状态机实验

朱河勤 PB16030899

2017 年 11 月 16 日

目录

1 实验目的

1 进一步学习时序逻辑电路

2 了解有限状态机的工作原理

3 学会使用“三段式”有限状态机设计电路

4 掌握按键去抖动、信号取边沿等处理技巧

2 设备外设

时钟（50MHz or 100MHz）、按键（使能输入) ,拨动开关（2个, 输入，复位用）、LED灯（1个）

3 要求

3.1 时序逻辑always块编码

1 使用异步复位方式

2 复位信号低电平有效

3 敏感变量列表中只能出现以下信号: 时钟（clk，50MHz/100MHz） 复位（rstn，n表示低电平有效）

3.2 编码

1 按功能划分模块

2 采用模块例化方式

3 顶层模块只负责子模块互联，不能包含assign、always等语句

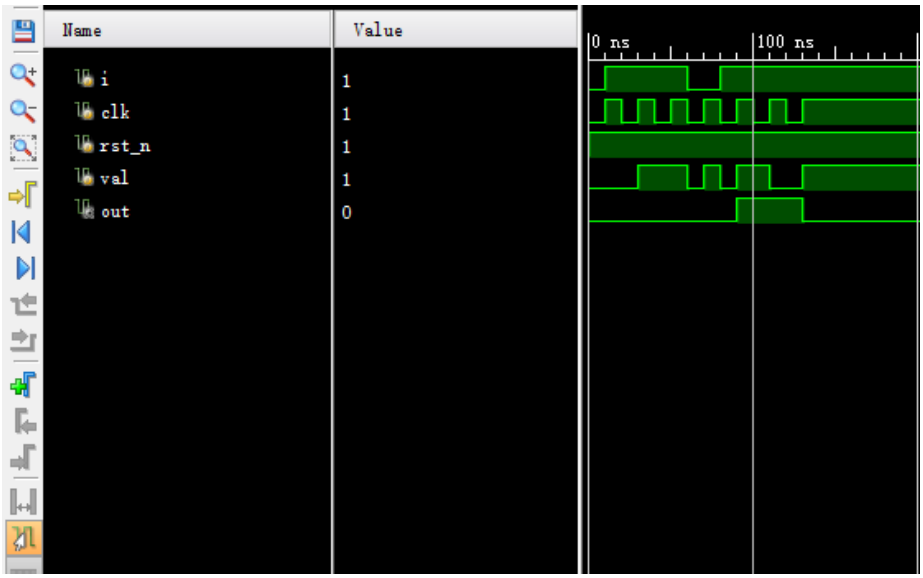
3.3 实验要求

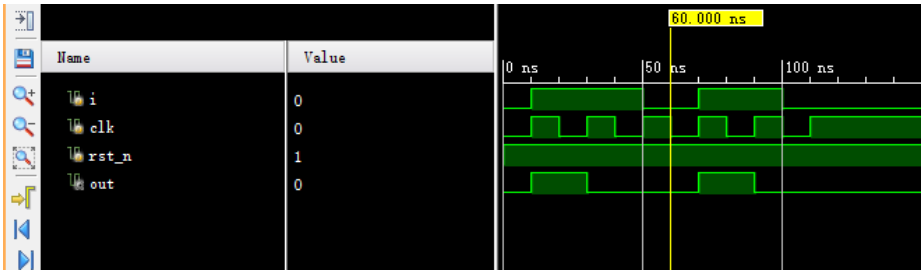
用三段式有限状态机实现序列检测功能电路按从高位到低位逐位串行输入一个序列每当检测到序列“1101”（不重叠）时，LED指示灯亮，否则灭，例如输入：１１０１１０１１０１输出：０００１０００００１用拨动开关输入检测序列按键按下的瞬间将拨动开关状态锁存注意防抖动（按键按下瞬间可能会有多次的电平跳变）

4 实验结果

成功满足要求，能够防按键抖动，能正确检查序列，并有异步复位功能。

仿真结果





5 实验分析

1 一定得按要求来写，我最开始没有写三段式，所以后来重写了

2 由于抖动时间为5ms–10ms,所有要计算去毛刺的时钟周期数，我用的50mhz的，所以取了250000

3 先去毛刺，然后得到上升沿，这里也有一些技巧

4 实验仿真也不能拘泥，比如仿真时，要检查去毛刺模块，可以把时钟周期数改小

6 源代码

```
module top(
        input sw,bt,clk,rst_n,
        output  led
        );
        wire val;
        wire ol;
//          cnt ins(.in(bt),.clk(clk),.rst_n(rst_n),.ol(ol));
        remove_glitch  (.key_in(bt),.clk(clk),.key_out(ol));
        valid (.clk(clk),.rst_n(rst_n),.sig(ol),.out(val));
        state  (.clk(clk),.val(val),.rst_n(rst_n),.data(sw),.y(led));
endmodule


module cnt(
        input in,clk,rst_n,
        output reg ol
        );
        reg flag;
        reg [19:0]ct;
        initial begin ct=0; flag =0;ol=0; end
        always @(*)
            begin
                if(~rst_n) begin ol<=0; ct<=0; end
                else
                   if(ct==250000) begin flag=0;ol=1;ct=0; end
                   else ol<=0;
                   if(flag) ct=ct+1;
                   else if(in)
                          begin flag=1;ct=ct+1 ; end
            end
endmodule
```

```
module valid(
    input clk,sig,rst_n,
    output reg out
    );
    reg flag;
    initial begin  flag =0;out=0;end
    always@(posedge clk,negedge rst_n)
        if(~rst_n) begin flag<=0; out<=0;  end
        else
        if(sig)
            if(flag==1) out<=0;
            else begin out<=1; flag<=1; end
        else begin flag<=0; out<=0;end
endmodule


module state(
    input rst_n,data,val,clk,
    output reg y
    );
    reg [1:0]st ;
    reg [1:0] last;
    parameter s0=2'b00,s1=2'b01,s2=2'b10,s3=2'b11;
    initial begin last=s0;y=0;  end
//    always @(posedge clk,negedge rst_n)
//        begin
//            if(~rst_n) begin st<=s0;y<=0; end
//            else
//                if(val)
//                begin
//                case(st)
```

```
//                  s0:begin     y<=0;if(data) st<=s1; end
//                  s1:if(data) begin st<=s2; y<=0; end
//                      else begin st<=s0; y<=0;  end
//                  s2:if(~data) begin st<=s3; y<=0; end
//                  s3:begin
//                       st<=s0;
//                       if(data) y<=1;else y<=0;
//                     end
//                  endcase
//                  end
//         end
    always @(posedge clk,negedge rst_n)
        if(~rst_n) st<=s0;
        else st<=last;



    always @(*)
    begin
        if(val)
        case(last)
            s0:if(data) last<=s1;
             s1:if(data) begin last<=s2; end
                        else begin last<=s0;end
             s2:if(~data)  last<=s3;
             s3:last<=s0;
             endcase
         end


      always@(*)
              if(val&&st==2'b11&&data)y<=1;
              else y<=0;
```

endmodule

状态原理图