哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY

实验报告

开课学期：2022 秋季

课程名称：数字逻辑设计（实验）

实验名称：密码锁设计

实验性质：综合设计型

实验学时：6 地点：T2615

学生班级：6

学生学号：210110617

学生姓名：黄锦鹏

评阅教师：

报告成绩：

实验与创新实践教育中心制

2022 年 12 月

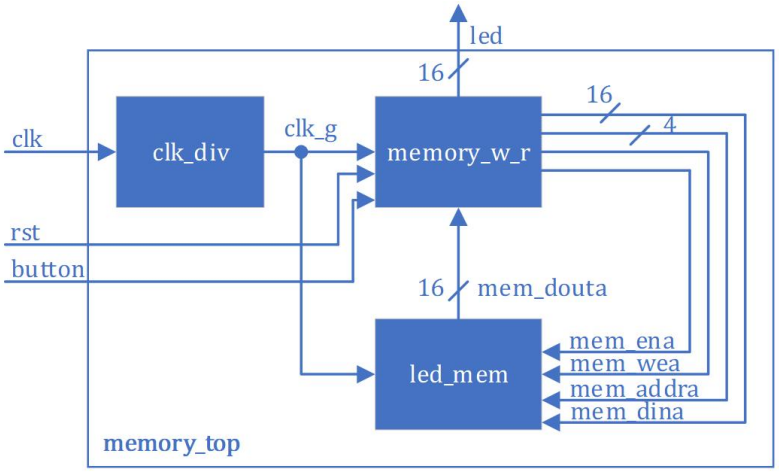
注：本设计报告中各个部分如果页数不够，请大家自行扩页，原则是一定要把报告写详细，能说明设计的成果和特色。报告中应该叙述设计中的每个模块。设计报告将是评定每个人成绩的重要组成部分（**设计内容及报告写作**都作为评分依据）。

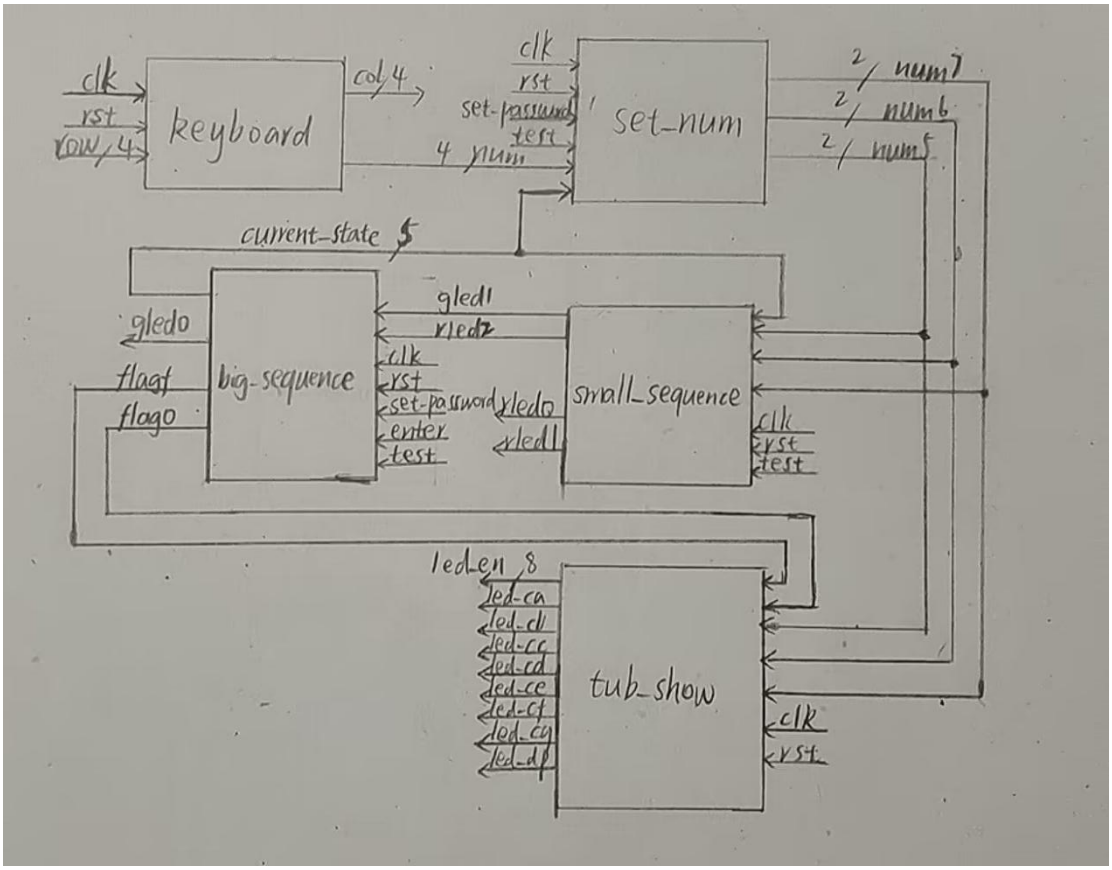
设计的功能描述
概述基本功能、详细描述自行扩展的功能 基本功能：实现 3 位数字的密码锁，每位数字取值 1、2、3。 A.按 S1 复位进入未设置密码状态，数码管显示 00000000。 B.若未设置密码或者已解锁，按 S2 进入设置密码状态，数码管初始不显示任何数字，通过 4*4 小键盘输入 3 位密码，在数码管中显示，输入除 1、2、3 以外的数字无效，输入满 3 位后按 S5 确定，进入确定密码状态，GLD0 亮。 C.若已设置密码，按 S3 进入输入验证密码状态，通过 4*4 小键盘输入 3 位密码，输入满 3 位后按 S5 确定，进入状态机验证状态，如果正确进入解锁状态，如果错误，累计错误次数加 1，显示对应数量红灯,累计 3 次系统锁住,进入密码锁锁住状态,数码管显示 FFFFFFFF，除了复位所有功能无效。 D.每次进入新状态，数码管清除上个状态的输入，每个状态下，同步显示键盘输入到数码管。

系统功能详细设计

用硬件框图描述系统主要功能及各模块之间的相互关系

要求有信号名、位宽、模块说明，可以参考下面的框图（仅为示例），须有密码处理模块、数码管显示处理模块、按键处理模块，其他模块不限，不可用 vivado 的 RTL 截图。





密码处理模块：small_sequence

数码管显示模块：tub_show

按键处理模块：set_num

按键模块：keyboard

密码锁状态处理模块：big_sequence

状态描述、状态转移图及状态编码，包括功能状态转移图、密码匹配的状态转移图；

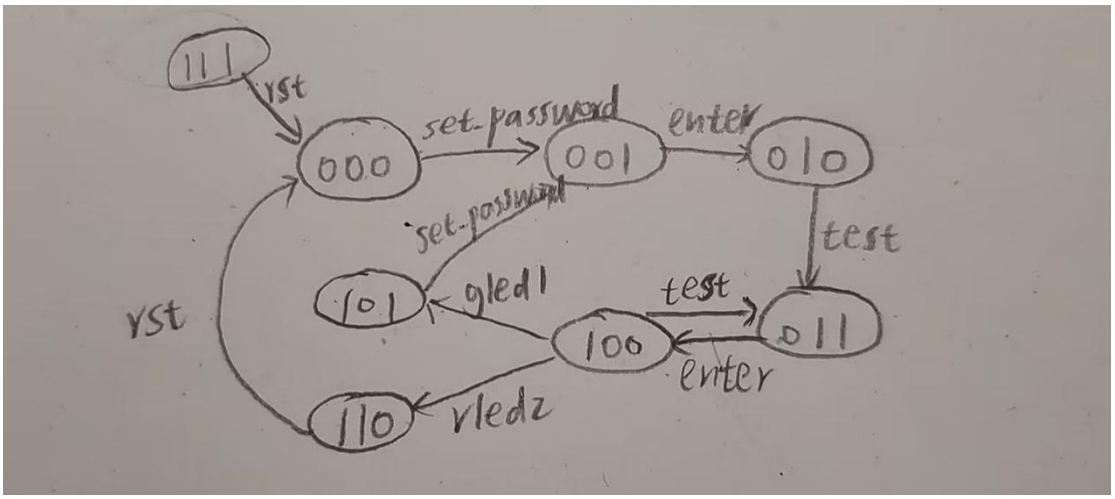
功能状态转移

八个状态：IDLE 未工作状态，S0 未设置密码状态，S1 设置密码即输入将要设置的密码状态，S2 确认设置密码状态，S3 验证密码即输入将要验证的密码状态，S4 状态机验证密码状态，S5 解锁状态，S6 密

码锁锁定状态。

状态编码：S0--000,S1--001,S2--010,S3--011,S4--100,S5--101,S6--110,IDLE--111。

状态转移图：

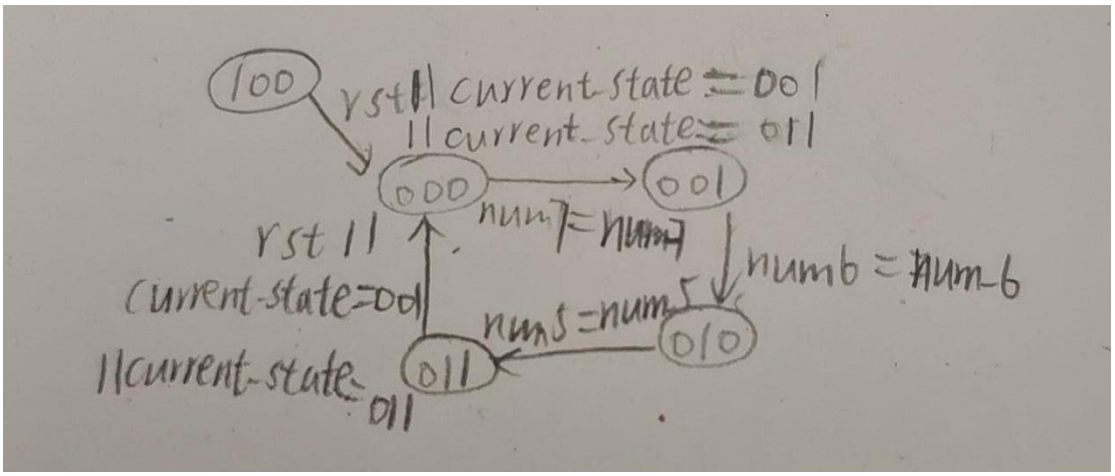


密码匹配状态转移

五个状态：IDLE 未工作状态，S0 成功匹配 0 位密码状态，S1 成功匹配 1 位密码状态，S2 成功匹配 2 位密码状态，S3 成功匹配 3 位密码状态。

状态编码：S0--000,S1--001,S2--010,S3--011,IDLE--100。

状态转移图：



各模块描述
包括模块功能，输入、输出端口、变量含义及主要设计代码 硬件框图有的模块都需体现 **密码处理模块：small_sequence** 功能：密码设置和密码匹配，其中密码匹配使用状态机实现； 输入端口：时钟端 clk，复位端 rst，密码锁状态 big_current-state，第一位密码 num7，第二位密码 num6，第三位密码 num5，验证密码信号 test； 输出端口：匹配成功状态绿灯信号 gled1，匹配失败一次红灯信号 rled0，匹配失败两次红灯信号 rled1，匹配失败三次红灯信号 rled2； 变量含义：num_7 是设置的第一位密码，num_6 是设置的第二位密码，num_5 是设置的第二位密码，cnt 是计时器，等待状态机验证完密码，验证完后 cnt 等于 10，current_state 是状态机目前状态，next_state 是状态机下一个状态，flag 判断是否让红灯亮起，flag 为 0 时，亮起一个红灯并把 flag 置 1，S1 到 S4 是功能状态机状态，IDLE 到 s3 是密码匹配状态机状态； 主要设计代码： 设置密码

```
always @ (posedge clk or posedge rst) begin
    if(rst) begin
        num_7 <= 1'b0;
        num_6 <= 1'b0;
        num_5 <= 1'b0;
        end
    else if(big_current_state == S2 && test == 1'b0) begin
        num_7 <= num7;
        num_6 <= num6;
        num_5 <= num5;
        end
end
```

状态机验证密码

```
always @ (posedge clk or posedge rst) begin
    if(rst)
        next_state <= s0;
    else if(big_current_state == S1 || big_current_state == S3)
        next_state <= s0;
    else if(big_current_state == S4)
        case(current_state)
            s0: if(num7 == num_7)
                    next_state <= s1;
                else ;
            s1: if(num6 == num_6)
                    next_state <= s2;
                else ;
            s2: if(num5 == num_5)
                    next_state <= s3;
                else ;
            default: ;
        endcase
end
```

数码管显示模块：tub_show

功能：显示输入的数字，确认状态下显示 00000000，锁住状态下显示 FFFFFFFF；

输入端口：时钟端 clk，复位端 rst，第一位密码 num7，第二位密码 num6，第三位密码 num5，flag0，flagf；

输出端口：八位数码管使能信号 led_en，数码管段触发信号 led_ca 到 led_cg 和小数点触发信号 led_dp；

变量含义：flag0 为 1 时显示 00000000，flagf 为 1 时显示 FFFFFFFF，都为 0 时，显示输入密码；

主要设计代码：

```
always @ (posedge clk) begin
    if(flag0 == 1'b1) begin
        led_ca <= 0;led_cb <= 0;led_cc <= 0;led_cd <= 0;led_ce <= 0;led_cf <= 0;led_cg <= 1;led_dp <= 1;
    end
    else if(flagf == 1'b1) begin
        led_ca <= 0;led_cb <= 1;led_cc <= 1;led_cd <= 1;led_ce <= 0;led_cf <= 0;led_cg <= 0;led_dp <= 1;
    end
    else begin
        if(~led_en[7]) begin
            if(num7 == 2'd0) begin
                led_ca <= 1;led_cb <= 1;led_cc <= 1;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
            end
            else if(num7 == 2'd1) begin
                led_ca <= 1;led_cb <= 0;led_cc <= 0;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
            end
            else if(num7 == 2'd2) begin
                led_ca <= 0;led_cb <= 0;led_cc <= 1;led_cd <= 0;led_ce <= 0;led_cf <= 1;led_cg <= 0;led_dp <= 1;
            end
            else if(num7 == 2'd3) begin
                led_ca <= 0;led_cb <= 0;led_cc <= 0;led_cd <= 0;led_ce <= 1;led_cf <= 1;led_cg <= 0;led_dp <= 1;
            end
        end

        else if(~led_en[6]) begin
            if(num6 == 2'd0) begin
                led_ca <= 1;led_cb <= 1;led_cc <= 1;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
            end
            else if(num6 == 2'd1) begin
                led_ca <= 1;led_cb <= 0;led_cc <= 0;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
            end
            else if(num6 == 2'd2) begin
                led_ca <= 0;led_cb <= 0;led_cc <= 1;led_cd <= 0;led_ce <= 0;led_cf <= 1;led_cg <= 0;led_dp <= 1;
            end
            else if(num6 == 2'd3) begin
                led_ca <= 0;led_cb <= 0;led_cc <= 0;led_cd <= 0;led_ce <= 1;led_cf <= 1;led_cg <= 0;led_dp <= 1;
            end
        end
```

```
        else if(~led_en[5]) begin
            if(num5 == 2'd0) begin
                led_ca <= 1;led_cb <= 1;led_cc <= 1;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
            end
            else if(num5 == 2'd1) begin
                led_ca <= 1;led_cb <= 0;led_cc <= 0;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
            end
            else if(num5 == 2'd2) begin
                led_ca <= 0;led_cb <= 0;led_cc <= 1;led_cd <= 0;led_ce <= 0;led_cf <= 1;led_cg <= 0;led_dp <= 1;
            end
            else if(num5 == 2'd3) begin
                led_ca <= 0;led_cb <= 0;led_cc <= 0;led_cd <= 0;led_ce <= 1;led_cf <= 1;led_cg <= 0;led_dp <= 1;
            end
        end
        else begin
            led_ca <= 1;led_cb <= 1;led_cc <= 1;led_cd <= 1;led_ce <= 1;led_cf <= 1;led_cg <= 1;led_dp <= 1;
        end
    end
end
```

按键处理模块：set_num

功能：依次设置 num7，num6 和 num5；

输入端口：时钟端 clk，复位端 rst，设置密码信号 set_password，验证密码信号 test，按键输入 num，功能状态机状态 current_state；

输出端口：第一位密码 num7，第二位密码 num6，第三位密码 num5；

变量含义：num7，num6 和 num5 为 0 时表示未设置；

主要设计代码：

```
always @ (posedge clk or posedge rst) begin
    if(rst) begin
        num7 <= 2'd0;
        num6 <= 2'd0;
        num5 <= 2'd0;
    end
    else if(set_password || test) begin
        num7 <= 1'b0;
        num6 <= 1'b0;
        num5 <= 1'b0;
    end
    else if(current_state == 5'd1 || current_state == 5'd3) begin
        if(num7 == 1'b0 && (num == 2'd1 || num == 2'd2 || num == 2'd3))
            num7 <= num;
        else if(num7 != 1'b0 && num6 == 1'b0 &&(num == 2'd1 || num == 2'd2 || num == 2'd3))
            num6 <= num;
        else if(num6 != 1'b0 && num5 == 1'b0 &&(num == 2'd1 || num == 2'd2 || num == 2'd3))
            num5 <= num;
    end
end
```

按键模块：keyboard

功能：输出按键按下的值；

输入端口：时钟端 clk，复位端 rst，行输入信号 row；

输出端口：列扫描信号 col，按下的数字 keyboard_num；

变量含义：row 和 col 分别为行信号和列信号，key 存储每个列扫描信号的行输入信号，key_r 是上一个时钟沿的 key 值，key_posedge 存储按键位置，按下时，相应位置为 1；

主要设计代码：

```
always @(posedge clk, posedge reset) begin
    if (reset == 1) begin
        keyboard_num <= 0;
    end else if (key_posedge) begin
        if (key_posedge[0]) keyboard_num <= 'hd;
        else if (key_posedge[1]) keyboard_num <= 'hc;
        else if (key_posedge[2]) keyboard_num <= 'hb;
        else if (key_posedge[3]) keyboard_num <= 'ha;
        else if (key_posedge[4]) keyboard_num <= 'hf;
        else if (key_posedge[5]) keyboard_num <= 'h9;
        else if (key_posedge[6]) keyboard_num <= 'h6;
        else if (key_posedge[7]) keyboard_num <= 'h3;
        else if (key_posedge[8]) keyboard_num <= 'h0;
        else if (key_posedge[9]) keyboard_num <= 'h8;
        else if (key_posedge[10]) keyboard_num <= 'h5;
        else if (key_posedge[11]) keyboard_num <= 'h2;
        else if (key_posedge[12]) keyboard_num <= 'he;
        else if (key_posedge[13]) keyboard_num <= 'h7;
        else if (key_posedge[14]) keyboard_num <= 'h4;
        else if (key_posedge[15]) keyboard_num <= 'h1;
    end else begin
        keyboard_num <= 0;
    end
end

always @(posedge clk, posedge reset) begin
    if (reset == 1)           col <= 4'b1111;
    else if (col ==  4'b1111) col <= 4'b1110;
    else if (cnt_end)         col <= {col[2:0], col[3]};
end


always @(posedge clk, posedge reset) begin
    if (reset == 1) key_r <= 0;
    else key_r <= key;
end

always @(posedge clk, posedge reset) begin
    if (reset == 1) key <= 0;
    else if (cnt_end) begin
        if (col[0] == 0) key[3:0]   <= ~row;
        if (col[1] == 0) key[7:4]   <= ~row;
        if (col[2] == 0) key[11:8]  <= ~row;
        if (col[3] == 0) key[15:12] <= ~row;
    end
end
```

密码锁状态处理模块：big_sequence

功能：利用状态机处理密码锁功能状态；

输入端口：时钟端 clk，复位端 rst，设置密码信号 set_password，确认信号 enter，验证密码信号 test，匹配成功状态绿灯信号 gled1，匹配失败三次红灯信号 rled2；

输出端口：flag0，flagf，密码设置成功信号 gled0，功能状态机状态 current_state；

变量含义：flag0 为 1 时数码管显示 00000000，flagf 为 1 时数码管显示 FFFFFFFF；

主要设计代码：

```
always @ (posedge clk or posedge rst) begin
    if(rst)
        current_state <= IDLE;
    else
        current_state <= next_state;
end
```

```
always @ (posedge clk or posedge rst) begin
    if(rst)
        next_state <= S0;
    else
        case(current_state)
            S0: if(set_password == 1'b1)
                    next_state <= S1;
                else ;
            S1: if(enter == 1'b1)
                    next_state <= S2;
                else ;
            S2: if(test == 1'b1)
                    next_state <= S3;
                else ;
            S3: if(enter == 1'b1)
                    next_state <= S4;
                else ;
            S4: if(gled1 == 1'b1)
                    next_state <= S5;
                else if(rled2 == 1'b1)
                    next_state <= S6;
                else if(test == 1'b1)
                    next_state <= S3;
            S5: if(set_password == 1'b1)
                    next_state <= S1;
                else ;
```

```
            S6: ;
            default: ;
        endcase
end

always @ (posedge clk) begin
        case(current_state)
            S0: begin
                flag0 <= 1'b1;
                flagf <= 1'b0;
                gled0 <= 1'b0;
                end
            S1: begin
                flag0 <= 1'b0;
                gled0 <= 1'b0;
                end
            S2: begin
                flag0 <= 1'b1;
                gled0 <= 1'b1;
                end
            S3: begin
                flag0 <= 1'b0;
                end
            S4: begin
                flag0 <= 1'b1;
                end

            S5: begin
                flag0 <= 1'b1;
                end
            S6: begin
                flag0 <= 1'b0;
                flagf <= 1'b1;
                end
            default ;
        endcase
end
```

调试报告

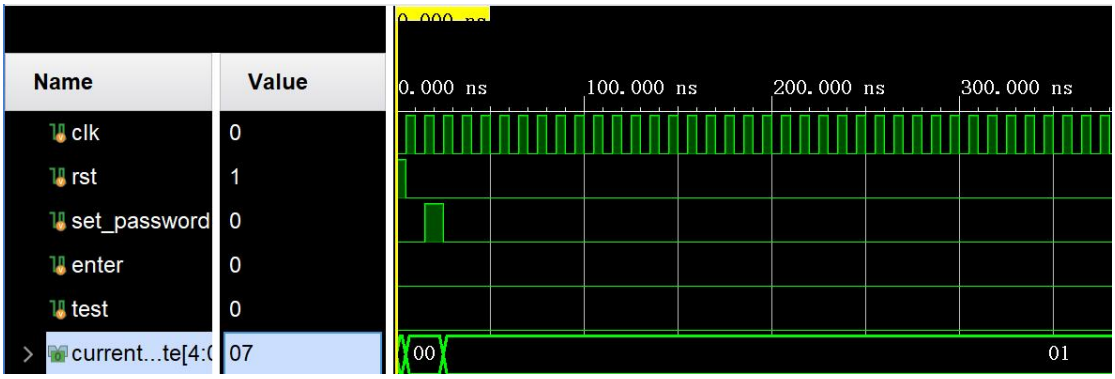
仿真波形截图及仿真分析

（1）需分析功能状态转移，包括密码设定、密码匹配成功、密码匹配失败、密码锁定，

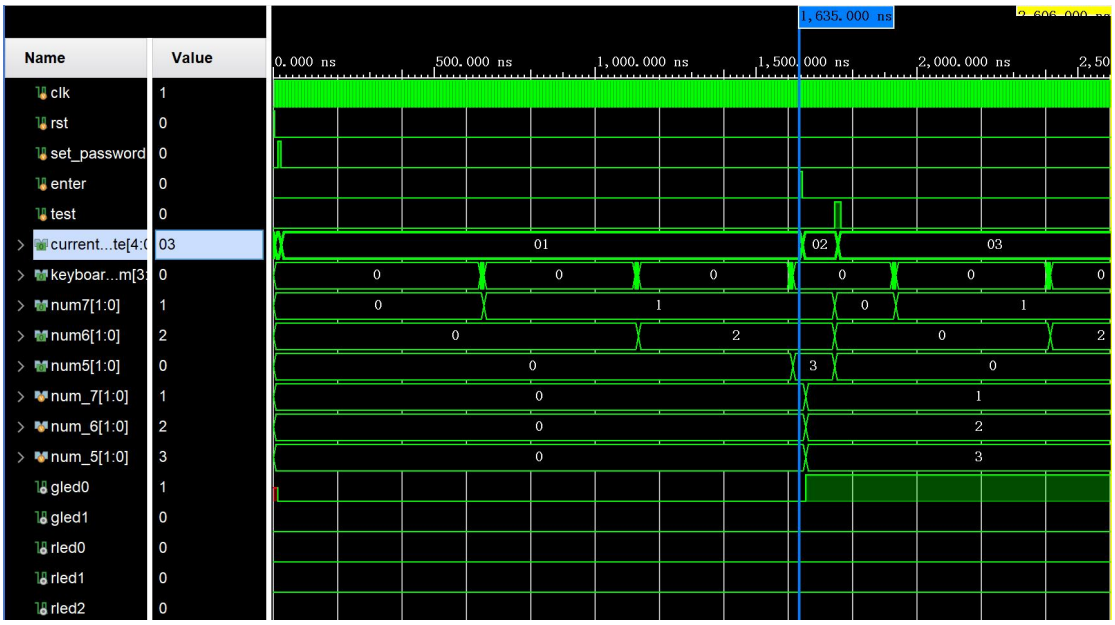
（2）需分析密码匹配状态机的转移过程，匹配失败和成功各一次即可。

（3）只需分析顶层信号和提到的状态变量，其他信号自行决定。

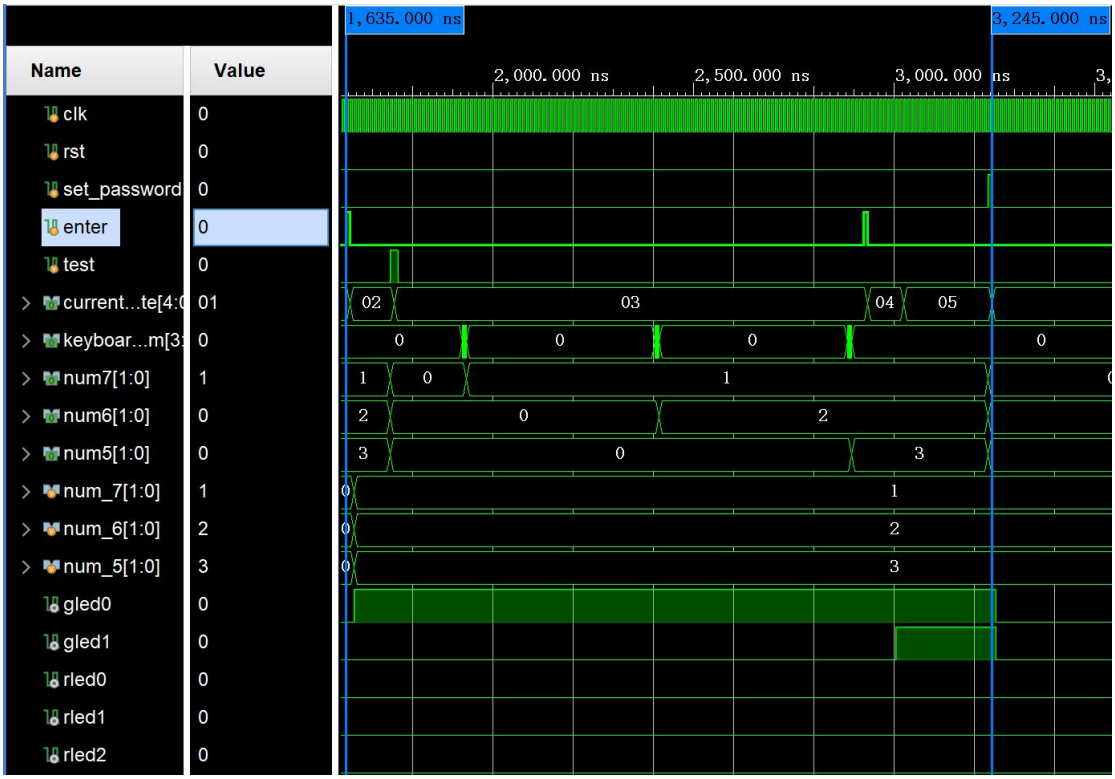
功能状态转移



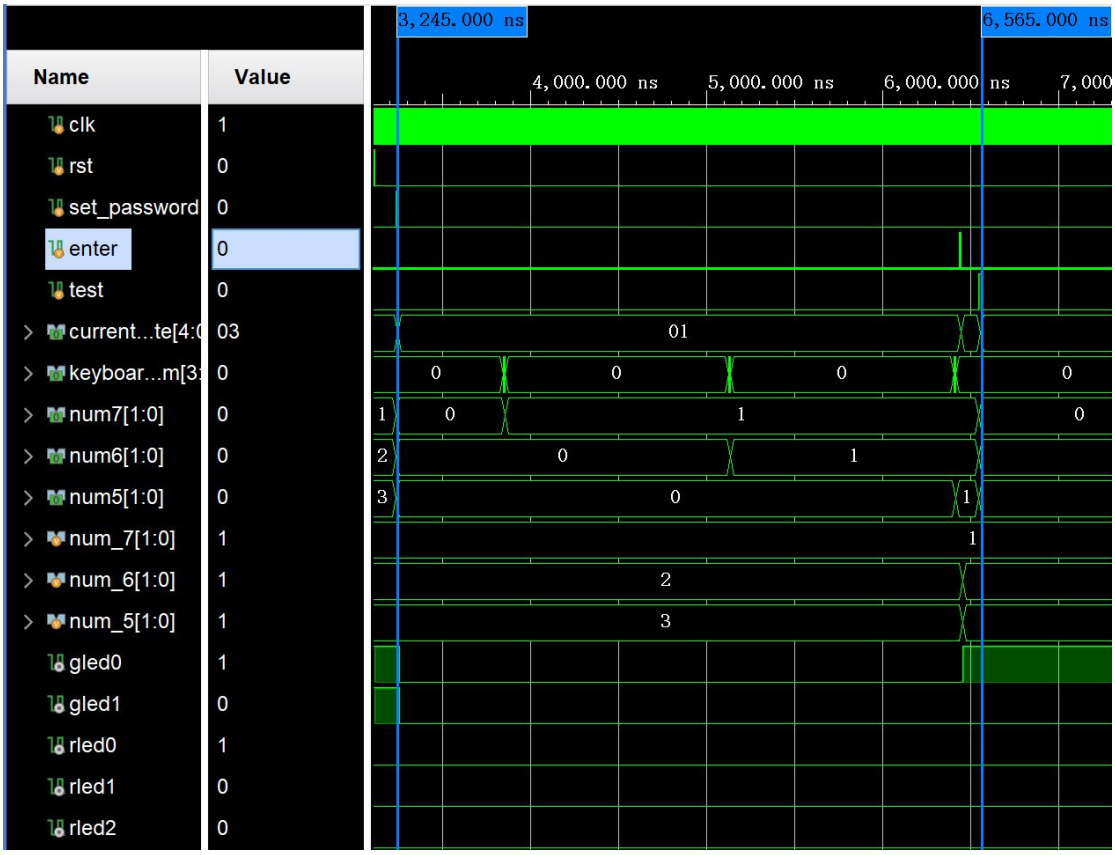
首先按下复位键，进入 S0 未设置密码状态，再按下 set_password，进入 S1 输入设置密码状态；



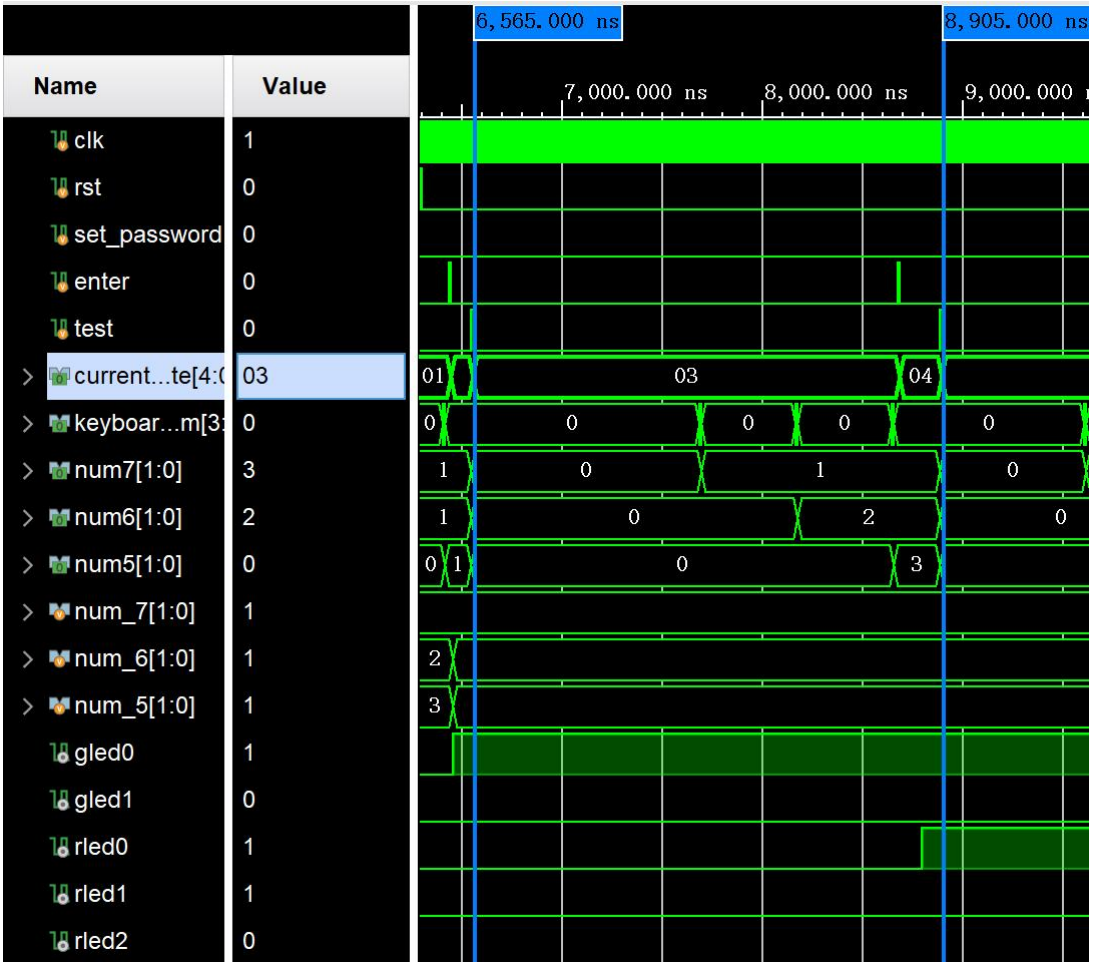
随后蓝色标记线处按下 enter 键，进入 S2 确认设置密码状态，密码 num_7,num_6 和 num_5 分别被设置为 1，2，3，gled0 亮起；



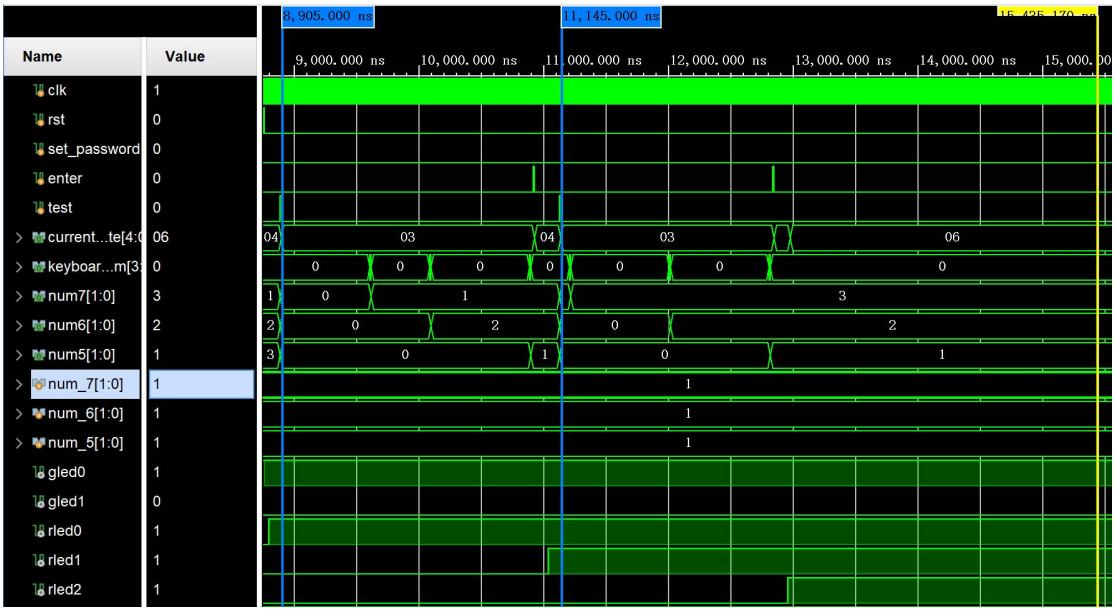
再按下 test 键，进入 S3 输入验证密码状态，输入 1，2，3，按下 enter 键，进入 S4 密码匹配状态，密码匹配成功，进入 S5 解锁状态，gled1 亮起；



再按下 set_password 键，进入 S1 输入设置密码状态，gled0 和 gled1 均熄灭，输入设置密码 1，1，1，按下 enter 键，进入 S2 确认设置密码状态，num_7,num_6 和 num_5 分别被设置为 1，1，1，gled0 亮起；



再按下 test 键，进入 S3 输入验证密码状态，输入验证密码 1，2，3，再按下 enter 键，进入 S4 匹配密码状态，密码匹配失败，rled0 亮起；

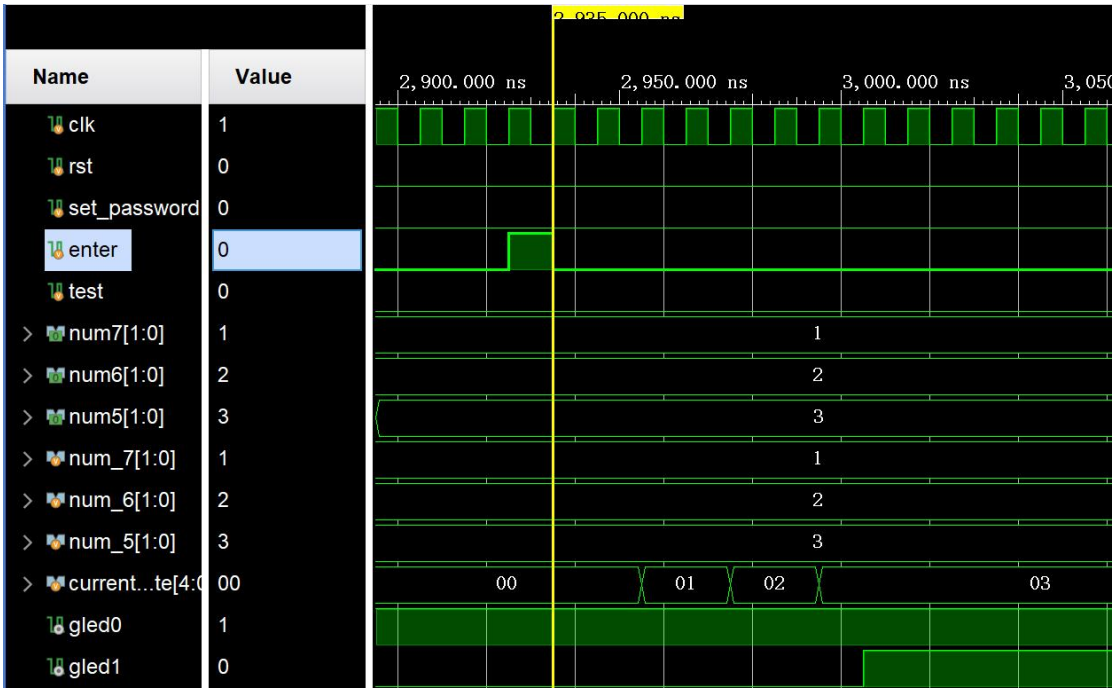


再按下 test 键，进入 S3 输入验证密码状态，进行第二次验证，输入验证密码 1，2，1，按下 enter 键，进入 S4 密码匹配状态，密码匹配失败，rled1 亮起；

再按下 test 键，进入 S3 输入验证密码状态，进行第三次验证，输入验证密码 3，2，1，按下 enter 键，进入 S4 密码匹配状态，密码匹配失败，rled2 亮起，进入 S6 密码锁锁定状态；

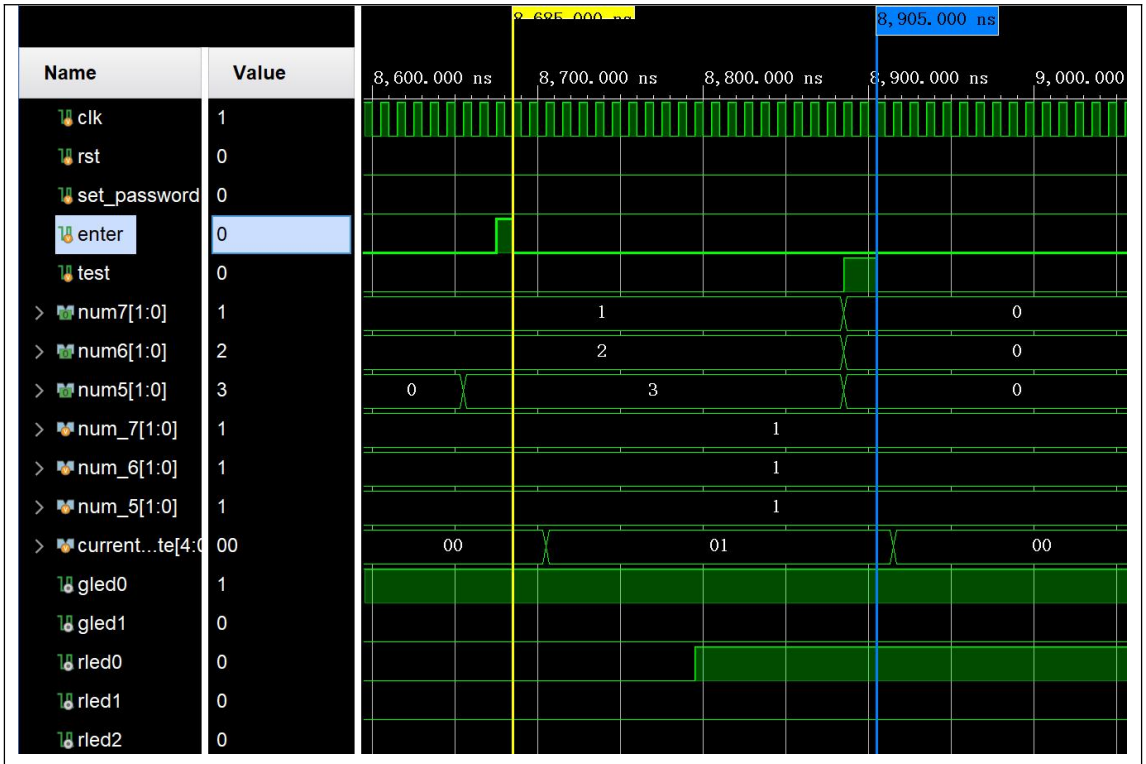
密码匹配状态转移

成功



按下 enter 键开始匹配，num_7 等于 num7，进入 S1 状态，num_6 等于 num6，进入 S2 状态，num_5 等于 num5，进入 S3 状态，匹配成功，gled1 亮起；

失败



按下 enter 键开始匹配，num_7 等于 num7，进入 S1 状态，num_6 不等于 num6，仍然是 S1 状态，无论 num_5 等不等于 num5，仍然是 S1 状态，匹配失败一次，rled0 亮起；

设计过程中遇到的问题及解决方法

1.如何安排 gled0，gled1，rled0，rled1，rled2？如果安排不当，就会在两个 always 块中赋值，产生冲突。

解决办法：在功能状态机中对 gled0 赋值，进入 S2 确认密码状态下，gled0 赋值 1，在密码匹配状态机中对 gled1，rled0，rled1，rled2 赋值，匹配成功时，gled1 赋值 1，根据失败次数对 rled0，rled1 和 rled2 赋值。

2.什么情况下数码管显示 00000000，显示 FFFFFFFF，显示输入数字？

解决办法：设置两个变量 flag0 和 flagf，flag0 为 1 时，显示 00000000，flagf 为 1 时，显示 FFFFFFFF，都为 0 时，显示输入数字。

3.在上板中发现设置密码不正确，看了仿真后发现，设置密码在 S2 状态最后变回了 000。

解决办法：在按下 test 键后，num7，num6，num5 立即清 0，如图，

```
else if(set_password || test) begin
    num7 <= 1'b0;
    num6 <= 1'b0;
    num5 <= 1'b0;
end
```

但是下一个时钟沿，又把 num7，num6，num5 赋给 num_7，num_6，num_5，所以我给赋值条件加入 test==1'b0，在按下 test 时，不再赋值，这样避免了设置密码清 0，如图。

```
else if(big_current_state == S2 && test == 1'b0) begin
    num_7 <= num7;
    num_6 <= num6;
    num_5 <= num5;
    end
```

4.如何判断密码匹配状态机匹配完成？

解决办法：设置一个计时器，从开始匹配计时十个周期，再来判断状态机状态，此时，状态机一定匹配完成。

课程设计总结

包括设计的总结和还需改进的内容以及收获

总结：本次实验综合了前几次实验的内容包括数码管显示和状态机等，要求我们掌握自顶向下的结构化设计方法，锻炼了我们的设计能力和解决实际问题的能力。

还需改进的内容：功能状态机和密码匹配状态机状态容易搞混。

收获：学习了 4*4 键盘的行列扫描原理，并运用于实践中，增强了问题分析能力和解决能力。