

# Lab 5 report

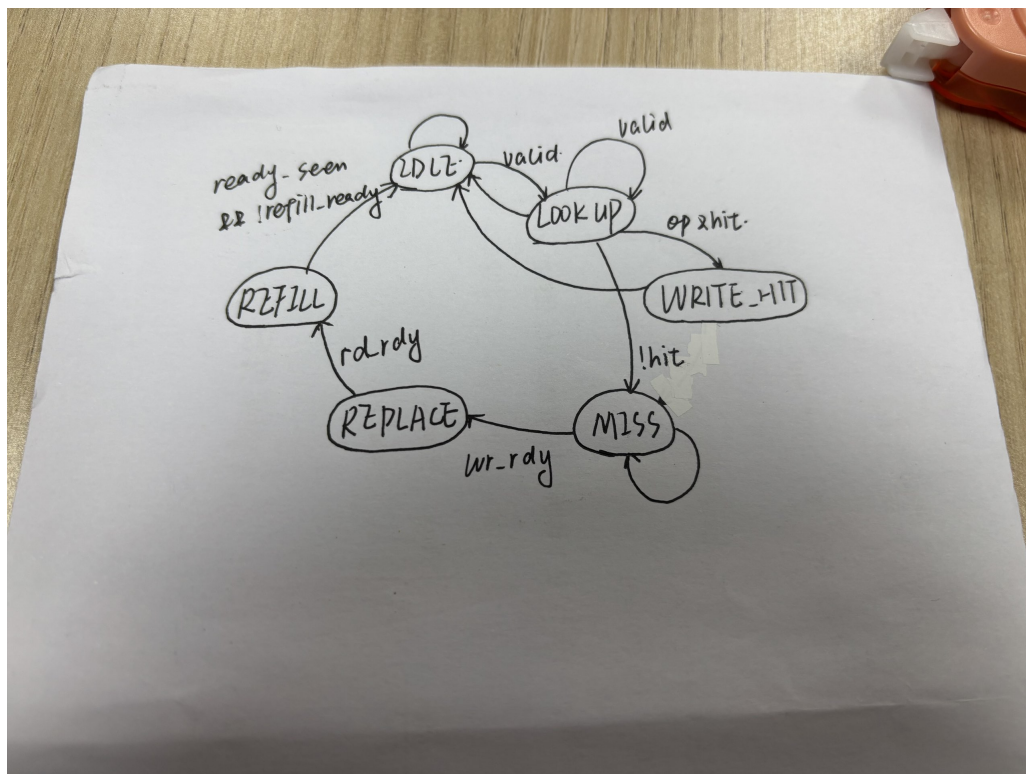
PB24111630 黄雯佩

## 实验目的与内容

1. 请用自己的语言简述本次实验做的内容以及本次实验的目的，不要照抄实验PPT哦！
  - (1) 了解cache的原理、工作机制
  - (2) 增强调整时序的能力
  - (3) 增强仿真看波形图的能力

## 逻辑设计

1. 如果存在状态机，请绘制出状态机的状态转换图；



2. 请贴出你认为较为核心的代码以及自己有两点的设计代码，并加以解释说明。

```

38     reg        ready_seen; // 用于记录在 REFILL 状态下是否已经捕捉到了下一个脉冲
39
40     always @(posedge clk) begin
41         if (!rstn) begin
42             current_state <= IDLE;
43             ready_seen    <= 1'b0;
44         end else begin
45             current_state <= next_state;
46             // 记录脉冲捕捉状态
47             if (current_state == REFILL && refill_ready)
48                 ready_seen <= 1'b1;
49             else if (current_state == IDLE)
50                 ready_seen <= 1'b0;
51         end
52     end
53
54     // 2. 合并后的次态计算与输出控制 (组合逻辑)
55     always @(*) begin
56         // --- 设置默认值, 防止产生锁存器 (Latch) ---
57         next_state = current_state;
58         miss       = 1'b0;
59         write_hit   = 1'b0;
60         addr_ok     = 1'b0; // 除非在某个状态下显式地将它设为 1, 否则默认为 0
61         data_ok     = 1'b0;
62         miss_start  = 1'b0;
63         wr_req      = 1'b0;
64         rd_req      = 1'b0;
65
66         case (current_state)
67             IDLE: begin
68                 addr_ok = 1'b1; // IDLE 状态随时准备接收新请求
69                 if (valid) begin
70                     next_state = LOOKUP;
71                 end
72             end
73             LOOKUP: begin
74                 // 记录命中/未命中
75                 if (valid) begin
76                     if (hit)
77                         write_hit = 1'b1;
78                     else
79                         miss = 1'b1;
80                 end
81                 if (write_hit)
82                     next_state = WRITEBACK;
83                 else
84                     next_state = MISS;
85             end
86             MISS: begin
87                 miss = 1'b1;
88                 if (wr_rdy) // 不管是否要写回, 都转移到 REPLACE 状态
89                     next_state = REPLACE;
90                 else
91                     next_state = MISS;
92             end
93             REPLACE: begin
94                 miss = 1'b1;
95                 if (select_valid) // 确保替换有效的时候再写回内存, 否则不用
96                     wr_req = 1'b1; // 向 AXI 发起写回请求
97                 if (rd_rdy) begin
98                     next_state = REFILL;
99                     rd_req = 1'b1; // 向 AXI 发起读请求
100                 end
101                 else
102                     next_state = REPLACE;
103             end
104             REFILL: begin
105                 miss = 1'b1;
106                 if (ready_seen && !refill_ready) begin // refill_ready 为高后空转一个周期, 等数据进入 WBUFFER
107                     next_state = IDLE;
108                     data_ok = 1'b1; // 在跳回 IDLE 的同时确认数据有效
109                 end
110                 else begin
111                     next_state = REFILL;
112                 end
113             end
114             default: next_state = IDLE;
115         endcase
116     end

```

refill\_ready置1后空转一个周期, 等数据进入WBUFFER

```
18
19 // 输出给 Cache RAM
20 output reg [1:0] ret_cnt, // 已返回数据计数
21 output reg [127:0] final_data, // 最终拼好的128位行
22 output reg refill_ready // 重填就绪信号
23 );
24
25 reg [31:0] data_storage [3:0]; // 存储 AXI 返回的原始数据
26 wire [1:0] target_bank = offset[3:2];
27
28 always @(posedge clk) begin
29     if (!rstn || miss_start) begin
30         ret_cnt <= 2'b00;
31     end
32     else if (ret_valid) begin
33         data_storage[ret_cnt] <= ret_data;
34         ret_cnt <= ret_cnt + 1'b1;
35     end
36 end
37
38 always @(posedge clk) begin
39     if (!rstn)
40         refill_ready <= 1'b0;
41     else if (ret_valid && (ret_last == 1'b1))
42         refill_ready <= 1'b1;
43     else
44         refill_ready <= 1'b0;
45 end
46
47 always @(*) begin
48     final_data = {data_storage[3], data_storage[2], data_storage[1], data_storage[0]};
49
50     if (op) begin
51         case (target_bank)
52             2'b00: begin
53                 if (wstrb[0]) final_data[ 7: 0] = wdata[ 7: 0];
```

读取来自内存的数据并合并为cache行

仿真结果与分析

1. 请给出你使用的仿真文件的运行结果截图，并对结果加以阐释；

```
index f6 finished
index f6 finished
index f7 finished
index f7 finished
index f8 finished
index f8 finished
index f9 finished
index f9 finished
index fa finished
index fa finished
index fb finished
index fb finished
index fc finished
index fc finished
index fd finished
index fd finished
index fe finished
index fe finished
index ff finished
=====
Test end!
----PASS!!!
$finish called at time : 708615 ns : File "E:/CoursesData/AnalogandDigitalCircuitsLab/cacheLab/cache_verify/testbench/testbench.v" Line 35
run: Time (s): cpu = 00:00:10 ; elapsed = 00:00:10 . Memory (MB): peak = 3128.059 ; gain = 0.000
```

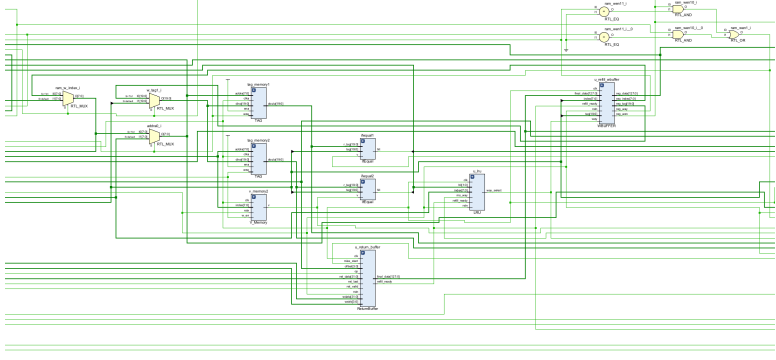
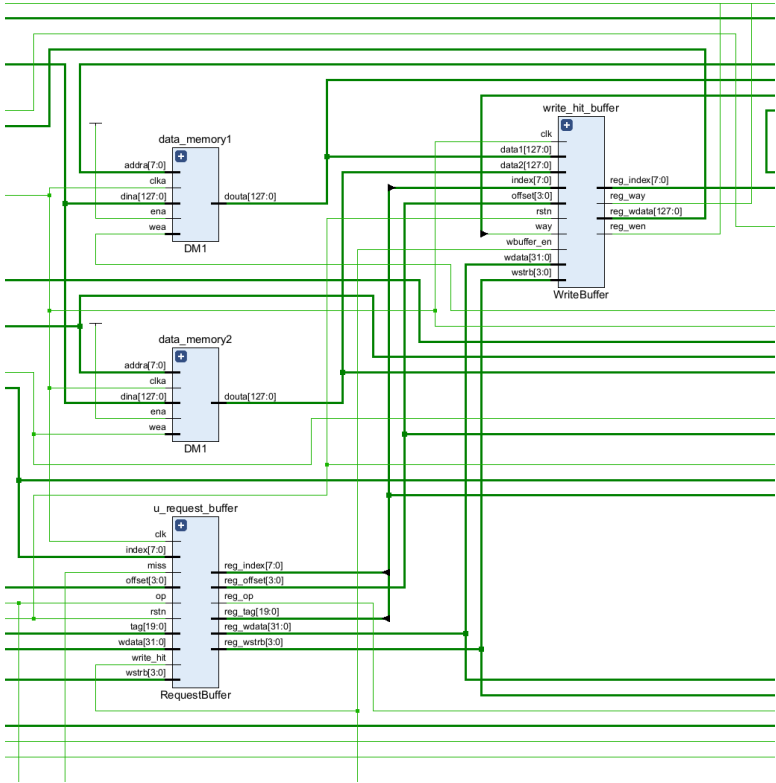
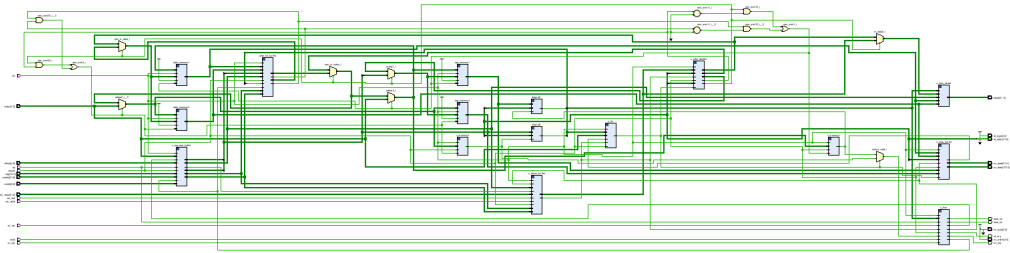
pass!!!

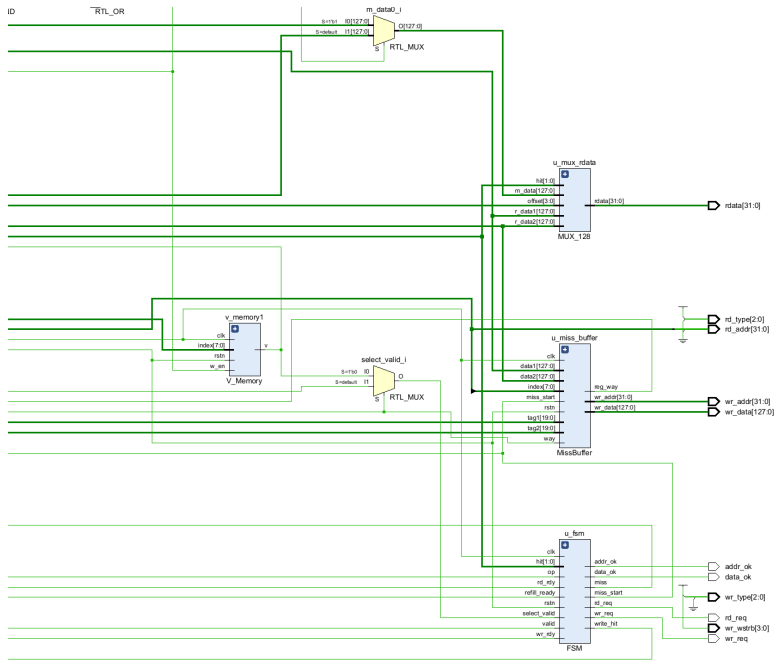
2. 请贴出你编写的有特点的仿真测试文件，并说明你在编写仿真测试文件时，对各类情况的考虑（选做）。

多亏助教给了仿真文件

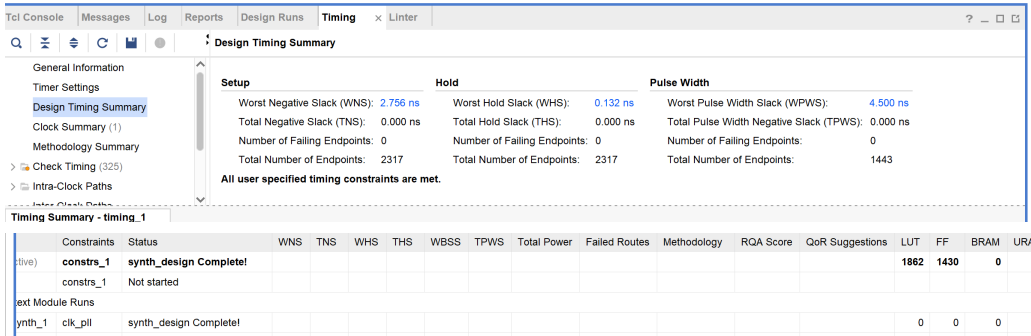
电路设计与分析

1. 请给出完整的RTL电路图。若某模块较为复杂，也可以再给出该模块的RTL电路图；



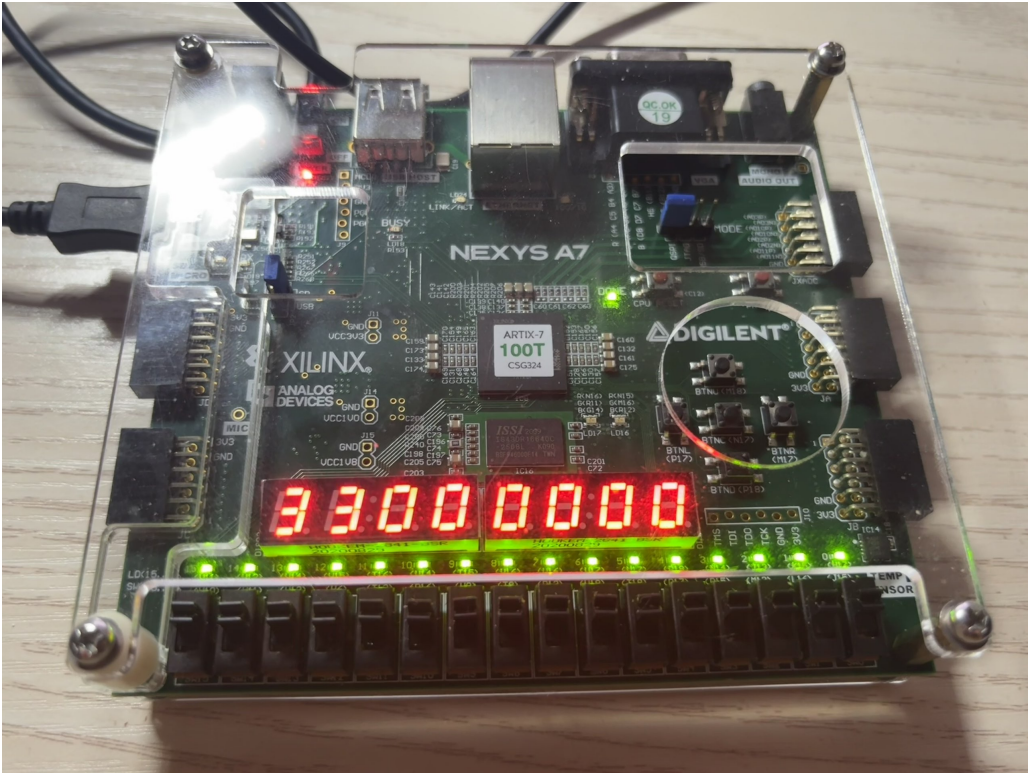


2. 查看并在此附上资源使用情况，并截图证明WNS为非负数。



测试结果与分析

1. 请拍照并附上实验上板结果，以佐证设计的正确性；







2. 对实验上板结果进行简要的说明。

复位后从七段数码管前两位从00-ff后停止

## 总结

1. 请对本次实验中你完成的任务进行简要总结，并总结自己的收获和体验；

上周四搞完的，那几天满脑子cachecache的，果然隔了一个周末就记不住什么了，连着搞了三天晚上，真是没debug过这么多，虽然到底也没弄清楚怎么能保存每次打开的仿真波形图窗口，可是看了n次index 00failed什么replace不对后，看到那个pass太感动了，真的纯靠自己发现不用ai修改了好多bug啊，一下子有自己是计算机专业的学生的感觉了

好像每次实验都是这样哎，做之前做的时候一头雾水，动不动要琢磨好久需要哪些模块，可是一旦鼓捣出来了，又发现就这么回事了

每次到跨年的时候，都怪感慨的哎，为什么要报这个提高班呢？觉得对专业方向不了解，想体验一下硬件吧，翻了翻聊天记录，发现是觉得这是一个参加比赛的机会，乐，当时问了助教和一个学长，感觉是不太建议的样子，不过还是想试试吧ddl之前报了，报完之后才去看评课社区，看着说一周动不动就要花七八个小时吓一跳，想着周末估计要搭上了，结果呢；最初的目的达到了吗？没有吧，一开始担心自己跟不上还听得挺认真的，后来忙起来周四去了也是去debug，第三次开始就没再听哎，看着不少同学做附加实验，想做又无力的感觉，不太好受的，十月份十一月份确实好忙，虽然不少程度上是忙着玩哎，确实发现自己没有做附加实验或者自己研究的精力了，不过还是没有太大的动力去鼓捣吧，毕竟这学期基本上都是周末不干啥，周一开始赶ddl赶到周四下午结束o(π\_π)o，因此想想最初的目的没了，并没有接触到多少硬件的东西，感觉自己的能力也不适合去打比赛什么的。可是，要是再选一次，还会这样选吧，虽然花了好多时间，好像也没有收获什么，但是总觉得真的体验到了，虽说具体体验到了什么也说不清楚，可是每次最后鼓捣出来的那一下子真的好爽好快乐好有成就感，或许体验到了怎么让FPGA按照我的意愿工作吧。下学期还有没有这个班不知道哎，会不会再去报也不知道的样子，哎好是想上的又感觉不太有性价比的滋味。

想想真的说不出来一般一周要花多久哎，估摸一下七八个小时？基本上都是周三周四搞的。

这学期也是结束啦，希望能有一个好好的结果吧

2. 如果对本次实验的设计或助教、老师有建议，可以在这里写下，助教和老师会认真阅读并讨论哦！

助教伟大！！非常及时得回复了我的好多问题，还有几次凌晨秒回我，太感动了

