串行密码锁

2017011620 计73 李家昊

实验目的

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

实验任务

基础内容

设计一个4位十六进制串行电子密码锁，具体功能如下：

1. 设置密码。用户串行设置4位十六进制密码。
2. 验证密码。用户串行输入密码，如果密码符合则点亮开锁灯，若不符合则点亮错误灯。

研究内容

1. 密码预置。为管理员创建万用密码以备管理。
2. 系统报警。开锁3次失败点亮报警灯，并锁定密码锁，只有输入管理员密码才可开锁，并解除报警。

代码分析

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity lock is
    port(
        rst: in std_logic;
        clk: in std_logic;
        code: in std_logic_vector(3 downto 0);
        mode: in std_logic_vector(1 downto 0);
        unlock: out std_logic;
        err: out std_logic;
        alarm: out std_logic
    );
end lock;

architecture state of lock is
    type password is array(3 downto 0) of std_logic_vector(3 downto 0);
    type states is (s0, s1, s2, s3, s4);
    signal state: states:= s0;
```

```
    signal pwd: password;
    signal admin_pwd: password:=("1111", "1111", "1111", "1111");
    signal cnt: integer:=0;
    signal admin_cnt: integer:= 0;
begin
    process(clk, rst)
    begin
        if(rst = '1') then
            state <= s1;
            if(cnt < 3) then
                unlock <= '0';
                err <= '0';
            end if;
        elsif (rising_edge(clk)) then -- clock
            if(cnt > 2) then -- cnt is 3. ban all usage except admin password.
                if(code = admin_pwd(admin_cnt)) then -- good bit
                    case state is
                    when s1 => state <= s2; admin_cnt <= admin_cnt + 1;
                    when s2 => state <= s3; admin_cnt <= admin_cnt + 1;
                    when s3 => state <= s4; admin_cnt <= admin_cnt + 1;
                    when s4 =>
                        state <= s0;
                        cnt <= 0;
                        alarm <= '0';
                        err <= '0';
                        unlock <= '1';
                        admin_cnt <= 0;
                    when others=> NULL;
                    end case;
                else -- bad bit
                    state <= s0;
                    admin_cnt <= 0;
                end if;
            else
                case mode is
                when "00" => -- set password
                    case state is
                    when s1 => pwd(0) <= code; state <= s2;
                    when s2 => pwd(1) <= code; state <= s3;
                    when s3 => pwd(2) <= code; state <= s4;
                    when s4 => pwd(3) <= code; state <= s0; unlock <= '1'; cnt <= 0;
                    when others => NULL;
                    end case;
                when "01" => -- verify password
                    case state is
                    when s1 =>
                        if (code = pwd(0)) then
                            state <= s2;
                        else
                            state <= s0;
                            err <= '1';
                            if(cnt > 1) then
                                alarm <= '1';
```

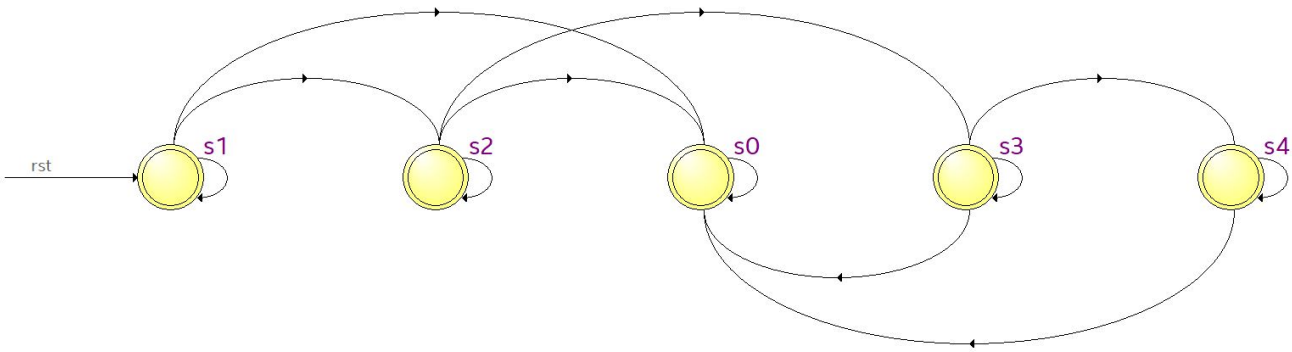
```
                    end if;
                    cnt <= cnt + 1;
                end if;
            when s2 =>
                if (pwd(1) = code) then
                    state <= s3;
                else
                    state <= s0;
                    err <= '1';
                    if(cnt > 1) then
                        alarm <= '1';
                    end if;
                    cnt <= cnt + 1;
                end if;
            when s3 =>
                if (pwd(2) = code)then
                    state <= s4;
                else
                    state <= s0;
                    err <= '1';
                    if(cnt > 1) then
                        alarm <= '1';
                    end if;
                    cnt <= cnt + 1;
                end if;
            when s4 =>
                if(pwd(3) = code) then
                    state <= s0;
                    unlock <= '1';
                    cnt <= 0;
                else
                    state <= s0;
                    err <= '1';
                    if(cnt > 1) then
                        alarm <= '1';
                    end if;
                    cnt <= cnt + 1;
                end if;
            when others => NULL;
            end case;
        when "10" => -- verify admin password
            case state is
            when s1 =>
                if (code = admin_pwd(0)) then
                    state <= s2;
                else
                    state <= s0;
                    err <= '1';
                    if(cnt > 1) then
                        alarm <= '1';
                    end if;
                    cnt <= cnt + 1;
                end if;
```

```
                    when s2 =>
                        if (code = admin_pwd(1)) then
                            state <= s3;
                        else
                            state <= s0;
                            err <= '1';
                            if(cnt > 1) then
                                alarm <= '1';
                            end if;
                            cnt <= cnt + 1;
                        end if;
                    when s3 =>
                        if (code = admin_pwd(2))then
                            state <= s4;
                        else
                            state <= s0;
                            err <= '1';
                            if(cnt > 1) then
                                alarm <= '1';
                            end if;
                            cnt <= cnt + 1;
                        end if;
                    when s4 =>
                        if(code = admin_pwd(3)) then
                            state <= s0;
                            unlock <= '1';
                            cnt <= 0;
                        else
                            state <= s0;
                            err <= '1';
                            if(cnt > 1) then
                                alarm <= '1';
                            end if;
                            cnt <= cnt + 1;
                        end if;
                    when others => NULL;
                    end case;
                when others => NULL;
                end case;
            end if;
        end if;
    end process;
end state;
```

使用状态机实现逻辑功能。内部的变量state记录了当前状态，根据当前状态及输入，完成状态转移，并输出相应的指示灯。

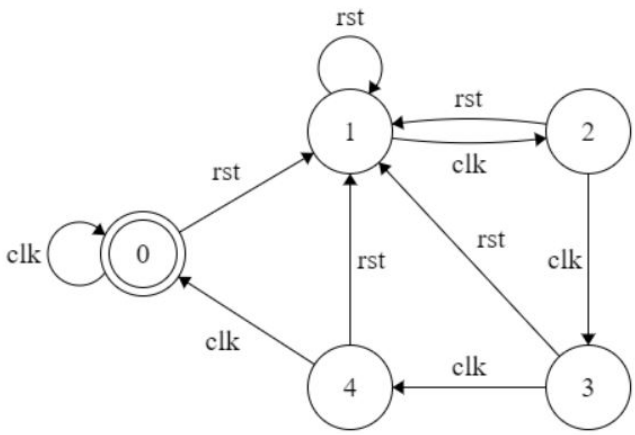
状态机分析

Quartus综合出来的状态机如下



设置密码模式

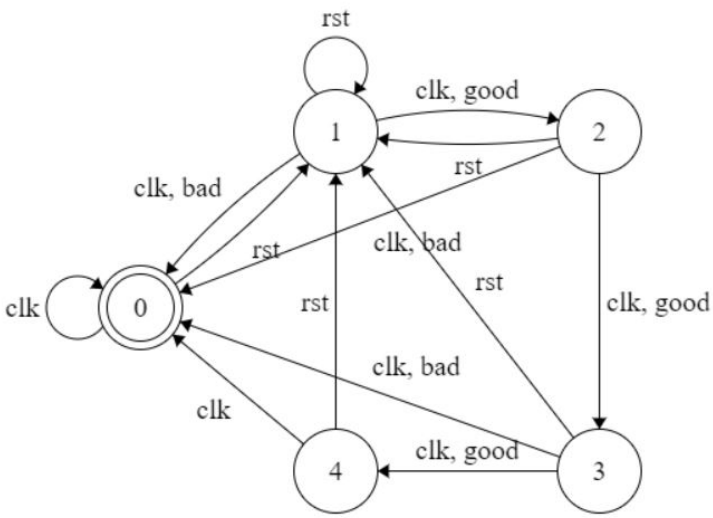
状态机如下



设置密码模式下，首先进入初始状态s0，按下rst后到达状态s1，开始设置密码，每按下一次clk设置一位密码，设置密码过程中若按下rst，则回到状态s1，按下4次clk后，设置完成，回到初始状态s0，并点亮开锁灯。

验证密码模式

状态机如下



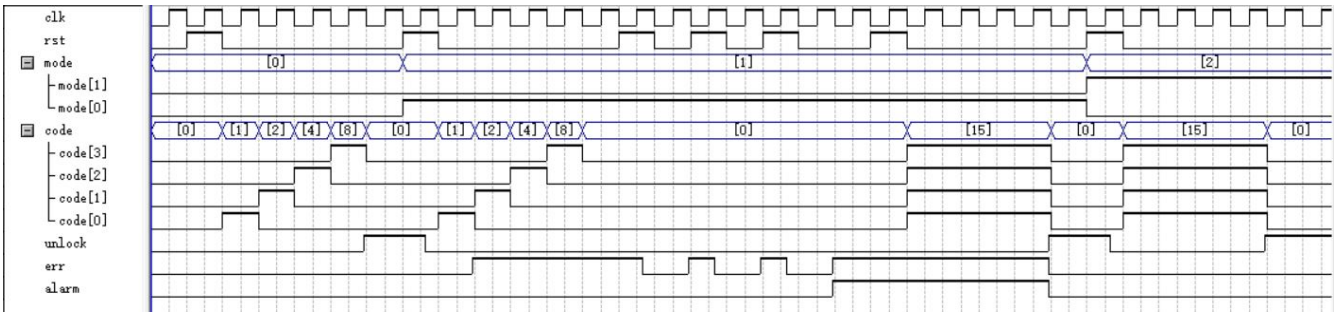
验证密码模式下，首先进入初始状态s0，按下rst后到达状态s1，开始验证密码，每按下一次clk验证一位密码，验证密码过程中若某一位发生错误，则回到状态s1，并点亮错误灯，正确验证4位密码后，验证成功，回到初始状态s0，并点亮开锁灯。

若三次输错密码，则点亮报警灯，并锁定密码锁，开启验证管理员密码模式。

验证管理员密码模式

执行逻辑与状态机与验证用户密码模式相同，这里不再赘述。

仿真结果



首先rst置1，mode置为00，为设置密码模式，设置密码为1248，设置完成后开锁灯点亮；然后mode置为01，为验证密码模式，串行输入密码为1248，验证成功，开锁灯点亮；然后输错三次密码，每次输错密码时，错误灯点亮，第三次输错密码时，错误灯、报警灯一同点亮，并锁定密码锁，接下来输入管理员密码FFFF，密码锁解除锁定，开锁灯点亮，错误灯、报警灯熄灭。最后mode置为10，为验证管理员密码模式，输入管理员密码，开锁灯点亮。

功能测试

实际电路能正常设置密码，验证用户密码，验证管理员密码，三次输入错误后报警，输入管理员密码后解锁，功能测试结果与仿真结果相符。

实验小结

1. 通过本次实验，我用代码实现了状态机，对状态转移逻辑有了更深刻的理解，加强了数字电路的设计能力。
2. 更加熟悉VHDL的各种语法，如type, array, case等语法的使用方法。
3. 更加熟练的掌握电路的仿真技巧，利用软件仿真对电路的功能进行验证和分析。
4. 感谢助教的耐心指导！