目录

[摘要 1](#_Toc22)

[1.前言 1](#_Toc31853)

[1.1项目背景 1](#_Toc24174)

[1.2目的意义 2](#_Toc29322)

[2.主要任务 2](#_Toc2638)

[2.1项目概述 2](#_Toc32622)

[2.2需求分析 2](#_Toc184)

[3.总体设计 3](#_Toc22795)

[3.1功能设计 3](#_Toc236)

[3.2硬件设计 4](#_Toc26222)

[3.2.1 可编程并行接口芯片intel8255 4](#_Toc3938)

[3.2.2 可编程中断控制器intel8259A 6](#_Toc30027)

[3.2.3可编程计数/定时器芯片intel8254 8](#_Toc6580)

[4.详细设计 9](#_Toc183)

[4.1整体思路 9](#_Toc2788)

[4.1.1主程序流程图 9](#_Toc15086)

[4.1.2显示子程序流程图 11](#_Toc23184)

[4.1.3按键扫描子程序流程图 12](#_Toc23066)

[4.1.4按键处理字程序流程图 13](#_Toc14107)

[4.2实验线路设计 14](#_Toc26736)

[4.3代码设计 15](#_Toc10709)

[5.遇到的问题和解决方法 29](#_Toc6693)

[6.总结 29](#_Toc19878)

## 摘要

在日常生活中，交通灯作为管理交通、协调车辆运行的一个便捷的手段，对人类生活的正常运行起着很大的作用。其中各种交通工具、行人都要根据交通灯的指示来决定自己的行进状态，此中，通行时间的规则协调了他们的步伐，极大的降低了由交通混乱引起的各种事故的发生率。因此，一个完善的交通系统，交通灯是其中不可或缺的一部分，而一个完善的交通灯程序可以更加有效的管理当前道路中路面状况，使行人、车辆的行进变得更加顺畅。

## 1.前言

### 1.1项目背景

随着我国国民经济的快速发展，我国机动车辆也发展迅速，而出于种种各种原因导致的城镇道路建设的相对滞后，交通拥挤和堵塞的现象时常发生，如何利用当今的计算机和自动控制技术，有效地缓解当今的道路压力，提高各交通路口的通行能力，减少交通事故的发生，是当今很值得研究的一个课题。目前，国内的交通灯一般设在十字路口。丁字路口等路段，在醒目位置用红、绿、黄三种颜色的指示灯加上一个倒计时的显示器来控制行车，本次我们的课程设计也是针对此来展开的，主要设计一个简单的交通信号灯系统。

### 1.2目的意义

这次课程设计主要是培养我们学习设计汇编语言程序的能力，此次设计我们预期设计出一个可以由人为调整各颜色信号灯时间的交通信号灯，希望通过在掌握所学课程内容的基础上，结合实际的应用背景来设计出一个交通信号灯系统，以此来具备独立设计汇编程序和接口的基本能力，并通过在实际应用中，通过不断的发现问题和解决问题，更好的培养自己的专业素养和技术水平。

## 2.主要任务

### 2.1项目概述

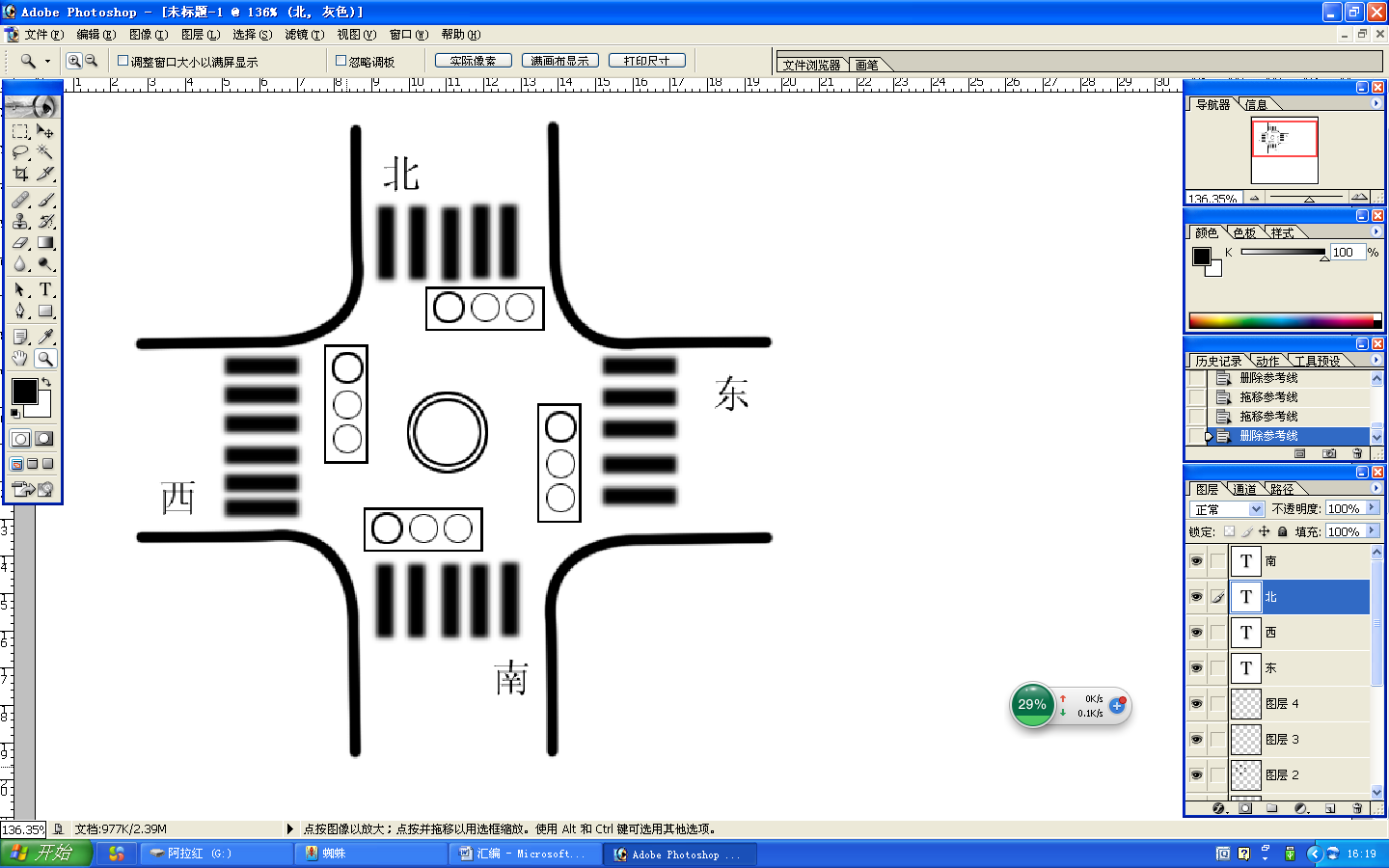
交通灯信号的设计主要是为了针对随着社会发展所出现的道路超负荷承载的情况，设计一个新型的交通控制系统已经亟不可待。随着人们消费水平的提高，私人车辆不断地增加。人多、车多道路少的道路交通状况越来越明显。车辆的增加虽然反映出了国家的整体进步，但是也给人民带来了其他的一些负面的影响。

当前全国大城市中普遍存在着道路拥挤、车辆堵塞、交通秩序混乱的现象，如何解决城市交通问题已成为全社会关注的焦点和大众的迫切呼声。城市交通路口实现交通信号控制是城市交通管理现代化的基本标志之一，是提高交通管理效能的重要技术手段。红绿灯控制器是控制交叉路口交通信号的设备。它是交通信号控制的重要组成部分。各种交通控制方案，最终都要由红绿灯控制器来实现。

### 2.2需求分析

本次设计需要运用所学知识，设计出一台交通信号灯，模拟路口交通信号。系统设定为以十字路口的单向车辆通行控制为研究对象，东南西北四方方向的路口都设有红灯、黄灯、绿灯三色交通灯（东西为一向，南北为一向），当绿灯亮时，可以通行。设计系统功能需求如下：

1. 交通信号灯可以控制东西。南北两个方向的交通，红绿黄灯各用一个发光二极管对应，其中，红灯用一个红色发光二极管表示，绿灯和光灯均用一个绿色发光二极管代表
2. 用一个数码管显示通行道路的通行时间，其中，东西向和南北向通行时间各为40秒，红绿灯切换中间黄灯闪烁5秒



其中，东西向用一组信号灯，南北向用一组信号灯，东西向和南北向各需红绿黄三个信号灯，所以共需六盏灯，两组发光二极管即四个绿色发光二极管和两个黄色发光二极管，交替亮灭来设计控制系统，即可达到交通信号灯的控制目的。由此设计要求可知，要求东西或南北向通行40秒，红绿灯切换中间黄灯亮5秒。现做如下要求：绿灯亮放行，红灯亮禁止通行，黄灯亮为警示提醒时间。

则某方向的交通灯在绿灯亮40秒，放行之后，黄灯亮5秒，接着红灯亮45秒，禁止通行，由于且课题的研究对象是单向通行灯，所以在任一时刻只能有一个方向的绿灯亮起，，按照此思想，系统主要包括四个状态：

|  |  |  |
| --- | --- | --- |
| 状态 | 东西向 | 南北向 |
| 1 | 绿灯 | 红灯 |
| 2 | 黄灯 | 红灯 |
| 3 | 红灯 | 绿灯 |
| 4 | 红灯 | 黄灯 |

## 3.总体设计

### 3.1功能设计

1.具体工作流程如下：

东西向红灯，南北向绿灯，持续一段时间（自己设定，本次实验设定为40秒），南北向转为黄灯持续5秒，东西向红灯转为绿灯

东西向绿灯后，南北向红灯，持续相同时间（自己设定，本次实验设定为40秒）后，东西向转为黄灯持续5秒，南北向红灯转为绿灯

按照以上规律依次持续变化

2. 各方向要有两位数码管分别显示准行时间或禁行时间，并以秒倒计时。

3. 各方向红、绿灯亮的时间为11～ 99秒（可灵活的任意设定），以满足不同交通路口的需要。

4. 设置功能按键：

在发生交通事故时，东西方向和南北方向同时禁行。

照此思路，系统以8086CPU为控制器，控制并行接口8255A驱动交通信号灯按照此设定方案进行自动变化。

### 3.2硬件设计

#### 3.2.1 可编程并行接口芯片intel8255

8255芯片是Intel公司生产的可编程并行I/O接口芯片，有3个8位并行I/O口。具有3个通道3种工作方式的可编程并行接口芯片（40引脚）。 其各口功能可由软件选择，使用灵活，通用性强。8255可作为[单片机](https://baike.baidu.com/item/%E5%8D%95%E7%89%87%E6%9C%BA/102396" \t "_blank)与多种外设连接时的中间接口电路。

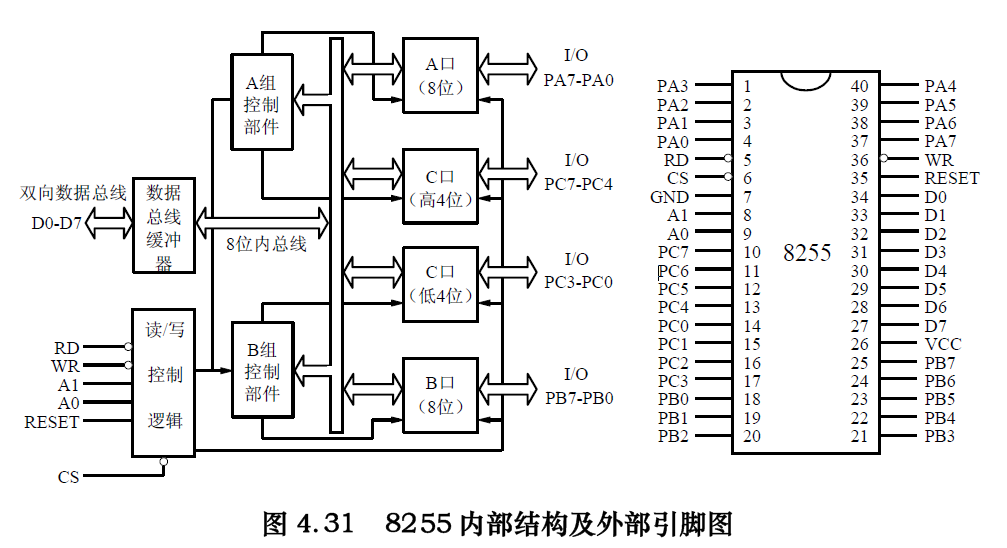


图1 8255A可编程并行接口

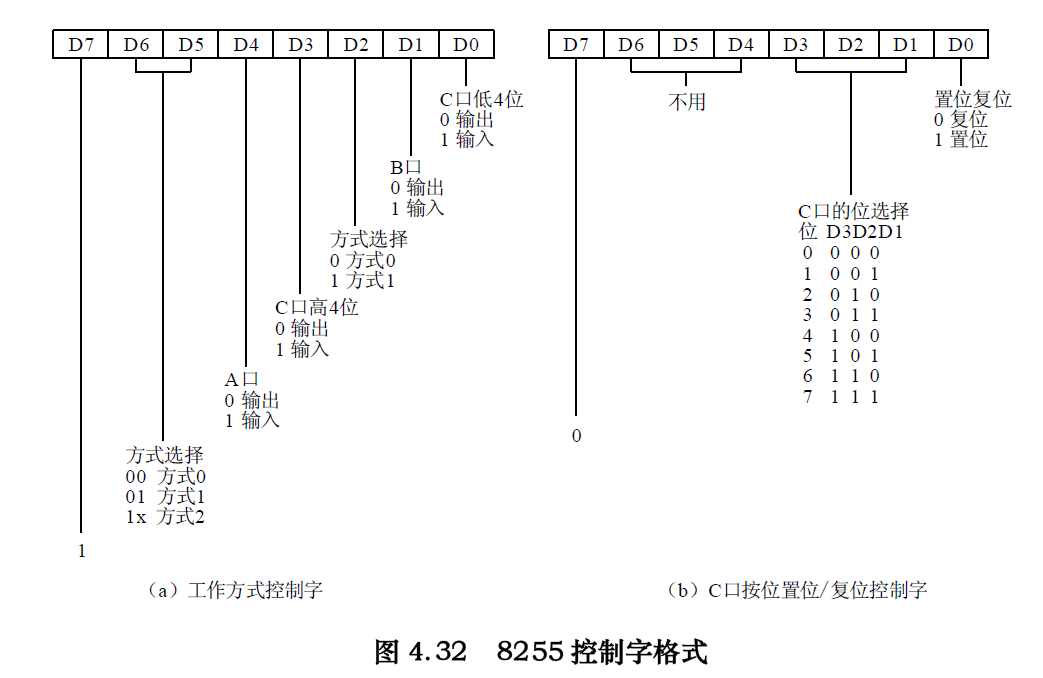


图2 8255控制字格式

RESET:复位输入线，当该输入端处于高电平时，所有内部寄存器（包括控制寄存器）均被清除，所有I/口均被置成输入方式。

CS:芯片选择信号线，当这个输入引脚为低电平时,即CS=0时,表示芯片被选中，允许8255与CPU进行通讯;CS=1时,8255无法与CPU做数据传输.

RD:读信号线，当这个输入引脚为低跳变沿时,即/RD产生一个低脉冲且/CS=0时,允许8255通过数据总线向CPU发送数据或状态信息，即CPU从8255读取信息或数据。

WR:写入信号，当这个输入引脚为低跳变沿时,即/WR产生一个低脉冲且/CS=0时,允许CPU将数据或控制字写入8255。

D0～D7:三态双向数据总线，8255与CPU数据传送的通道，当CPU 执行输入输出指令时，通过它实现8位数据的读/写操作，控制字和状态信息也通过数据总线传送。

PA0～PA7:端口A输入输出线，一个8位的数据输出锁存器/缓冲器， 一个8位的数据输入锁存器。 工作于三种方式中的任何一种；

PB0～PB7:端口B输入输出线，一个8位的I/O锁存器， 一个8位的输入输出缓冲器。 不能工作于方式二；

PC0～PC7:端口C输入输出线，一个8位的数据输出锁存器/缓冲器， 一个8位的数据输入缓冲器。端口C可以通过工作方式设定而分成2个4位的端口， 每个4位的端口包含一个4位的锁存器，分别与端口A和端口B配合使用，可作为控制信号输出或状态信号输入端口。不能工作于方式一或二。

A1,A0:地址选择线,用来选择8255的PA口,PB口,PC口和控制寄存器.

当A1=0,A0=0时,PA口被选择;

当A1=0,A0=1时,PB口被选择;

当A1=1,A0=0时,PC口被选择;

当A1=1.A0=1时,控制寄存器被选择.

8255A具有3个相互独立的输入/输出通道端口，用+5V单电源供电，能在以下三种方式下工作。

方式0————基本输入输出方式；

方式1————选通输入/出方式；

方式2————双向选通输入/输出方式；

**1) 工作方式0(基本输入输出方式)**

功能：方式0不使用联络信号，也不使用中断，A口和B口可定义为输入或输出口，C口分成两个部分（高四位和低四位），C口的两个部分也可分别定义为输入或输出。在方式0，所有口输出均有锁存，输入只有缓冲，无锁存，C口还具有按位将其各位清0或置1的功能。常用于与外设无条件的数据传送或接收外设的数据。

**2） 工作方式1(选通输入输出方式)**

A口借用C口的一些信号线用作控制和状态信号，组成A组，B口借用C口的一些信号线用作控制和状态信号，组成B组。在方式1下，C口的某些位被占用。

方式1的输出：当A口工作于方式1且用作输出口时，C口的PC7线用作输出缓冲器满OBF信号，PC6用作外设B口输出，B口输出收到数据后的响应信号ACK，PC3用作中断请求输出信号线INTR。当B口工作于方式l且用作输出口时，C口的PC1线用作输出缓冲器满OBF信号，PC2用作外设收到数据后的响应信号ACK，PC0用作中断请求输出信号线INTR。

**3) 工作方式2 (双向输入输出方式)**

功能：方式2是A组独有的工作方式。外设既能在A口的8条引线上发送数据，又能接收数据。此方式也是借用C口的5条信号线作控制和状态线，A口的输入和输出均带有锁存。

#### 3.2.2 可编程中断控制器intel8259A

8259A是[专门](https://baike.baidu.com/item/%E4%B8%93%E9%97%A8/8698309" \t "_blank)为了对8085A和8086/8088进行中断[控制](https://baike.baidu.com/item/%E6%8E%A7%E5%88%B6/17222" \t "_blank)而设计的芯片，它是可以用程序控制的中断控制器。单个的8259A能管理8级向量优先级中断。在不增加其他电路的情况下，最多可以级联成64级的向量优级[中断系统](https://baike.baidu.com/item/%E4%B8%AD%E6%96%AD%E7%B3%BB%E7%BB%9F/10480702" \t "_blank)。8259A有多种工作方式，能用于各种系统。各种工作方式的设定是在初始化时通过软件进行的。在[总线控制器](https://baike.baidu.com/item/%E6%80%BB%E7%BA%BF%E6%8E%A7%E5%88%B6%E5%99%A8/8829516" \t "_blank)的控制下，8259A芯片可以处于编程状态和操作状态，编程状态是CPU使用IN或OUT指令对8259A芯片进行初始化编程的状态。

8259A的内部连接及外部管脚引出如下图：

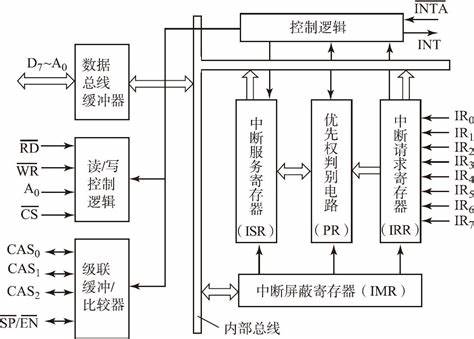


图3 8259A内部结构框图

主要功能

1）具有8级优先权控制，通过级联可扩展至64级优先权控制。

2）对任何一个级别的中断源都可以由编程进行屏蔽或开放。

3）在中断响应周期，可提供响应的中断向量，从而使CPU迅速找到中断服务程序的入口地址。

4）可以通过编程来选择不同的工作方式。

引脚信号

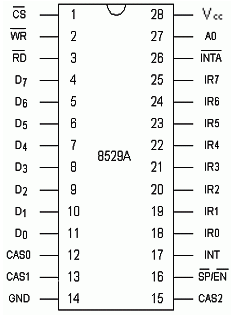


图4 8259A引脚图

8259A的引脚信号排列图如图所示。图中的引脚信号可分为三类:

1）和CPU总线相连接的信号

CS:片选输入信号，低电平有效，通过地址译码逻辑电路与地址总线相连。

WR:写控制输入信号，低电平有效。有效时，表明当前正在向8259A写入命令字。

RD:读控制输入信号，低电平有效。有效时，表明CPU正在向8259A读数据。

D0-D7：双向三态数据线，是CPU和8259A进行数据、命令状态传送的通道。

Ao:片内地址选择信号，用来指出当前8259A的哪个端口被访问。每片8259A有两端口地址。

INTA:接收来自CPU的中断响应信号。8259A要求中断响应信号由两个负脉冲构成。

INT:与CPU的INTR端相连，由8259A向CPU发出中断请求信号。

2）来自外部8个中断源的中断请求信号:IR0-IR7

3）多片8259A级联的级联信号

CAS0-CAS2：级联信号线， 作为主片与从片的连接线，主片为输出，从片为输入。

SP/EN:双向信号线，用于从片选择或总线驱动器的控制信号。

在对8259进行编程时，首先必须进行初始化。一般先使用CLI指令将所有的可屏蔽中断禁止，然后写入初始化命令字。8259有一个状态机控制对寄存器的访问，不正确的初始化顺序会造成异常初始化。在初始化主片8259时，写入初始化命令字的顺序是：ICW1、ICW2、ICW3、然后是ICW4，初始化从片8259的顺序与初始化主片8259的顺序是相同的。系统启动时，主片8259已被初始化，且4号中断源（IR4）提供给与PC联机的串口通信使用，其它中断源被屏蔽。中断矢量地址与中断号之间的关系如下表所示：

#### 3.2.3可编程计数/定时器芯片intel8254

8254是8253的改进型，基本功能与8253相近但比8253具有更优良的性能。

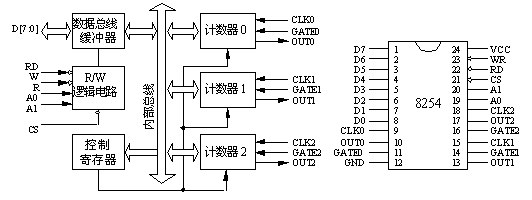


图5 8254的内部结构图和引脚图

8254具有以下基本功能：

（1）有3个独立的16位计数器；

（2）每个计数器可按二进制或十进制（BCD）计数；

（3）每个计数器可编程工作于6种不同工作方式；

（4）8254每个计数器允许的最高计数频率为10MHz（8253为2MHz）；

（5）8254有读回命令（8253没有），除了可以读出当前计数单元的内容外，还可以读出状态寄存器的内容；

（6）计数脉冲可以是有规律的时钟信号，也可以是随机信号。计数初值公式为：C:\Users\Administrator\AppData\Roaming\Tencent\Users\459335735\QQ\WinTemp\RichOle\9)6ZY5RY716004IRSFCR5FQ.png

其中，C:\Users\Administrator\AppData\Roaming\Tencent\Users\459335735\QQ\WinTemp\RichOle\D~9MZ57D{[UAZ_DNSA0]{QQ.png是输入时钟脉冲的频率，C:\Users\Administrator\AppData\Roaming\Tencent\Users\459335735\QQ\WinTemp\RichOle\62O)IU@NC{NH0TFX8QE{4)B.png是输出波形的频率。

8254的工作方式如下述：

（1）方式0：计数到0结束输出正跃变信号方式。

（2）方式1：硬件可重触发单稳方式。

（3）方式2：频率发生器方式。

（4）方式3：方波发生器。

（5）方式4：软件触发选通方式。

（6）方式5：硬件触发选通方式。

8254的控制字有两个：一个用来设置计数器的工作方式，称为方式控制字；另一个用来设置读回命令，称为读回控制字。这两个控制字共用一个地址，由标识位来区分。控制字格式如表1所示。



表1 8254的方式控制字格式

## 4.详细设计

### 4.1整体思路

4.1.1主程序流程图

对于项目的总流程，首先初始化开中断，对各个芯片进行相应的初始化设置，然后不断执行显示子程序、清屏子程序、键盘扫描子程序，获取按键的值，若果有按键按下，则执行按键处理子程序，否则继续执行显示子程序。

具体主程序的流程图与初始化流程图如下：

图7 设计总流程图图 8 初始化流程图

4.1.2显示子程序流程图

首先确定为自动模式还是手动显示模式，然后定义不同的循环变量以及数码管控制变量，再设置A口、C口控制LED灯的显示，显示数码管。具体流程图如下：



图9 显示流程图

4.1.3按键扫描子程序流程图

在执行键盘扫描子程序时，首先使输出端口全为0，然后逐行扫描从端口读入数据，传输按键的值，具体流程图如下：



图10 按键扫描流程图

4.1.4按键处理字程序流程图

扫描键盘输入，并将扫描结果进行处理后送数码管显示。将键盘进行编号，记作0～F，当按下其中一个按键时：

1）数字键“0~9”： 仅在设置通行时间时按键有效（设置键：“B”）,实现通行时间设置的组合，数码管实时显示“XX——XX”（XX表示预设的时间）

2）确定键“F”：将时间预设值修改，按下后进入正常状态。

3）取消键“E”：退出时间设置模式和功能键模式。

具体流程图如下：



图11 按键处理字程序流程图

### 4.2实验线路设计

综合利用实验箱上的资源，利用8254单元输入1MHz的时钟信号，连接MIR7定时向控制系统发出中断信号，实现倒计时的功能，通过8255单元控制的键盘及数码管显示单元，动态扫描通行时间的变化以及功能键模式下的显示，实时扫描键盘的输入，通过软件处理的方式，实现各个键盘的处理，开关及LED显示单元则来用显示交通灯的变化，B口输出数码管段码，A口与C口共同控制数码管的位选、键盘扫描以及LED灯的亮灭。

其中C口PC4、PA6、PA7为东西方向指示灯，PA7-红，PA6-绿，PC4-黄。PC5~PC7为南北方向指示灯 PC7-红，PC6-绿，PC5-黄

下图为整个交通灯控制系统的接线图：



图6 实验连线图

### 4.3代码设计

;=======================================================

; 文件名: traffic.asm

; 功能描述: 交通灯控制系统

; 8255的 B口控制数码管的段显示，

; A口控制键盘列扫描及数码管的位驱动，

; C口控制键盘的行扫描。

;=======================================================

IOY0 EQU 0600H ;片选IOY0对应的端口始地址

MY8255\_A EQU IOY0+00H\*2 ;8255的A口地址

MY8255\_B EQU IOY0+01H\*2 ;8255的B口地址

MY8255\_C EQU IOY0+02H\*2 ;8255的C口地址

MY8255\_CON EQU IOY0+03H\*2 ;8255的控制寄存器地址

A8254 EQU 06C0H ;8254计数器0端口地址

B8254 EQU 06C2H ;8254计数器1端口地址

C8254 EQU 06C4H ;8254计数器2端口地址

CON8254 EQU 06C6H ;8254 控制寄存器端口地址

SSTACK SEGMENT STACK

DW 200 DUP(?)

SSTACK ENDS

DATA SEGMENT

; DATBLE是 将需要输入按键的值对应需要给的显示器的值

; 比如按键1表示的值是1 但是我们送给显示器的是06H

; 该程序是通过判断按键按下 获取其代表的偏移量（相对于DTABLE）

; 比如按键1的偏移量是1 我们扫描按键 得出一个值 1

; 然后利用该值在DTABLE中找到需要输出值的对应显示代码值

; 从B口送出去即可

TIME\_COUNT DW 0 ;计时，单位为s

STATE DW ?,?,?,? ;每个状态的结束时间

GREEN DW 0AH

YELLOW DW 03H

;东西：|-----------------|----|----------------------|

; 绿 40s 黄5s 红 45s

;南北：|----------------------|-----------------|----|

; 红 45s 绿 40s 黄5s

; STATE1: PA6-PC7 绿-红

; STATE2: PA4-PC7 黄-红

; STATE3: PA7-PC6 红-绿

; STATE4: PA7-PC5 红-黄

DTABLE DB 3FH,06H,5BH,4FH,66H,6DH,7DH,07H

DB 7FH,6FH,77H,7CH,39H,5EH,79H,71H

CD DB ?,?,?,?,?,? ;数码管显示缓冲区

A\_SUB DB ? ;A口的PA7和PA6

CX\_SUB DW 0 ;存中断时的CX值，意义：时钟

LED\_A DB 0H

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START:

;===========================================

; 初始化

;===========================================

MOV AX, DATA

MOV DS, AX

XOR CX,CX

CALL INIT\_TIME

; 把CD中的值全部初始化为00H

; 说明初始偏移量全为0

MOV SI,OFFSET CD

MOV AL,00H

MOV [SI],AL ;清显示缓冲

MOV [SI+1],AL

MOV [SI+2],AL

MOV [SI+3],AL

MOV [SI+4],AL

MOV [SI+5],AL

; 中断向量表设置

PUSH DS

MOV AX, 0H

MOV DS, AX

MOV AX, OFFSET MIR7 ;取中断入口地址

MOV SI, 003CH ;中断矢量地址

MOV [SI], AX ;填MIR7的偏移矢量

MOV AX, CS ;段地址

MOV SI, 003EH

MOV [SI], AX ;填MIR7的段地址矢量

MOV AX, 0H

MOV DS, AX

MOV AX, OFFSET MIR6 ;取中断入口地址

MOV SI, 0038H ;中断矢量地址

MOV [SI], AX ;填MIR6的偏移矢量

MOV AX, CS ;段地址

MOV SI, 003AH

MOV [SI], AX ;填MIR6的段地址矢量

CLI

POP DS

;初始化主片8259

MOV AL, 11H

OUT 20H, AL ;ICW1

MOV AL, 08H

OUT 21H, AL ;ICW2

MOV AL, 04H

OUT 21H, AL ;ICW3

MOV AL, 01H

OUT 21H, AL ;ICW4

MOV AL, 2FH ;OCW1

OUT 21H, AL

;INIT 8255

MOV DX,MY8255\_CON

MOV AL,81H

OUT DX,AL

;8254

MOV DX, CON8254 ;8254

MOV AL, 36H ;0011 0110计数器0，方式3

OUT DX, AL

MOV DX, A8254

MOV AL, 10H ;03E8H --> 1000

OUT DX, AL

MOV AL, 27H

OUT DX, AL

MOV DI,OFFSET CD+5

STI

;===========================================

; 主程序

;===========================================

BEGIN: MOV SI,OFFSET CD

MOV AL,00H

MOV [SI+4],AL

MOV [SI+5],AL

CALL DIS

CALL CLEAR

JMP BEGIN

;===========================================

; CLEAR 清屏子程序

;===========================================

;就是使得所有的灯熄灭 00H表示全不亮 瞬间 很快

CLEAR: MOV DX,MY8255\_B

MOV AL,00H

OUT DX,AL

RET

;===========================================

; DIS 显示子程序

;===========================================

DIS: PUSH AX

MOV SI,OFFSET CD

; 0DFH=1101 1111 对应PA7 PA6 PA5...PA1 PA0

; 由电路图 得出 X1-PA0 X2-PA1.....

; 6个显示器 从左到右依次是 X1 X2 X3... X5 X6

; 所以 对应的PA: PA0 PA1 PA2 PA3 PA4 PA5

; 这里初始是0DFH 代表 1 1 1 1 1 0

; 意思是 第六个显示 开始显示数字

; 哈哈 这里其实是从X6到X1依次显示的

; 每个数字显示间隔很快 我们会认为是6个数字一起显示 其实是逐个显示

MOV DL,0DFH

MOV AL,DL

AGAIN: PUSH DX

; 把AL送给A口 觉得开放哪个灯 （这里要看电路图 A口也控制灯的开放）

MOV DX,MY8255\_A

PUSH AX ;对PA7 PA6特殊处理

AND AL,3FH

MOV BL,[LED\_A]

OR AL,BL

OUT DX,AL

POP AX

MOV AL,[SI] ; 把3000H--3005H中存的偏移量（相对）取出

MOV BX,OFFSET DTABLE ; 获取DTABLE的首地址

AND AX,00FFH ; 因为后面会有加法运算 先把ah清0 这样ax就是

; al的值，防止出错

ADD BX,AX ; 获取需要的值的偏移量（这个是绝对偏移量）

MOV AL,[BX] ; 获取显示数字需要的值 例 显示0需要3FH

MOV DX,MY8255\_B ; 送往B口 显示数字 12-1\2

OUT DX,AL

CALL DALLY ; 延时

INC SI ; 移动SI 读取下一个偏移量

POP DX

MOV AL,DL ; DL: 控制哪个灯的开放 开始是0DF 1101 1111

; 取后6位（看电路图 只连了6根线）即01 1111

; 赋值给AL

TEST AL,01H ; 测试AL 看是否为11 1110

; 6个灯 一次显示需要循环6次

; 这里第六次结束是 AL=11 1110

; 对于灯 就是x1灯显示完（灯：X6->X1）

JZ OUT1 ; 6次循环完成后 跳出

ROR AL,1 ; 循环右移

; 例 第一个灯亮 AL=01 1111

; 则 第二个灯亮 为 10 1111

; 所以需要循环右移

; 反映在灯上 则是左移（不要绕进去了哦）

MOV DL,AL

JMP AGAIN ; 跳回 继续显示 需循环6次

OUT1: POP AX

RET

;===========================================

; DALLY子程序 延时作用 RET为子程序结束标记

;===========================================

DALLY: PUSH CX

MOV CX,0006H

T1: MOV AX,009FH

T2: DEC AX

JNZ T2

LOOP T1

POP CX

RET

;===========================================

; PUTBUF

;===========================================

; 将获得的偏移量存入CD中

; 便于后面的显示

; 显示其实就是从CD中读取偏移量

; 然后在table中找到真正的值即可

PUTBUF: MOV SI,DI ;存键盘值到相应位的缓冲中

MOV [SI],AL ;先存入地址CD+5 再递减 也就是下一个存入偏移量的是CD+4

DEC DI

MOV AX,OFFSET CD-1

CMP DI,AX

JNZ GOBACK

MOV DI,OFFSET CD+5

GOBACK: RET

;===========================================

; MIR7

;===========================================

MIR7:

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV AX,[TIME\_COUNT]

INC AX

MOV [TIME\_COUNT],AX

CMP AX,100

JNE MID

XOR AX,AX

MOV [TIME\_COUNT],AX

MOV DX, A8254

MOV AL, 10H ;2710H --> 1000

OUT DX, AL

MOV AL, 27H

OUT DX, AL

MOV CX,[CX\_SUB]

CMP CX,[STATE]

JL STATE1

MOV BX,[STATE+2]

CMP CX,BX

JL STATE2

MOV BX,[STATE+4]

CMP CX,BX

JL JMP\_STATE3

MOV BX,[STATE+6]

CMP CX,BX

JL JMP\_STATE4

XOR CX,CX ;清零

JMP STATE1

STATE1: MOV AX,0131H ; 绿-红 A：40H C:80H

INT 10H

MOV DX,MY8255\_A

MOV AL,80H

MOV [LED\_A],AL

OUT DX,AL

MOV [A\_SUB],AL

MOV DX,MY8255\_C

MOV AL,40H

OUT DX,AL

;倒计时

MOV AX,[STATE]

SUB AX,CX

MOV BL,0AH

DIV BL

MOV [CD],AH ;余数，个位

MOV [CD+1],AL

MOV AX,[STATE+2]

SUB AX,CX

DIV BL

MOV [CD+2],AH

MOV [CD+3],AL

JMP NEXT

MID: JMP RETURN

JMP\_STATE3:JMP STATE3

STATE2: MOV AX,0132H ; 黄-红 A：10H C：80H

INT 10H

MOV DX,MY8255\_A

MOV AL,80H

MOV [LED\_A],AL

OUT DX,AL

MOV [A\_SUB],AL

MOV DX,MY8255\_C

MOV AL,20H

OUT DX,AL

;倒计时

MOV AX,[STATE+2]

SUB AX,CX

MOV BL,0AH

DIV BL

MOV [CD],AH ;余数，个位

MOV [CD+1],AL ;余数，十位

MOV AX,[STATE+2]

SUB AX,CX

DIV BL

MOV [CD+2],AH

MOV [CD+3],AL

JMP NEXT

JMP\_STATE4:JMP STATE4

STATE3: MOV AX,0133H ; 红-绿 A：80H C:40H

INT 10H

MOV DX,MY8255\_A

MOV AL,40H

MOV [LED\_A],AL

OUT DX,AL

MOV [A\_SUB],AL

MOV DX,MY8255\_C

MOV AL,80H

OUT DX,AL

;倒计时

MOV AX,[STATE+6]

SUB AX,CX

MOV BL,0AH

DIV BL

MOV [CD],AH ;余数，个位

MOV [CD+1],AL ;余数，十位

MOV AX,[STATE+4]

SUB AX,CX

DIV BL

MOV [CD+2],AH

MOV [CD+3],AL

JMP NEXT

STATE4: MOV AX,0134H ; 红-黄 A：80H C：20H

INT 10H

MOV DX,MY8255\_A

MOV AL,00H

MOV [LED\_A],AL

OUT DX,AL

MOV [A\_SUB],AL

MOV DX,MY8255\_C

MOV AL,90H

OUT DX,AL

;倒计时

MOV AX,[STATE+6]

SUB AX,CX

MOV BL,0AH

DIV BL

MOV [CD],AH ;余数，个位

MOV [CD+1],AL ;余数，十位

MOV AX,[STATE+6]

SUB AX,CX

DIV BL

MOV [CD+2],AH

MOV [CD+3],AL

JMP NEXT

NEXT:

INC CX

MOV [CX\_SUB],CX

RETURN: MOV AL, 20H

OUT 20H, AL ;中断结束命令

POP DX

POP CX

POP BX

POP AX

IRET

;===========================================

; INIT\_TIME 对时间设置的初始化

;===========================================

INIT\_TIME:

PUSH AX

PUSH BX

PUSH CX

PUSH SI

XOR CX,CX

MOV [CX\_SUB],CX

MOV [TIME\_COUNT],CX

MOV AX,[GREEN]

MOV BX,[YELLOW]

MOV SI,OFFSET STATE

ADD CX,AX

MOV [SI],CX

ADD CX,BX

MOV [SI+2],CX

ADD CX,AX

MOV [SI+4],CX

ADD CX,BX

MOV [SI+6],CX

ADD CX,AX

MOV [SI+8],CX

XOR AX,AX

MOV [TIME\_COUNT],AX

POP SI

POP CX

POP BX

POP AX

RET

;===========================================

; CCSCAN 键盘扫描子程序

;===========================================

; 原理是 先向全部列输出低电平

; 然后从C口读入 行电平

; 如果没有按键按下 所有行应该均为高电平

; 反之 若有按键按下 则开始仔细判断出到底是哪个按键按下 具体判断方法是：

; 先向第一列输出低电平（从左到右）

; 然后从C口读入行电平 利用 AND

; 判断哪一行是否为低电平即可(后面为了计算方便取反了行电平)

; 若行全为高 为开始向下一列输出低电平 循环4次即可

CCSCAN: MOV AL,00H

MOV DX,MY8255\_A

OUT DX,AL ; 向所有列输出 低电平

MOV DX,MY8255\_C

IN AL,DX ;读所有行电平

;原来没有任何键按下 4行全为1

;这里取反 变成 0000 便于后面的判断

NOT AL

; 假设没有按键按下

; 0000&1111=0

; 结果为0 ZF=1

AND AL,0FH

RET

;=====================================

; MIR6 中断

;=====================================

MIR6:

CLI

PUSH AX

PUSH BX

PUSH CX

PUSH DX

PUSH SI

PUSH DI

MOV SI,OFFSET CD

MOV AL,00H

MOV [SI],AL ;清显示缓冲

MOV [SI+1],AL

MOV [SI+2],AL

MOV [SI+3],AL

MOV [SI+4],AL

MOV [SI+5],AL

MOV DI,OFFSET CD+5

BEGIN2:

CALL DIS

CALL CLEAR

CALL CCSCAN

JNZ INK1

JMP BEGIN2

INK1:

CALL DIS

CALL DALLY

CALL DALLY

CALL CLEAR

CALL CCSCAN

JNZ INK2

JMP BEGIN2

INK2:

MOV CH,0FEH ; FEH=1111 1110（对应关系：PA7 PA6..PA1 PA0 ）

MOV CL,00H ; 初始对于行的偏移量 为0

COLUM:

MOV AL,CH

MOV DX,MY8255\_A

OUT DX,AL

MOV DX,MY8255\_C

IN AL,DX

L1: TEST AL,01H ;is L1?

JNZ L2

MOV AL,00H ;L1

JMP KCODE

L2: TEST AL,02H ;is L2?

JNZ L3

MOV AL,04H ;L2

JMP KCODE

L3: TEST AL,04H ;is L3?

JNZ L4

MOV AL,08H ;L3

JMP KCODE

L4: TEST AL,08H ;is L4?

JNZ NEXT2

MOV AL,0CH ;L4

KCODE: ADD AL,CL ;得到总的偏移量

CMP AL,0EH ;取消键

JZ RETURN2

CMP AL,0FH ;确定键

JZ ENSURE\_BTN

SHOW: CALL PUTBUF

PUSH AX

KON: CALL DIS

CALL CLEAR

CALL CCSCAN

JNZ KON

POP AX

NEXT2: INC CL ; CL相当于 行偏移量

MOV AL,CH

TEST AL,08H ; 08H=0000 1000 当AL为1111 0111 && 0000 1000 结果为0

JZ KERR ; 4次列循环结束 跳KERR

ROL AL,1

MOV CH,AL

JMP COLUM

KERR:

JMP BEGIN2

RETURN2: MOV AL, 20H

OUT 20H, AL ;中断结束命令

POP DI

POP SI

POP DX

POP CX

POP BX

POP AX

STI

IRET

ENSURE\_BTN:CALL SET\_TIME

JMP RETURN2

;=============================================

; SET\_TIME 设置时间子程序

;=============================================

SET\_TIME:

PUSH AX

PUSH BX

PUSH CX

XOR AX,AX

MOV SI,OFFSET CD

MOV CX,0AH

MOV AL,[SI+5] ;十位

MOV BL,[SI+4] ;个位

MUL CL

ADD AL,BL

MOV [GREEN],AX

MOV AL,[SI+3] ;十位

MOV BL,[SI+2] ;个位

MUL CL

ADD AL,BL

MOV [YELLOW],AX

CALL INIT\_TIME

MOV AL,00H

MOV [SI],AL ;清显示缓冲

MOV [SI+1],AL

MOV [SI+2],AL

MOV [SI+3],AL

MOV [SI+4],AL

MOV [SI+5],AL

CALL CLEAR

PUSH CX

POP BX

POP AX

RET

CODE ENDS

END START

## 5.遇到的问题和解决方法

（1）需求分析时各状态的定义不明确，红灯、绿灯分配时间不合理。

解决方法：通过查阅资料，获取了不同方案的交通灯通行时间方案，最终讨论出合理的时间分配方案。

（2）经查阅相关资料，按键时会出现按键抖动现象，可能会影响程序的正常运行。

解决方法：在按键响应后进行一段时间的延时。

（3）设置通行时间时两位数的时间不好处理。

解决方法：运用合理的公式将数字键组成（11~99）的数字，利用计数器原理实现不同按键功能的切换。

（4）8254 计数器1出现故障

解决方法：计数器1本来就是用来分频的，我们不用计数器1，只用计数器了，在程序内存中定义一个时钟变量，给计数器1设定初值为10000，每产生100次中断，才进行交通灯时钟的变化，这样便可以做到1s一个时钟信号。

## 6.总结

此次项目的设计，我们设计出了一套实验交通灯控制系统的方案并将其初步实现。此次课程设计设计成本低，且符合日常生活的实际需求，对缓解城市交通压力问题有一定的现实意义。

并且在设计过程中我们小组了解了8086芯片中各个引脚的功能，怎样去使用8255这个可编程并行接口芯片，以及如何使用8253定时及输出一定频率的输出，对接口技术有了更加深入的了解。同时小组成员一块配合，共同搜集材料解决设计中碰到的一系列问题，这更加增强了我们团队合作的意识。