

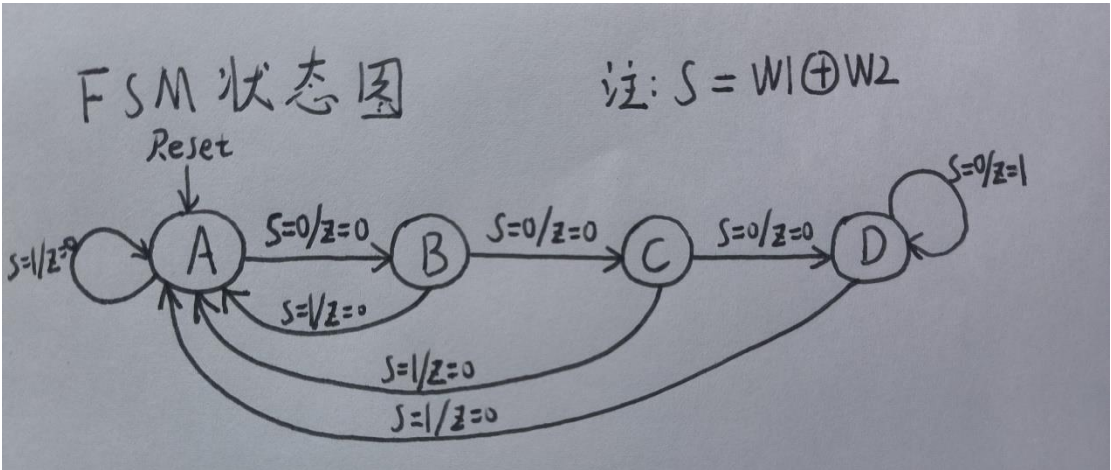
第三次作业设计报告

姓名：梁桐
学号：2023370018
班级：10012209

题目：一个时序电路有两个输入 w_1 , w_2 和一个输出 z ，并实现对两个输入的输入序列进行比较。如果在任意 4 个连续时钟周期中 $w_1=w_2$ ，电路产生输出 $z=1$ ，否则 $z=0$ 。输入输出特性如下：
 w_1 :0110111000110
 w_2 :1110101000111
 z :0000100001110
设计并实现上述功能电路。

1. FSM 状态图

分析：已知判断 z 的条件取决于 $w_1=w_2$ 的连续次数，则可令 $s = w_1 \oplus w_2$ ，从而简化思路。



2. 状态表

现态	次态		输出	
	$S=0$	$S=1$	$S=0$	$S=1$
A	B	A	0	0
B	C	A	0	0
C	D	A	0	0
D	D	A	1	0

3. 状态分配表

现态	次态		输出 z	
	S=0	S=1	S=0	S=1
$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	z	z
00	01	00	0	0
01	10	00	0	0
10	11	00	0	0
11	11	00	1	0

4. 输出和次态逻辑表达式

$$Y_1 = \bar{S} \bar{y}_1 + \bar{S} y_2$$

$$Y_2 = \bar{S} y_1 + \bar{S} y_2$$

$$z = \bar{S} y_1 y_2$$

$y_2 y_1$	00	01	11	10
S=0	0	1	1	1
S=1	0	0	0	0

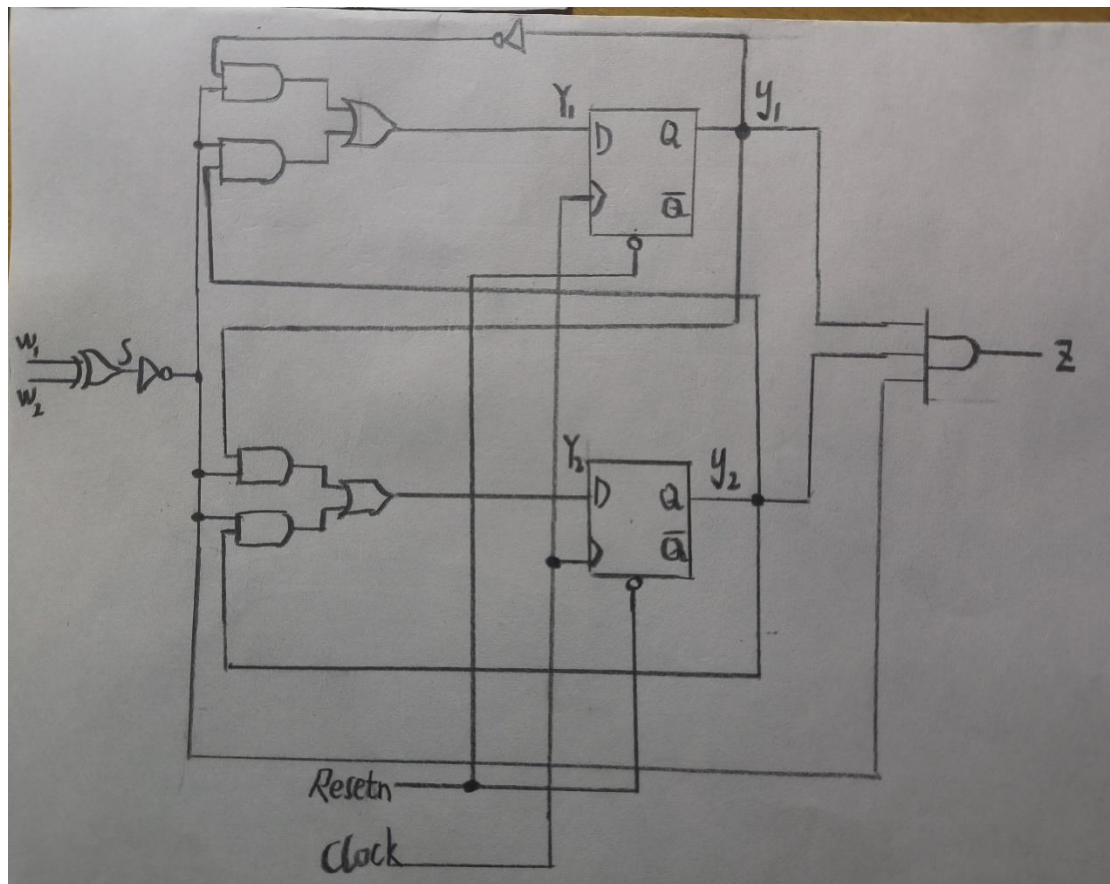
$y_2 y_1$	00	01	11	10
S=0	1	0	1	1
S=1	0	0	0	0

$Y_2 = \bar{S} y_1 + \bar{S} y_2$
 $Y_1 = \bar{S} \bar{y}_1 + \bar{S} y_2$

$y_2 y_1$	00	01	11	10
S=0	0	0	1	0
S=1	0	0	0	0

$$z = \bar{S} y_2 y_1$$

5. 电路图



6. Verilog 代码

6.1 testbench 代码

```
`timescale 1ns/1ps

module code6_9_testbench;

    reg clk_test, Resetn_test, w1_test, w2_test;
    wire z_test;

    initial begin
        clk_test = 0;
        Resetn_test = 0;
        w1_test = 0;
        w2_test = 0;
    end

    always #10 clk_test  = ~clk_test ;

    initial begin
```

```

#10;
Resetn_test = 1;
w1_test = 0;
w2_test = 0;
#20 w1_test = 1;w2_test = 0;
#20 w1_test = 0;w2_test = 1;
#20 w1_test = 1;w2_test = 0;
#20 w1_test = 0;w2_test = 1;
//90ns 之后是题目示例数据
#20 w1_test = 0;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 1;w2_test = 1;//此处 z 开始为 1
#20 w1_test = 1;w2_test = 0;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 0;w2_test = 0;//此处 z 开始为 1
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 0;w2_test = 1;

#20 w1_test = 1;w2_test = 0;
#20 w1_test = 0;w2_test = 1;
#20 w1_test = 1;w2_test = 0;
#20 w1_test = 0;w2_test = 1;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 1;w2_test = 1;

#20 w1_test = 0;w2_test = 1;
Resetn_test = 0;//测试复位信号
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 0;w2_test = 0;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 1;w2_test = 1;
#20 w1_test = 0;w2_test = 0;
end

```

```

code6_9 UUT_code6_9 (
    .Clock(clk_test),
    .Resetn(Resetn_test),
    .w1(w1_test),
    .w2(w2_test),
    .z(z_test)
);
endmodule

```

6.2 检测器模块代码

```

module code6_9(Clock, Resetn, w1, w2, z);

    input Clock, Resetn, w1, w2;
    output reg z;
    reg [2:1] y, Y;
    wire s;
    // 状态定义
    parameter [2:1] A = 2'b00, B = 2'b01, C = 2'b10, D = 2'b11;

    // 定义下一个状态和输出的组合电路
    assign s = w1 ^ w2;

    // 状态迁移逻辑
    always @(s, y)
        case (y)
            // 状态 A
            A: if (s) begin
                    Y = A; // 下一个状态为 A
                    z = 0; // 输出 z 为 0
                end
            else begin
                    Y = B; // 下一个状态为 B
                    z = 0; // 输出 z 为 0
                end
            // 状态 B
            B: if (s) begin
                    Y = A; // 下一个状态为 A
                    z = 0; // 输出 z 为 0
                end
            else begin
                    Y = C; // 下一个状态为 C

```

```

        z = 0; // 输出 z 为 0
    end
// 状态 C
C: if (s) begin
    Y = A; // 下一个状态为 A
    z = 0; // 输出 z 为 0
end
else begin
    Y = D; // 下一个状态为 D
    z = 0; // 输出 z 为 0
end
// 状态 D
D: if (s) begin
    Y = A; // 下一个状态为 A
    z = 0; // 输出 z 为 0
end
else begin
    Y = D; // 下一个状态为 D
    z = 1; // 输出 z 为 1
end
endcase

// 定义时序块
always @(negedge Resetn or posedge Clock)
    if (Resetn == 0)
        y <= A; // 复位时将状态置为 A
    else
        y <= Y; // 根据组合逻辑计算得到的下一个状态更新 y
endmodule

```

6.3 仿真波形图（从第 90ns 起是题目所给示例数据）

