**西安电子科技大学**

微机原理课程设计  **课程实验报告**

**实验名称** 交通信号灯自动控制模拟指示系统

计算机科学与技术 学院 1903011 班

成 绩

姓名 学号

同作者 袁铮 张峻凡 等

实验日期 2022 年 5 月 8 日

实验地点E-Ⅱ-312实验批次第四批

|  |
| --- |
| 指导教师评语：  指导教师：  年 月 日 |
| **实验报告内容基本要求及参考格式**  一、实验目的  二、实验所用仪器（或实验环境）  三、实验基本原理及步骤（或方案设计及理论计算）  四、实验数据记录（或仿真及软件设计）  五、实验结果分析及回答问题（或测试环境及测试结果） |

**交通信号灯自动控制模拟指示系统**

1. **课程设计目的**

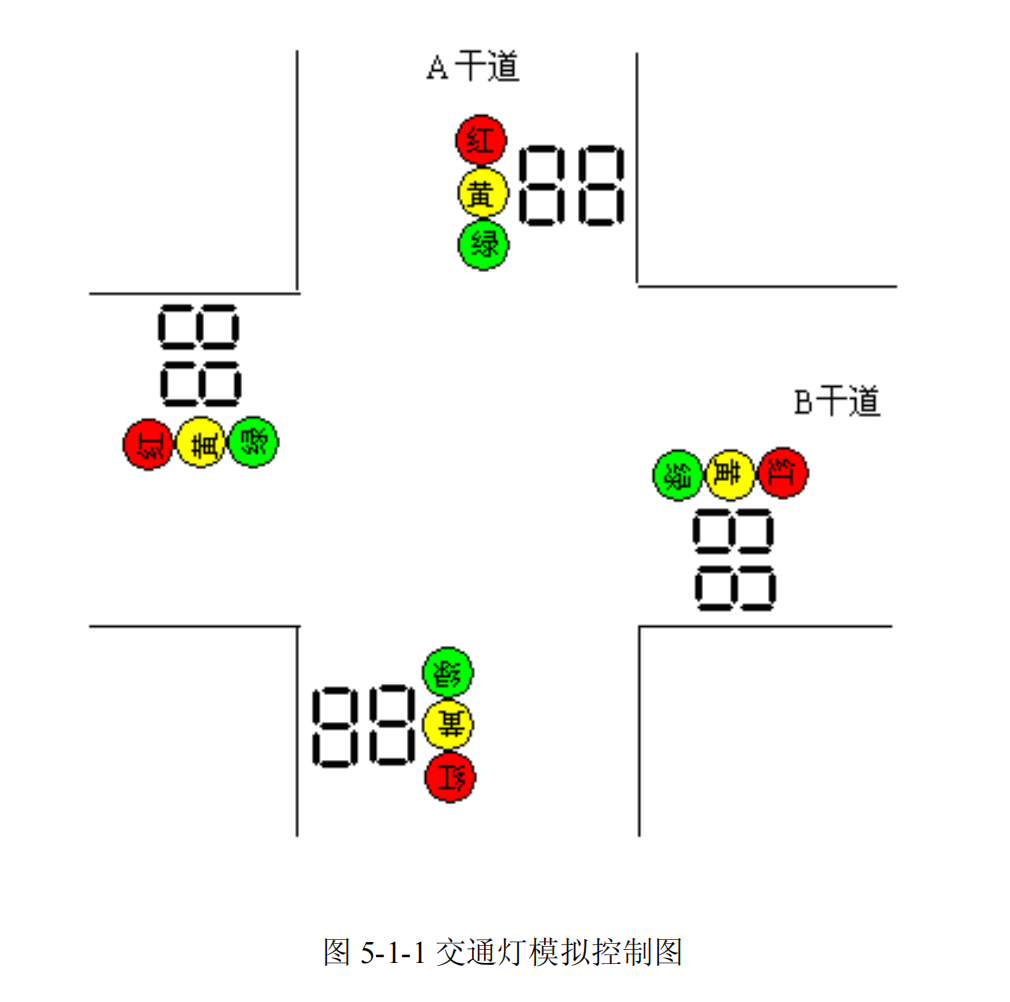
1. 掌握 CPU 与各芯片管脚连接方法，提高接口扩展硬件电路的连接能力。

2. 加深对定时器/计数器和并行接口芯片的工作方式和编程方法的理解。

3. 掌握交通信号灯自动控制系统的设计思路和实现方法。

1. **课程设计内容**

设计并实现十字路口通信号自动控制模拟指示系统。设该路口由 A、B 两条通行干道相交而成，四个路口各设一组红、黄、绿三色信号灯，用两位数码管作倒计时显示。



1. **系统功能与设计要求**
2. **基本功能要求**

（1）以秒为计时单位，两位数码管以十进制递减计数显示通行（绿灯）剩余时间，在递减计数回零瞬间转换。十字路口交通灯的变化规律及控制时序：

① 南北口的绿灯、东西路口的红灯同时亮 30 秒，同时南北路口数码管递减

显示绿灯剩余时间 30，29，28……0 秒。

② 南北路口的黄灯闪烁 5 秒钟，同时东西路口的红灯继续亮。

③ 南北路口的红灯、东西路口的绿灯同时亮 30 秒，同时东西路口数码管递

减显示绿灯剩余时间 30，29，28……0 秒。

④ 南北路口的红灯继续亮，同时东西路口的黄灯闪烁 5 秒钟。

⑤ 转① 重复。

（2）通过键盘可以对红、黄、绿三色信号灯所亮时间在 0～99 秒内任意设定。

（3）十字路口的通行起始状态可自行设定，系统启动后自动运行，按“Q”键退出。

**2. 发挥部分**

（1）增加人工干预模式。在特殊情况下可通过人工干预，手动控制 A,B 道路交通灯的切换时间，并可以随时切换为自动运行模式。

（2）增加夜间控制功能，交通灯在进入夜间模式后，A、B 两个干道上红、绿灯均不亮，黄色信号灯闪烁显示。

（3）增加两位红色信号灯倒计时显示。使系统同时显示 A,B 干道的红绿灯（通行/等待）时间。

**四、设计思路**

交通信号灯的亮灭时间及数码管显示时间可以通过计数/定时器（8254）来控制，8254的时钟源采用时钟信号发生器与分频电路提供，通过计算获得计数初值。按照需要设定工作方式。交通信号灯及数码管可以采用系统提供的相应模块，控制可以通过 8255 可编程并行接口，如：PA 口控制红黄绿交通灯的亮灭，PB 口和 PC 口控制时间显示数码管的段，位。也可使用基本并行 I/O 接口（74LS244，74LS273）。人工干预及夜间控制可以采用开关模块（K0~K7）进行模拟控制。

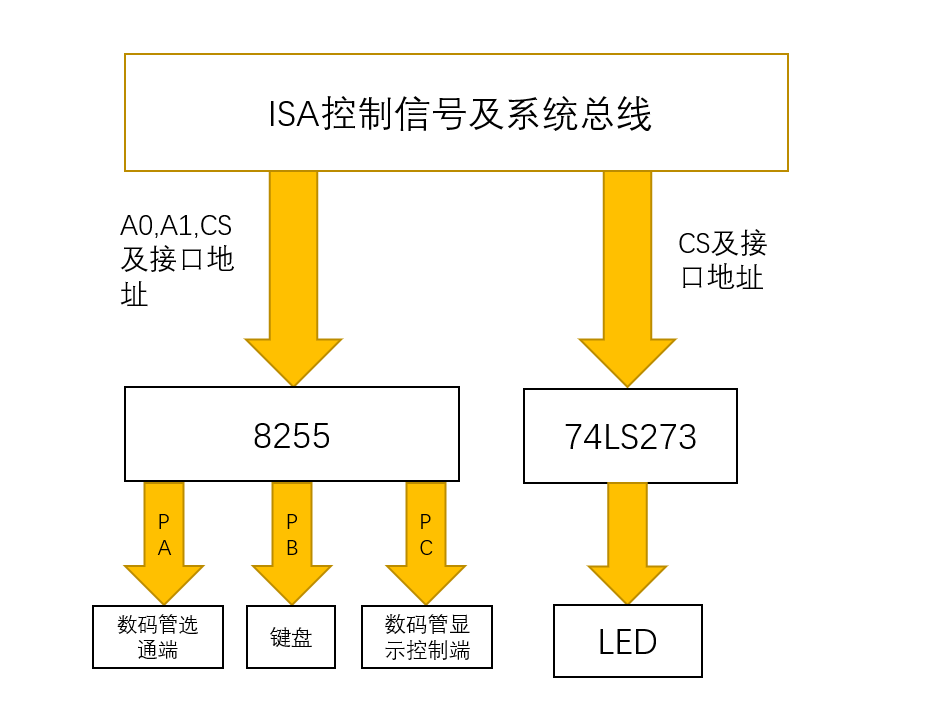
**五、操作步骤**

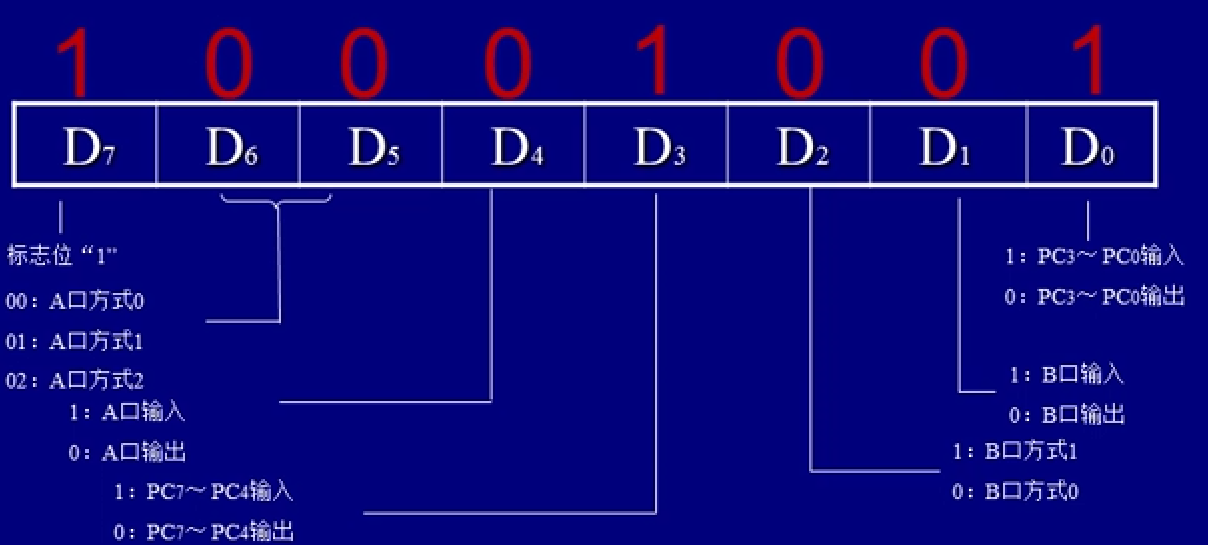
**1．硬件设计方案及硬件连线图**

* 1. **设计方案**

在本次实验中我们为了实现控制信号灯的切换及依序显示以及数码管显示倒计时和键盘控制，需要使用4个八位输入输出接口来输出控制信号并获取键盘读入状态。根据以上情况，我们决定使用74LS273基本并行 I/O 接口及8255 可编程并行接口实现上述功能。

* 1. **连线图**



* 1. **8255控制字的计算**

易知本次课程设计PA、PB、PC均工作在方式0，且PA、PC为输出端口，PB读取键盘输入，故而8255控制字计算为89H（10001001B）。

**2.软件设计方案及流程图**

**a) 设计方案**

红绿灯输出设置独立于其他模块，LED倒计时显示和键盘模块通过8255控制，通过delay函数实现延时和循环检测键盘读入实现手动控制，默认倒计时和红绿灯输出功能均为自动设置，无需人为干预。

**b) 具体实现思路：**

数码管部分：

使用SegArray[]数组存储数码管显示数字，使用buffer[8]函数存储显示内容缓存数据。实验中首先计算出倒计时，而后采用使用DIR()函数不断轮询扫描数码管，并将缓存数据逐步输出到数码管中。

键盘读入倒计时则循环采用Allkey()函数轮询判断按键闭合，通过key()函数得出按键闭合状态并计算数值，最终通过DIR()函数将数值输入到数码管中。

红绿灯及退出部分：

使用led\_data[]函数存储待显示数据，并通过delay函数设定延时，自动设置完毕后程序自动按指定循环输出显示内容。实验中同样使用Allkey()循环判断键盘是否闭合，读取退出按键后执行循环退出，return main函数。

1. **系统测试结果**

红绿灯输出及退出设置以及数码管显示部分均能正常进行，但两者连接部分因为接口复用和循环冲突问题难以解决。

最终解决方案：换用接口并重写循环函数，最终测试结果预期正常。

**七、实验心得**

本次微机原理与系统设计课程设计中，我们小组合作完成了交通信号灯自动控制模拟指示系统 。其中我负责红绿灯输出及键盘控制程序退出部分。

通过微机实验，让我对微机硬件等有了进一步的了解，同时，作为主编程的人员， 我也对c嵌入式编程及汇编语言有了更熟悉的掌握。在这次实验中，我认识到团队合作非常重要，有效的分工合作可以节省不少时间，避免不必要的麻烦，成员的冲突可能会导致项目时间大大浪费，也有可能促进新思路的诞生。此次课设中就因为分工太散，缺乏整合人员导致最后项目整合模块草草收场。因此，正确调度团队成员的分工，适当分配讨论时间非常重要。

总而言之，通过此次工程设计我将微机原理课的理论真正进行了实践，同时还提高了团队协作能力，c及汇编水平也得到进一步提高，感谢学校给我们提供这样的平台来提升我们的能力！

**八、附录：**

**课程设计实验环境：**

1. 硬件配置：

微机（Pentium 4） 一台

微机接口技术实验箱 一个

ISA–PCI 转接卡 一块

连接电缆 一条

微机接口技术实验讲义 一本

连接导线 若干条

2. 软件环境： Windows XP/2000/Win 7 平台 星研集成开发环境

**程序代码（主要修改的代码）：**

Traffic\_light.c

#define PA\_Addr 0x270

#define PB\_Addr 0x271

#define PC\_Addr 0x272

#define CON\_Addr 0x273

#define u8 unsigned char

#define u16 unsigned int

extern char inportb( unsigned int );

extern void outportb( unsigned int, char);

u8 delay(u16 ms)

{

u16 i;

while(ms--)

{

i = 10;

do

{if(Allkey()) return 1;}

while(--i);

}

return 0;

}

void realdelay(u16 ms)

{

u16 i;

while(ms--)

{

i = 100;

do

{;}while(--i);

}

}

u8 AllKey()

{

u8 i;

outportb(PB\_Addr, 0x0);

i = ~inportb(PC\_Addr);

return i;

}

void Quit()

{

outportb(PA\_Addr, 0xff);

outportb(PB\_Addr, 0xff);

}

void main()

{

u8 i, j, tag = 0;

u8 led\_data[] = {0xbe, //东西绿灯，南北红灯

0xbf, //东西黄灯闪烁，南北红灯

0xbd, //东西黄灯亮，南北红灯

0xeb, //东西红灯，南北绿灯

0xfb, //东西红灯，南北黄灯闪烁

0xdb}; //东西红灯，南北黄灯亮

outportb(CON\_Addr, 0x89);

outportb(PA\_Addr, 0xff);

while (1)

{

outportb(PA\_Addr, led\_data[(0+tag)%6]);

if(delay(5000)) {

Quit();

return;

}

for (i = 0; i < 6; i++)

{

outportb(PA\_Addr, led\_data[(1+tag)%6]);

if(delay(500)){

Quit();

return;

}

outportb(PA\_Addr, led\_data[(2+tag)%6]);

if(delay(500)){

Quit();

return;

}

}

outportb(PA\_Addr, led\_data[(3+tag)%6]);

if(delay(5000)){

Quit();

return;

}

for (i = 0; i < 6; i++)

{

outportb(PA\_Addr, led\_data[(4+tag)%6]);

if(delay(500)){

Quit();

return;

}

outportb(PA\_Addr, led\_data[(5+tag)%6]);

if(delay(500)){

Quit();

return;

}

}

}

}

Kd.c

#define PA\_Addr 0x270

#define PB\_Addr 0x271

#define PC\_Addr 0x272

#define CON\_Addr 0x273

#define u8 unsigned char

#define u16 unsigned int

extern char inportb( unsigned int ); //读I/O

extern void outportb( unsigned int, char);

u8 buffer[8] = {0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10}; //置显示缓冲器初值

u8 result[8] = {0};

u16 i=0;

u16 flag = 0;

//u8 number[16] = {0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f};//对应数字

//u8 result[6] //[0,1]绿[2,3]黄[4，5]红

void delay(u16 ms)

{

u16 i;

while(ms--)

{

i = 100;

do

{;}while(--i);

}

}

//扫描数码管

void DIR()

{

u8 i, dig = 0xfe;

u8 SegArray[]={0xC0,0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8,

0x80, 0x90, 0x88, 0x83, 0xC6, 0xA1, 0x86, 0x8E, 0xFF};

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

{

outportb(PA\_Addr, SegArray[buffer[i]]); //段数据

outportb(PB\_Addr, dig); //选择数码管

delay(2); //延迟3ms

outportb(PB\_Addr, 0xff);

dig = ((dig << 1) | 1);

}

}

//扫描数码管

void DIR1()

{

u8 i, dig = 0xfe;

u8 SegArray[]={0xC0,0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8,

0x80, 0x90, 0x88, 0x83, 0xC6, 0xA1, 0x86, 0x8E, 0xFF};

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

{

outportb(PA\_Addr, SegArray[buffer[i]]); //段数据

outportb(PB\_Addr, dig); //选择数码管

delay(500); //延迟3ms

outportb(PB\_Addr, 0xff);

dig = ((dig << 1) | 1);

}

}

u8 AllKey()

{

u8 i;

outportb(PB\_Addr, 0x0);

i = (~inportb(PC\_Addr) & 0x3);

return i;

}

u8 key()

{

u8 i, j, keyResult;

u8 bNoKey = 1;

while(bNoKey)

{

if (AllKey() == 0) //调用判有无闭合键函数

{

DIR();

DIR();

continue;

}

DIR();

DIR();

if (AllKey() == 0) //调用判有无闭合键函数

continue;

i = 0xfe;

keyResult = 0;

do

{

outportb(PB\_Addr, i);

j = ~inportb(PC\_Addr);

if (j & 3)

{

bNoKey = 0;

if (j & 2) //1行有键闭合

keyResult += 8;

}

else //没有键按下

{

keyResult++; //列计数器加1

i = ((i << 1) | 1);

}

}while(bNoKey && (i != 0xff));

}

if (!bNoKey)

{

while(AllKey()) //判断释放否

{

DIR();

}

}

return keyResult;

}

void main()

{

u8 num[8]={0x10};

u8 pos = 0;

outportb(CON\_Addr, 0x89);//PA、PB输出，PC输入

while(1) //pos是数码管每个数字的位置 此处循环为不断刷新数据

{

buffer[pos] = key(); //key（）返回当前数字的数据

if(pos==0||pos==1||pos==2||pos==3)

result[pos] = buffer[pos];

pos++;

pos &= 0x7; //mod8

if(buffer[pos-1] == 0x0F) break;

}

while(1){

for(i=0;i<2;i++){//使用result数组作为最终数组

num[i] = result[i];

}

while(1)

{

for(i=0;i<5;i++)

{

DIR();

DIR();

}

buffer[pos]=num[pos];

pos++;

pos &= 0x7;

if(pos==0)

{

if(num[0]!=0)

num[0]--;

else {

num[0]=9;

num[1]--;

}

if(num[0]==0&&num[1]==0)break;

//delay(1000);

}

}

for(i=0;i<2;i++){//使用result数组作为最终数组

num[i] = result[i+2];

}

while(1)

{

for(i=0;i<5;i++)

{

DIR();

DIR();

}

buffer[pos]=num[pos];

pos++;

pos &= 0x7;

if(pos==0)

{

if(num[0]!=0)

num[0]--;

else {

num[0]=9;

num[1]--;

}

}

if(num[0]==0&&num[1]==0)break;

}

/\* for(i=0;i<2;i++){//使用result数组作为最终数组

if((flag=result[i]+result[i+2])>10){

num[i] = result[i+2]+result[i]-10;num[i+1]++;

}

while(1)

{

for(i=0;i<5;i++)

{

DIR();

DIR();

}

buffer[pos]=num[pos];

pos++;

pos &= 0x7;

if(pos==0)

{

if(num[0]!=0)

num[0]--;

else {

num[0]=9;

num[1]--;

}

}

if(num[0]==0&&num[1]==0)break;

}\*/

}

}