**郑州轻工业大学**

**结课报告**

课程名称：FPGA系统设计（硬件设计方向）

报告题目：基于FPGA的交通灯控制器的程序设计

姓 名： 王含越

学 号： 542007010214

专业班级： 计算机20-2

学 院：计算机与通信工程学院

指导教师： 耿 鑫

成 绩：

日 期： 2022年 6 月 2日

目录

[**摘要** 1](#_Toc104193994)

[**第一章 引言** 1](#_Toc104193995)

[**1.1设计背景** 1](#_Toc104193996)

[**1.2设计意义与愿景** 1](#_Toc104193997)

[**第二章 程序设计部分** 2](#_Toc104193998)

[**2.1 软件环境介绍** 2](#_Toc104193999)

[**2.2程序设计思路** 2](#_Toc104194000)

[**2.3模块说明** 3](#_Toc104194001)

[**2.4系统设计原理** 4](#_Toc104194002)

[**第三章 代码设计** 6](#_Toc104194003)

[3.1 交通灯时序模块 6](#_Toc104194004)

[3.2数码管显示模块 10](#_Toc104194005)

[3.3按键延迟模块 15](#_Toc104194006)

[3.4顶层模块 16](#_Toc104194007)

[3.5仿真图 18](#_Toc104194008)

[**第四章 结语** 19](#_Toc104194009)

[**4.1心得体会** 19](#_Toc104194010)

**摘要**

通过FGPA程序控制，来简单实现自动售货机的模拟运行，使得其可以完成自动售货机的基本功能，如自动出货，自动找零等功能，同时为了界面的美观，还相应设计了灯光指引效果，以期待在实际应用中达到更加优秀的视觉效果和使用体验。

**第一章 引言**

**1.1设计背景**

随着科技的进步，生活水平的不断提高，人力成本逐年增加，于是自动售饮料机便应用而生，它将人们从简单的劳动中释出来，让人们有时间去做更有意义的事情。自动售货机利用了FGPA控制技术与计算机相结合，FPGA技术的实用性、程序输入简化、操作简单等优点，其具有较高的稳定性和准确性。自动售饮料机使售卖效率大大提升，给人类社会带来了很大的进步。本文主要描述自动售货机使用FGPA技术的具体实现以及应用。

FPGA 是英文 Field Programmable Gate Array 的缩写，即现场可编程门阵列，它是在 PAL、可编程逻辑器件与 FPGA ·3· GAL、EPLD 等可编程器件的基础上进一步发展的产物。它是作为专用集成电路（ASIC）领域中 的一种半定制电路而出现的，既解决了定制电路的不足，又克服了原有可编程器件门电路数有 限的缺点。

**1.2设计意义与愿景**

自动售饮料机是在无人的情况下完成饮料的销售。它 是集机、电、光一体化的自动商业化设备。当顾客在投币口进行投币后，自动售饮料机会根据货币的数额来判断是否能够购买此饮料，并作出相应的反应。在许多发达城市以及发 展中城市的商业街、大型娱乐场所等各种消费场所已经普遍 存在自动售饮料机，小到学校、小区等地方也存在自动售饮 料机。随着自动售饮料机零售业的不断壮大，一定程度上反 映了劳动密集型的产业构造逐渐地向技术密集型社会转变。 从广义上讲，自动售饮料机就是投币便可产生消费的一种机 械; 从狭义上讲，就是无人化、自动化销售的机械; 从供给的条件上来讲，自动售饮料机的出现很好地解决了人力资源短 缺的问题，同时满足了销售者的要求，节约了人力成本。体 积小、投资少，吸引着购买人的欲望，一定程度上刺激了消费 的产生。

**第二章 程序设计部分**

**2.1 软件环境介绍**

使用**Quartus II软件:**

Quartus II 是Altera公司推出的综合性CPLD/FPGA开发软件，软件支持原理图、VHDL、VerilogHDL以及AHDL（Altera Hardware 支持Description Language）等多种设计输入形式，内嵌自有的综合器以及仿真器，可以完成从设计输入到硬件配置的完整PLD设计流程。

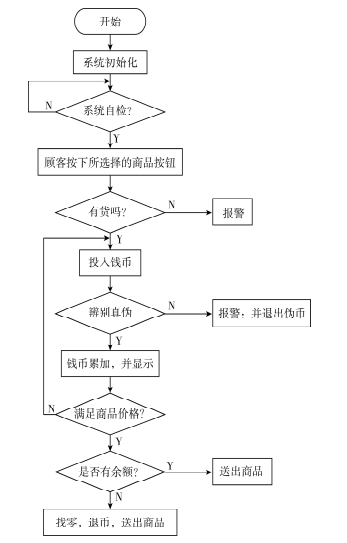
Quartus II可以在Windows、Linux以及Unix上使用，除了可以使用Tcl脚本完成设计流程外，提供了完善的用户图形界面设计方式。具有运行速度快，界面统一，功能集中，易学易用等特点。

Quartus II支持Altera的IP核，包含了LPM/MegaFunction宏功能模块库，使用户可以充分利用成熟的模块，简化了设计的复杂性、加快了设计速度。对第三方EDA工具的良好支持也使用户可以在设计流程的各个阶段使用熟悉的第三方EDA工具。



**2.2工作原理**

从投币口送入货币，然后经过传感器采集数据、识 别 器 判 别 人 民 币 的 真 伪 并 判 别 面 值。识 别 器 把 信 息 数 据传给通信模块。通信模块与售货机的主控系统通信， 主控系统显示面值，启动售货机的面板键，显示出哪个 货 道 有 货，哪 个 货 道 已 售 完，并 等 待 顾 客 按 键 选 择 商 品。顾客选择商品后，售货机自动把商品送出，等待顾 客取走。显示余额，如果金额足够多，顾客可以选择找 币或继续买商品；如果款额不足，售货机经过延时予以 退币。退币分 2 种情况：一种是由纸币识别器完成退纸 币，另 一 种 是 由 硬 币 的 通 信 模 块 完 成 退 硬 币。系 统 复 零，完成售货，如图表1所示



图表 1 程序流程

(1)本程序设计调用了IEEE 库，IEEE库是VHDL设计中最为常用的库，它包含有IEEE标准的程序包和其他一些支持工业标准的程序包。本设计采用了STD\_LOGIC\_1164,STD\_LOGIC\_ARITH, STD\_LOGIC\_UNSIGNED程序包。

(2）以关键词ENTITY引导，END traffic结尾的语句部分，称为实体。VHDL的实体描述了电路器件的外部情况及各信号端口的基本性质。定义了各端口信号的数据类型，主要是STD\_LOGIC（标准逻辑位数据类型)、STD\_LOGIC\_VECTOR(标准逻辑矢量数据类型)。这些都满足上面调用的IEEE库中的程序包。

(3)以关键词ARCHITECTURE 引导，END结尾的语句部分，成为结构体。结构体负责描述电路器件的内部逻辑功能或电路结构。本设计的结构体中包含6个进程，分别是分频器进程、计数器进程、状态寄存器进程、数码管驱动进程、七段数码管显示驱动进程。

**2.3模块说明**

该自动售货机设计思路采用top-down思想，先从其应用层进行考虑，再使用硬件具体完成应用层的要求。主要分为5个模块，分别是：消抖模块、计算模块、状态机控制模块、led模块、显示模块；

功能分析，它的输入有：系统时钟、复位信号、[2:0]drink（表示所选货物，拨码开关控制 分别为1元，3.5元，5元）、四种钱币的信号（0.5元，1元，5元，10元 由按键控制）；它的输出有：数码管的位选和段选（sel[3:0],seg[7:0]）和一个led的控制信号；

**2.4系统设计原理**

交通灯控制模块将需要显示的时间数据连接到数码管显示模块，同时将状态信号连接到led灯控制模块，然后数码管显示模块和led灯控制模块驱动交通信号灯外设工作。系统总体框架图如图所示。

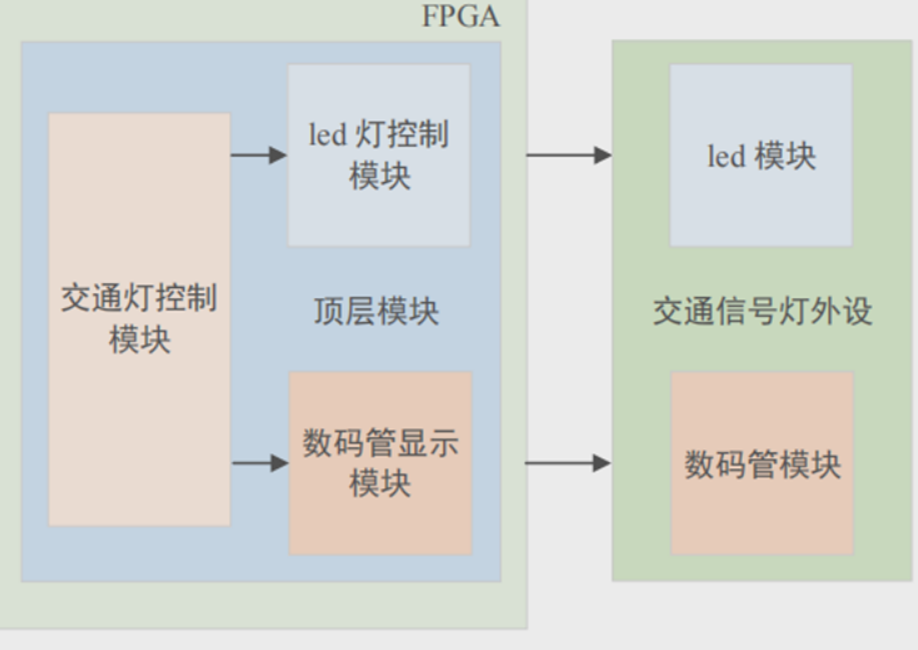


图2 交通信号灯总体框架图

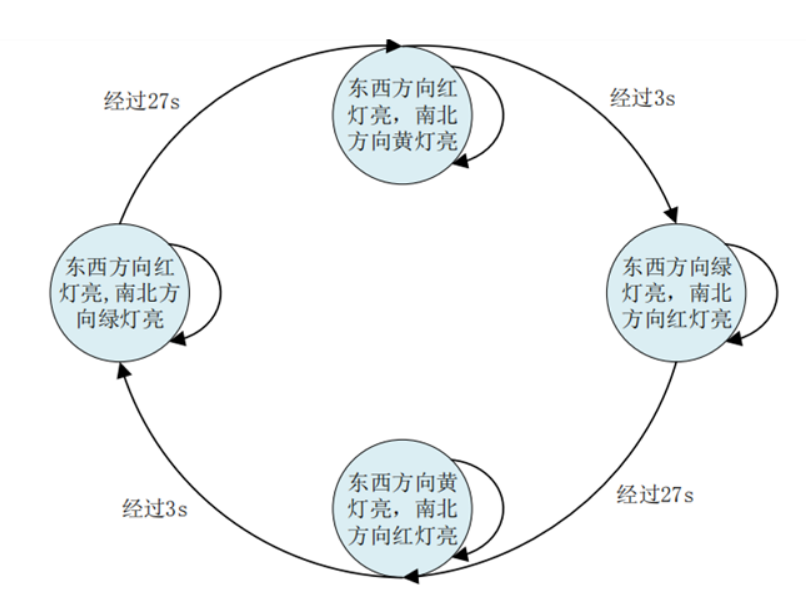
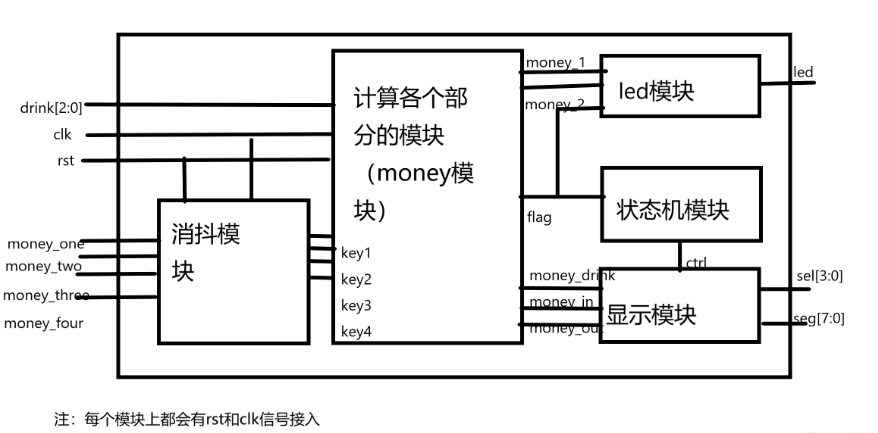


图3 系统软件流程图

**第三章 代码设计**

## 3.1 TOP模块

本模块的实现主要是各个端口的例化，想要清晰的的实现本模块，画图是最简单的方式，所以，这个自动售货机的结构图如图表2所示：



图表 2 结构图

该功能的具体实现代码为：

module top(

input clk, //系统时钟

input rst, //复位信号

input [2:0] drink,//货物

input money\_one, //0.5

input money\_two, //1

input money\_three, //5

input money\_four, //10

output wire [3:0] sel,//数码管位选

output wire [7:0] seg,//数码管段选

output wire led

);

wire [7:0] money\_1;

wire [7:0] money\_2;

wire [2:0] flag;

wire [11:0] money\_in;

wire [11:0] money\_drink;

wire [11:0] money\_out;

wire [2:0] ctrl;

//消抖部分

xiaodou s0(

.clk(clk),

.rst(rst),

.key\_in(money\_one),

.key\_flag(key1)

);

xiaodou s1(

.clk(clk),

.rst(rst),

.key\_in(money\_two),

.key\_flag(key2)

);

xiaodou s2(

.clk(clk),

.rst(rst),

.key\_in(money\_three),

.key\_flag(key3)

);

xiaodou s3(

.clk(clk),

.rst(rst),

.key\_in(money\_four),

.key\_flag(key4)

);

//计算部分

money s4(

.clk(clk),

.rst(rst),

.key1(key1),

.key2(key2),

.key3(key3),

.key4(key4),

.drink(drink),

.money\_1(money\_1),

.money\_2(money\_2),

.flag(flag),

.money\_in(money\_in),

.money\_drink(money\_drink),

.money\_out(money\_out)

);

//状态机

zhaungtai s5(

.clk(clk),

.rst(rst),

.flag(flag),

.ctrl(ctrl)

);

//数码管显示部分

xianshi s6(

.clk(clk),

.rst(rst),

.ctrl(ctrl),

.date1(money\_drink),

.date2(money\_in),

.date3(money\_out),

.sel(sel),

.seg(seg)

);

//led灯控制部分

led s7(

.clk(clk),

.rst(rst),

.flag(flag),

.money\_1(money\_1),

.money\_2(money\_2),

.led(led)

);

endmodule

## 3.2消抖模块

本模块的主要功能是对四个有按键输入的信号进行消抖，并输出一个脉冲信号；

## 3.3计算模块

本模块的主要功能是计算所选的饮料的价格、输入的价格和所要找零的钱数并将其转化为BCD码（方便在数码管上显示）;

本模块的的输入有：系统时钟、复位信号、drink[2:0]、四个消抖后的钱币信号；输出有： money\_1所选货物的价格（二进制）、money\_2/所支付的钱数（二进制）flag状态机状态转化信号、money\_in所支付的钱数（BCD）、 money\_drink所选货物的价格（BCD）、 money\_out 找零的钱数（BCD）。

本模块的实现困难之处是在数据转化为BCD码上，对于本模块我采用了三种实现方法：money\_drink转化为BCD码：

这个信号的实现较为简单、直接通过一个译码器来实现：

相关代码：

module money(

input clk,

input rst,

input key1, //消抖后的信号

input key2, //消抖后的信号

input key3, //消抖后的信号

input key4, //消抖后的信号

input [2:0] drink,//货物

output reg [7:0] money\_1, //所选货物的价格（二进制）

output reg [7:0] money\_2, //所支付的钱数（二进制）

output reg[2:0] flag, //状态机状态转化信号

output reg[11:0] money\_in, //所支付的钱数（BCD）

output reg[11:0] money\_drink, //所选货物的价格（BCD）

output reg[11:0] money\_out //找零的钱数（BCD）

);

reg [7:0] money\_3; //找零的钱数（二进制）

//所负的总钱数并转化为bcd码

always@(posedge clk or negedge rst)begin

if(!rst) begin

money\_in<=12'd0;

money\_2<='d0;

end

else if(money\_in[3:0]>4'b1001)begin

money\_in[3:0]<=money\_in[3:0]-4'b1010;

money\_in[7:4]<=money\_in[7:4]+1'b1;

end

else if(money\_in[7:4]>4'b1001)begin

money\_in[7:4]<=money\_in[3:0]-4'b1010;

money\_in[11:8]<=money\_in[7:0]+1'b1;

end

else if(key1) begin

money\_in[3:0]<=money\_in[3:0]+4'b0101;

money\_2<=money\_2+'d5;

end

else if(key2) begin

money\_in[7:4]<=money\_in[7:4]+4'b0001;

money\_2<=money\_2+'d10;

end

else if(key3) begin

money\_in[7:4]<=money\_in[7:4]+4'b0101;

money\_2<=money\_2+'d50;

end

else if(key4) begin

money\_in[11:8]<=money\_in[11:8]+4'b0001;

money\_2<=money\_2+'d100;

end

else

money\_in<=money\_in;

end

//所选饮料的钱数并将其转化为bcd码

always@(posedge clk or negedge rst) begin

if(rst==0)

money\_1<=0;

else begin

case(drink)

3'd000:begin money\_1<='d0 ;money\_drink<=12'b0000\_0000\_0000; end

3'd001:begin money\_1<='d10;money\_drink<=12'b0000\_0001\_0000; end

3'd010:begin money\_1<='d35;money\_drink<=12'b0000\_0011\_0101; end

3'd011:begin money\_1<='d45;money\_drink<=12'b0000\_0100\_0101; end

3'd100:begin money\_1<='d50;money\_drink<=12'b0000\_0101\_0000; end

3'd101:begin money\_1<='d60;money\_drink<=12'b0000\_0110\_0000; end

3'd110:begin money\_1<='d85;money\_drink<=12'b0000\_1000\_0101; end

3'd111:begin money\_1<='d95;money\_drink<=12'b0000\_1001\_0101; end

endcase

end

end

//计算所要找的钱并将其转化为BCD码

always@(posedge clk or negedge rst) begin

if(rst==0)

money\_3<=0;

else if(money\_2>=money\_1&&flag[2]==0)

money\_3<=money\_2-money\_1;

else

money\_3<=money\_3;

end

always@(posedge clk or negedge rst) begin

if(rst==0)

money\_out<=0;

else if(money\_out[3:0]>4'b1001)begin

money\_out[3:0]<=money\_out[3:0]-4'b1010;

money\_out[7:4]<=money\_out[7:4]+1'b1;

end

else if(money\_out[7:4]>4'b1001)begin

money\_out[7:4]<=money\_out[3:0]-4'b1010;

money\_out[11:8]<=money\_out[7:0]+1'b1;

end

else if(money\_3>='d90&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b1001;

money\_out[3:0]<=money\_3-'d90;

end

else if(money\_3>='d80&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b1000;

money\_out[3:0]<=money\_3-'d80;

end

else if(money\_3>='d70&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0111;

money\_out[3:0]<=money\_3-'d70;

end

else if(money\_3>='d60&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0110;

money\_out[3:0]<=money\_3-'d60;

end

else if(money\_3>='d50&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0101;

money\_out[3:0]<=money\_3-'d50;

end

else if(money\_3>='d40&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0100;

money\_out[3:0]<=money\_3-'d40;

end

else if(money\_3>='d30&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0011;

money\_out[3:0]<=money\_3-'d30;

end

else if(money\_3>='d20&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0010;

money\_out[3:0]<=money\_3-'d20;

end

else if(money\_3>='d10&&flag[2]==0) begin

money\_out[7:4]<=money\_out[7:4]+4'b0001;

money\_out[3:0]<=money\_3-'d10;

end

end

//状态机状态转化信号

always@(posedge clk or negedge rst) begin

if(rst==0)

flag<=0;

else if({key1,key2,key3,key4}>0&&flag[0]<=1)

flag[1]<=1;

else if(money\_3>0&&flag[1]<=1)

flag[2]<=1;

else if(drink>0)

flag[0]<=1;

else

flag<=0;

end

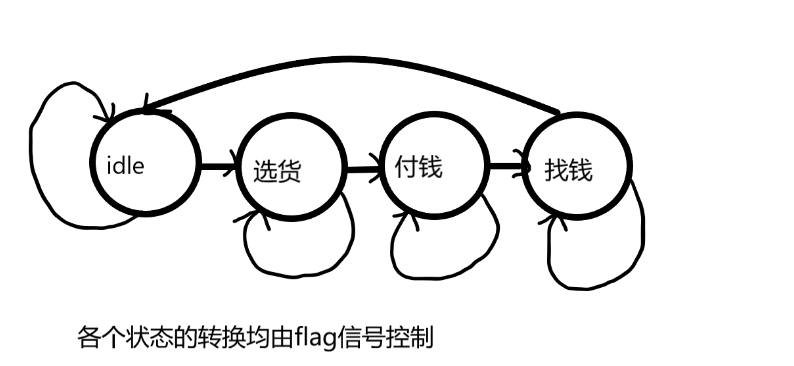
endmodule

## 3.4状态机控制模块

本模块的主要功能是控制各个状态的转换，将会有一下状态：IDLE,选货，付钱，找钱；本模块的的输入有：系统时钟、复位信号、flag信号；输出有：一个控制信号来控制数码管的显示；

状态机的工作为：当有人选货时也就是drink>0时flag[0]=1,进入选货状态；当有钱币投入时，也就是{key1,key2,key3,key4}>0（flag[1]=1）,进入付钱状态；当所支付的钱数大于等于所选货的钱数时（我在这里加了一个并且本状态至少保持1s，原因是如果一次性所支付的钱数大于或者等于货物的价格时，数码管上只会显示一个时钟周期（20ns）的所支付钱数，时间太短，我们肉眼无法察觉），进入找钱状态；当进入找钱状态五秒后回到IDLE；

状态机的状态转移图如图表3所示：



图表 3 状态图

相关代码：

module zhaungtai(

input clk,

input rst,

input [2:0] flag,

output reg [2:0] ctrl

);

localparam

IDLE = 4'b0001,

XUANHUO = 4'b0010,

FUQIAN = 4'b0100,

ZHAOQIAN = 4'b1000;

reg [3:0] c\_stats;

reg [3:0] n\_stats;

reg en\_1;

reg en\_2;

localparam TIME\_1S='d49\_999\_999;

/\* localparam TIME\_1S='d4; \*/

reg [25:0] cnt\_1s;

reg [25:0] cnt\_1s2;

reg [2:0] cnt\_1;

//三段式状态机

always@(posedge clk or negedge rst) begin

if(rst==0)

c\_stats<= IDLE;

else

c\_stats<= n\_stats;

end

always@(\*)begin

case(c\_stats)

IDLE: begin

if(flag[0])

n\_stats<=XUANHUO;

else

n\_stats<=IDLE;

end

XUANHUO: begin

if(flag[1])

n\_stats<=FUQIAN;

else

n\_stats<=XUANHUO;

end

FUQIAN: begin

if(flag[2]&&cnt\_1s2==TIME\_1S)

n\_stats<=ZHAOQIAN;

else

n\_stats<=FUQIAN;

end

ZHAOQIAN: begin

if(cnt\_1=='d5)

n\_stats<=IDLE;

else

n\_stats<=ZHAOQIAN;

end

endcase

end

always@(posedge clk or negedge rst) begin

if(rst==0) begin

ctrl<=0;

en\_1<=0;

en\_2<=0;

end

else if(c\_stats==IDLE) begin

ctrl<=3'b000;

en\_1<=0;

en\_2<=0;

end

else if(c\_stats==ZHAOQIAN) begin

ctrl<=3'b100;

en\_1<=1;

en\_2<=0;

end

else if(c\_stats==FUQIAN) begin

ctrl<=3'b010;

en\_1<=0;

en\_2<=1;

end

else if(c\_stats==XUANHUO) begin

ctrl<=3'b001;

en\_1<=0;

en\_2<=0;

end

end

//

always@(posedge clk or negedge rst) begin

if(rst==0)

cnt\_1s<='d0;

else if(cnt\_1s == TIME\_1S)

cnt\_1s<='d0;

else if(en\_1==1)

cnt\_1s<= cnt\_1s + 1'd1;

end

always@(posedge clk or negedge rst) begin

if(rst==0)

cnt\_1<=0;

else if(cnt\_1=='d4)

cnt\_1<=0;

else if(cnt\_1s==TIME\_1S)

cnt\_1<=cnt\_1+1'd1;

else

cnt\_1<=cnt\_1;

end

always@(posedge clk or negedge rst) begin

if(rst==0)

cnt\_1s2<='d0;

else if(cnt\_1s2 == TIME\_1S)

cnt\_1s2<='d0;

else if(en\_2==1)

cnt\_1s2<= cnt\_1s2 + 1'd1;

end

endmodule

## 3.5 LED模块

本模块的主要功能是实现当输入金额小于饮料价值时，LED灯100ms闪烁；输入金额与饮料价值相等时，LED灯常亮；输入金额大于饮料价值时，显示多出的钱数，LED灯2s闪烁。

本模块的的输入有：系统时钟、复位信号、flag信号、money\_1、money\_2；输出：led信号；

相关代码:

module led(

input clk,

input rst,

input [2:0] flag, //状态转化信号

input [7:0] money\_1,//所选货物的价格（二进制）

input [7:0] money\_2,//所支付的钱数（二进制）

output reg led

);

localparam

TIME\_1S ='d49\_999\_999,

TIME\_100MS ='d4999\_999;

reg [22:0] cnt\_100ms;

reg [25:0] cnt\_1s;

always@(posedge clk or negedge rst)begin

if(rst==0)

cnt\_1s<=0;

else if(cnt\_1s==TIME\_1S)

cnt\_1s<=0;

else

cnt\_1s<=cnt\_1s+1'd1;

end

always@(posedge clk or negedge rst)begin

if(rst==0)

cnt\_100ms<=0;

else if(cnt\_100ms==TIME\_100MS)

cnt\_100ms<=0;

else

cnt\_100ms<=cnt\_100ms+1'd1;

end

always@(posedge clk or negedge rst)begin

if(rst==0)

led<=0;

else if(money\_1>money\_2 && cnt\_100ms==TIME\_100MS)

led<=~led;

else if(money\_1==money\_2 && flag>0)

led<=1;

else if(money\_1<money\_2 && cnt\_1s==TIME\_1S)

led<=~led;

end

endmodule

## 3.6显示模块

本模块的主要功能是将此状态下所要展示的数据显示在数码管上；本模块的的输入有：系统时钟、复位信号、ctrl信号、date1,date2,date3；输出：数码管的段选和位选；

无论是产品的价格和所支付的钱数亦或是所要找零的钱数都会出现浮点数，我们的解决办法是给所有数据乘以10，在数据的第二位让其小数点常亮，也就是当动态扫描到sel[1]时；seg[7]为0；其余位置seg[7]都为1；

相关代码

module xianshi(

input clk,

input rst,

input [2:0] ctrl,

input [11:0] date1,//数据信号

input [11:0] date2,//数据信号

input [11:0] date3,//数据信号

output reg [3:0] sel, // 数码管位选（选择当前要显示的数码管）

output reg [7:0] seg // 数码管段选（选择当前要显示的内容）

);

reg [14:0] cnt\_1; //分频记数的计数器

reg clk\_out; //分频后的时钟信号

/\* reg [3:0] sel\_r; //表示那个数码管亮 \*/

reg [3:0] date\_tmp;

reg [11:0] disp\_data;

localparam TIME='d24999;

/\* localparam TIME='d4; \*/

always@(posedge clk or negedge rst) begin

if(rst==0)

disp\_data<=0;

else

case(ctrl)

3'b001:disp\_data<=date1;

3'b010:disp\_data<=date2;

3'b100:disp\_data<=date3;

default:disp\_data<=0;

endcase

end

//cnt\_1的记数模块

always@(posedge clk or negedge rst) begin

if(rst==0)

cnt\_1 <= 15'd0;

else if(cnt\_1 ==TIME)

cnt\_1 <= 15'd0;

else

cnt\_1 <= cnt\_1+1'd1;

end

//时钟分频模块

always@(posedge clk or negedge rst) begin

if(rst==0)

clk\_out<=1'b0;

else if(cnt\_1== TIME)

clk\_out<=~clk\_out;

else

clk\_out<=clk\_out;

end

//移位寄存器

always@(posedge clk\_out or negedge rst) begin

if(rst==0)

sel<=4'b1110;

else

sel[2:0]<={sel[1:0],sel[2]};

/\* if(rst==0)

sel\_r<=4'b0001;

else

sel\_r<={sel\_r[2:0],sel\_r[3]}; \*/

end

//四选一多路器

always@(\*) begin

case(sel)

4'b1110 : date\_tmp <= disp\_data[3:0];

4'b1101 : date\_tmp <= disp\_data[7:4];

4'b1011 : date\_tmp <= disp\_data[11:8];

default:date\_tmp<=4'b0000;

endcase

end

//字典（译码器）

always@(\*) begin

if(sel[2:0]==3'b101)begin

case(date\_tmp)

4'h0:seg<=8'b01000000;

4'h1:seg<=8'b01111001;

4'h2:seg<=8'b00100100;

4'h3:seg<=8'b00110000;

4'h4:seg<=8'b00011001;

4'h5:seg<=8'b00010010;

4'h6:seg<=8'b00000010;

4'h7:seg<=8'b01111000;

4'h8:seg<=8'b00000000;

4'h9:seg<=8'b00010000;

endcase

end

else begin

case(date\_tmp)

4'h0:seg<=8'b11000000;

4'h1:seg<=8'b11111001;

4'h2:seg<=8'b10100100;

4'h3:seg<=8'b10110000;

4'h4:seg<=8'b10011001;

4'h5:seg<=8'b10010010;

4'h6:seg<=8'b10000010;

4'h7:seg<=8'b11111000;

4'h8:seg<=8'b10000000;

4'h9:seg<=8'b10010000;

endcase

end

end

endmodule

EDA是电子设计自动化 (Elechonics Des5p AM• toM6On)的缩写。由于

它是一门刚刚发展起来的新技术，涉及面广，内容丰富，理解各异，所以日前尚无一个确切的定义。但从印DA技术的几个主要方面的内容来看，可以理解为：EDA技术是以大规模可编程逻辑器件为设计裁体，以便件描述语言为系统逻辑描述的主要表达方式，以计算机、大规模可编程逻辑器件的开发软件及实验开发系统为设计工具，道过有关的开发软件，自动完成用软件的方式设计电子系统到硬件系统的一门新技术。可以实现逻辑编译、逻辑化简、逻辑分割、逻辑综合及优化，逻辑布局布线、逻辑仿真完成对于特定口标芯片的适配编译、逻辑映射、编程下裁等工作，最终形成集成电子系统或专用集成芯片。EDA技术是伴随着计算机、集成电路、屯子系统的设计发展起来的，全今已有30多年的历程。大致可以分为三个发展阶段。3.5仿真图

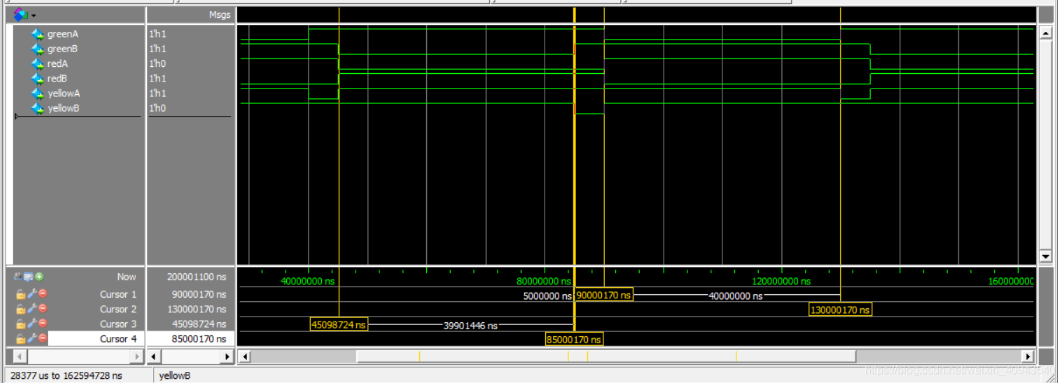


图4 绿灯仿真图

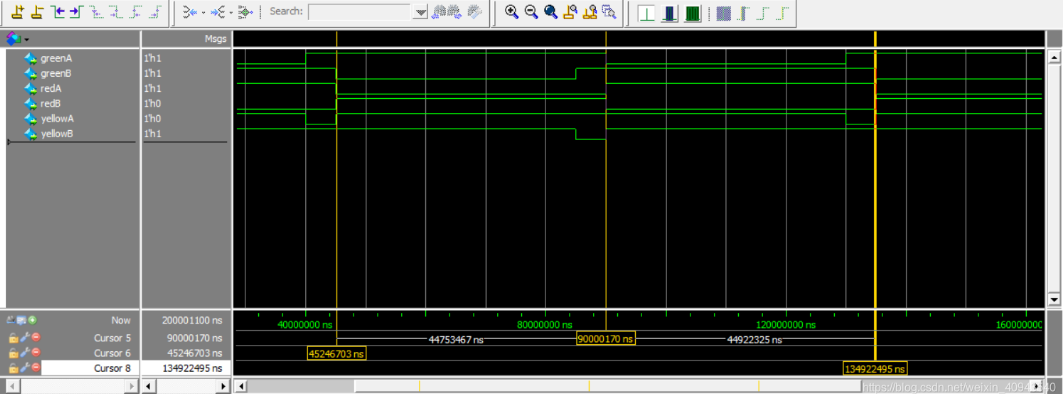


图5 红灯仿真图

**第四章 结语**

**4.1心得体会**

交通灯的设计心得体会总结通过此次课程设计，使我更加扎实的掌握了有关电子线路方面的知识，在设计过程中虽然遇到了一些问题，但经过一次又一次的思考，一遍又一遍的检查终于找出了原因所在，也暴露出了前期我在这方面的知识欠缺和经验不足。从拿到课程设计任务书后经过这么多天的设计，完成了一个具备基本功能的交通灯控制器，它是由FPGA开发板和外部led、数码管显示电路组成。实现了东西、南北两个方向的信号灯交替工作且两个方向通行时间不同，由两位共阳数码管倒计时显示通行时间，红黄绿三种led灯提示通行状态以及遇到紧急情况时可人为将两个方向都设置为红灯的功能。

课程设计完全交给我们自由发挥，充分挖掘了我们的独立思考、探索能力，经过了查找资料、方案设计、方案论证、程序编写、电路设计、硬件连接以及调试等流程，饱满的工作量，完整的经历了交通灯控制器的设计过程，理解了交通灯的工作原理以及FPGA硬件控制原理。这些都可以运用到我们以后的学习及工作中。