**北京科技大学数字逻辑实验报告**

学院： 计算机与通信工程学院 专业： 物联网工程 班级： 物联201

姓名： 赵方程 学号： 42024137 实验日期： 2021 年 12 月 15 日

**实验名称：**实验五 状态机综合实验

**实验目的：**

本实验的目的是运用状态机原理实现一个具有实用功能的应用，将状态机原理运用到项目开发中。该实验需要设计者在分析阶段对电路各个模块之间的接口有较为清晰的认识，熟练状态机的设计，培养自顶向下的设计思路，进一步理解状态机原理在数字电路设计过程中的便捷之处。为后续复杂数字系统设计打好基础。

**实验内容：**

**实验5.1——电子秤设计**

1. 功能

* 单次计价：输入物品的重量、单价，显示物品的总价（=重量\*单价）。
* 累计计价：
* 第一次按下累计按键，记住当前物品的总价（当前物品记为物品1），数码管依次显示:

AC 次数 应付总价

* 继续输入物品2的重量、单价，显示物品2的总价。按下累计按键，将本次物品2的总价累加进之前的用户应付总价中，数码管显示：

AC 次数 总价

* 依次购买物品3、4……，每个物品后都通过累计按键将本次物品总价累计到应付总价中
* 退出累计状态：按下清除累计按键，恢复普通状态。

1. 显示

重量：2个数码管（十进制显示）

单价：2个数码管（十进制显示）

总价：4个数码管（十进制显示）

AC：显示这两个字母

1. 接口

* 输入

重量：拨码开关，4位（二进制输入）

单价：拨码开关，4位（二进制输入）

累计：按键，1个

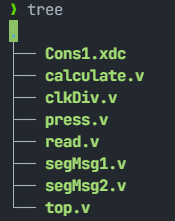
复位：按键，1个

* 输出

数码管：8个

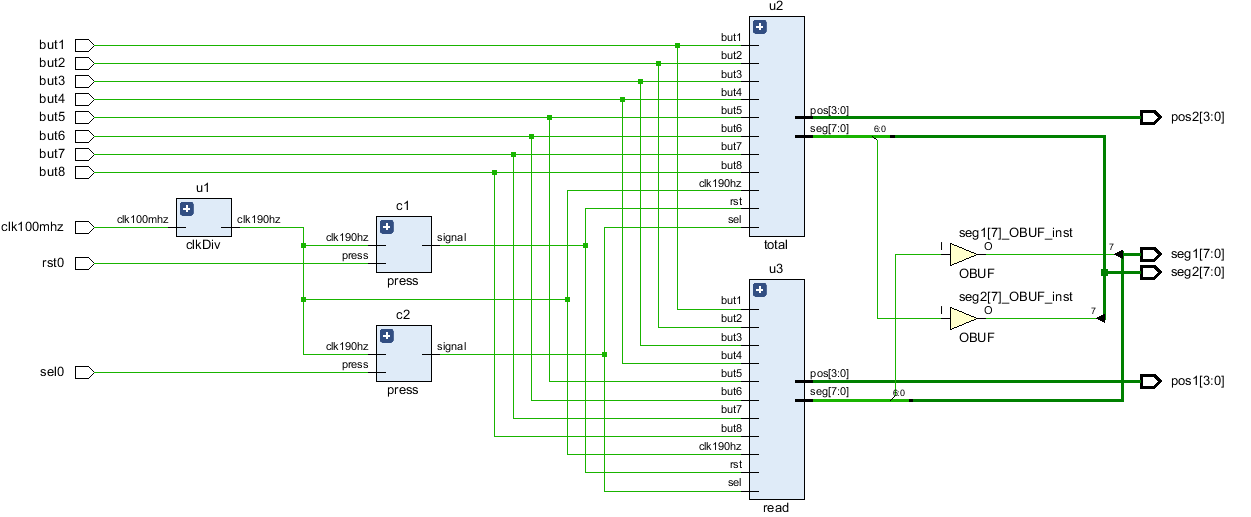
**实验步骤：**

系统设计



1. Cons1.xdc 管脚约束
2. calclulate.v 计算总价
3. clkDiv.v 分频
4. press.v 按键消抖
5. read.v 读取单价/重量
6. segMsg1.v 驱动总价 （右侧）数码管
7. serMsg2.v 驱动重量/单价 （左侧）数码管
8. top.v 顶层文件

RTL结构设计



源代码

`timescale 1ns / 1ps

module top(

    input clk100mhz,

    input rst,

    input sel,

////////////////////////////////////////

    input but1, // input number

    input but2, // input number

    input but3, // input number

    input but4, // input number

////////////////////////////////////////

    input but5, // input unit price

    input but6, // input unit price

    input but7, // input unit price

    input but8, // input unit price

////////////////////////////////////////

    output [3:0] pos1, // First LED display

    output [7:0] seg1, // First LED display

////////////////////////////////////////

    output [3:0] pos2, // Second LED display

    output [7:0] seg2  // Second LED display

);

    wire clk190hz, clk3hz;

    wire rst\_, sel\_;

    clkDiv u1(clk100mhz, clk190hz, clk3hz);

    //\* read AND display `number & unit price` AND calculate the price

    read   u2(clk190hz, rst\_, sel\_, but1, but2, but3, but4, but5, but6, but7, but8, pos1, seg1);

    //\* add to `total price` AND display

    calculate  u3(clk190hz, rst\_, sel\_, but1, but2, but3, but4, but5, but6, but7, but8, pos2, seg2);

    press  c1(clk190hz, rst, rst\_);

    press  c2(clk190hz, sel, sel\_);

endmodule

`timescale 1ns / 1ps

module calculate(

    input clk190hz,

    input rst,

    input sel,

/////////////////////////////////////

    input but1,

    input but2,

    input but3,

    input but4,

/////////////////////////////////////

    input but5,

    input but6,

    input but7,

    input but8,

/////////////////////////////////////

    output [3:0] pos,

    output [7:0] seg

);

    wire [3:0] mid1;

    wire [3:0] mid2;

    assign mid1 = {but4,but3,but2,but1};

    assign mid2 = {but8,but7,but6,but5};

    segMsg1 u(clk190hz, rst, sel, mid1, mid2, pos, seg);

endmodule

`timescale 1ns / 1ps

module read(

    input clk190hz,

    input rst,

    input sel,

////////////////////////////////////////////////////////////////

    input but1,

    input but2,

    input but3,

    input but4,

////////////////////////////////////////////////////////////////

    input but5,

    input but6,

    input but7,

    input but8,

////////////////////////////////////////////////////////////////

    output [3:0] pos,

    output [7:0] seg

);

    wire [3:0] mid1, mid2;

    reg  [3:0] dataBus1, dataBus2;  // weight and unit price

    reg  [3:0] dataBus3, dataBus4;  // weight and unit price

    assign mid1 = {but4,but3,but2,but1};

    assign mid2 = {but8,but7,but6,but5};

    //\* convert `buts` to weight and unit price

    always @(posedge clk190hz)  begin

        case(mid1)

            4'b0000: begin

                dataBus1 = 4'b0000;

                dataBus2 = 4'b0000;

            end

            4'b0001: begin

                dataBus1 = 4'b0001;

                dataBus2 = 4'b0000;

            end

            4'b0010: begin

                dataBus1 = 4'b0010;

                dataBus2 = 4'b0000;

            end

            4'b0011: begin

                dataBus1 = 4'b0011;

                dataBus2 = 4'b0000;

            end

            4'b0100: begin

                dataBus1 = 4'b0100;

                dataBus2 = 4'b0000;

            end

            4'b0101: begin

                dataBus1 = 4'b0101;

                dataBus2 = 4'b0000;

            end

            4'b0110: begin

                dataBus1 = 4'b0110;

                dataBus2 = 4'b0000;

            end

            4'b0111: begin

                dataBus1 = 4'b0111;

                dataBus2 = 4'b0000;

            end

            4'b1000: begin

                dataBus1 = 4'b1000;

                dataBus2 = 4'b0000;

            end

            4'b1001: begin

                dataBus1 = 4'b1001;

                dataBus2 = 4'b0000;

            end

            4'b1010: begin

                dataBus1 = 4'b0000;

                dataBus2 = 4'b0001;

            end

            4'b1011: begin

                dataBus1 = 4'b0001;

                dataBus2 = 4'b0001;

            end

            4'b1100: begin

                dataBus1 = 4'b0010;

                dataBus2 = 4'b0001;

            end

            4'b1101: begin

                dataBus1 = 4'b0011;

                dataBus2 = 4'b0001;

            end

            4'b1110: begin

                dataBus1 = 4'b0100;

                dataBus2 = 4'b0001;

            end

            4'b1111: begin

                dataBus1 = 4'b0101;

                dataBus2 = 4'b0001;

            end

        endcase

        case(mid2)

            4'b0000: begin

                dataBus3 = 4'b0000;

                dataBus4 = 4'b0000;

            end

            4'b0001: begin

                dataBus3 = 4'b0001;

                dataBus4 = 4'b0000;

            end

            4'b0010: begin

                dataBus3 = 4'b0010;

                dataBus4 = 4'b0000;

            end

            4'b0011: begin

                dataBus3 = 4'b0011;

                dataBus4 = 4'b0000;

            end

            4'b0100: begin

                dataBus3 = 4'b0100;

                dataBus4 = 4'b0000;

            end

            4'b0101: begin

                dataBus3 = 4'b0101;

                dataBus4 = 4'b0000;

            end

            4'b0110: begin

                dataBus3 = 4'b0110;

                dataBus4 = 4'b0000;

            end

            4'b0111: begin

                dataBus3 = 4'b0111;

                dataBus4 = 4'b0000;

            end

            4'b1000: begin

                dataBus3 = 4'b1000;

                dataBus4 = 4'b0000;

            end

            4'b1001: begin

                dataBus3 = 4'b1001;

                dataBus4 = 4'b0000;

            end

            4'b1010: begin

                dataBus3 = 4'b0000;

                dataBus4 = 4'b0001;

            end

            4'b1011: begin

                dataBus3 = 4'b0001;

                dataBus4 = 4'b0001;

            end

            4'b1100: begin

                dataBus3 = 4'b0010;

                dataBus4 = 4'b0001;

            end

            4'b1101: begin

                dataBus3 = 4'b0011;

                dataBus4 = 4'b0001;

            end

            4'b1110: begin

                dataBus3 = 4'b0100;

                dataBus4 = 4'b0001;

            end

            4'b1111: begin

                dataBus3 = 4'b0101;

                dataBus4 = 4'b0001;

            end

        endcase

    end

    segMsg2 u(clk190hz, rst, sel, dataBus1, dataBus2, dataBus3, dataBus4, pos, seg);

endmodule

`timescale 1ns / 1ps

module segMsg1(

    input clk190hz,

    input rst,

    input sel,

    input [3:0] mid1,

    input [3:0] mid2,

    output reg [3:0] pos,

    output reg [7:0] seg

);

    wire [15:0] dataBus;

    reg [1:0] posC;

    reg [3:0] dataP;

    reg [15:0] totalNum;

    reg rst\_delay = 0;

    reg sel\_delay = 0;

    //\* like flag (isChanged)

    reg [3:0] mid1\_previous, mid2\_previous;

    assign dataBus = mid1 \* mid2; //\* calculate

    always @(posedge clk190hz) begin

        if(mid1!=mid1\_previous||mid2!=mid2\_previous) begin

            rst\_delay = 0;

            sel\_delay = 0;

            mid1\_previous<=mid1;

            mid2\_previous<=mid2;

        end

        if (rst||rst\_delay) begin

            totalNum=0;

            pos   = 4'b1111;

            posC  = 0;

            dataP = 0;

            rst\_delay=1;

        end

        else if(sel||sel\_delay) begin

            if (sel\_delay == 0) begin

                totalNum = totalNum + dataBus;

                sel\_delay = 1;

            end

            case (posC)

                0: begin

                    pos   <= 4'b0001;

                    dataP <= totalNum%10;

                end

                1: begin

                    pos   <= 4'b0010;

                    dataP <= (totalNum/10)%10;

                end

                2: begin

                    pos   <= 4'b0100;

                    dataP <= (totalNum/100)%10;

                end

                3: begin

                    pos   <= 4'b1000;

                    dataP <= (totalNum/1000)%10;

                end

            endcase

            posC = posC + 1;

        end

        else begin

            case (posC)

                0: begin

                    pos   <= 4'b0001;

                    dataP <= dataBus%10;

                end

                1: begin

                    pos   <= 4'b0010;

                    dataP <= (dataBus/10)%10;

                end

                2: begin

                    pos   <= 4'b0100;

                    dataP <= (dataBus/100)%10;

                end

                3: begin

                    pos   <= 4'b1000;

                    dataP <= (dataBus/1000)%10;

                end

            endcase

            posC = posC + 1;

        end

    end

    // Drive display

    always @(dataP)

        case (dataP)

            0: seg = 8'b0011\_1111;

            1: seg = 8'b0000\_0110;

            2: seg = 8'b0101\_1011;

            3: seg = 8'b0100\_1111;

            4: seg = 8'b0110\_0110;

            5: seg = 8'b0110\_1101;

            6: seg = 8'b0111\_1101;

            7: seg = 8'b0000\_0111;

            8: seg = 8'b0111\_1111;

            9: seg = 8'b0110\_1111;

            10:seg = 8'b0100\_0000;

            11:seg = 8'b0000\_0000;

            default: seg = 8'b0000\_1000;

        endcase

endmodule

`timescale 1ns / 1ps

module segMsg2(

    input clk190hz,

    input rst,

    input sel,

    ///////////////////////////////

    input [3:0] dataBus1,

    input [3:0] dataBus2,

    ///////////////////////////////

    input [3:0] dataBus3,

    input [3:0] dataBus4,

    ///////////////////////////////

    output reg [3:0] pos,

    output reg [7:0] seg

);

    reg [3:0] dataBus1\_previous = 0, dataBus2\_previous = 0, dataBus3\_previous = 0, dataBus4\_previous = 0;

    reg [1:0] posC;

    reg [3:0] dataP;

    reg [3:0] num;

    reg rst\_delay = 0;

    reg sel\_delay = 0;

    always @(posedge clk190hz)

    begin

        if(dataBus1!=dataBus1\_previous||dataBus2!=dataBus2\_previous||

            dataBus3!=dataBus3\_previous||dataBus4!=dataBus4\_previous) begin

            rst\_delay = 0; //\*   set to 0    //

            sel\_delay = 0; //\* to swith back //

            dataBus1\_previous <= dataBus1;

            dataBus2\_previous <= dataBus2;

            dataBus3\_previous <= dataBus3;

            dataBus4\_previous <= dataBus4;

        end

        if (rst||rst\_delay) begin

            if (rst\_delay == 0) begin

                num   = 0;

                pos   = 4'b1111;

                posC  = 0;

                dataP = 0;

                rst\_delay = 1;

            end

        end

        else if(sel||sel\_delay) begin

            if (sel\_delay == 0) begin

                num = num + 1;

                sel\_delay = 1;

            end

            case (posC) //\* "ACXX"

                0: begin

                    pos   <= 4'b0001;

                    dataP <= num%10;

                end

                1: begin

                    pos   <= 4'b0010;

                    dataP <= (num/10)%10;

                end

                2: begin

                    pos   <= 4'b0100;

                    dataP <= 4'b1011;

                end

                3: begin

                    pos   <= 4'b1000;

                    dataP <= 4'b1010;

                end

            endcase

            posC = posC + 1;

        end

        else begin

            case (posC)

                0: begin

                    pos   <= 4'b0001;

                    dataP <= dataBus1;

                end

                1: begin

                    pos   <= 4'b0010;

                    dataP <= dataBus2;

                end

                2: begin

                    pos   <= 4'b0100;

                    dataP <= dataBus3;

                end

                3: begin

                    pos   <= 4'b1000;

                    dataP <= dataBus4;

                end

            endcase

            posC = posC + 1;

        end

    end

    always @(posC)

    begin

        case (dataP)

            0: seg = 8'b0011\_1111;

            1: seg = 8'b0000\_0110;

            2: seg = 8'b0101\_1011;

            3: seg = 8'b0100\_1111;

            4: seg = 8'b0110\_0110;

            5: seg = 8'b0110\_1101;

            6: seg = 8'b0111\_1101;

            7: seg = 8'b0000\_0111;

            8: seg = 8'b0111\_1111;

            9: seg = 8'b0110\_1111;

            10:seg = 8'b0111\_0111;

            11:seg = 8'b0011\_1001;

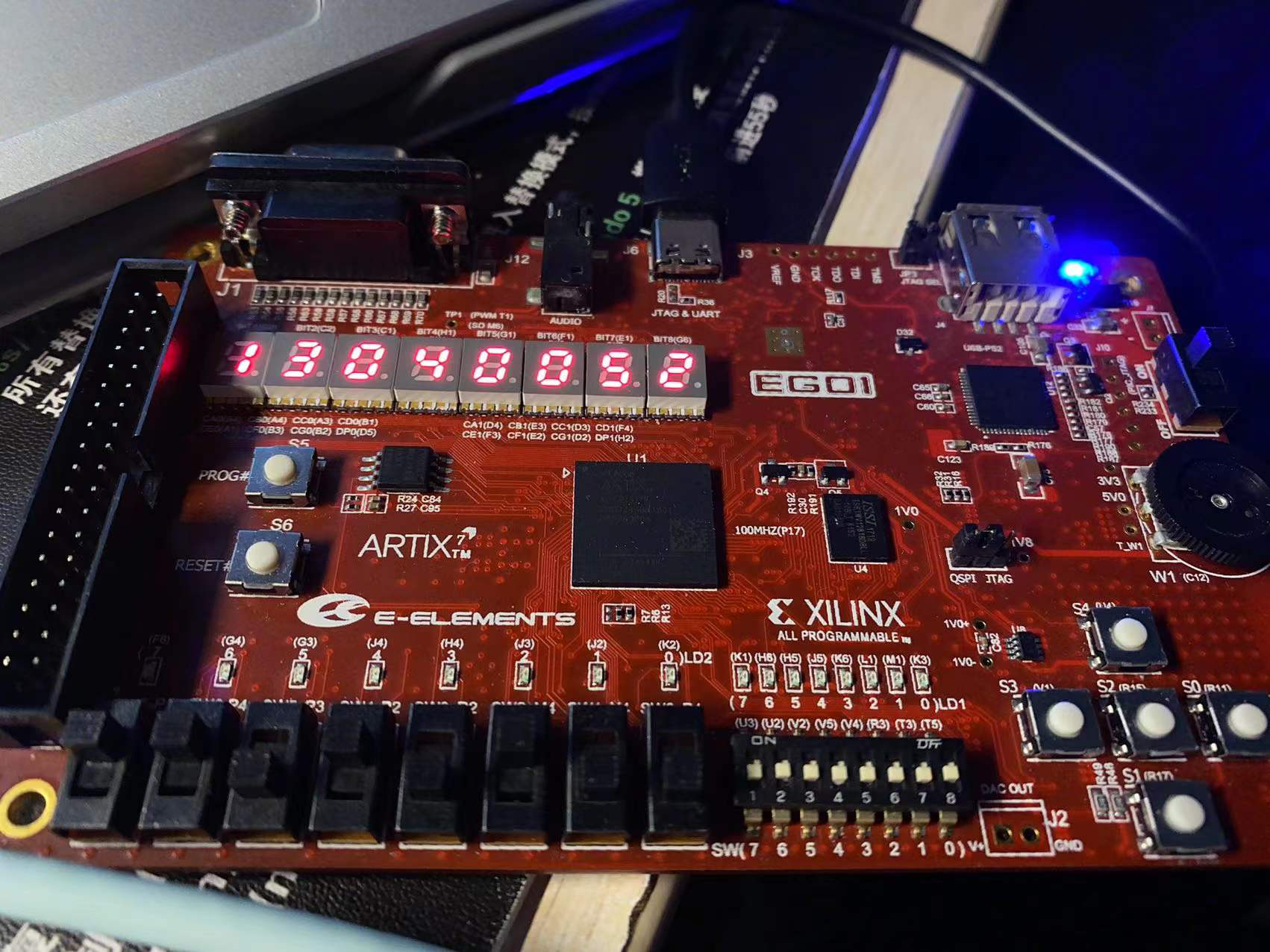
            default: seg = 8'b0000\_1000;

        endcase

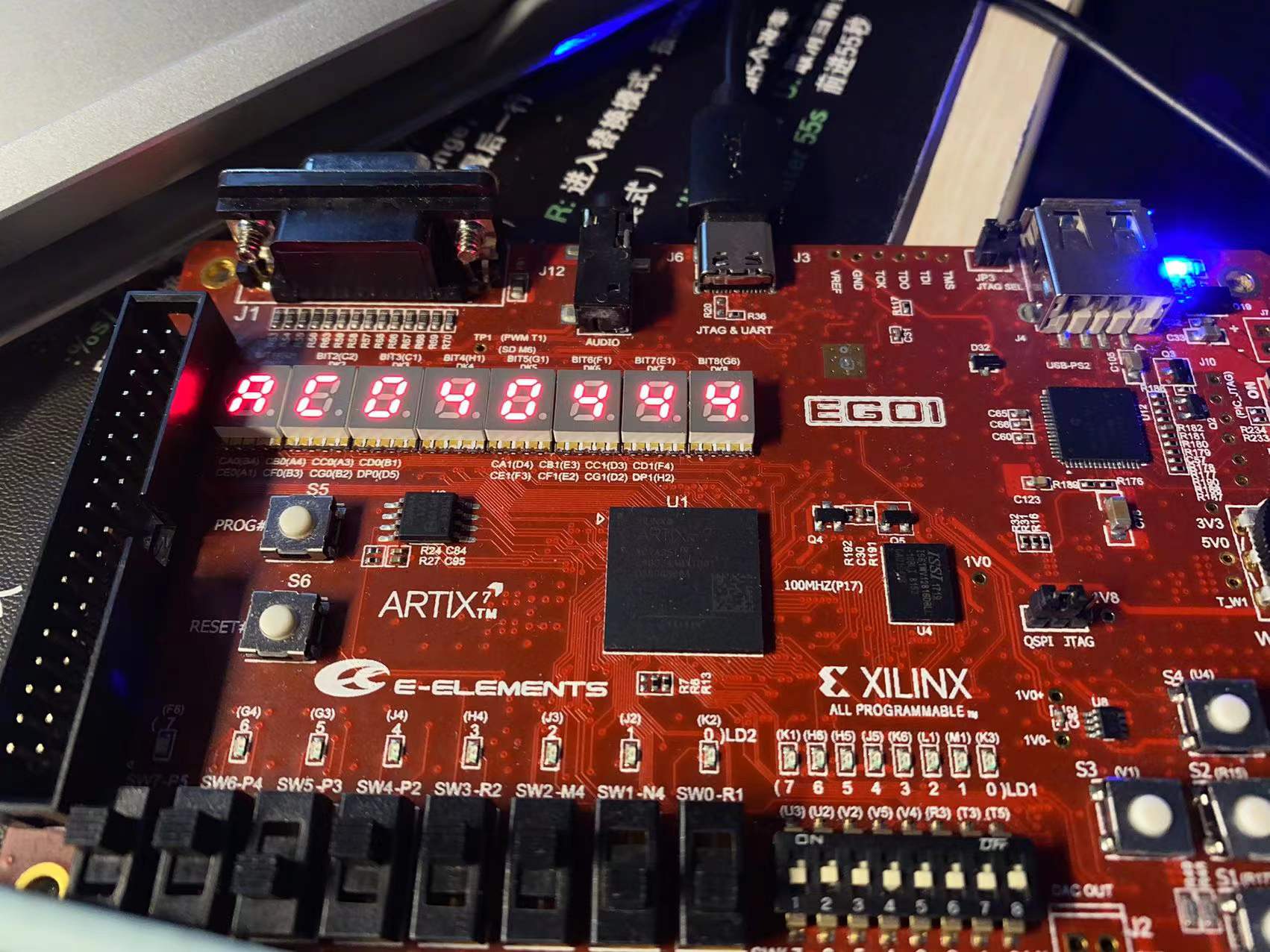
    end

endmodule

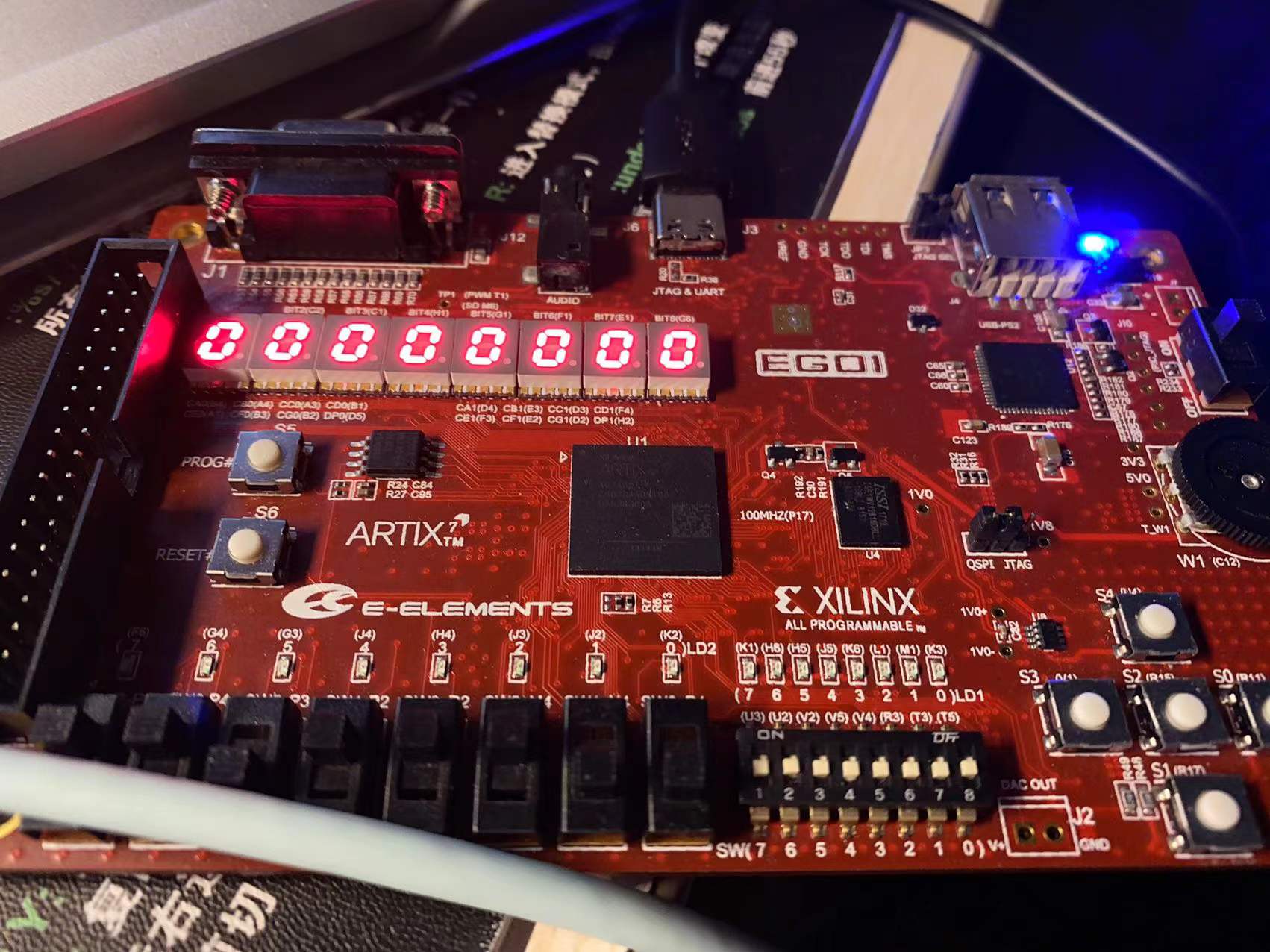
**实验结果与分析：**

****

**输入状态**

****

**累加状态**

****

**清零状态**