洛 阳 理 工 学 院

课 程 设 计 报 告

课程名称 微控制系统课程实践

设计题目 十字路口交通灯控制器

专　　业 物联网工程

班　　级 B230408

学　　号 B23040815

姓　　名 张佳一

完成日期 2024 年 12 月 27 日

课 程 设 计 任 务 书

设计题目： 十字路口交通灯控制器

设计内容与要求：

一、设计内容

设计一个十字路口交通灯控制器。用单片机控制 LED 灯模拟指示。模拟东西方向的十字路口交通信号控制情况。要求能利用按键设置两个方向的通行时间（绿灯点亮的时间）和暂缓通行时间（黄灯点亮的时间），系统的工作符合一般交通灯控制要求

二、设计要求

1.分析系统功能，确定系统设计方案，掌握总体设计的方法与思路。

2.系统硬件设计，确定外设与单片机的硬件接口。掌握单片机系统外部接口的扩展设计方法。

3.系统软件设计，结合硬件设计，编写相应控制程序，并进行 Protuse 仿真执行。

4.熟练掌握程序烧录及调试过程。

5.按照要求撰写课程设计论文。

指导教师： 刘保罗

2024 年 12 月 15 日

课 程 设 计 评 语

成绩： 指导教师：_____

年 月 日

十字路口交通灯控制器

摘 要

近年来随着科技的飞速发展，单片机的应用正在不断深入，同时带动传统控制检测技术日益更新。在实时检测和自动控制的单片机应用系统中，单片机往往作为一个核心部件来使用，仅单片机方面知识是不够的，还应根据具体硬件结构软硬件结合，加以完善。

本设计旨在构建一个十字路口交通灯控制器，该控制器使用 AT89C52 单片机来控制 LED 灯模拟交通信号灯的指示。该系统能够通过按键设置东西方向的通行时间（绿灯）和暂缓通行时间（黄灯），以满足一般交通灯控制的要求。设计内容涵盖了系统功能的分析、硬件设计、软件设计以及仿真执行，同时包括了调试过程。设计通过两位一体共阴极数码管显示，并能通过按键对定时进行设置。本系统实用性强、操作简单、扩展功能强。

关键词： 交通灯，单片机，显示，计时

目 录

摘 要.....I

目 录.....II

一、设计目标与内容.....1

1 设计目标.....1

2 设计内容.....1

3 设计要求.....1

二、系统设计.....2

三、功能模块设计.....4

1 主控制器.....4

2 数码管显示模块.....4

3 LED 显示模块.....5

4 数码管倒计时模块.....6

5 红绿灯切换模块.....7

6 按键控制模块.....8

四、仿真与实物演示.....9

1 系统仿真.....9

2 调试中遇到的问题.....12

3 实物演示效果.....12

总结.....13

附 录.....14

1 红绿灯倒计时显示函数.....14

2 白天模式下 LED 切换.....15

3 数码管动态显示.....17

4 定时中断.....18

一、设计目标与内容

1 设计目标

(1) 实现东西方向交通灯的绿灯和黄灯的控制，以模拟真实交通灯的工作状态。

(2) 通过按键输入，允许用户自定义东西方向的通行时间和暂缓通行时间。

(3) 确保系统的工作符合一般交通灯控制的要求，即红灯、绿灯、黄灯的顺序和时序控制。

(4) 通过 Protuse 软件进行仿真，验证系统设计的准确性和可靠性。

(5) 熟练掌握程序烧录及调试过程，确保系统稳定运行。

2 设计内容

(1) 系统功能分析：明确交通灯控制器的基本功能和操作流程。

(2) 硬件设计：选择合适的单片机和外设，设计电路图，并确定硬件接口。

(3) 软件设计：编写控制程序，实现交通灯的逻辑控制。

(4) 按键设置：设计按键接口，允许用户设置通行时间和暂缓通行时间。

(5) LED 模拟：使用 LED 灯模拟交通灯的红、黄、绿灯状态。

(6) 仿真执行：在 Protuse 软件中进行仿真，测试系统的实际工作情况。

3 设计要求

(1) 功能实现：系统应能够实现基本的交通灯控制功能，包括红、黄、绿灯的顺序切换。

(2) 用户交互：系统应提供用户交互界面，通过按键输入设置通行和暂缓时间。

(3) 仿真测试：在 Protuse 软件中进行仿真测试，确保设计的交通灯控制器在理论上是可行的。

(4) 程序烧录与调试：程序烧录后，系统应能稳定运行，调试过程中应能迅速定位并解决问题

二、系统设计

本设计方案旨在构建一个十字路口交通灯控制器，使用 51 单片机作为核心控制单元，通过 LED 灯模拟交通信号灯的状态。系统将允许用户通过按键设置东西方向的绿灯和红灯持续时间，或者南北方向的绿灯和红灯持续时间。也可以通过按键切换夜间/白天模式，或者设置东西方向常绿等。

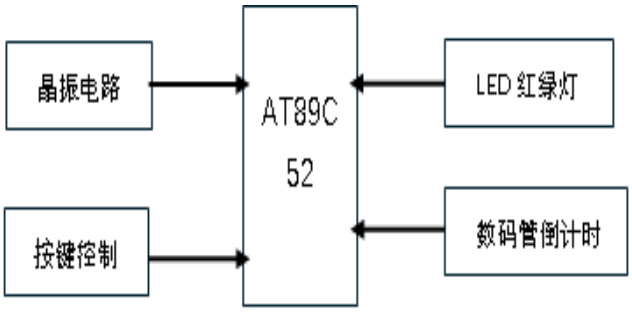

图 2.1 总体框图

键盘设置模块对系统输入模式选择及具体通行时间设置的信号，系统进入正常工作状态，执行交通灯状态显示控制，同时将时间数据倒计时输入到 LED 数码管上实时显示。

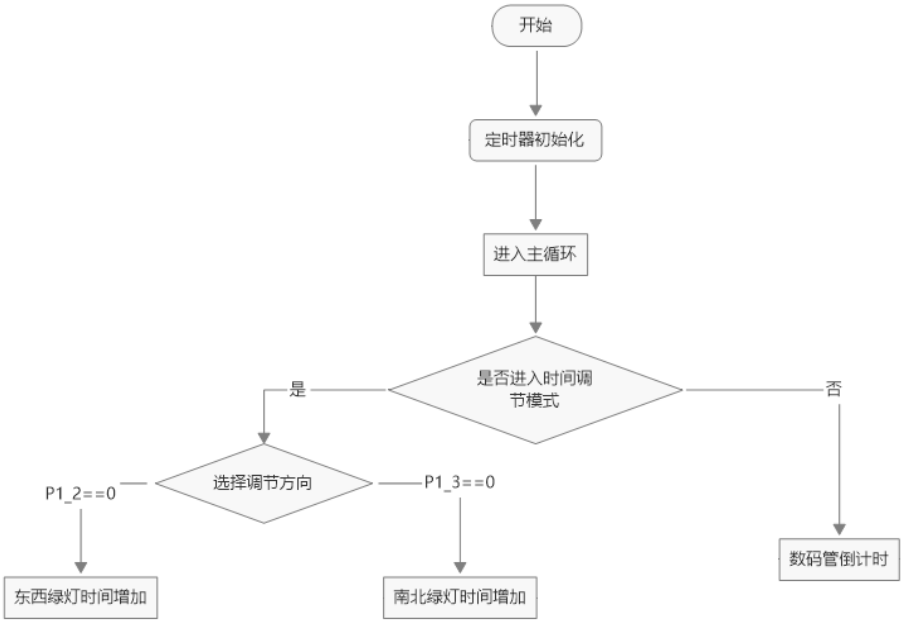

图 2.2 主循环流程图

如图 2.2 所示，在 main 函数中首先进行定时器的初始化，打开中断。然后进入主循环，在主循环中采用查询的方式来判断调节绿灯时间的按键是否按下，如果按下调节开关，则进入时间调节模式，否则，在主循环中一直进行数码管的动态显示。

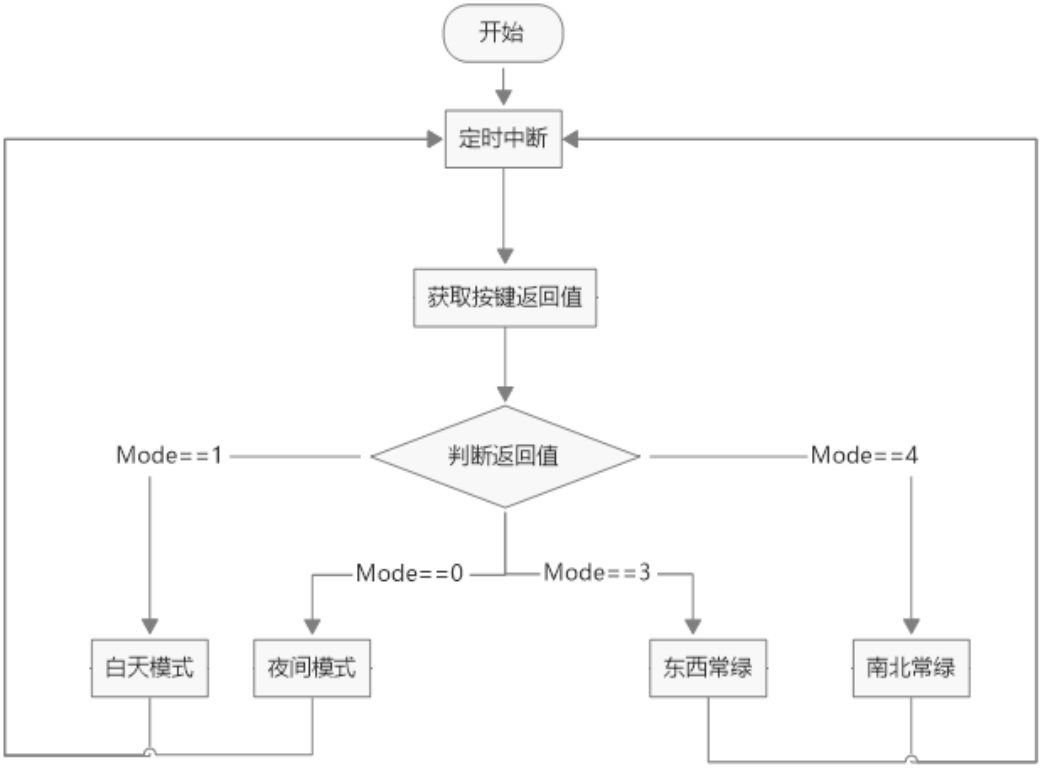

图 2.3 定时中断流程图

本项目中开启了定时器中断 0，通过每隔一秒计数加一，进而调控各个方向红绿灯的切换。同时在这里面也采用了查询的方式来不断的检查 key()函数的返回值，通过对返回值的判断而进入不同的模式。

如图 2.3 所示，进入中断后除了每隔一秒计数加一外，还不断的查询着按键的返回值，当返回值为 1 时，则进入白天模式，红绿灯正常运行；当返回值为 0 时，进入夜间模式，各个方向的黄灯都开始闪烁；当按键为 3 时东西东西常绿，为 4 时南北常绿。

三、功能模块设计

1 主控制器

主控制器是整个交通灯系统的大脑，负责协调和控制其他模块的工作。它使用 51 单片机实现，具体功能包括初始化系统、处理定时器中断、更新交通灯状态、响应按键输入以及切换不同的工作模式。

利用单片机的中断服务程序来处理定时器中断，更新交通灯状态。通过检测特定端口的电平变化来响应按键输入，从而调整时间设置或切换模式。

main.c 文件中包含了主控制器的主要逻辑，包括无限循环检测按键输入和调用 Countdown 函数来更新交通灯状态。中断服务程序负责处理定时器中断，更新 ModeChose 和 count 变量，并根据 ModeChose 的值调用不同的模式函数。

2 数码管显示模块

```
8  void nixie1(unsigned char location,unsigned char  number) {
9      switch (location) {
10         case 11:
11             P1_5 = 0;
12             P1_4 = 1;
13             P1_6 = 0;
14             P1_7 = 0;
15             break;
16         case 12:
17             P1_5 = 1;
18             P1_4 = 0;
19             P1_6 = 0;
20             P1_7 = 0;
21             break;
22
23         case 21:
24             P1_4 = 0;
25             P1_5 = 0;
26             P1_6 = 0;
27             P1_7 = 1;
28             break;
29         case 22:
30             P1_4 = 0;
31             P1_5 = 0;
32             P1_6 = 1;
33             P1_7 = 0;
34             break;
35     }
36     P0 = Digital_tube[number];
37     Delay(1);
38     P0=0xff;
39 }
40
41 void Twonixie1(unsigned char  number){
42     nixie1(11,number%10);
43     nixie1(12,number/10);
44 }
```

图 3.1 数码管代码

如图 3.1 所示，首先定义函数 void nixie1(unsigned char location,unsigned char number)；来选择其中一个数码管来显示一位数字。进而通过 void Twonixie1(unsigned char number)；来使东西方向的数码管来显示指定的两位数。然后，在主循环内通过对数码管的快速切换，达到所有数码管同时显示的效果。

3 LED 显示模块

```
4  /*********************************************
5   *
6   *          南北方向
7   *
8   *********************************************/
9  void Red1_ON(void){
10     P2_0=0;
11 }
12 void Red1_OFF(void){
13     P2_0=1;
14 }
15 void Red1_Turn(void){
16     P2_0=~P2_0;
17 }
18 void Green1_ON(void){
19     P2_1=0;
20 }
21 void Green1_OFF(void){
22     P2_1=1;
23 }
24 void Green1_Turn(void){
25     P2_1=~P2_1;
26 }
27 void Yellow1_ON(void){
28     P2_2=0;
29 }
30 void Yellow1_OFF(void){
31     P2_2=1;
32 }
33 void Yellow1_Turn(void){
34     P2_2=~P2_2;
35 }
36 void Yellow1_Flash(void){
37     if(count%5==0){
38         Yellow1_Turn();
```

图 3.2 LED 部分代码

通过设置 GPIO 端口的电平来点亮或熄灭相应的 LED 灯，为每个 IED 灯可能需要用到的功能定义一个便于识别的函数，以便于后序的直接调用。

4 数码管倒计时模块

```
void Countdown(unsigned char Green_Time1,unsigned char Green_Time2){

    if(Num<Green_Time1)
    {
        //1方向的绿灯
        Twonixie1(Green_Time1-Num);

        Twonixie2(Green_Time1+Yellow_Time-Num);//2红灯
    }
    else if(Num>=Green_Time1&&Num<(Green_Time1+Yellow_Time))
    {
        //1方向的黄灯变为红灯
        Twonixie1((Green_Time1+Yellow_Time)-Num);//1黄灯
        //2方向的红灯变为绿灯
        Twonixie2(Green_Time1+Yellow_Time-Num);//2红灯

    }
    if(Num>=(Green_Time1+Yellow_Time)&&Num<(Green_Time2+Green_Time1+Yellow_Time))
    {
        Twonixie1((Green_Time1+Yellow_Time+Green_Time2+Yellow_Time)-Num);//1红灯
        Twonixie2((Green_Time2+Yellow_Time+Green_Time1)-Num);//2绿灯

    }
    if(Num>=(Green_Time1+Yellow_Time+Green_Time2)&&Num<(Green_Time1+Yellow_Time*2+Green_Time2))
    {
        //2方向的绿灯变为红灯

        //1方向的红灯变为绿灯
        Twonixie1((Green_Time1+Yellow_Time*2+Green_Time2)-Num);//1红灯

        Twonixie2((Green_Time1+Yellow_Time*2+Green_Time2)-Num);//2黄灯

    }
}
```

图 3.3 数码管倒计时

数码管倒计时显示可以提醒驾驶员在信号灯颜色发生改变的时间、在“停止”和“通过”两者间作出合适的选择。驾驶员和行人普遍都愿意选择有倒计时显示的信号控制方式，并且认为有倒计时显示的路口更安全。

如图 3.3 所示，在此段代码中 Num 在定时计数器中实现每秒自增，由于在初始化时方向 1 默认绿灯方向 2 默认红灯，所以在 Num 小于方向 1 的绿灯时间时，将 Green_Time1-Num 即为该方向的绿灯倒计时，将该数值传递给方向 1 的数码管显示函数 Twonixie1()；同时将红灯的数值传递给方向 1 的数码管显示函数，即可完成一次倒计时，然后计算出各个灯变化的时间结点，将此方向的数值传递给相应的数码管显示函数，即可完成一轮倒计时。

5 红绿灯切换模块

```
45 void DayMode(unsigned char Green_Time1,unsigned char Green_Time2){
46
47     if((Num-1)==0){
48         //初始化1方向为绿灯，2方向为红灯
49         Green1_ON();
50         Green2_OFF();
51         Red2_ON();
52         Red1_OFF();
53
54         Yellow1_OFF();
55         Yellow2_OFF();
56
57     }
58
59     if(Green_Time1<=Num&&Num<= (Green_Time1+Yellow_Time)){
60         //1方向的绿灯变为黄灯
61         Green1_OFF();
62         Yellow1_Flash();
63     }
64
65     if(Num==(Green_Time1+Yellow_Time))
66     {
67         //1方向的黄灯变为红灯
68         Yellow1_OFF();
69         Red1_ON();
70         //2方向的红灯变为绿灯
71         Red2_OFF();
72         Green2_ON();
```

图 3.4 白天模式部分代码

如图 3.4 所示，在函数运行时先设置其初始方向，通过 LED 显示模块函数的调用来设置方向 1 为绿灯，方向 2 为红灯。与数码管倒计时模块类似，通过计算出各个方向红绿灯切换的时间结点，来改变 LED 灯，进而实现红绿灯切换显示的功能。

```
90 void NightMode(void){
91     Num=0;
92     Red1_OFF();
93     Red2_OFF();
94     Green1_OFF();
95     Green2_OFF();
96     Yellow2_Flash();
97     Yellow1_Flash();
98 }
99
```

图 3.5 夜间模式

如图 3.5 所示，在夜间模式中通过 LED 函数的调用即可快速实现夜间红绿灯的控制，让两个方向的黄灯闪烁，其余都关闭即可。

与之相似的还有设置东西和南北方向的常绿，将该方向的绿灯打开反方向的红灯打开，其余关闭即可。

6 按键控制模块

本项目的按键控制通过查询和中断两种方式完成。

```
unsigned char Key()
{

    if(P1_0==0){Delay(20);while(P1_0==0);Delay(20);ModeChose=2;}
    if(P1_1==0){Delay(20);while(P1_1==0);Delay(20);ModeChose=3;}
    if(P1_2==0){Delay(20);while(P1_2==0);Delay(20);ModeChose=!ModeChose;ldelay=0;}
    return ModeChose;
}
```

图 3.6 按键获取返回值

如图 3.6 所示，此代码放在循环中通过不断查询获取不同的返回值，进而决定执行什么功能。而通过此函数主要在夜间/白天模式，东西/南北常绿这几个模式之间进行选择。

```
94  Green_time333() interrupt 0{
95      if(P3_2==0){
96          if(qiehuan==0){
97              Green_Time1++;
98          }
99          if(qiehuan==1){
100             Green_Time2++;
101         }
102     }
103 }
104
105
106 Green_time222() interrupt 2{
107     if(P3_3==0){
108         if(qiehuan==0){
109             Green_Time1--;
110         }
111         if(qiehuan==1){
112             Green_Time2--;
113         }
114     }
115 }
```

图 3.7 按键调整时间

在进行两个方向的时间调节时，为解决在使用查询方法时按键一直按压导致其他程序停止的问题，所以将其写进两个外部中断里，使其不妨碍其他程序的运行，切换按键可以切换加减的方向时间。通过安检模块的控制，使得整个系统具有灵活性，实用性。

四、仿真与实物演示

1 系统仿真

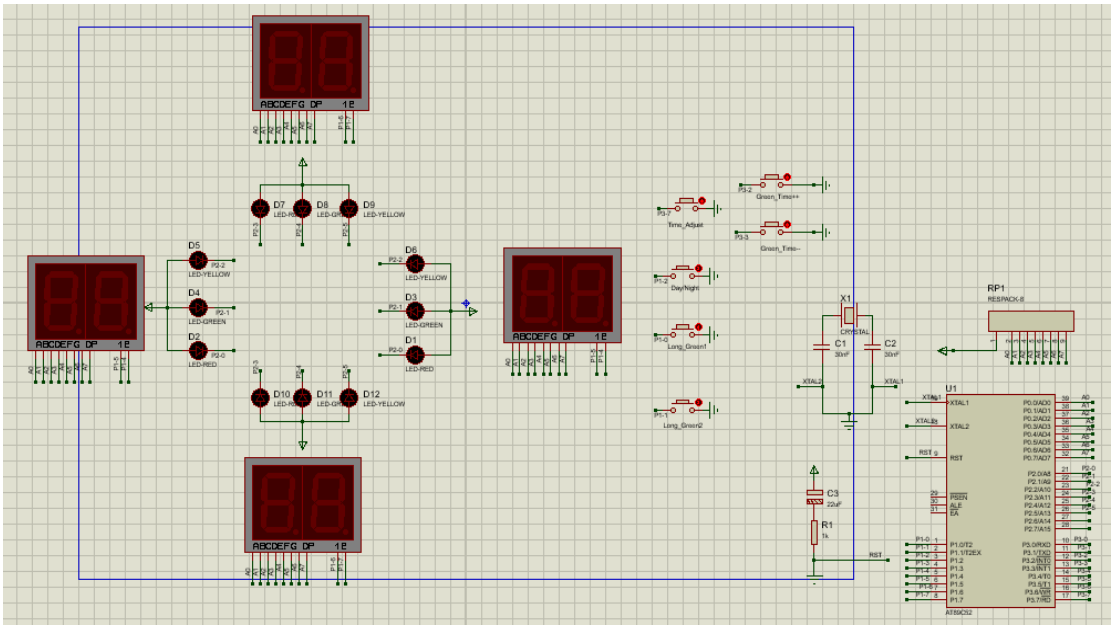

图 4.1 硬件连接

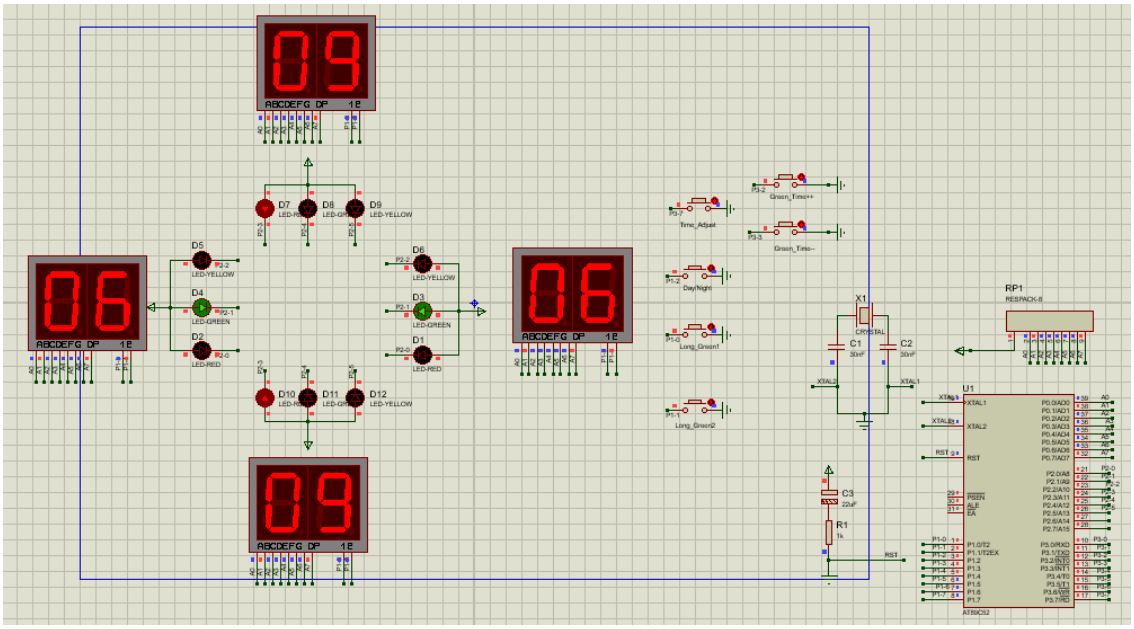

图 4.2 白天模式

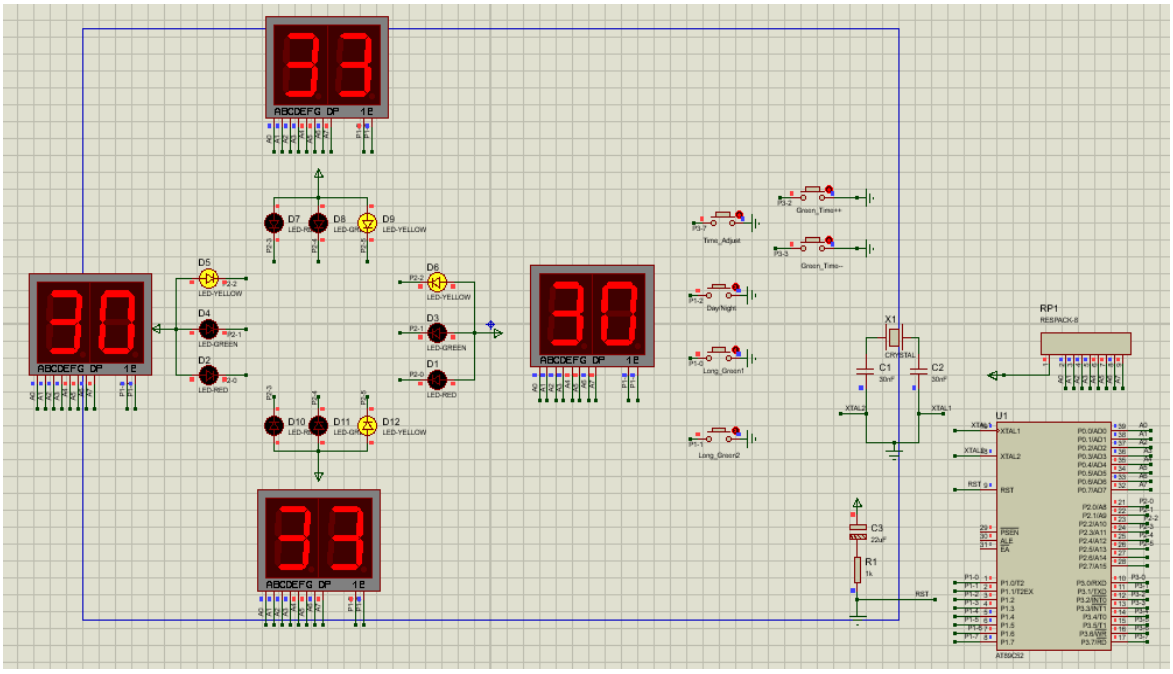

图 4.3 夜间模式

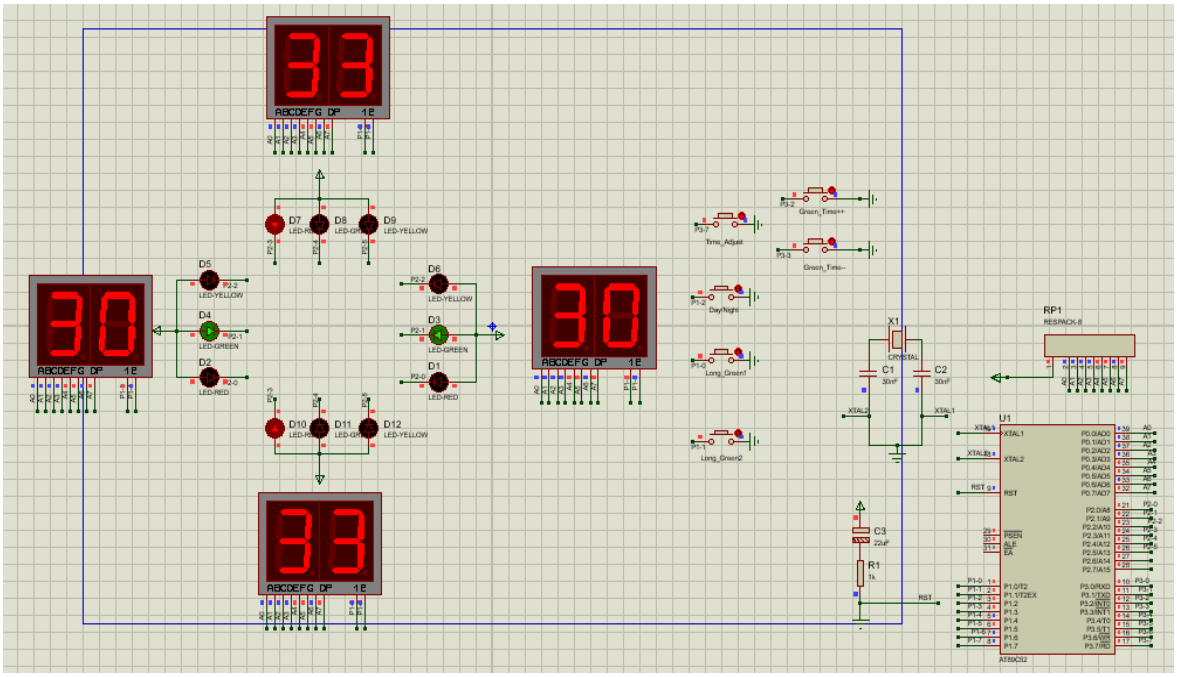

图 4.4 东西常绿

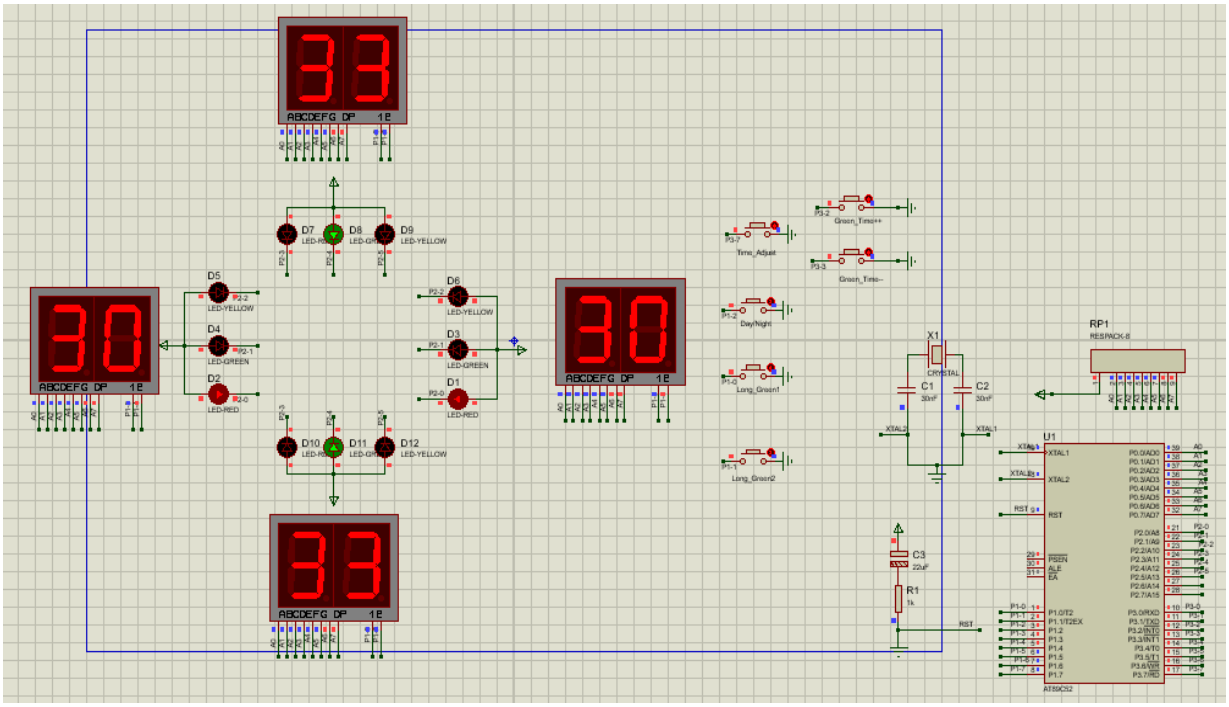

图 4.5 南北常绿

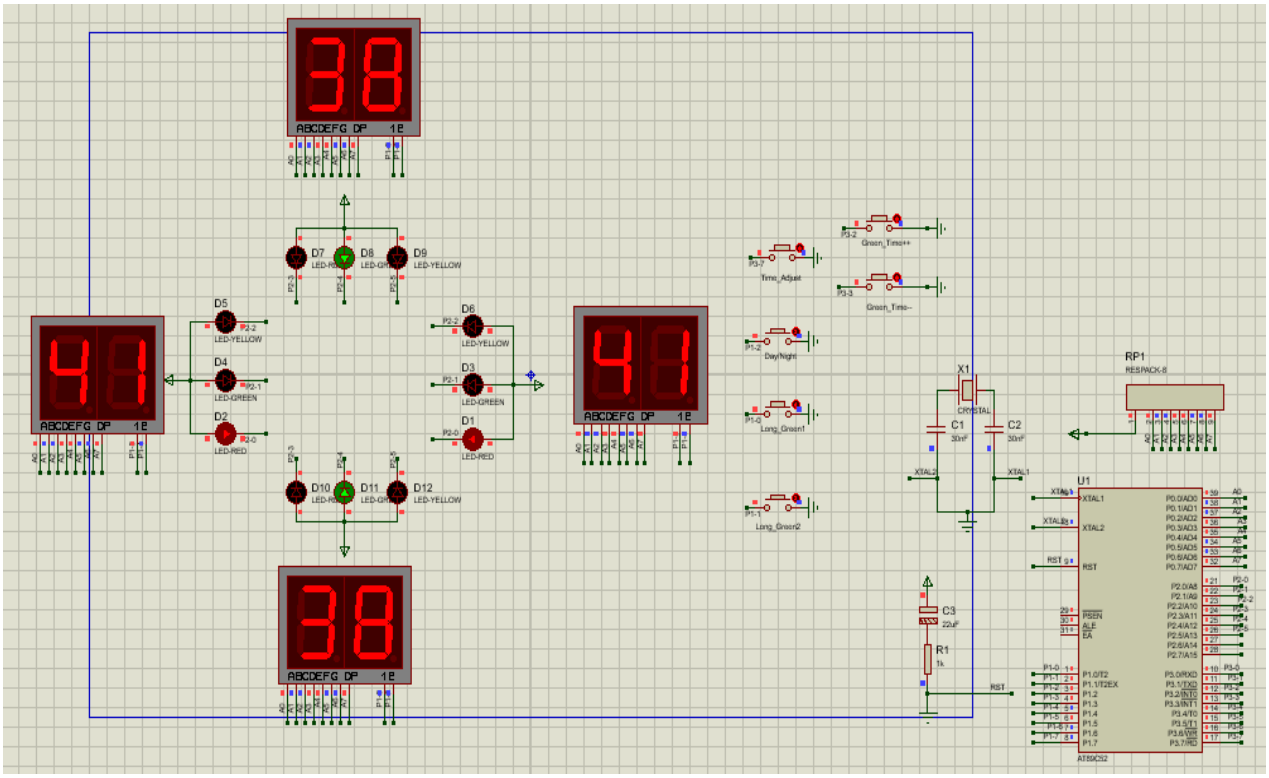

图 4.6 时间调节

2 调试中遇到的问题

定时器配置问题：定时器初值设置不正确可能导致定时不准确，影响倒计时和交通灯切换的时序。定时器中断频率设置不当可能导致 LED 闪烁过快或过慢。

数码管显示问题：数码管显示的数字可能不正确，需要检查数码管的段码是否正确。数码管刷新频率不够可能导致显示闪烁或不稳定。

代码逻辑错误：倒计时逻辑可能存在错误，导致倒计时不准确或不更新。交通灯状态切换逻辑可能存在错误，导致交通灯状态不正确。

模式切换逻辑问题：模式切换可能没有正确执行，或者某些模式没有按预期工作。检查 DayMode、NightMode、Long_Green1 和 Long_Green2 等函数的实现，确保它们正确地设置了交通灯的状态。

功能实现问题：某些功能可能没有实现，例如在调整红绿灯时间的模块中由于考虑不全，导致只能更改一侧的时间。解决方案是通过画图理清思路确保不遗漏所有情况。

按键控制问题：最开始在主循环中采用查询的方式来更改红绿灯的时间，但当按键一直按下不松时，循环就会一直卡在按键判定，无法正常现实数码管的倒计时。解决方法时利用外部中断将两个按键写入中断里使其不影响主循环的进程。

3 实物演示效果

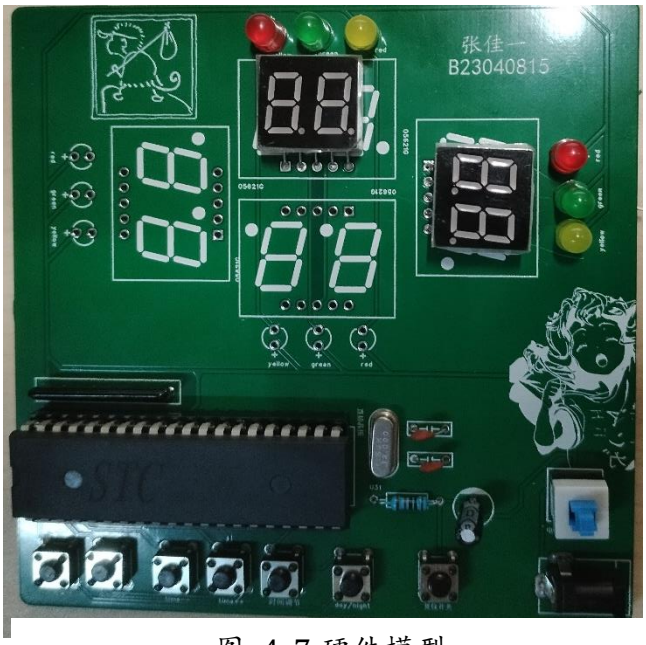

图 4.7 硬件模型

总结

在构建十字路口交通灯控制器的这个课程设计中我学到了许多东西，归纳起来主要有以下几个方面。

（1）通过这次课程设计，我能将以前所学到的专业知识与实践相联系，将所学到的知识充分运用到本次设计中。同时，我也认识到自己知识上不足的地方，体会到了所学理论知识的重要性，知识掌握得越多，设计得就更全面、更顺利、更好。

（2）进一步熟悉了单片机的知识。通过本次设计，我对单片机的基本原理、内部结构、各引脚功能、定时器和中断的应用都有了更深刻的理解。并且，能够以单片机为基础元件设计一个简单的系统。

（3）通过本次设计，熟悉了设计一个项目所必经的几个阶段。在设计过程中，我首先分析了系统的功能需求，并确定了设计方案。随后进行了硬件设计，包括选择合适的单片机和外设，并确定了硬件接口。在软件设计方面结合硬件设计编写了控制程序，并在 Protuse 软件中进行了仿真执行。这不仅锻炼了我独立完成设计工作的能力，更重要的是了解了一个电子产品的设计流程，为将来投入工作增加了宝贵的经验，奠定了坚实的基础。

（4）提高了自己查找资料的能力。在设计过程中，我碰到了一些暂时无法解决的问题，于是我通过上网查阅和图书馆借阅资料，或是通过与老师同学交流一步步地解决了。从中我懂得了我们这个专业的知识面相当广泛，我们需要不断通过各种途径更新自己的知识，不断充实自己，同时要懂得与他人交流意见，积极听取别人的建议，懂得不断学习的重要性。

本次课程设计不仅让我深入了解了嵌入式系统的设计流程，还锻炼了我的实践能力。虽然在设计和调试过程中遇到了不少挑战，但通过不断学习和努力，我最终成功实现了一个功能完整的十字路口交通灯控制器。

在指导老师的精心指导和严格要求下，获得了丰富的理论知识，极大地提高了实践能力，并对当前电子领域的研究状况和发展方向有了一定的了解，单片机领域这对我今后进一步学习计算机方面的知识有极大的帮助。这不仅增强了我对专业知识的理解，也为将来的学习和工作打下了坚实的基础。

附 录

1 红绿灯倒计时显示函数

```
void Countdown(unsigned char Green_Time1,unsigned char Green_Time2){

    if(Num<Green_Time1)
    {
        //1 方向的绿灯
        Twonixie1(Green_Time1-Num);

        Twonixie2(Green_Time1+Yellow_Time-Num);//2 红灯
    }
    else if(Num>=Green_Time1&&Num<(Green_Time1+Yellow_Time))
    {
        //1 方向的黄灯变为红灯
        Twonixie1((Green_Time1+Yellow_Time)-Num);//1 黄灯
        //2 方向的红灯变为绿灯
        Twonixie2(Green_Time1+Yellow_Time-Num);//2 红灯

    }
    if(Num>=(Green_Time1+Yellow_Time)&&Num<(Green_Time2+Green_Time1+
Yellow_Time))
    {
        Twonixie1((Green_Time1+Yellow_Time+Green_Time2+Yellow_Time)-
Num);//1 红灯
        Twonixie2((Green_Time2+Yellow_Time+Green_Time1)-Num);//2 绿灯

    }
    if(Num>=(Green_Time1+Yellow_Time+Green_Time2)&&Num<(Green_Time1+
```

```
Yellow_Time*2+Green_Time2))
    {
        //2 方向的绿灯变为红灯

        //1 方向的红灯变为绿灯
        Twonixie1((Green_Time1+Yellow_Time*2+Green_Time2)-Num);//1 红灯

        Twonixie2((Green_Time1+Yellow_Time*2+Green_Time2)-Num);//2 黄灯

    }
}
```

2 白天模式下 LED 切换

```
void DayMode(unsigned char Green_Time1,unsigned char Green_Time2){

    if((Num-1)==0){
        //初始化 1 方向为绿灯，2 方向为红灯
        Green1_ON();
        Green2_OFF();
        Red2_ON();
        Red1_OFF();

        Yellow1_OFF();
        Yellow2_OFF();

    }

    if(Green_Time1<=Num&&Num<= (Green_Time1+Yellow_Time)){
```

```
        //1 方向的绿灯变为黄灯
        Green1_OFF();
        Yellow1_Flash();
    }

    if(Num==(Green_Time1+Yellow_Time))
    {
        //1 方向的黄灯变为红灯
        Yellow1_OFF();
        Red1_ON();
        //2 方向的红灯变为绿灯
        Red2_OFF();
        Green2_ON();

    }

    if((Green_Time1+Yellow_Time+Green_Time2)<=Num&&Num<=
(Green_Time1+Yellow_Time+Green_Time2+Yellow_Time)){

        //2 方向的绿灯变为黄灯
        Green2_OFF();
        Yellow2_Flash();
    }

    if(Num==(Green_Time1+Yellow_Time+Green_Time2+Yellow_Time))
    {
        Num =0;
    }
}
```

3 数码管动态显示

```
void nixie1(unsigned char location,unsigned char    number) {
    switch (location) {
        case 11:
            P1_5 = 0;
            P1_4 = 1;
            P1_6 = 0;
            P1_7 = 0;
            break;
        case 12:
            P1_5 = 1;
            P1_4 = 0;
            P1_6 = 0;
            P1_7 = 0;
            break;

        case 21:
            P1_4 = 0;
            P1_5 = 0;
            P1_6 = 0;
            P1_7 = 1;
            break;
        case 22:
            P1_4 = 0;
            P1_5 = 0;
            P1_6 = 1;
            P1_7 = 0;
            break;
    }
```

```
    P0 = Digital_tube[number];
    Delay(1);
    P0=0xff;
}

void Twonixie1(unsigned char    number){
    nixie1(11,number%10);
    nixie1(12,number/10);
}

void Twonixie2(unsigned char    number){
    nixie1(21,number%10);
    nixie1(22,number/10);
}
```

4 定时中断

```
timer () interrupt 1{
    ModeChose=Key();
    count++;
    if(count==20)
    {
        count=0;
        Num++;

    }
    switch(ModeChose){
        case 1:
            DayMode(Green_Time1,Green_Time2);
            break;
```

```
        case 0:NightMode();
            break;
        case 2:Long_Green1();
            break;
        case 3:Long_Green2();
            break;
    }

    TH0=15536/256;
    TL0=15536%256;
}
```