浙江工业大学

《可编程逻辑器件及应用》
课程实验报告

学生姓名 凌智城

指导教师 龚树凤

专业班级 通信工程 1803 班

培养类别 全日制本科

所在学院 信息工程学院

提交日期 2021 年 1 月 4 日

（完成正文后，再根据实际页码对目录页码进行补正）

目　录

（目录内容可以根据个人实际情况，适当调整）

实验四：交通灯控制系统的设计与实现

4.1 设计任务

设计一个交通灯控制系统。通过控制交通道路的通行和等待时间来实现交通控制，实现红绿灯状态控制并显示当前状态持续的时间。具有紧急状态、测试状态和正常工作三种状态。紧急状态用于处理一些突发的状态，如戒严等，此时双向路口禁止通行；测试状态可用于检测信号灯和数码管的硬件是否正常；正常工作状态则用于双向路口的信号灯控制。

4.2 设计方案

交通灯控制系统通常控制十字路口两个方向的信号灯，两个方向中车流量比较大的道路称为主干道，其绿灯的时间较长，而另一个方向就是次干道。两个路口的工作原理是相同的，主要区别是红、绿灯的时长不同，所以可以先实现一个路口的控制模块，然后再用该模块构成一个十字路口的控制系统。

a) 一个路口控制模块的设计：

复位状态、正常工作状态、紧急状态和信号灯测试状态，rst_n(复位信号)、emergency(紧急状态信号)和 test(测试状态信号)是状态控制输入信号。主干道在复位信号无效后，红、黄、绿灯的输出信号 ryg_light 立即为 3‘b001，即绿灯亮，由于 green_time=6，所以绿灯持续时间应为 6 秒钟，在 wait_time 输出分别为 6、5、4、3、2、1 后，ryg_light 为 3’b010，黄灯亮，黄灯持续 2 秒后，ryg_light=3'b100，红灯亮，红灯持续 9 秒钟后，ryg_light=3'b001，绿灯再次亮。次干道在复位信号无效后，红、黄、绿灯的输出信号 ryg_light=3‘b100，即红灯先亮。红灯持续 9 秒后依次是绿灯亮 6 秒、黄灯亮 2 秒。紧急状态控制信号 emergency 变化为 1 后电路的工作状态，wait_time=8'h88(图中显示的符号是十进制数，为-120)，信号灯 ryg_light=110，即红、黄灯同时亮的状态。当 emergency 无效后，输出信号又继续之前的工作状态。测试状态控制信号 test 变化为 1 后的工作状态，wait_time=8'h88(图中有符号十进制数为-120)，信号灯 ryg_light 交替为 3'b000 和 3'b111，即红、黄、绿信号灯交替同时亮或同时灭，用于测试信号灯的故障。当 test 信号无效后，输出信号又继续之前的工作状态。

b) 双向路口控制模块的设计：

该模块在实例化 traffic_con 模块时主要是设置主、次干道标志 prim_flag 以及主、次干道的信号灯时间。为了保证双向信号灯的同步，即主干道绿灯亮时次干道应为红灯，主干道绿灯结束后黄灯亮时，次干道仍为红灯，因此次干道的红灯时间应为主干道绿灯与主干道黄灯之和。同理，主干道的红灯时间应为次干道绿灯与次干道黄灯之和，即次干道绿灯时间为主干道红灯时间减去主干道黄灯时间(主、次干道的黄灯亮灯时间相同)。

4.3Verilog HDL 源代码

1) 一个路口控制模块

a) traffic_con.v

```
//凌智城 201806061211 通信工程 1803 班
module
traffic_con(clk,rst_n,prim_flag,red_time,green_time,yellow_time,wait_time,ryg_light,emergency,test
);

parameter on=1'b1,off=1'b0;
input clk,rst_n;                    //1Hz 时钟信号，低电平复位信号
input prim_flag;                    //主、次干道标志，1 为主干道，0 为次干道
input [7:0]red_time,green_time,yellow_time;//红绿黄灯时间（秒）
input emergency,test;               //紧急状态控制信号，信号灯测试控制信号
output reg[7:0]wait_time;           //当前状态的倒计时时间输出
output reg[2:0]ryg_light;           //红、黄、绿信号灯状态输出

reg cnt;                                //计数器
reg [7:0]ticks,n;                   //ticks 当前状态会达到的最终计数值，n 当前状态的时钟计数
值
reg [1:0]s,state;                   //s 子状态，00 绿，01 黄，10 红；state 工作状态，00 紧急，
01 测试，10 复位，11 正常

initial                 //初始化红灯，计数从 0 开始，子状态绿灯，当前时钟计数值设置为 0
    begin
        ryg_light<={on,off,off};
        cnt<=1'b0;
        s<=2'b00;
        ticks<=8'b11111111;
        n<=0;
    end

always@(s or state)
begin
if(state==2'b11)
            case(s)                 //当前子状态信号灯处理
                2'b00:
                        ticks=green_time;
                2'b01:
                        ticks=yellow_time;
                2'b10:
                        ticks=red_time;
            endcase
end
```

```
always@(posedge clk)
begin
    cnt<=~cnt;                          //cnt 实现闪烁处理功能

    begin
    if(~rst_n)                          //复位处理
        begin
            state<=2'b10;
            ryg_light<={off,off,off};   //复位状态时，全灭
            cnt<=1'b0;
            if(prim_flag)               //如果设置成主干道，则初始绿灯
                s<=2'b00;
            else                        //若为次干道，则初始为红灯
                s<=2'b10;
            n<=0;
        end
    else if(emergency)                  //紧急状态处理
        begin
            state<=2'b00;
            ryg_light<={on,on,off};
        end
    else if(test)                       //测试状态处理
        begin
            state<=2'b01;
            if(~cnt)
                ryg_light<={on,on,on};
            else
                ryg_light<={off,off,off};
        end
    else                                //正常工作状态处理
        begin
            state<=2'b11;
            case(s)                     //当前子状态信号灯处理
                2'b00:
                        ryg_light<={off,off,on};
                2'b01:
                        ryg_light<={off,on,off};
                2'b10:
                        ryg_light<={on,off,off};
            endcase
            if(n==ticks)
                begin
                    if(s==2'b10)        //子状态切换
```

```
                    s<=2'b00;
                else
                    s<=s+1;
                n<=1;
            end
        else
            n<=n+1;                         //当前时钟计时
    end
    wait_time=(state==2'b11)?ticks-n+1:8'h88;        //计算倒计时时间
end
endmodule
```

b) traffic_con.vt

```
//凌智城 201806061211 通信工程 1803
`timescale 1 ps/ 1 ps
module traffic_con_vlg_tst();
// constants
// test vector input registers
reg clk;
reg emergency;
reg [7:0] green_time;
reg prim_flag;
reg [7:0] red_time;
reg rst_n;
reg test;
reg [7:0] yellow_time;
// wires
wire [2:0]    ryg_light;
wire [7:0]    wait_time;

// assign statements (if any)
traffic_con i1 (
// port map - connection between master ports and signals/registers
    .clk(clk),
    .emergency(emergency),
    .green_time(green_time),
    .prim_flag(prim_flag),
    .red_time(red_time),
    .rst_n(rst_n),
    .ryg_light(ryg_light),
    .test(test),
    .wait_time(wait_time),
    .yellow_time(yellow_time)
```

```
);

always                          //周期设置为 10
    begin
        #5 clk=1'b1;
        #5 clk=1'b0;
    end

initial
begin
    clk <= 0;
    rst_n <= 0;
    emergency <= 0;
    test <= 0;
    prim_flag <= 1;             //先设置成主干道
    red_time <= 9;              //红绿黄时间设置为 9，6，2
    green_time <= 6;
    yellow_time <= 2;
    #20 rst_n<=1;               //再过 20 的时候低电平复位信号取消
    #350                        //再经过 350，信号复位，prim_flag=0 设置成次干道
    begin
        rst_n<=0;
        prim_flag <= 0;
    end
    #20 rst_n<=1;               //再过 20 的时候低电平复位信号取消
    #350                        //再经过 350，更改红绿黄灯等待时间
    begin
        red_time <= 12;
        green_time <= 8;
        yellow_time <= 4;
    end
    #350 emergency<=1;          //再经过 350，更改为紧急状态测试
    #350
    begin                       //再经过 350，取消紧急状态，更改为测试状态测试
        emergency<=0;
        test<=1;
    end
end

endmodule
```

2) 双向路口控制模块

a) traffic_top.v

```
//凌智城  201806061211  通信工程 1803
```

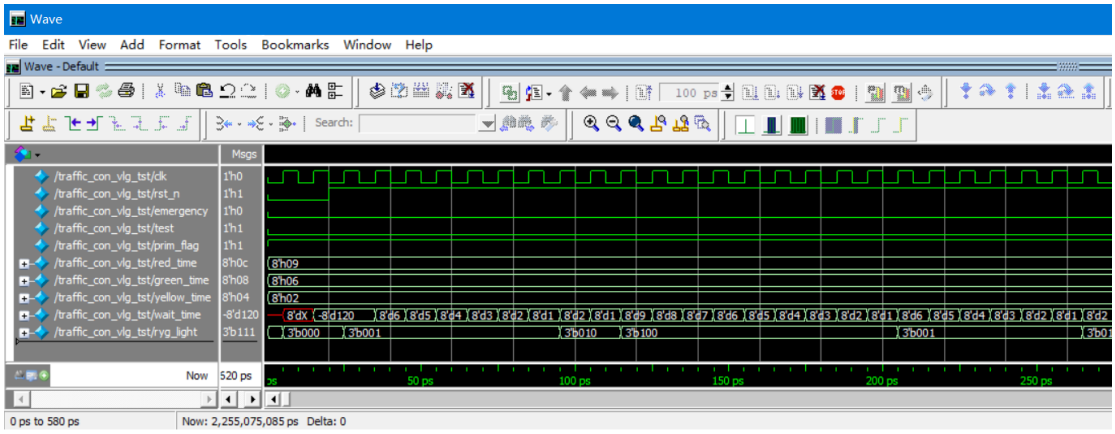
```
module
traffic_top(clk,rst_n,prim_red_time,prim_green_time,prim_yellow_time,prim_wait_time,seco_wait_t
ime,prim_ryg_light,seco_ryg_light,emergency,test);

input clk,rst_n;                          //1Hz 时钟信号,低电平复位信号
input[7:0]prim_red_time,prim_green_time,prim_yellow_time; //主干道红绿黄时间（秒）
input emergency,test;                                     //紧急状态控制信号，信
号灯测试控制信号
output[7:0]prim_wait_time,seco_wait_time;                 //主干道次干道倒计时时间
output[2:0]prim_ryg_light,seco_ryg_light;                 //主干道次干道红黄绿信号灯

wire[7:0]seco_red_time,seco_green_time,seco_yellow_time;

assign seco_red_time=prim_green_time+prim_yellow_time;
        //(11=9+2)，次干道红灯时间=主干道绿灯时间+主干道黄灯时间
assign seco_green_time=prim_red_time-prim_yellow_time;
        //(4=6-2)，次干道绿灯时间=主干道红灯时间-主干道黄灯时间
assign seco_yellow_time=prim_yellow_time;
        //(2=2)，主次干道黄灯时间相同

traffic_con
primary_light(clk,rst_n,1'b1,prim_red_time,prim_green_time,prim_yellow_time,prim_wait_time,pri
m_ryg_light,emergency,test);
traffic_con
secondary_light(clk,rst_n,1'b0,seco_red_time,seco_green_time,seco_yellow_time,seco_wait_time,se
co_ryg_light,emergency,test);

endmodule
```

b) traffic_top.vt

```
//凌智城 201806061211 通信工程 1803
`timescale 1 ps/ 1 ps
module traffic_top_vlg_tst();
// constants
// test vector input registers
reg clk;
reg emergency;
reg [7:0] prim_green_time;
reg [7:0] prim_red_time;
reg [7:0] prim_yellow_time;
reg rst_n;
reg test;
// wires
wire [2:0]    prim_ryg_light;
wire [7:0]    prim_wait_time;
```

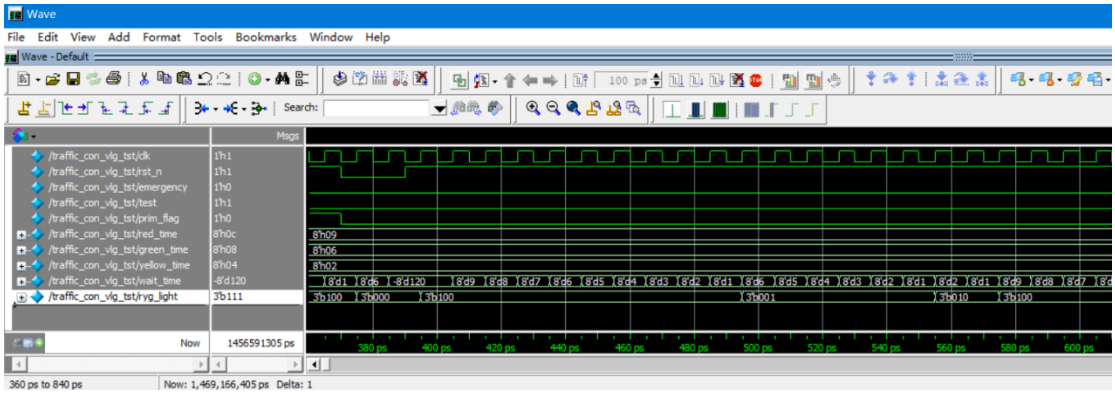
```
wire [2:0]    seco_ryg_light;
wire [7:0]    seco_wait_time;

// assign statements (if any)
traffic_top i1 (
// port map - connection between master ports and signals/registers
        .clk(clk),
        .emergency(emergency),
        .prim_green_time(prim_green_time),
        .prim_red_time(prim_red_time),
        .prim_ryg_light(prim_ryg_light),
        .prim_wait_time(prim_wait_time),
        .prim_yellow_time(prim_yellow_time),
        .rst_n(rst_n),
        .seco_ryg_light(seco_ryg_light),
        .seco_wait_time(seco_wait_time),
        .test(test)
);
always
        begin
                #5 clk=1'b1;
                #5 clk=1'b0;
        end

initial
begin
        clk <= 0;
        rst_n <= 0;
        emergency <= 0;
        test <= 0;
        prim_red_time <= 6;
        prim_green_time <= 9;
        prim_yellow_time <= 2;
        #20 rst_n<=1;
end
endmodule
```

4.4 实验结果与分析



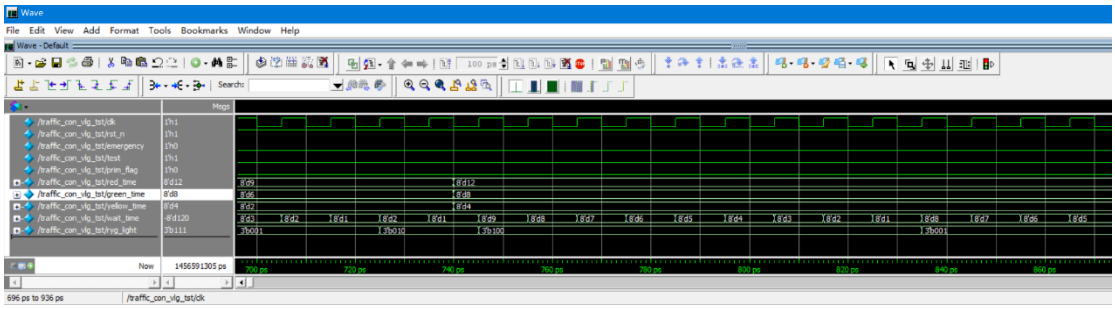
prim_flog=1，表示该路口是主干道。因此，在复位信号无效后，红、黄、绿灯的输出信号 ryg_light 立即为 3‘b001，即绿灯亮，由于 green_time=6，所以绿灯持续时间应为 6 秒钟，在 wait_time 输出分别为 6、5、4、3、2、1 后，ryg_light 为 3’b010，黄灯亮，黄灯持续 2 秒后，ryg_light=3'b100，红灯亮，红灯持续 9 秒钟后，ryg_light=3'b001，绿灯再次亮。

图 4-1 主干道复位仿真波形截图



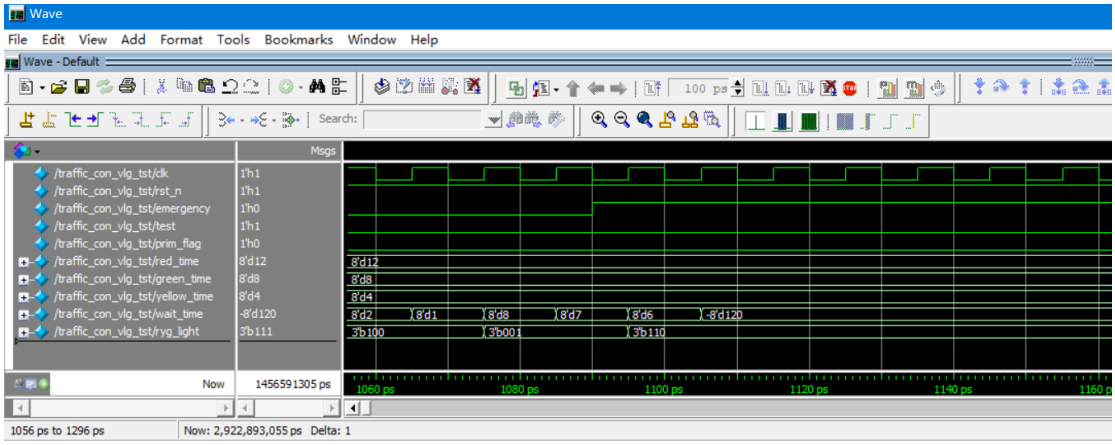
prim_flag=0，表示该路口是次干道。在复位信号无效后，红、黄、绿灯的输出信号 ryg_light=3‘b100，即红灯先亮。红灯持续 9 秒后依次是绿灯亮 6 秒、黄灯亮 2 秒。

图 4-1 次干道复位仿真波形截图



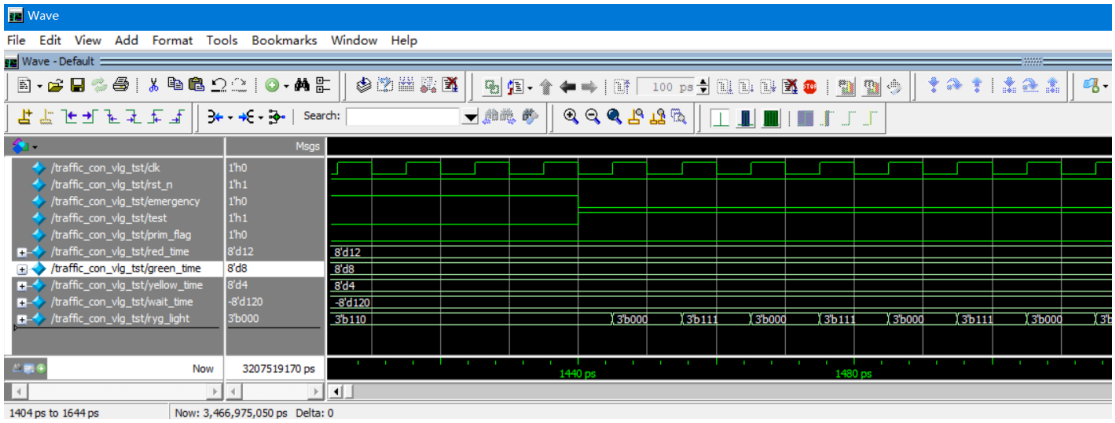
图中可以看出，红、绿、黄灯的通行时间分别由 9 秒、6 秒、2 秒变为 12 秒、8 秒和 4 秒后，新的灯时按照新输入时间运行；当前的绿灯状态结束后，随后的黄灯时间变为 4 秒，红灯时间变为 12 秒。

图 4-3 通行时间更新后仿真波形截图



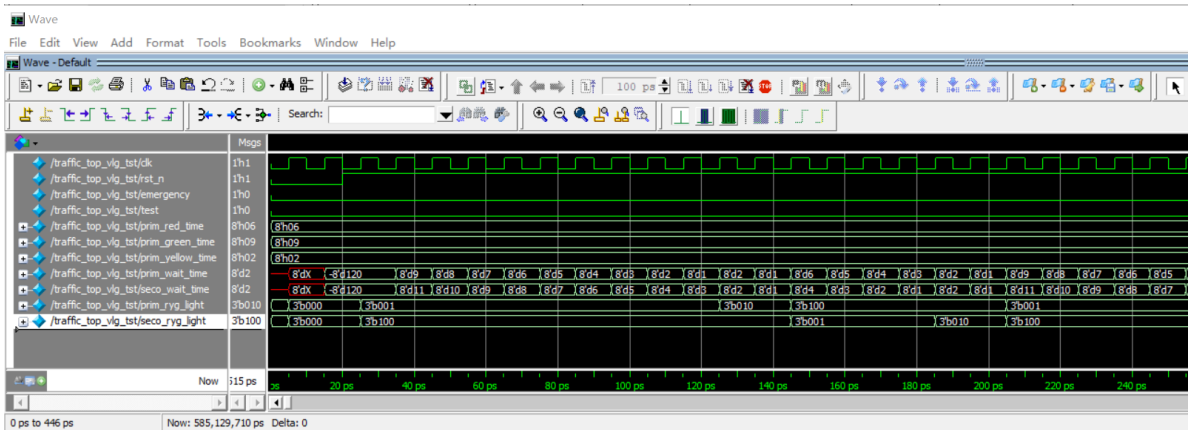
紧急状态控制信号 emergency 变化为 1 后电路的工作状态，wait_time=8'h88(图中显示的符号是十进制数，为-120)，信号灯 ryg_light=110，即红、黄灯同时亮的状态。当 emergency 无效后，输出信号又继续之前的工作状态。

图 4-4 紧急工作状态仿真波形截图



测试状态控制信号 test 变化为 1 后的工作状态，wait_time=8'h88(图中有符号十进制数为-120)，信号灯 ryg_light 交替为 3'b000 和 3'b111，即红、黄、绿信号灯交替同时亮或同时灭，用于测试信号灯的故障。当 test 信号无效后，输出信号又继续之前的工作状态。

图 4-5 测试工作状态仿真波形截图



双向路口控制模块的功能仿真波形，从图中可以看出输入信号中主干道的红、绿、黄灯时间分别为 6 秒、9 秒和 2 秒，因此次干道的红、绿、黄灯时间应分别为 11 秒、4 秒、2 秒，从图中可以看出复位信号由低变高后，主干道信号灯 prim_ryg_light=3'b001，即绿灯状态，次干道信号灯 seco_ryg_light=3'b100，即红灯状态。次干道的红灯时间(11 秒)是主干道的绿灯(9 秒)和黄灯(2 秒)时间之和，主干道的红灯时间(6 秒)是次干道的绿灯(4 秒)和黄灯(2 秒)时间之和。

图 4-6 双向路口控制模块仿真波形截图