



|  |
| --- |
| **计算机科学学院**  **《物联网项目案例》**  **物联网智能节点**  **设**  **计**  **报**  **告** |

项目组成员：何润

潘俊宏

肖粮

班级：2016级2班

指导教师：郭荣佐

项目评语：

项目得分：

2019年5月

目 录

[摘 要 III](#_Toc621)

[绪论 1](#_Toc29551)

[1.1 课题的开发背景 1](#_Toc15997)

[1.2 意义 1](#_Toc28626)

[1.3 国内外现状与发展趋势 2](#_Toc13315)

[1.4 内容安排 4](#_Toc7360)

[第二章 系统总体设计方案 5](#_Toc9903)

[2.1 主要设计内容与功能 5](#_Toc2892)

[2.2 方案论证与选择 6](#_Toc25582)

[2.3 系统总体方案设计 8](#_Toc24029)

[2.4 系统原理图 8](#_Toc25502)

[2.5 本章小结 10](#_Toc2479)

[第三章 系统控制芯片分析 11](#_Toc22446)

[3.1 STC89C51单片机介绍 11](#_Toc2569)

[3.1.1 主要功能、性能参数 11](#_Toc18207)

[3.2 单片机最小系统 12](#_Toc16483)

[3.2.1 时钟电路 13](#_Toc7819)

[3.2.2 复位电路 13](#_Toc17656)

[3.3 本章小结 15](#_Toc32086)

[第四章 系统硬件设计 16](#_Toc14740)

[4.1 总体设计及功能描述 16](#_Toc27000)

[4.2 各功能模块硬件设计 17](#_Toc9833)

[4.2.1 单片机控制模块的设计 17](#_Toc23152)

[4.2.2 显示电路设计 19](#_Toc19386)

[4.2.3 直流电机驱动电路设计 21](#_Toc9709)

[4.2.4 报警电路 21](#_Toc19089)

[4.2.5 按键模块设计 23](#_Toc10635)

[4.3 本章小结 25](#_Toc3194)

[第五章 软件设计、仿真与调试 32](#_Toc24745)

[5.1 软件设计 32](#_Toc9216)

5.2 仿真

5.3 调试

5.4 系统集成

[结 论 33](#_Toc27320)

[总 结 34](#_Toc10247)

[附录1 系统设计原理图 36](#_Toc13933)

[附录2 元件清单 37](#_Toc25255)

附件3 实物图

[附录4 系统源程序 38](#_Toc4056)

# 摘要

## 中文摘要

城市拥堵现象近年来日益严重，为了提高道路的通行效率，有必要设计一个远程动态控制交通灯系统。本系统以51单片机为控制核心，通过HC05模块蓝牙控制交通信号灯系统，通过设计动态适应算法以达到交通信号灯动态控制。该系统能进行远程控制指令、手/自动运行方式切换，以应对日间繁忙、日间正常、夜间车量以及紧急情况等多种交通状况。该系统为城市交通管理提出切实可行的控制方案，可作为城市交通统一调配和调度的探索案例。

关键词：交通灯，单片机

## ABSTRACT

Urban congestion has become more and more serious in recent years. In order to improve road traffic efficiency, it is necessary to design a remote dynamic control traffic light system. The system uses 51 single-chip microcomputer as the control core, and controls the traffic signal system through the HC05 module Bluetooth, and designs the dynamic adaptation algorithm to achieve the dynamic control of the traffic signal. The system can be used to switch between remote control commands and hand/automatic operation modes to cope with various traffic conditions such as busy daytime, normal daytime, nighttime traffic and emergency situations. The system proposes a practical and feasible control scheme for urban traffic management, which can be used as an exploration case for unified deployment and dispatch of urban traffic.

Keywords: traffic light，SCM

# 绪论

## 1.1课题的开发背景

当今，红绿灯安装在各个道口上，已经成为疏导交通车辆最常见和最有效的手段。但这一技术在19世纪就已出现了。1858年，在英国伦敦主要街头安装了以燃煤气为光源的红，蓝两色的机械扳手式信号灯，用以指挥马车通行。这是世界上最早的交通信号灯。1868年，英国机械工程师纳伊特在伦敦威斯敏斯特区的议会大厦前的广场上，安装了世界上最早的煤气红绿灯。它由红绿两以旋转式方形玻璃提灯组成，红色表示“停止”，绿色表示“注意”。1869年1月2日，煤气灯爆炸，使警察受伤，遂被取消。电气启动的红绿灯出现在美国，这种红绿灯由红绿黄三色圆形的投光器组成，1914年始安装于纽约市5号大街的一座高塔上。红灯亮表示“停止”，绿灯亮表示“通行”。1918年，又出现了带控制的红绿灯和红外线红绿灯。带控制的红绿灯，一种是把压力探测器安在地下，车辆一接近红灯便变为绿灯；另一种是用扩音器来启动红绿灯，司机遇红灯时按一下嗽叭，就使红灯变为绿灯。红外线红绿灯当行人踏上对压力敏感的路面时，它就能察觉到有人要过马路。红外光束能把信号灯的红灯延长一段时间，推迟汽车放行，以免发生交通事故。信号灯的出现，使交通得以有效管制，对于疏导交通流量、提高道路通行能力，减少交通事故有明显效果。1968年，联合国《道路交通和道路标志信号协定》对各种信号灯的含义作了规定。绿灯是通行信号，面对绿灯的车辆可以直行，左转弯和右转弯，除非另一种标志禁止某一种转向。左右转弯车辆都必须让合法地正在路口内行驶的车辆和过人行横道的行人优先通行。红灯是禁行信号，面对红灯的车辆必须在交叉路口的停车线后停车。黄灯是警告信号，面对黄灯的车辆不能越过停车线，但车辆已十分接近停车线而不能安全停车时可以进入交叉路口

电子技术的飞速发展，给古老的锁具生产带来了巨大的变革，现代的电子技术与机械技术相结合，产生了一大批先进的电子类产品。尤其是单片机的发展异常迅速。由于单片机的特殊结构形式，在某些应用领域中，它承担了一些通用的微型计算机无法完成的工作，它是一种高性能，低价格的处理器。集成度高，体积小，可靠性高，控制功能强，电压低。由于单片机具有这些特点，在人类的生活应用中得以十分广泛。

## 1.2意义

当前，大量的信号灯电路正向着数字化、小功率、多样化、方便人、车、路三者关系的协调，多值化方向发展随着社会经济的发展，城市交通问题越来越引起人们的关注。这些城市纷纷修建城市高速道路，在高速道路建设完成初期，它们也曾有效地改善了交通状况。然而，随着交通量的快速增长和缺乏对高速道路的系统研究和控制，高速道路没有充分发挥出预期的作用。而城市高速道路在构造上的特点，也决定了城市高速道路的交通状况必然受高速道路与普通道路耦合处交通状况的制约。人、车、路三者关系的协调，已成为交通管理部门需要解决的重要问题之一。

城市交通控制系统是用于城市交通数据检测、交通信号灯控制与交通疏导的计算机综合管理系统，它是现代城市交通监控指挥系统中最重要的组成部分。

## 1.3国内外现状与发展趋势

随着各种交通工具的发展和交通指挥的需要，第一盏名副其实的三色灯(红、黄、绿三种标志)于1918年诞生。它是三色圆形四面投影器，被安装在纽约市五号街的一座高塔上，由于它的诞生，使城市交通大为改善。

黄色信号灯的发明者是我国的胡汝鼎，他怀着“科学救国”的抱负到美国深造，在大发明家爱迪生为董事长的美国通用电器公司任职员。一天，他站在繁华的十字路口等待绿灯信号，当他看到红灯而正要过去时，一辆转弯的汽车呼地一声擦身而过，吓了他一身冷汗。回到宿舍，他反复琢磨，终于想到在红、绿灯中间再加上一个黄色信号灯，提醒人们注意危险。他的建议立即得到有关方面的肯定。于是红、黄、绿三色信号灯即以一个完整的指挥信号家族，遍及全世界陆、海、空交通领域了。

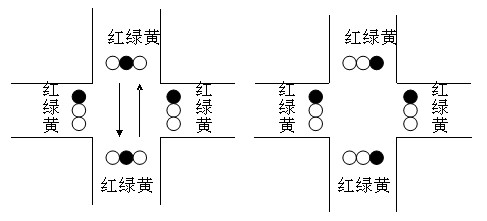
## 1.4内容安排

1. 1～2周，熟悉课题，收集材料，作开题报告；
2. 3～5周，展开整个毕业设计流程，设计系统及程序功能，完整设计控制思想；
3. 6～8周，编程及系统调试，进一步完善控制系统；
4. 9～10周，对毕业设计进行改善，指导学生完成设计报告；
5. 11周，评阅，答辩

# 第二章 系统总体设计方案

## 2.1 主要设计内容与功能

设在十字路口，分为东西向和南北向，在任一时刻只有一个方向通行，另一方向禁行，持续一定时间，经过短暂的过渡时间，将通行禁行方向对换。其具体状态如下图所示。说明：黑色表示亮，白色表示灭。交通状态从状态1开始变换，直至状态6然后循环至状1，周而复始，即如图（图2-1）所示：直至状态6然后循环至状态1，通过具体的路口交通灯状态的演示分析我们可以把这四个状态归纳如下：

****

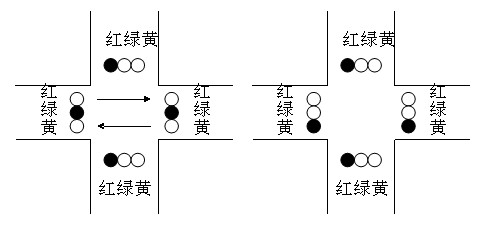
****

图2-1交通状态

东西方向红灯灭，同时绿灯亮，南北方向黄灯灭，同时红灯亮，倒计时20秒。此状态下，东西向禁止通行，南北向允许通行。

东西方向绿灯灭，同时黄灯亮，南北方向红灯亮，倒计时5秒。此状态下，除了已经正在通行中的其他所以车辆都需等待状态转换。

南北方向红灯灭，同时绿灯亮，东西方向黄灯灭，同时红灯亮，倒计时30秒。此状态下，东西向允许通行，南北向禁止通行。

南北方向绿灯灭，同时黄灯亮，东西方向红灯亮，倒计时5秒。此状态下，除了已经正在通行中的其他所以车辆都需等待状态转换。

下面我们可以用图表表示灯状态和行止状态的关系如下：

表2-1交通状态及红绿灯状态

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 状态1 | 状态3 | 状态4 | 状态6 |
| 东西向 | 禁行 | 等待变换 | 通行 | 等待变换 |
| 南北向 | 通行 | 等待变换 | 禁行 | 等待变换 |
| 东西红灯 | 1 | 1 | 0 | 0 |
| 东西黄灯 | 0 | 0 | 0 | 1 |
| 东西绿灯 | 0 | 0 | 1 | 0 |
| 南北红灯 | 0 | 0 | 1 | 1 |
| 南北绿灯 | 1 | 0 | 0 | 0 |
| 南北黄灯 | 0 | 1 | 0 | 0 |

东西南北四个路口均有红绿黄3灯和数码显示管4个，在任一个路口，遇红灯禁止通行，转绿灯允许通行，之后黄灯亮警告行止状态将变换。状态及红绿灯状态如表2-1所示。说明：0表示灭，1表示亮。

本设计能模拟基本的交通控制系统，用红绿黄灯表示禁行，通行和等待的信号发生，还能进行倒计时显示。按键可以控制禁行、深夜模式、复位、东西通行、南北通行、时间加、时间减、切换等功能。共四个二位阴极数码管，东南西北各一个显示时间，四个数码管的阴极都接到STC89C51的P1口，阳极接到74HC245芯片上，通过P0口控制74HC245芯片，起到驱动放大作用。共12个发光二极管，四个路口每个路口各有一个红（禁行）、黄（警告）发光二极管，四个路口的二极管接到P2口，按键接P3口。

本设计能模拟基本的交通控制系统，用红绿黄灯表示禁行，通行和等待的信号发生，还能进行倒计时显示。按键可以控制禁行、深夜模式、复位、东西通行、南北通行、时间加、时间减、切换等功能。共四个二位阴极数码管，东南西北各一个显示时间，四个数码管的阴极都接到STC89C51的P1口，阳极接到74HC245芯片上，通过P0口控制74HC245芯片，起到驱动放大作用。共12个发光二极管，四个路口每个路口各有一个红（禁行）、黄（警告）发光二极管，四个路口的二极管接到P2口，按键接P3口。

## 2.2 方案论证与选择

### 2.2.1 方案一

### 2.2.2 方案二

### 2.2.3 方案论证与选择

## 2.3 系统总体方案设计

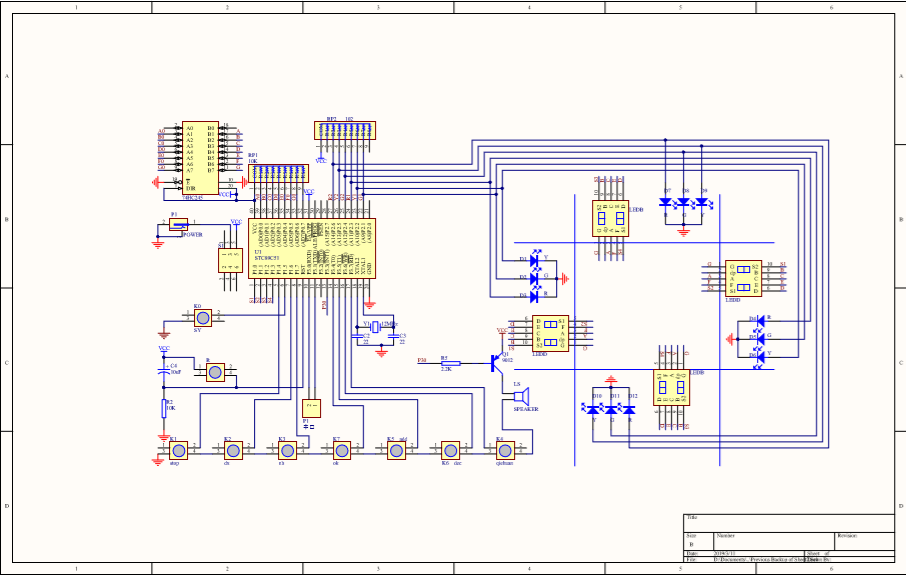
十字路口车辆穿梭，行人熙攘，车行车道，人行人道，有条不紊。那么靠什么来实现这井然秩序呢？靠的就是交通信号灯的自动指挥系统。交通信号灯控制方式很多。本系统采用STC89C51单片机以及单片机最小系统和74HC245驱动电路以及外围的按键和数码管显示等部件，设计一个基于单片机的交通灯设计。设计通过两位一体共阴极数码管显示，并能通过按键对定时进行设置。本系统实用性强、操作简单、扩展功能强。

在相同的时间里提高通车的质量、效率。并能在高峰期根据实际状况结合方程式控制按钮来调整主次干道的通车时间，降低交通拥挤堵塞现象。并使交通控制系统具有紧急控制，使救护车、救护车通过时， 使两个方向均亮红灯，救护车和消防车通过后，恢复原来状态，增加对出现特殊情况的处理能力。

本单片机控制交通灯系统，可用单片机直接控制信号灯的状态变化，基本上可以指挥交通的具体通行，当然，接入LED数码管就可以显示倒计时以提醒行使者，更具人性化。本系统在此基础上，单片机对此进行具体处理，及时调整控制指挥。如图（图2-2）所示：

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。在此过程中还要实时捕捉违规检测和紧急按键信号，以达到对异常状态进行实时控制的目的。急停按键和违规检测随时调用中断。

## 2.4 系统原理图



## 2.5 本章小结

单片机

红黄绿信号灯

8段LED数码管

复位电路

最小系统外围接口

按键控制

驱动显示

单片机

系统的总体框图

据此，本设计系统以单片机为控制核心，连接成最小系统，和按键设置模块等产生输入，信号灯状态模块，LED倒计时模块和接受输出。系统的总体框图如上所示。

# 第三章 系统控制芯片分析

## 3.1 STC89C51单片机介绍

### 3.1.1 主要功能、性能参数

89C51-III 单片机学习板是一款基于 8 位单片机处理芯片 STC89C52RC 的系统。其功能强 大，可以实现单片机开发的多种要求，学习、开发者可以根据需要选配多种常用模块，达到 实验及教学的目的。 89C51-III 单片机学习板功能强大，具有报警，跑马灯、串行通信（max232）、 段码液晶 （msm0801LCD）和字符液晶显示(LCD1602)、电机控制(L298)、A/D 转换(TLC2543)、D/A 转换 (TLC5615)、温度采集(DS1602)、数字信号合成(AD9851)、实时时钟电路(DS1302)、4—20mA 输出、PWM 输出（UC3842） 、红外检测（KSM-603LM）控制等十七种功能，供学习者学习开发使 用。89C51-III 单片机学习板采用的芯片都是常用芯片，使学习者对常用电子产品进一步学习 理解。

## 3.2 单片机最小系统

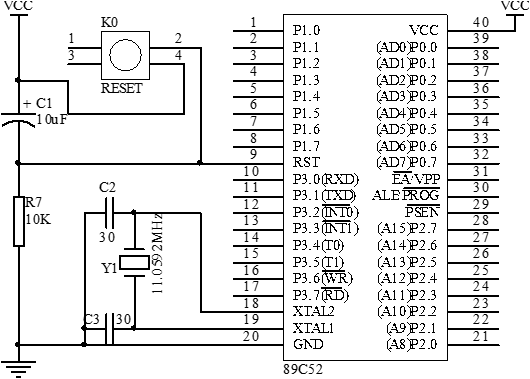
### 3.2.1 时钟电路

STC89C51单片机的时钟信号通常有两种方式产生：一是内部时钟方式，二是外部时钟方式。在单片机内部有一振荡电路，只要在单片机的XTAL1和XTAL2引脚外接石英晶体（简称晶振），就构成了自激振荡器并在单片机内部产生时钟脉冲信号。图中电容C1和C2的作用是稳定频率和快速起振，电容值在5-30pF，典型值为30pF。晶振CYS的振荡频率范围在1.2-12MHz间选择，典型值为12MHz和11.0592MHz。

当在STC89C51单片机的RST引脚引入高电平并保持2个机器周期时，单片机内部就执行复位操作（若该引脚持续保持高电平，单片机就处于循环复位状态）。

### 3.2.2 复位电路

复位电路通常采用上电自动复位和按钮复位两种方式。最简单的上电自动复位电路中上电自动复位是通过外部复位电路的电容充电来实现的。只要Vcc的上升时间不超过1ms,就可以实现自动上电复位。时钟频率用6MHZ时C取22uF,R取1KΩ。除了上电复位外，有时还需要按键手动复位。本设计就是用的按键手动复位。按键手动复位有电平方式和脉冲方式两种。其中电平复位是通过RST端经过电阻与电源Vcc接通而实现的。系统图如图3-2所示



单片机最小系统原理图

## 3.3 本章小结

单片机是在集成电路芯片上集成了各种元件的微型计算机，这些元件包括中央处理器CPU、数据存储器RAM、程序存储器ROM、定时/计数器、中断系统、时钟部件的集成和I/O接口电路。由于单片机具有体积小、价格低、可靠性高、开发应用方便等特点，因此在现代电子技术和工业领域应用较为广泛，在智能仪表中单片机是应用最多、最活跃的领域之一。在控制领域中,现如今人们更注意计算机的底成本、小体积、运行的可靠性和控制的灵活性。在各类仪器、仪表中引入单片机，使仪器仪表智能化，提高测试的自动化程度和精度，提高计算机的运算速度，简化仪器仪表的硬件结构，提高其性能价格比。

# 第四章 系统硬件设计

## 4.1 总体设计及功能描述

实现本设计要求的具体功能，可以选用stc89C51单片机及外围器件构成最小控制系统，12个发光二极管分成4组红绿黄三色灯构成信号灯指示模块，8个LED东西南北各两个构成倒计时显示模块，若干按键组成时间设置和模式选择按钮和紧急按钮等。

本系统以单片机为核心，组成一个处理、自动控制为一身的闭环控制系统。系统硬件电路由单片机、状态灯、LED显示、驱动电路、按键等组成。其具体的硬件电路总图如图3-1所示。

其中P0，P1，用于送显LED数码管的型和位，P2用于控制红绿黄发光二极管，XTAL1和XTAL2接入晶振时钟电路，REST引脚接上复位电路，P3用于口按键控制。

## 4.2 各功能模块硬件设计

### 4.2.1 单片机控制模块的设计

本设计能模拟基本的交通控制系统，用红绿黄灯表示禁行，通行和等待的信号发生，还能进行倒计时显示。按键可以控制禁行、深夜模式、复位、东西通行、南北通行、时间加、时间减、切换等功能。共四个二位阴极数码管，东南西北各一个显示时间，四个数码管的阴极都接到STC89C51的P1口，阳极接到74HC245芯片上，通过P0口控制74HC245芯片，起到驱动放大作用。共12个发光二极管，四个路口每个路口各有一个红（禁行）、黄（警告）发光二极管，四个路口的二极管接到P2口，按键接P3口。

### 4.2.2 显示电路设计

74HC245译码器可接受3位二进制加权地址输入（A0, A1和A2），并当使能时，提供8个互斥的低有效输出（Y0至Y7）。74HC245特有3个使能输入端：两个低有效（E1和E2）和一个高有效（E3）。除非E1和E2置低且E3置高，否则74HC138将保持所有输出为高。利用这种复合使能特性，仅需4片74HC245芯片和1个反相器，即可轻松实现并行扩展，组合成为一个1-32（5线到32线）译码器。任选一个低有效使能输入端作为数据输入，而把其余的使能输入端作为选通端，则74HC245亦可充当一个8输出多路分配器，未使用的使能输入端必须保持绑定在各自合适的高有效或低有效状态。

74HC245作用原理于高性能的存贮译码或要求传输延迟时间短的数据传输系统,在 高性能存贮器系统中,用这种译码器可以提高译码系统的效率。将快速赋能电路用于高速存贮器时,译码器的延迟时间和存贮器的赋能时间通常小于存贮器的典型存取时间,这就是说由肖特基钳位的系统译码器所引起的有效系统延迟可以忽略不计。HC138 按照三位二进制输入码和赋能输入条件,从8 个输出端中译出一个 低电平输出。两个低电平有效的赋能输入端和一个高电平有效的赋能输入端减少了扩展所需要的外接门或倒相器,扩展成24 线译码器不需外接门;扩展成32 线译码器,只需要接一个外接倒相器。在解调器应用中,赋能输入端可用作数据输入端。



图3-5 74HC245电路图

设计中将1脚接VCC，19脚接地，整个芯片是输入端输入高则输出端输出高，输入端输入低则输出端输出地，只是相当于驱动作用。

### 4.2.3 直流电机驱动电路设计

### 4.2.4 报警电路

## 4.3 本章小结

# 第五章 软件设计、仿真与调试

## 5.1 软件设计

全部控制程序实际上分为若干模块：键盘设置处理程序，状态灯控制程序，LED显示程序，消抖动延时程序，次状态判断及处理程序，紧停或违规判断程序，中断服务子程序，车流量计数程序，红绿灯时间调整程序等。

整个软件程序方面主要分两大部分：按键处理程序和50ms扫描程序。流程图如图（图4-1）所示。

设置字型码和字位码，完成显示

初始化外部中断

定义状态数组

宏定义

返回while(1)函数

进入while(1)循环

I/O初始化

定义字位码函数

定义共阴极字型编码表

调用显示控制函数void display（）

进入主函数main()

定义函数变量并初始化

定时器0初始化

调用Buzzer()函数

宏定义

I/O初始化

定义共阴极字型编码表

定义函数变量并初始化

定义状态数组

定义字位码函数

进入主函数main()

定时器0初始化

初始化外部中断

进入while(1)循环

调用显示控制函数void display（）

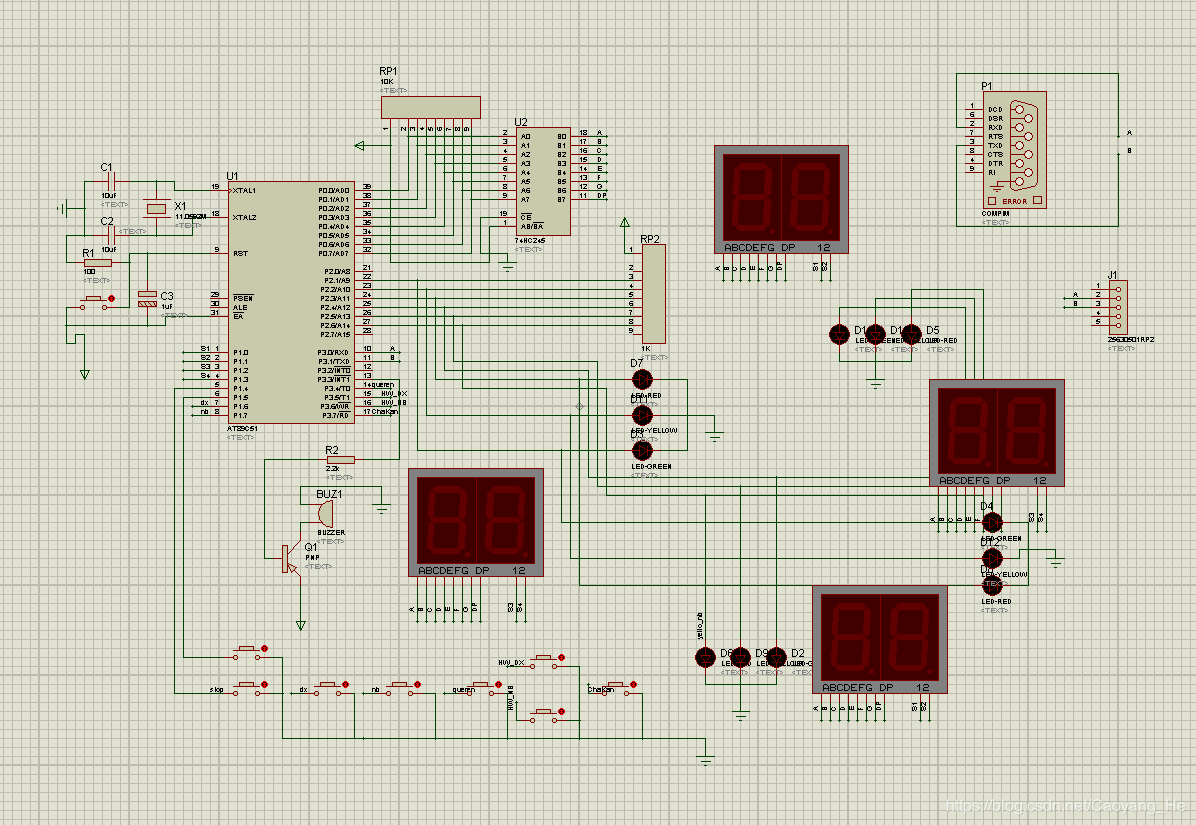
调用Buzzer()函数

设置字型码和字位码，完成显示

返回while(1)函数

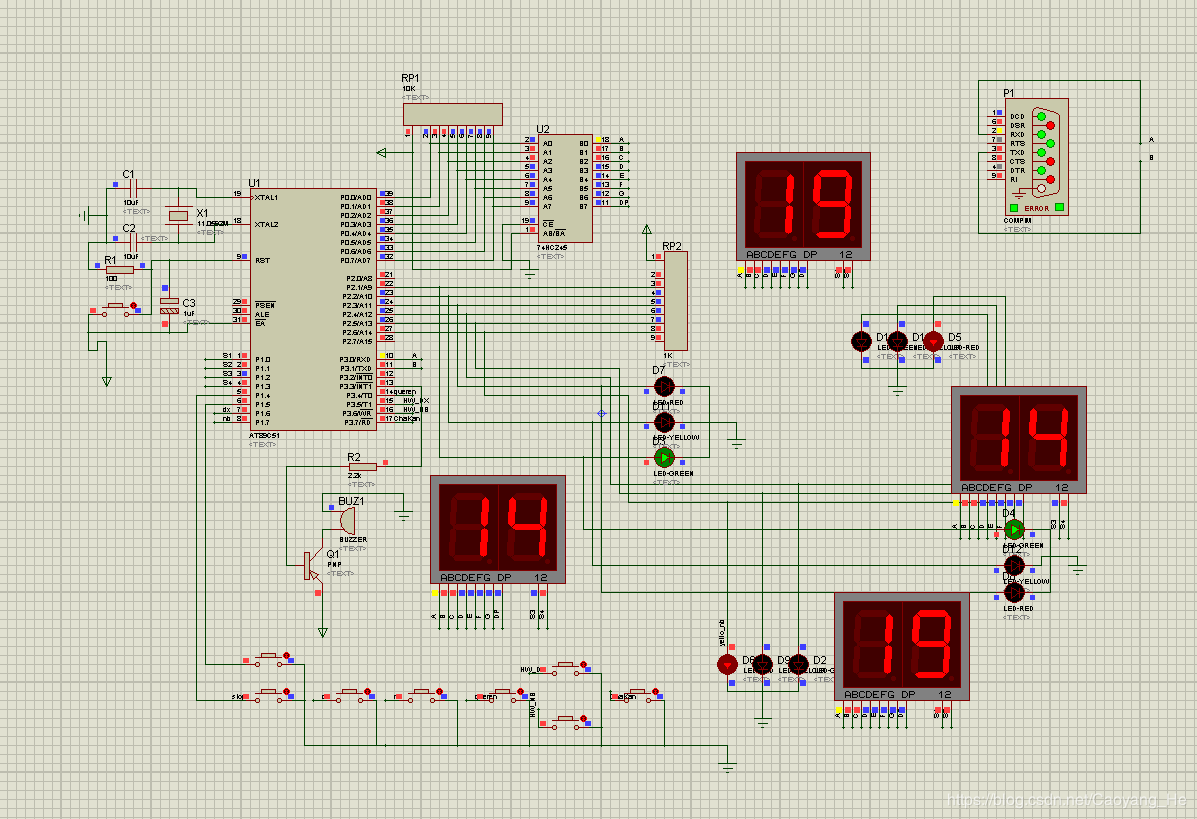
系统总的流程图

## 仿真

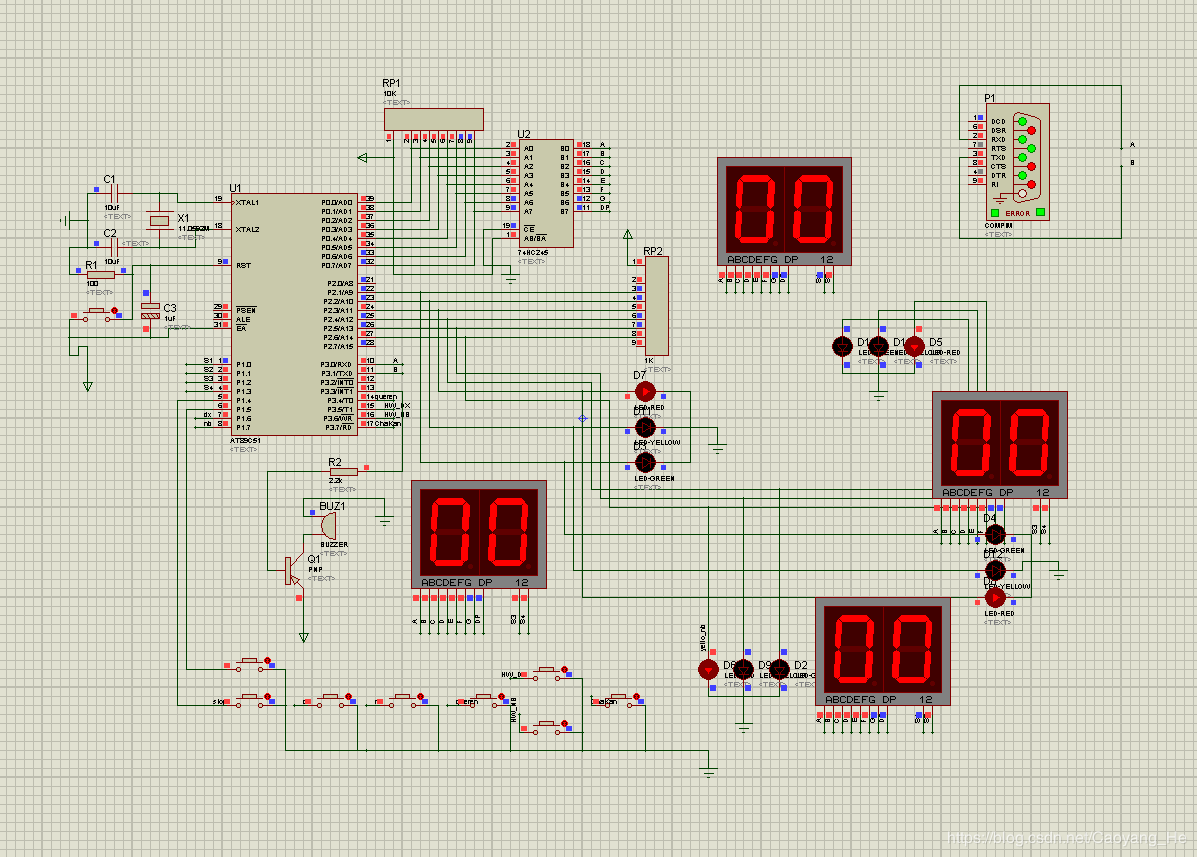


## 调试

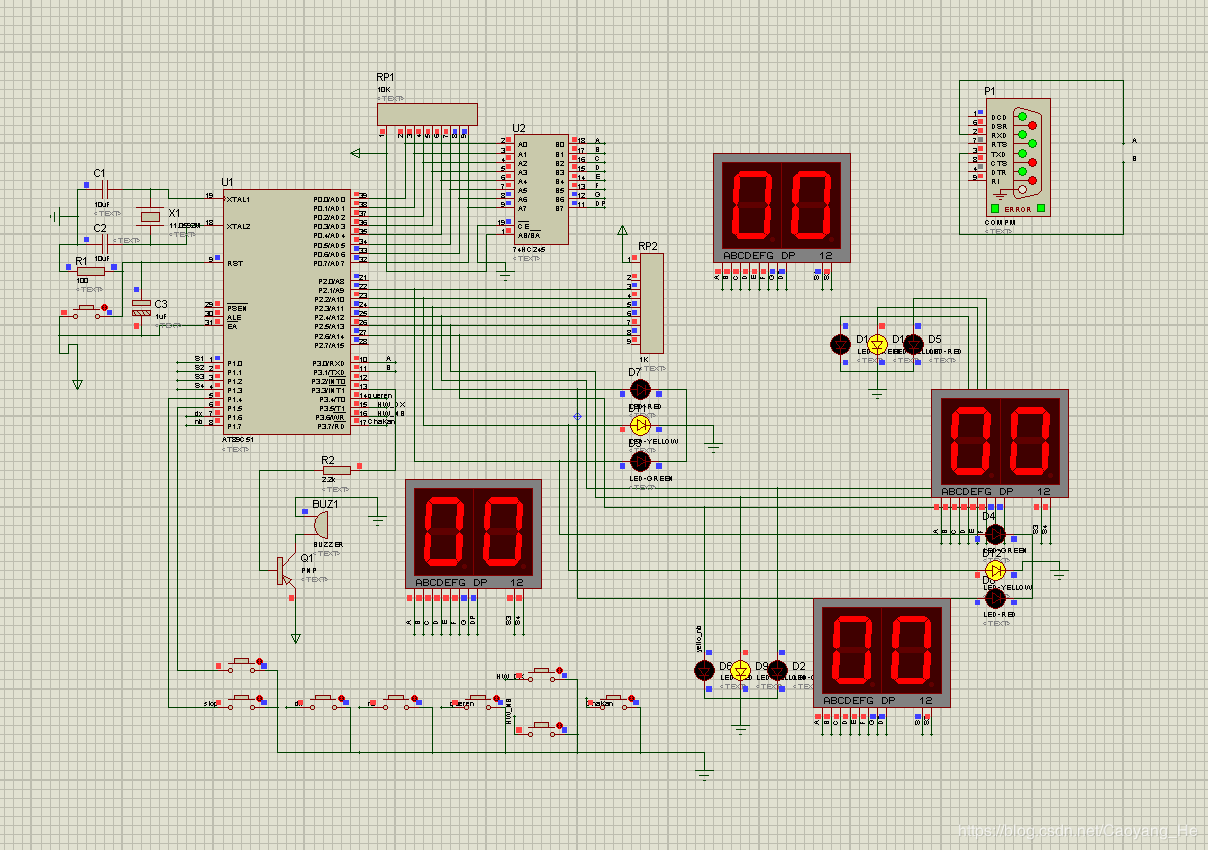
正常运行



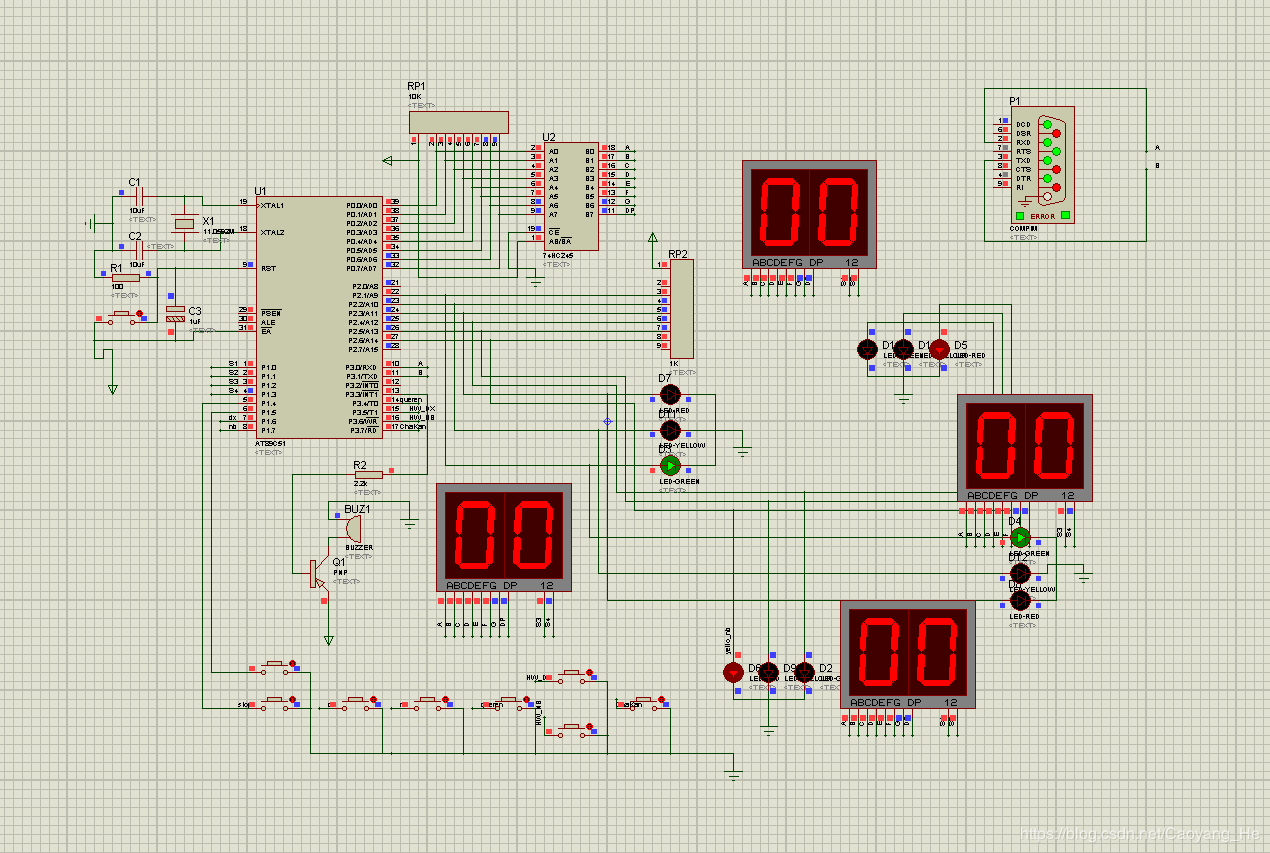
停止模式（红灯全亮）



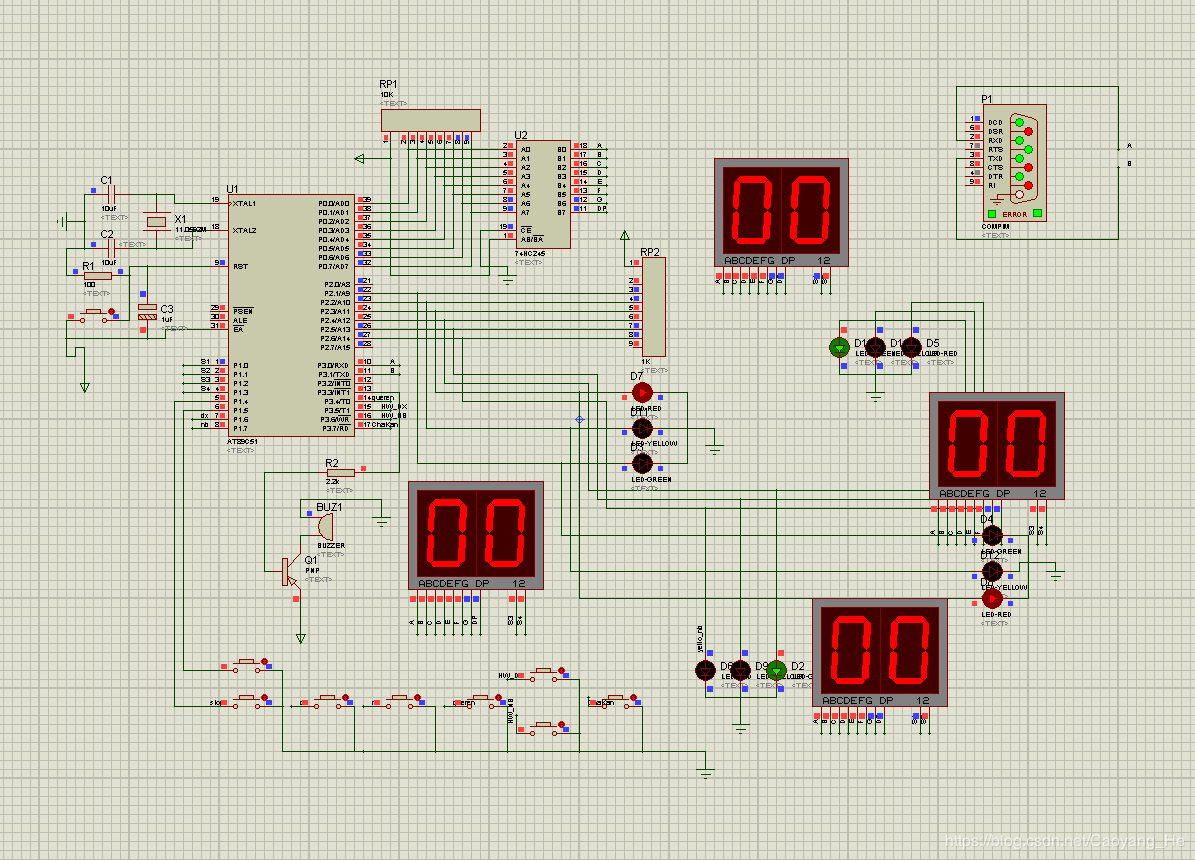
深夜模式（黄灯全亮）



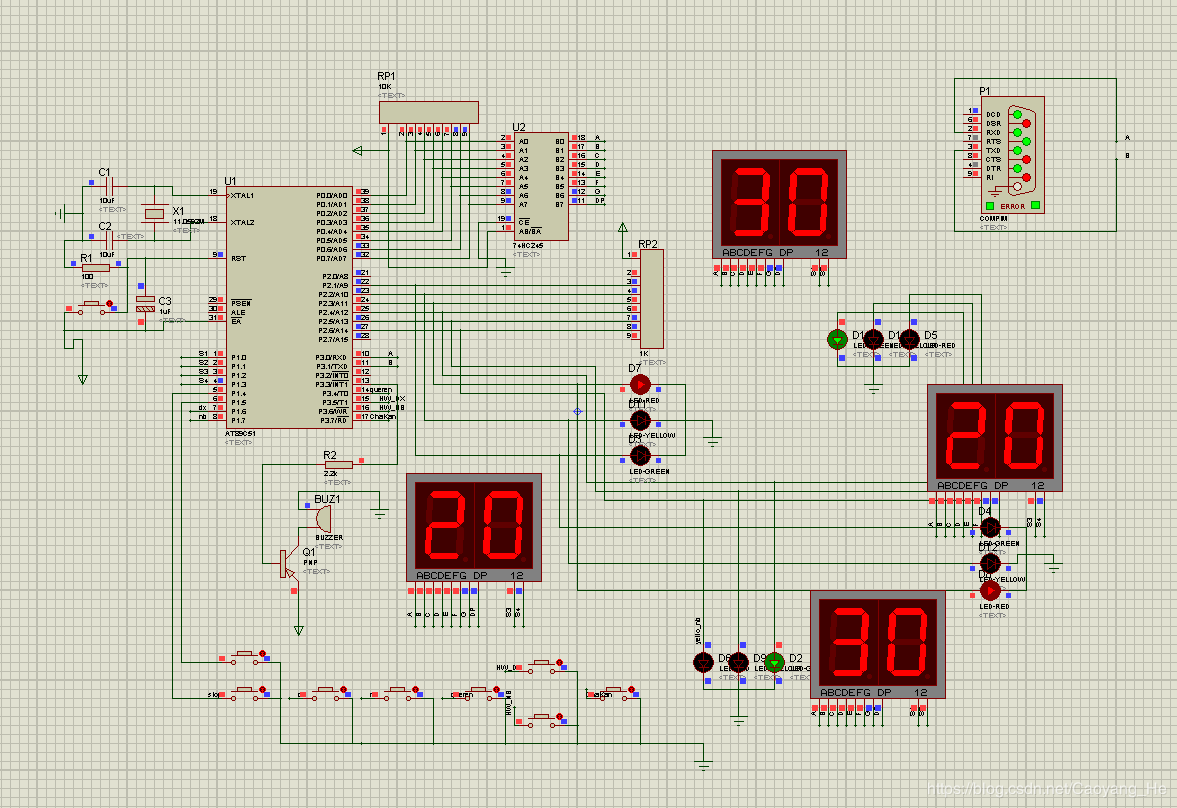
东西通行模式



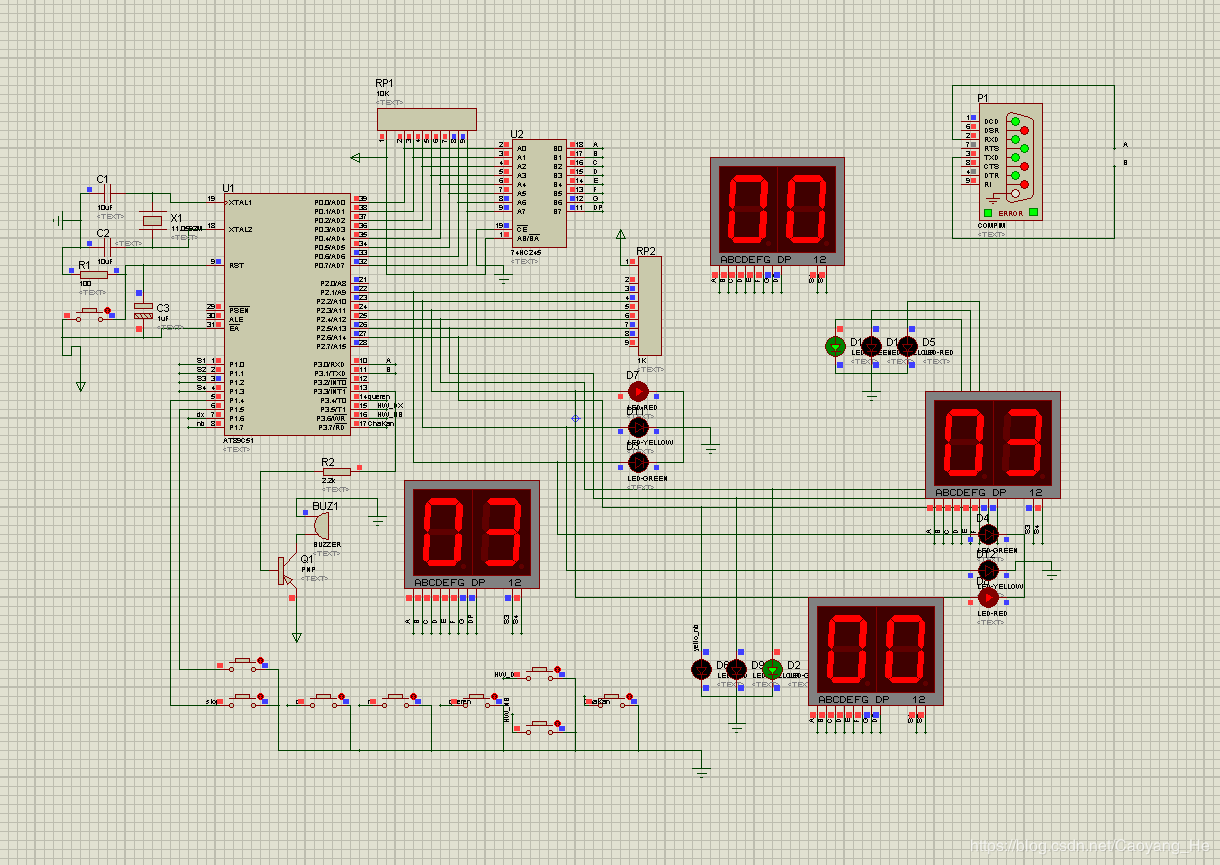
南北通行模式



查看预设的时间



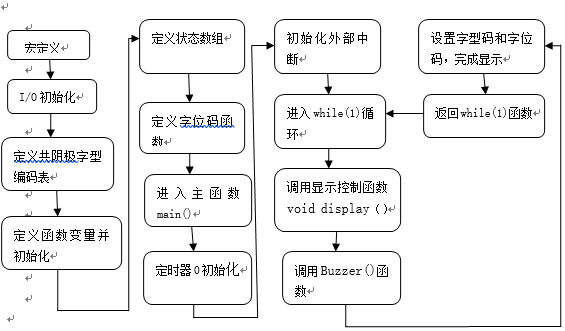
查看车流量

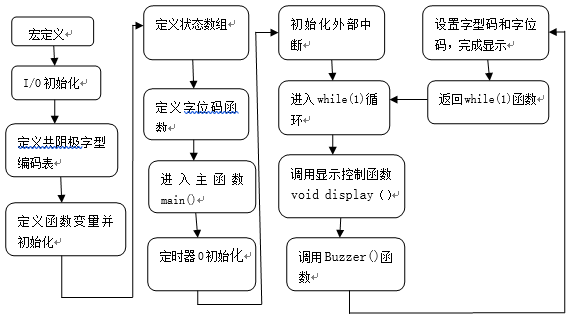


## 系统集成

全部控制程序实际上分为若干模块：键盘设置处理程序，状态灯控制程序，LED显示程序，消抖动延时程序，次状态判断及处理程序，紧停或违规判断程序，中断服务子程序，车流量计数程序，红绿灯时间调整程序等。

整个软件程序方面主要分两大部分：按键处理程序和50ms扫描程序。流程图如图（图4-1）所示。





# 结 论

该系统各个功能均可完美实现

# 总 结

经过这次毕业设计，我们学到了不少东西。归纳起来，主要有以下几点：

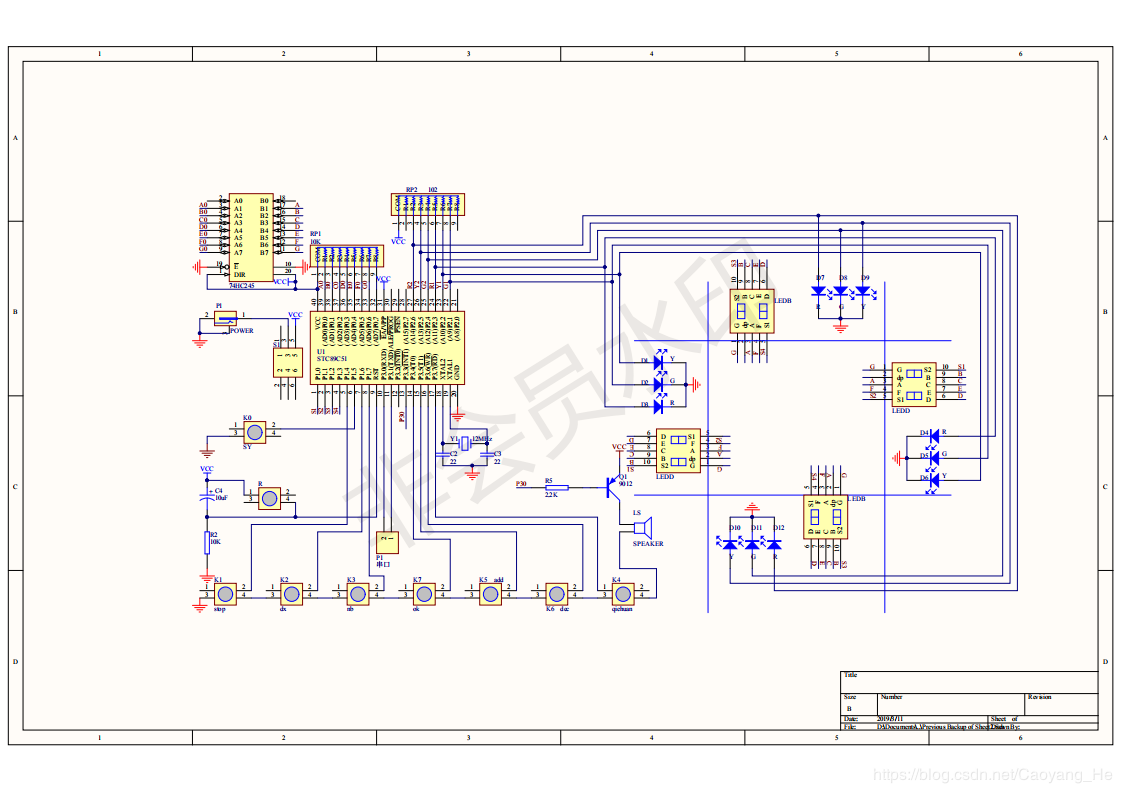
（1）通过这次项目，能将以前所学到的专业知识与实践相联系，将所学到的知识充分运用到本次设计中。同时，也认识到自己知识上不足的地方，体会到了所学理论知识的重要性，知识掌握得越多，设计得就更全面、更顺利、更好。

（2）进一步熟悉了单片机的知识。通过本次设计，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。

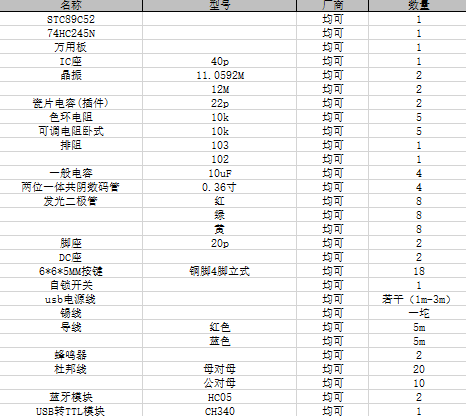
（3）通过本次设计，熟悉了设计一个项目所必经的几个阶段。本次设计从理论研究到硬件原理图设计，从元器件的选择到板的制作，从软件编程到最后的调试过程都由我独立完成。这不仅锻炼了我独立完成设计工作的能力，更重要的是了解了一个电子产品的设计流程，为将来投入工作增加了宝贵的经验，奠定了坚实的基础。

（4）提高了自己查找资料的能力。在设计过程中，碰到了一些暂时无法解决的问题，于是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。

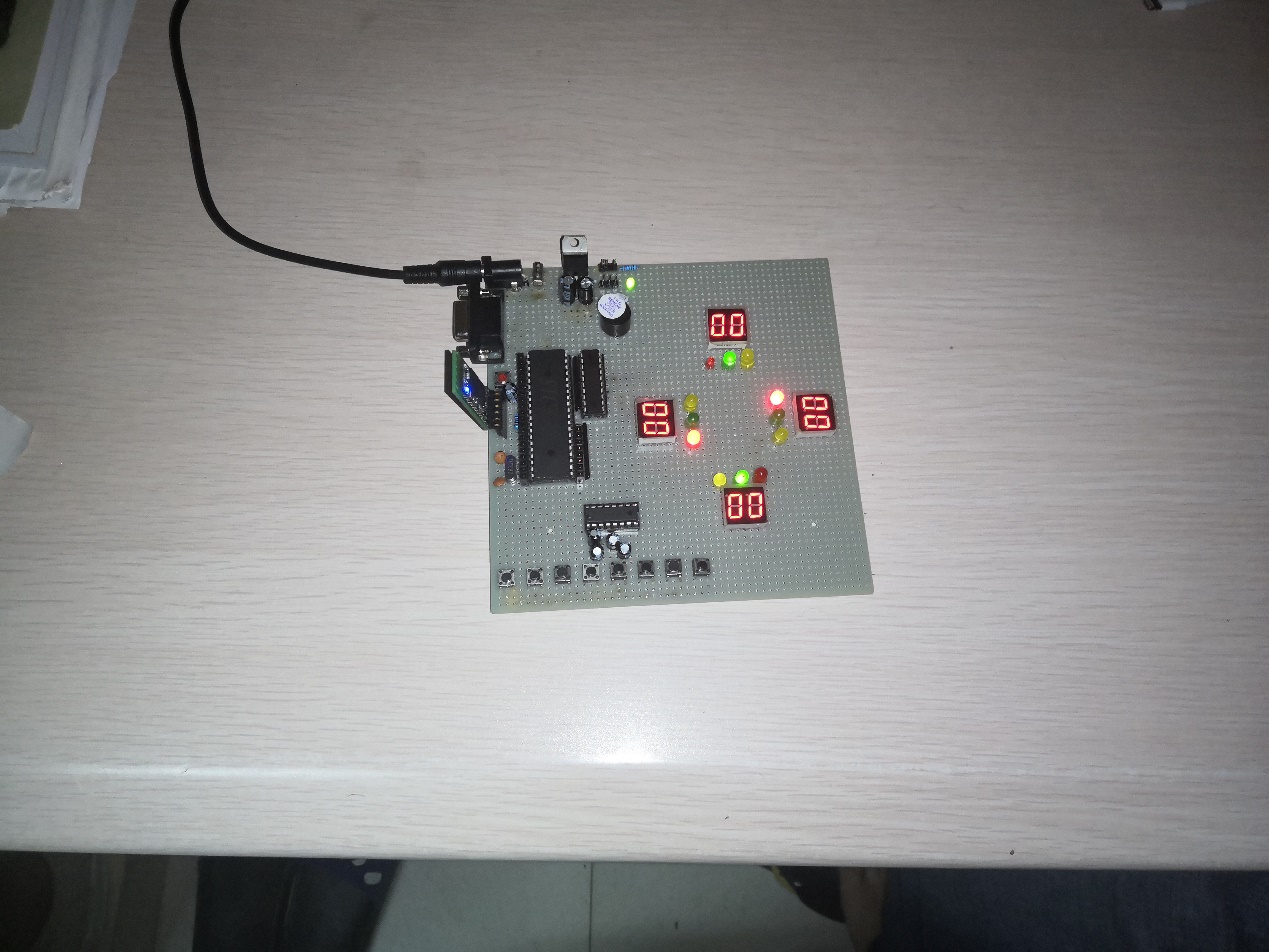
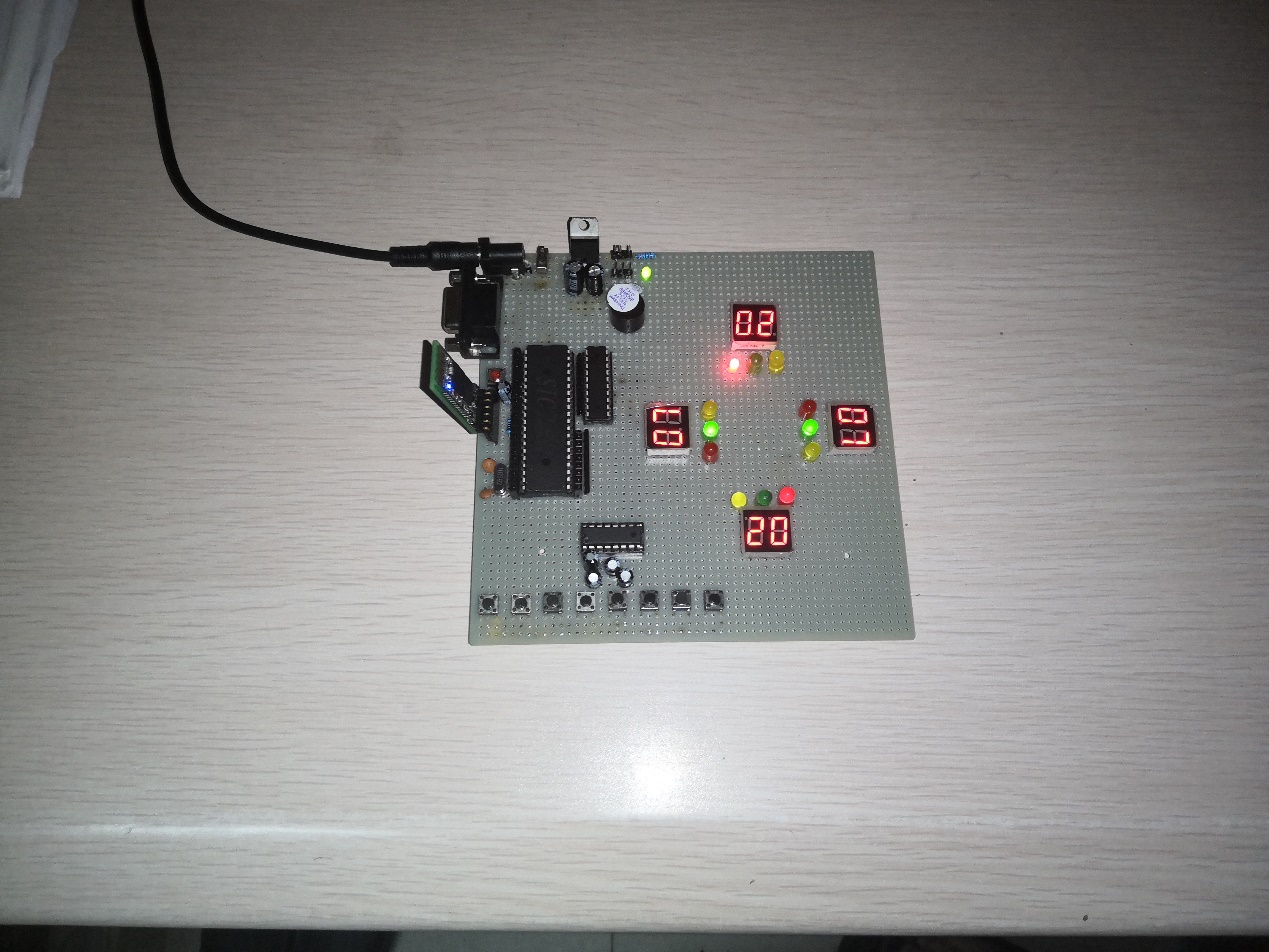
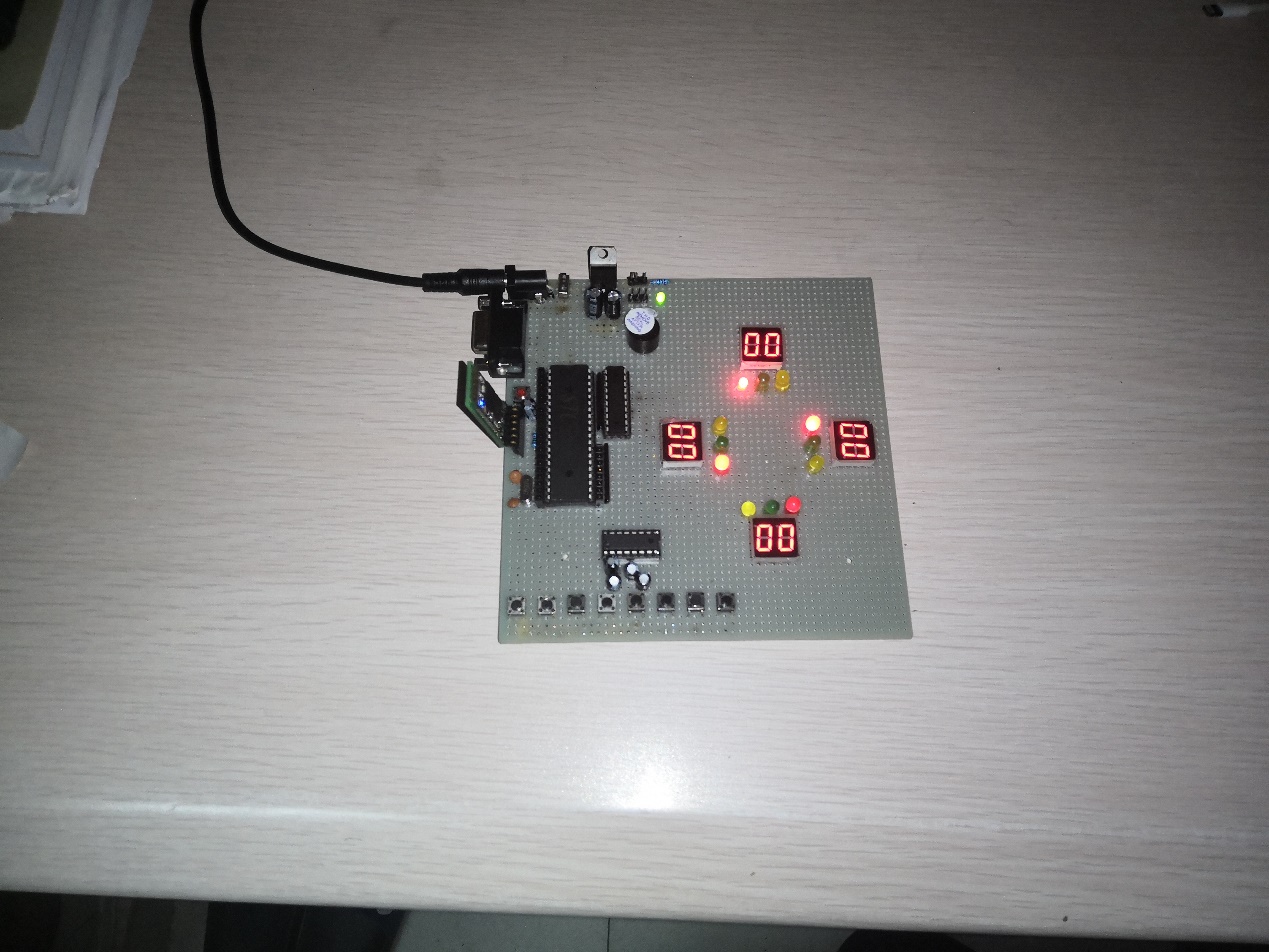
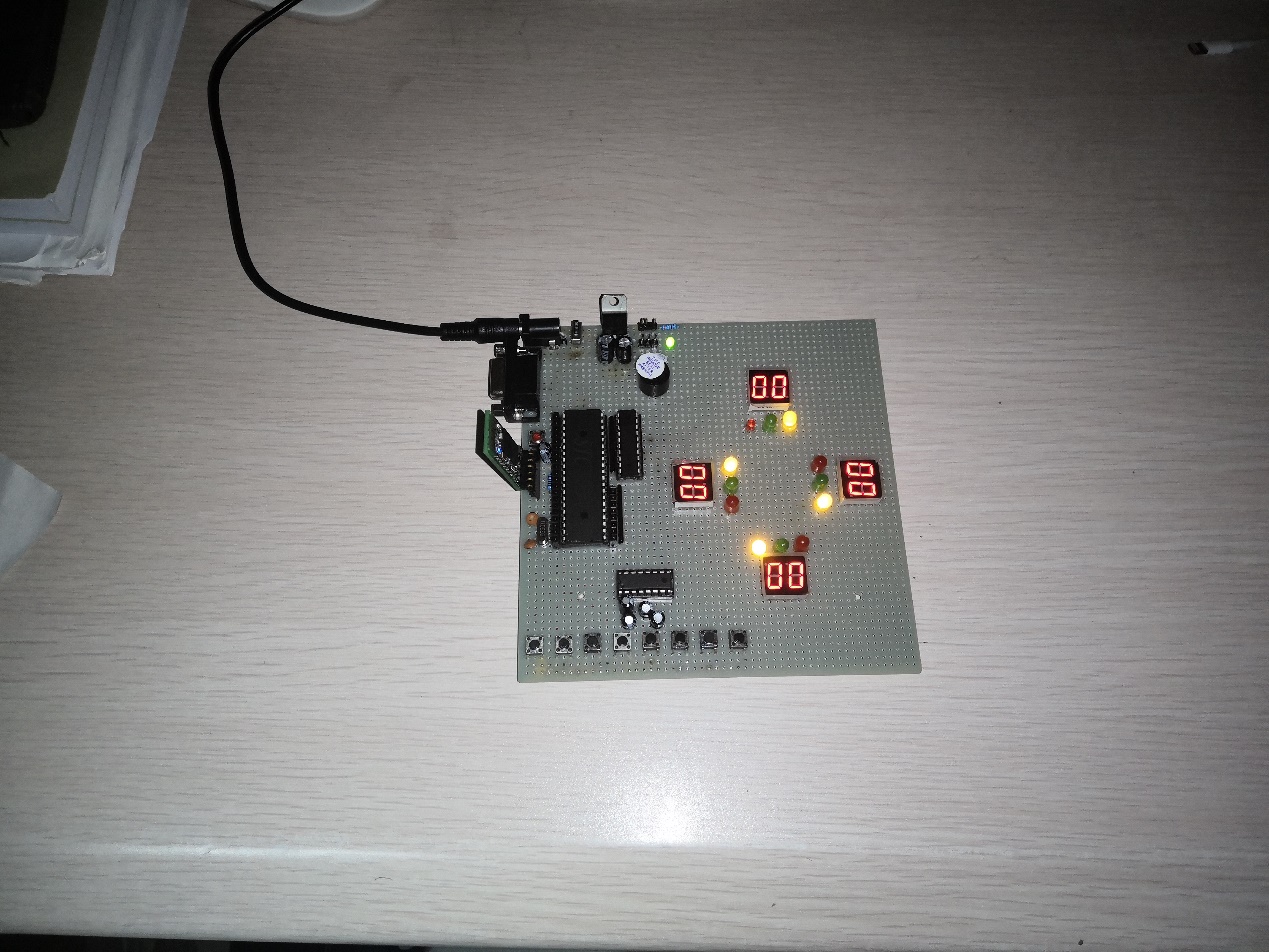
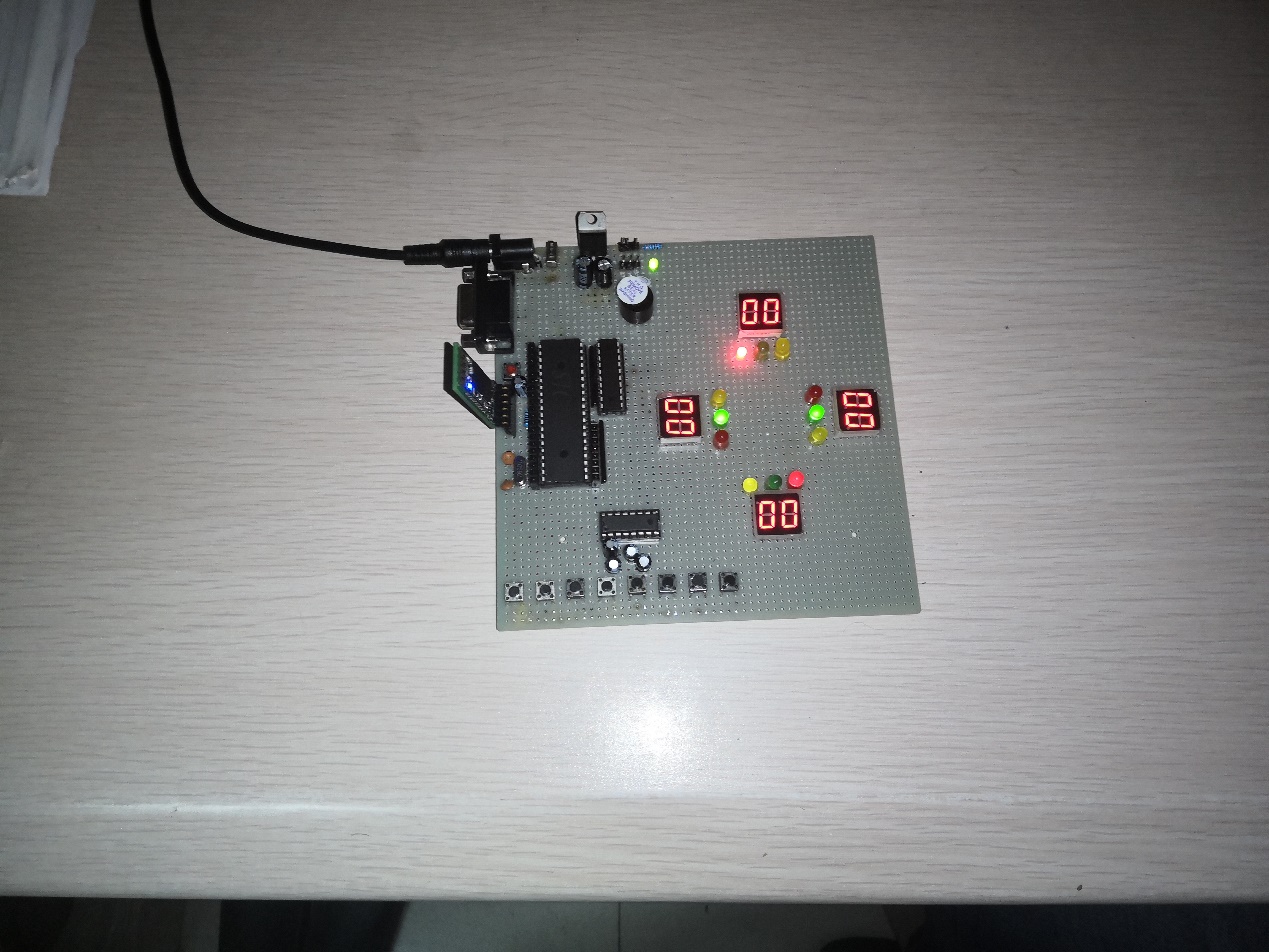
# 附录1 系统设计原理图

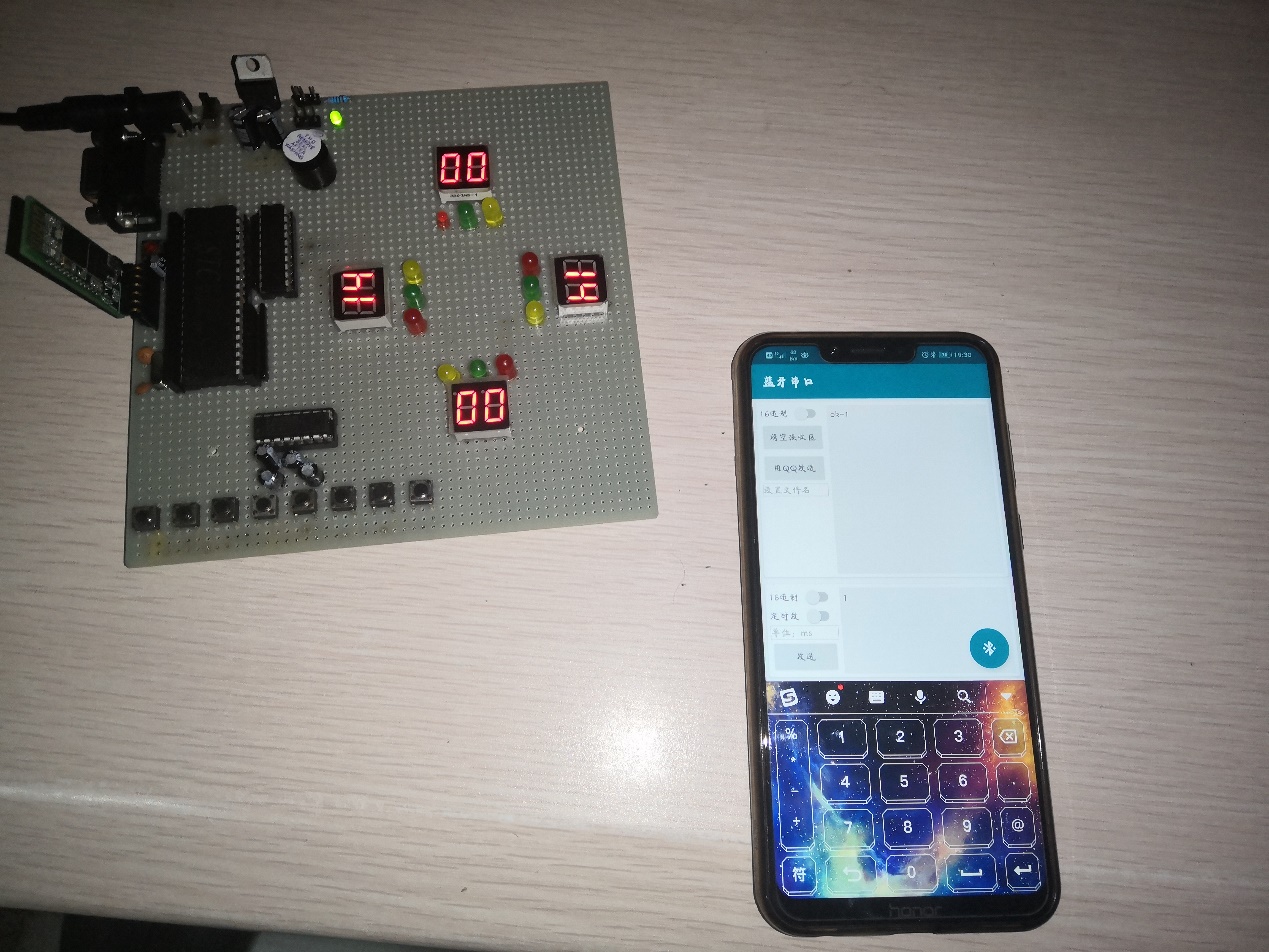
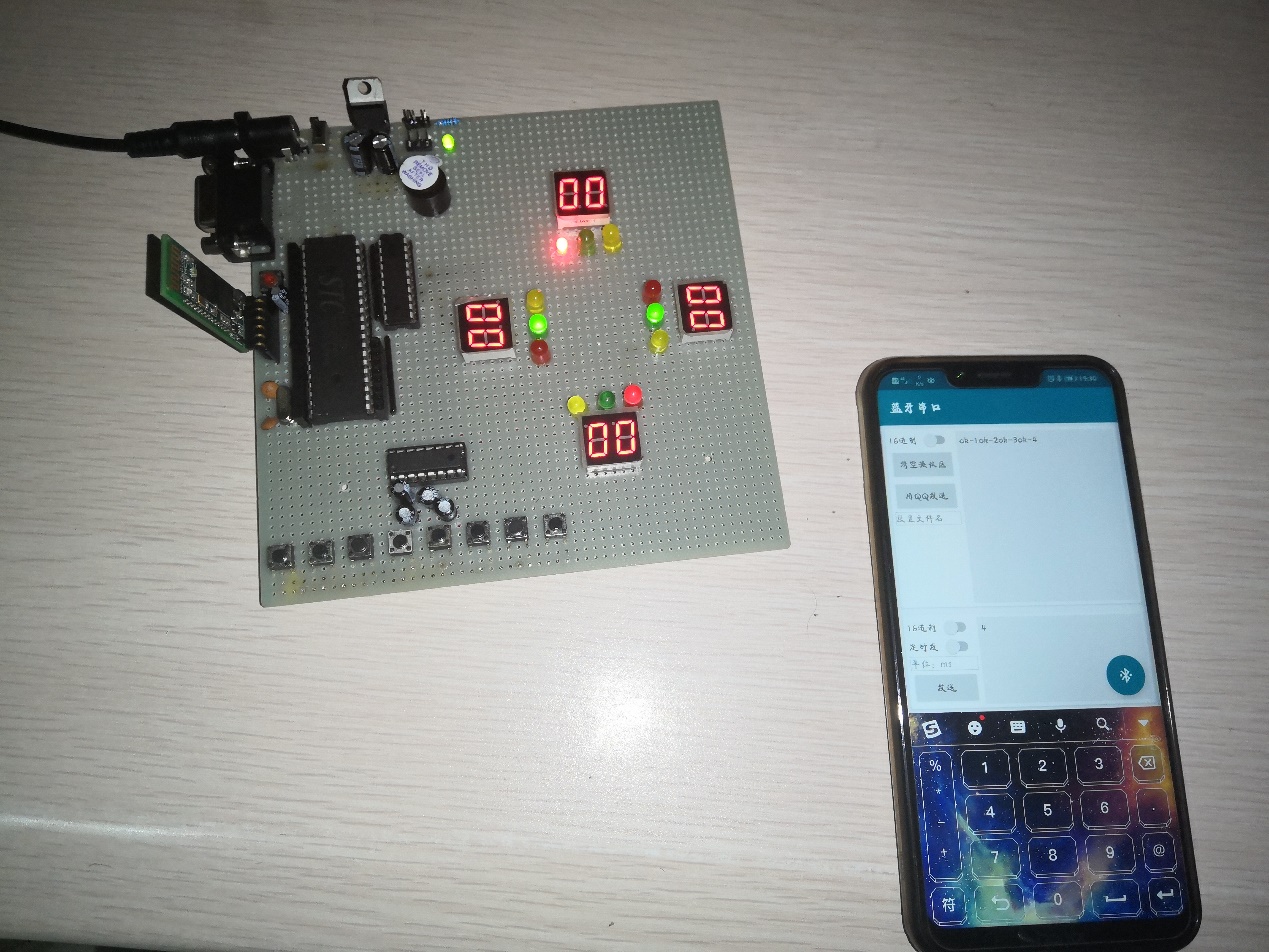
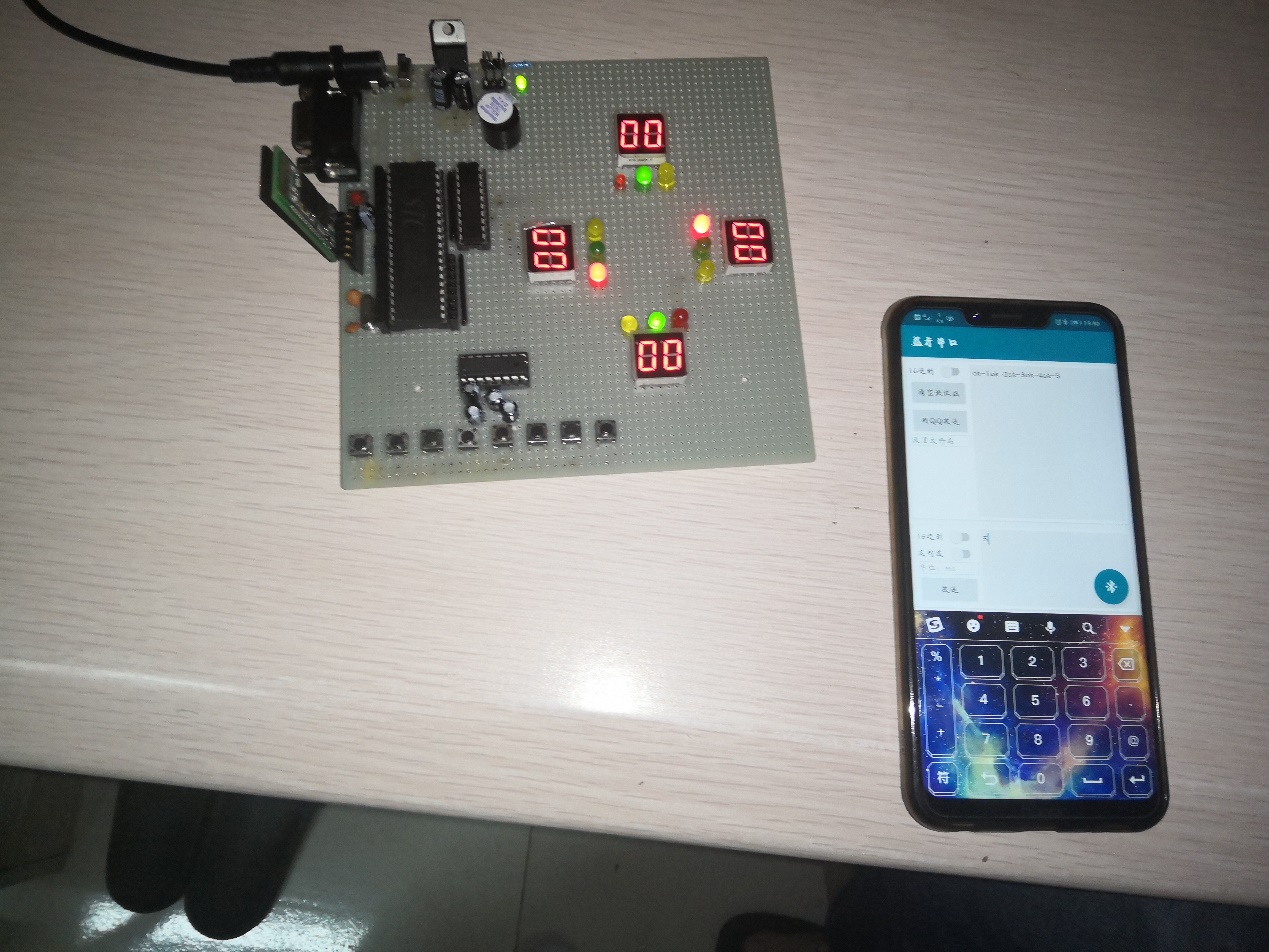


# 附录2 元件清单



# 附件3 实物图





# 附录4 系统源程序

#include <reg52.h> //头文件

#define uchar unsigned char

#define uint unsigned int //宏定义

sfr T2MOD = 0xC9;

uchar data buf[4]; //秒显示的变量

uchar data sec\_dx = 20; //东西数默认

uchar data sec\_nb = 30; //南北默认值

uchar data set\_timedx = 20; //设置东西方向的时间

uchar data set\_timenb = 30; //设置南北方向的时间

uchar data num\_che\_dx = 0; //设置变量用于记录东西通行车量的个数

uchar data num\_che\_nb = 0; //设置变量用于记录南北通行车辆的个数

uchar data xianshi\_fx = 0; //显示方式

int n;

uchar data countt0, countt1; //定时器0中断次数

//定义6组开关

sbit k1 = P1 ^ 5; //夜间模式

sbit k2 = P1 ^ 4; //禁止

sbit k3 = P3 ^ 4; //确认

sbit k4 = P1 ^ 6; //东西通行

sbit k5 = P1 ^ 7; //南北通行

sbit hw\_dx = P3 ^ 5; //使用此引脚接红外探头检测"东西"方向通过的车辆当检测到南北有车辆通行时，该引脚会呈现低电平

sbit hw\_nb = P3 ^ 6; //使用此引脚接红外探头检测"南北"方向通过的车辆当检测到南北有车辆通行时，该引脚会呈现低电平

sbit k6 = P3 ^ 7; //切换方向

sbit Red\_nb = P2 ^ 6; //南北红灯标志

sbit Yellow\_nb = P2 ^ 5; //南北黄灯标志

sbit Green\_nb = P2 ^ 4; //南北绿灯标志

sbit Red\_dx = P2 ^ 3; //东西红灯标志

sbit Yellow\_dx = P2 ^ 2; //东西黄灯标志

sbit Green\_dx = P2 ^ 1; //东西绿灯标志

sbit Buzz = P3 ^ 3;

bit set = 0; //调时方向切换键标志 =1时，南北，=0时，东西

bit dx\_nb = 0; //东西南北控制位

bit shanruo = 0; //闪烁标志位

bit yejian = 0; //夜间黄灯闪烁标志位

uchar code table[11] = {

//共阴极字型码

0x3f, //--0

0x06, //--1

0x5b, //--2

0x4f, //--3

0x66, //--4

0x6d, //--5

0x7d, //--6

0x07, //--7

0x7f, //--8

0x6f, //--9

0x00 //--NULL

};

//函数的声明部分

void delay(int ms); //延时子程序

void key(); //按键扫描子程序

void key\_to1(); //键处理子程序

void key\_to2();

void key\_to3();

void key\_to4();

void key\_to5();

void display(); //显示子程序

void logo(); //开机LOGO

void Buzzer();

//主程序

void main()

{

TMOD = 0X11; //定时器设置

TH1 = 0X3C;

TL1 = 0XB0;

TH0 = 0X3C; //定时器0置初值 0.05S

TL0 = 0XB0;

T2MOD = 0x01; //自动重载

T2CON = 0x30; //T2用做发送接收时钟

TH2 = 0xFF; //9600波特率,11.0592Mhz晶振

TL2 = 0xDC;

RCAP2H = 0xFF;

RCAP2L = 0xDC;

SCON = 0x50; //串口方式1,1个起始位,1个停止位,8位数据，可变波特率

PCON = 0X00; //波特率不加倍

TR2 = 1; //启动T2

ES = 1; //开串口中断

EA = 1; //开总中断

ET0 = 1; //定时器0中断开启

ET1 = 1; //定时器1中断开启

TR0 = 1; //启动定时0

TR1 = 0; //关闭定时1

EX0 = 1; //开外部中断0

EX1 = 1; //开外部中断1

logo(); //开机初始化

P2 = 0Xc3; // 开始默认状态，东西绿灯，南北黄灯

sec\_nb = sec\_dx + 5; //默认南北通行时间比东西多5秒

while (1) //主循环

{

key(); //调用按键扫描程序

display(); //调用显示程序

}

}

void key\_to1() //夜间模式设置函数

{

TR0 = 0; //关定时器

P2 = 0x00;

TR1 = 1;

sec\_dx = 00; //四个方向的时间都为00

sec\_nb = 00;

}

void key\_to2() //禁止通行模式设置函数

{

TR0 = 0; //关定时器

P2 = 0x00; //灭显示

Red\_dx = 1;

Red\_nb = 1; //全部置红灯

TR1 = 0;

sec\_dx = 00; //四个方向的时间都为00

sec\_nb = 00;

}

void key\_to3() //确认设置函数

{

TR0 = 1; //启动定时器0

xianshi\_fx = 0; //设置重新显示

sec\_nb = set\_timenb; //从中断回复，仍显示设置过的数值

sec\_dx = set\_timedx; //显示设置过的时间

TR1 = 0; //关定时器1

if (set == 0) //时间倒时到0时

{

P2 = 0X00; //灭显示

Green\_dx = 1; //东西绿灯亮

Red\_nb = 1; //南北红灯亮

sec\_nb = sec\_dx + 5; //回到初值

}

else

{

P2 = 0x00; //南北绿灯，东西红灯

Green\_nb = 1;

Red\_dx = 1;

sec\_dx = sec\_nb + 5;

}

}

void key\_to4() //东西通行设置函数

{

TR0 = 0; //关定时器0

TR1 = 0; //关定时器1

P2 = 0x00; //灭显示

Green\_dx = 1; //东西方向置绿灯

Red\_nb = 1; //南北方向为红灯

sec\_dx = 00; //四个方向的时间都为00

sec\_nb = 00;

}

void key\_to5() //南北通行设置函数

{

TR0 = 0; //关定时器0

TR1 = 0; //关定时器1

P2 = 0x00; //灭显示

Green\_nb = 1; //置南北方向为绿灯

Red\_dx = 1; //东西方向为红灯

sec\_nb = 00; //四个方向的时间都为00

sec\_dx = 00;

}

//函数的定义部分

void key(void) //按键扫描子程序

{

if (k1 != 1) //当K1（夜间模式）按下

{

display(); //调用显示，用于延时消抖

if (k1 != 1) //如果确定按下

{

key\_to1(); //夜间模式设置函数

do

{

display(); //调用显示，用于延时

} while (k1 != 1); //等待按键释放

}

}

if (k2 != 1) //当K2（禁止）键按下时

{

display(); //调用显示，用于延时消抖

if (k2 != 1) //如果确定按下

{

key\_to2(); //停止通行设置函数

do

{

display(); //调用显示，用于延时

} while (k2 != 1); //等待按键释放

}

}

if (k3 != 1) //当K3（确认）键按下时

{

display(); //调用显示，用于延时消抖

if (k3 != 1) //如果确定按下

{

key\_to3();

do

{

display(); //调用显示，用于延时

} while (k3 != 1); //等待按键释放

}

}

if (k4 != 1) //当K4（东西通行）键按下时

{

display(); //调用显示，用于延时消抖

if (k4 != 1) //如果确定按下

{

key\_to4(); //东西通行函数

do

{

display(); //调用显示，用于延时

} while (k4 != 1); //等待按键释放

}

}

if (k5 != 1) //当K5（南北通行）键按下时

{

display(); //调用显示，用于延时消抖

if (k5 != 1) //如果确定按下

{

key\_to5(); //南北通行函数

do

{

display(); //调用显示，用于延时

} while (k5 != 1); //等待按键释放

}

}

if (hw\_dx != 1) //东西红外探头

{

display(); //调用显示，用于延时消抖和按键检测方法基本相同

if (hw\_dx != 1) //如果确定有车辆经过

{

if (dx\_nb == 0) //东西通行的时间

{

num\_che\_dx++; //用于记录东西绿灯时经过的车辆数

}

else

{

Buzz = 0; //蜂鸣器开

}

while (hw\_dx != 1)

; //等待释放

{

display(); //调用显示，用于延时

}

Buzz = 1; //蜂鸣器关

}

}

if (hw\_nb != 1) //南北红外探头

{

display(); //调用显示，用于延时消抖和按键检测方法基本相同

if (hw\_nb != 1) //如果确定有车辆经过

{

if (dx\_nb == 1) //东西通行的时间

{

num\_che\_nb++; //用于记录南北绿灯时经过的车辆数

}

else

{

Buzz = 0; //蜂鸣器开

}

while (hw\_nb != 1)

; //等待释放

{

display(); //调用显示，用于延时

}

Buzz = 1; //蜂鸣器关

}

}

if (k6 != 1) //（查看）键按下

{

display(); //调用显示，用于延时消抖

if (k6 != 1) //如果确定按下

{

xianshi\_fx++;

if (xianshi\_fx >= 3)

xianshi\_fx = 2;

while (k6 != 1)

; //等待按键释放

{

display(); //调用显示，用于延时

}

}

}

}

void display(void) //显示子程序

{

if (xianshi\_fx == 0) //正常显示

{

buf[1] = sec\_nb / 10; //第1位 东西显示秒十位

buf[2] = sec\_nb % 10; //第2位 东西显示秒个位

buf[3] = sec\_dx / 10; //第3位 南北显示秒十位

buf[0] = sec\_dx % 10; //第4位 南北显示秒个位

}

if (xianshi\_fx == 1) //查看通行时间

{

buf[1] = set\_timenb / 10; //第1位 东西通行秒十位

buf[2] = set\_timenb % 10; //第2位 东西通行秒个位

buf[3] = set\_timedx / 10; //第3位 南北通行秒十位

buf[0] = set\_timedx % 10; //第4位 南北通行秒个位

}

if (xianshi\_fx == 2) //查看红外计数值

{

buf[1] = num\_che\_nb / 10; //第1位 东西红外计数值十位

buf[2] = num\_che\_nb % 10; //第2位 东西红外计数值个位

buf[3] = num\_che\_dx / 10; //第3位 南北红外计数值十位

buf[0] = num\_che\_dx % 10; //第4位 南北红外计数值个位

}

P1 = 0xff; // 初始灯为灭的

P0 = 0x00; ////灭显示

P1 = 0xfe; //片选LED1

P0 = table[buf[1]]; //送东西时间十位的数码管编码

delay(1); //延时

P1 = 0xff; //关显示

P0 = 0x00; //灭显示

P1 = 0xfd; //片选LED2

P0 = table[buf[2]]; //送东西时间个位的数码管编码

delay(1); //延时

P1 = 0xff; //关显示

P0 = 0x00; //关显示

P1 = 0Xfb; //片选LED3

P0 = table[buf[3]]; //送南北时间十位的数码管编码

delay(1); //延时

P1 = 0xff; //关显示

P0 = 0x00; //关显示

P1 = 0Xf7; //片选LED4

P0 = table[buf[0]]; //送南北时间个位的数码管编码

delay(1); //延时

}

void time0(void) interrupt 1 using 1 //定时中断子程序

{

TH0 = 0X3C; //重赋初值

TL0 = 0XB0; //12m晶振50ms//重赋初值

TR0 = 1; //重新启动定时器

countt0++; //软件计数加1

if (countt0 == 10) //加到10也就是半秒

{

if ((sec\_nb <= 5) && (dx\_nb == 0) && (shanruo == 1)) //东西黄灯闪

{

Green\_dx = 0;

Yellow\_dx = 0;

Buzz = 0; //蜂鸣器关

}

if ((sec\_dx <= 5) && (dx\_nb == 1) && (shanruo == 1)) //南北黄灯闪

{

Green\_nb = 0;

Yellow\_nb = 0;

Buzz = 0; //蜂鸣器关

}

}

if (countt0 == 20) // 定时器中断次数=20时（即1秒时）

{

countt0 = 0; //清零计数器

sec\_dx--; //东西时间减1

sec\_nb--; //南北时间减1

if ((sec\_nb <= 5) && (dx\_nb == 0) && (shanruo == 1)) //东西黄灯闪

{

Green\_dx = 0;

Yellow\_dx = 1;

Buzz = 1; //蜂鸣器关

}

if ((sec\_dx <= 5) && (dx\_nb == 1) && (shanruo == 1)) //南北黄灯闪

{

Green\_nb = 0;

Yellow\_nb = 1;

Buzz = 1; //蜂鸣器关

}

if (sec\_dx == 0 && sec\_nb == 5) //当东西倒计时到0时，重置5秒，用于黄灯闪烁时间

{

sec\_dx = 5;

shanruo = 1;

}

if (sec\_nb == 0 && sec\_dx == 5) //当南北倒计时到0时，重置5秒，用于黄灯闪烁时间

{

sec\_nb = 5;

shanruo = 1;

}

if (dx\_nb == 0 && sec\_nb == 0) //当黄灯闪烁时间倒计时到0时，

{

Buzz = 1; //蜂鸣器开

P2 = 0x00; //重置东西南背方向的红绿灯

Green\_nb = 1;

Red\_dx = 1;

dx\_nb = !dx\_nb;

shanruo = 0;

if (num\_che\_nb > set\_timenb / 2) //如果此时南北通行的车辆数大于预设通行量

set\_timenb = set\_timenb + 5;

if (num\_che\_nb == 0) //如果南北方向无车辆通行,每次递减5秒

set\_timenb = set\_timenb - 5;

if (set\_timenb <= 15)

set\_timenb = 15;

sec\_nb = set\_timenb; //重赋南北方向的起始值

sec\_dx = set\_timenb + 5; //重赋东西方向的起始值

num\_che\_nb = 0; //清零

}

if (dx\_nb == 1 && sec\_dx == 0) //当黄灯闪烁时间到

{

P2 = 0X00; //重置东西南北的红绿灯状态

Green\_dx = 1; //东西绿灯亮

Red\_nb = 1; //南北红灯亮

dx\_nb = !dx\_nb; //取反

shanruo = 0; //闪烁

if (num\_che\_dx > set\_timedx / 2) //如果此时南北通行的车辆数大于预设通行量

set\_timedx = set\_timedx + 5;

if (num\_che\_dx == 0) //如果东西方向无车辆通行,每次递减5秒

set\_timedx = set\_timedx - 5;

if (set\_timedx <= 15)

set\_timedx = 15;

sec\_dx = set\_timedx; //重赋东西方向的起始值

sec\_nb = set\_timedx + 5; //重赋南北方向的起始值

num\_che\_dx = 0; //清零

}

}

}

void time1(void) interrupt 3 //定时中断子程序

{

TH1 = 0X3C; //重赋初值

TL1 = 0XB0; //12m晶振50ms//重赋初值

countt1++; //软件计数加1

if (countt1 == 10) // 定时器中断次数=10时（即0.5秒）

{

Yellow\_nb = 0; //南北黄灯灭

Yellow\_dx = 0; //东西黄灯灭

}

if (countt1 == 20) // 定时器中断次数=20时（即1秒时）

{

countt1 = 0; //清零计数器

Yellow\_nb = 1; //南北黄灯亮

Yellow\_dx = 1; //东西黄灯亮

}

}

//外部中断0

void int0(void) interrupt 0 using 1

{

}

//外部中断1

void int1(void) interrupt 2 using 1

{

}

void Com\_Int(void) interrupt 4 //串口中断子函数

{

uchar i;

uchar receive\_data, ruturn\_data[8];

EA = 0;

if (RI == 1) //当硬件接收到一个数据时，RI会置位

{

RI = 0;

receive\_data = SBUF; //接收到的数据

if (receive\_data == '1') //夜间模式

{

key\_to1();

display();

ruturn\_data[0] = 'o';

ruturn\_data[1] = 'k';

ruturn\_data[2] = '-';

ruturn\_data[3] = '1';

ruturn\_data[4] = '\0';

}

else if (receive\_data == '2') //停止模式

{

key\_to2();

display();

ruturn\_data[0] = 'o';

ruturn\_data[1] = 'k';

ruturn\_data[2] = '-';

ruturn\_data[3] = '2';

ruturn\_data[4] = '\0';

}

else if (receive\_data == '3') //确认（重现）模式

{

key\_to3();

display();

ruturn\_data[0] = 'o';

ruturn\_data[1] = 'k';

ruturn\_data[2] = '-';

ruturn\_data[3] = '3';

ruturn\_data[4] = '\0';

}

else if (receive\_data == '4') //东西通行模式

{

key\_to4();

display();

ruturn\_data[0] = 'o';

ruturn\_data[1] = 'k';

ruturn\_data[2] = '-';

ruturn\_data[3] = '4';

ruturn\_data[4] = '\0';

}

else if (receive\_data == '5') //南北通行模式

{

key\_to5();

display();

ruturn\_data[0] = 'o';

ruturn\_data[1] = 'k';

ruturn\_data[2] = '-';

ruturn\_data[3] = '5';

ruturn\_data[4] = '\0';

}

else

{

ruturn\_data[0] = 'e';

ruturn\_data[1] = 'r';

ruturn\_data[2] = 'r';

ruturn\_data[3] = '0';

ruturn\_data[4] = 'r';

ruturn\_data[5] = '\0'; //错误命令

}

}

for (i = 0; i < 8; i++)

{

SBUF = ruturn\_data[i]; //将要发送的数据放入到发送寄存器

while (!TI)

; //等待发送数据完成

TI = 0; //清除发送完成标志位

}

EA = 1;

}

void logo() //开机的Logo "- - - -"

{

for (n = 0; n < 50; n++) //循环显示----50次

{

P0 = 0x40; //送形“-”

P1 = 0xfe; //第一位显示

delay(1); //延时

P1 = 0xfd; //第二位显示

delay(1); //延时

P1 = 0Xfb; //第三位显示

delay(1); //延时

P1 = 0Xf7; //第四位显示

delay(1); //延时

P1 = 0xff; //灭显示

}

}

void delay(int ms) //延时子程序

{

uint j, k;

for (j = 0; j < ms; j++) //延时ms

for (k = 0; k < 124; k++)

; //大约1毫秒的延时

}