浙江大学

本科实验报告

课程名称：计算机体系结构

姓　　名：刘思锐

学　　院：计算机科学与技术学院

系：

专　　业：计算机科学与技术

学　　号：3200102708

指导教师：陈文智

2022 年　11 月　20 日

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：Pipelined CPU with Cache

学生姓名：刘思锐 专业：计算机科学与技术 学号：3200102708

同组学生姓名：陈镛屹 指导老师：陈文智

实验地点：曹西 301 实验日期：2022 年 11 月 15 日

一、 实验目的和要求

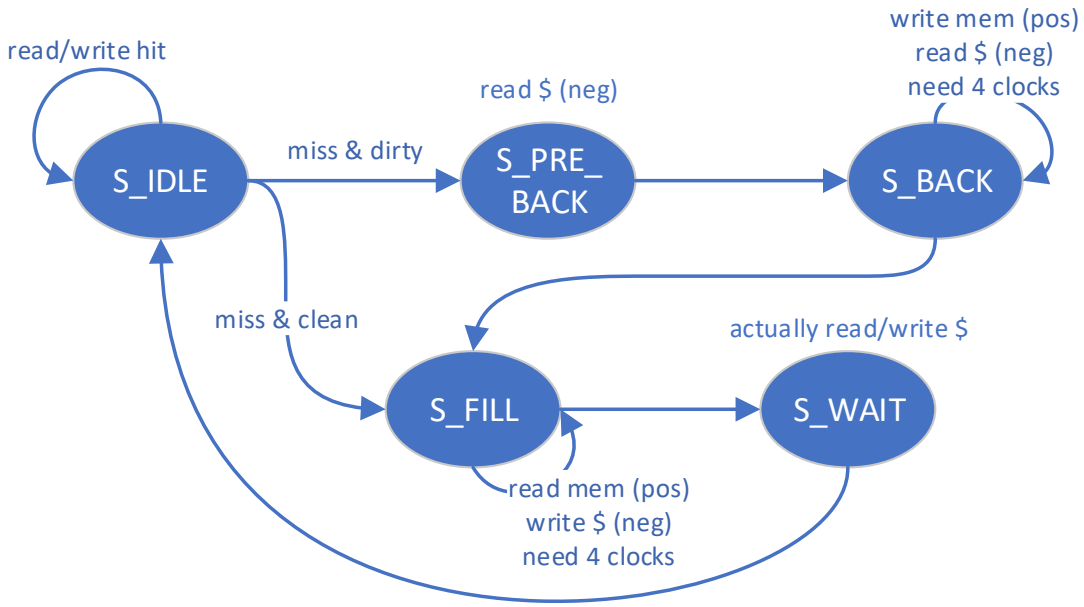
1. Understand the principle of Cache Management Unit (CMU) and State Machine of CMU.
2. Master the design methods of CMU and Integrate it to the CPU.
3. Master verification methods of CMU and compare the performance of CPU when it has cache or not.

二、 实验内容和原理

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.

三、 实验过程和数据记录

本次实验基于 lab3 Cache 设计，新增内容较少，只需要按照 PPT 所给状态图完成状态机。



3.1 idle 状态

```
S_IDLE: begin
    if (en_r || en_w) begin
        if (cache_hit)
            next_state = S_IDLE;     // TO FILL IN
        else if (cache_valid && cache_dirty)
            next_state = S_PRE_BACK;     // TO FILL IN
        else
            next_state = S_FILL;     // TO FILL IN
    end
    next_word_count = 2'b00;
end
```

3.2 back 状态

注意到本实验中为了模拟 Memory 慢于 Cache，这里手动设置了计数器，每四个时钟才会切换下一个状态。

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

    if (mem_ack_i)
        next_word_count = word_count + 1;     // TO FILL IN
    else
        next_word_count = word_count;
end
```

3.3 fill 状态

同 3.2 节，这里使用了 word_count 人为模拟 Mem 延迟。

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

    if (mem_ack_i)
        next_word_count = word_count + 1;     // TO FILL IN
    else
        next_word_count = word_count;
end
```

3.4 stall 信号生成

```
assign stall = (next_state != S_IDLE);     // TO FILL IN
```

四、 实验结果分析

NO.	Instruction	Addr.	Label	ASM	Comment
0	00000013	0	__start:	addi x0, x0, 0	
1	01c00083	4		lb x1, 0x01C(x0)	# F0F0F0F0 in 0x1C # FFFFFFF0 miss, read 0x010~0x01C to set 1 line 0
2	01c01103	8		lh x2, 0x01C(x0)	# FFFFF0F0 hit
3	01c02183	C		lw x3, 0x01C(x0)	# F0F0F0F0 hit
4	01c04203	10		lbu x4, 0x01C(x0)	# 000000F0 hit
5	01c05283	14		lhu x5, 0x01C(x0)	# 0000F0F0 hit
6	21002003	18		lw x0, 0x210(x0)	# miss, read 0x210~0x21C to cache set 1 line 1
7	abcde0b7	1C		lw x7, 20(x0)	
8	402200b3	20		lui x1 0xABCDE	
9	71c08093	24		addi x1, x1, 0x71C	# x1 = 0xABCDE71C
10	00100023	28		sb x1, 0x0(x0)	# miss, read 0x000~0x00C to cache set 0 line 0
11	00101223	2C		sh x1, 0x4(x0)	# hit
12	00102423	30		sw x1, 0x8(x0)	# hit

NO.	Instruction	Addr.	Label	ASM	Comment
13	20002303	34		lw x6, 0x200(x0)	# miss, read 0x200~0x20C to cache set 0 line 1
14	40002383	38		lw x7, 0x400(x0)	# miss, write 0x000~0x00C back to ram, then read 0x400~40C to cache set 0 line 0
15	41002403	3C		lw x8, 0x410(x0)	# miss, no write back because of clean, read 0x410~41C to chache set 1 line 0
16	0ed06813	40	loop:	lori x16, x0, 0xED	# end
17	ffdff06f	44		jal x0, loop	

本实验没有单独的仿真文件，上板后对照 PPT 可见处理器正常 Stall。（工程中的 ROM 缺少了 PPT 上位于 0x20 位置的 lui 指令，因此从 0x20 开始此后所有的指令的位置相比 PPT 提前了 4。）

五、 讨论与心得

第一次上板时没有发现工程的 ROM 与 PPT 有所区别，以为是自己的程序没有正确处理 stall 信号，但是 Lab3 中的仿真波形又与参考波形完全相同，当时在实验室非常惊慌，debug 也无从下手，好在最后通过开发板上的指令显

示找到了问题。

除此之外本次实验需要自己编写的代码不超过 10 行，状态机也非常清晰，实验过程中并没有遇到大的困难。