);

    endmodule
```

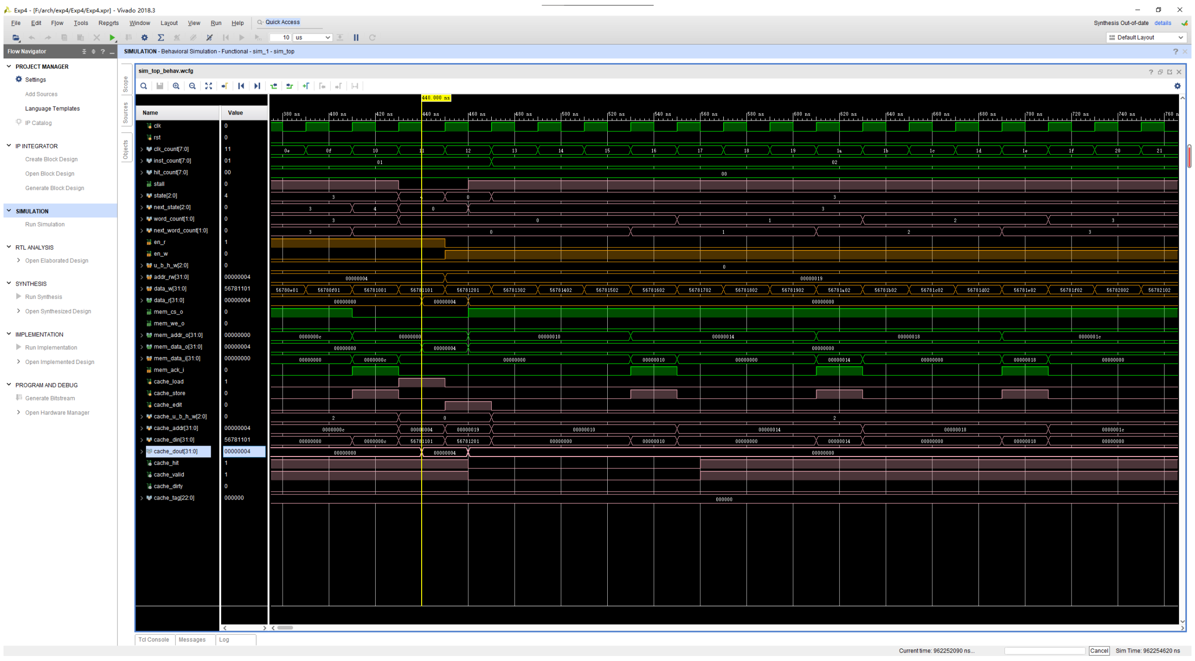
四、 实验结果分析

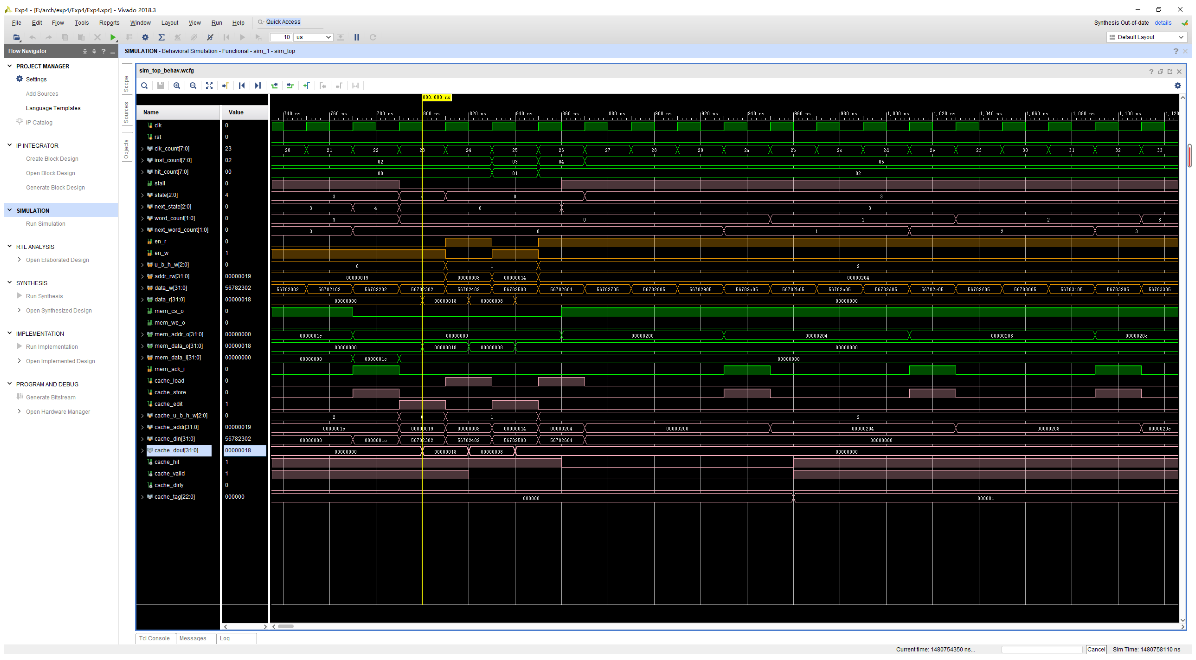
本实验通过仿真波形和上板单步运行观察指令执行是否正确来验证正确性：

仿真波形

对比仿真波形可知，cache实现了预期功能。

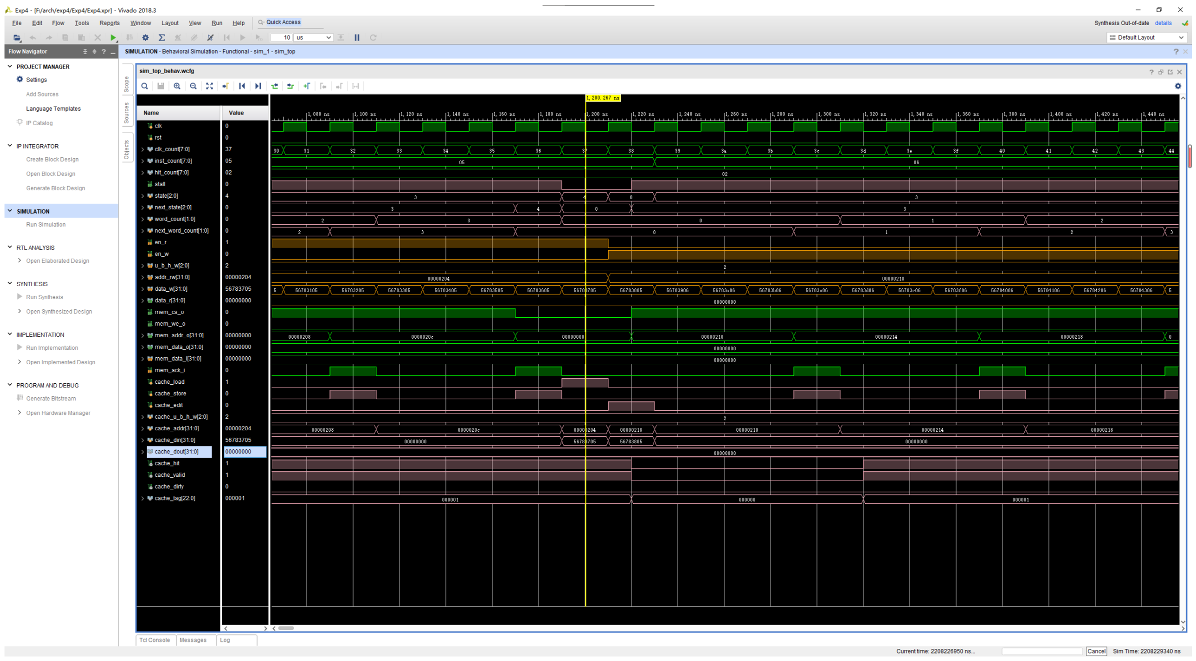





Exp4 - [F:/arch/exp4/Exp4/Exp4.xpr] - Vivado 2018.3
SIMULATION - Behavioral Simulation - Functional - sim_1 - sim_top
sim_top_behav.wcfg
800.000 ns

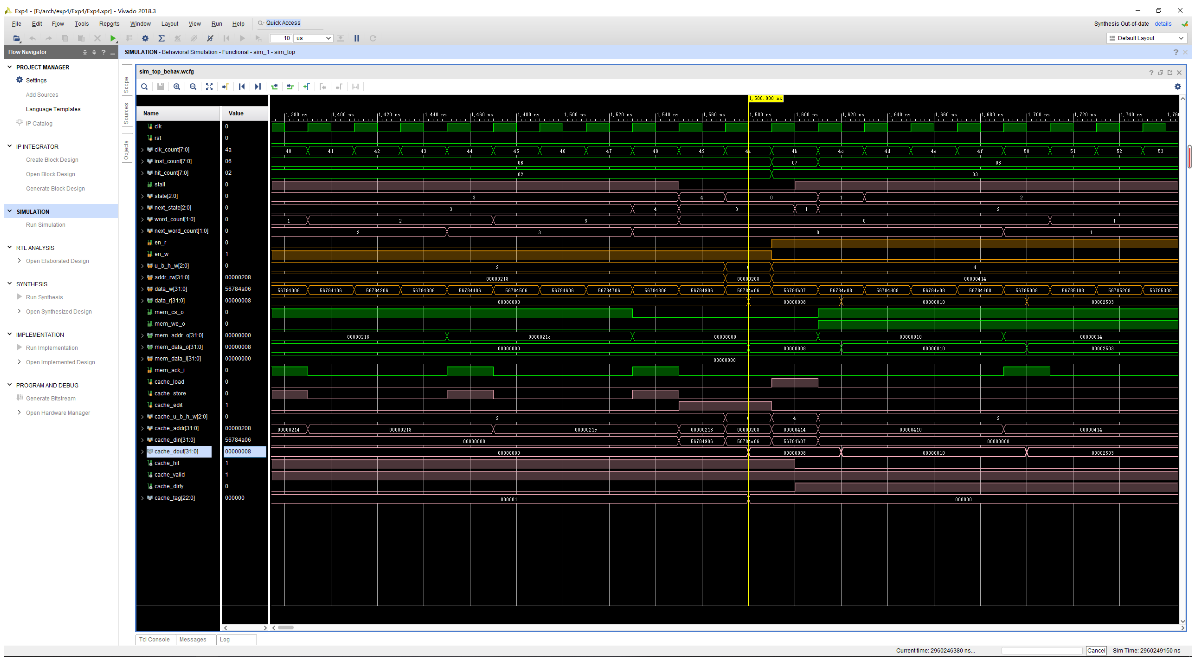
Name	Value
clk	0
rst	0
clk_count[7:0]	23
inst_count[7:0]	02
hit_count[7:0]	00
stall	0
state[2:0]	4
next_state[2:0]	0
word_count[1:0]	0
next_word_count[1:0]	0
en_r	0
en_w	1
u_b_h_w[2:0]	0
addr_rw[31:0]	00000019
data_w[31:0]	56782302
data_r[31:0]	00000018
mem_cs_o	0
mem_we_o	0
mem_addr_o[31:0]	00000000
mem_data_o[31:0]	00000018
mem_data_i[31:0]	00000000
mem_ack_i	0
cache_load	0
cache_store	0
cache_edit	1
cache_u_b_h_w[2:0]	0
cache_addr[31:0]	00000019
cache_din[31:0]	56782302
cache_dout[31:0]	00000018
cache_hit	1
cache_valid	1
cache_dirty	0
cache_tag[22:0]	000000

Current time: 1480754350 ns... Sim Time: 1480758110 ns


Exp4 - [F:/arch/exp4/Exp4/Exp4.xpr] - Vivado 2018.3
SIMULATION - Behavioral Simulation - Functional - sim_1 - sim_top
sim_top_behav.wcfg
1,200.267 ns

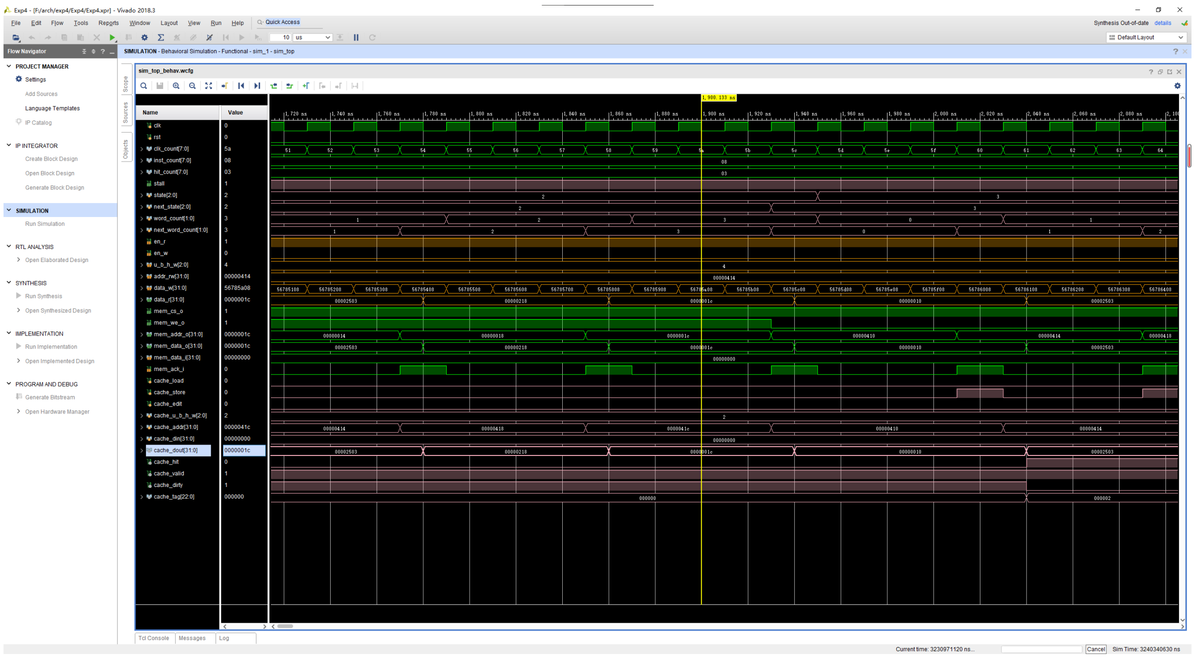
Name	Value
clk	0
rst	0
clk_count[7:0]	37
inst_count[7:0]	05
hit_count[7:0]	02
stall	0
state[2:0]	4
next_state[2:0]	0
word_count[1:0]	0
next_word_count[1:0]	0
en_r	1
en_w	0
u_b_h_w[2:0]	2
addr_rw[31:0]	00000204
data_w[31:0]	56783705
data_r[31:0]	00000000
mem_cs_o	0
mem_we_o	0
mem_addr_o[31:0]	00000000
mem_data_o[31:0]	00000000
mem_data_i[31:0]	00000000
mem_ack_i	0
cache_load	1
cache_store	0
cache_edit	0
cache_u_b_h_w[2:0]	2
cache_addr[31:0]	00000204
cache_din[31:0]	56783705
cache_dout[31:0]	00000000
cache_hit	0
cache_valid	1
cache_dirty	0
cache_tag[22:0]	000001

Current time: 2208226950 ns... Sim Time: 2208229340 ns


Exp4 - [F:/arch/exp4/Exp4/Exp4.xpr] - Vivado 2018.3
SIMULATION - Behavioral Simulation - Functional - sim_1 - sim_top
sim_top_behav.wcfg
1,580.000 ns

Name	Value
clk	0
rst	0
clk_count[7:0]	4a
inst_count[7:0]	06
hit_count[7:0]	02
stall	0
state[2:0]	0
next_state[2:0]	0
word_count[1:0]	0
next_word_count[1:0]	0
en_r	0
en_w	1
u_b_h_w[2:0]	0
addr_rw[31:0]	00000208
data_w[31:0]	56784a06
data_r[31:0]	00000008
mem_cs_o	0
mem_we_o	0
mem_addr_o[31:0]	00000000
mem_data_o[31:0]	00000008
mem_data_i[31:0]	00000000
mem_ack_i	0
cache_load	0
cache_store	0
cache_edit	1
cache_u_b_h_w[2:0]	0
cache_addr[31:0]	00000208
cache_din[31:0]	56784a06
cache_dout[31:0]	00000008
cache_hit	
cache_valid	1
cache_dirty	0
cache_tag[22:0]	000000

Current time: 2960246380 ns... Sim Time: 2960249150 ns



Name	Value
clk	0
rst	0
clk_count[7:0]	5a
inst_count[7:0]	08
hit_count[7:0]	03
stall	1
state[2:0]	2
next_state[2:0]	2
word_count[1:0]	3
next_word_count[1:0]	3
en_r	1
en_w	0
u_b_h_w[2:0]	4
addr_rw[31:0]	00000414
data_w[31:0]	56785a08
data_r[31:0]	0000001c
mem_cs_o	1
mem_we_o	1
mem_addr_o[31:0]	0000001c
mem_data_o[31:0]	0000001c
mem_data_i[31:0]	00000000
mem_ack_i	0
cache_load	0
cache_store	0
cache_edit	0
cache_u_b_h_w[2:0]	2
cache_addr[31:0]	0000041c
cache_din[31:0]	00000000
cache_dout[31:0]	0000001c
cache_hit	0
cache_valid	1
cache_dirty	1
cache_tag[22:0]	000000

```
clk_count:   51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64
inst_count:  08
hit_count:   03
state:       2 3
next_state:  2 3
word_count:  1 2 3 0 1
next_word_count: 1 2 3 0 1 2
u_b_h_w:     4
addr_rw:     00000414
data_w:      56785108 56785208 56785308 56785408 56785508 56785608 56785708 56785808 56785908 56785a08 56785b08 56785c08 56785d08 56785e08 56785f08 56786008 56786108 56786208 56786308 56786408
data_r:      00002503 00000218 0000001c 00000010 00002503
mem_addr_o:  00000014 00000018 0000001c 00000410 00000414 00000418
mem_data_o:  00002503 00000218 0000001c 00000010 00002503
mem_data_i:  00000000
cache_u_b_h_w: 2
cache_addr:  00000414 00000418 0000041c 00000410 00000414
cache_din:   00000000
cache_dout:  00002503 00000218 0000001c 00000010 00002503
cache_tag:   000000 000002
```

Current time: 3230971120 ns...

Sim Time: 3240340630 ns