Lab1

实验目的与内容

1. 掌握ALU的逻辑运算功能
2. 掌握数据通路和有限状态机的设计方法
3. 掌握组合电路和时序电路，以及参数化、结构化的Verilog 描述方法

逻辑设计

ALU单元

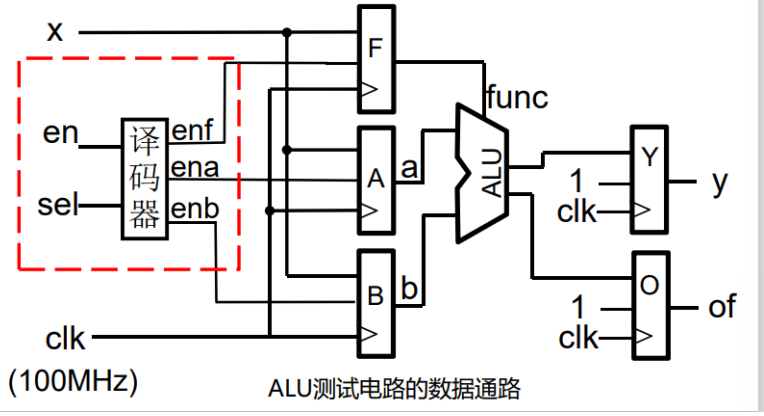
- 在这一单元中，我们需要完成ALU的设计，该设计非常简单明了，在一个case中针对不同的function进行不同的计算即可，唯一需要注意的是无符号数的比较，我们采用如下代码进行无符号数的比较：

```
y_reg = a + b;
of_reg = (a[WIDTH - 1] == 1'b1 && b[WIDTH - 1] == 1'b1 && y_reg[WIDTH - 1] ==
1'b0)||
         (a[WIDTH - 1] == 1'b0 && b[WIDTH - 1] == 1'b0 && y_reg[WIDTH - 1] ==
1'b1);
```

- 上述的表达式来自上学期模拟与数字电路课本的结论，在此就不过多阐述
- 我们接着设计ALU_test模块，这个模块我们增加译码器单元，并且按照PPT上所述的内容搭建完整的ALU_test模块

译码器真值表

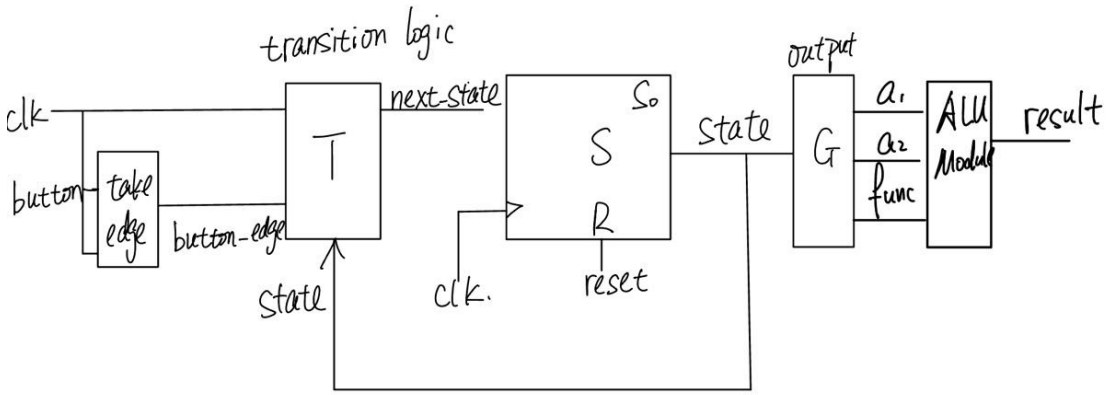
en	sel	ena	enb	enf
1	00	1	0	0
1	01	0	1	0
1	10	0	0	1
1	11	0	0	0
0	xx	0	0	0



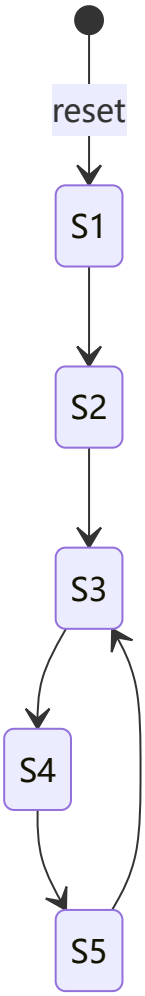
ALU测试电路的数据通路

FLS模块

- 该模块中我们首先画出数据通路



- 我们绘制出他的状态转换图



- 我们采用S1和S2两个状态读取初始的d1和d2，并且用后三个状态决定加数和结果，并且将他们输出到结果当中
- 我们依照上述思路，写出三段式状态机，其中第一段为

```
// 第一段
always@(posedge clk)
    if(rst)
        state <= S1;
    else
        state <= next_state;
```

- 第二段代码描述状态转移，在这一状态中，我们的状态分配按照上述画出的状态机描述

```
// 第二段
always@(*)
if(rst)
    next_state = S1;
else if(en1)
    case(state)
        S1: next_state = S2;
        S2: next_state = S3;
        S3: next_state = S4;
        S4: next_state = S5;
        S5: next_state = S3;
        default: next_state = S1;
    endcase
else
    next_state = state;
```

- 第三段代码我们设计在不同的状态中的输出

```
// 第三段
always@(posedge clk)
case(state)
    S1:
    begin
        d_1 <= d;
        f_reg <= d_1;
    end
    S2:
    begin
        d_2 <= d;
        f_reg <= d_2;
    end
    S3:
    begin
        func <= d[3:0];
        c_1 <= d_1;
        c_2 <= d_2;
        d_3 <= c_3;
        f_reg <= c_3;
    end
    S4:
    begin
```

```
        func <= d[3:0];
        c_1 <= d_2;
        c_2 <= d_3;
        d_1 <= c_3;
        f_reg <= c_3;
    end
    S5:
    begin
        func <= d[3:0];
        c_1 <= d_3;
        c_2 <= d_1;
        d_2 <= c_3;
        f_reg <= c_3;
    end
    default: f_reg <= 7'b0;
endcase
assign f = f_reg;
```

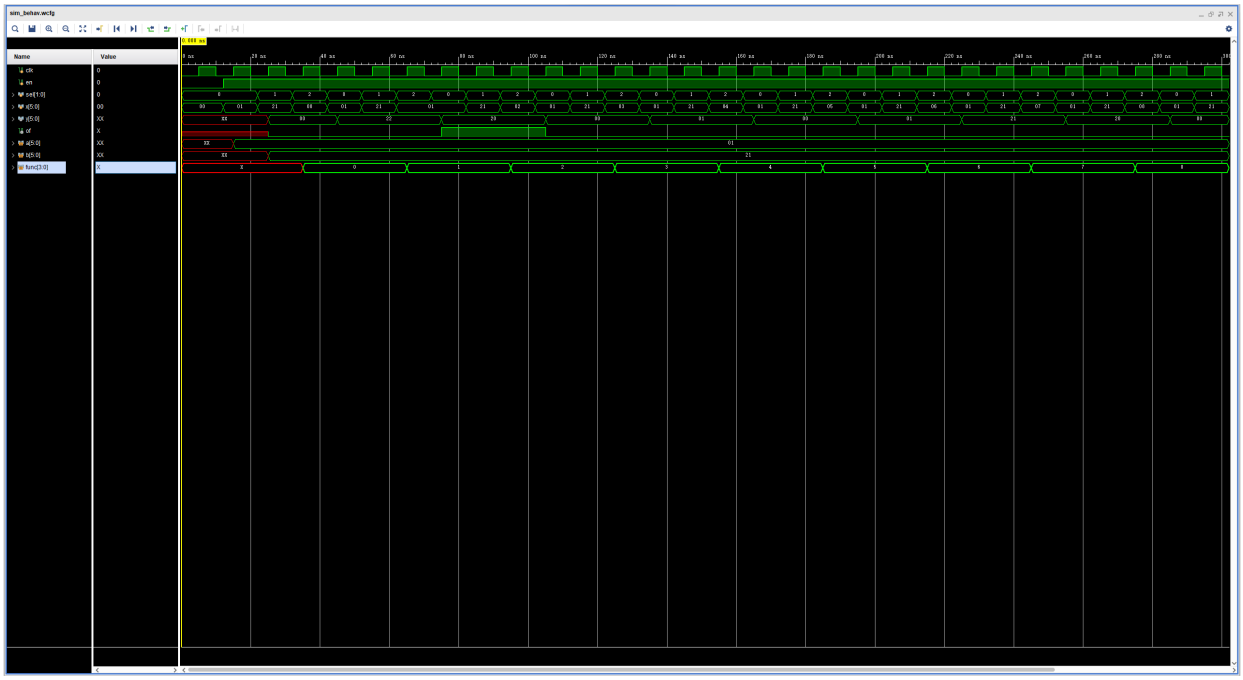
- 最后，我们例化ALU模块。使得程序正常运行

```
alu alu1(
    .a(c_1),
    .b(c_2),
    .func(func),
    .y(c_3),
    .of(of)
);
```

功能仿真

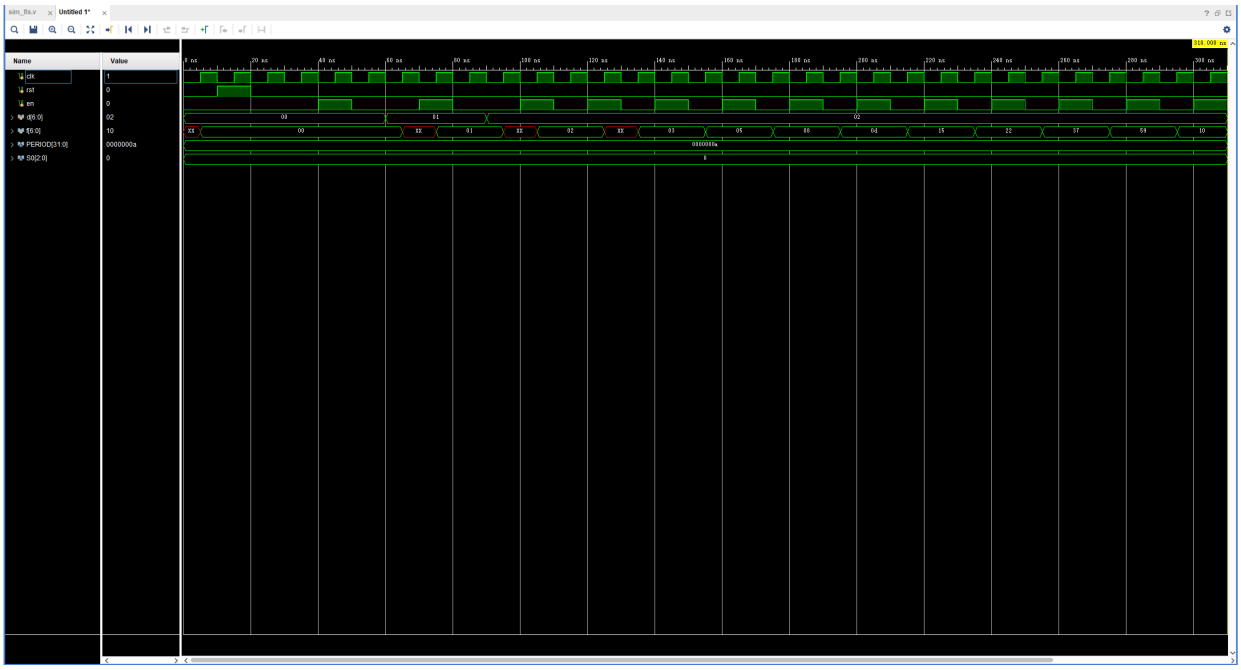
ALU

在本模块的仿真中，我们测试了各个`function`的正确性，我们可以看到在减法模式中出现了溢出，of成功置为1，并且我们完成了所有的选作部分的测试样例



FLS

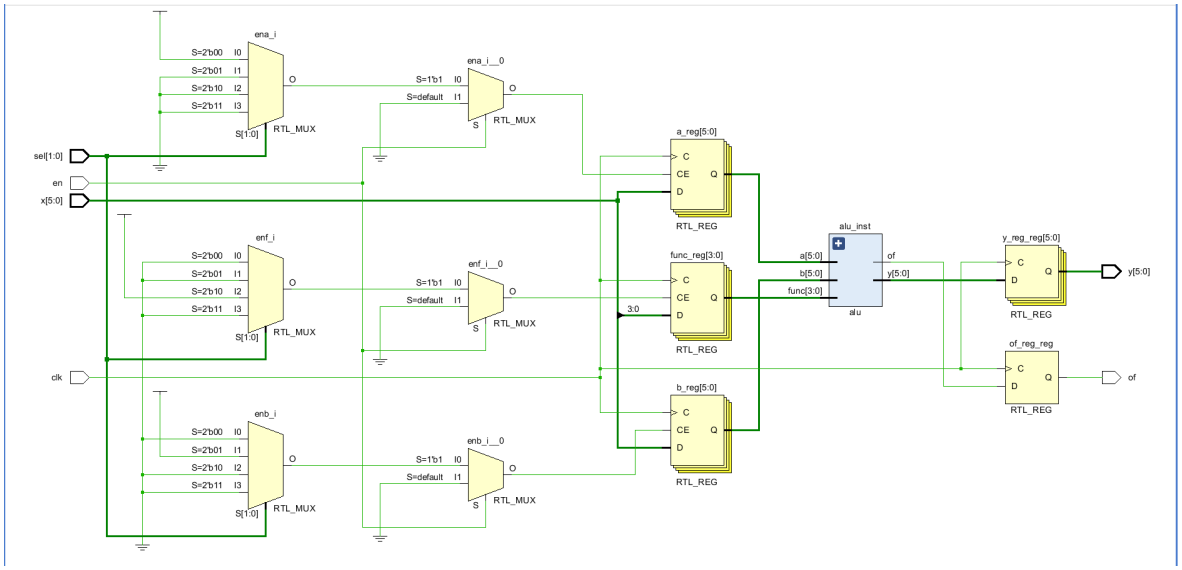
我们完成FLS的模块仿真，我们首先读取前两个数值内容，然后根据我们的`func`，设置不同的输出



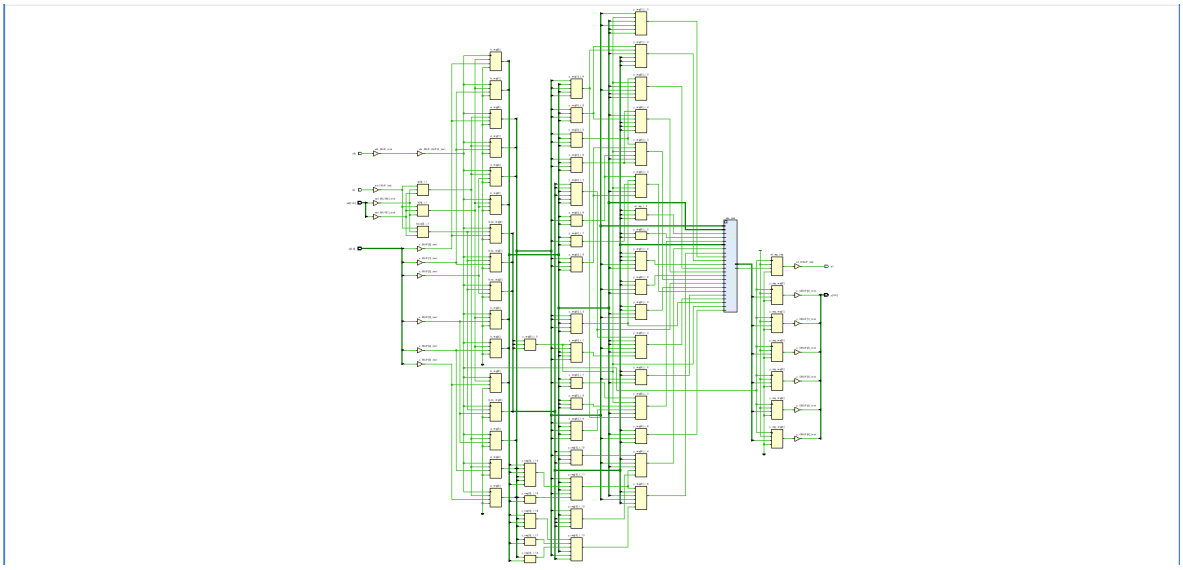
电路设计与分析

ALU

1. 我们首先绘制出ALU的RTL图

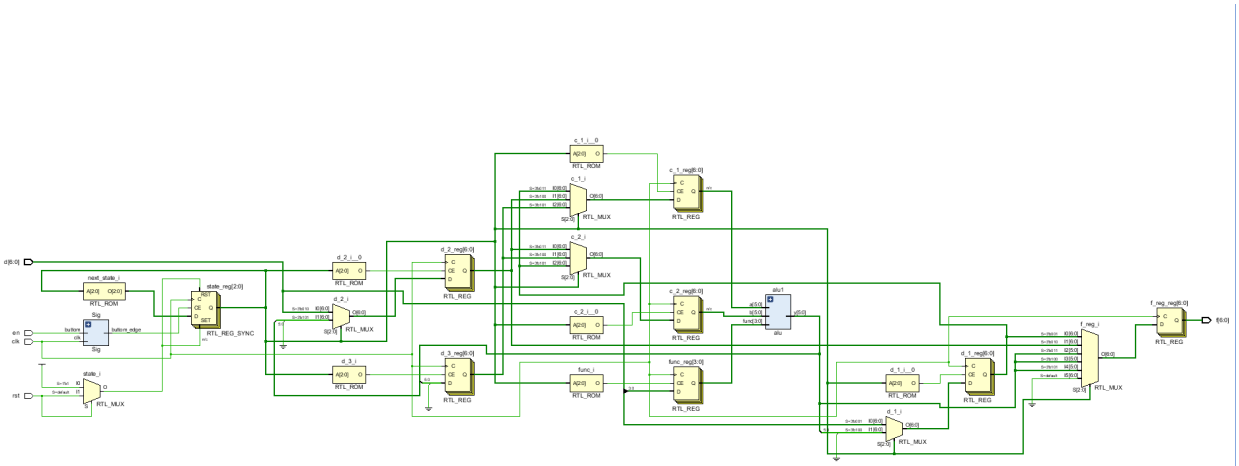


2. 接着我们绘制出综合后的电路图



3. RTL电路图是将HDL语言转换为相关的逻辑电路图，而综合后的电路图则是将RTL代码转换为门级网表，即将RTL代码转换为门级电路。在综合后，电路已经被映射到器件的LUT和FF中，并且经过了优化。

FLS



总结

1. 本次实验复习了Verilog相关语法
2. 本次实验学会了RTL电路图与综合后的电路图的区别