Moore vs. Mealy FSMs

Alyssa P. Hacker has a snail that crawls down a paper tape with 1's and 0's on it. The snail smiles whenever the last two digits it has crawled over are **1101**. Design **Moore** and **Mealy** FSMs of the snail's brain.

What to turn in:

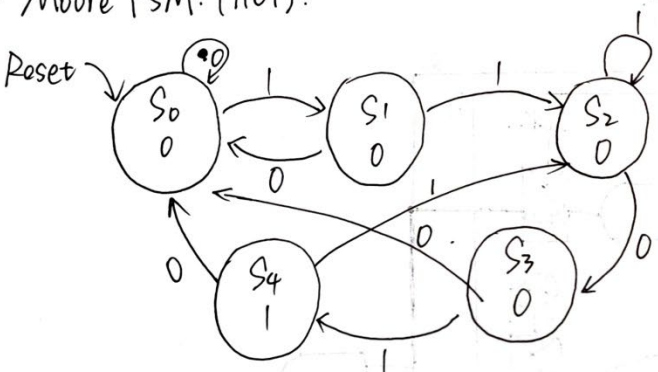
1. A completed State Transition Diagram for the FSM.
2. Tables listing (1) next state in terms of current state and inputs, and (2) output in terms of current state and inputs.
4. The binary encoding for each state.
5. The revised copy of your tables, using your binary encoding.
6. Boolean logic equations for the outputs and each bit of the next state in terms of the state and inputs.
7. schematic of the FSM.
8. Verilog code for the FSM design and testbench, snapshot of the simulation waveform

Part 1--Moore FSM

同济大学 TONGJI UNIVERSITY
地址：中国上海市四平路1239号 邮编：200092
1239 SIPING ROAD SHANGHAI CHINA 200092
电话(TEL):+86 21- 传真(FAX)::+86 21-
网址(WEB):www.tongji.edu.cn

HW-07 FSM.

1. Moore FSM. (1101).



2. State Transition Table.

S(current)	Input	S'(next)	Output
S0	0	S0	0
S0	1	S1	0
S1	0	S0	0
S1	1	S2	0
S2	0	S3	0
S2	1	S2	0
S3	0	S0	0
S3	1	S4	0
S4	0	S0	1
S4	1	S2	1

4. Binary Encoding.

S0 S1 S2 S3 S4

000 001 010 011 100.

5. Encoding Trans.

$S_2S_1S_0$	Input	$S_2'S_1'S_0'$	Output
000	0	000	0
000	1	001	0
001	0	000	0
001	1	010	0
010	0	011	0
010	1	010	0
011	0	000	0
011	1	100	0
100	0	000	1
100	1	010	1

同济大学 TONGJI UNIVERSITY
地址：中国上海市四平路1239号 邮编：
1239 SIPING ROAD SHANGHAI CHIN
电话(TEL):+86 21-　　传真(FAX)
网址(WEB):www.tongji.edu.cn

6、 Boolean Equations:

(1) Input: A.

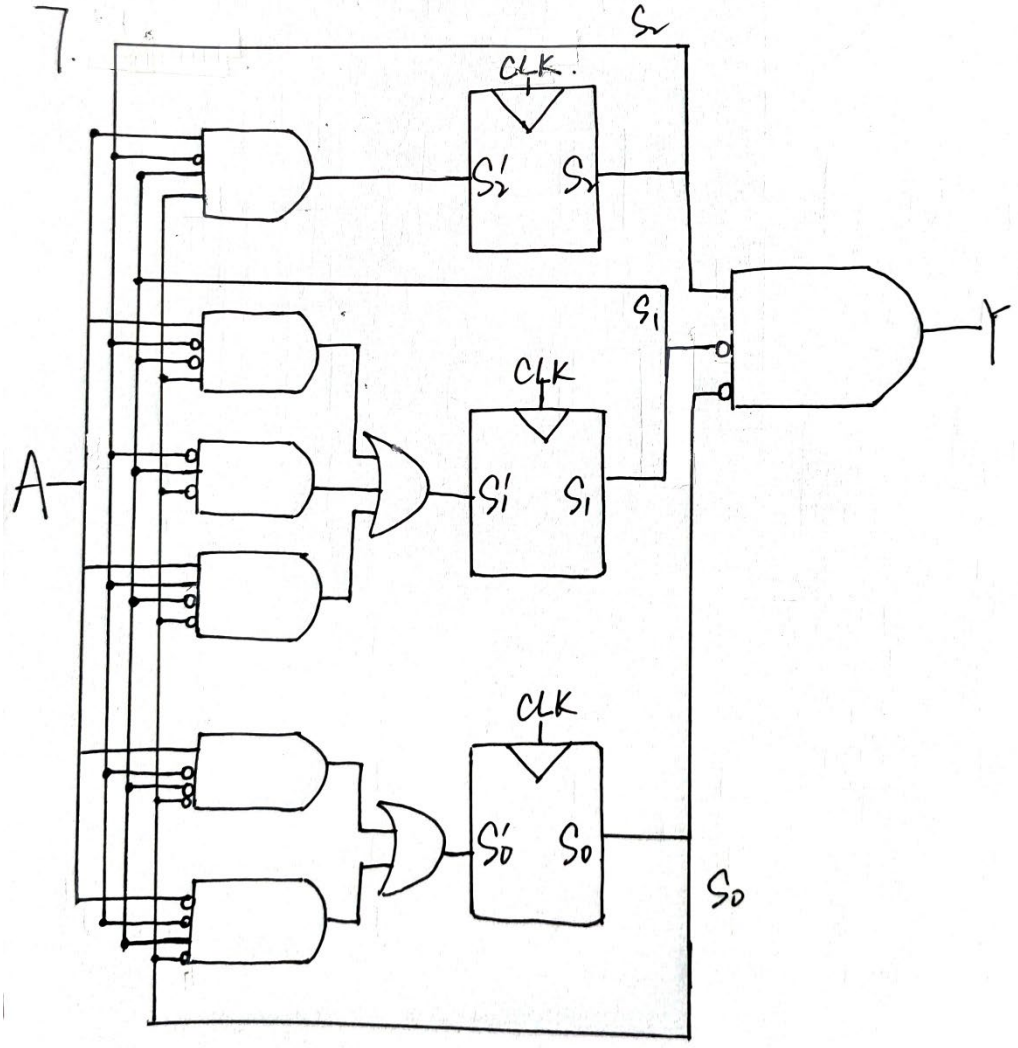
$S_2' = \overline{S_2}S_1S_0A$

$S_1' = \overline{S_2}\overline{S_1}S_0A + \overline{S_2}S_1\overline{S_0} + S_2\overline{S_1}\overline{S_0}A$

$S_0' = \overline{S_2}\overline{S_1}\overline{S_0}A + \overline{S_2}\cdot S_1\overline{S_0}\cdot\overline{A}$

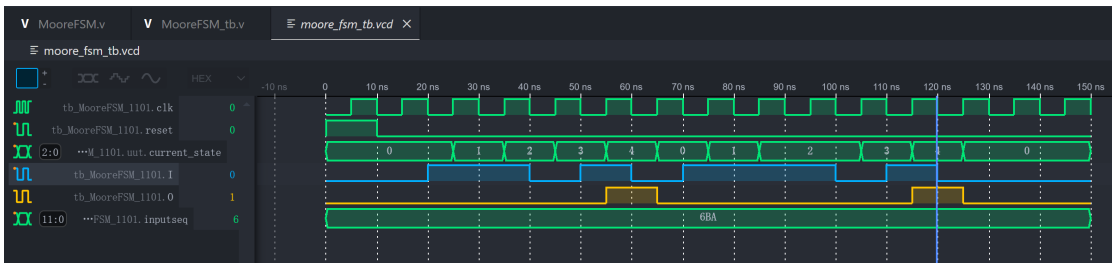
(2) output: $Y = S_2\cdot\overline{S_1}\overline{S_0}$

7.

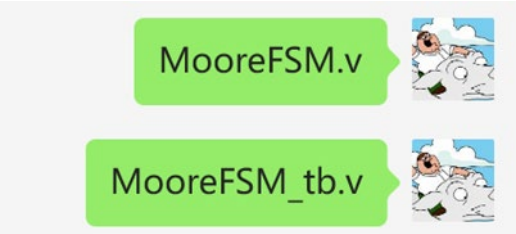


Here are the snapshots of my MooreFSM.

In the sequence 12'b011010111010 ,it turns 1 from 0 at the 6th and 12th clk.




```
PS D:\1_课程文件\数字电子技术\Verilog_Learning\Lecture6_Sequence> iverilog -o moore.out MooreFSM.v MooreFSM_tb.v
PS D:\1_课程文件\数字电子技术\Verilog_Learning\Lecture6_Sequence> vvp moore.out
VCD info: dumpfile moore_fsm_tb.vcd opened for output.
Time = 20000 ns, Input = 0, State = 000, Output = 0
Time = 30000 ns, Input = 1, State = 001, Output = 0
Time = 40000 ns, Input = 1, State = 010, Output = 0
Time = 50000 ns, Input = 0, State = 011, Output = 0
Time = 60000 ns, Input = 1, State = 100, Output = 1
Time = 70000 ns, Input = 0, State = 000, Output = 0
Time = 80000 ns, Input = 1, State = 001, Output = 0
Time = 90000 ns, Input = 1, State = 010, Output = 0
Time = 100000 ns, Input = 1, State = 010, Output = 0
Time = 110000 ns, Input = 0, State = 011, Output = 0
Time = 120000 ns, Input = 1, State = 100, Output = 1
Time = 130000 ns, Input = 0, State = 000, Output = 0
MooreFSM_tb.v:36: $finish called at 150000 (1ps)
```




```verilog
module MooreFSM_1101 (
    input  clk,
    input  reset,
    input  I,
    output reg O );

    parameter S0 = 3'b000;
    parameter S1 = 3'b001;
    parameter S2 = 3'b010;
    parameter S3 = 3'b011;
    parameter S4 = 3'b100;

    reg [2:0] current_state, next_state;

    always @(posedge clk or posedge reset) begin
        if (reset)
            current_state <= S0;    // 复位时回到初始状态S0
        else
            current_state <= next_state; // 正常工作时更新状
态   end

    always @(*) begin
        case (current_state)
            S0: next_state = (I == 1'b1) ? S1 : S0;
            S1: next_state = (I == 1'b1) ? S2 : S0;
            S2: next_state = (I == 1'b0) ? S3 : S2;
            S3: next_state = (I == 1'b1) ? S4 : S0;
            S4: next_state = (I == 1'b1) ? S2 : S0;
            default: next_state = S0;
        endcase
    end

    always @(*) begin
        O = (current_state == S4) ? 1'b1 : 1'b0;
    end

endmodule
```

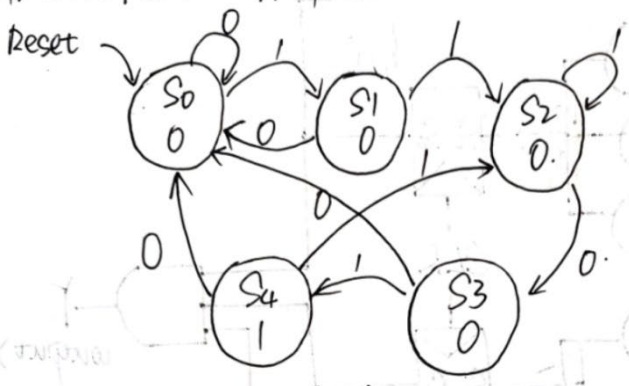
```
`timescale 1ns/1ps
module tb_MooreFSM_1101();

    reg clk, reset, I;
    wire O;
    integer i; // 循环计数器
    reg [11:0] inputseq = 12'b011010111010;

    initial begin
        $dumpfile("moore_fsm_tb.vcd");
        $dumpvars(0, tb_MooreFSM_1101);
    end

    MooreFSM_1101 uut (
        .clk(clk),
        .reset(reset),
        .I(I),
        .O(O)
    );
    initial begin
        clk = 0;
        forever #5 clk = ~clk;
    end

    initial begin

        reset = 1; I = 0;
        #10; reset = 0;

        for (i = 11; i >= 0; i = i - 1) begin
            I = inputseq[i];
            #10;
            $display("Time = %0t ns, Input = %b, State = %b, Output = %b",
                    $time, I, uut.current_state, O);
        end
        #20 $finish;
    end

endmodule
```

Part 2--Mealy FSM

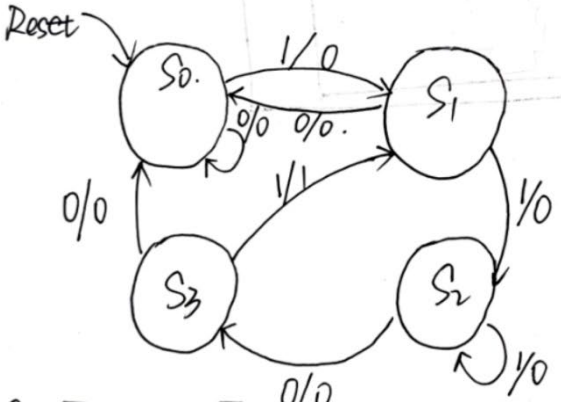
同济大学 TONGJI UNIVERSITY
地址：中国上海市四平路1239号 邮编：200092
1239 SIPING ROAD SHANGHAI CHINA 200092
电话(TEL):+86 21- 传真(FAX)::+86 21-
网址(WEB):www.tongji.edu.cn

HW-L7. FSM.

1. ① Moore FSM. — 1101



② Mealy FSM. — 1101 ☆.



4. Binary Encoding.

① State S1 S0.
S0 0 0.
S1 0 1
S2 1 0
S3 1 1.

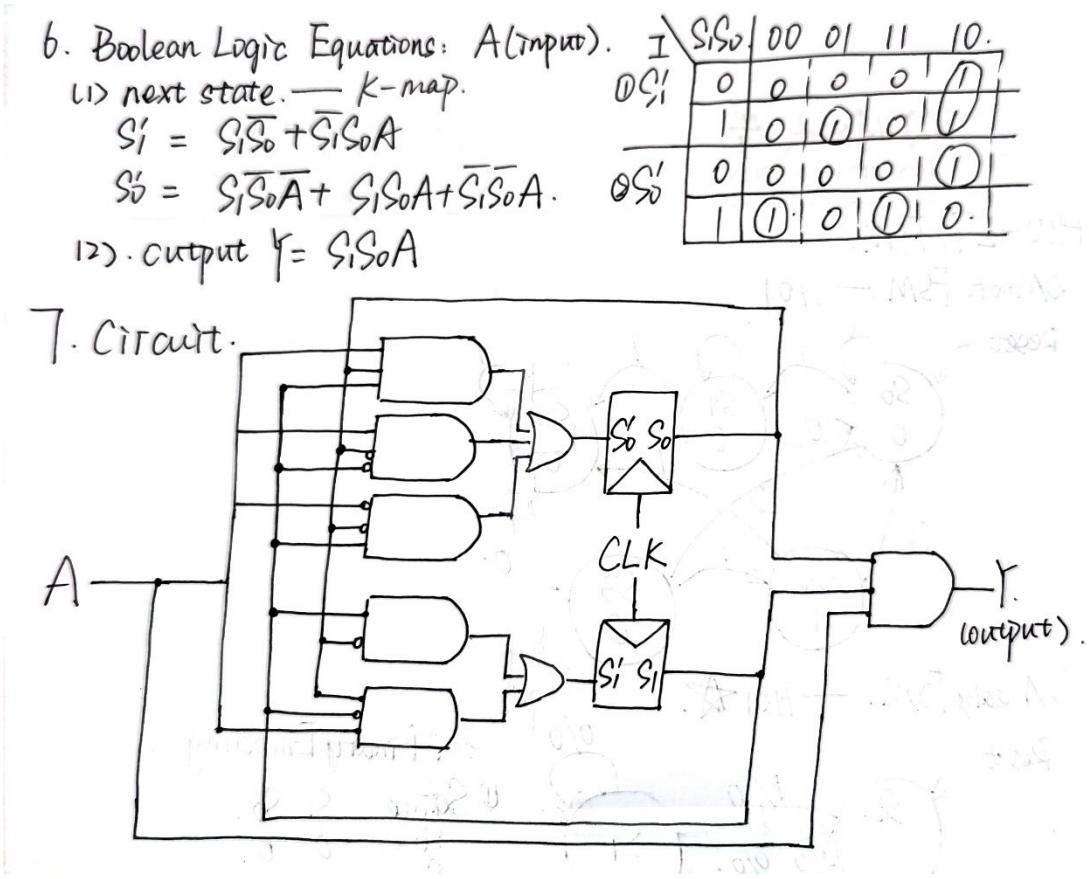
② Output → 0/1.

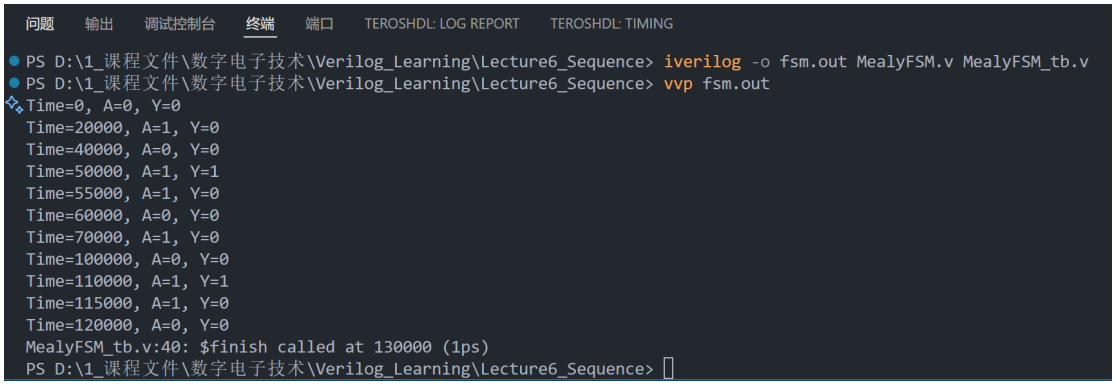
2. State Transition Table (Mealy).

S	Input	S'	Output.
S0	0	S0	0
S0	1	S1	0
S1	0	S0	0
S1	1	S2	0
S2	0	S3	0
S2	1	S2	0
S3	0	S0	0
S3	1	S1	1

5. Binary Encoding Table.

State	Input	State'	Output.
00	0	00	0
00	1	01	0
01	0	00	0
01	1	10	0
10	0	11	0
10	1	10	0
11	0	00	0
11	1	01	1





Here are the snapshots of my MealyFSM.

In the sequence 12'b011010111010 ,it turns 1 from 0 at the 5th and 11th clk.

```
module MealyFSM (
    input wire clk,
    input wire reset,
    input wire A,
    output reg Y );

    parameter S0 = 2'b00;
    parameter S1 = 2'b01;
    parameter S2 = 2'b10;
    parameter S3 = 2'b11;
    reg [1:0] Scurr;
    reg [1:0] Snext;

    // 时序
    always @(posedge clk or posedge reset)
begin   if (reset) Scurr <= S0;
        else Scurr <= Snext;
    end

    // 组合
    always @(*) begin
        case (Scurr)
            S0: begin
                if (A == 1'b0) begin
                    Snext = S0;
                    Y = 1'b0;
                end else begin
                    Snext = S1;
                    Y = 1'b0;
                end
            end
            S1: begin
                if (A == 1'b0) begin
                    Snext = S0;
                    Y = 1'b0;
                end else begin
                    Snext = S2;
                    Y = 1'b0;
                end
            end
            S2: begin
                if (A == 1'b0) begin
                    Snext = S3;
                    Y = 1'b0;
                end else begin
                    Snext = S2;
                    Y = 1'b0;
                end
            end
            S3: begin
                if (A == 1'b0) begin
                    Snext = S0;
                    Y = 1'b0;
                end else begin
                    Snext = S1;
                    Y = 1'b1;
                end
            end
            default: begin
                Snext = S0;
                Y = 1'b0;
            end
        endcase
    end
endmodule
```

```
`timescale 1ns / 1ps
module tb_MealyFSM;

    reg clk;
    reg reset;
    reg A;
    wire Y;

    MealyFSM dut (
        .clk(clk),
        .reset(reset),
        .A(A),
        .Y(Y)
    );
    initial begin
        $dumpfile("MealyFSM.vcd");
        $dumpvars(0, tb_MealyFSM);
    end

    always begin
        #5 clk = ~clk;
    end
    // 复位
    initial begin
        reset = 1;#10; // 初始复位
        reset = 0;
    end

    // 输入序列
    reg [11:0] inputseq = 12'b011010111010;
    integer i;

    initial begin
        // 初始化
        clk = 0;
        A = 0;
        wait (reset == 0);
        // 逐位提供输入序列
        for (i = 11; i >= 0; i = i - 1) begin
            A = inputseq[i];
            #10;
        end

        $finish;
    end
    initial begin
        $monitor("Time=%0t, A=%b, Y=%b", $time, A,
Y); end

endmodule
```