浙江大学实验报告

课程名称：计算机体系结构 实验类型：综合

实验项目名称：Topic 3. Cache Design

指导教师：何水兵 完成时间：2023.11.7

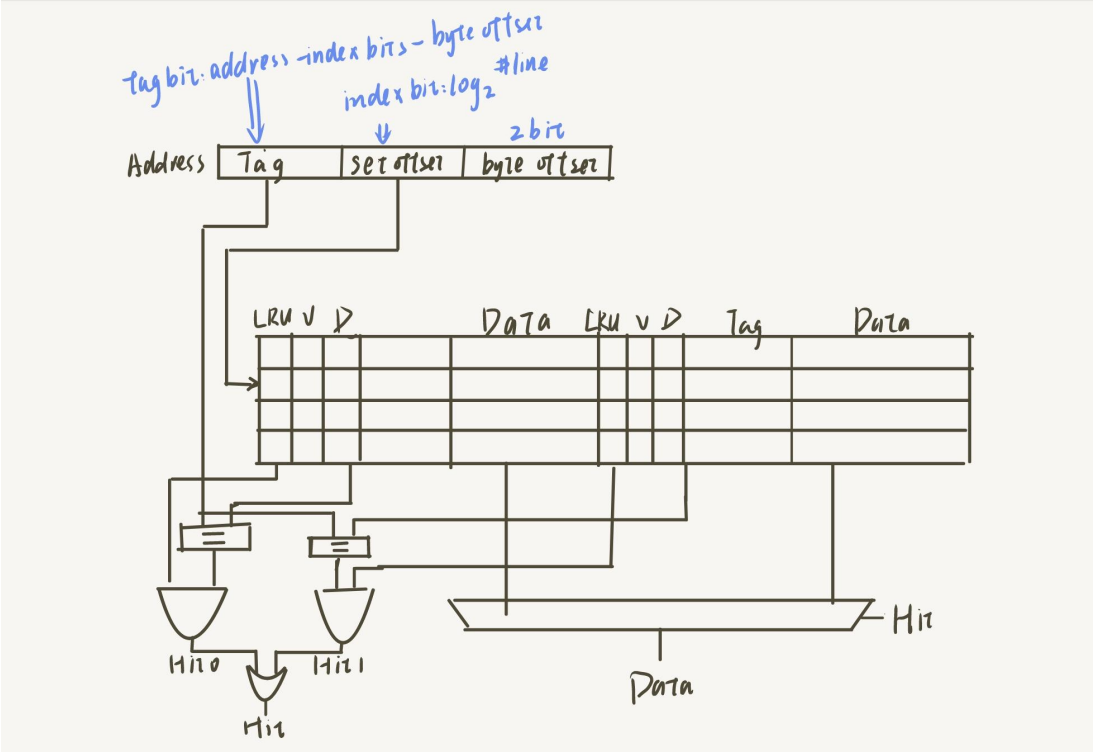
姓名	詹含蓓	**学号**	3210106333
同组学生姓名	居圣桐	**学号**	3210105812
分工信息	詹含蓓：共同完成 居圣桐：共同完成		

一、 实验目的和要求

- Understand Cache Line.
- Understand the principle of Cache Management Unit (CMU) and State Machine of CMU.
- Master the design methods of CMU.
- Master the design methods of Cache Line.
- master verification methods of Cache Line.

二、 实验内容和原理

- 二路组相连 cache 结构和访问方式如下：
  - 首先在地址当中计算 index 和 offset 的位数，其余为 tag 位数
  - 通过 index 查找 cache 当中的 line 地址
  - 由于此次实验当中的 cache 是二路主关联，所以一个 line 当中存放两个信息。首先判断 valid 位，接着分别比对两个 tag 是否和地址当中的 tag 相同，相同则 hit，两个都不同则 miss
  - 写步骤：找到 LRU 中指向非最近使用的 block 进行替换，判断是否为脏页，如果为脏则写入 memory



三、　实验过程和数据记录及结果分析

（一）实验过程

1. 根据地址区分写出 tag，index 内容。

```
assign addr_tag = addr[31:9];              //need to fill in
assign addr_index = addr[8:4];             //need to fill in
assign addr_element1 = {addr_index, 1'b0};
assign addr_element2 = {addr_index, 1'b1};      //need to fill in
assign addr_word1 = {addr_element1, addr[ELEMENT_WORDS_WIDTH+WORD_BYTES_WIDTH-1:WORD_BYTES_WIDTH]};
assign addr_word2 = {addr_element2, addr[ELEMENT_WORDS_WIDTH+WORD_BYTES_WIDTH-1:WORD_BYTES_WIDTH]};          //need to fill in
```

2. 生成 cache 内部信号。本实验使用的是二路主关联，因此所有的信号都分为两种。

```
assign word1 = inner_data[addr_word1];
assign word2 = inner_data[addr_word2];                //need to fill in
assign half_word1 = addr[1] ? word1[31:16] : word1[15:0];
assign half_word2 = addr[1] ? word2[31:16] : word2[15:0];          //need to fill in
assign byte1 = addr[1] ?
               addr[0] ? word1[31:24] : word1[23:16] :
               addr[0] ? word1[15:8] :  word1[7:0]   ;
assign byte2 =  addr[1] ?
               addr[0] ? word2[31:24] : word2[23:16] :
               addr[0] ? word2[15:8] :  word2[7:0]   ;               //need to fill in

assign recent1 = inner_recent[addr_element1];
assign recent2 = inner_recent[addr_element2];             //need to fill in
assign valid1 = inner_valid[addr_element1];
assign valid2 = inner_valid[addr_element2];              //need to fill in
assign dirty1 = inner_dirty[addr_element1];
assign dirty2 = inner_dirty[addr_element2];;              //need to fill in
assign tag1 = inner_tag[addr_element1];
assign tag2 = inner_tag[addr_element2];                 //need to fill in

assign hit1 = valid1 & (tag1 == addr_tag);
assign hit2 = valid2 & (tag2 == addr_tag);                //need to fill in
```

3. 根据内部信号生成输出信号。本实验用的是 LRU 策略，因此要输出的是非 recent 的块的信号。通过判断 recent1 的值是否为 1 来判断需要输出的是哪一个块。

```
always @ (posedge clk) begin
    valid <= recent1 ? valid2 : valid1;                  //need to fill in
    dirty <= recent1 ? dirty2 : dirty1;                  //need to fill in
    tag <= recent1 ? tag2 : tag1;                       //need to fill in
    hit <= hit1 | hit2;                        //need to fill in
```

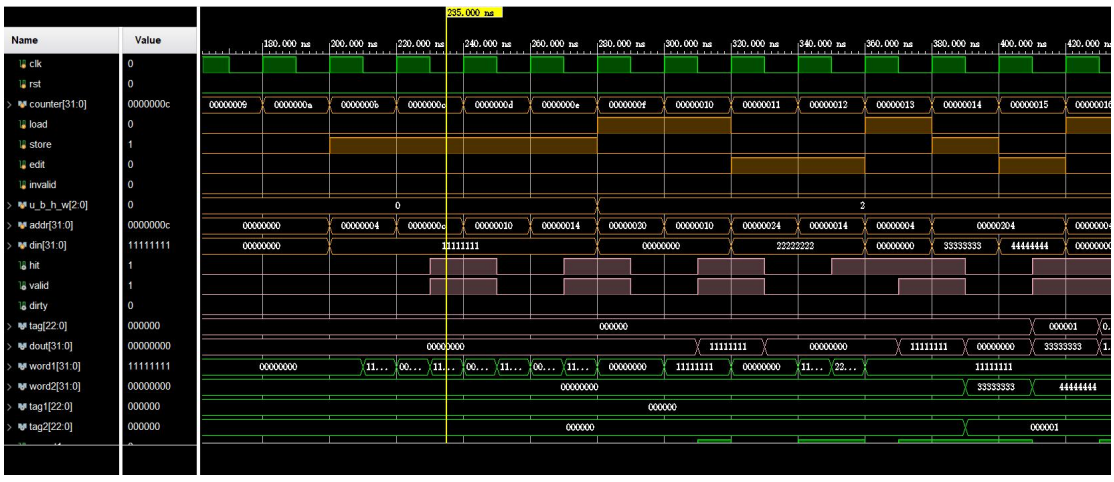
4. 编写 cache 换页行为。发生换页时，整个块中的内容改为输入的数据，并重置 valid、dirty 和 tag 信息。

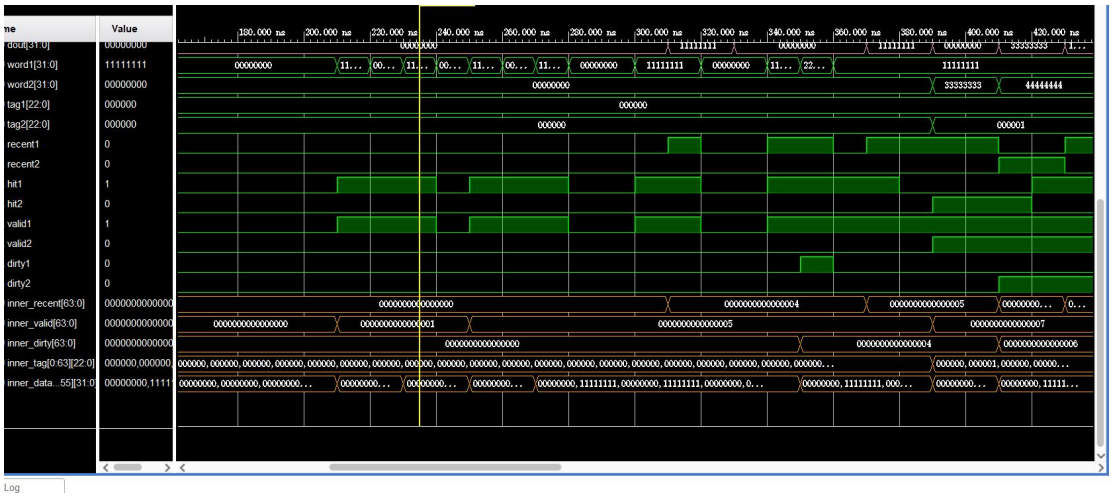
```
// otherwise the refresh process will be affected
if (load) begin
    if (hit1) begin
        dout <=
            u_b_h_w[1] ? word1 :
            u_b_h_w[0] ? {u_b_h_w[2] ? 16'b0 : {16{half_word1[15]}}, half_word1} :
            {u_b_h_w[2] ? 24'b0 : {24{byte1[7]}}, byte1};

        // inner_recent will be refreshed only on r/w hit
        // (including the r/w hit after miss and replacement)
        inner_recent[addr_element1] <= 1'b1;
        inner_recent[addr_element2] <= 1'b0;
    end
    else if (hit2) begin
            //need to fill in
        dout <=
            u_b_h_w[1] ? word2 :
            u_b_h_w[0] ? {u_b_h_w[2] ? 16'b0 : {16{half_word2[15]}}, half_word2} :
            {u_b_h_w[2] ? 24'b0 : {24{byte1[7]}}, byte2};

        // inner_recent will be refreshed only on r/w hit
        // (including the r/w hit after miss and replacement)
        inner_recent[addr_element1] <= 1'b1;
        inner_recent[addr_element2] <= 1'b0;
    end
end
```

（二）实验结果及分析

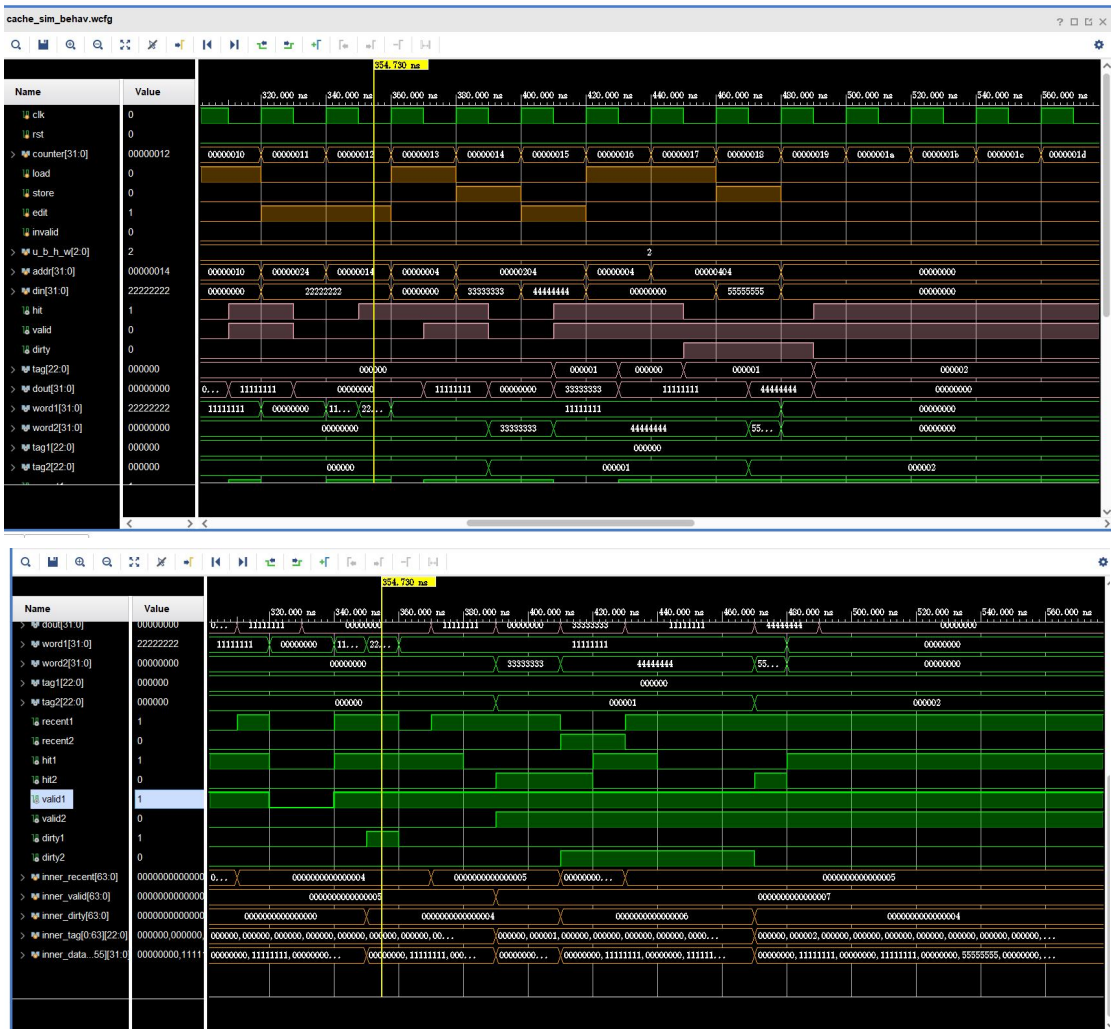
运行仿真，和 ppt 结果进行对比。仿真结果如下：

1. 进行 ld 的时候 line 0 hit 了第一个 block，recent1 设为 1；进行 store 的时候同理，将其 recent 进行设置





2. 进行 edit 操作，hit 了 line 当中的 block1，把其 dirty 位置于 1，同时 recent1 设置为 1



四、讨论与心得

本次实验主要是对 lab4 进行铺垫，其中根据实验代码文档当中的提示可以比

较好地完成这个实验。容易出现错误的地方就是在于输出变量的选择。我们需要选择输出的是要被替换掉的那个块的信号，这个部分和第四个模块是高度相关的，因此完成 lab3 有助于我理解 lab4 的完成。