Lab1

逻辑设计

有限状态机(sort.v)

```
case (current_state)
    HLT: current_state = HLT;
    LOAD: current_state = CX01;
    CX01: current_state = CX12;
    CX12: current_state = CX23;
    CX23: current_state = CX01s;
    CX01s: current_state = CX12s;
    CX12s: current_state = CX01t;
    CX01t: current_state = HLT;
    default:;
endcase
```

按照冒泡排序的顺序. `s`, `t`分别表示第二/三次比较

电路设计

ALU

根据组合逻辑, 选择运算方式后按照规则生成`cf`, `zf`, `of`位

sort

设置四个`register`, 通过四个`mux`选择`register`的输入, 两个`mux`选择ALU的输入

核心代码

ALU

```
always @(*) begin
    {cf, zf, of} = 3'b0;
    y = 0;
    case (m)
    ADD: begin
        {cf, y} = a + b;
        of = (a[WIDTH-1] ^~ b[WIDTH-1]) ? (a[WIDTH-1] ^ y[WIDTH-1]) : 0;
    end
    SUB: begin
        {cf, y} = a - b;
        of = (a[WIDTH-1] ^ b[WIDTH-1]) ? (a[WIDTH-1] ^ y[WIDTH-1]) : 0;
    end
    AND: y = a & b;
    OR: y = a | b;
    XOR: y = a ^ b;
    default:;
    endcase
    zf = ~|y;
end
```

`cf`位直接按照前一位的数字, `of`按照`可能出现溢出的情况?进一步一处条件:不溢出`

sort

```
always @(*) begin
        {m0, m1, m2, m3, m4, m5, en0, en1, en2, en3, done} = 15'b0;
        case (current_state)
            LOAD: begin
                m0 = 0; en0 = 1;
```

```
                m1 = 1; en1 = 1;
                m2 = 2; en2 = 1;
                m3 = 3; en3 = 1;
            end
            CX01, CX01s, CX01t: begin
                m4 = 0; m5 = 1;
                m0 = 1; m1 = 0;
                en0 = of^result[N-1]; en1 = of^result[N-1];
            end
            CX12, CX12s: begin
                m4 = 1; m5 = 2;
                m1 = 2; m2 = 1;
                en1 = of^result[N-1]; en2 = of^result[N-1];
            end
            CX23: begin
                m4 = 2; m5 = 3;
                m2 = 3; m3 = 2;
                en2 = of^result[N-1]; en3 = of^result[N-1];
            end
            HLT: done = 1;
            default:;
        endcase
end
```

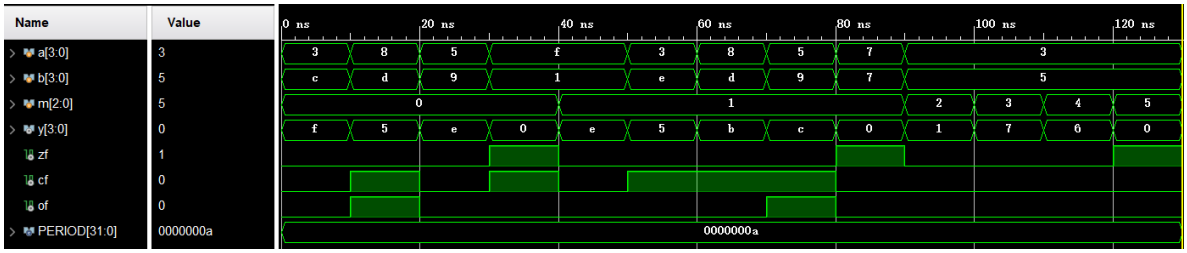
`m1`->`m4`是`register`左边的输入选择器, 以m1为例, 在直接输入端和其他寄存器中算则

> mux4_8 #(N) M1 (i1, r0, x1, r2, r3, m1);

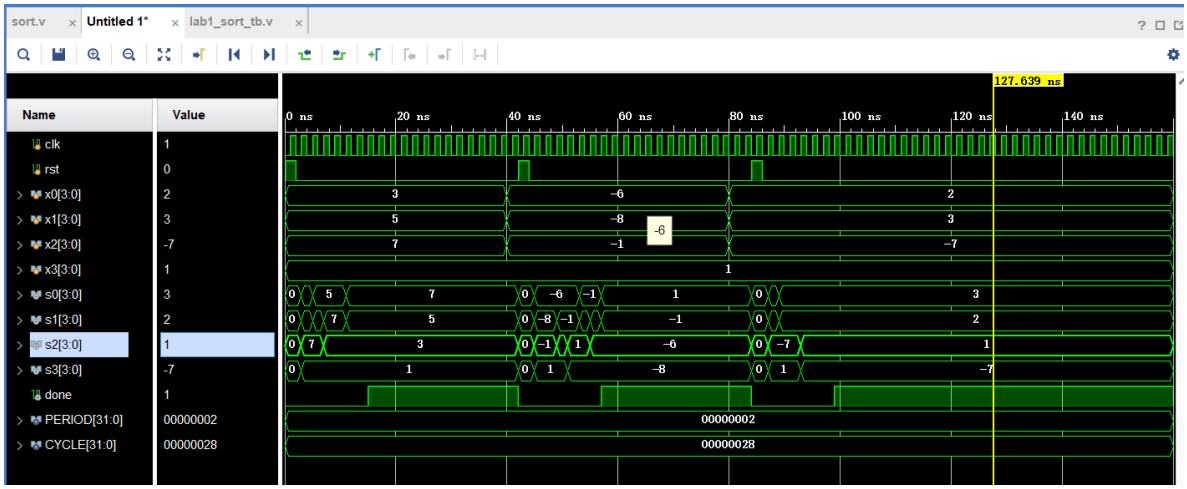
`m5`, `m6`为ALU选择运算数. 是否交换由`of`和`sf`(代码中用运算结果最高位代替)共同决定

仿真结果

ALU



sort



结果分析

ALU可得到正确运算结果, sort能完成有符号数的递增排序

实验总结

1. 每个寄存器连接的外置和接口应该尽可能一致, 以保证稳定性, 可扩展性同时方便调试
2. case应列举完所有情况
3. 复习了FSM的两段式写法以及组合逻辑/时序电路的描述

意见建议

作为第一次复习实验, 难度适中, 作业量恰当

思考题

1. 将二段式的第二段中的`en`全部取反
2. 两个ALU每个配置两个mux, 每个状态分别计算:
    1. 前两个寄存器和后两个的较大/小数
    2. 最大的数和最小的数
    3. 中间的两个数的大小比较