

## Lab3——有限状态机



姓名：王昱

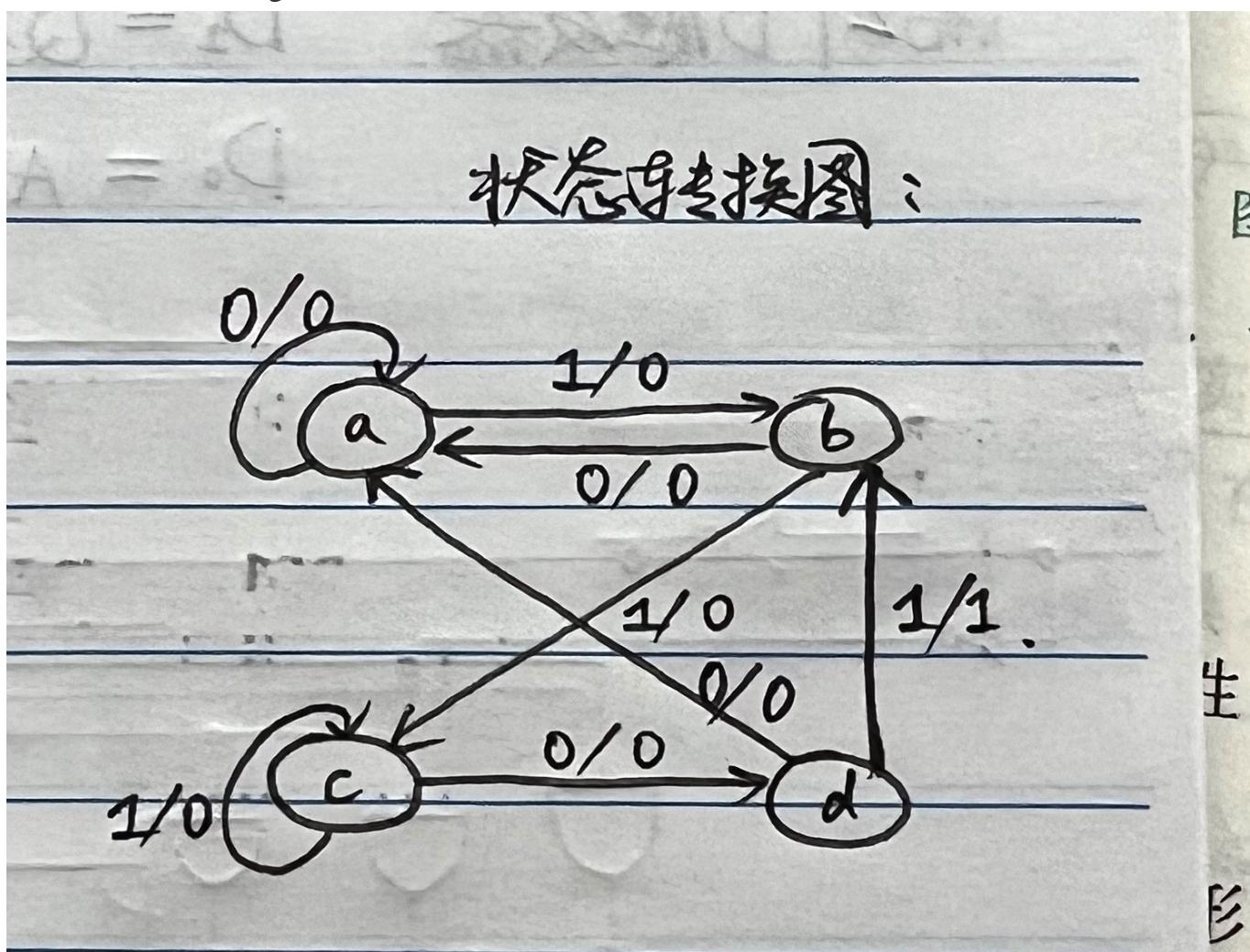
学号：PB21030814

## 一. 实验内容与目的

- ① Mealy、Moore型电路的区别
- ② 两种方式描述FSM——两段式、三段式
- ③ 设计序列检测器，监测1101

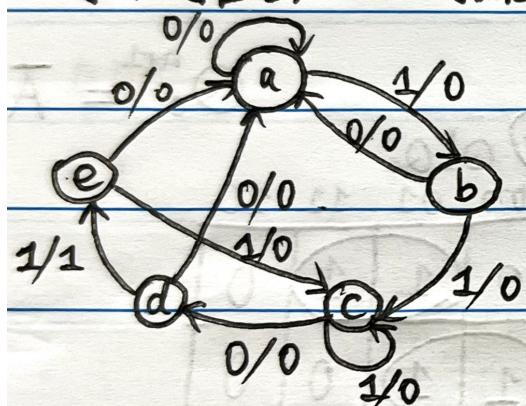
## 二. 逻辑设计

- 有限状态机
- 两段式Mealy型的状态转换图(4种状态)



- 三段式Moore型的状态转换图(5种状态)

输入信号A      输出信号Z



a: "0"    b: "1"

c: "11"    d: "110"

e: "1101"

这里注意Moore型的FSM输出仅与状态有关，与输入无关，所以要用五位独热码来表示。

- 核心代码

```
//两段式
module two_FSM_detect1101(
    input x,
    input rstn,
    input clk,
    output reg y1,
    output reg [1:0] sl
);
//顺序码
parameter s0 = 2'b00,
          s1 = 2'b01,
          s2 = 2'b10,
          s3 = 2'b11;
//存放状态的两个变量
reg [1:0] NS,CS;
always @(posedge clk or negedge rstn) begin
    if(!rstn)
        CS <= s0;
    else
        CS <= NS;
end
always @(*) begin
    y1 = 0;
    sl = 2'b00;
    NS = s0;
    case(CS)
        s0:begin
            sl = 2'b00;
            y1 = 0;
            NS = x ? s1 : s0;
        end
        s1:begin
            sl = 2'b01;
        end
    endcase
end
```

```
    y1 = 0;
    NS = x ? s2 : s0;
end
s2:begin
    s1 = 2'b10;
    y1 = 0;
    NS = x ? s2 : s3;
end
s3:begin
    s1 = 2'b11;
    y1 = x ? 1 : 0;
    NS = x ? s1 : s0;
end
default:begin
    y1 = 0;
    s1 = 2'b00;
    NS = s0;
end
endcase
end
endmodule
```

两段式*Mealy*型用两个`always`块实现，其中组合逻辑电路部分用了大量的条件运算符，使得代码简洁易读。`case`语句的功能是根据 $CS$ (现态)和 $x$ (输入变量)来决定输出。

```

//三段式

module three_FSM_detect1101(
    input x,
    input rstn,
    input clk,
    output reg yr,
    output reg [4:0] sr
);
//独热码

parameter s0 = 5'b00001,
          s1 = 5'b00010,
          s2 = 5'b00100,
          s3 = 5'b01000,
          s4 = 5'b10000;

//存放状态的两个变量

reg [4:0] NS,CS;
always @(posedge clk or negedge rstn) begin
    if(!rstn)
        CS <= s0;
    else
        CS <= NS;
end
always @(*) begin
    NS = s0;
    case(CS)
        s0: NS = x ? s1 : s0;
        s1: NS = x ? s2 : s0;
        s2: NS = x ? s3 : s0;
        s3: NS = x ? s4 : s0;
        s4: NS = x ? s2 : s0;
        default: NS = s0;
    endcase
end

```

```

always @(posedge clk or negedge rstn) begin
    if(!rstn)
        {yr,sr} <= 6'b000001;
    else begin
        {yr,sr} <= 6'b000001;
        case(NS)
            s0: {yr,sr} <= 6'b000001;
            s1: {yr,sr} <= 6'b000010;
            s2: {yr,sr} <= 6'b000100;
            s3: {yr,sr} <= 6'b001000;
            s4: {yr,sr} <= 6'b110000;
            default: {yr,sr} <= 6'b000001;
        endcase
    end
end
endmodule

```

三段式Moore型用三个`always`块实现，同样组合逻辑电路部分使用了大量的条件运算符使得代码简洁易读。使用拼接运算符简化了代码。

### 三. 仿真结果与分析

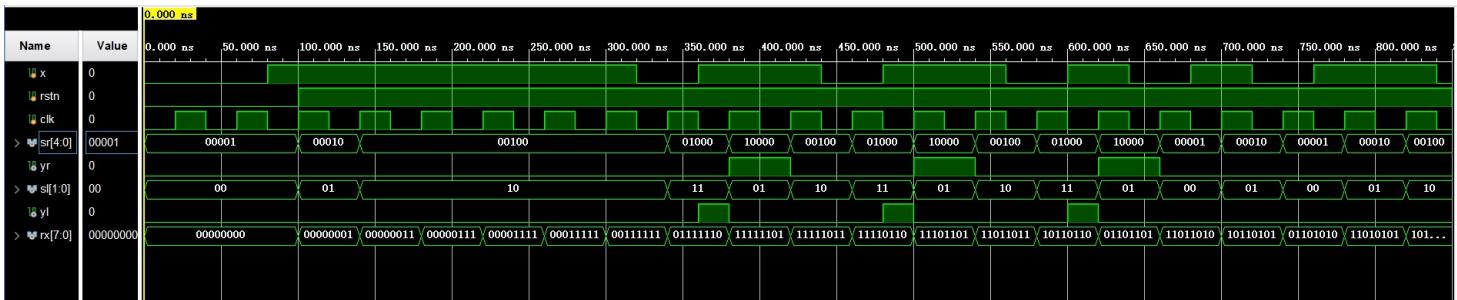
- 这里仅展示序列检测器的仿真结果
- 代码部分

```

module SQD_tb();
    reg x,rstn,clk;
    wire [4:0] sr;
    wire yr;
    wire [1:0] sl;
    wire yl;
    wire [7:0] rx;
    SQD SQD_test(
        .x(x),
        .rstn(rstn),
        .clk(clk),
        .sr(sr),
        .yr(yr),
        .sl(sl),
        .yl(yl),
        .rx(rx)
    );
    initial begin
        clk = 1'b0;
        rstn = 1'b0;
        x = 1'b0;
        # 100 rstn = 1'b1;
    end
    always #20 clk = ~clk;
    always #40 x = {$random} % 2;
endmodule

```

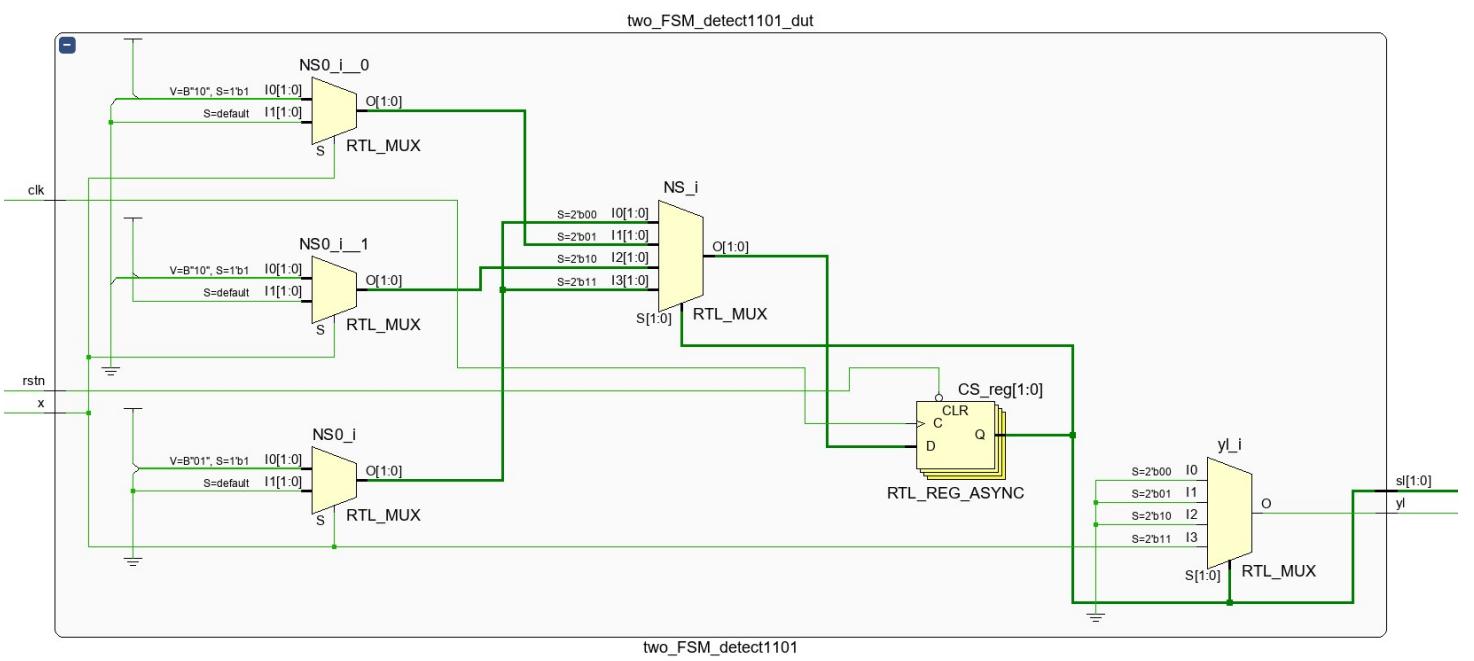
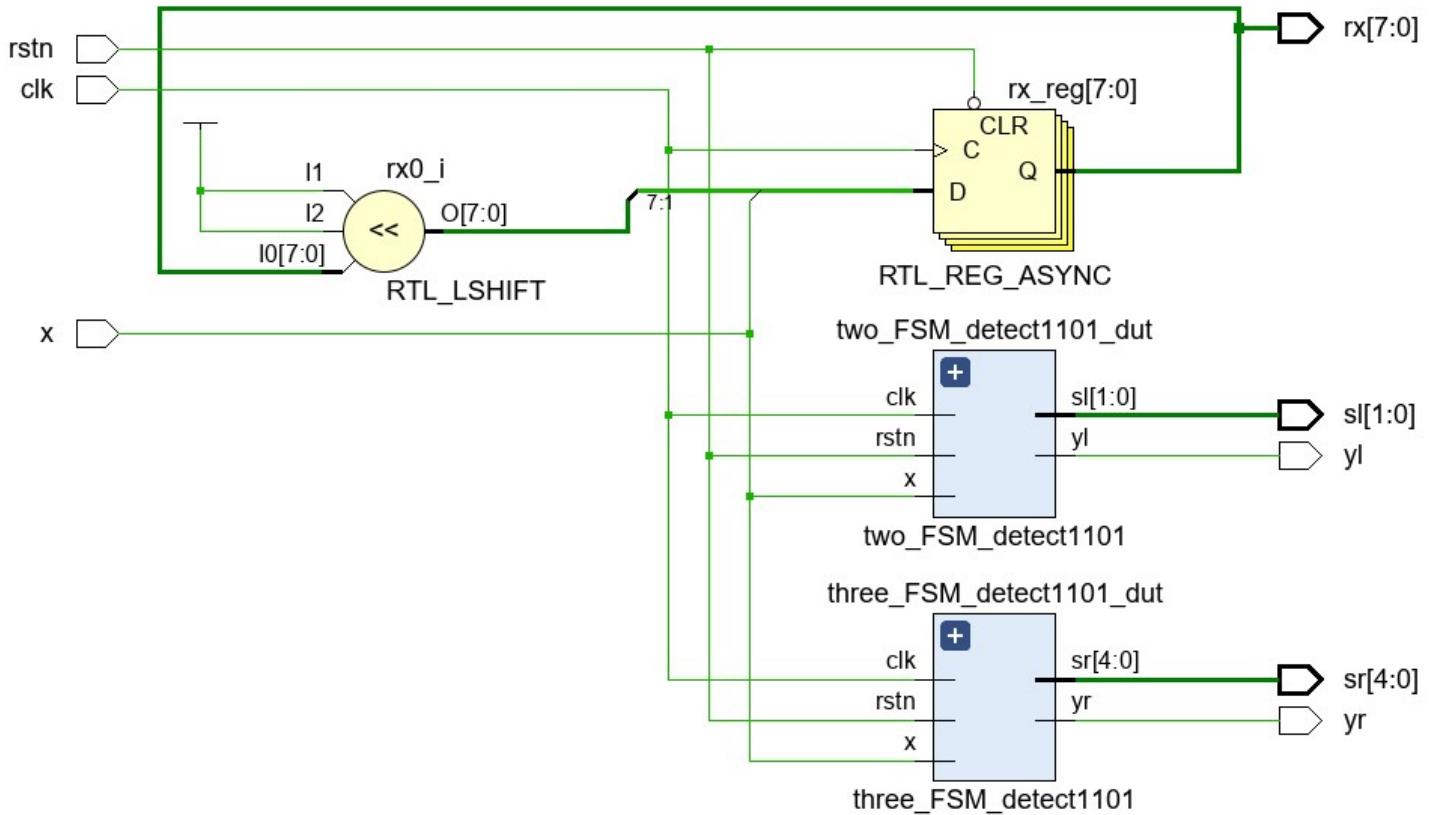
- 仿真结果截图

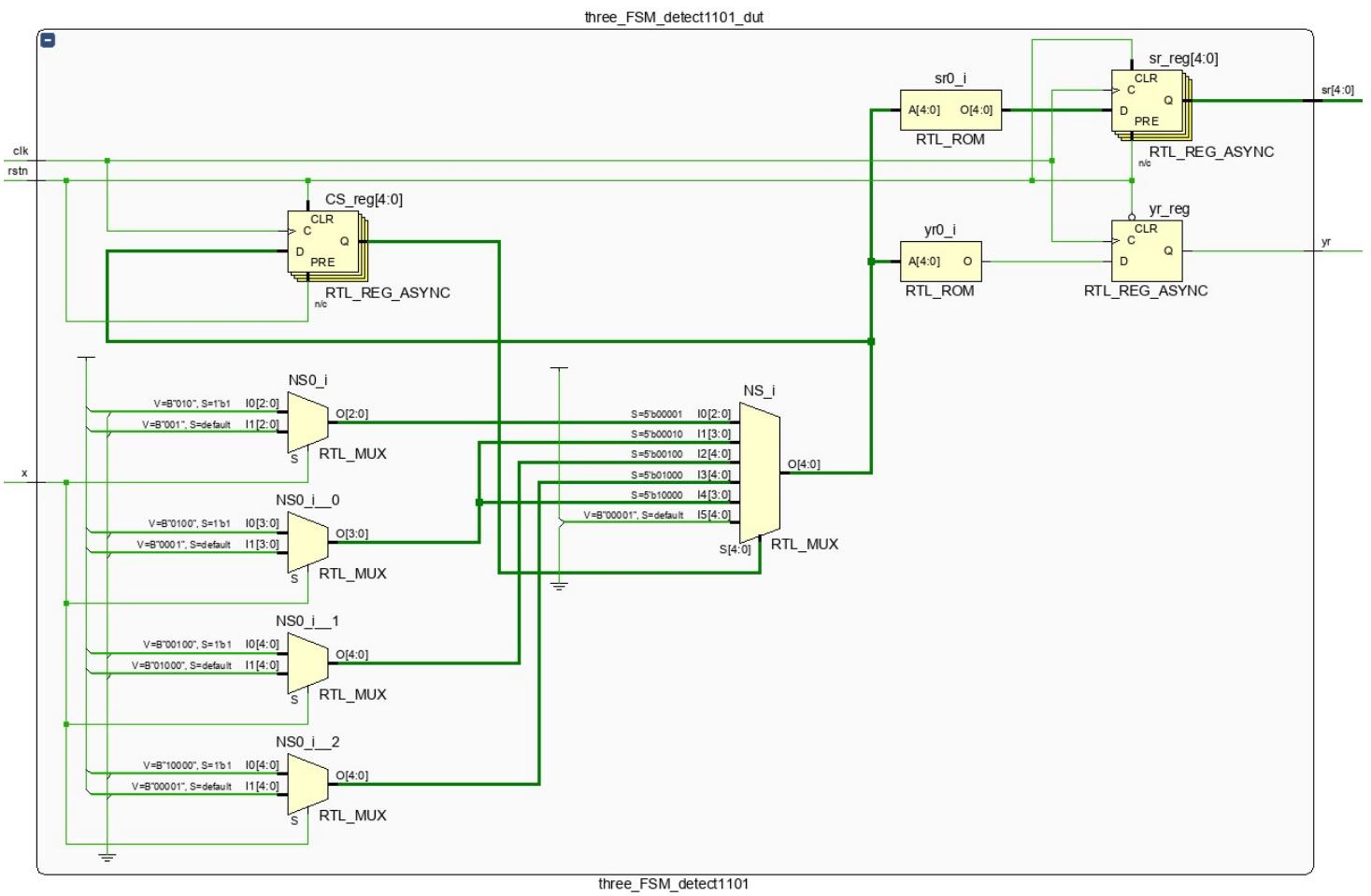


由截图可以看到， $rx$ 代表的最近输入的8位序列与模拟的输入结果一致。独热码 $sr$ 和顺序码 $sl$ 也与实际结果(状态转换图)一致。同时可以观察到组合逻辑电路与时序逻辑电路的区别：在350ns到450ns之间检测到了1101序列，但是二段式Mealy型在一输入1的时候 $yl$ 立刻变为1而三段式Moore型等到时钟上升沿到来之后 $yr$ 才变为1。这是因为 $yl$ 在组合电路中输出而 $yr$ 在时序电路中输出。

#### 四. 电路设计与分析

- RTL电路图



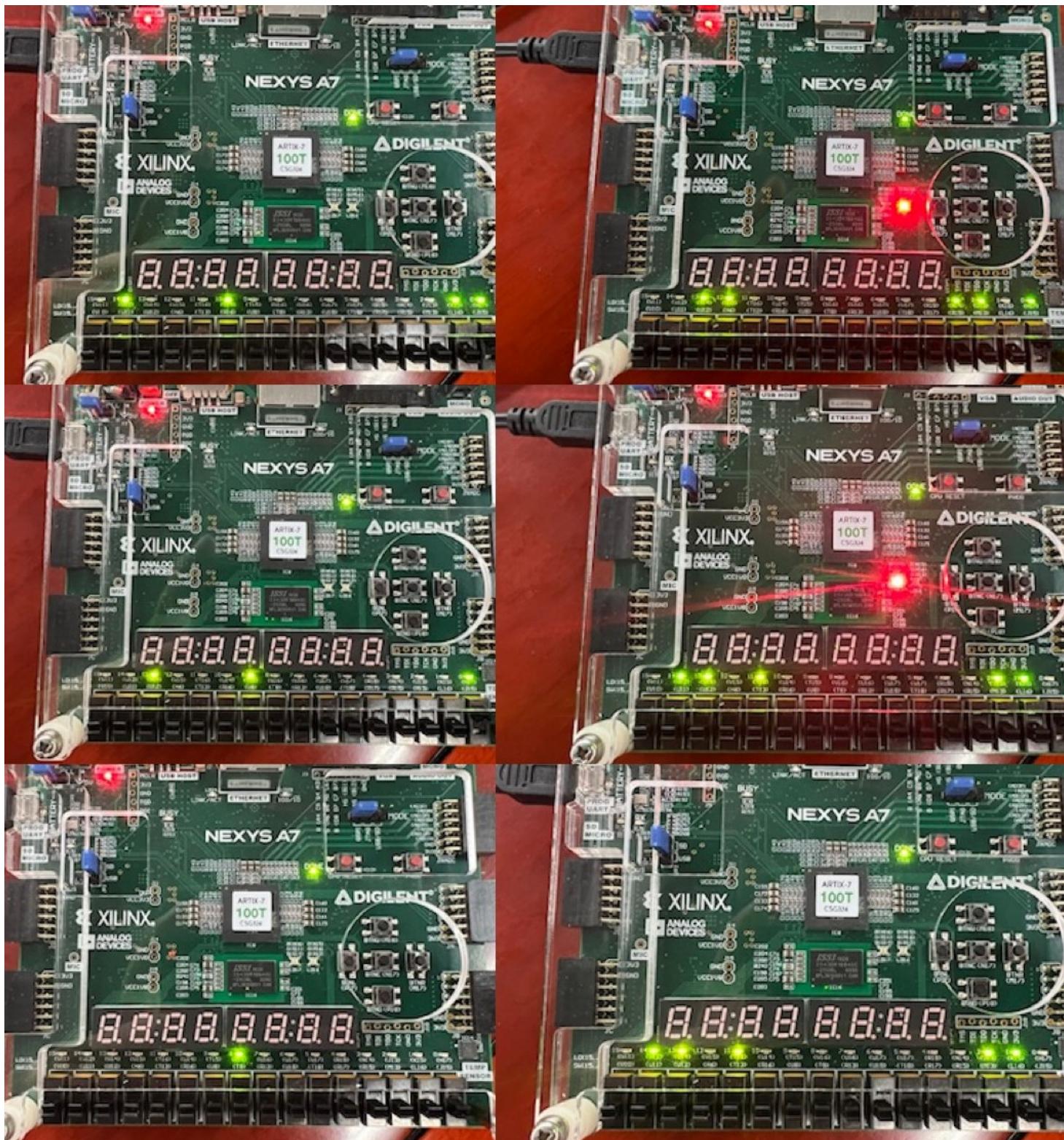


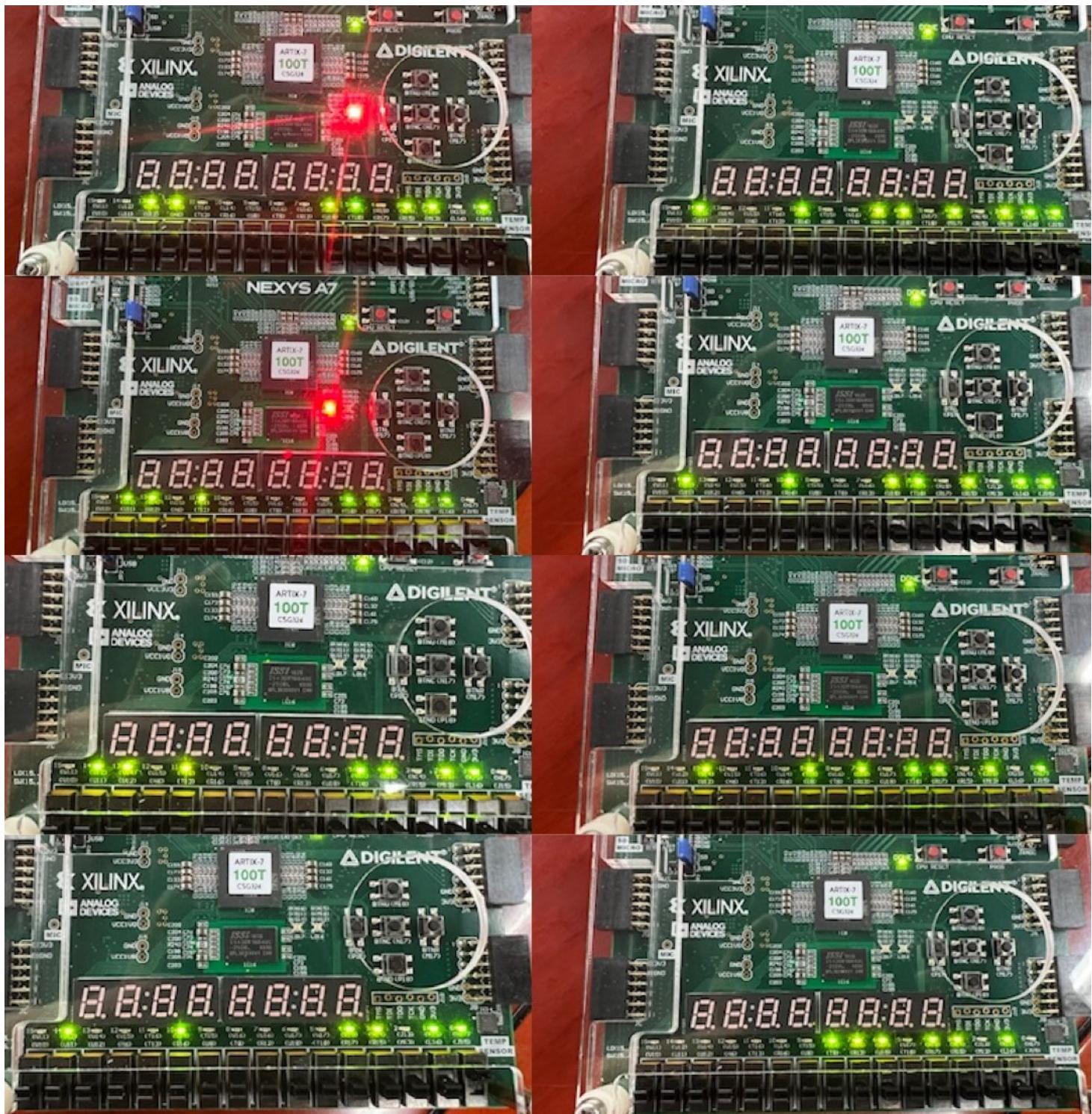
three\_FSM\_detect1101

## • 资源使用情况

Name	1	Slice LUTs (63400)	Slice Registers (126800)	Slice (15850)	LUT as Logic (63400)	Bonded IOB (210)	BUFGCTRL (32)
SQD		12	27	12	12	20	1
three_FSM_detect1101_dut (three_FSM_detect1101)		10	11	5	10	0	0
two_FSM_detect1101_dut (two_FSM_detect1101)		2	2	1	2	0	0

## 五. 测试结果与分析





该结果与状态转换图以及仿真结果一致

## 六. 总结

- ①在实验中出现了抖动现象，一开始并不知道，以为开发板出现了问题。后来询问助教之后解决了该问题。
- ②一开始独热码采用了四位，忽略了Moore型时序电路的特点——

只与状态有关而与输入无关。后经修改解决该问题。

③收获：学会了用两段式、三段式方法描述*FSM*

④*Tips*：助教能不能每次把测试用例发一下来验证我们做的是否正确。