

## 实验五 使用状态机实现数字逻辑电路

第 1 步： 5 分

实验目的：（简要写出）

- (1) 理解状态机的基本概念和应用场合。
- (2) 掌握状态机的 verilog 实现代码。
- (3) 学习分析并应用状态机来描述实现逻辑电路。

第 2 步： 5 分

实验原理：（简要写出）

时序电路在工作时通常是在电路的有限个状态间按一定的规律转换的, 所以有时也将时序电路称为有限状态机, 简称状态机。

另外也可把状态机理解为一种思想方法, 即描述有限多个状态、以及在这些状态之间转移和动作的数学模型。

状态机的基本要素有 3 个, 分别是状态、输出和输入。

状态: 也叫状态变量。在逻辑设计中, 使用状态划分逻辑顺序和时序规律。时序电路的状态是一个状态变量集合, 这些状态变量在任意时刻的值都包含了为确定电路的未来行为而必须考虑的所有历史信息。具有  $n$  位二进制状态变量的电路就有  $2^n$  种可能的状态。

输出: 指在某一个状态时特定发生的事件。

输入: 指状态机中进入每个状态的条件, 有的状态机没有输入条件, 其中的状态转移较为简单, 有的状态机有输入条件, 当某个输入条件存在时才能转移到相应的状态。

根据状态机的输出是否与输入条件相关, 可将状态机分为两大类, 即 Moore 型状态机和 Mealy 型状态机。

Moore 型状态机: Moore 型状态机的输出仅仅依赖于当前状态, 而与输入条件无关。

Mealy 型状态机: Mealy 型状态机的输出不仅依赖于当前状态, 而且取决于该状态的输入条件。

第 3 步： 8 分

预习题（截图或拍照）

## 实验五 使用状态机实现数字逻辑电路

专业-班级: 自动化1班

学号: 210320111

姓名: 吕家良

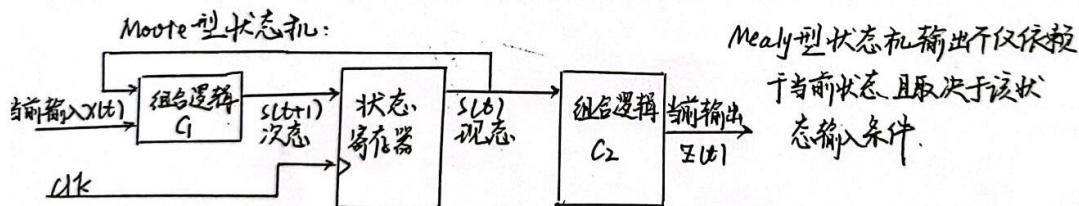
实验检查记录表:

预习	一	必做完成时间

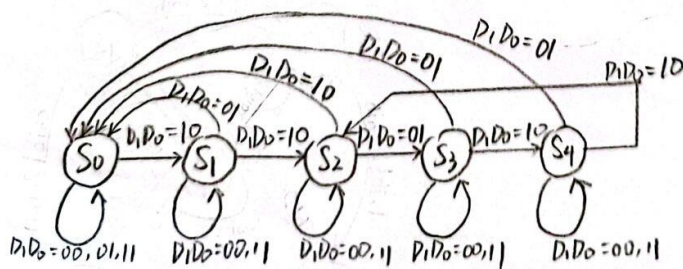
1. 填空: 状态机的基本要素分别是 状态、输入 和 输出。
2. 填空: 根据 状态机输出是否与输入有关, 可将状态机分为两大类, 即 Moore 型状态机和 Mealy 型状态机。
3. 填空: 指导书中推荐的使用 HDL 语言描述状态机的写法是: 使用 3 个 always 模块, 一个 always 模块 采用同步时序的方式描述状态转移; 一个 always 模块 采用组合逻辑的方式判断状态转移条件; 第三个 always 模块 描述每个状态的输出。
4. 请简述什么是状态机。

状态机是描述有限多个状态, 及在状态之间转移和动作的数学模型

5. 请画出 Moore 型状态机的结构框图。Mealy 型状态机和它有什么区别?



6. 实验内容 5.5.1 使用状态机设计实现 1101 序列检测器。请分析并画出该序列检测器的状态转换图。



第 5 步: 15 分

### 5.5.1 使用状态机设计实现 1101 序列检测器

- (1) 简述实验现象。

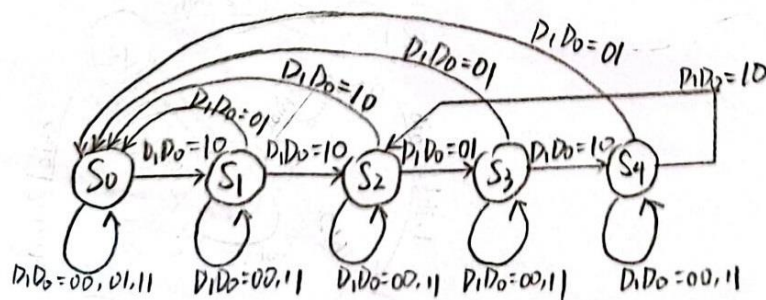
本实验使用 LD2\_4~0 五个 LED 灯分别显示状态 4~0。  
 当处于状态 0 时，依次按下 1101，状态依次变化至 1,2,3,4。  
 当按下错误的按键，转换至状态 0。  
 状态 4 下若按下 S1，进入状态 2。(1101101 后面的 1101 可被检测)  
 在任意状态下，按下复位按键 S4，立刻恢复状态 0。

第 6 步：15 分

### 5.5.1 使用状态机设计实现 1101 序列检测器

#### (2) 状态转换图 (拍照或截图)

6. 实验内容 5.5.1 使用状态机设计实现 1101 序列检测器。请分析并画出该序列检测器的状态转换图。



第 7 步：30 分

### 5.5.1 使用状态机设计实现 1101 序列检测器

#### (3) verilog 源代码、约束文件 (拍照或截图)

```

23 module Clock_divide(
24     input clk,
25     output divclk
26 );
27 //产生200Hz时钟信号
28 reg[19:0] divclk_reg;
29 reg out;
30
31 always@(posedge clk)
32 begin
33     if(divclk_reg < 500000 - 1)
34     begin
35         divclk_reg <= divclk_reg + 1;
36         out <= 0;
37     end
38     else
39     begin
40         divclk_reg <= 0;
41         out <= 1;
42     end
43 end
44 assign divclk = out;
45 endmodule
46

```

```

22 //有按键按下即输出脉冲
23 module Key_Pulse(
24     input divclk,
25     input din0, din1,
26     output dout
27 );
28 reg[2:0] in_reg, in_reg_last;
29 reg out_reg;
30
31 always@(posedge divclk)
32 begin
33     in_reg_last = in_reg;
34     in_reg[2] = in_reg_last[1];
35     in_reg[1] = in_reg_last[0];
36     if(din0 || din1) in_reg[0] = 1;
37     else in_reg[0] = 0;
38     if(in_reg == 3'b011) out_reg = 1;
39     else out_reg = 0;
40 end
41 assign dout = out_reg;
42 endmodule
43

```

```

23 module Sequence(
24     input clk,
25     input din0, din1,
26     input clr,
27     output[4:0] out
28 );
29 wire cclk;
30 wire keyout;
31 reg[2:0] state = 0;
32 reg[4:0] out_reg;
33 Clock_divide clock_divide(.clk(clk), .divclk(cclk));
34 Key_Pulse key_pulse(.din0(din0), .din1(din1), .divclk(cclk), .dout(keyout));
35
36 always@(posedge keyout or posedge clr)
37 begin
38     if(clr) state <= 3'b000;
39     else
40         begin
41             case(state)
42             0:
43                 begin
44                     if(din1 && ~din0) state <= 3'b001;
45                     else if(~din1 && din0) state <= 3'b000;
46                     else state <= state;
47                 end
48             1:
49                 begin
50                     if(din1 && ~din0) state <= 3'b010;
51                     else if(~din1 && din0) state <= 3'b000;
52                     else state <= state;
53                 end
54             2:
55                 begin
56                     if(din1 && ~din0) state <= 3'b000;
57                     else if(~din1 && din0) state <= 3'b011;
58                     else state <= state;
59                 end
60             3:
61                 begin
62                     if(din1 && ~din0) state <= 3'b100;
63                     else if(~din1 && din0) state <= 3'b000;
64                     else state <= state;
65                 end
66             4:
67                 begin
68                     if(din1 && ~din0) state <= 3'b010;
69                     else if(~din1 && din0) state <= 3'b000;
70                     else state <= state;
71                 end
72             default: state <= 3'b000;
73             endcase
74         end
75     end
76     always@(state)
77     begin
78         out_reg = 1 << state;
79     end
80     assign out = out_reg;
81 endmodule

```



```

1  set_property PACKAGE_PIN P17 [get_ports clk];
2  set_property PACKAGE_PIN R17 [get_ports din1];
3  set_property PACKAGE_PIN R11 [get_ports din0];
4  set_property PACKAGE_PIN U4 [get_ports clr];
5  set_property PACKAGE_PIN J4 [get_ports out[4]];
6  set_property PACKAGE_PIN H4 [get_ports out[3]];
7  set_property PACKAGE_PIN J3 [get_ports out[2]];
8  set_property PACKAGE_PIN J2 [get_ports out[1]];
9  set_property PACKAGE_PIN K2 [get_ports out[0]];
10
11 set_property IOSTANDARD LVCMOS33 [get_ports clk];
12 set_property IOSTANDARD LVCMOS33 [get_ports din1];
13 set_property IOSTANDARD LVCMOS33 [get_ports din0];
14 set_property IOSTANDARD LVCMOS33 [get_ports clr];
15 set_property IOSTANDARD LVCMOS33 [get_ports out[4]];
16 set_property IOSTANDARD LVCMOS33 [get_ports out[3]];
17 set_property IOSTANDARD LVCMOS33 [get_ports out[2]];
18 set_property IOSTANDARD LVCMOS33 [get_ports out[1]];
19 set_property IOSTANDARD LVCMOS33 [get_ports out[0]];
20
21 set_property BITSTREAM.CONFIG.UNUSEDPIN PULLNONE [current_design];

```

## 实验思考

### 第 1 题： 20 分

(1) 请画出时序图分析说明图 5-6 所示电路为何可在按键按下时产生一个单脉冲信号？

当按下按键时，Q1, Q2, Q3 先后跳变至高电平，松开时跳变至低电平。由于 3 个触发器依次相差 1clk，因此存在 1clk 使  $Q1Q2Q3'=1$ ，产生 1clk 的脉冲信号。

由于产生脉冲信号的条件为按键按下时长大于 2clk，故能达到消抖效果。

