

COD LAB6 实验报告

姓名：刘芷辰

学号：PB21111728

日期：2023.6.14

1. 实验题目

流水线CPU指令集拓展

2. 实验目标

- 理解流水线CPU的结构和工作原理
- 熟练掌握流水线CPU数据通路和控制器的设计和描述方法
- 理解流水线CPU的调试方法
- 了解riscv指令集各指令的执行方式

3. 实验内容

3.1 指令集拓展（主要修改部分）

R型

- 代码

```
1 7'b0110011: begin
2     jal = 0;
3     jalr = 0;
4     br_type = 0;
5     rf_we = 1;
6     rf_wd_sel = 2'b00;
7     alu_src1_sel = 0;
8     alu_src2_sel = 0;
9     case(inst[14:12])
10         3'b000: begin
11             if(inst[30] == 0) alu_func = 0; //add
```

```

12         else alu_func = 1; //sub
13     end
14     3'b111: alu_func = 5; //and
15     3'b110: alu_func = 6; //or
16     3'b100: alu_func = 7; //xor
17     3'b010: alu_func = 4; //slt
18     3'b011: alu_func = 3; //sltu
19     3'b001: alu_func = 9; //sll
20     3'b101: begin
21         if(inst[30] == 0) alu_func = 8; //srl
22         else alu_func = 10; //sra
23     end
24     default: alu_func = 15;
25 endcase
26 mem_we = 0;
27 imm_type = 3'b000;
28 if(inst[19:15] != 5'h0) rf_re0 = 1'b1;
29 else rf_re0 = 1'b0;
30 if(inst[24:20] != 5'h0) rf_re1 = 1'b1;
31 else rf_re1 = 1'b0;
32 end
33

```

- 主要区别是根据inst[14:12]选择不同的alu_func

I型

- 代码

```

1 7'b0010011: begin
2     jal = 0;
3     jalr = 0;
4     br_type = 0;
5     rf_we = 1;
6     rf_wd_sel = 2'b00;
7     alu_src1_sel = 0;
8     alu_src2_sel = 1;
9     case(inst[14:12])
10        3'b000: alu_func = 0; //addi
11        3'b111: alu_func = 5; //andi
12        3'b110: alu_func = 6; //ori
13        3'b100: alu_func = 7; //xori

```

```

14         3'b010: alu_func = 4; //slti
15         3'b011: alu_func = 3; //sltiu
16         3'b001: alu_func = 9; //slli
17         3'b101: begin
18             if(inst[30] == 0) alu_func = 8; //srli
19             else alu_func = 10; //srai
20         end
21         default: alu_func = 15;
22     endcase
23     mem_we = 0;
24     if((inst[14:12] == 3'b101) & (inst[31:25] == 7'b0100000))
25         imm_type = 3'b110;
26     else
27         imm_type = 3'b001;
28         if(inst[19:15] != 5'h0) rf_re0 = 1'b1;
29         else rf_re0 = 1'b0;
30         rf_re1 = 1'b0;
31     end
32

```

- 主要区别是根据inst[14:12]选择不同的alu_func，并且对于imm_type，srai需要特殊处理为3'b110

B型

- 代码

```

1  7'b1100011: begin
2      jal = 0;
3      jalr = 0;
4      case(inst[14:12])
5          3'b000: br_type = 2; //beq
6          3'b100: br_type = 1; //blt
7          3'b001: br_type = 3; //bne
8          3'b101: br_type = 4; //bge
9          3'b110: br_type = 5; //bltu
10         3'b111: br_type = 6; //bgeu
11         default: br_type = 0;
12     endcase
13     rf_we = 0;
14     rf_wd_sel = 2'b11;
15     alu_src1_sel = 1;
16     alu_src2_sel = 1;
17     alu_func = 0;

```

```

18         mem_we = 0;
19         imm_type = 3'b010;
20         if(inst[19:15] != 5'h0) rf_re0 = 1'b1;
21         else rf_re0 = 1'b0;
22         if(inst[24:20] != 5'h0) rf_re1 = 1'b1;
23         else rf_re1 = 1'b0;
24     end
25

```

```

1  module Branch(
2      input [31:0]op1,
3      input [31:0]op2,
4      input [2:0]br_type,
5      output reg br
6  );
7
8      always@(*)
9      begin
10         case(br_type)
11             3'b001: begin                //blt
12                 if($signed(op1) < $signed(op2))
13                     br = 1;
14                 else
15                     br = 0;
16             end
17             3'b010: begin                //beq
18                 if(op1 == op2)
19                     br = 1;
20                 else
21                     br = 0;
22             end
23             3'b011: begin                //bne
24                 if(op1 != op2)
25                     br = 1;
26                 else
27                     br = 0;
28             end
29             3'b100: begin                //bge
30                 if($signed(op1) >= $signed(op2))
31                     br = 1;
32                 else
33                     br = 0;
34             end
35             3'b101: begin                //bltu
36                 if(op1 < op2)

```

```

37         br = 1;
38     else
39         br = 0;
40     end
41     3'b110: begin
42         if (op1 >= op2) //bgeu
43             br = 1;
44         else
45             br = 0;
46         end
47         default: br = 0;
48     endcase
49 end
50
51 endmodule

```

- 主要区别是根据inst[14:12]选择不同的br_type,然后在branch模块中根据br_type进行不同的比较来确定br，需要注意的是由于br_type种类变多，在lab5中的两位应该改为3位，并同时修改CPU模块中相应的位宽

3.2 实验结果

test

- 烧写
已在线下检查

4. 总结

更加了解了计算机体系结构，对riscv指令集的架构理解更加深入