**实验2：时序逻辑与状态机** **实验报告**

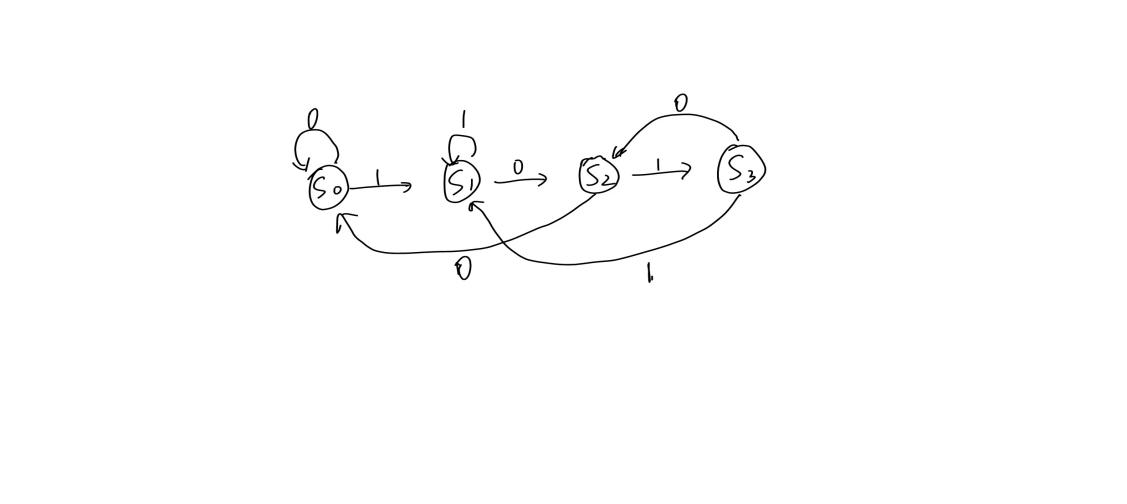
姓名： 刘贤彬 学号： 23336159

1. **实验内容**

使用FSM有限状态机进行序列检测，使用状态机检测“1011”，串行输入的测试序列为“1110110110111011”，输出信号为valid有效信号，检测到时输出高，否则为低，考虑序列叠加情况，比如“1011011”，则有两个“1011”这需要输出两个valid有效信号。

1. **实验过程**

（1）首先，做出状态机转移图，如图所示：



（2）然后写出状态机实现代码：

`timescale 1ns / 1ps

module seq\_detector(

input clk,

input rst,

input in,

output reg out

);

parameter ST0 = 0;

parameter ST1 = 1;

parameter ST2 = 2;

parameter ST3 = 3;

parameter ST4 = 4;

reg [2:0] state;

reg [2:0] next\_state;

initial

begin

next\_state = ST0;

end

always @ (in or state)

begin

case (state)

ST0:

begin

if(in == 1)

next\_state = ST1;

else

next\_state = ST0;

end

ST1:

begin

if(in == 0)

next\_state = ST2;

else if(in == 1)

next\_state = ST1;

end

ST2:

begin

if(in == 1)

next\_state = ST3;

else

next\_state = ST0;

end

ST3:

begin

if(in == 1)

next\_state = ST4;

else if( in == 0)

next\_state = ST2;

end

endcase

end

always @ (posedge clk or negedge rst)

begin

if(!rst)

state <= ST0;

else

state <= next\_state;

end

always @ (state or posedge clk)

begin

if(state == ST4)

begin

state = ST1;

out = 1;

end

else

out = 0;

end

endmodule

说明：ST4是个临时参数。

（3）仿真Testbench代码：

`timescale 1ns / 1ps

module seq\_detector\_tb(

);

reg clk;

reg rst;

reg in;

wire out;

initial

begin

clk = 0;

rst = 1;

#400

rst = 0;

#50

rst = 1;

end

initial

begin

in = 0;

#20

in = 1;

#20

in = 1;

#20

in = 1;

#20

in = 0;

#20

in = 1;

#20

in = 1;

#20

in = 0;

#20

in = 1;

#20

in = 1;

#20

in = 0;

#20

in = 1;

#20

in = 1;

#20

in = 1;

#20

in = 0;

#20

in = 1;

#20

in = 1;

end

always #10 clk = ~clk;

seq\_detector u\_seq\_detector(

.clk(clk),

.rst(rst),

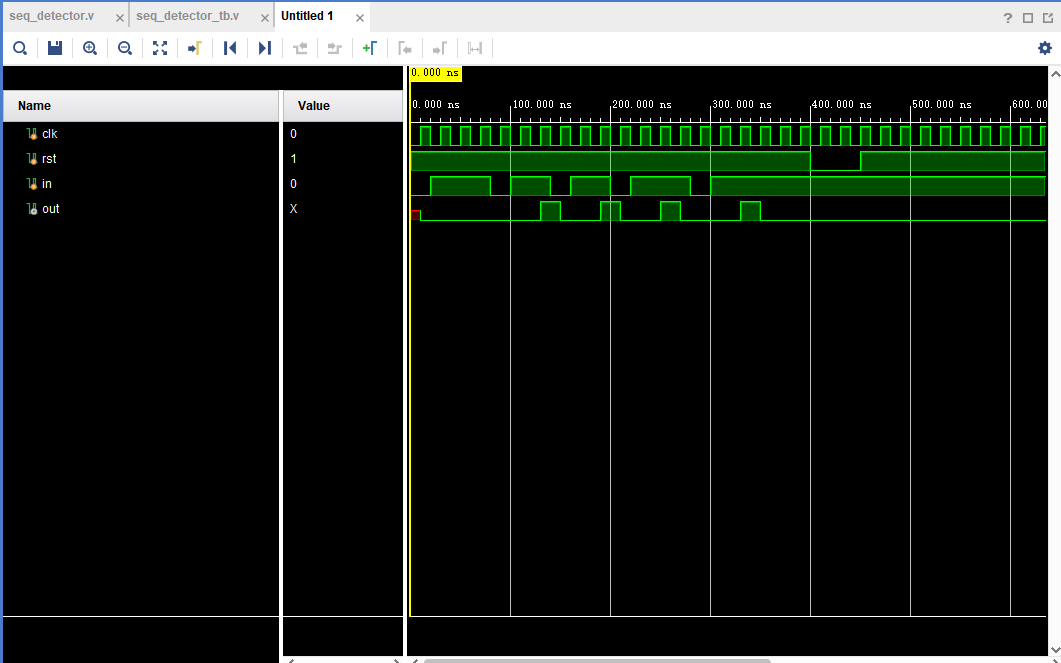
.in(in),

.out(out)

);

endmodule

1. **结果分析**

仿真结果如下图：

分析：检测序列是1110110110111011，从图里可以看出，第一个1011后输出1，并且可以检测出1011011中序列叠加情况。

1. **总结**

实现状态机，首先要画出状态机转移图，这个是最重要的蓝图，然后我们可以根据这个蓝图实现状态机的代码，最后写Testbench仿真代码。同时, 通过这次实验，初步熟悉了verilog代码编程。