

数电实验8

实验题目

- 信号处理与有限状态机

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

实验环境

- VLAB: vlab.ustc.eud.cn
- FPGAOL: fpgaol.ustc.edu.cn
- Logisim、Vivado

实验步骤

Step1.信号整形及去毛刺

Step2.取信号边沿技巧

Step3. 有限状态机介绍

Step4.有限状态机 Verilog 实现

T1

修改为三部分有限状态机

```
1 module test(input clk,rst, output led);
2   parameter a = 2'b00;
3   parameter b = 2'b01;
4   parameter c = 2'b10;
5   parameter d = 2'b11;
6   reg [1:0] cnt;
7   reg [1:0] next_cnt;
8   //有限状态机第一部分
9   always@(*)
10     begin
11       case(cnt)
12         a:next_cnt <= 2'b01;
13         b:next_cnt <= 2'b10;
14         c:next_cnt <= 2'b11;
```

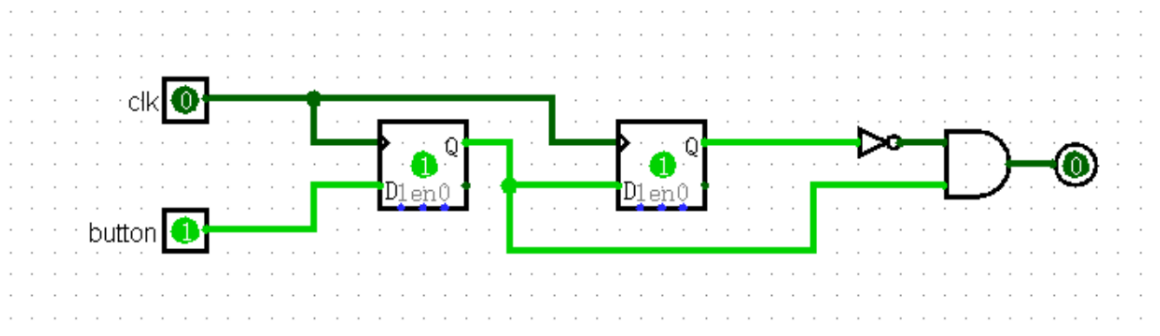
```

15         d:next_cnt <= 2'b00;
16     endcase
17 end
18 //有限状态机第二部分
19 always@(posedge clk or posedge rst)
20 begin
21     if(rst)
22         cnt <= 2'b00;
23     else
24         cnt<=next_cnt;
25     end
26 //有限状态机第三部分
27     assign led = (cnt==2'b11) ? 1'b1 : 1'b0;
28 endmodule

```

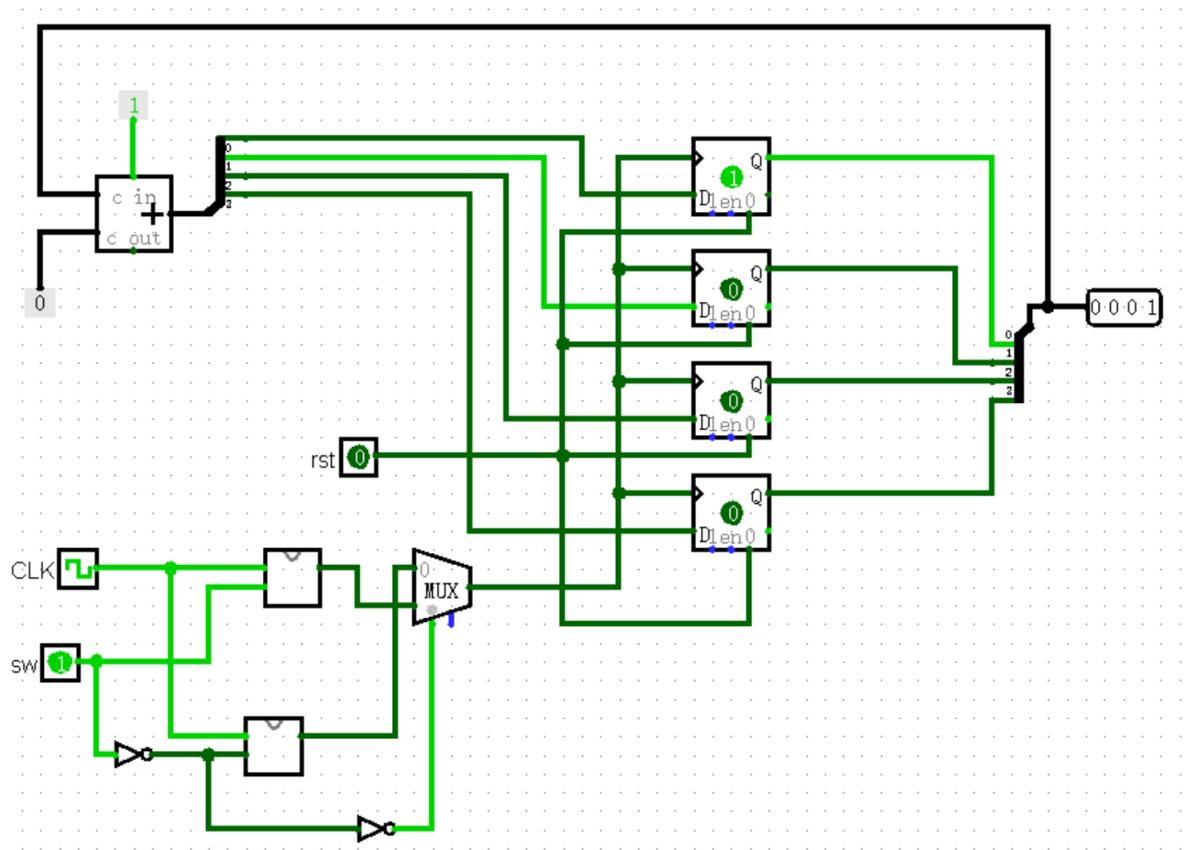
T2

实例化模块生成脉冲信号电路：



计数器电路：

通过四个D触发器和上述例化的模块完成计数，sw上升或下降就在右侧寄存器完成计数，rst表示置位信号，rst=1则寄存器置0。



T3

信号整形去毛刺:

```

1  module jitter_clr(
2  input clk,
3  input button,
4  output button_clean);
5      reg [9 :0] cnt;
6  always@(posedge clk) begin
7      if(button==1'b0) cnt <= 10'b00000_00000;
8      else if(cnt < 512) cnt <= cnt + 10'b00000_00001;
9  end
10 assign button_clean = cnt[9];
11 endmodule

```

取信号边沿:

```

1  module signal_edge(//取信号边沿
2  input clk,
3  input button,
4  output button_redge);
5  reg button_r1,button_r2;
6  always@(posedge clk)
7      button_r1 <= button;
8  always@(posedge clk)
9      button_r2 <= button_r1;
10 assign button_redge = button_r1 & (~button_r2);
11 endmodule

```

设计文件:

```
1 module Counter(  
2   input CLK,rst,button,sw,  
3   output reg [3:0] d,  
4   output reg an  
5 );  
6 wire add_cleant, sub_cleant, add_clean, sub_clean;  
7 reg [7:0] counter;  
8 reg [2:0] cnt;  
9  
10 jitter_clr j1(CLK, button, add_cleant);  
11 signal_edge e1(CLK, add_cleant, add_clean);  
12  
13 always@(posedge CLK)  
14   begin  
15     if(rst)  
16       counter<=8'b0001_1111;  
17     else  
18       begin  
19         if(sw==1)  
20           begin  
21             if(add_clean)  
22               counter <= counter + 8'b0000_0001;  
23           end  
24         else  
25           begin  
26             if(add_clean)  
27               counter <= counter - 8'b0000_0001;  
28           end  
29       end  
30   end  
31 always@(posedge CLK)  
32   begin  
33     cnt=cnt+1;  
34     an=cnt[2];  
35     if(an) d<=counter[7:4]; //an不等于0, 表示cnt[2]不为0, 表示的就是前四个数码管  
36     else d<=counter[3:0]; //后四个数码管  
37   end  
38 endmodule
```

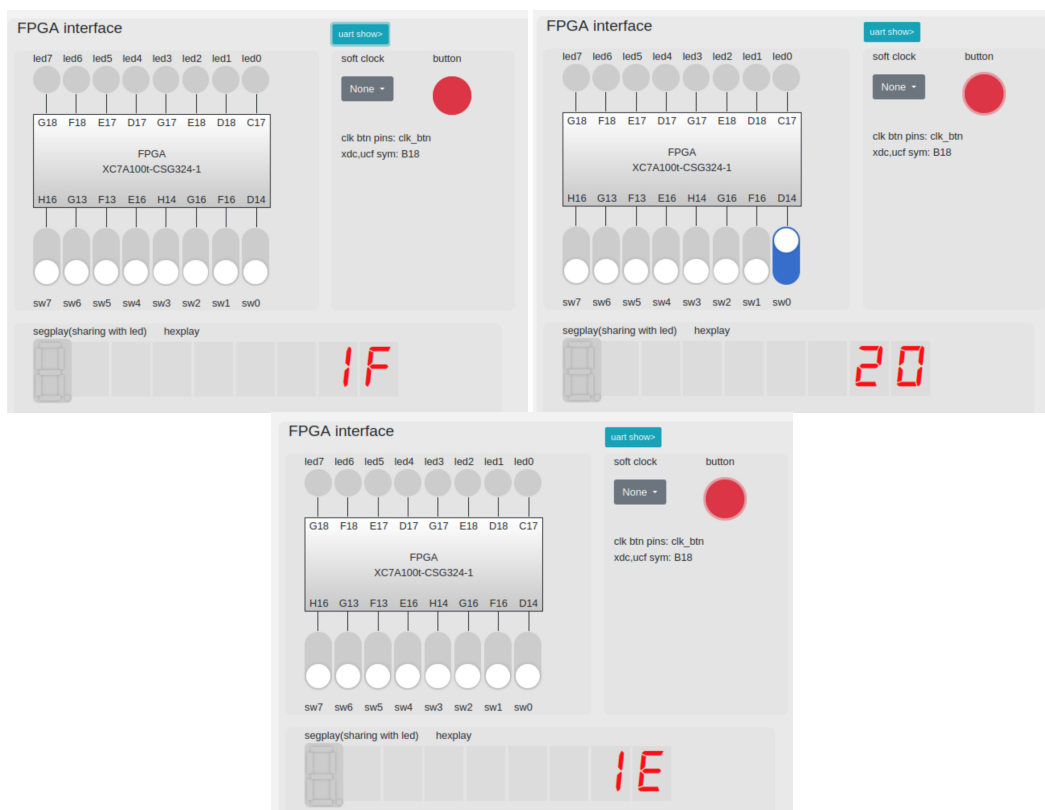
约束文件:

```

1 set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports { CLK
  }];
2
3 set_property -dict { PACKAGE_PIN D14     IOSTANDARD LVCMOS33 } [get_ports { sw
  }];
4 set_property -dict { PACKAGE_PIN F16     IOSTANDARD LVCMOS33 } [get_ports {
  rst }];
5
6 set_property -dict { PACKAGE_PIN A14     IOSTANDARD LVCMOS33 } [get_ports {
  d[0] }];
7 set_property -dict { PACKAGE_PIN A13     IOSTANDARD LVCMOS33 } [get_ports {
  d[1] }];
8 set_property -dict { PACKAGE_PIN A16     IOSTANDARD LVCMOS33 } [get_ports {
  d[2] }];
9 set_property -dict { PACKAGE_PIN A15     IOSTANDARD LVCMOS33 } [get_ports {
  d[3] }];
10 set_property -dict { PACKAGE_PIN B17     IOSTANDARD LVCMOS33 } [get_ports { an
  }];
11
12 set_property -dict { PACKAGE_PIN B18     IOSTANDARD LVCMOS33 } [get_ports {
  button }];

```

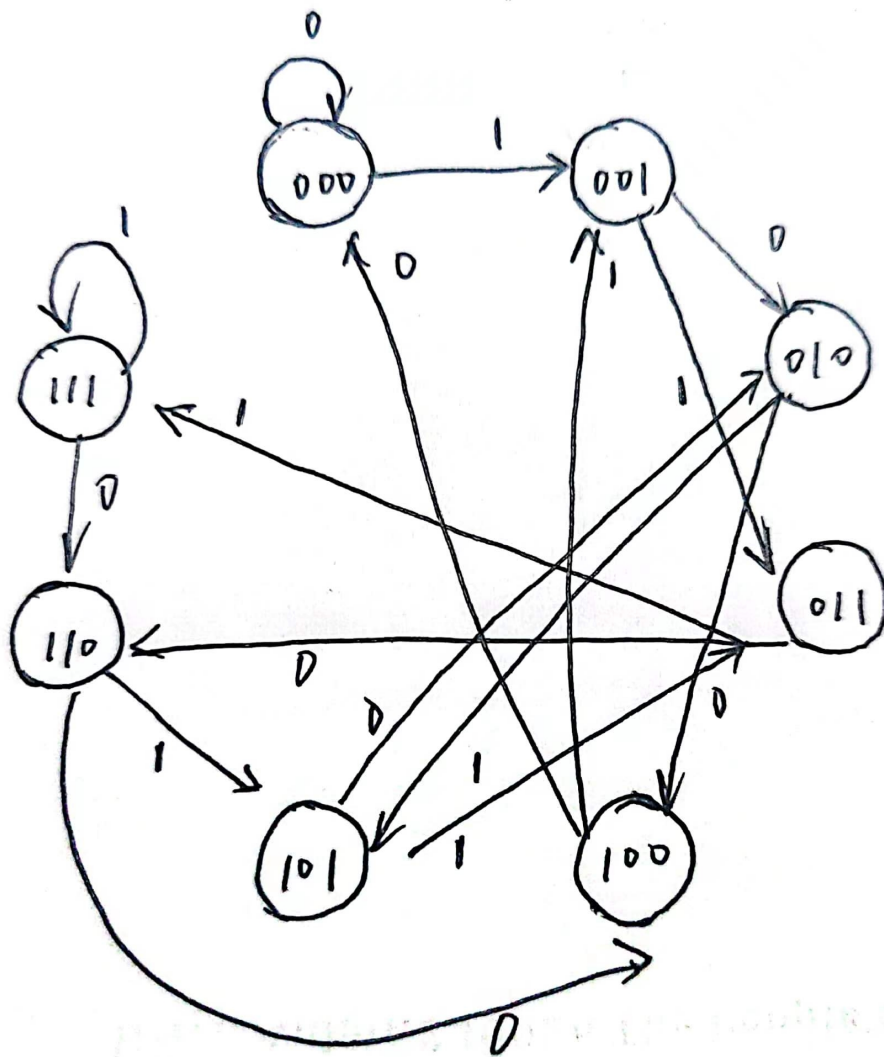
烧写显示结果



当sw[0]为0时递减，当sw[0]为1时递增。

T4

状态图：



信号整形去毛刺:

```

1 module jitter_clr(
2   input clk,
3   input button,
4   output button_clean);
5   reg [9:0] cnt;
6   always@(posedge clk)
7     begin
8       if(button==1'b0) cnt <= 10'b000000_00000;
9       else if(cnt < 512) cnt <= cnt + 10'b000000_00001;
10    end
11   assign button_clean = cnt[9];
12 endmodule

```

取信号边沿:

```

1 module signal_edge(
2   input clk,
3   input button,
4   output button_redge);
5   reg button_r1,button_r2;
6   always@(posedge clk)
7     button_r1<= button;
8   always@(posedge clk)
9     button_r2<= button_r1;
10  assign button_redge = button_r1 &(~button_r2);
11 endmodule

```

约束文件:

```

1 set_property -dict { PACKAGE_PIN E3      IOSTANDARD LVCMOS33 } [get_ports { CLK
2   }];
3 set_property -dict { PACKAGE_PIN A14     IOSTANDARD LVCMOS33 } [get_ports {
4   d[0] }];
5 set_property -dict { PACKAGE_PIN A13     IOSTANDARD LVCMOS33 } [get_ports {
6   d[1] }];
7 set_property -dict { PACKAGE_PIN A16     IOSTANDARD LVCMOS33 } [get_ports {
8   d[2] }];
9 set_property -dict { PACKAGE_PIN A15     IOSTANDARD LVCMOS33 } [get_ports {
10  d[3] }];
11 set_property -dict { PACKAGE_PIN B17     IOSTANDARD LVCMOS33 } [get_ports {
12  an[0] }];
13 set_property -dict { PACKAGE_PIN B16     IOSTANDARD LVCMOS33 } [get_ports {
14  an[1] }];
15 set_property -dict { PACKAGE_PIN A18     IOSTANDARD LVCMOS33 } [get_ports {
16  an[2] }];
17
18 set_property -dict { PACKAGE_PIN B18     IOSTANDARD LVCMOS33 } [get_ports {
19  BTN }];
20
21 set_property -dict { PACKAGE_PIN D14     IOSTANDARD LVCMOS33 } [get_ports {
22  SW }];
23
24 set_property -dict { PACKAGE_PIN F16     IOSTANDARD LVCMOS33 } [get_ports {
25  rst }];

```

设计文件;

```

1 module T4(
2   input SW,
3   input CLK,
4   input BTN,
5   input rst,
6   output reg [3:0] d,
7   output reg [2:0] an);
8
9   parameter [2:0] s0=3'b000,s1=3'b001, s2=3'b010;
10  parameter [2:0] s3=3'b011,s4=3'b100,s5=3'b101,s6=3'b110,s7=3'b111;
11  reg [2:0] state,next_state,l_state;

```

```

12 reg [3:0] counter=4'b0000;
13 reg [6:0] cnt;
14 wire BTnt,BTN_edge;
15
16 jitter_clr c1(CLK,BTN,BTnt);
17 signal_edge e1(CLK,BTnt,BTN_edge);
18
19 always@(posedge CLK)
20 begin
21     if(rst)
22     begin
23         l_state <= 0;
24         state <= s0;
25         counter<=0;
26     end
27     else if(BTN_edge)
28     begin
29         l_state<=state;
30         state <=next_state;
31         if(next_state==s4)
32             if(state == s6)
33                 counter=counter+4'b0001;
34     end
35 end
36
37 always@(*) begin
38     case(state)
39     s0:
40         begin
41             if(!SW)//sw=0,000->000,s0->s0
42             begin
43                 if(BTN_edge) next_state = s0;
44             end
45             else //sw=1,000->001,s0->s1
46             begin
47                 if(BTN_edge) next_state = s1;
48             end
49         end
50     s1:
51         begin
52             if(!SW) //001->010
53             begin
54                 if(BTN_edge)
55                     next_state = s2;
56             end
57             else
58             begin //001->011
59                 if(BTN_edge) next_state = s3;
60             end
61         end
62     s2:
63         begin
64             if(!SW)
65             begin//010->100
66                 if(BTN_edge)

```



```

67         next_state = s4;
68     end
69     else
70         begin //010->101
71             if(BTN_edge) next_state = s5;
72         end
73     end
74 s3:
75     begin
76         if(!SW)//011->110
77             begin
78                 if(BTN_edge) next_state = s6;
79             end
80         else
81             begin //011->111
82                 if(BTN_edge) next_state = s7;
83             end
84         end
85 s4:
86     begin
87         if(!SW) //100->000
88             begin
89                 if(BTN_edge) next_state = s0;
90             end
91         else //100->001
92             begin
93                 if(BTN_edge) next_state = s1;
94             end
95         end
96 s5:
97     begin
98         if(!SW) //101->010
99             begin
100                 if(BTN_edge)
101                     next_state=s2;
102                 end
103             else
104                 begin //101->011
105                     if(BTN_edge) next_state =s3;
106                 end
107             end
108 s6:
109     begin
110         if(!SW)
111             begin //110->100
112                 if(BTN_edge)
113                     next_state=s4;
114                 end
115             else //110->101
116                 begin
117                     if(BTN_edge) next_state=s5;
118                 end
119             end
120 s7:
121     begin

```

```

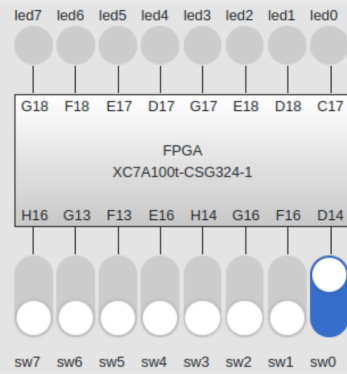
122         if(!SW) //111->110
123             begin
124                 if(BTN_edge) next_state = s6;
125             end
126         else
127             begin //111->111
128                 if(BTN_edge) next_state=s7;
129             end
130         end
131     default:
132         begin
133             if(!SW)
134                 begin
135                     if(BTN_edge) next_state = s2;
136                 end
137             else
138                 begin
139                     if(BTN_edge) next_state=s1;
140                 end
141             end
142         endcase
143     end
144
145     always@(posedge CLK)
146     begin
147         cnt=cnt+1;
148         an<=cnt[6:4];
149         if(an==3'b000) d<={3'b000,state[0]};
150         if(an==3'b001) d<={3'b000,state[1]};
151         if(an==3'b010) d<={3'b000,state[2]};
152         if(an==3'b011) d<={3'b000,l_state[2]};
153         if(an==3'b100) d<=counter;
154         if(an==3'b101) d<={1'b0, state};
155         if(an==3'b110) d<=0;
156         if(an==3'b111) d<=0;
157     end
158 endmodule

```

实验结果显示

代码共有8个状态，后四位表示当前出现的序列，序列的后三位的二进制数转化为10进制后，对应的是相应的状态，比如后三位是011代表状态s3

FPGA interface



uart show>

soft clock

button

None ▾

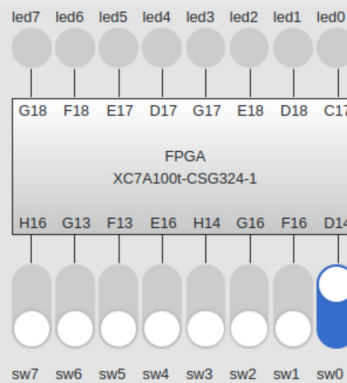
clk btn pins: clk_btn
xdc,ucf sym: B18

segplay(sharing with led) hexplay



00100001

FPGA interface



uart show>

soft clock

button

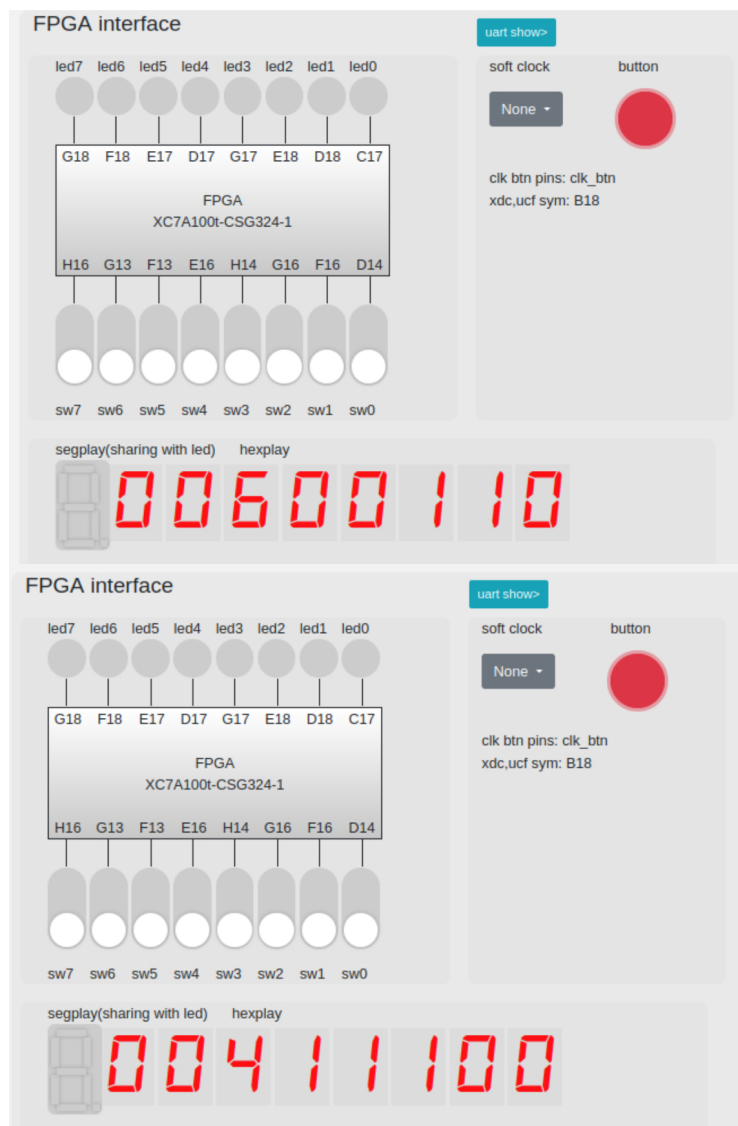
None ▾

clk btn pins: clk_btn
xdc,ucf sym: B18

segplay(sharing with led) hexplay



00300011



总结

难度：本次实验难度较大

任务量：大

建议：手册可以再完善亿点点吗

收获：加深了对有限状态机的掌握和了解，更熟练掌握Vivado和Logisim