

串行密码锁 实验报告

2018011365 张鹤潇

一、实验目的

- 学会使用状态机来控制电路工作，在不同的状态下完成相应的功能；
- 进一步掌握时序逻辑电路的基本分析和设计方法；
- 利用软件仿真对数字电路的逻辑功能进行验证和分析。

二、实验任务

设计一个四位16进制串行电子密码锁，其具体功能如下：

- 设置密码：用户串行设置四位16进制密码；
- 验证密码：用户串行输入密码，如密码符合则点亮开锁灯，否则点亮错误灯；
- 密码预置：为管理员创建万用密码，以备管理；
- 系统报警：开锁三次失败后点亮报警灯，锁定密码锁，只有输入管理员密码才可开锁。

三、代码实现

密码锁entity定义如下：

```
entity codedLock is
    port(
        rst, clk: in std_logic;
        code: in std_logic_vector(3 downto 0);
        mode: in std_logic; -- 0: set pwd; 1: verify pwd
        unlocked, err, alarm: out std_logic
    );
    type IntArray4 is array (3 downto 0) of integer;
end codedLock;
```

architecture实现如下：

```
architecture lock of codedLock is
    signal pwd: IntArray4;
    signal state: integer := 0;
    signal cnt: integer := 0; -- 用户密码输入错误次数/管理员密码正确位数
    signal alarm_signal: std_logic := '0';
    signal err_signal: std_logic := '0';
begin
    process(clk, rst)
    begin
        if (rst = '1') then
            unlocked <= '0';
```

```
err_signal <= '0';
state <= 0;
if (alarm_signal = '1') then -- 密码错误次数不清零
    cnt <= 0;
end if;
elsif (clk'event and clk = '1') then -- clk上升沿读密码
    if (alarm_signal = '1') then -- 锁定状态
        if (CONV_INTEGER(code) = 1) then -- 管理员密码 1111
            if (cnt < 3) then
                cnt <= cnt + 1;
            else -- 管理员验证成功, 清除alarm
                cnt <= 0;
                alarm_signal <= '0';
                state <= 0;
            end if;
        else
            cnt <= 0;
        end if;
    elsif (mode = '0') then -- set pwd
        case state is
            when 0 =>
                pwd(0) <= CONV_INTEGER(code); state <= 1;
            when 1 =>
                pwd(1) <= CONV_INTEGER(code); state <= 2;
            when 2 =>
                pwd(2) <= CONV_INTEGER(code); state <= 3;
            when 3 =>
                pwd(3) <= CONV_INTEGER(code); state <= 7; unlocked <= '1';
            when others => NULL;
        end case;
    elsif (mode = '1') then -- varify pwd
        case state is
            when 0 => -- varify 1st bit
                if (CONV_INTEGER(code) = pwd(0)) then
                    state <= 4;
                    err_signal <= '0';
                else
                    err_signal <= '1';
                    if (cnt < 2) then
                        cnt <= cnt + 1;
                    else -- fails 3 times, set err = 1
                        alarm_signal <= '1';
                        cnt <= 0;
                    end if;
                end if;
            when 4 => -- varify 2nd bit
                if (CONV_INTEGER(code) = pwd(1)) then
                    state <= 5;
                    err_signal <= '0';
                else
                    err_signal <= '1';
                    state <= 0;
                    if (cnt < 2) then
                        cnt <= cnt + 1;
                    else -- fails 3 times, set err = 1
```

```

        alarm_signal <= '1';
        cnt <= 0;
    end if;
end if;
when 5 => -- varify 3rd bit
    if (CONV_INTEGER(code) = pwd(2)) then
        state <= 6;
        err_signal <= '0';
    else
        err_signal <= '1';
        state <= 0;
        if (cnt < 2) then
            cnt <= cnt + 1;
        else -- fails 3 times, set err = 1
            alarm_signal <= '1';
            cnt <= 0;
        end if;
    end if;
when 6 => -- varify 4th bit
    if (CONV_INTEGER(code) = pwd(3)) then
        state <= 7;
        err_signal <= '0';
        unlocked <= '1';
    else
        err_signal <= '1';
        state <= 0;
        if (cnt < 2) then
            cnt <= cnt + 1;
        else -- fails 3 times, set err = 1
            alarm_signal <= '1';
            cnt <= 0;
        end if;
    end if;
when others => NULL;
end case;
end if;
end if;
end process;

process(alarm_signal)
begin
    alarm <= alarm_signal;
end process;

process(err_signal)
begin
    err <= err_signal;
end process;
end;

```

使用状态机的方法实现串行密码锁，共定义了8种状态。

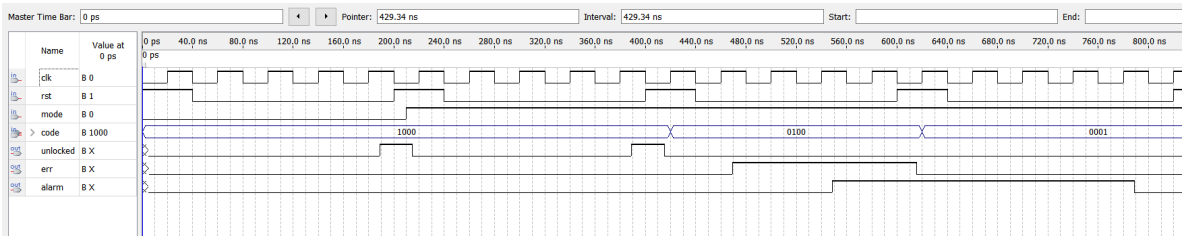
- 0号状态：初始状态，如果mode=0，设置第一位密码，进入1号状态；如果mode是1，判断第一位密码是否正确，正确则进入4号状态，错误则留在0号状态。
- 1号状态：如果mode=0，设置第二位密码，进入2号状态。

- 2号状态：如果mode=0，设置第三位密码，进入3号状态。
- 3号状态：如果mode=0，设置第四位密码，进入7号状态。
- 4号状态：如果mode=1，判断第二位密码是否正确，正确则进入5号状态，错误则回到0号状态。
- 5号状态：如果mode=1，判断第三位密码是否正确，正确则进入6号状态，错误则回到0号状态。
- 6号状态：如果mode=1，判断第四位密码是否正确，正确则进入7号状态，错误则回到0号状态。
- rst=1将使得状态机回到0号状态；连续三次密码验证错误，状态机将停留在0号状态，直到输入管理员密码1111为止。

四、 仿真结果

本机仿真

用Modelsim-Altera在本机上进行Time simulation.



仿真流程：

1. 设置密码8888， unlocked变为1；按下rst， unlocked复原；
2. 验证正确密码8888， unlocked变为1；按下rst， unlocked复原；
3. 验证错误密码4444， 错误1次后err变为1， 错误三次后alarm变为1；
4. 按下rst， err=0， 而alarm不变，说明系统处于锁定状态；
5. 输入管理员密码1111， alarm警报消失。

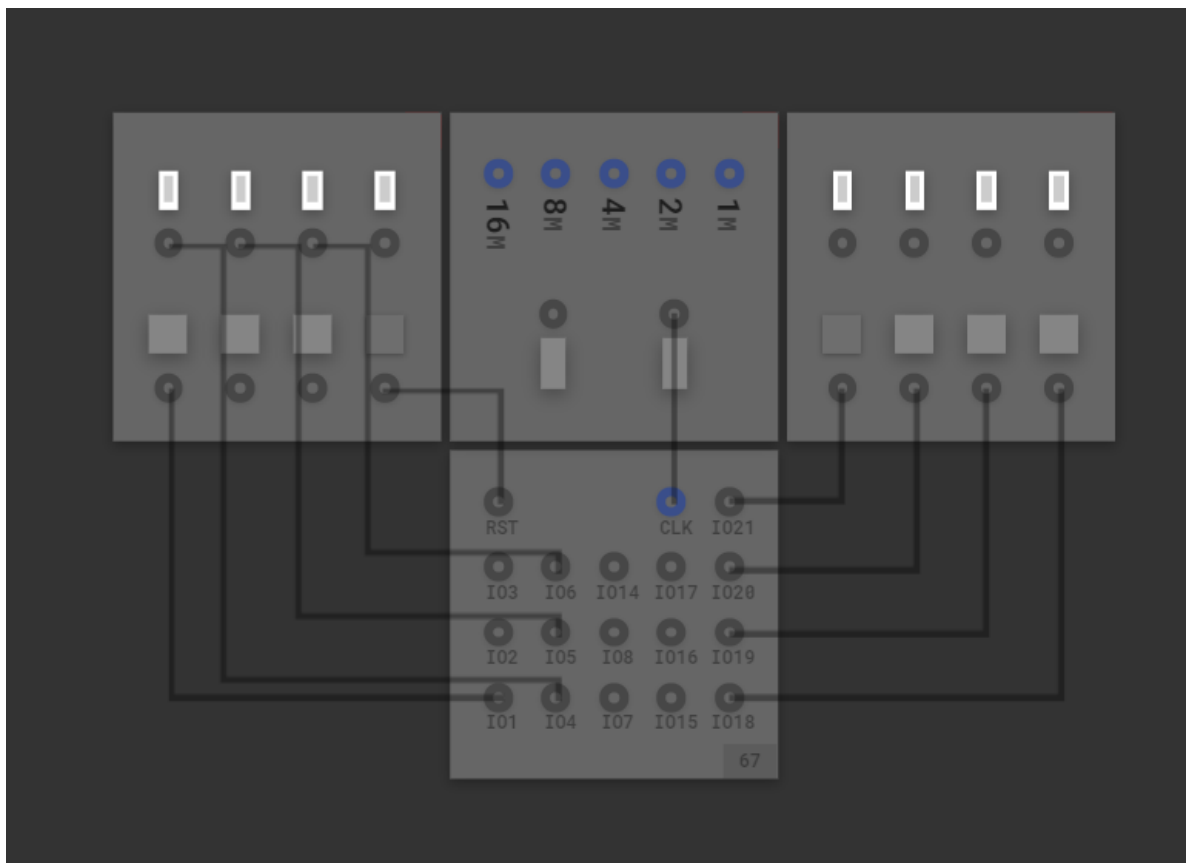
JieLab模拟实验

令mode=0， 设置密码8888, unlocked信号灯亮起。

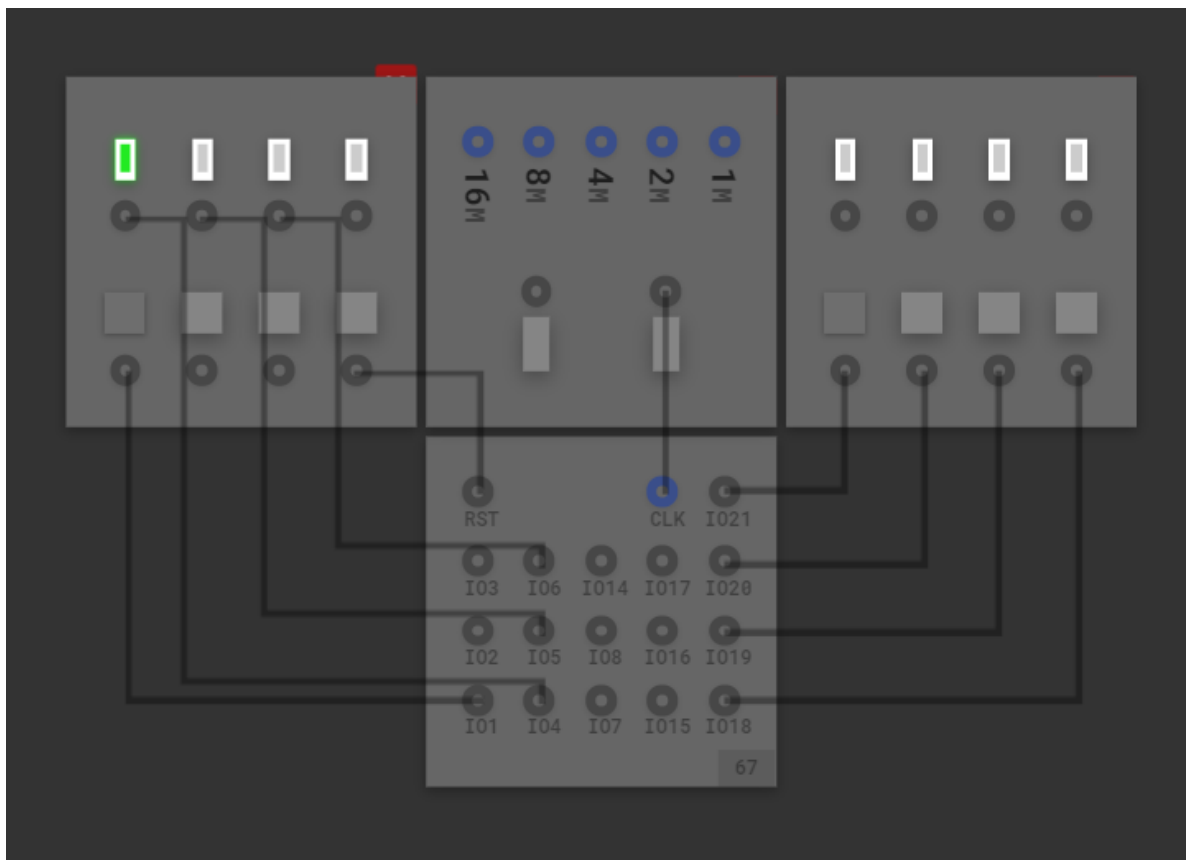
A breadboard simulation in JieLab showing the implementation of the digital circuit. It includes logic components like flip-flops, multiplexers, and comparators, connected to represent the state machine logic. LEDs are used to indicate the unlocked, error, and alarm states.

```
1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.std_logic_arith.all;
4 use ieee.std_logic_unsigned.all;
5
6 entity codedLock is
7     port(
8         rst, clk: in std_logic;
9         code: in std_logic_vector(3 downto 0);
10        mode: in std_logic; -- 0: set pwd; 1: verify pwd
11        unlocked, err, alarm: out std_logic
12    );
13    type IntArray4 is array (3 downto 0) of integer;
14 end codedLock;
15
16 architecture lock of codedLock is
17     signal pwd: IntArray4;
18     signal state: integer := 0;
19     signal cnt: integer := 0; -- 用户密码输入错误次数/管理员密码正确位数
20     signal alarm_signal: std_logic := '0';
21     signal err_signal: std_logic := '0';
22 begin
23     process(clk, rst)
24     begin
25         if (rst = '1') then
26             unlocked <= '0';
```

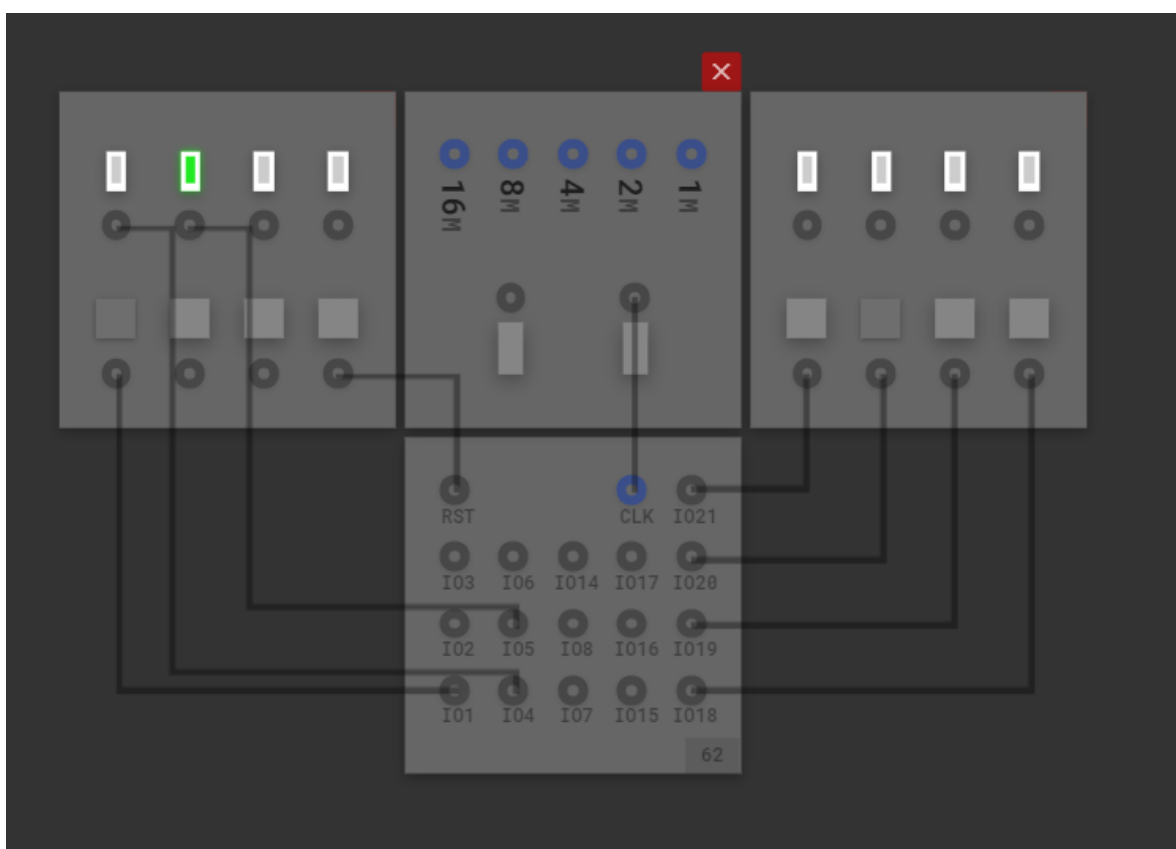
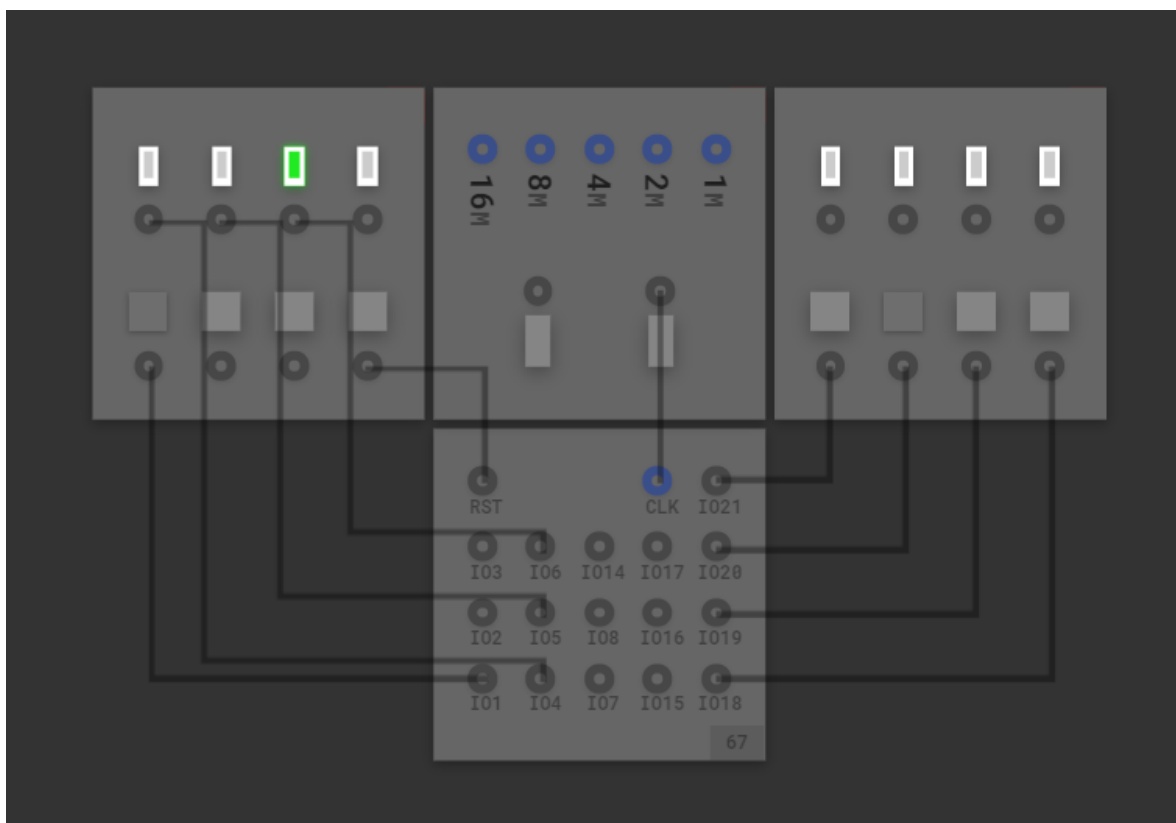
按下rst， unlocked信号灯熄灭。



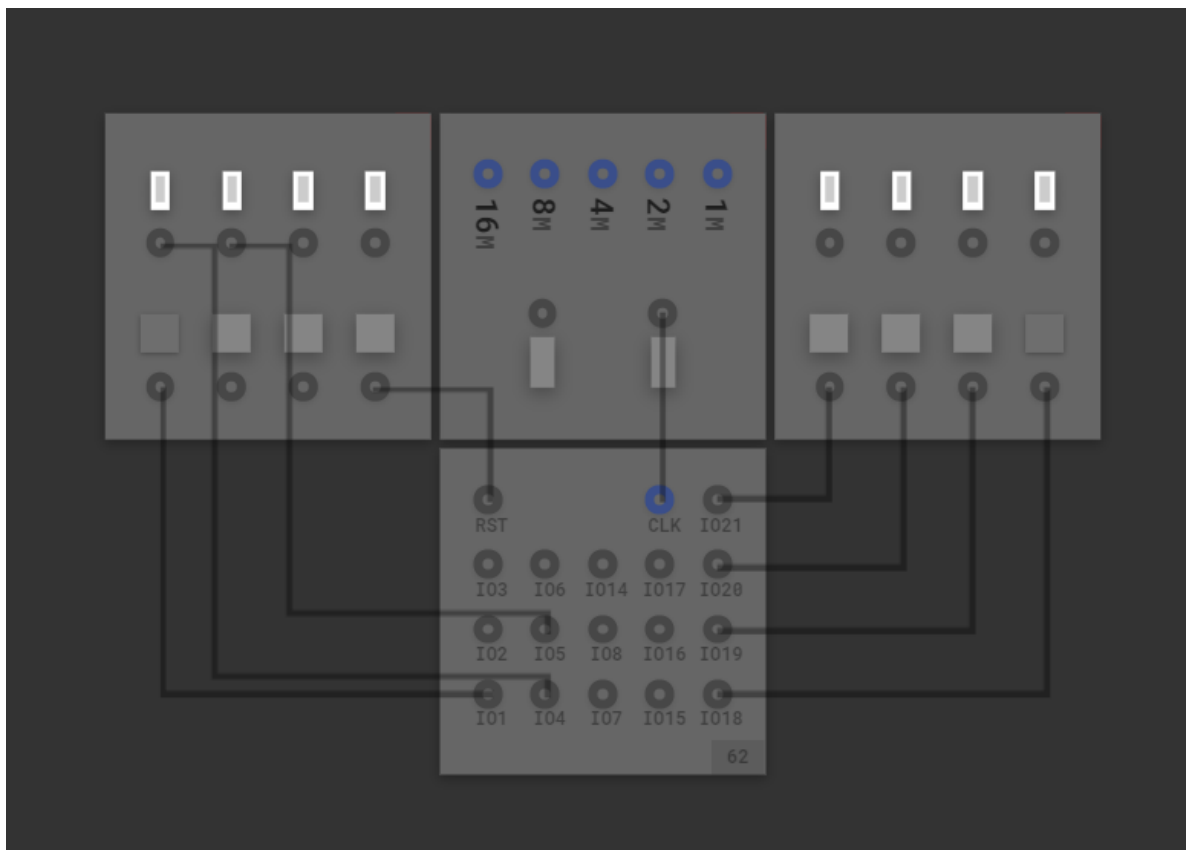
令mode=1, 验证密码8888, unlocked信号灯亮起。



按下rst; 再输入错误密码4, err信号亮起; 连续错误三次, alarm信号亮起。



输入管理员密码，alarm熄灭。



五、总结反思

这是我本学期第四次CPLD实验，感谢老师助教为这门课付出的心血，感谢Jielab平台的相关开发人员。经过本次实验，我对如何通过VHDL实现状态机有了更深入的了解。