**实验三 时序电路设计实验报告**

**姓名**：刘博文  **学号**：1120201883

**班级**：07112003 **手机**：18510892300

注：黑色字体内容不能改动，蓝色字体内容（为示例或说明）需删除和修改。

1. **实验题目**

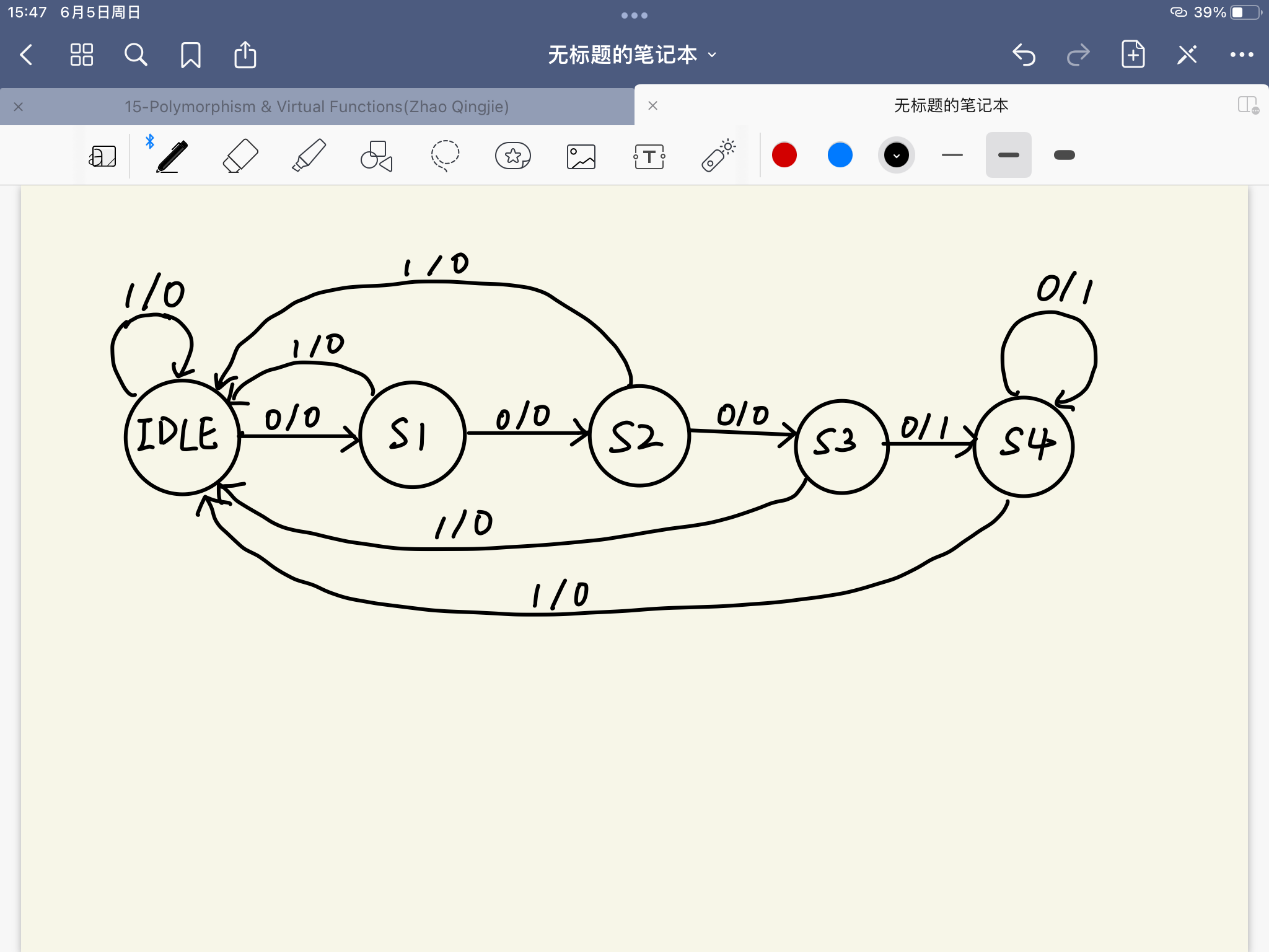
设计一个串行数据子序列检测器。当连续 输入 4 个或 4 个以上的 0 时，输出为 1，其他情况下输出为 0

1. **电路设计**
   1. **规范化**

输入：每个时钟周期内接受一个0或者1

输出：是否连续输入了4个或4个以上的0，若是输出1，若不是输出0

* 1. **形式化**



* 1. **状态分配**

IDLE状态：0

S1状态：1

S2状态：2

S3状态：3

S4状态：4

1. **电路实现**

module seq\_check(

    input clk,

    input rst\_n,

    input in,

    output reg out

);

parameter STATE\_IDLE = 4'd0;

parameter STATE\_S1   = 4'd1;

parameter STATE\_S2   = 4'd2;

parameter STATE\_S3   = 4'd3;

parameter STATE\_S4   = 4'd4;

reg [3:0] current\_state;

reg [3:0] next\_state;

always @ (posedge clk or negedge rst\_n) begin

    //复位则回到默认态

    if(~rst\_n) begin

        current\_state <= STATE\_IDLE;

    end

    else begin

        current\_state <= next\_state;

    end

end

always @(\*) begin

    case (current\_state)

        //只要输入0就进入下一状态

        STATE\_IDLE: next\_state = (in=='b0)  ?  STATE\_S1  :  STATE\_IDLE;

        STATE\_S1  : next\_state = (in=='b0)  ?  STATE\_S2  :  STATE\_IDLE;

        STATE\_S2  : next\_state = (in=='b0)  ?  STATE\_S3  :  STATE\_IDLE;

        STATE\_S3  : next\_state = (in=='b0)  ?  STATE\_S4  :  STATE\_IDLE;

        //已经检测出来连续4个0输入，继续输入0仍然停在

        STATE\_S4  : next\_state = (in=='b0)  ?  STATE\_S4  :  STATE\_IDLE;

        default:

        next\_state = STATE\_IDLE;

    endcase

end

always @ (posedge clk or negedge rst\_n) begin

    if(~rst\_n) begin

        out <= 'b0;

    end

    //状态达到S4，输出1，代表得到目标子序列

    else if(current\_state==STATE\_S4) begin

        out <= 'b1;

    end

    //没有达到状态S4，输出0，代表没有得到目标子序列

    else begin

        out <= 'b0;

    end

end

endmodule

1. **电路验证**
   1. **TestBench**

`timescale 1ps/1ps

module tb();

reg clk;

reg rst\_n;

reg in;

wire out;

always #1 clk<=~clk;

seq\_check u\_seq\_check(

    .clk   (clk  ),

    .rst\_n (rst\_n),

    .in    (in   ),

    .out   (out  )

);

initial begin

    clk<= 'b0;

    rst\_n <='b0;

    in<='b1;

    #10 rst\_n <='b1;

    //1个0子序列检测

    #2 in<='b0;

    #2 in<='b1;

    //2个0子序列检测

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b1;

    //3个0子序列检测

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b1;

    //4个及以上0子序列检测

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b0;

    #2 in<='b0;

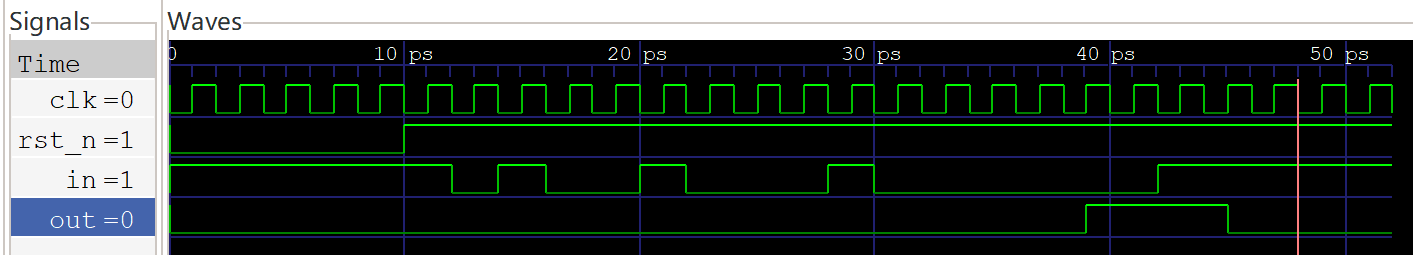
#2 in<='b1;

#10;

end

endmodule

* 1. **仿真结果**



从第10ps复位开始：

输入1个0，然后输入1，out为0，正确

输入2个0，然后输入1，out为0，正确

输入3个0，然后输入1，out为0，正确

输入5个0，然后输入1，out在输入第4个0之后变为1，并在输入1之后变为0，正确

1. **实验心得**

心得：时序电路的verilog实现的核心是状态机图，确定了正确的状态和状态转移条件，那么离得出实现就一步之遥了