**数字电路实验报告**

**组号:22**

**题目：汽车尾灯控制**

**小组分工**

**电路设计和仿真：**

**FPGA编程和创新：**

**实验报告撰写：**

**一．设计题目及要求**

设计一个汽车尾灯控制电路，用LED指示灯模拟汽车双侧尾灯，并用开关作为转弯信号控制器。左转弯控制器开启指示汽车左侧尾灯点亮（闪烁）；右转弯控制器开启指示汽车右侧尾灯点亮（闪烁）；如果两个开关同时开启，双侧尾灯应急闪烁。

**二．电路总体设计方案**

(1).设计思路

此设计任务是设计一个汽车尾灯控制电路，在芯片型号均不限的情况下实现对汽车尾灯显示状态的控制。电路大致由控制电路、计数器、显示电路等模块组成。在左右各三个指示灯的基础上，实现汽车运行的基本四种情况，即 ：正常行驶，左、右转弯，刹车。用一片74LS139译码器来产生四种控制状态；用两片74LS160计数器来分别控制3个左转LED灯和3个右转LED灯，通过开关控制左转右转。

(2).设计方案

左转尾灯L1L2L3,右转尾灯R1R2R3

1. 当输入为00时，汽车正常运行，6个LED灯熄灭, 所以通过74139译码后输出为为“1111”不做任何处理。
2. 当输入为01时，汽车左转弯，L3亮-L3L2亮-L3L2L1亮-全灭- L3亮的循环。。

具体实现是通过74LS139对“01”译码为“1011”然后通过译码后地低电平Y1通过一个非门控制计数器74LS160地ENT和ENP端开始计数,从而控制灯地点亮方式,计数范围为0000（全灭）→0001（L3点亮）→0010（L3L2点亮）→0011（L3L2L1点亮）→0100（异步清零）→0000……循环计数就实现了循环点亮地这个过程。

1. 当输入10时，汽车右转弯，R1亮-R1R2亮-R1R2R3亮-全灭-R1亮依次循环。

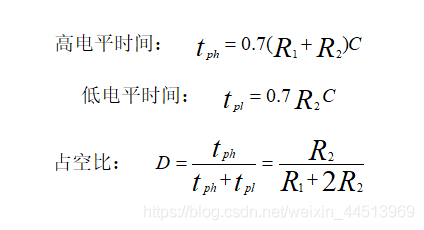
具体实现是通过74LS139对“010”译码为“1101” 然后通过译码后地低电平Y2通过一个非门控制计数器74LS160地ENT和ENP端开始计数，从而控制灯地点亮方式，计数范围为0000（全灭）→0001（R1点亮）→0010（R1R2点亮）→0011（R1R2R3点亮）→0100（异步清零）→0000……循环计数就实现了循环点亮的这个过程。

1. 当输入11时，汽车刹车，所有LED灯一起闪烁，亮灭亮灭循环。

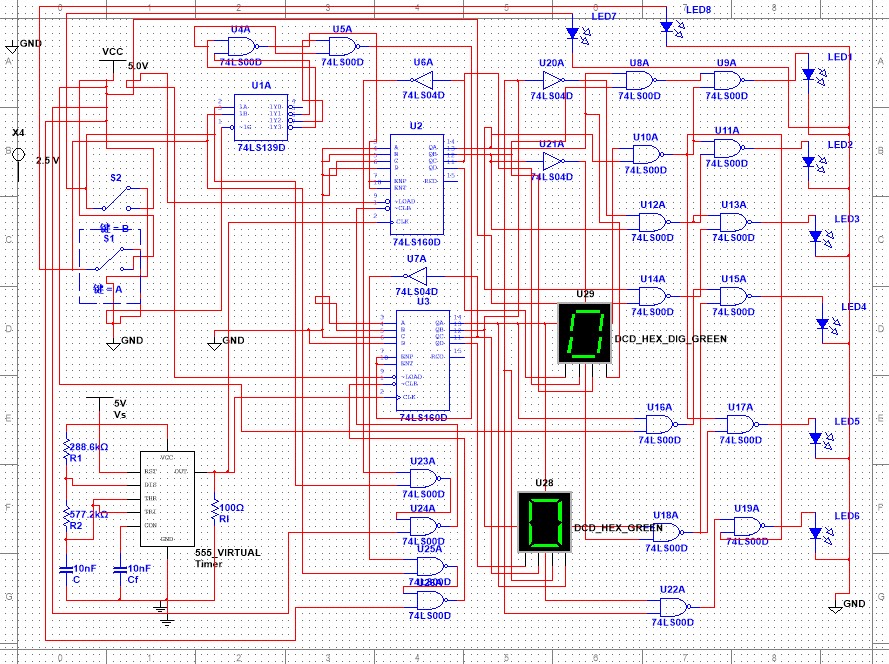
通过列出每个LED灯与74LS160的输出之间的真值表即可得出每个LED灯的表达式。

(3)．时钟信号源电路

由于汽车灯是地点亮是给人地不同地信息及该车将要发生地动作，所以汽车地灯在闪烁地时候不能超过一定地频率但是频率也不能太小，所以我们在设计地时候是采用地555定时器设计地一个脉冲产生源，占空比约为60%，通过实际发现考虑到门电路的延时性，频率为100HZ时反应较快，这样就能让人很清楚地明白该汽车地动作以采取相应地动作，从而避免交通事故地发生。



于是可取出各部件值，从而设计出如下电路。



**三.功能模块介绍.**

LED1-L1,LED2-L2, LED3-L3, LED4-R1,LED5-R2,LED6-R3

1.S1S2打开时，汽车正常行驶，所有LED灯熄灭。

2.当左转开关S2闭合时，上面三个左转灯按照L3亮-L3L2亮-L3L2L1亮-全灭-L3亮依次循环点亮。同时LED7亮表示左转，数码管U29开始随着循环开始计数循环。

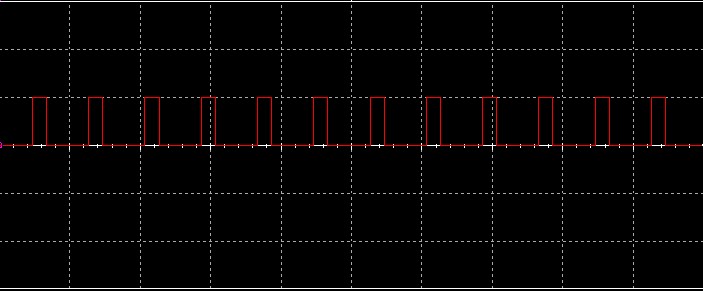
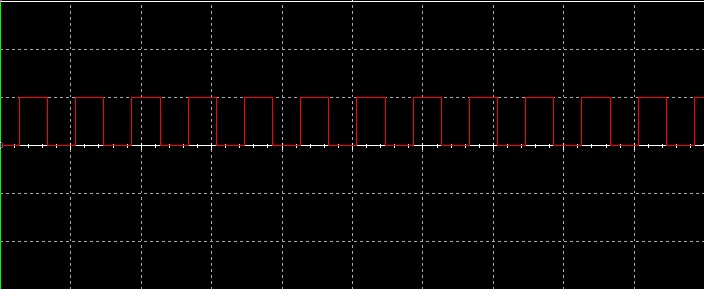
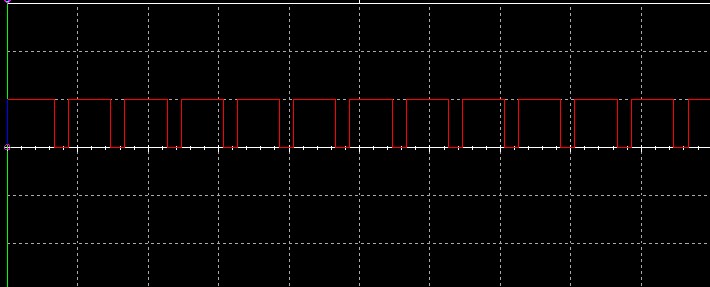
3.当右转开关S1闭合时，下面三个右转灯按照R1亮-R1R2亮-R1R2R3亮-全灭-R1亮依次循环点亮。同时LED8亮表示右转，数码管U28开始随着循环开始计数循环。

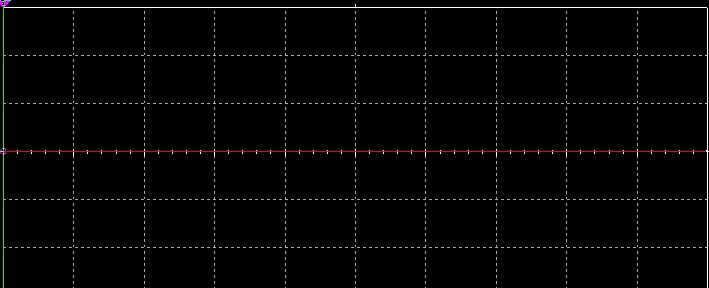
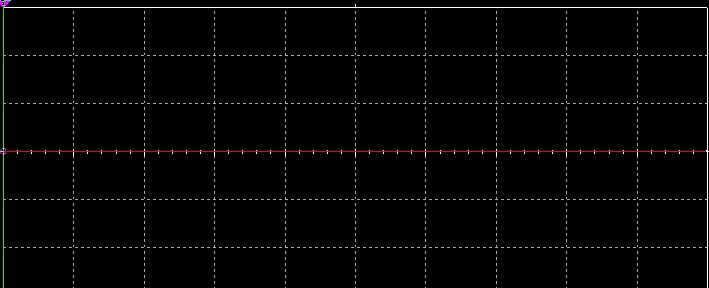
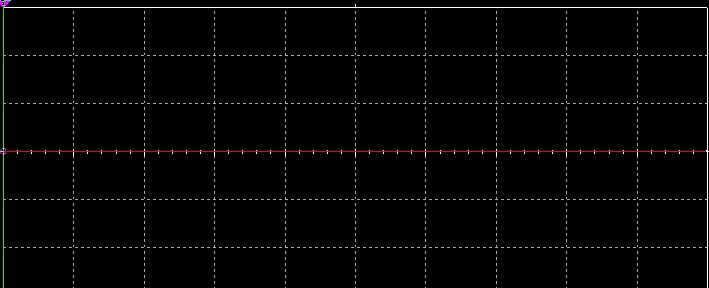
4.当S1S2同时闭合时，表示刹车，6个转向灯同时亮灭循环。同时两个数码管计数循环。

五．仿真功能测试。

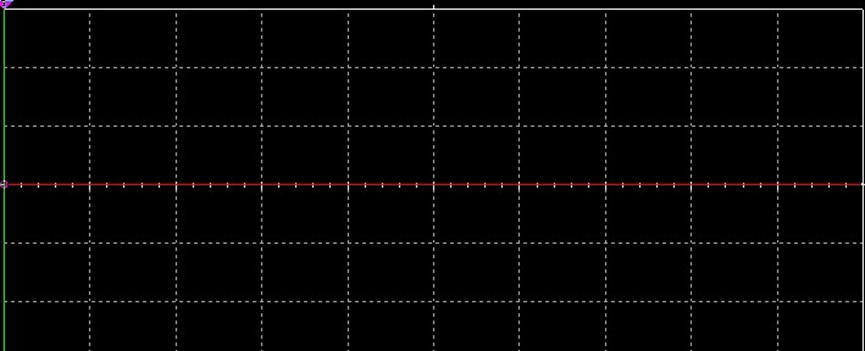
以下6个LED灯顺序为L1、L2、L3、R1、R2、R3

1.左转功能仿真

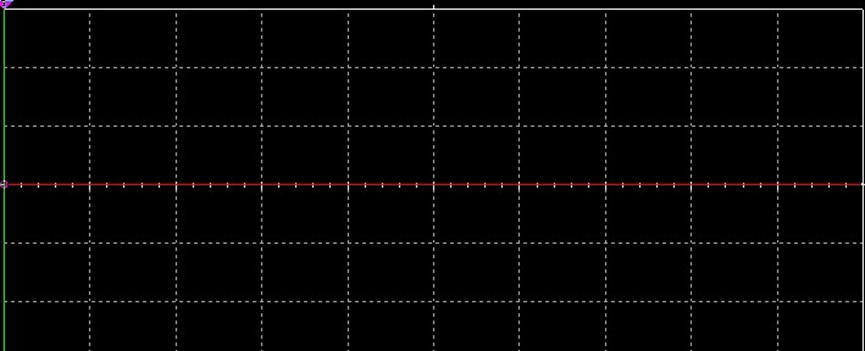


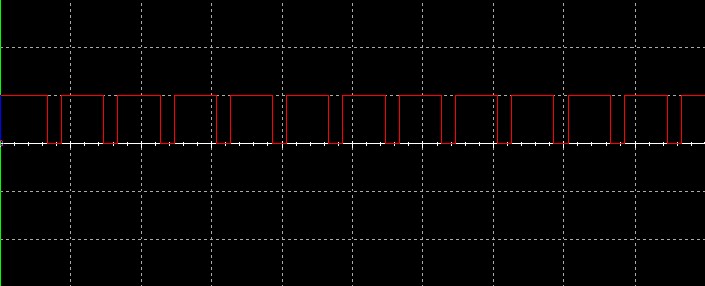


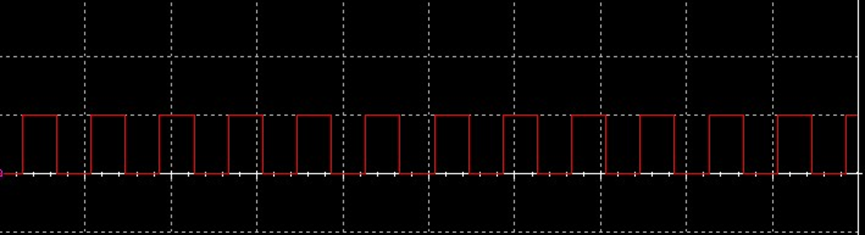
2.右转功能仿真

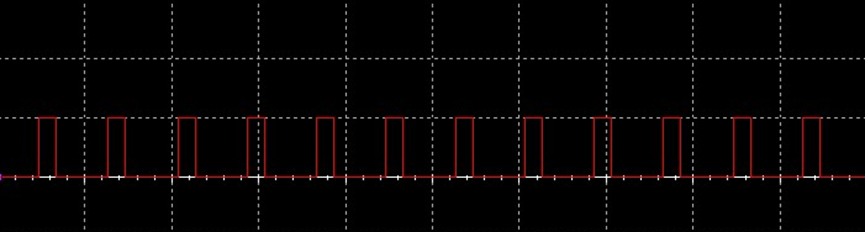




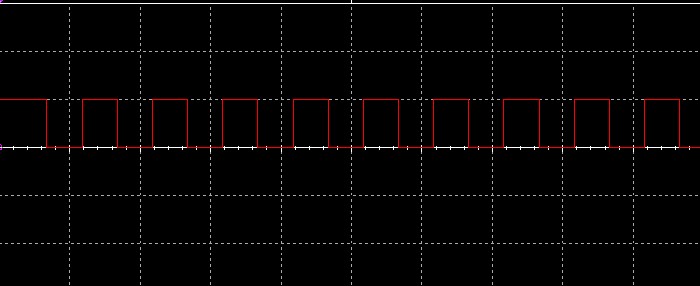
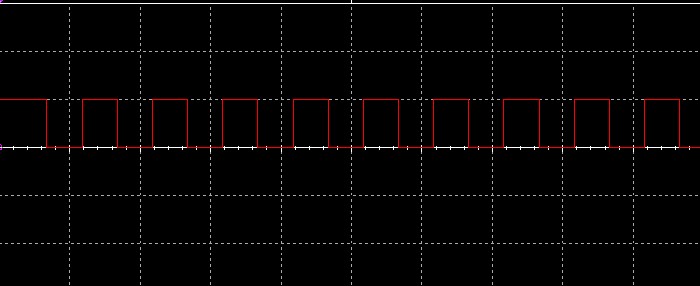
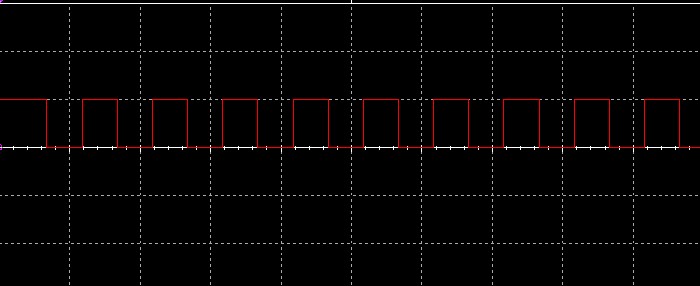
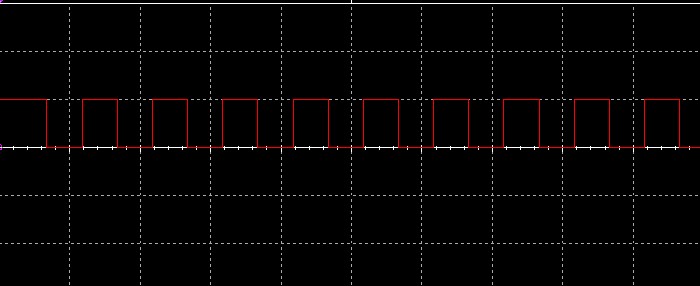
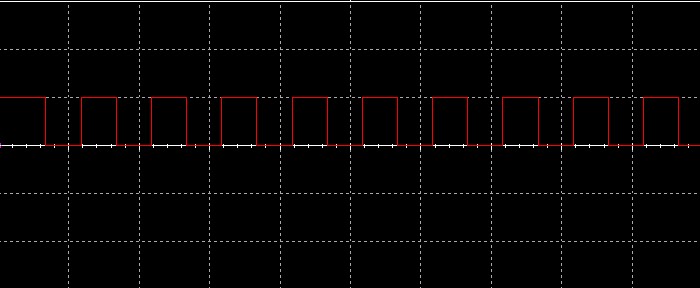
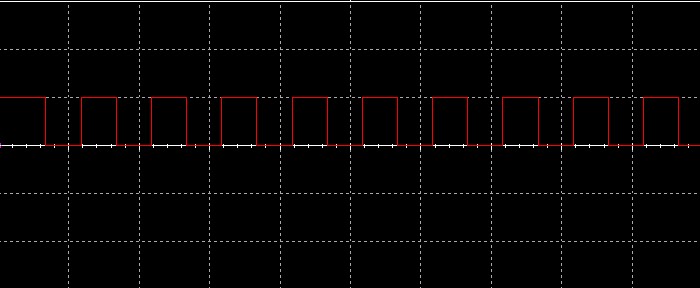








3.刹车功能仿真



**四．和FPGA程序实现**

**1.基本功能**：

在没有操作的情况下，fpga无任何响应，拨下左转开关，左指示灯亮，拨下右转开关，右指示灯亮，左右同时拨下，指示灯都亮

2．**Fpga扩展创新**

A．流水灯

按下左转开关，左边3个led流水向左点亮；按下右转开关，右边3个led向右流水点亮；两个同时按下，6个灯同时闪烁

B．rgb呼吸灯

按下左转开关，深蓝色rgb灯呼吸闪烁；按下右转开关，绿色rgb灯呼吸闪烁；同时按下，浅蓝色rgb呼吸闪烁

C.数码管

左转，数码管成3字形状，开口向左；右转，数码管成E字形状，开口向右；两个同时按下，数码管成三字形状

**3.代码分析：**

代码用到了译码器，分频器，调节占空比，状态机，计数器

1. 本人先从流水灯做起：

通过代码来调整led的状态实现，流水的效果，IDLE表示都熄灭，L1到R3分别表示6个led流水灯，通过状态转换，在按键不变的情况下，由当前状态转换到下一个状态，以时钟上升沿作为敏感信号，每次实现下个状态给当前状态赋值，而后以状态和左右开关作为敏感信号，通过判断状态类型，来点亮相应的led灯，从而实现流水的效果

module carlight(RA,RB,RC,LA,LB,LC,clk,rst,LEFT,RIGHT);

input clk,rst;

input LEFT,RIGHT;

output reg LA,LB,LC,RA,RB,RC;

reg[2:0] state,next\_state;

parameter[2:0] IDLE=3'b000,

L1=3'b001,//1

L2=3'b010,//2

L3=3'b011,//3

R1=3'b100,//4

R2=3'b101,//5

R3=3'b110,//6

LR3=3'b111;//7

wire clk1h;

divide #(.WIDTH(32),.N(12000000)) u2 ( //传递参数

.clk(clk),

.rst\_n(rst), //例化的端口信号都连接到定义好的信号

.clkout(clk1h)

);

//时序逻辑

always @(posedge clk1h or negedge rst)

begin

if(~rst)//低电平复位

state<=IDLE;

else

state<=next\_state;

end

//组合逻辑

always@(state or LEFT or RIGHT)

begin

next\_state=3'bxxx; //随机态

case(state)

IDLE: //复位情况

begin

if(LEFT&&(!RIGHT))

next\_state=L1;

else if(RIGHT&&(!LEFT))

next\_state=R1;

else if(LEFT&&RIGHT)

next\_state=LR3;

else

next\_state=IDLE;

end

L1:

begin

if(!(LEFT&&RIGHT))//不是刹车

next\_state=L2;

else

next\_state=LR3;

end

L2:

begin

if(!(LEFT&&RIGHT))

next\_state=L3;

else

next\_state=LR3;

end

L3:

begin

next\_state=IDLE;

end

R1:

begin

if(!(LEFT&&RIGHT))//不是刹车

next\_state=R2;

else

next\_state=LR3;

end

R2:

begin

if(!(LEFT&&RIGHT))

next\_state=R3;

else

next\_state=LR3;

end

R3:

begin

next\_state=IDLE;

end

LR3:

begin

next\_state=IDLE;

end

endcase

end

//组合逻辑

always@(state or LEFT or RIGHT)

begin

if(~rst)

begin

LC=1'b1;LB=1'b1;LA=1'b1;RC=1'b1;RB=1'b1;RA=1'b1;

end

else

begin

LC=1'b1;LB=1'b1;LA=1'b1;RC=1'b1;RB=1'b1;RA=1'b1;

case(state)

IDLE:begin LC=1'b1;LB=1'b1;LA=1'b1;RC=1'b1;RB=1'b1;RA=1'b1; end

L1: begin LC=1'b1;LB=1'b1;LA=1'b0;RC=1'b1;RB=1'b1;RA=1'b1; end

L2: begin LC=1'b1;LB=1'b0;LA=1'b1;RC=1'b1;RB=1'b1;RA=1'b1; end

L3: begin LC=1'b0;LB=1'b1;LA=1'b1;RC=1'b1;RB=1'b1;RA=1'b1; end

R1: begin LC=1'b1;LB=1'b1;LA=1'b1;RC=1'b1;RB=1'b1;RA=1'b0; end

R2: begin LC=1'b1;LB=1'b1;LA=1'b1;RC=1'b1;RB=1'b0;RA=1'b1; end

R3: begin LC=1'b1;LB=1'b1;LA=1'b1;RC=1'b0;RB=1'b1;RA=1'b1; end

LR3: begin LC=1'b0;LB=1'b0;LA=1'b0;RC=1'b0;RB=1'b0;RA=1'b0; end

endcase

end

end

endmodule

B而后，本人在流水灯的基础上加入了呼吸灯，这里是直接调用的电子森林的呼吸灯模块，利用了两个计数器控制占空比，实现呼吸的效果，通过逻辑判断，在不同开关状态下，实现不同状态的转变

//产生计数器cnt1

always@(posedge clk or negedge rst) begin

if(!rst) begin

cnt1<=13'd0;

end

else if(cnt1>=CNT\_NUM-1)

cnt1<=1'b0;

else

cnt1<=cnt1+1'b1;

end

//产生计数器cnt2

always@(posedge clk or negedge rst)

begin

if(!rst) begin

cnt2<=13'd0;

flag<=1'b0;

end

else if(cnt1==CNT\_NUM-1) begin //当计数器1计满时计数器2开始计数加一或减一

if(!flag) begin //当标志位为0时计数器2递增计数，表示呼吸灯效果由暗变亮

if(cnt2>=CNT\_NUM-1) //计数器2计满时，表示亮度已最大，标志位变高，之后计数器2开始递减

flag<=1'b1;

else

cnt2<=cnt2+1'b1;

end

else begin

if(cnt2<=0) //当标志位为高时计数器2递减计数

flag<=1'b0; //计数器2级到0，表示亮度已最小，标志位变低，之后计数器2开始递增

else

cnt2<=cnt2-1'b1;

end

end

else

cnt2<=cnt2; //计数器1在计数过程中计数器2保持不变

end

//呼吸灯 判断条件

always@(state or LEFT or RIGHT)

begin

if(~rst)

begin

led1=1'b1;led2=1'b1;

end

else

begin

led1=1'b1;led2=1'b1;

if(LEFT&&(!RIGHT))

begin led1= (cnt1<cnt2)?1'b0:1'b1;

led2=1'b1;

end

else if((~LEFT)&&RIGHT)

begin led2= (cnt1<cnt2)?1'b0:1'b1;

led1=1'b1;

end

else if(LEFT&&RIGHT)

begin

led2= (cnt1<cnt2)?1'b0:1'b1;

led1= (cnt1<cnt2)?1'b0:1'b1;

end

else

begin

led1=1'b1;//led2=1'b1;

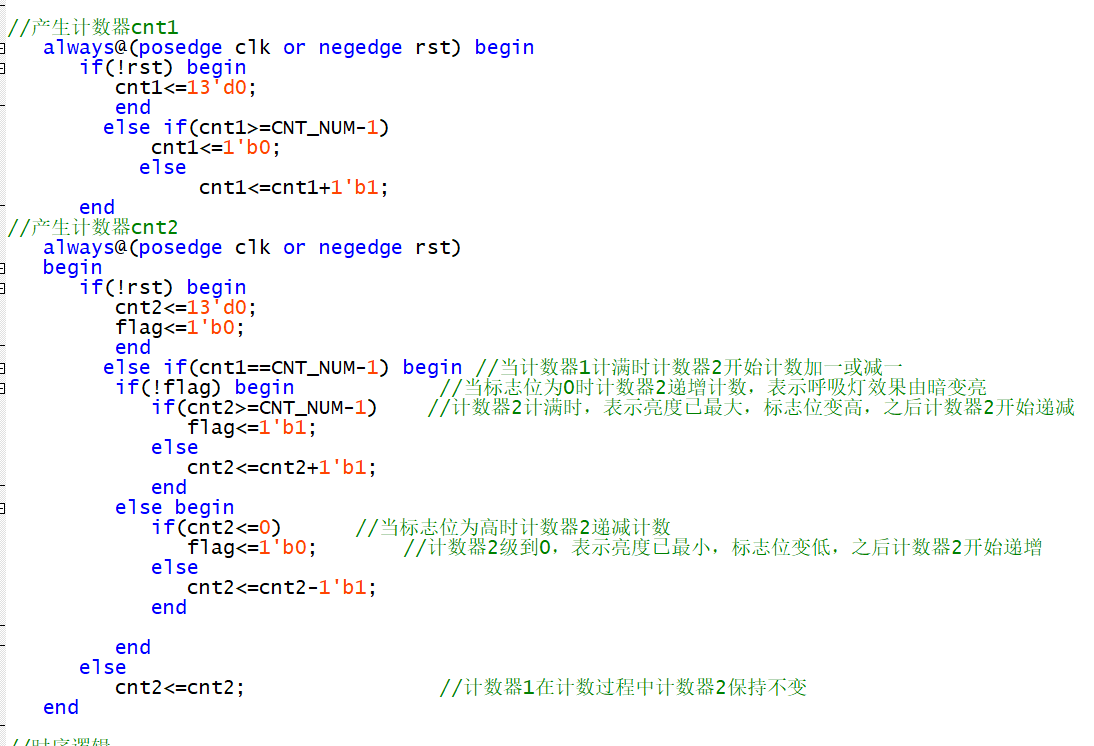
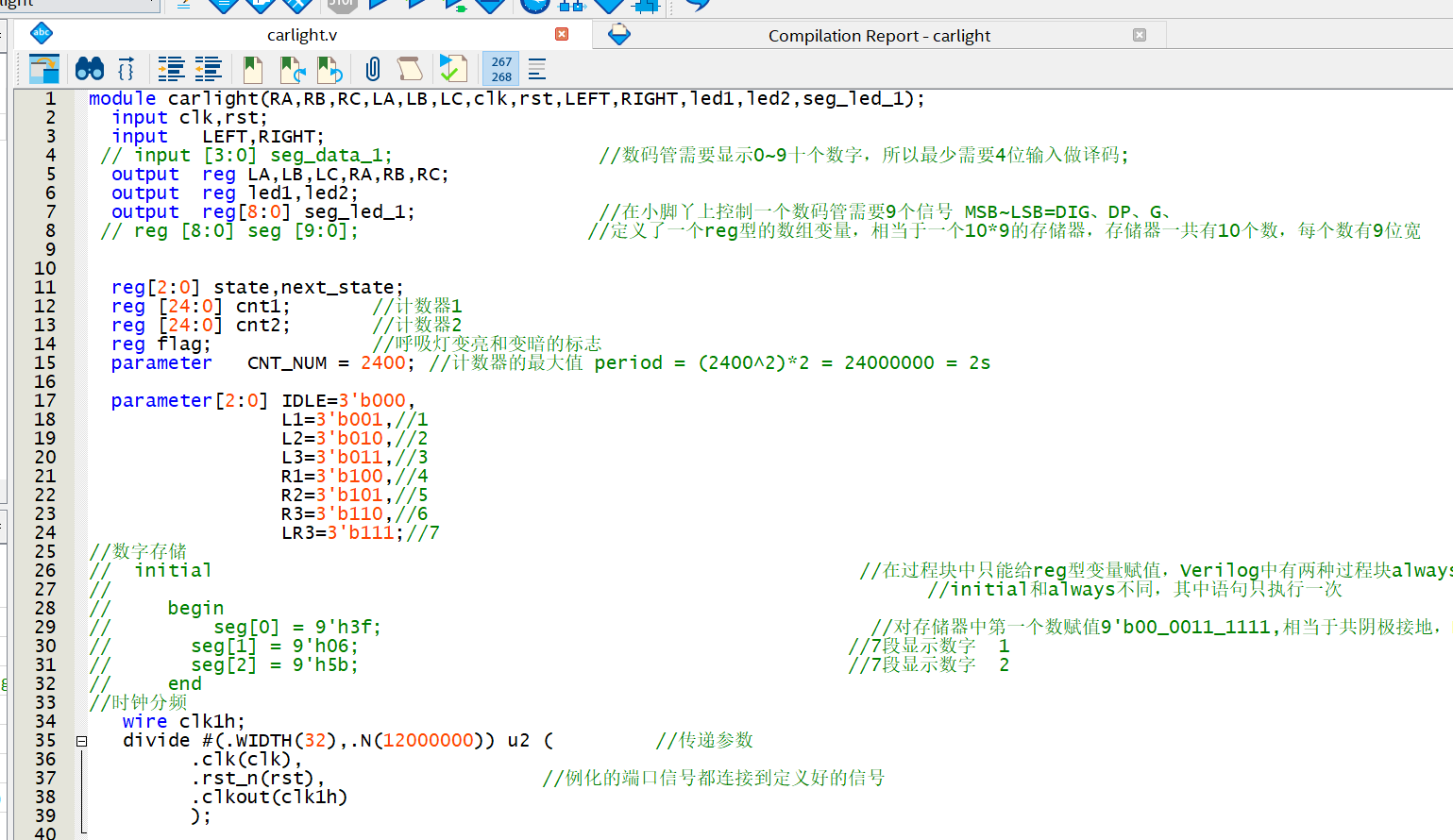
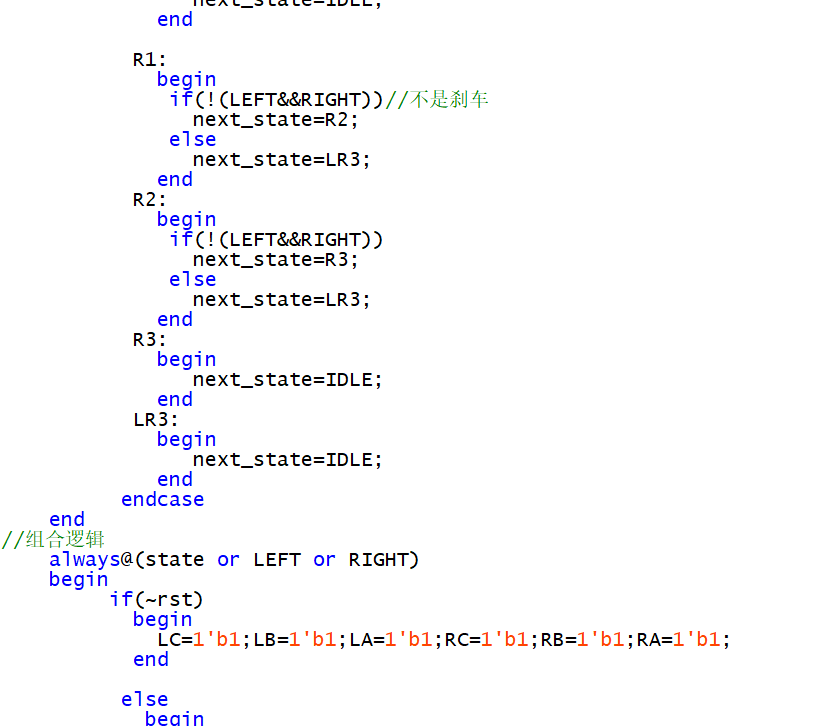
end

end

end

c最后，我在流水灯和呼吸灯的基础上加上了数码管，只需要给数码管分配赋值的位数即可，在上面判断呼吸灯的条件下，嵌入数码管的赋值语句

下面赋上全部代码





代码总结：可能逻辑上面还是存在一些问题，rgb呼吸灯预期效果:两个rgb灯闪烁没有实现，只实现了一个rgb灯的闪烁，数码管也只是采用了一种静态点亮的方式

1. **仿真**

仿真文件

`timescale 1 ps/ 1 ps

module carlight\_vlg\_tst();

//reg

reg eachvec;

reg LEFT;

reg RIGHT;

reg clk;

reg rst;

// wires

wire LA;

wire LB;

wire LC;

wire RA;

wire RB;

wire RC;

wire led1;

wire led2;

wire [8:0] seg\_led\_1;

//信号初始化

initial begin

clk = 1'b0;

rst = 1'b0;

#200 //延时200ns

rst = 1'b1;

end

//生成时钟

always

begin

#100 clk=~clk;

end

//例化待测模块

carlight i1 (

.LA(LA),

.LB(LB),

.LC(LC),

.LEFT(LEFT),

.RA(RA),

.RB(RB),

.RC(RC),

.RIGHT(RIGHT),

.clk(clk),

.led1(led1),

.led2(led2),

.rst(rst),

.seg\_led\_1(seg\_led\_1)

);

initial

begin

// rst=1'b0;

// rst=#200 1'b1;

#200 LEFT=1'b0;RIGHT=1'b0;

#200 LEFT=1'b1;RIGHT=1'b0;//左转

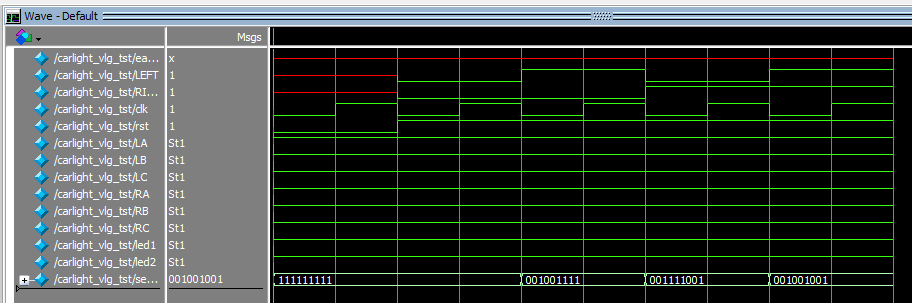
#200 LEFT=1'b0;RIGHT=1'b1;//左转

#200 LEFT=1'b1;RIGHT=1'b1;//右转

#200 $stop;

end

//$display("Running testbench");

endmodule

可能是逻辑有问题，也可能是仿真文件没写好，并没有仿真出led、呼吸灯和数码管实际效果

1. **个人感想:**

从开始什么也不会，到在网上查资料，在电子森林一点一点学习和摸索，4天花了几十个小时来做设计，虽然还有欠缺，但自己还是挺满意了，就像代码里面的状态机一样，你永远不知道下个状态是什么样，唯独能做好的是当前的状态。感谢老师在课程中给与的指导。