郑州轻工业大学

**实验报告**

**课程名称：** 交通灯控制

**姓 名：** 原彬贺

**院 （系）：** 计算机与通信工程学院

**专业班级：** 计算机科学与技术2002班

**学 号：** 542001020223

**指导教师：** 耿鑫

**成 绩：**

**时间：** 2021 **年** 4 **月** 27 **日**

**一、任务**

完成基于Verilog的十字路口交通灯控制器的设计。

**二、要求**

Verilog语言编写程序，实现对十字路口红黄绿交通灯功能，具有清零和复位功能。

**三、实验内容**

1、掌握十字路口交通灯控制的功能，理解Verilog语言多always过程语句编程的方法；

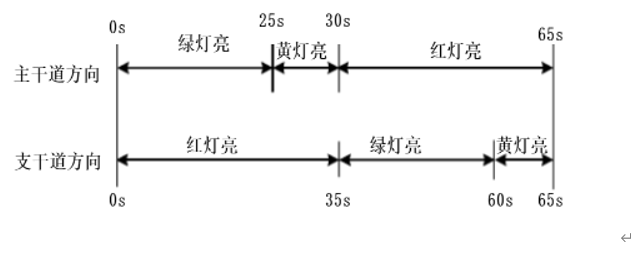
2、使用Verilog语言编程，实现简单的交通灯控制功能；

3、编写testbench文件，用Multisim仿真，输出波形图，并验证实验结果。

**四、实验过程及结果**

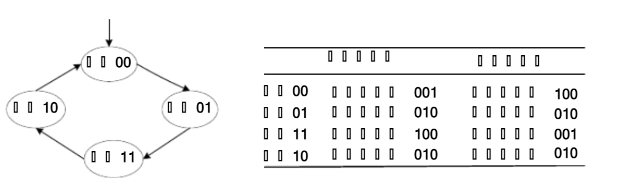
交通等控制的原理

本次实验是对十字路口，进行南北和东西直行情况下交通灯控制。设定东西方向为主干道方向，根据交通灯的亮的规则，在初始状态下四个方向的都为红灯亮启，进入正常工作状态后，当主干道上绿灯亮时，支干道上红灯亮，持续35S后，主干道和支干道上的黄灯都亮启，持续5S后，主干道上红灯亮启，支干道上绿灯亮启持续25S,之后主干道和支干道上的黄灯都亮启5s，一个循环完成。循环往复的直行这个过程。



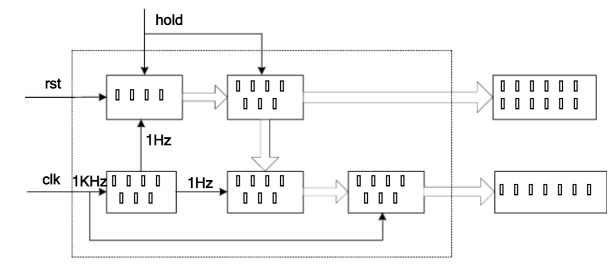
代码设计

根据设计原理，交通灯控制的关键是各个状态之间的转换和进行适当的时间延时，根据状态机的设计规范，本次设计了三个状态之间的循环转化，其真值表及状态转化图如下所示:



说明:该状态图为交通灯在正常情况下的状态转化图，进入控制后，状态0时主干道绿灯及支干道红灯亮起，进入状态01后两路黄灯亮起，状态11时主干道红灯及支干道绿灯亮起。进入10状态两路黄灯亮起。结束一一个循环，从00状态重新开始循环。

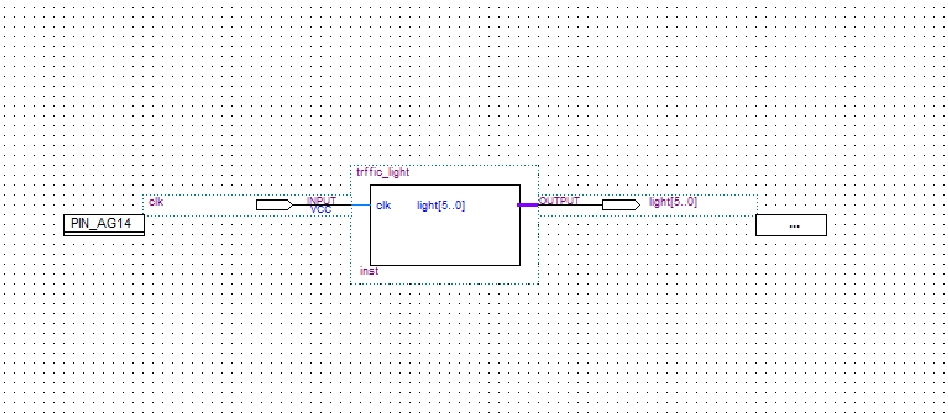
为实现控制与显示的功能，需要设计交通灯点亮顺序控制程序，倒数计时程序，七段数码管显示程序，数码管显示扫描程序，其系统结构图如下所示:

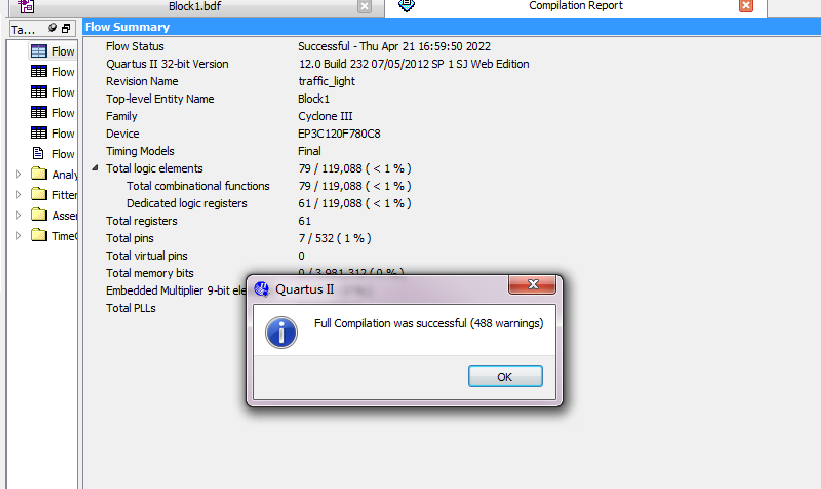


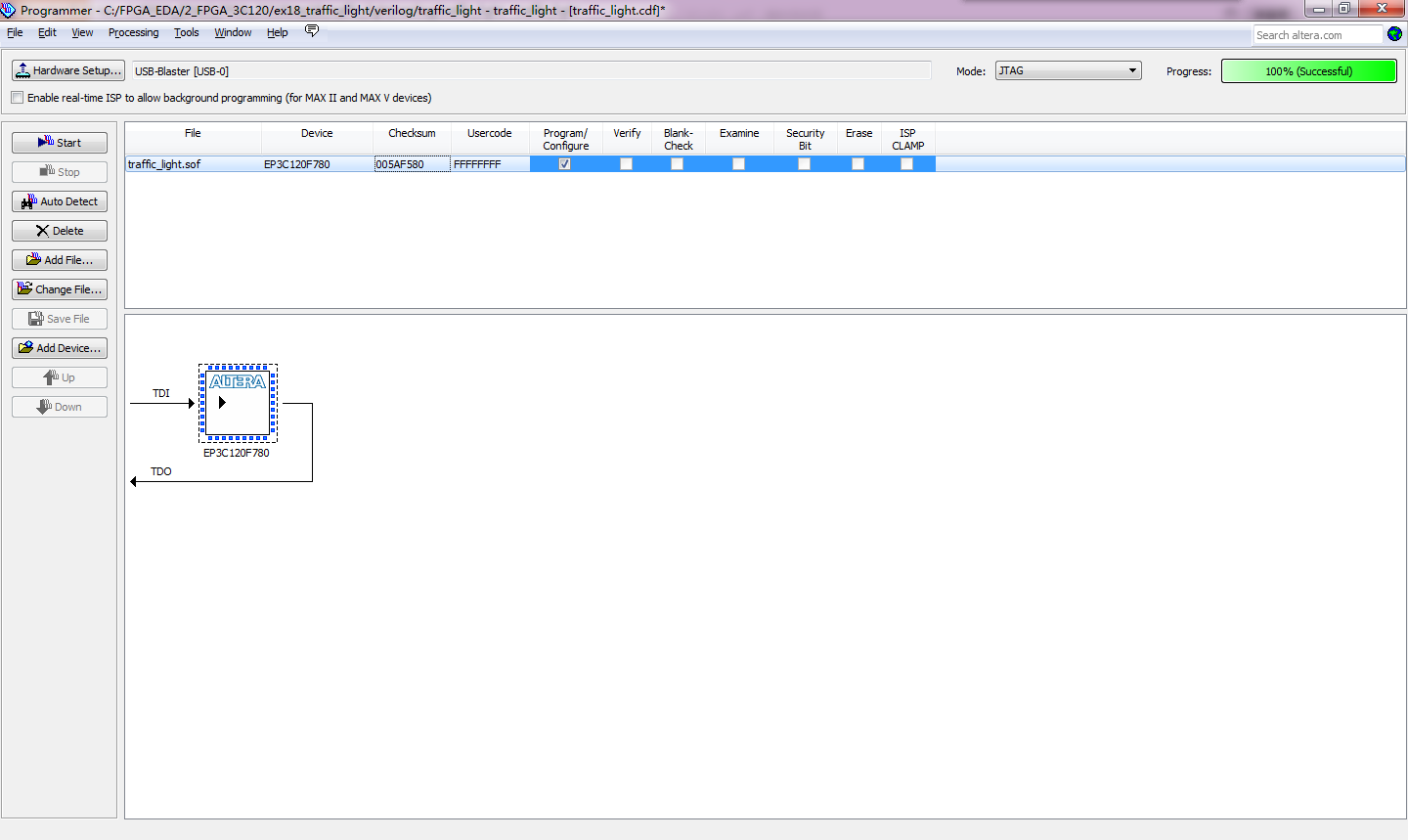
其中rst为复位信号，clk为时钟信号，hold为特殊情况控制信号,输入hold时两个方向红灯无条件亮起。

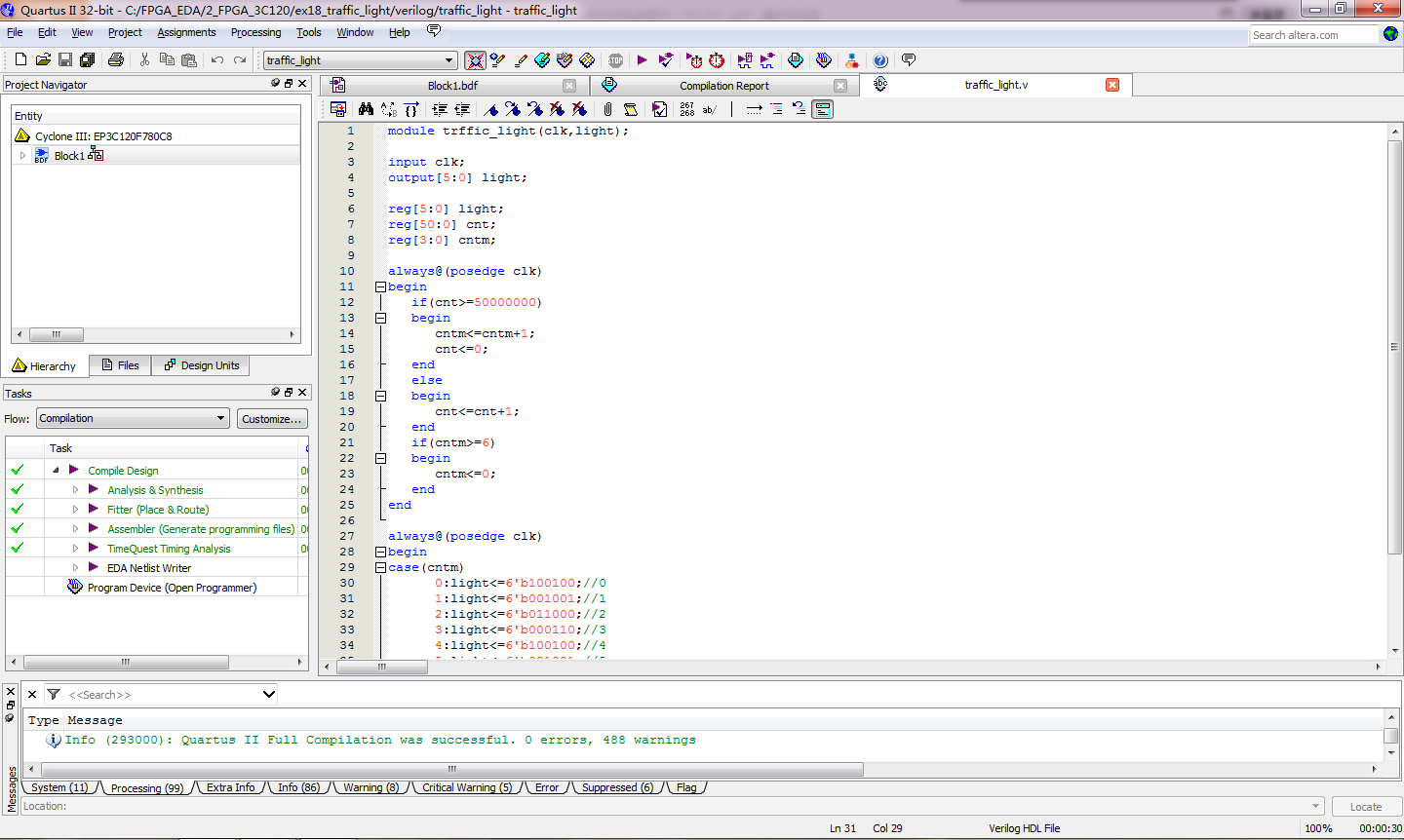
在QUARTUSII环境下，进行Verilog程序设计，完成交通灯控制器设计，观察生成的RTL流图。

按照相应步骤进行运行，然后进行灯的控制。





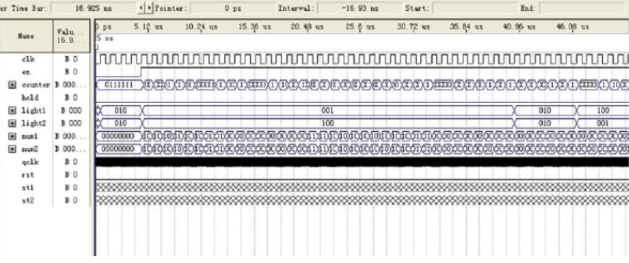




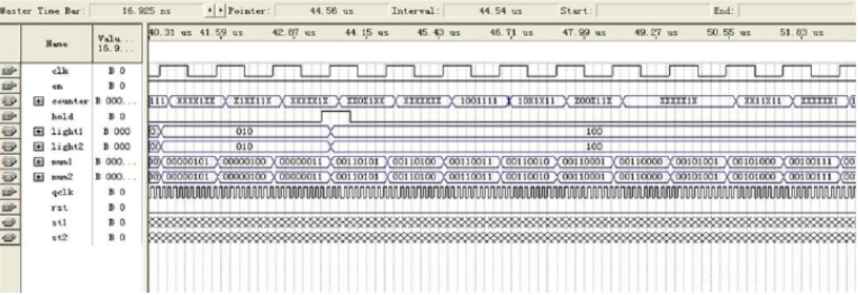
五、实验结果分析（包括编译、波形仿真等）

在QuartursII软件下创建工程，新建编辑设计文件，将程序输入，整体编译后，新建波形仿真文件。设置仿真时间，时钟周期，输入输出端口，进行波形仿真。具体仿真波形图及说明如下所示:

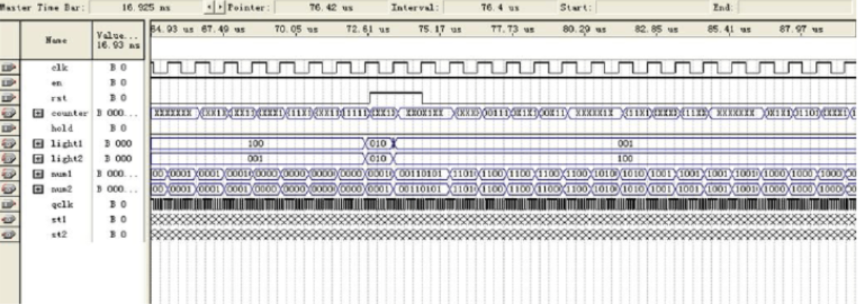
1、正常工作时波形仿真图

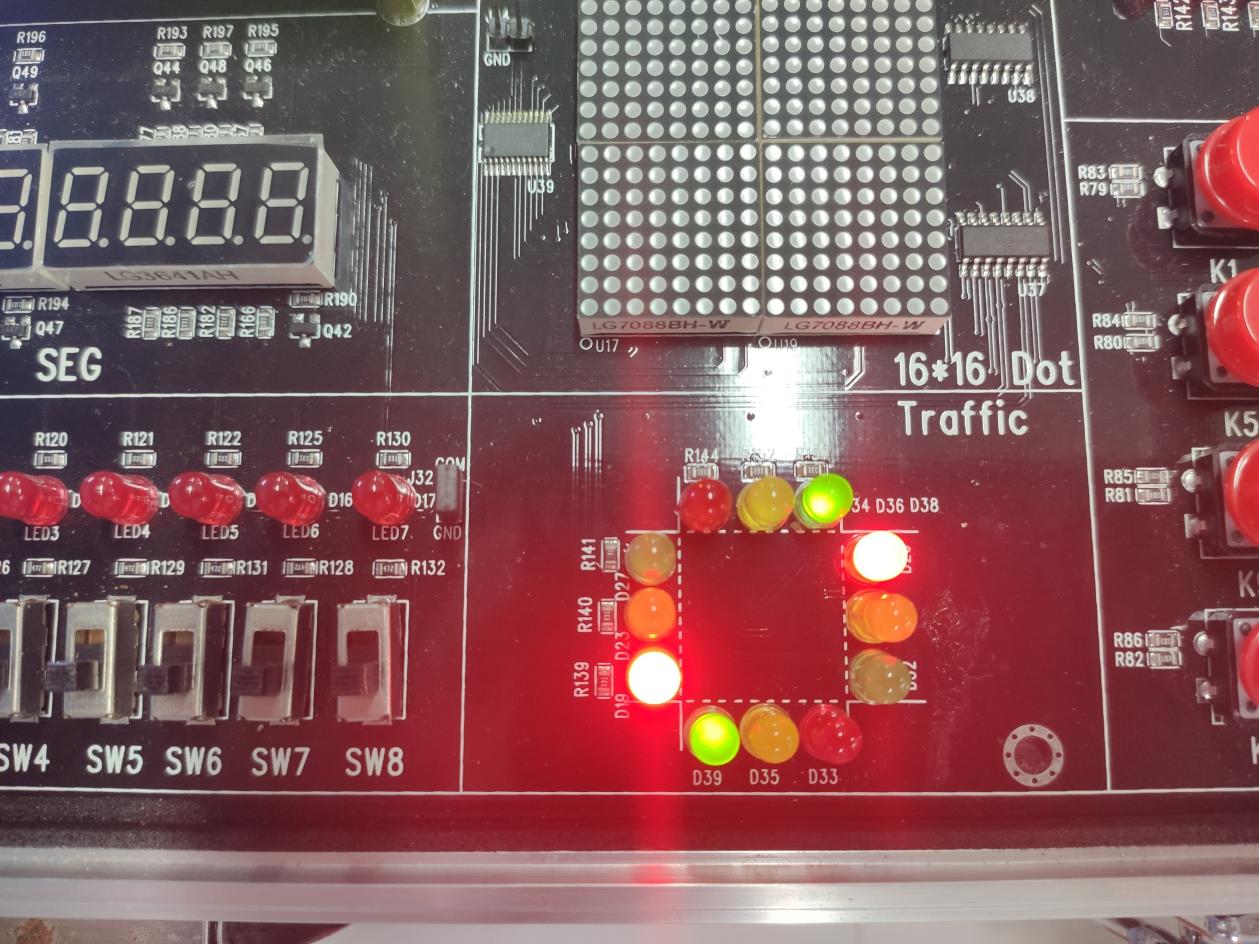
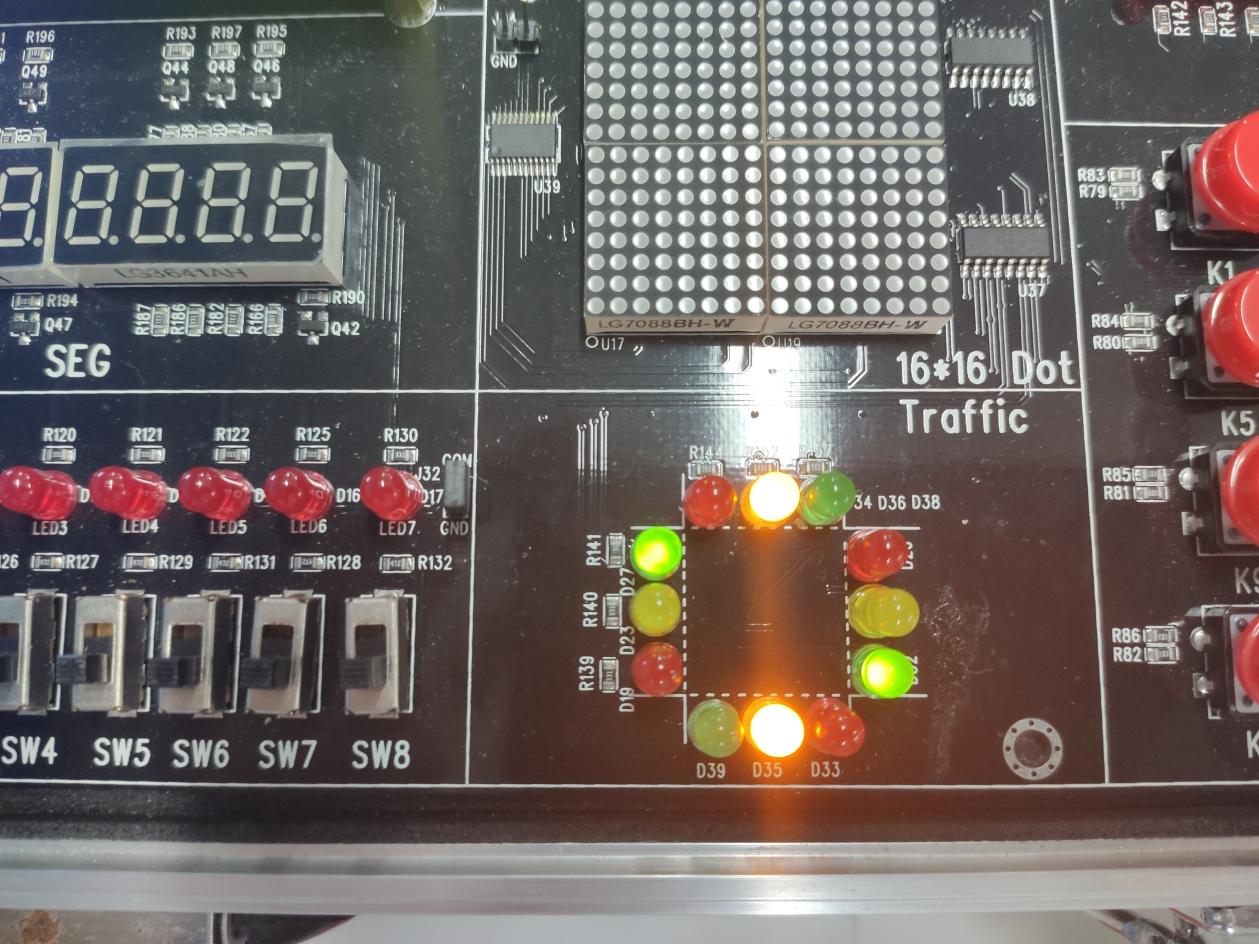


1. 特殊情况仿真波形



3、复位情况仿真波形





**六、实验总结和心得**

通过本次实验，制作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，通过对这次实验的具体实行，使我了解到在设计中自顶向下的方法，先设计其应用层顶层，有全局观，然后再逐步完善细节。这次实验不仅对我的技能提升有很大的帮助，还使我的思维有极大的拓展，在今后的开发路上走的更远。

代码样例：

module traffic (en,clk,qclk,rst,rst1,hold,num1,num2,light1,light2,counter,st1,st2);

Input en,clk,qclk,rst,hold,rst1;

output st1,st2;

output[7:0] num1,num2;

output[6:0]counter;

output[2:0] light1,light2;

reg tim1,tim2,st1,st2;

reg[1:0]state1,state2,ste;

reg[2:0]light1,light2;

reg[3:0]num;

reg[6:0]counter;

reg[7:0] num1,num2;

reg[7:0] red1,red2.,green1,green2,yellow1,yellow2;

always @(en)

if(!en)

begin//设置计数初信

green1<=8‘b00110101;

red1<=8'b00100101;

yellow1<=8'b00100101;

green2<=8'b00100101;

red2<=8'b00110101;

yellow2<=8'b00000101;

end

always @(posedge clk)

begin

if(rst) //复位与特殊情况控制

begin

light1<=3'b001;

num1<=green1;

end

else if(hold)

begin

light1<=3'b100;

num1<=green1;

end

else if(en)

begin //使能有效开始控制计数

if(!tim1)//开始控制

begin /主干道交通灯点高控制

tim1<=1;

case(state1)

2'b00：begin num1<=green1;light1<=3'b001;state1<=2'b01;end

2'b01:begin num1<=yellow1;light1<=3'b010;state1<=2'b11;end

2'b11:begin num1<=red1;light1<=3'b100;state1<=2'b10;end

2'b10:begin num1<=yellow1;light1<=3'b010;state1<=2'bO0;end

default:light1<=3'b100;

endcase

end

else

begin //用数计时

if(num1>0)

if(num1[3:0]==0]

begin

num1[3:0]<=4'b1001;

num1[7:4]<=num1[7:4]-1;

end

else num1[3:0]<=num1[3:0]-1;

if(num1==1) tim1<=0;

end

end

else

begin

light1<=3'b010;

num1=2'b00;

tim1<=0;

end

end

always @(posedge clk)

begin

if(rst)//复位与特殊情况控制begin

light2<=3'b100;

num2<=red2;

end

else if(hold)

begin

light2<=3'b100;

num2<=red2;

end

else if(en)

begin

if(!tim2)

begin

tim2<=1;

case(state1)

2'b00:begin num2<=red2;light2<=3'b100;state2<=2'b01;end

2'b01:begin num2<=yellow1;light2<=3'b010;state2<=2'b11;end

2'b11:begin num2<=green2;light2<=3'b001:state2<=2’b10;end

2'b10:begin num2<=yellow2;light2<=3'b010;state2<=2'b00;end

default:light2<=3'b100;

endcase

end

else

begin //倒数计时`

if(num2>0)

if(num2[3:0]==0)

begin

num2[3:0]<=4'b1001;

num2[7:4]<=num2[7:4]-1;

end

else num2[3:0]<=num2[3:0]-1;

if(num2==1) tim2<=0;

end

end

else

begin

tim2<=0;

state2<=2'b00;

light2<=3'b010;

end

end

always @(posedge qclk)

begin //数码管扫描

if(rst1)

begin

st1=O;

st2=0;

end

else

begin

case({st2,st1})

2'b00:begin num<=num1[3:0];{st2,st1}<=2'b01;end

2'b01:begin num<=num1[7:4]:{st2,st1}<=2'b10;end

2'b10:begin num<=num2[3:0];{st2,st1}<=2'b11;end

2'b11:begin num<=num2[7:4];{st2,st1}<=2'b00;end

endcase

end

end

always @(posedge qclk)

begin //数码管译码显示

case(num)

4'b0000:counter<=7'b0111111;//0

4'b0001:counter<=7'b0000110;//1

4'b0010:counter<=7'b1011011;//2

4'b0011:counter<=7'b1001111;//3

4'b0100:counter<=7'b1100110;//4

4'b0101:counter<=7'b1101101//5

4'b0110:counter<=7'b1111101;//6

4'b0111:counter<=7'b0000111;//7

4'b1000:counter<=7'b1111111;//8

4'b1001:counter<=7'b1101111;//9

default:counter<=7'b0111111;//0

endcase

end

endmodule