**实验报告**

实验题目： **有限状态机**  日期： 2018 年 11 月 30 日

姓名: 罗晏宸 学号: PB17000297 成绩:\_\_\_\_\_\_\_\_\_\_\_\_

**实验目的：**

1. 学习如何对Mealy型与Moore型的FSMs状态机分过程块建模
2. 设计实际的序列检测器并加以实现

**实验内容**（截图、照片与代码）

Lab10\_1\_1

代码Mealy\_3Processes\_SequenceDetector\_3Multiples.v

`timescale 1ns / 1ps

module Mealy\_3Processes\_SequenceDetector\_3Multiples(

input clock,

input ain,

input reset,

output reg [3:0] Count,

output reg yout

);

reg [1:0] state, nextstate;

parameter S0=2'b00, S1=2'b01, S2=2'b10;

always @(posedge clock or posedge reset)

    if (reset)

    begin

            state <= S0;

            Count <= 4'b0000;

        end

    else

    begin

        state <= nextstate;

        Count <= Count + {3'b000,ain};

    end

always @(state or ain)

begin

    yout = 1'b0;

    case(state)

        S0: yout = 1'b1;

        S1: ;

        S2: ;

    endcase

end

always @(state or ain)

begin

    case(state)

        S0: nextstate = ain ? S1 : S0;

        S1: nextstate = ain ? S2 : S1;

        S2: nextstate = ain ? S0 : S2;

    endcase

end

endmodule

代码Mealy\_3Processes\_SequenceDetector\_3Multiples\_tb.v

`timescale 1ns / 1ps

module Mealy\_3Processes\_SequenceDetector\_3Multiples\_tb();

    wire clock;

    reg ain,reset;

    wire yout;

    wire [3:0] Count;

    Mealy\_3Processes\_SequenceDetector\_3Multiples FSM (.clock(clock),.ain(ain),.reset(reset),.Count(Count),.yout(yout));

    GenerateClock CLK (clock);

    initial

    begin

        reset = 1; ain = 0;

        #20 reset = 0;

        #20 ain = 1;

        #20 ain = 0;

        #60 ain = 1;

        #40 ain = 0;

        #20 ain = 1;

        #10 reset = 1;

    end

endmodule

代码GenerateClock.v

`timescale 1ns / 1ps

module GenerateClock(output reg Clk);

parameter tON = 5,tOFF = 5;

initial

 begin

        Clk = 0;

        forever

        begin

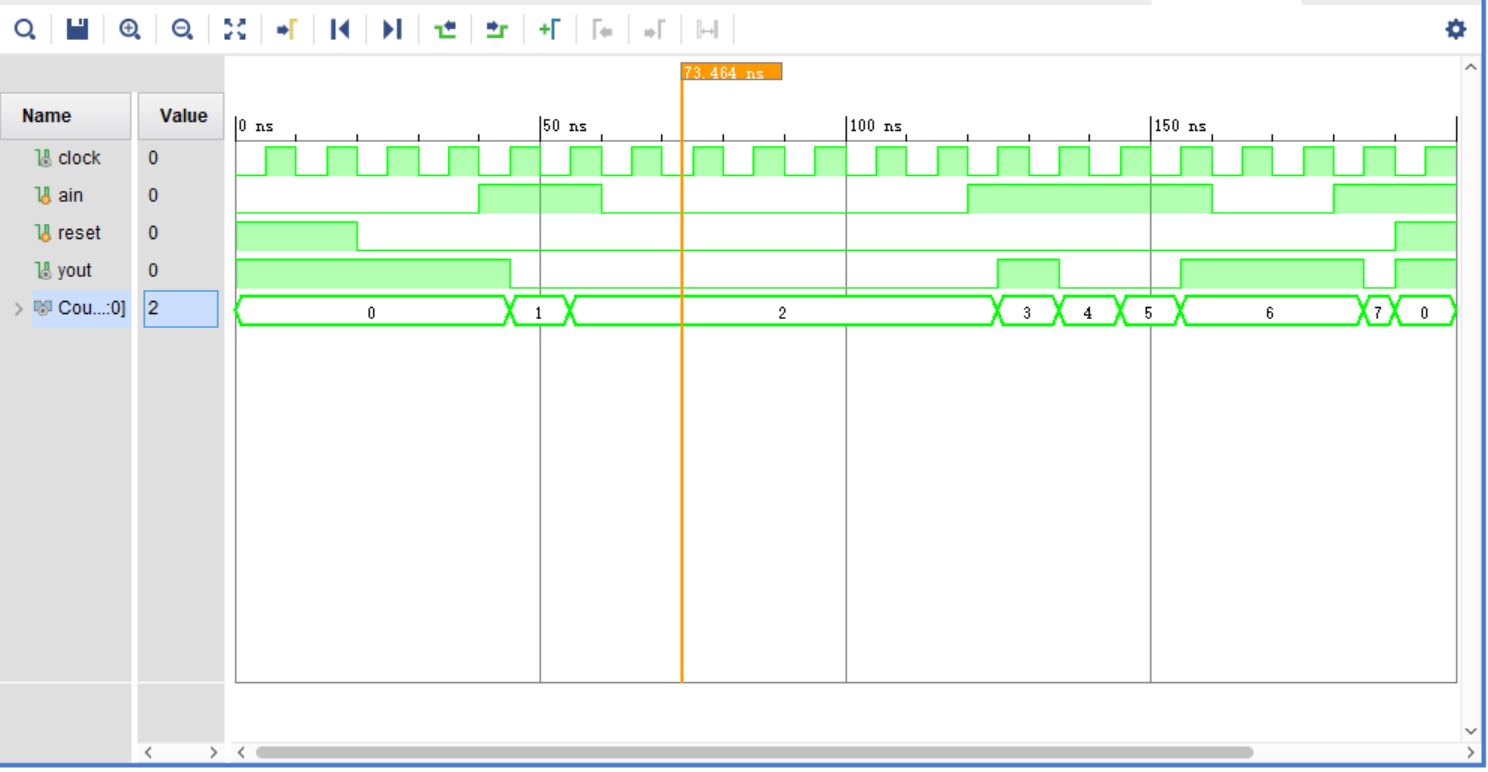
            # tOFF Clk = 1;

            # tON Clk = 0;

        end

end

endmodule



截图 1-lab10\_1\_1仿真



照片 1-lab10\_1\_1下载

“照片 1-lab10\_1\_1下载”的说明：本工程使用Mealy型FSM实现输出1的总数是否为3的倍数的检测，图中LED[0], LED[5], LED[6] 亮起，其余LED灯均灭，表明此时已输入(0110)B即(6)D个1，而6整除3，故yout=1

Lab10\_2\_1

代码Moore\_3Processes\_SequenceDetector.v

`timescale 1ns / 1ps

module Moore\_3Processes\_SequenceDetector(

input clock,

input [1:0] ain,

input reset,

output reg yout

);

reg [2:0] state, nextstate;

parameter S0 = 3'b000, S1 = 3'b001, S2 = 3'b010, S3 = 3'b011, S4 = 3'b100, S5 = 3'b101, S6 = 3'b110;

//XXXX, 0100, 1100, 1000, XX01, XX11, XX10

always @(posedge clock or posedge reset) // always block to update state

begin

    if (reset)

    begin

        state <= S0;

        yout <= 1;

    end

    else

        state <= nextstate;

end

always @(state) // always block to compute output

begin

    case(state)

        S1: yout = 0;

        S2: yout = 1;

        S3: yout = ~yout;

    endcase

end

always @(state or ain) // always block to compute nextstate

begin

        case (ain)

            2'b01: nextstate = S4;

            2'b11: nextstate = S5;

            2'b10: nextstate = S6;

            2'b00:

            begin

                nextstate = S0;

                case (state)

                    S4: nextstate = S1;

                    S5: nextstate = S2;

                    S6: nextstate = S3;

                endcase

            end

        endcase

end

endmodule

代码Moore\_3Processes\_SequenceDetector\_tb.v

`timescale 1ns / 1ps

module Moore\_3Processes\_SequenceDetector\_tb();

    wire clock,yout;

    reg reset;

    reg [1:0] ain;

    Moore\_3Processes\_SequenceDetector FSM (.clock(clock),.ain(ain),.reset(reset),.yout(yout));

    GenerateClock CLOCK (clock);

    initial

    begin

        reset = 1; ain = 2'b00;

        #20 reset = 0;

        #20 ain = 2'b11;

        #10 ain = 2'b10;

        #10 ain = 2'b00;

        #20 ain = 2'b10;

        #10 ain = 2'b00;

        #10 ain = 2'b11;

        #10 ain = 2'b00;

        #10 ain = 2'b01;

        #10 ain = 2'b00;

        #10 ain = 2'b10;

        #10 ain = 2'b11;

        #10 ain = 2'b00;

        #10 reset = 1;

        #10 reset = 0;

        #10 ain = 2'b10;

        #30 ain = 2'b00;

    end

endmodule

代码GenerateClock.v

`timescale 1ns / 1ps

module GenerateClock(output reg Clk);

parameter tON = 5,tOFF = 5;

initial

 begin

        Clk = 0;

        forever

        begin

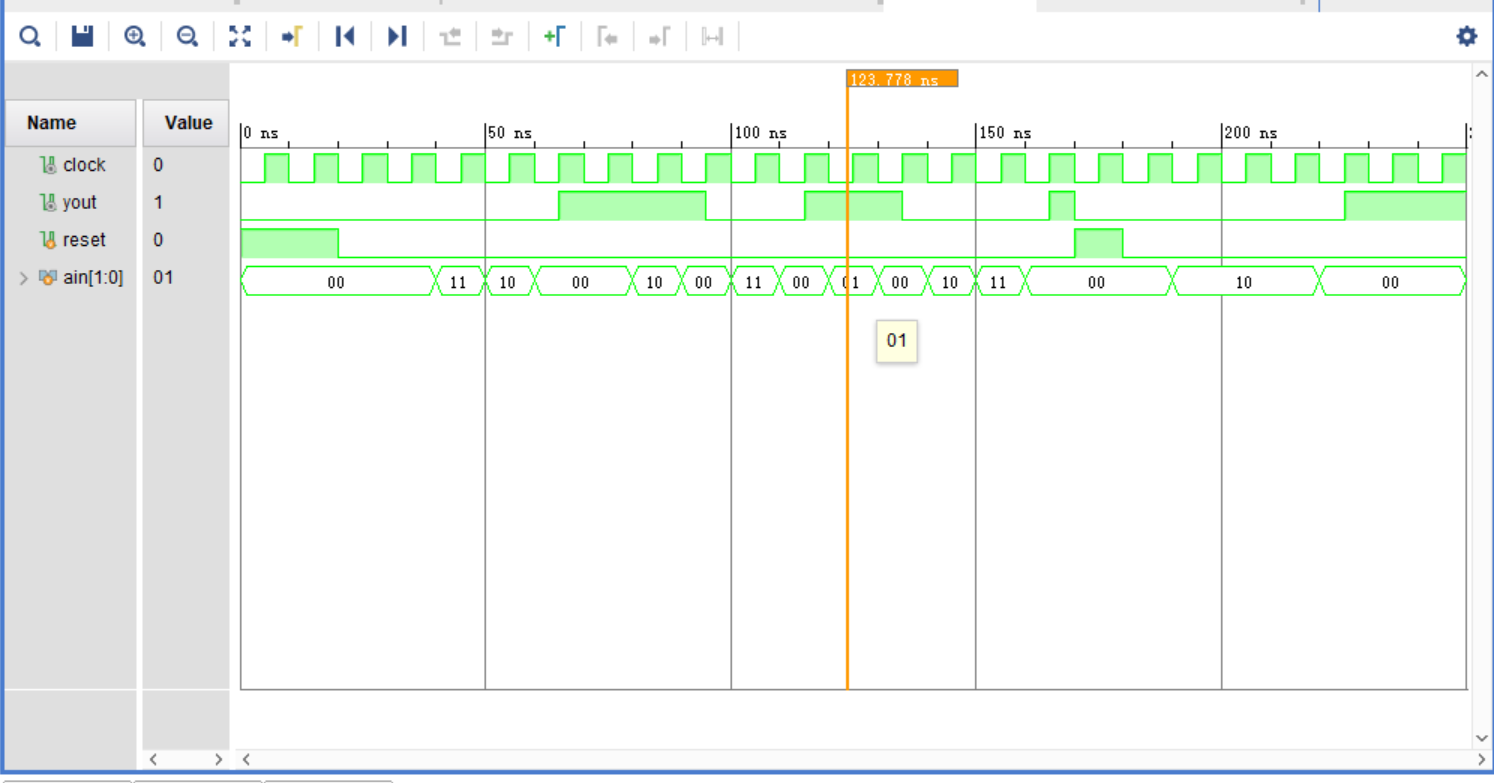
            # tOFF Clk = 1;

            # tON Clk = 0;

        end

end

endmodule



截图 2-lab10\_2\_1仿真

**实验总结**

本次实验中，学习了Mealy型与Moore型状态机的功能基础与过程块的实现方式，并通过实践锻炼了分析实际状态过程的能力，巩固了时序逻辑电路的开发技术，为之后更大规模的实际应用开发打下基础。