数字系统课程设计项目报告

DIGITAL SYSTEM CURRICULUM DESIGN PROJECT REPORT

设计题目：　地铁售票模拟系统

组员姓名：　钟 源　学号：04022212

丁玺潼　学号：04022415

目录

一、实验目的

1、熟悉 vivado 软件的使用方法。

2、学习并掌握 Verilog 硬件编程语言的体系及用法。

3、通过 vivado 运用 Verilog 语言编写有实际意义的数字逻辑系统。

4、掌握使用 Verilog 语言实现数字系统课程设计的基本方法。

二、实验要求及内容

功能描述：用于模仿地铁售票的自动售票，完成地铁售票的核心控制功能。

（1） 地铁售票机有两个进币孔，可以输入硬币和纸币，售货机有两个进币孔，一个是输入硬币，一个是输入纸币，硬币的识别范围是1 元的硬币，纸币的识别范围是5 元，10 元，20元。乘客可以连续多次投入钱币。

（2） 以南京市轨道交通1/2/3/4号线为基准进行设计考虑。站点数较多，需自行编码。

（3） 系统可以通过按键设定当前站点为4条线路中任意一站。

（4） 乘客买票时可以有两种选择，第一种，乘客已经知道所需费用，直接选择票价，如2元、3元或4元或更多。第二种，不知道票价，选择出站口，系统以目的地与当前站的站数来进行计算价格，计算方式参
间有可能有多种价格的，以最低价格为准。

（5） 得到票价单价后，选择所需购买的票数，然后进行投币，投入的钱币达到所需

金额时，售票机自动出票，并一次性找出余额，本次交易结束，等待下一次的交易。在投币期间，乘客可以按取消键取消本次操作，钱币自动一次性退出。

（6）

三、功能设计

3.1 基本功能

（1）地铁售票机可以有两种方式进行付费，可以输入硬币和纸币，售货机有两个进币孔，一个是输入硬币，一个是输入纸币，硬币的识别范围是 1 元的硬币，纸币的识别范围是 10 元，20 元，50 元，100 元，乘客可以连续多次投入钱币。

（2）售票机对站台进行编码，实现了 63 个站点之间的选择。

（4） 乘客可以自行选择入站与出站站点，售票机根据所选站点计算票价。

（4）乘客买票时可以有两种选择，第一种，乘客已经知道所需费用，直接选择票价。第二种，不知道票价，选择出站口，系统以目的地与当前站的站数来进行计算价格。

（5）得到票价单价后，乘客可以选择所需购买的票数，然后进行投币，投入的钱币达到所需金额时，售票机自动出票，并一次性找出余额，本次交易结束，等待下一次的交易。在投币期间，乘客可以按取消键取消本次操作，钱币自动一次性退出。

3.2 附加功能

（1）可以实时显示单个票价、总票价、付款数额、找零数额，并且可以通过开关来回切换，互不影响。

（2）既可以通过按钮加减票数，也可以选择手动键盘输入，大批量购票。

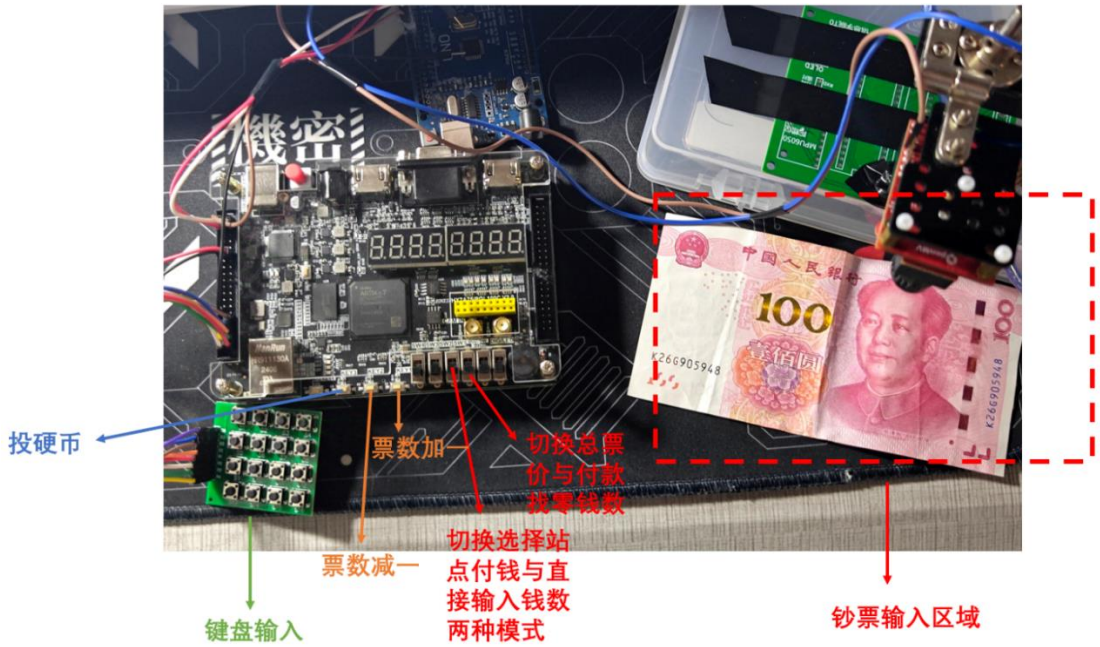
（3）可以利用 openmv 实时识别纸币，且稳定性好，支持多次纸币投放。

（4）通过 Arduino 板接受 openmv 的信息，并调整时序发送至 FPGA 板。

（5）不同状态可以透过开关和 led 灯指示。

（6）矩阵键盘输入时有蜂鸣器提醒用户。

3.3 操作区说明



（1）用户可以上下推开关 2，开关 2 下推时，代表用户输入进站口与出站口；开关 2 上推时，代表用户手动输入票价。

（2）用户通过键盘的 0~9 的按键，进行输入。

每输入一个数字后，再按除 0~9 之外的其他按键代表“OK”键，切换到下一个数码管位，进行输入。当开关 2 下拨时，前两个数码管代表入站，后两个数码管代表出站。当开关 2 上拨时，前两个数码管代表手动票价，后两个数码管代表票数。

（3）用户通过开关 1 切换每种方式下数码管的不同显示内容。

在开关 2 下拨时：当开关 1 下拨，后四位数码管显示单张票价与应付钱数；当开

关 1 上拨，后四位数码管显示已付钱数与找零。

在开关 2 上拨时：当开关 1 下拨，后四位数码管显示应付钱数；当开关 1 上拨，前四位数码管显示已付钱数，后四位数码管显示找零。

（4）KEY1，KEY2 分别代表加票与减票。

当 KEY1 按下时，购买票数加一；当 KEY2 按下时，购买票数减一，票数最小减少到 0。

（5）KEY3 代表硬币投入。

当 KEY3 按下时，代表投入 1 元硬币。

（6）钞票输入区域代表识别钞票，将纸币放在 openmv 下方后，会输出相应的价格。

四、设计方案详解

4.1 设计思路

在本次地铁售票模拟系统的设计方案中，我们采用了模块化的设计思路，将整个系统分为多个功能模块，包括硬币纸币识别模块、票价计算模块、显示模块、输入模块和控制模块。每个模块都有明确的功能和接口，便于调试和维护。

（1）**硬币纸币识别模块**：通过 openmv 摄像头识别纸币面额，并通过 Arduino 板接收 openmv 的信息，将纸币面额转换为数字信号，发送给 FPGA 板进行处理。

（2）**票价计算模块**：根据乘客依照地图选择的入站和出站站点，计算出相应的票价。票价计算考虑了不同站点之间的距离和票价政策，确保计算结果的准确性。

（3）**显示模块**：使用数码管显示票价信息，包括起点站、终点站、单张票价、票

数、总票价、已付金额和找零等。显示内容可以通过开关切换，以适应不同的显示需求。

（4）**输入模块**：包括矩阵键盘和硬币纸币投入按键，用户可以通过键盘输入站点或票价，通过按键模拟硬币和纸币的投入。

（5）**控制模块**：负责协调各个模块的工作，控制交易流程，包括投币、出票、找零等操作。控制模块还负责异常处理，如取消交易等。

通过这种模块化的设计，小组能够清晰地划分各个功能模块的职责，便于开发和维护。同时，模块化设计也提高了系统的可扩展性和可重用性。

4.2 模块设计

在模块设计方面，我们对每个功能模块进行了详细的设计，确保每个模块都能高效、准确地完成其功能。

（1）**硬币纸币识别模块**：我们采用了 openmv 摄像头进行纸币识别，通过图像识别技术识别纸币的面额。为了提高识别的准确性和稳定性，我们对 openmv 进行了多次参数调试和识别环境搭建，确保在不同光线和角度下都能准确识别纸币。

（2）**票价计算模块**：票价计算模块是系统的核心，它根据乘客选择的站点和票价政策计算出票价。我们从网络得到了具体的票价数据，总结在了 excel 表格中，随后应用 python 文件写完了票价计算文件，大大节省了项目开发时间和难度。

（3）**显示模块**：显示模块采用了数码管作为显示设备，通过 FPGA 控制数码管的显示内容。我们设计了一套显示逻辑，能够根据用户的输入和系统状态，动态显示各种票价信息。

（4）**输入模块**：输入模块包括矩阵键盘和硬币纸币投入按键。我们对矩阵键盘进行了优化，使其能够准确识别用户的输入。同时，我们还设计了一套按键消抖逻辑，

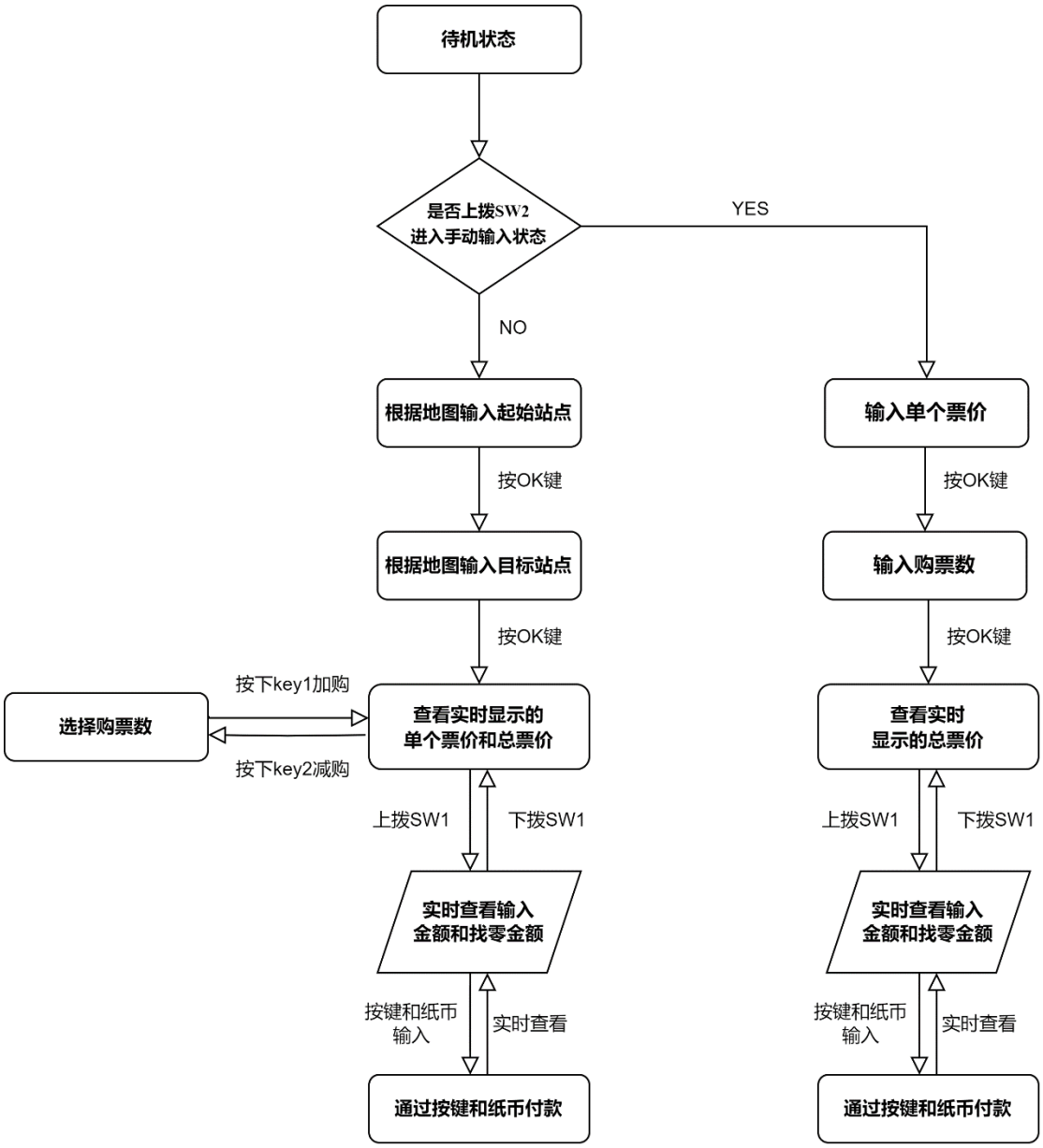
确保按键的稳定识别。

（5）**状态控制模块**：控制模块是系统的大脑，它负责协调各个模块的工作，控制交易流程。我们利用开关的上拨和下拨，总共分成了四种状态，并可以实现相互之间的无缝切换、相互独立、实时显示，较之往年的系统做出了巨大改进。

通过这些模块的设计，我们确保了系统的稳定性和可靠性，同时也提高了用户体验。

4.3 系统流程图

具体如图所示：



4.4 状态设计

在系统设计中，状态设计是非常重要的一部分。我们根据系统的工作流程，设计了以下主要状态：

1.**待机状态**：系统启动后，进入待机状态，等待用户输入。

2.**自动模式选站状态**：用户若保持默认开关状态（00），则可以通过矩阵键盘输入入站和出站站点，通过按键 key1 和 key2 可以选择票数，系统实时显示单张票价格和总价格。用户继续上拨开关 sw1，可以进入自动模式支付状态（11）。

3.**自动模式支付状态：**该状态下用户可以通过按键 key3 投一元硬币付款，也可以将纸钞放入识别区付款，系统会实时显示找零钱数，同时用户随时可以拨回开关 sw1 回到自动模式选站状态（00），以查看票价，并不影响支付状态。

4.**手动模式选站状态**：用户若上拨开关 sw2 进入手动模式选站状态（10），则可以通过矩阵键盘输入所需要票价和购票数，总票价。用户继续上拨开关 sw1，可以进入手动模式支付状态（11）。

5.**自动模式支付状态：**该状态下用户可以通过按键 key3 投一元硬币付款，也可以将纸钞放入识别区付款，系统会实时显示找零钱数，同时用户随时可以拨回开关 sw1 回到手动模式选站状态（10），以查看票价，并不影响支付状态。

6.**交易取消状态**：如果用户在购票过程通过矩阵键盘输入 4 个 0，系统将取消交易，并将已投入的钱币退还给用户。

4.5 程序设计亮点

在本次地铁自动售票机的程序设计中，我们注重了以下几个亮点：

（1）实时显示和状态切换：我们将系统分为 2 个模式 4 个状态，每个模式中两个状态可相互切换互不影响。而两模式之间有两个独立的账户，也可以相互切换互不影响。并且计算结果都能实时显示在界面中，不需要用户进行繁杂的操作，大大简化了流程。

（2）优化的硬币纸币识别算法：我们对 openmv 摄像头进行了优化，提高了纸币识别的准确性和稳定性。同时，我们通过按键模拟投入硬币，非常便捷。

（3）高效的票价计算算法：我们设计了一套高效的票价计算算法，能够快速准确地计算出票价。

（4）智能的显示逻辑：我们设计了一套智能的显示逻辑，能够根据用户的输入和

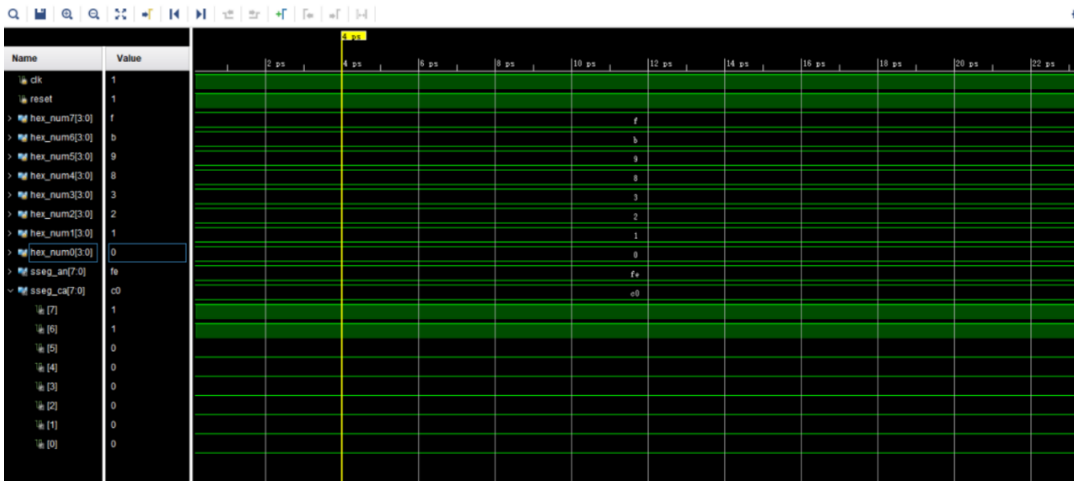
系统状态，动态显示各种票价信息。

（5）稳定的按键消抖逻辑：我们设计了一套稳定的按键消抖逻辑，确保了按键的稳定识别。

通过这些程序设计，我们确保了系统的稳定可靠，同时也提升了用户体验。

五、波形仿真及分析

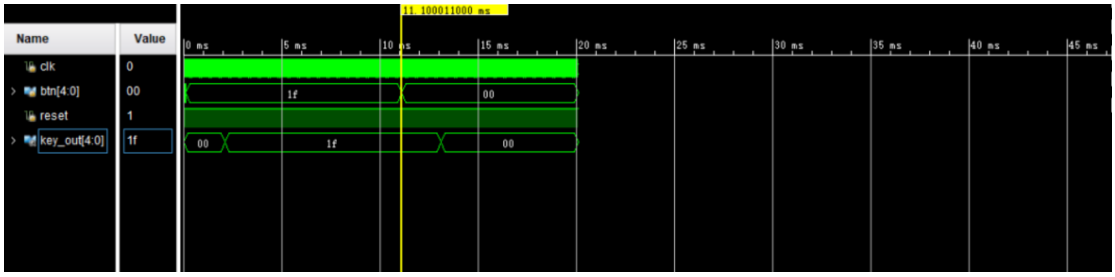
5.1 数码管仿真



数码管从 7 到 0 分别为不显示、显示 E、显示 9、显示 8、显示 3、显示 2、显示 1、显示 0，可得数码管仿真结果正确

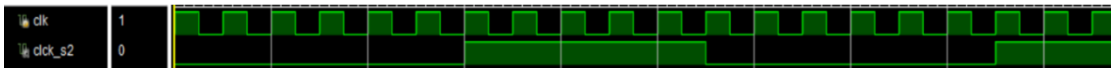
5.2 按键消抖仿真

由于程序中设计的按键消抖为 10ms，所以时间设置为 9ms，则不能识别出是键按下，如果时间设置为 11ms，可识别出键按下，经过一定的延时后进行响应。



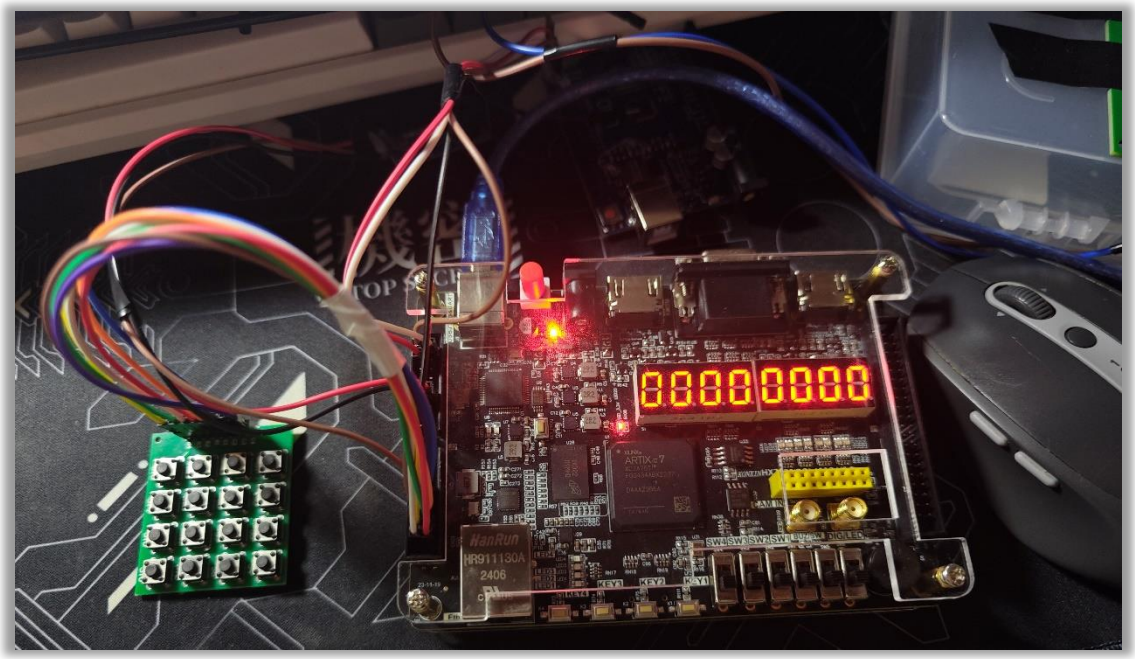
5.3 分频模块

每一个晶振上升沿记一次数，这样可以实现任意想要频率的分频，我们在程序中采用了仿真结果如图的五分频。

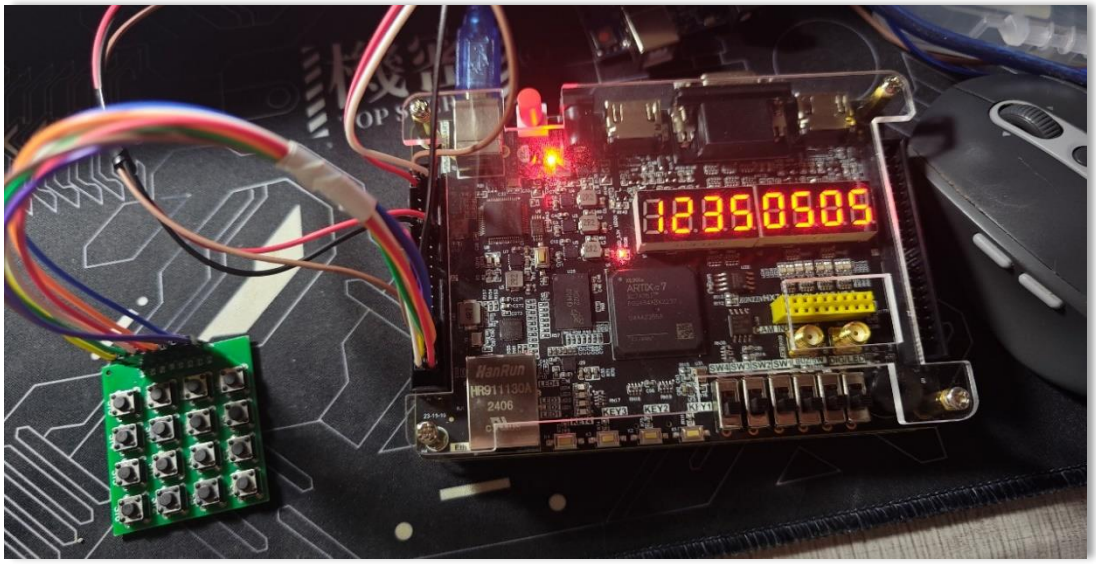


六、实物演示图

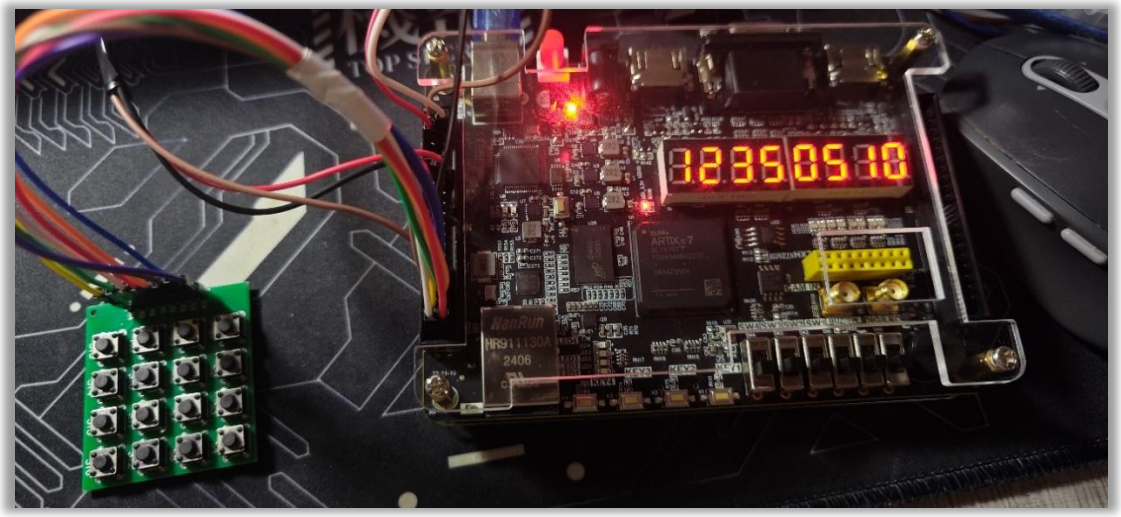
1．初始状态所有值全为 0

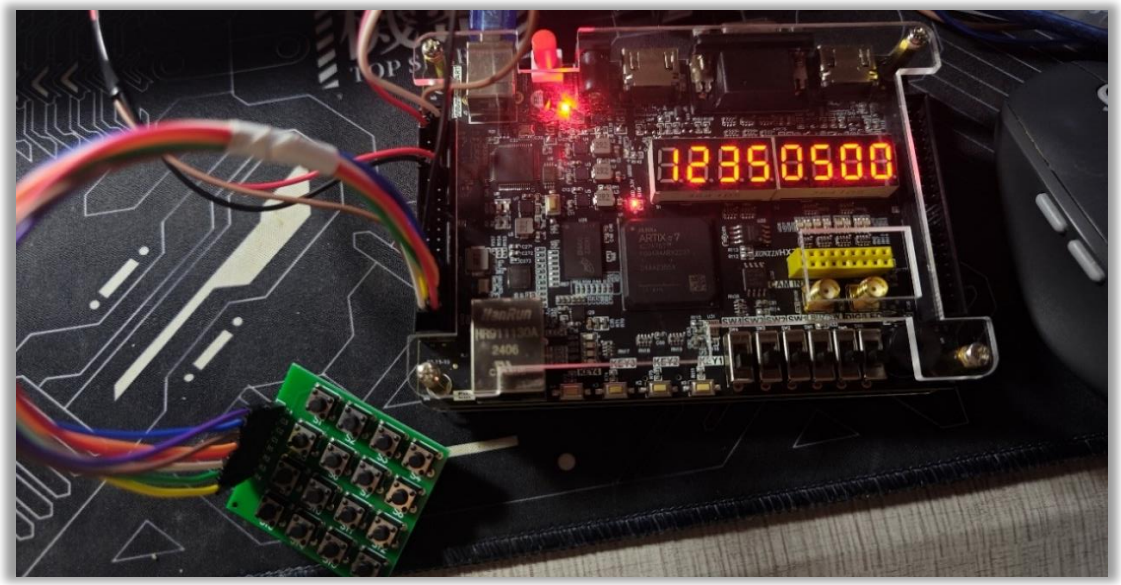


2．输入入站点为 12，出站点为 35 后，显示票价 5 元。

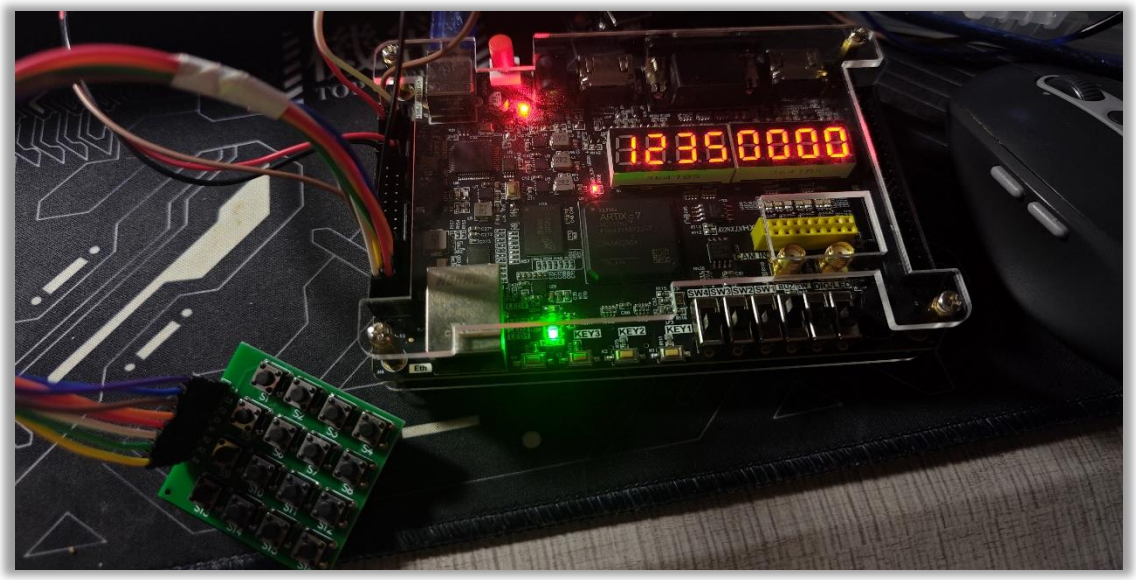


3．通过 KEY1 与 KEY2 实现票数的加减，从而使总票价发生变化。

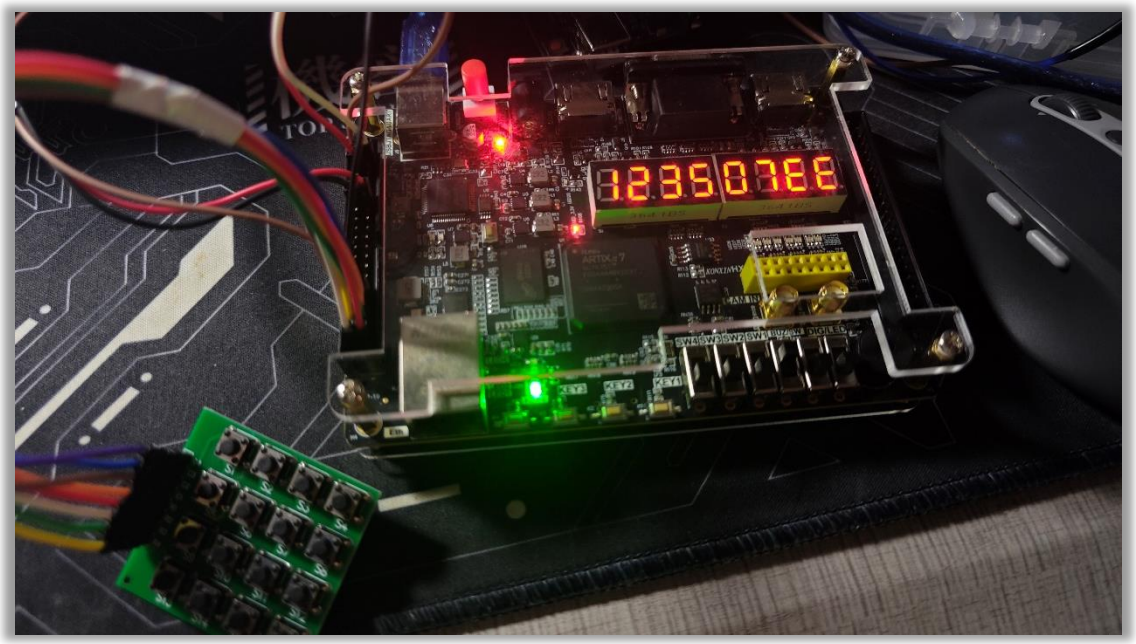




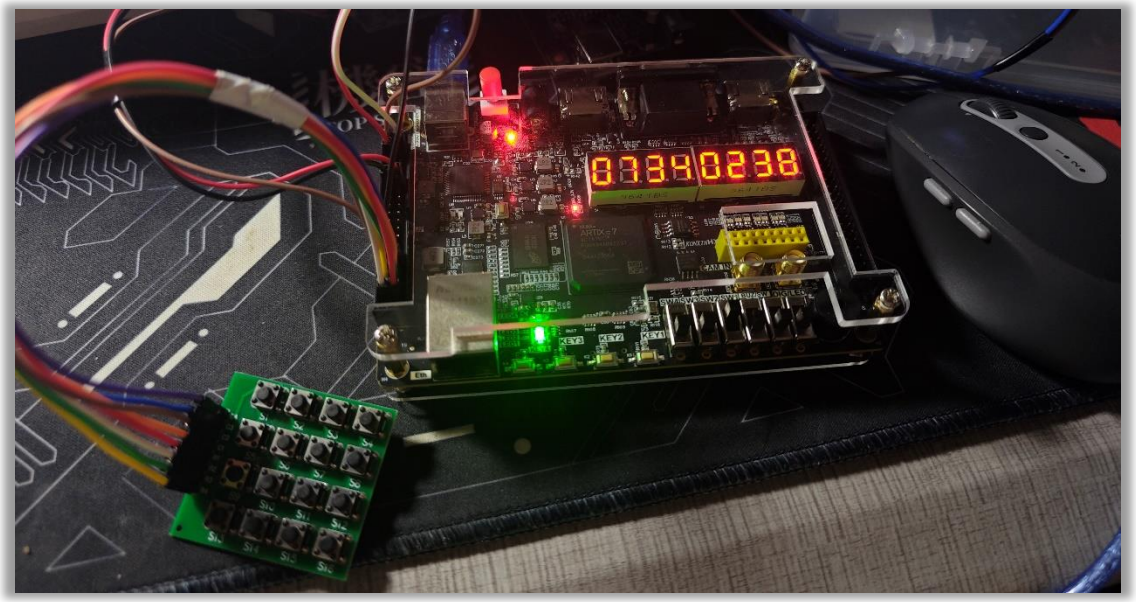
4．向上拨开关 1，切换显示已付金额与找零，由于此时应付钱数为 0，所以已付金额等于找零。



5．改变应付金额为 15 元后，当付款金额小于应付金额时，找零会显示“EE”。



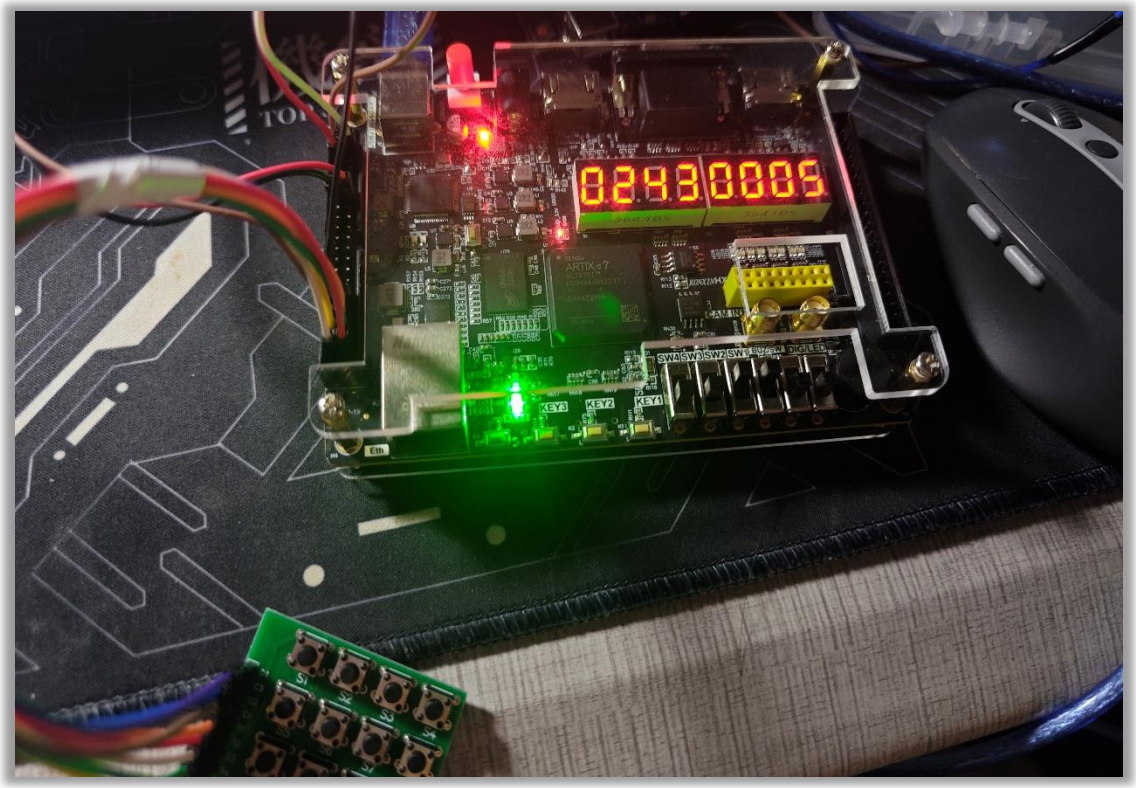
6．向上拨开关 2，更换为手动输入票价的模式，如图为输入单张票价为 7 元，总票数 34 张，总价为 238 元。

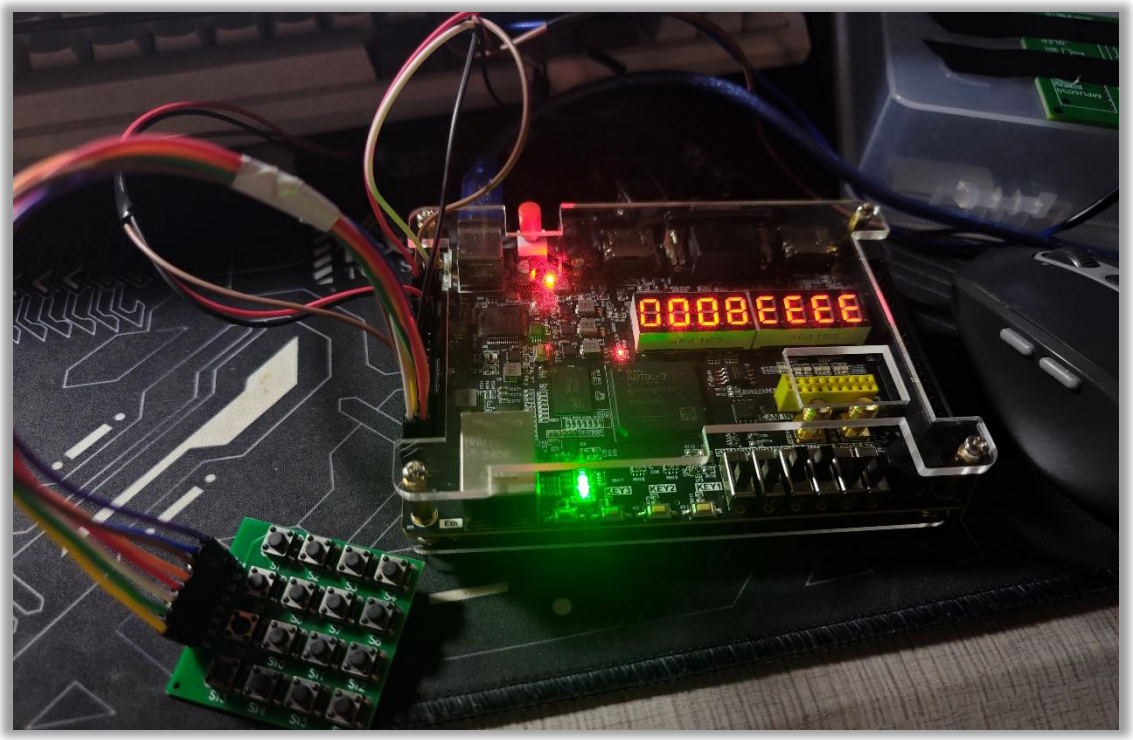


7．先上拨开关 1，此时前四位数码管显示已付钱数。

当已付钱数小于应付钱数时，后四位数码管显示“EEEE”；

当已付钱数大于应付钱数时，后四位数码管显示找零数；





七、小组分工

钟源：50%，包括 openmv 的纸币识别、FPGA 板块的开发和调试、蜂鸣器、地图绘制和状态机图绘制等。

丁玺潼：50%，包括 arduino 板的串口通信、FPGA 板块的开发和调试、矩阵键盘等。

八、总结与思考

在本次数字系统课程设计中，我们通过使用 Verilog 语言和 Vivado 软件，成功设计并实现了一个地铁自动售票模拟系统。在这个过程中，我们遇到了很多挑战，也学到了很多宝贵的经验。具体有：

1.团队合作的重要性：在项目开发过程中，我们深刻体会到了团队合作的重要性。我们发现两个人一起改代码出错率会比一个人改低很多，这也是我们自己总结的能在这么短时间将系统完成的一个重要原因。

2.理论与实践的结合：通过这次课程设计，我们将所学的理论知识应用到了实际项目中，加深了对 Verilog 语言和数字系统设计的理解。

3.问题解决能力的提升：在项目开发过程中，我们遇到了很多技术难题，如硬币纸币识别、票价计算、显示控制、蜂鸣器响的时间控制等。通过不断试错和尝试新的解决方案，我们提高了解决问题的能力。

4. 细节十分重要：在项目开发过程中，我们深刻体会到了细节的重要性。一个小小的错误或疏忽，都可能导致整个系统的失败。因此，我们在设计和调试过程中都非常注重细节。

通过这次课程设计，我们不仅提高了自己的专业技能，也锻炼了自己的团队合作能力和问题解决能力。我们相信，这些经验将对我们未来的学习和工作产生积极的影响。

九、致谢

在开始这次数字系统课程设计之前，我们对 Verilog 语言并没有太多的了解。非常感谢金子程老师对 Verilog 基本语法及 Vivado 使用方法的详细讲解，使我们能够迅速掌握这门语言。

附录 1、源程序代码

Github 网址：

DingXT1/Railway_System：专为铁路系统设计的 FPGA 系统 ---

DingXT1/Railway_System: FPGA system designed for Railway System (github.com)

1.top_design：

```
`timescale 1ns / 1ps

module top_design(
    input sys_clk,  //50MHZ
    input [4:1]sw,    // 开关输入信号
    input [3:0] row,  //行
    input [4:1] key,
    input [2:0] signal,
    output [15:8] P2IO,//P2 引脚
    output [3:0] col,   //列
    output [7:0] seg_sel,    //4 个数码管选通信号输出
    output [7:0] seg_value,   //段码输出，正电平
    output [4:1] led    // 去抖后的开关输出信号
    );
    wire[3:0] key_value;
    wire key_valid;
    //顶层状态
    wire [3:0] top_state;
    //数码管对应值
    //state 0
    wire [15:0] start_station;
    wire [7:0 ] start_station_value;
    wire [15:0] end_station;
    wire [7:0 ] end_station_value;
    wire [7:0]  price;
    wire [15:0] refer_money;
    wire [3:0]  ticket_num;
    wire [15:0] total_money;//总票价
    wire [15:0] pay_auto;
    wire [7:0]  pay_auto_coin;
    wire [15:0] recharge_auto;

    wire [7:0] price_hand_value;
    wire [15:0] price_hand_show;
    wire [7:0] ticket_hand_value;
    wire [15:0] ticket_hand_show;
    wire [31:0] total_hand_show;
    wire [31:0] recharge_hand_show;
    wire [31:0] pay_hand_show;
```

```
    wire [13:0] pay_hand_coin;

    wire [13:0] pay_auto_bill;
    wire [13:0] pay_hand_bill;
    //数码管对应值
    wire [3:0] segdisp_state;



    //键盘扫描模块
    key44 U1
    (
        .clk(sys_clk),  //50MHZ
        .reset(1'b0),
        .row(row),  //行
        .col(col),   //列
        .io_sound(P2IO[8]),
        .key_valid(key_valid),
        .key_value(key_value)  //键值
    );

    //输入输出处理模块
    IO_deal U2
    (
        .clk(sys_clk),  //50MHZ
        .rst(1'b1),
        .key_value(key_value),  //键值
        .key_valid(key_valid),
        .top_state(top_state),
        .start_station_value(start_station_value),
        .end_station_value(end_station_value),
        .start_station(start_station),
        .end_station(end_station),
        .price_hand_value(price_hand_value),
        .price_hand_show(price_hand_show),
        .ticket_hand_value(ticket_hand_value),
        .ticket_hand_show(ticket_hand_show),
        .segdisp_state(segdisp_state)
    );

    //显示模块
    seg_dtxs U3
    (
        .clk(sys_clk),  //50MHZ
```

```
        .rst(1'b1),
        .top_state(top_state),
        .start_station(start_station),
        .end_station(end_station),
        .refer_money(refer_money),
        .total_money(total_money),
        .pay_auto(pay_auto),   //输入的钱【8+8】
        .recharge_auto(recharge_auto),   //找零的钱【8+8】
        .price_hand_show(price_hand_show),
        .ticket_hand_show(ticket_hand_show),
        .total_hand_show(total_hand_show),
        .pay_hand_show(pay_hand_show),//手动输入价钱【8+8+8+8】
        .recharge_hand_show(recharge_hand_show),//手动找零价钱【8+8+8+8】

        .segdisp_state(segdisp_state),
        .seg_sel(seg_sel),
        .seg_value(seg_value)

    );

    //计算票价模块
    Select U4
    (
        .clk(sys_clk),//50MHZ
        .start_station_value(start_station_value),
        .end_station_value(end_station_value),
        .price(price)
    );

    //开关模块
    switch U5(
        .clk(sys_clk),//50MHZ
        .sw(sw),    // 开关输入信号
        .led(led),     // 去抖后的开关输出信号
        .top_state(top_state)
    );

    //数值转数码管模块
    num2seg U6(
        .clk(sys_clk),//50MHZ
        .pay_auto_coin( pay_auto_coin),
        .pay_auto_bill( pay_auto_bill),
        .pay_hand_coin(pay_hand_coin),
        .pay_hand_bill(pay_hand_bill),
```

```
        .ticket_num(ticket_num),
        .price(price),
        .refer_money(refer_money),//数码管制，不是十进制
        .total_money(total_money),//数码管制，不是十进制
        .pay_auto(pay_auto),//数码管制，不是十进制
        .recharge_auto(recharge_auto),//数码管制，不是十进制
        .ticket_hand_value(ticket_hand_value),
        .price_hand_value(price_hand_value),
        .total_hand_show(total_hand_show),//数码管制，不是十进制
        .pay_hand_show(pay_hand_show),//数码管制，不是十进制
        .recharge_hand_show(recharge_hand_show)//数码管制，不是十进制
    );

    //按钮模块
    button U7(
        .clk(sys_clk),//50MHZ
        .key(key),
        .top_state(top_state),
        .ticket_num(ticket_num),
        .pay_auto_coin(pay_auto_coin),
        .pay_hand_coin(pay_hand_coin)
    );

    //通信模块
     Bill U8(
        .clk(sys_clk),
        .top_state(top_state),
        .signal(signal),
        .pay_auto_bill(pay_auto_bill),
        .pay_hand_bill(pay_hand_bill)
    );

Endmodule
```

2.IO_deal：

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2024/09/10 09:40:37
// Design Name:
// Module Name: IO_deal
```

```
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
///////////


////////////////////////////////////////////////////////////
//第二个模块!
module IO_deal(
 input clk,
 input rst,         // 输入时钟和复位信号
 input [3:0] top_state,
 input [4:1] key_value,
 input [1:1] key_valid,
 input [3:0] ticket_num,
 output reg [3:0]   segdisp_state,//显示状态（以两位为单位）
 output reg [7:0]   start_station_value,//十进制
 output reg [7:0]   end_station_value,//十进制
 output reg [15:0]  start_station,//数码管制，不是十进制
 output reg [15:0]  end_station,//数码管制，不是十进制
 output reg [7:0]   price_hand_value,
 output reg [15:0]  price_hand_show,
 output reg [7:0]   ticket_hand_value,
 output reg [15:0]  ticket_hand_show
);
reg [1:1]flag;
reg [15:0] i;
reg [15:0] j;
reg [3:0] start_one_unit;
reg [3:0] start_ten_unit;
reg [3:0] end_one_unit;
reg [3:0] end_ten_unit;
reg [3:0] phv_one_unit;
reg [3:0] phv_ten_unit;
reg [3:0] thv_one_unit;
reg [3:0] thv_ten_unit;
```

```
 initial
    begin
    for(i=0;i<4;i=i+1)
     begin segdisp_state[i]=0;end
     start_station_value=0;
     end_station_value=0;
     start_station=16'b1100_0000_1100_0000;
     end_station=16'b1100_0000_1100_0000;

     price_hand_value=0;
     ticket_hand_value=0;
     price_hand_show=16'b1100_0000_1100_0000;
     ticket_hand_show=16'b1100_0000_1100_0000;
     ////////////////////////////////////
    end



always @(negedge key_valid)
begin

    flag=0;
    //状态重置
    if(segdisp_state >= 4)
    segdisp_state=0;
    ////////////////////起点[1]////////////////////////////
    if(segdisp_state==0)
    begin
        //0010
        if(top_state==4'b0010)
        begin
            case (key_value)
            4'h0: begin price_hand_show [6:0] =
7'b1000000;  phv_ten_unit = 0; end
            4'h1: begin price_hand_show [6:0] =
7'b1111001;  phv_ten_unit = 1; end
            4'h2: begin price_hand_show [6:0] =
7'b0100100;  phv_ten_unit = 2; end
            4'h3: begin price_hand_show [6:0] =
7'b0110000;  phv_ten_unit = 3; end
            4'h4: begin price_hand_show [6:0] =
7'b0011001;  phv_ten_unit = 4; end
```

```
            4'h5: begin price_hand_show [6:0] =
7'b0010010;  phv_ten_unit = 5; end
            4'h6: begin price_hand_show [6:0] =
7'b0000010;  phv_ten_unit = 6; end
            4'h7: begin price_hand_show [6:0] =
7'b1111000;  phv_ten_unit = 7; end
            4'h8: begin price_hand_show [6:0] =
7'b0000000;  phv_ten_unit = 8; end
            4'h9: begin price_hand_show [6:0] =
7'b0010000;  phv_ten_unit = 9; end
            default :begin segdisp_state = segdisp_state +1;  end
            endcase
            price_hand_show [7] = 1;//共极，不用管
        end
        //0000
        else
        begin
            case (key_value)
            4'h0: begin start_station [6:0] =
7'b1000000;  start_ten_unit = 0; end
            4'h1: begin start_station [6:0] =
7'b1111001;  start_ten_unit = 1; end
            4'h2: begin start_station [6:0] =
7'b0100100;  start_ten_unit = 2; end
            4'h3: begin start_station [6:0] =
7'b0110000;  start_ten_unit = 3; end
            4'h4: begin start_station [6:0] =
7'b0011001;  start_ten_unit = 4; end
            4'h5: begin start_station [6:0] =
7'b0010010;  start_ten_unit = 5; end
            4'h6: begin start_station [6:0] =
7'b0000010;  start_ten_unit = 6; end
            4'h7: begin start_station [6:0] =
7'b1111000;  start_ten_unit = 7; end
            4'h8: begin start_station [6:0] =
7'b0000000;  start_ten_unit = 8; end
            4'h9: begin start_station [6:0] =
7'b0010000;  start_ten_unit = 9; end
            default :begin segdisp_state = segdisp_state +1;  end
            endcase
            start_station [7] = 1;//共极，不用管
        end

        flag=1;
```

```
    end
    ////////////////////起点[2]////////////////////////////
    if(segdisp_state==1 && flag==0 )
    begin
        //0010
        if(top_state==4'b0010)
        begin
            case (key_value)
            4'h0: begin price_hand_show [14:8] =
7'b1000000;  phv_one_unit = 0; end
            4'h1: begin price_hand_show [14:8] =
7'b1111001;  phv_one_unit = 1; end
            4'h2: begin price_hand_show [14:8] =
7'b0100100;  phv_one_unit = 2; end
            4'h3: begin price_hand_show [14:8] =
7'b0110000;  phv_one_unit = 3; end
            4'h4: begin price_hand_show [14:8] =
7'b0011001;  phv_one_unit = 4; end
            4'h5: begin price_hand_show [14:8] =
7'b0010010;  phv_one_unit = 5; end
            4'h6: begin price_hand_show [14:8] =
7'b0000010;  phv_one_unit = 6; end
            4'h7: begin price_hand_show [14:8] =
7'b1111000;  phv_one_unit = 7; end
            4'h8: begin price_hand_show [14:8] =
7'b0000000;  phv_one_unit = 8; end
            4'h9: begin price_hand_show [14:8] =
7'b0010000;  phv_one_unit = 9; end
            default :begin segdisp_state=segdisp_state+1;end
            endcase
            price_hand_show [15] =1;//共极，不用管
        end
        //0000
        else
        begin
            case (key_value)
            4'h0: begin start_station [14:8] =
7'b1000000;  start_one_unit = 0; end
            4'h1: begin start_station [14:8] =
7'b1111001;  start_one_unit = 1; end
            4'h2: begin start_station [14:8] =
7'b0100100;  start_one_unit = 2; end
            4'h3: begin start_station [14:8] =
7'b0110000;  start_one_unit = 3; end
```

```
            4'h4: begin start_station [14:8] =
7'b0011001;  start_one_unit = 4; end
            4'h5: begin start_station [14:8] =
7'b0010010;  start_one_unit = 5; end
            4'h6: begin start_station [14:8] =
7'b0000010;  start_one_unit = 6; end
            4'h7: begin start_station [14:8] =
7'b1111000;  start_one_unit = 7; end
            4'h8: begin start_station [14:8] =
7'b0000000;  start_one_unit = 8; end
            4'h9: begin start_station [14:8] =
7'b0010000;  start_one_unit = 9; end
            default :begin segdisp_state=segdisp_state+1;end
            endcase
            start_station [15] =1;//共极，不用管
        end

        flag=1;
    end

   ////////////////////终点[1]////////////////////////////
    if(segdisp_state==2 && flag==0)
    begin
         //0010
        if(top_state==4'b0010)
        begin
            case (key_value)
            4'h0: begin ticket_hand_show [6:0] =
7'b1000000;  thv_ten_unit = 0; end
            4'h1: begin ticket_hand_show [6:0] =
7'b1111001;  thv_ten_unit = 1; end
            4'h2: begin ticket_hand_show [6:0] =
7'b0100100;  thv_ten_unit = 2; end
            4'h3: begin ticket_hand_show [6:0] =
7'b0110000;  thv_ten_unit = 3; end
            4'h4: begin ticket_hand_show [6:0] =
7'b0011001;  thv_ten_unit = 4; end
            4'h5: begin ticket_hand_show [6:0] =
7'b0010010;  thv_ten_unit = 5; end
            4'h6: begin ticket_hand_show [6:0] =
7'b0000010;  thv_ten_unit = 6; end
            4'h7: begin ticket_hand_show [6:0] =
7'b1111000;  thv_ten_unit = 7; end
```

```
            4'h8: begin ticket_hand_show [6:0] =
7'b0000000;  thv_ten_unit = 8; end
            4'h9: begin ticket_hand_show [6:0] =
7'b0010000;  thv_ten_unit = 9; end
            default :begin segdisp_state = segdisp_state +1;  end
            endcase
            ticket_hand_show [7] = 1;//共极，不用管
        end
        //0000
        else
        begin
            case (key_value)
            4'h0: begin end_station [6:0] = 7'b1000000;  end_ten_unit =
0; end
            4'h1: begin end_station [6:0] = 7'b1111001;  end_ten_unit =
1; end
            4'h2: begin end_station [6:0] = 7'b0100100;  end_ten_unit =
2; end
            4'h3: begin end_station [6:0] = 7'b0110000;  end_ten_unit =
3; end
            4'h4: begin end_station [6:0] = 7'b0011001;  end_ten_unit =
4; end
            4'h5: begin end_station [6:0] = 7'b0010010;  end_ten_unit =
5; end
            4'h6: begin end_station [6:0] = 7'b0000010;  end_ten_unit =
6; end
            4'h7: begin end_station [6:0] = 7'b1111000;  end_ten_unit =
7; end
            4'h8: begin end_station [6:0] = 7'b0000000;  end_ten_unit =
8; end
            4'h9: begin end_station [6:0] = 7'b0010000;  end_ten_unit =
9; end
            default :begin segdisp_state=segdisp_state+1;end
            endcase
            end_station [7] = 1;//共极，不用管
        end

        flag=1;
    end

    ////////////////////终点[2]////////////////////////////
    if(segdisp_state==3 && flag==0)
    begin
         //0010
```

```
        if(top_state==4'b0010)
        begin
            case (key_value)
            4'h0: begin ticket_hand_show [14:8] =
7'b1000000;  thv_one_unit = 0; end
            4'h1: begin ticket_hand_show [14:8] =
7'b1111001;  thv_one_unit = 1; end
            4'h2: begin ticket_hand_show [14:8] =
7'b0100100;  thv_one_unit = 2; end
            4'h3: begin ticket_hand_show [14:8] =
7'b0110000;  thv_one_unit = 3; end
            4'h4: begin ticket_hand_show [14:8] =
7'b0011001;  thv_one_unit = 4; end
            4'h5: begin ticket_hand_show [14:8] =
7'b0010010;  thv_one_unit = 5; end
            4'h6: begin ticket_hand_show [14:8] =
7'b0000010;  thv_one_unit = 6; end
            4'h7: begin ticket_hand_show [14:8] =
7'b1111000;  thv_one_unit = 7; end
            4'h8: begin ticket_hand_show [14:8] =
7'b0000000;  thv_one_unit = 8; end
            4'h9: begin ticket_hand_show [14:8] =
7'b0010000;  thv_one_unit = 9; end
            default :begin segdisp_state = segdisp_state +1;  end
            endcase
            ticket_hand_show [15] = 1;//共极，不用管
        end
        //0000
        else
        begin
            case (key_value)
            4'h0: begin end_station [14:8] = 7'b1000000;  end_one_unit =
0; end
            4'h1: begin end_station [14:8] = 7'b1111001;  end_one_unit =
1; end
            4'h2: begin end_station [14:8] = 7'b0100100;  end_one_unit =
2; end
            4'h3: begin end_station [14:8] = 7'b0110000;  end_one_unit =
3; end
            4'h4: begin end_station [14:8] = 7'b0011001;  end_one_unit =
4; end
            4'h5: begin end_station [14:8] = 7'b0010010;  end_one_unit =
5; end
```

```
            4'h6: begin end_station [14:8] = 7'b0000010;  end_one_unit =
6; end
            4'h7: begin end_station [14:8] = 7'b1111000;  end_one_unit =
7; end
            4'h8: begin end_station [14:8] = 7'b0000000;  end_one_unit =
8; end
            4'h9: begin end_station [14:8] = 7'b0010000;  end_one_unit =
9; end
            default :begin segdisp_state=segdisp_state+1;end
            endcase
            end_station [15] = 1;//共极，不用管
        end

        flag=1;
    end

    start_station_value=start_ten_unit*10+start_one_unit;
    end_station_value=end_ten_unit*10+end_one_unit;
    price_hand_value=phv_ten_unit*10+phv_one_unit;
    ticket_hand_value=thv_ten_unit*10+thv_one_unit;

end
endmodule
```

3. seg_dtxs：

```
`timescale 1ns / 1ps
module seg_dtxs(
    input clk,
    input rst,           // 输入时钟和复位信号
    input [3:0]  top_state,
    input [15:0]  start_station,  //起始站点【8+8】
    input [15:0]  end_station,   //结束站点【8+8】
    ////////////////////////////////////////////
    input [15:0]  refer_money,   //单票价【8+8】
    input [15:0]  total_money,   //总票价【8+8】
    input [15:0]  pay_auto,   //输入的钱【8+8】
    input [15:0]  recharge_auto,   //找零的钱【8+8】
    ////////////////////////////////////////////
    input [15:0] price_hand_show,//手动单票价【8+8】
    input [15:0] ticket_hand_show,//手动票数【8+8】
    input [31:0] total_hand_show,//手动总票价【8+8+8+8】
    input [31:0] pay_hand_show,//手动输入价钱【8+8+8+8】
    input [31:0] recharge_hand_show,//手动找零价钱【8+8+8+8】
```

```
    input [3:0]  segdisp_state,//数码管显示状态的 4 位数值。
    output reg [7:0] seg_value,        // 输出 8 位数码管【显示数据】
    output reg [7:0] seg_sel        // 输出 8 位数码管共阳/共阴【控制信号】
    );


    reg [3:0] num;            // 上级计数器，用于选择当前激活的数码管
    reg [16:0] counter;      // 下级计数器，用于产生时间延迟
    reg i;
 initial
     begin
    for(i=0;i<8;i=i+1)
    begin
     seg_sel[i]=0;
    end
    seg_value=8'b11000000;
    end

    // 产生 1ms 延时【根据时钟信号】
    always @(posedge clk or negedge rst) begin
        if (!rst)
            counter <= 0;    // 复位时，计数器清零
        else if (counter == 99_999)
            counter <= 0;    // 达到 1ms 后归零
        else
            counter <= counter + 1; // 否则计数加一
    end

     ////////////////////////////////////////////////////////////////////
//////////////
    //每隔 1ms，num 计数器加 1，直到 7，然后归零。
    //num 用选择当前激活的数码管。
    always @(posedge clk or negedge rst) begin
        if (!rst)
            num <= 0;         // 复位时，dm 清零
        else if ((counter == 99_999) && (num == 7))//【1ms】+num 为 7
            num <= 0;         // 归零
        else if (counter == 99_999)
            num <= num + 1;    // 每 1ms num 加 1
        else
            num <= num;        // 否则保持不变
    end
```

```
    ////////////////////////////////////////////////////////////////////////
/////////////
    //数码管显示，每隔 1ms
    always @(posedge clk) begin
        case (num)
            0: begin
                seg_sel= 8'b0111_1111;
                ////////////////////////////////////////////////
                case(top_state)
                //状态 2
                4'b0010:begin seg_value = price_hand_show[7:0]; end
                 //状态 3
                4'b0011:begin seg_value = pay_hand_show[7:0]; end
                //状态 0+状态 1
                default:begin seg_value = start_station[7:0];  end
                endcase
                ////////////////////////////////////////////////
                end

            1: begin
                seg_sel= 8'b1011_1111;
                ////////////////////////////////////////////////
                case(top_state)
                //状态 2
                4'b0010:begin seg_value = price_hand_show[15:8]; end
                 //状态 3
                4'b0011:begin seg_value = pay_hand_show[15:8]; end
                //状态 0+状态 1
                default:begin seg_value = start_station[15:8];  end
                endcase
                ////////////////////////////////////////////////
               end

            2: begin
                seg_sel= 8'b1101_1111;
                ////////////////////////////////////////////////
                case(top_state)
                //状态 2
                4'b0010:begin seg_value = ticket_hand_show[7:0]; end
                 //状态 3
                4'b0011:begin seg_value = pay_hand_show[23:16]; end
                //状态 0+状态 1
                default:begin seg_value = end_station[7:0];  end
                endcase
```

```
        ////////////////////////////////////////////////

    end

    3: begin
        seg_sel= 8'b1110_1111;
        ////////////////////////////////////////////////
        case(top_state)
        //状态 2
        4'b0010:begin seg_value = ticket_hand_show[15:8]; end
         //状态 3
        4'b0011:begin seg_value = pay_hand_show[31:24]; end
        //状态 0+状态 1
        default:begin seg_value = end_station[15:8];  end
        endcase
        ////////////////////////////////////////////////

       end

    4: begin
        seg_sel= 8'b1111_0111;
        ////////////////////////////////////////////////
        case(top_state)
        //状态 1
        4'b0001:begin seg_value =  pay_auto[7:0]; end
        //状态 2
        4'b0010:begin seg_value = total_hand_show[7:0]; end
         //状态 3
        4'b0011:begin seg_value = recharge_hand_show[7:0]; end
        //状态 0
        default:begin seg_value = refer_money[7:0];  end
        endcase
        ////////////////////////////////////////////////
       end

    5: begin
        seg_sel= 8'b1111_1011;
        ////////////////////////////////////////////////
        case(top_state)
        //状态 1
        4'b0001:begin seg_value =  pay_auto[15:8]; end
        //状态 2
        4'b0010:begin seg_value = total_hand_show[15:8];  end
         //状态 3
```

```
            4'b0011:begin seg_value = recharge_hand_show[15:8]; end
            //状态 0
            default:begin seg_value = refer_money[15:8];  end
            endcase
            ////////////////////////////////////////////
           end

        6: begin
            seg_sel= 8'b1111_1101;
            ////////////////////////////////////////////
            case(top_state)
            //状态 1
            4'b0001:begin seg_value =  recharge_auto[7:0]; end
            //状态 2
            4'b0010:begin seg_value = total_hand_show[23:16];  end
             //状态 3
            4'b0011:begin seg_value = recharge_hand_show[23:16]; end
            //状态 0
            default:begin seg_value = total_money[7:0];  end
            endcase
            ////////////////////////////////////////////

            end

        7: begin
            seg_sel= 8'b1111_1110;
            ////////////////////////////////////////////
            case(top_state)
            //状态 1
            4'b0001:begin seg_value =  recharge_auto[15:8]; end
            //状态 2
            4'b0010:begin seg_value = total_hand_show[31:24];  end
             //状态 3
            4'b0011:begin seg_value = recharge_hand_show[31:24]; end
            //状态 0
            default:begin seg_value = total_money[15:8];  end
            endcase
            ////////////////////////////////////////////

            end

        //原则上不会执行
        default: seg_sel <= 8'b0111_1111; // 默认激活第一个数码管
    endcase
```

```
    end

endmodule
```

4. select：

```
//////////////////////////////////////////////////
module Select(
input clk,
input [7:0] start_station_value,
input [7:0] end_station_value,
output reg  [7:0] price
);
initial
begin
    price=0;
end

always @(clk) begin

if(start_station_value == 0 && end_station_value == 0)
    price = 0;
else
if(start_station_value == 0 && end_station_value == 1)
    price = 2;
else
if(start_station_value == 0 && end_station_value == 2)
    price = 2;
else
if(start_station_value == 0 && end_station_value == 3)
    price = 2;
else
if(start_station_value == 0 && end_station_value == 4)
    price = 3;
```

……（省略上万行）

```
if(start_station_value == 61 && end_station_value == 59)
    price = 2;
else
if(start_station_value == 61 && end_station_value == 60)
    price = 2;
else
if(start_station_value == 61 && end_station_value == 61)
    price = 0;
else
if(start_station_value == 61 && end_station_value == 62)
```

```
    price = 2;

end

endmodule
```

5.switch：

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////
// Company:
// Engineer:
//
// Create Date: 2024/09/10 15:26:11
// Design Name:
// Module Name: switch
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
////////////
module switch(
    input clk,          // 时钟信号
    input [4:1]sw,    // 开关输入信号
    output reg [4:1]led,     // 去抖后的开关输出信号
    output reg [3:0]top_state
);
initial
begin
top_state=4'b0000;
end
//其他所有状态--状态 0
//if(top_state ~= 0001 && top_state ~= 0011 &&  top_state ~= 0111
&&  top_state ~= 1111)
//0001--状态 1
```

```
//0011--状态 2
//0111--状态 3
//1111--状态 4


    always @(posedge clk)
    begin
        led[1]=~sw[1];//下拨为 1，上拨为 0
        top_state[0]=~sw[1];
        led[2]=~sw[2];
        top_state[1]=~sw[2];
        led[3]=~sw[3];
        top_state[2]=~sw[3];
        led[4]=~sw[4];
        top_state[3]=~sw[4];
    end



endmodule
```

```

```

6.num2seg：

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////
// Company:
// Engineer:
//
// Create Date: 2024/09/10 15:44:15
// Design Name:
// Module Name: State_one
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
```

```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
///////////


module num2seg(
input clk,
input [3:0]  ticket_num,
input [7:0]  price,
input [7:0]  pay_auto_coin,
input [13:0] pay_auto_bill,
input [7:0]  ticket_hand_value,
input [7:0]  price_hand_value,
input [13:0] pay_hand_coin,
input [13:0] pay_hand_bill,

output reg [15:0] refer_money,//数码管制，不是十进制
output reg [15:0] total_money,//数码管制，不是十进制
output reg [15:0] pay_auto,//数码管制，不是十进制
output reg [15:0] recharge_auto,//数码管制，不是十进制

output reg [31:0] total_hand_show,//数码管制，不是十进制
output reg [31:0] pay_hand_show,//数码管制，不是十进制
output reg [31:0] recharge_hand_show//数码管制，不是十进制
    );

    reg [3:0]ten_unit;
    reg [3:0]one_unit;

    reg [7:0]total;
    reg [3:0]ten_unit_2;
    reg [3:0]one_unit_2;

    reg [7:0]pay_auto_value;
    reg [3:0]ten_unit_3;
    reg [3:0]one_unit_3;

    reg [7:0]recharge_auto_value;
    reg [3:0]ten_unit_4;
    reg [3:0]one_unit_4;

    reg [13:0]total_hand_value;
    reg [13:0]total_hand_temp;
    reg [3:0]thou_unit_5;
```

```
    reg [3:0]hun_unit_5;
    reg [3:0]ten_unit_5;
    reg [3:0]one_unit_5;

    reg [13:0]pay_hand_value;
    reg [13:0]pay_hand_temp;
    reg [3:0]thou_unit_6;
    reg [3:0]hun_unit_6;
    reg [3:0]ten_unit_6;
    reg [3:0]one_unit_6;

    reg [13:0]recharge_hand_value;
    reg [13:0]recharge_hand_temp;
    reg [3:0]thou_unit_7;
    reg [3:0]hun_unit_7;
    reg [3:0]ten_unit_7;
    reg [3:0]one_unit_7;

    reg [7:0]test;
    initial
    begin
        refer_money=16'b1111_1001_1111_1001;
        total_money=16'b1111_1001_1111_1001;
        total_hand_show=32'b1100_0000_1100_0000_1100_0000_1100_0000;
        pay_hand_show=32'b1100_0000_1100_0000_1100_0000_1100_0000;//数码
管制，不是十进制
        recharge_hand_show=32'b1000_0110_1000_0110_1000_0110_1000_0110;/
/数码管制，不是十进制
    end
    always@ (posedge clk)
    begin

            ten_unit=price / 10;
            one_unit=price % 10;

            total=ticket_num*price;
            ten_unit_2=total / 10;
            one_unit_2=total % 10;

            pay_auto_value=pay_auto_coin+pay_auto_bill;
            ten_unit_3=pay_auto_value / 10;
            one_unit_3=pay_auto_value % 10;
```

```
            total_hand_value=ticket_hand_value*
price_hand_value;
            total_hand_temp=total_hand_value;
            one_unit_5=total_hand_temp % 10;//提取个位
            total_hand_temp=total_hand_temp/10;
            ten_unit_5=total_hand_temp % 10; //提取十位
            total_hand_temp=total_hand_temp/10;
            hun_unit_5=total_hand_temp % 10; //提取百位
            total_hand_temp=total_hand_temp/10;
            thou_unit_5=total_hand_temp % 10;  //提取千位

            pay_hand_value = pay_hand_coin + pay_hand_bill;
            pay_hand_temp=pay_hand_value;
            one_unit_6=pay_hand_temp % 10;//提取个位
            pay_hand_temp=pay_hand_temp/10;
            ten_unit_6=pay_hand_temp % 10; //提取十位
            pay_hand_temp=pay_hand_temp/10;
            hun_unit_6=pay_hand_temp % 10; //提取百位
            pay_hand_temp=pay_hand_temp/10;
            thou_unit_6=pay_hand_temp % 10;  //提取千位

            //单票价
            case (ten_unit)
            4'h0: begin refer_money [6:0] = 7'b1000000;    end
            4'h1: begin refer_money [6:0] = 7'b1111001;    end
            4'h2: begin refer_money [6:0] = 7'b0100100;    end
            4'h3: begin refer_money [6:0] = 7'b0110000;    end
            4'h4: begin refer_money [6:0] = 7'b0011001;    end
            4'h5: begin refer_money [6:0] = 7'b0010010;    end
            4'h6: begin refer_money [6:0] = 7'b0000010;    end
            4'h7: begin refer_money [6:0] = 7'b1111000;    end
            4'h8: begin refer_money [6:0] = 7'b0000000;    end
            4'h9: begin refer_money [6:0] = 7'b0010000;    end
            default:refer_money [6:0]=7'b111_1111;
            endcase
            refer_money[7]=1;

            case (one_unit)
            4'h0: begin refer_money [14:8] = 7'b1000000;    end
            4'h1: begin refer_money [14:8] = 7'b1111001;    end
            4'h2: begin refer_money [14:8] = 7'b0100100;    end
            4'h3: begin refer_money [14:8] = 7'b0110000;    end
            4'h4: begin refer_money [14:8] = 7'b0011001;    end
            4'h5: begin refer_money [14:8] = 7'b0010010;    end
```

```
4'h6: begin refer_money [14:8] = 7'b0000010;    end
4'h7: begin refer_money [14:8] = 7'b1111000;    end
4'h8: begin refer_money [14:8] = 7'b0000000;    end
4'h9: begin refer_money [14:8] = 7'b0010000;    end
default:refer_money [14:8]=7'b111_1111;
endcase
refer_money[15]=1;

//总票价
case (ten_unit_2)
4'h0: begin total_money [6:0] = 7'b1000000;    end
4'h1: begin total_money [6:0] = 7'b1111001;    end
4'h2: begin total_money [6:0] = 7'b0100100;    end
4'h3: begin total_money [6:0] = 7'b0110000;    end
4'h4: begin total_money [6:0] = 7'b0011001;    end
4'h5: begin total_money [6:0] = 7'b0010010;    end
4'h6: begin total_money [6:0] = 7'b0000010;    end
4'h7: begin total_money [6:0] = 7'b1111000;    end
4'h8: begin total_money [6:0] = 7'b0000000;    end
4'h9: begin total_money [6:0] = 7'b0010000;    end
default:total_money [6:0]=7'b111_1111;
endcase
total_money[7]=1;

case (one_unit_2)
4'h0: begin total_money [14:8] = 7'b1000000;    end
4'h1: begin total_money [14:8] = 7'b1111001;    end
4'h2: begin total_money [14:8] = 7'b0100100;    end
4'h3: begin total_money [14:8] = 7'b0110000;    end
4'h4: begin total_money [14:8] = 7'b0011001;    end
4'h5: begin total_money [14:8] = 7'b0010010;    end
4'h6: begin total_money [14:8] = 7'b0000010;    end
4'h7: begin total_money [14:8] = 7'b1111000;    end
4'h8: begin total_money [14:8] = 7'b0000000;    end
4'h9: begin total_money [14:8] = 7'b0010000;    end
default:total_money [14:8]=7'b111_1111;
endcase
total_money[15]=1;

//输入价钱
case (ten_unit_3)
4'h0: begin pay_auto [6:0] = 7'b1000000;    end
4'h1: begin pay_auto [6:0] = 7'b1111001;    end
4'h2: begin pay_auto [6:0] = 7'b0100100;    end
```

```
            4'h3: begin pay_auto [6:0] = 7'b0110000;    end
            4'h4: begin pay_auto [6:0] = 7'b0011001;    end
            4'h5: begin pay_auto [6:0] = 7'b0010010;    end
            4'h6: begin pay_auto [6:0] = 7'b0000010;    end
            4'h7: begin pay_auto [6:0] = 7'b1111000;    end
            4'h8: begin pay_auto [6:0] = 7'b0000000;    end
            4'h9: begin pay_auto [6:0] = 7'b0010000;    end
            default:pay_auto [6:0]=7'b111_1111;
            endcase
            pay_auto[7]=1;

            case (one_unit_3)
            4'h0: begin pay_auto [14:8] = 7'b1000000;    end
            4'h1: begin pay_auto [14:8] = 7'b1111001;    end
            4'h2: begin pay_auto [14:8] = 7'b0100100;    end
            4'h3: begin pay_auto [14:8] = 7'b0110000;    end
            4'h4: begin pay_auto [14:8] = 7'b0011001;    end
            4'h5: begin pay_auto [14:8] = 7'b0010010;    end
            4'h6: begin pay_auto [14:8] = 7'b0000010;    end
            4'h7: begin pay_auto [14:8] = 7'b1111000;    end
            4'h8: begin pay_auto [14:8] = 7'b0000000;    end
            4'h9: begin pay_auto [14:8] = 7'b0010000;    end
            default:pay_auto [14:8]=7'b111_1111;
            endcase
            pay_auto[15]=1;

            //找零价钱
            if(pay_auto_value < total)begin
recharge_auto[15:8]=8'b1000_0110;recharge_auto[7:0]=8'b1000_0110; end
            else
            begin
                recharge_auto_value=pay_auto_value-total;
                ten_unit_4=recharge_auto_value / 10;
                one_unit_4=recharge_auto_value % 10;

                case (ten_unit_4)
                4'h0: begin recharge_auto [6:0] = 7'b1000000;    end
                4'h1: begin recharge_auto [6:0] = 7'b1111001;    end
                4'h2: begin recharge_auto [6:0] = 7'b0100100;    end
                4'h3: begin recharge_auto [6:0] = 7'b0110000;    end
                4'h4: begin recharge_auto [6:0] = 7'b0011001;    end
                4'h5: begin recharge_auto [6:0] = 7'b0010010;    end
                4'h6: begin recharge_auto [6:0] = 7'b0000010;    end
                4'h7: begin recharge_auto [6:0] = 7'b1111000;    end
```

```
            4'h8: begin recharge_auto [6:0] = 7'b0000000;    end
            4'h9: begin recharge_auto [6:0] = 7'b0010000;    end
            default:recharge_auto [6:0]=7'b111_1111;
            endcase
            recharge_auto[7]=1;

            case (one_unit_4)
            4'h0: begin recharge_auto [14:8] = 7'b1000000;    end
            4'h1: begin recharge_auto [14:8] = 7'b1111001;    end
            4'h2: begin recharge_auto [14:8] = 7'b0100100;    end
            4'h3: begin recharge_auto [14:8] = 7'b0110000;    end
            4'h4: begin recharge_auto [14:8] = 7'b0011001;    end
            4'h5: begin recharge_auto [14:8] = 7'b0010010;    end
            4'h6: begin recharge_auto [14:8] = 7'b0000010;    end
            4'h7: begin recharge_auto [14:8] = 7'b1111000;    end
            4'h8: begin recharge_auto [14:8] = 7'b0000000;    end
            4'h9: begin recharge_auto [14:8] = 7'b0010000;    end
            default:recharge_auto [14:8]=7'b111_1111;
            endcase
            recharge_auto[15]=1;
        end

        //手动——总价钱
        case (thou_unit_5)
        4'h0: begin total_hand_show [6:0] = 7'b1000000;    end
        4'h1: begin total_hand_show [6:0] = 7'b1111001;    end
        4'h2: begin total_hand_show [6:0] = 7'b0100100;    end
        4'h3: begin total_hand_show [6:0] = 7'b0110000;    end
        4'h4: begin total_hand_show [6:0] = 7'b0011001;    end
        4'h5: begin total_hand_show [6:0] = 7'b0010010;    end
        4'h6: begin total_hand_show [6:0] = 7'b0000010;    end
        4'h7: begin total_hand_show [6:0] = 7'b1111000;    end
        4'h8: begin total_hand_show [6:0] = 7'b0000000;    end
        4'h9: begin total_hand_show [6:0] = 7'b0010000;    end
        default:total_hand_show [6:0]=7'b111_1111;
        endcase
        total_hand_show[7]=1;

        case (hun_unit_5)
        4'h0: begin total_hand_show [14:8] = 7'b1000000;    end
        4'h1: begin total_hand_show [14:8] = 7'b1111001;    end
        4'h2: begin total_hand_show [14:8] = 7'b0100100;    end
        4'h3: begin total_hand_show [14:8] = 7'b0110000;    end
        4'h4: begin total_hand_show [14:8] = 7'b0011001;    end
```

```
            4'h5: begin total_hand_show [14:8] = 7'b0010010;    end
            4'h6: begin total_hand_show [14:8] = 7'b0000010;    end
            4'h7: begin total_hand_show [14:8] = 7'b1111000;    end
            4'h8: begin total_hand_show [14:8] = 7'b0000000;    end
            4'h9: begin total_hand_show [14:8] = 7'b0010000;    end
            default:total_hand_show [14:8]=7'b111_1111;
            endcase
            total_hand_show[15]=1;

            case (ten_unit_5)
            4'h0: begin total_hand_show [22:16] = 7'b1000000;    end
            4'h1: begin total_hand_show [22:16] = 7'b1111001;    end
            4'h2: begin total_hand_show [22:16] = 7'b0100100;    end
            4'h3: begin total_hand_show [22:16] = 7'b0110000;    end
            4'h4: begin total_hand_show [22:16] = 7'b0011001;    end
            4'h5: begin total_hand_show [22:16] = 7'b0010010;    end
            4'h6: begin total_hand_show [22:16] = 7'b0000010;    end
            4'h7: begin total_hand_show [22:16] = 7'b1111000;    end
            4'h8: begin total_hand_show [22:16] = 7'b0000000;    end
            4'h9: begin total_hand_show [22:16] = 7'b0010000;    end
            default:total_hand_show [22:16]=7'b111_1111;
            endcase
            total_hand_show[23]=1;

            case (one_unit_5)
            4'h0: begin total_hand_show [30:24] = 7'b1000000;    end
            4'h1: begin total_hand_show [30:24] = 7'b1111001;    end
            4'h2: begin total_hand_show [30:24] = 7'b0100100;    end
            4'h3: begin total_hand_show [30:24] = 7'b0110000;    end
            4'h4: begin total_hand_show [30:24] = 7'b0011001;    end
            4'h5: begin total_hand_show [30:24] = 7'b0010010;    end
            4'h6: begin total_hand_show [30:24] = 7'b0000010;    end
            4'h7: begin total_hand_show [30:24] = 7'b1111000;    end
            4'h8: begin total_hand_show [30:24] = 7'b0000000;    end
            4'h9: begin total_hand_show [30:24] = 7'b0010000;    end
            default:total_hand_show [30:24]=7'b111_1111;
            endcase
            total_hand_show[31]=1;

            //手动—输入价钱
            case (thou_unit_6)
            4'h0: begin pay_hand_show [6:0] = 7'b1000000;    end
            4'h1: begin pay_hand_show [6:0] = 7'b1111001;    end
            4'h2: begin pay_hand_show [6:0] = 7'b0100100;    end
```

```
            4'h3: begin pay_hand_show [6:0] = 7'b0110000;    end
            4'h4: begin pay_hand_show [6:0] = 7'b0011001;    end
            4'h5: begin pay_hand_show [6:0] = 7'b0010010;    end
            4'h6: begin pay_hand_show [6:0] = 7'b0000010;    end
            4'h7: begin pay_hand_show [6:0] = 7'b1111000;    end
            4'h8: begin pay_hand_show [6:0] = 7'b0000000;    end
            4'h9: begin pay_hand_show [6:0] = 7'b0010000;    end
            default:pay_hand_show [6:0]=7'b0011001;
            endcase
            pay_hand_show[7]=1;

            case (hun_unit_6)
            4'h0: begin pay_hand_show [14:8] = 7'b1000000;    end
            4'h1: begin pay_hand_show [14:8] = 7'b1111001;    end
            4'h2: begin pay_hand_show [14:8] = 7'b0100100;    end
            4'h3: begin pay_hand_show [14:8] = 7'b0110000;    end
            4'h4: begin pay_hand_show [14:8] = 7'b0011001;    end
            4'h5: begin pay_hand_show [14:8] = 7'b0010010;    end
            4'h6: begin pay_hand_show [14:8] = 7'b0000010;    end
            4'h7: begin pay_hand_show [14:8] = 7'b1111000;    end
            4'h8: begin pay_hand_show [14:8] = 7'b0000000;    end
            4'h9: begin pay_hand_show [14:8] = 7'b0010000;    end
            default:pay_hand_show [14:8]=7'b111_1111;
            endcase
            pay_hand_show[15]=1;

            case (ten_unit_6)
            4'h0: begin pay_hand_show [22:16] = 7'b1000000;    end
            4'h1: begin pay_hand_show [22:16] = 7'b1111001;    end
            4'h2: begin pay_hand_show [22:16] = 7'b0100100;    end
            4'h3: begin pay_hand_show [22:16] = 7'b0110000;    end
            4'h4: begin pay_hand_show [22:16] = 7'b0011001;    end
            4'h5: begin pay_hand_show [22:16] = 7'b0010010;    end
            4'h6: begin pay_hand_show [22:16] = 7'b0000010;    end
            4'h7: begin pay_hand_show [22:16] = 7'b1111000;    end
            4'h8: begin pay_hand_show [22:16] = 7'b0000000;    end
            4'h9: begin pay_hand_show [22:16] = 7'b0010000;    end
            default:pay_hand_show [22:16]=7'b111_1111;
            endcase
            pay_hand_show[23]=1;

            case (one_unit_6)
            4'h0: begin pay_hand_show [30:24] = 7'b1000000;    end
            4'h1: begin pay_hand_show [30:24] = 7'b1111001;    end
```

```
            4'h2: begin pay_hand_show [30:24] = 7'b0100100;    end
            4'h3: begin pay_hand_show [30:24] = 7'b0110000;    end
            4'h4: begin pay_hand_show [30:24] = 7'b0011001;    end
            4'h5: begin pay_hand_show [30:24] = 7'b0010010;    end
            4'h6: begin pay_hand_show [30:24] = 7'b0000010;    end
            4'h7: begin pay_hand_show [30:24] = 7'b1111000;    end
            4'h8: begin pay_hand_show [30:24] = 7'b0000000;    end
            4'h9: begin pay_hand_show [30:24] = 7'b0010000;    end
            default:pay_hand_show [30:24]=7'b111_1111;
            endcase
            pay_hand_show[31]=1;

            //手动-找零价钱
            if(pay_hand_value < total_hand_value)
                begin
                recharge_hand_show[31:16]=16'b1000_0110_1000_0110;
                recharge_hand_show[15:0]=16'b1000_0110_1000_0110;
                end
            else
            begin
                recharge_hand_value=pay_hand_value-
total_hand_value;
                recharge_hand_temp=recharge_hand_value;
                one_unit_7=recharge_hand_temp % 10;//提取个位
                recharge_hand_temp=recharge_hand_temp/10;
                ten_unit_7=recharge_hand_temp % 10; //提取十位
                recharge_hand_temp=recharge_hand_temp/10;
                hun_unit_7=recharge_hand_temp % 10; //提取百位
                recharge_hand_temp=recharge_hand_temp/10;
                thou_unit_7=recharge_hand_temp % 10;  //提取千位

            case (thou_unit_7)
            4'h0: begin recharge_hand_show [6:0] = 7'b1000000;    end
            4'h1: begin recharge_hand_show [6:0] = 7'b1111001;    end
            4'h2: begin recharge_hand_show [6:0] = 7'b0100100;    end
            4'h3: begin recharge_hand_show [6:0] = 7'b0110000;    end
            4'h4: begin recharge_hand_show [6:0] = 7'b0011001;    end
            4'h5: begin recharge_hand_show [6:0] = 7'b0010010;    end
            4'h6: begin recharge_hand_show [6:0] = 7'b0000010;    end
            4'h7: begin recharge_hand_show [6:0] = 7'b1111000;    end
            4'h8: begin recharge_hand_show [6:0] = 7'b0000000;    end
            4'h9: begin recharge_hand_show [6:0] = 7'b0010000;    end
            default:recharge_hand_show [6:0]=7'b111_1111;
            endcase
```

```
            recharge_hand_show[7]=1;

            case (hun_unit_7)
            4'h0: begin recharge_hand_show [14:8] = 7'b1000000;    end
            4'h1: begin recharge_hand_show [14:8] = 7'b1111001;    end
            4'h2: begin recharge_hand_show [14:8] = 7'b0100100;    end
            4'h3: begin recharge_hand_show [14:8] = 7'b0110000;    end
            4'h4: begin recharge_hand_show [14:8] = 7'b0011001;    end
            4'h5: begin recharge_hand_show [14:8] = 7'b0010010;    end
            4'h6: begin recharge_hand_show [14:8] = 7'b0000010;    end
            4'h7: begin recharge_hand_show [14:8] = 7'b1111000;    end
            4'h8: begin recharge_hand_show [14:8] = 7'b0000000;    end
            4'h9: begin recharge_hand_show [14:8] = 7'b0010000;    end
            default:recharge_hand_show [14:8]=7'b111_1111;
            endcase
            recharge_hand_show[15]=1;

            case (ten_unit_7)
            4'h0: begin recharge_hand_show [22:16] = 7'b1000000;    end
            4'h1: begin recharge_hand_show [22:16] = 7'b1111001;    end
            4'h2: begin recharge_hand_show [22:16] =
7'b0100100;    end
            4'h3: begin recharge_hand_show [22:16] = 7'b0110000;    end
            4'h4: begin recharge_hand_show [22:16] =
7'b0011001;    end
            4'h5: begin recharge_hand_show [22:16] = 7'b0010010;    end
            4'h6: begin recharge_hand_show [22:16] = 7'b0000010;    end
            4'h7: begin recharge_hand_show [22:16] = 7'b1111000;    end
            4'h8: begin recharge_hand_show [22:16] = 7'b0000000;    end
            4'h9: begin recharge_hand_show [22:16] = 7'b0010000;    end
            default:recharge_hand_show [22:16]=7'b111_1111;
            endcase
            recharge_hand_show[23]=1;

            case (one_unit_7)
            4'h0: begin recharge_hand_show [30:24] = 7'b1000000;    end
            4'h1: begin recharge_hand_show [30:24] = 7'b1111001;    end
            4'h2: begin recharge_hand_show [30:24] =
7'b0100100;    end
            4'h3: begin recharge_hand_show [30:24] = 7'b0110000;    end
            4'h4: begin recharge_hand_show [30:24] =
7'b0011001;    end
            4'h5: begin recharge_hand_show [30:24] = 7'b0010010;    end
            4'h6: begin recharge_hand_show [30:24] = 7'b0000010;    end
```

```
            4'h7: begin recharge_hand_show [30:24] = 7'b1111000;    end
            4'h8: begin recharge_hand_show [30:24] = 7'b0000000;    end
            4'h9: begin recharge_hand_show [30:24] = 7'b0010000;    end
            default:recharge_hand_show [30:24]=7'b111_1111;
            endcase
            recharge_hand_show[31]=1;

            end
    end

endmodule
```

7.button：

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////
// Company:
// Engineer:
//
// Create Date: 2024/09/12 01:44:01
// Design Name:
// Module Name: get_ticket_num
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
////////////
module button(
    input  clk,
    input  [4:1]key,
    input  [3:0]top_state,
    output reg  [3:0] ticket_num,
    output reg  [7:0] pay_auto_coin,
    output reg  [13:0] pay_hand_coin
    );
    reg[3:0]neg_ticket_num;
```

```
    reg[3:0]pos_ticket_num;

    parameter N = 20;
    parameter MAX_COUNT = (1 << N) - 1;

    reg [4:0]flag;
    reg [N-1:0] counter [7:0];
    reg [4:1] key_in_d1, key_in_d2;
    initial
    begin
        pos_ticket_num=1;
        neg_ticket_num=0;
        ticket_num=1;
        pay_auto_coin=-1;
        pay_hand_coin=0;
        //初始状态必须为0000
    end

    always @(posedge clk) begin
        key_in_d1 <= key;
        key_in_d2 <= key_in_d1;
    end


    integer i;
    always @(posedge clk) begin
        for (i = 1; i < 5; i = i + 1) begin
            if (key_in_d2[i] == flag[i]) begin  counter[i] <= 0; end
            else begin
                counter[i] <= counter[i] + 1;
                if (counter[i] == MAX_COUNT) begin
                    flag [i] <= key_in_d2[i];
                end
            end
        end
    end

    always@(posedge flag[1])
    begin
        pos_ticket_num =pos_ticket_num +1;
    end

   always@(posedge flag[2])
    begin
```

```
        if(neg_ticket_num < pos_ticket_num)begin neg_ticket_num
=neg_ticket_num +1; end
    end

    always@(posedge flag[3])
    begin
        if(top_state==4'b0010 || top_state==4'b0011)begin
pay_hand_coin=pay_hand_coin+1;end
        else begin pay_auto_coin=pay_auto_coin+1;  end
    end

    always@(posedge clk)
    begin
         ticket_num = pos_ticket_num - neg_ticket_num;
    end

endmodule
```

8.Bill：

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2024/09/12 20:08:49
// Design Name:
// Module Name: Bill
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module Bill(
    input clk,
```

```
    input [3:0] top_state,
    input [2:0] signal,
    output reg [13:0] pay_auto_bill,
    output reg [13:0] pay_hand_bill
    );

    reg [1:0] bill_state;
    reg [7:0] auto_bill;
    reg [7:0] hand_bill;
    reg [1:0] bill_flag;

    initial
    begin
        auto_bill=0;
        hand_bill=0;

    end

  always@(clk)
    begin
        bill_flag=signal[0];
    end

    always @(posedge bill_flag)
    begin
            bill_state[0]=signal[2];
            bill_state[1]=signal[1];
    //状态 2+状态 3
        if(top_state==4'b0010 || top_state==4'b0011)
        begin
             case(bill_state)
                2'b00:begin hand_bill=10; end
                2'b01:begin hand_bill=20; end
                2'b10:begin hand_bill=50; end
                2'b11:begin hand_bill=100; end
             default :begin end
             endcase
             pay_hand_bill = pay_hand_bill + hand_bill;
        end
     //状态 0+状态 1
        else
        begin
            case(bill_state)
                2'b00:begin auto_bill=10; end
```

```
                2'b01:begin auto_bill=20; end
                2'b10:begin auto_bill=50; end
                2'b11:begin auto_bill=100; end
            default :begin end
            endcase
            pay_auto_bill = pay_auto_bill + auto_bill;
        end
    end

endmodule
```

附录 2、仿真程序代码