模拟与数字电路实验报告

1. 实验题目： 实验 08 信号处理及有限状态机

2. 实验目的： 进一步熟悉 FPGA 开发的整体流程，掌握几种常见的信号处理技巧，掌握有限状态机的设计方法，能够使用有限状态机设计功能电路

3. 实验平台： Vivado 及 Logisim 软件；

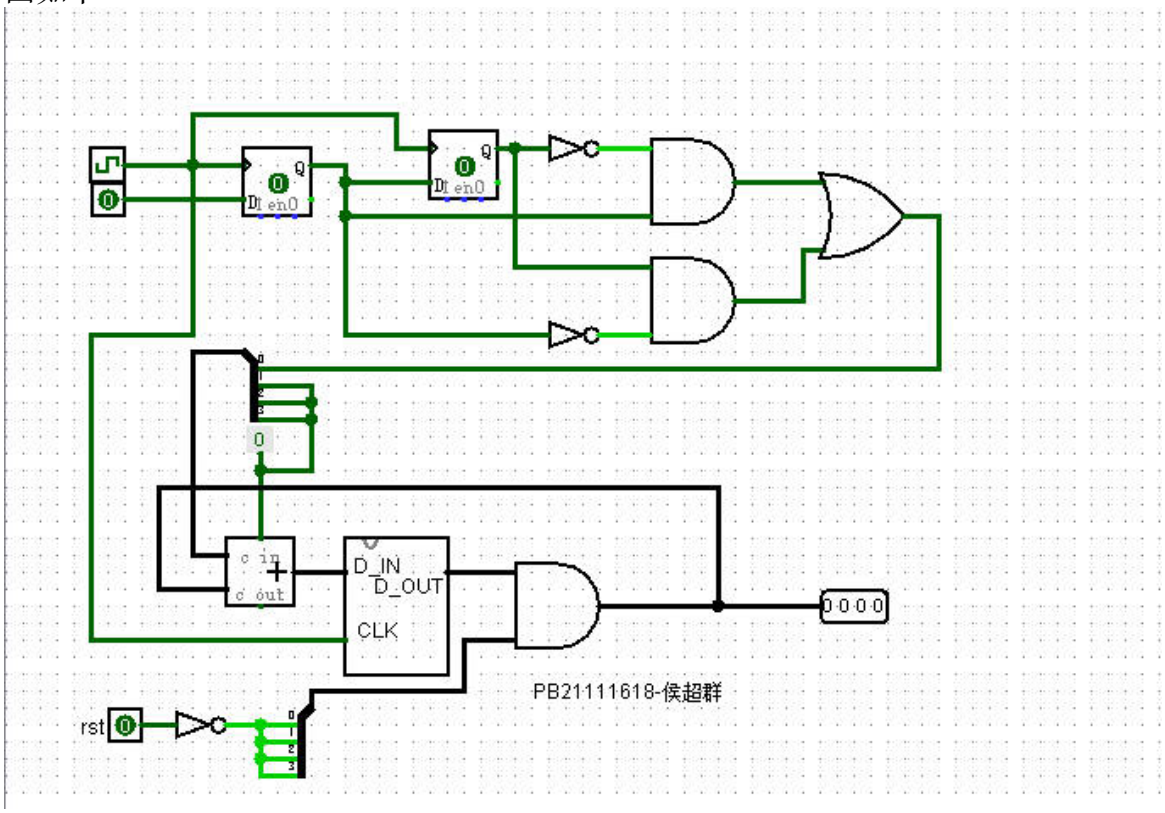
4. 实验练习：

1). 问题一： 将实验手册中的代码改成三段式有限状态机形式如下：

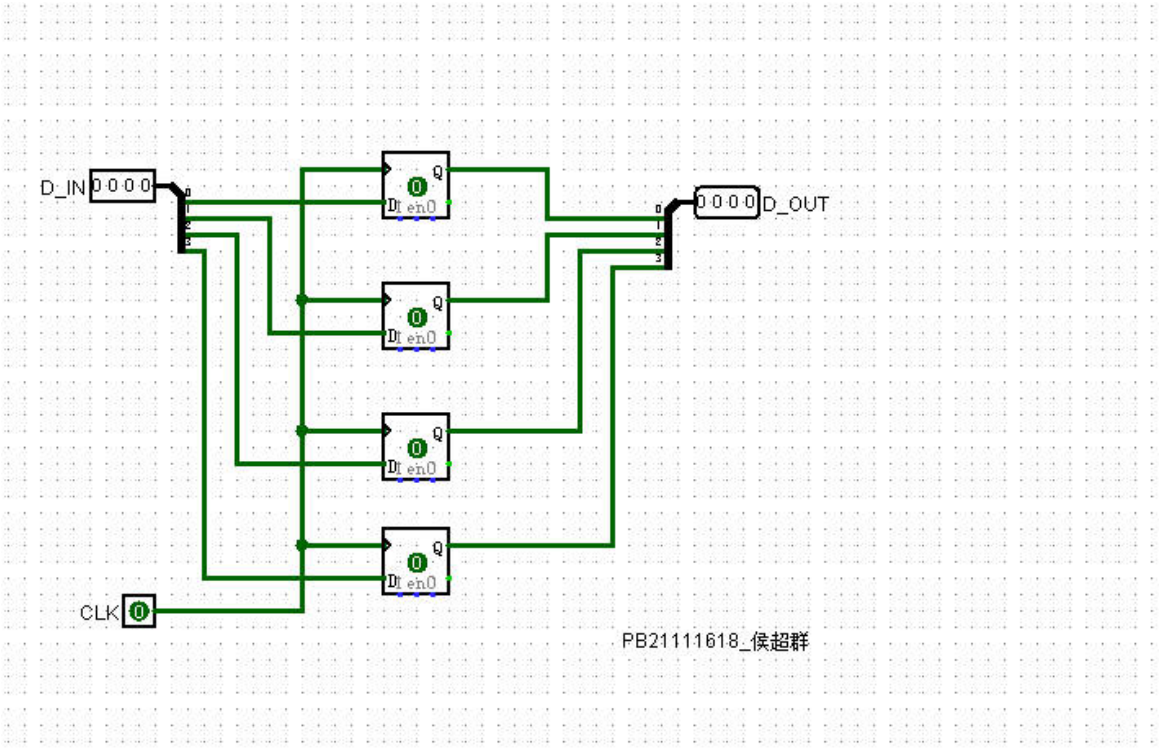
```
module problem1(
input clk,
input rst,
output led
);
parameter C_0 = 2'b00;
parameter C_1 = 2'b01;
parameter C_2 = 2'b10;
parameter C_3 = 2'b11;
reg [1:0] curr_state;
reg [1:0] next_state;
//第一部分
always@(*)
begin
    case(curr_state)
        C_0 : next_state = C_1;
        C_1 : next_state = C_2;
        C_2 : next_state = C_3;
        C_3 : next_state = C_0;
    endcase
end
//第二部分
always@(posedge clk or posedge rst)
begin
    if(rst)
        curr_state <= C_0;
    else
        curr_state <= next_state;
```

```
29 end
30 //第三部分
31 assign led = (curr_state == 2'b11) ? 1'b1 : 1'b0;
32 endmodule
```

2). 问题二： 在 logisim 中设计 4bit 位宽的计数器电路，由于根据 sw 电平变化进行计数，需要对信号进行取边沿，对实验手册中的设计进行稍加修改，使其在电平翻转均能产生脉冲；截图如下：



其中调用 4bit 寄存器模块，如下：

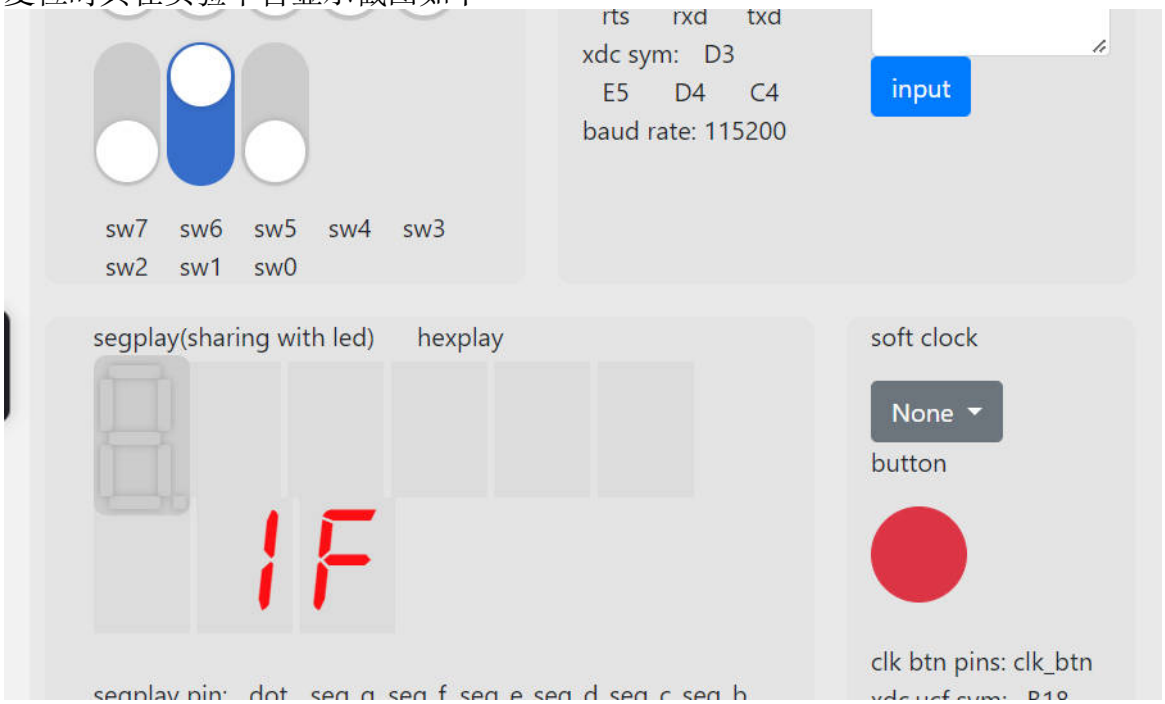


3). 问题三： 设计 8 位十六进制计数器，主要对于开关按钮进行取边沿操作，使其产生计数，具体代码如下：

```
module problem3(
input clk, button, rst, sw,
output reg [3:0]D,
output reg [2:0]AN
);
//一个时钟周期
reg button_r1, button_r2;
wire button_edge;
always@(posedge clk)
    button_r1 <= button;
always@(posedge clk)
    button_r2 <= button_r1;
assign button_edge = button_r1 & (~button_r2);
//扫描
reg [15:0]cnt;
reg [7:0]led;
always@(posedge clk)
begin
    if(cnt >= 10000)
```

```
20          cnt <= 0;
21      else
22          cnt <= cnt + 1;
23  end
24
25  always@(posedge clk)
26  begin
27      if(cnt == 0)
28      begin
29          if(AN == 3'b000)
30              AN = 3'b001;
31          else
32              AN = 3'b000;
33      end
34  end
35
36  always@(posedge clk)
37  begin
38      if(AN == 3'b000)
39          D <= led[3:0];
40      else
41          D <= led[7:4];
42  end
43  //计数
44  always@(posedge clk)
45  begin
46      if(rst == 1)
47          led = 8'b00011111;
48      if(button_edge)
49      begin
50          if(sw)
51              led = led + 8'b00000001;
52          else
53              led = led - 8'b00000001;
54      end
55  end
56  endmodule
```

复位时其在实验平台显示截图如下：



4). 问题四： 使用有限状态机设计该序列监测电路，共 5 个状态，可将其化简为 4 种。并通过对开关信号的取边沿判断有限状态机的下一步走向，最后赋给相关管脚；
代码如下：

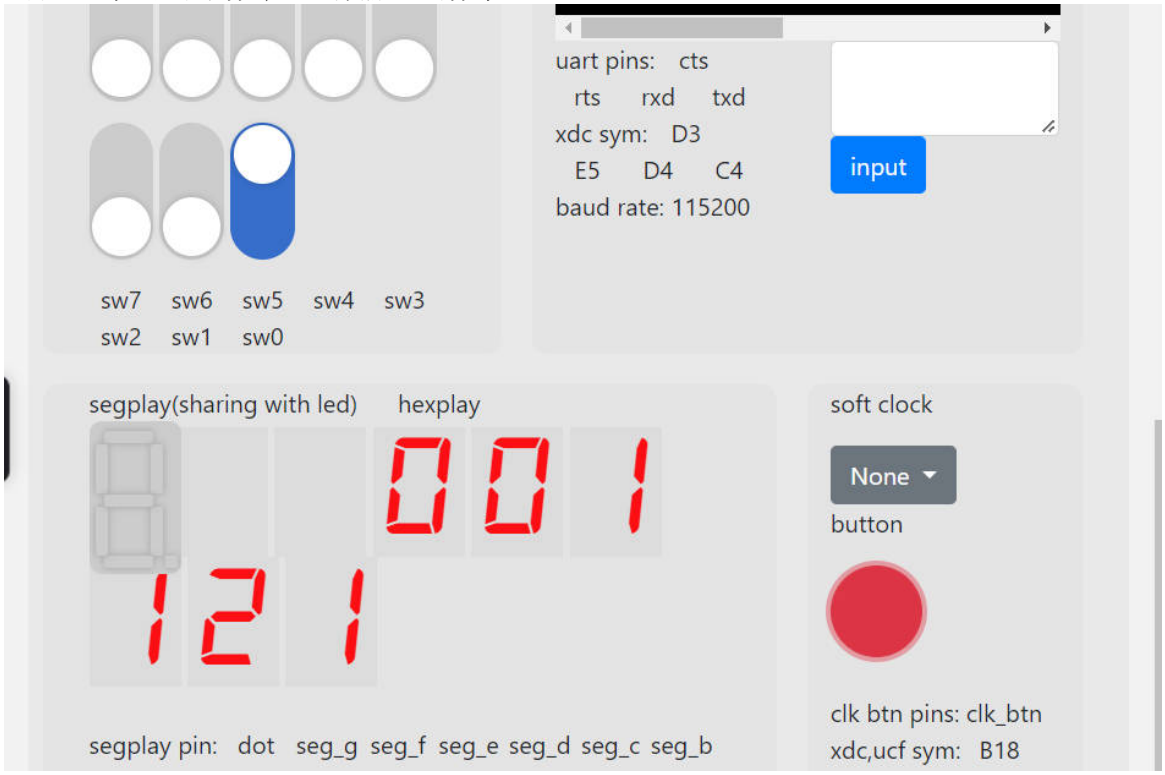
```
module problem4(
    input sw, clk, button,
    output reg [3:0]D,
    output reg [2:0]AN
        );
    reg [3:0]count;
    reg [15:0]num;
    //一个时钟周期
    reg button_r1, button_r2;
    wire button_edge;
    always@(posedge clk)
        button_r1 <= button;
    always@(posedge clk)
        button_r2 <= button_r1;
    assign button_edge = button_r1 & (~button_r2);
    //扫描
    reg [15:0]cnt;
    always@(posedge clk)
```

```
19      begin
20          if(cnt >= 2500)
21              cnt <= 0;
22          else
23              cnt <= cnt + 1;
24      end
25      always@(posedge clk)
26      begin
27          if(cnt == 0)
28          begin
29              if(AN == 3'b101)
30                  AN <= 3'b000;
31              else
32                  AN <= AN + 3'b001;
33          end
34      end
35      always@(posedge clk)
36      begin
37          if(AN == 3'b000)
38              D <= num[3:0];
39          else if(AN == 3'b001)
40              D <= count;
41          else if(AN == 3'b010)
42              D <= num[3:0];
43          else if(AN == 3'b011)
44              D <= num[7:4];
45          else if(AN == 3'b100)
46              D <= num[11:8];
47          else
48              D <= num[15:12];
49      end
50      //状态机
51      parameter C_0 = 2'b00;
52      parameter C_1 = 2'b01;
53      parameter C_11 = 2'b10;
54      parameter C_110 = 2'b11;
55      reg [1:0] curr_state;
```

```
56      reg [1:0] next_state;
57      always@(*)
58      begin
59          if(sw)
60          begin
61              case(curr_state)
62                  C_0: next_state = C_1;
63                  C_1: next_state = C_11;
64                  C_11: next_state = C_11;
65                  C_110: next_state = C_1;
66                  default: next_state = C_0;
67              endcase
68          end
69          else
70          begin
71              case(curr_state)
72                  C_0: next_state = C_0;
73                  C_1: next_state = C_0;
74                  C_11: next_state = C_110;
75                  C_110: next_state = C_0;
76                  default: next_state = C_0;
77              endcase
78          end
79      end
80
81      always@(posedge clk)
82      begin
83          if(button_edge)
84          begin
85              if(curr_state == C_110 && next_state == C_0)
86                  count = count + 1;
87              curr_state <= next_state;
88              //根据状态变化
89              num[15:12] <= num[11:8];
90              num[11:8] <= num[7:4];
91              num[7:4] <= num[3:0];
92              num[3:0] <= {3'b000,sw};
```

```
93          end
94      end
95
96      endmodule
```

当输入 0011001110011 时其在实验平台显示截图如下：分别显示最近输入的 4 个数值，检测到目标序列的个数，当前输入的数值



5. 总结与思考：

1). 收获：通过本次实验对于 FPGA 有了相当了解，同时对于 Vivado 软件的使用了有了更为深入的了解，学会了信号处理以及有限状态机的使用；

2). 评价：实验内容由于有前面实验的铺垫相对而言不是较难，设置合理；

3). 建议：实验内容设置合理，无较大建议；