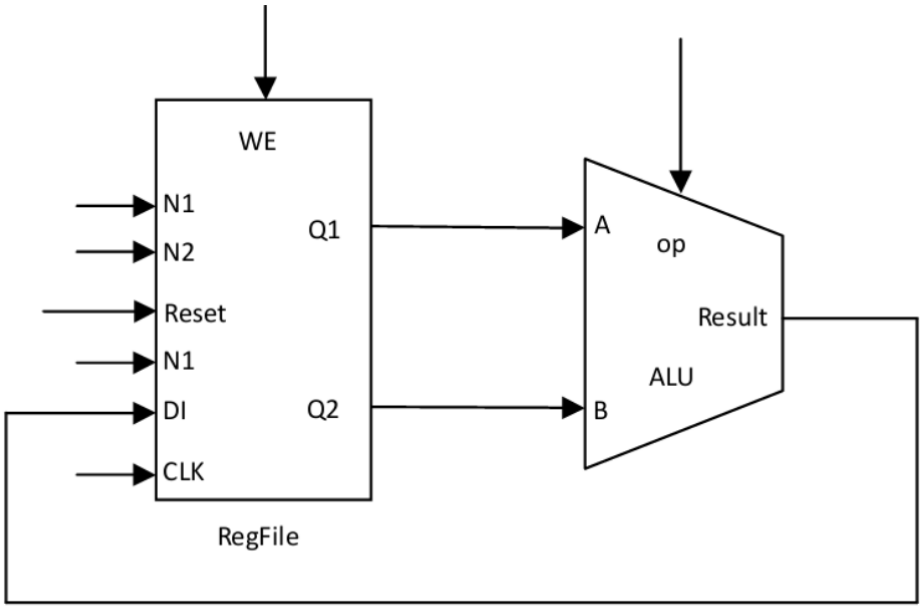
19302010011 王海伟

Lab3 REGFiles



ALU和7段数码管直接沿用上一个lab的内容，但是需要简单的修改一下，ALU的输入变成了两个8位的数据，输出为8位，其他部分大致相同，数码管部分需要两个8位输入，分别使用两个7段数码管显示，由于a2g会存在复用，所有必须在一个module里面完成显示。

对于RegFile而言，需要注意的点在于时序逻辑因为引入了大量的always块，要注意不能在不同的always块当中对相同的reg赋值，这样做就相当于在同一个reg上连接了两个输入，会导致冲突，对于同一个reg或者output的赋值需要放到同一个always块。例如：

```
always@(posedge rst or negedge clk) begin
    if(we) begin
        data[N1]<=DI;
    end

    i = 0;
    if(rst)begin
        for(i=0;i<32;i=i+1) begin
            data[i]<= i;
        end

    end
end
```

然后需要处理三个模块之间的连接

```
wire [7:0]F1;
wire [7:0]F2;
reg [7:0]F3;
wire [7:0]F4;
assign F4 = F3;

REG_File regfile(clk2, SW[13], SW[12], SW[4:0], SW[10:5], F3, F1, F2);
ALU alu(SW[15:14], F1, F2, F4);
segmentDisplay segmentDisplay1(CLK100MHZ, F1, F2, A2G, AN[3:0]);
```

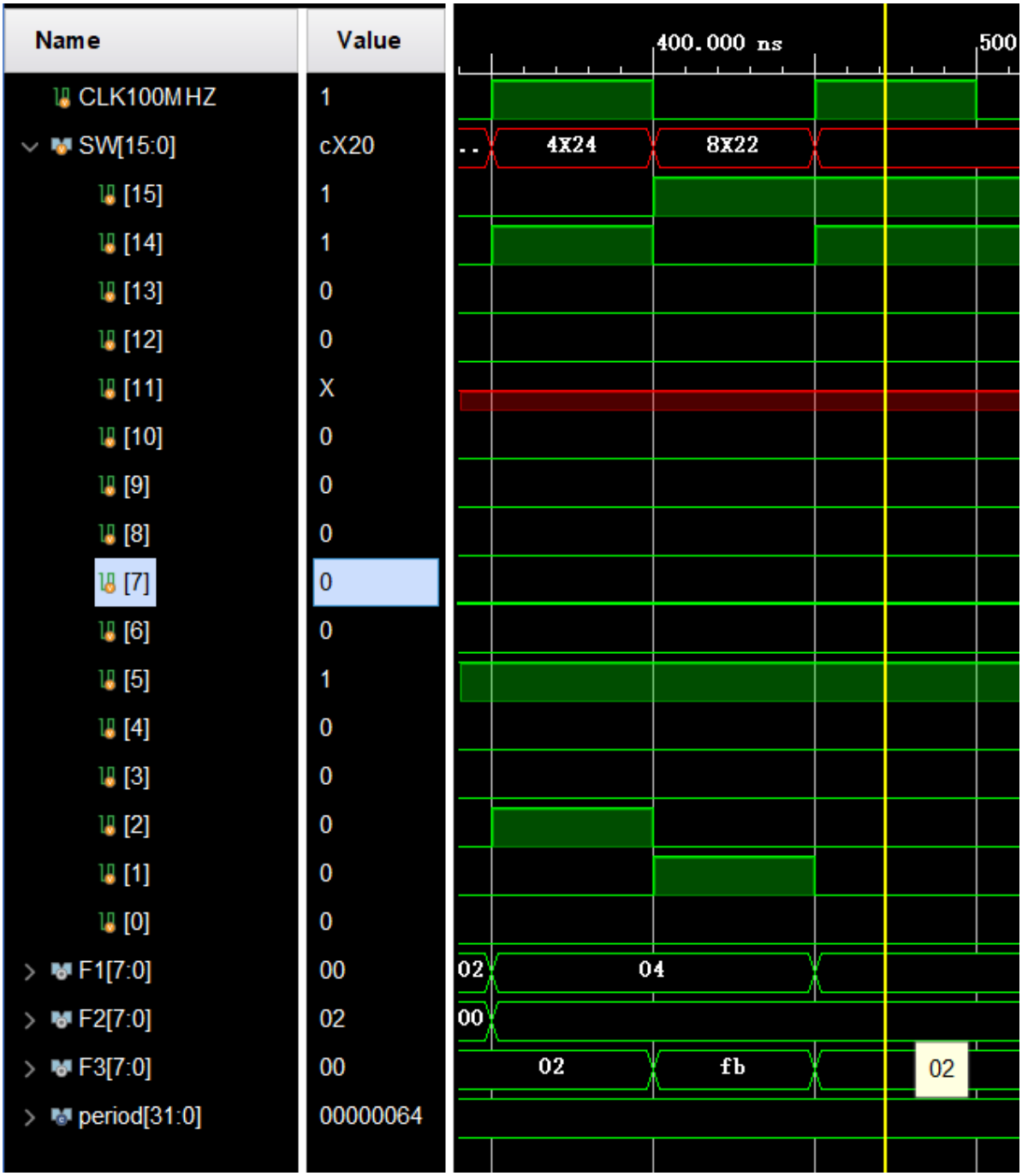
还需要进行分频处理，因为直接使用原本的时钟的话上板测试的时候看不清楚we=1是发生的情况，所以需要降低REGFiles的时钟频率

```
always@(posedge CLK100MHZ)
    begin
        count = count + 1;
        if(count > 200000000)
        begin
            count = 0;
            clk2 = ~clk2;
        end
    end
```

在仿真的时候可以省去分频，和数码管

alu加法和we=1回写N1的测试

减法，取反，乘法的测试


Name
Value
400.000 ns
500
CLK100MHZ
1
SW[15:0]
cX20
4X24
8X22
[15]
1
[14]
1
[13]
0
[12]
0
[11]
X
[10]
0
[9]
0
[8]
0
[7]
0
[6]
0
[5]
1
[4]
0
[3]
0
[2]
0
[1]
0
[0]
0
F1[7:0]
00
02
04
F2[7:0]
02
00
F3[7:0]
00
02
fb
02
period[31:0]
00000064