1　实验一 跑马灯实验

1.1　实验目的与要求

熟悉星研集成环境软件或熟悉 Keil C51 集成环境软件的使用方法。

熟悉 MCS51 汇编指令，能自己编写简单的程序，控制硬件。

1.2　实验设备

SUN 系列实验仪一套、PC 机一台

1.3　实验内容

①熟悉星研集成环境软件或熟悉 Keil C51 集成环境软件的安装和使用方法。

②照接线图编写程序：使用 P1 口控制 F5 区的 8 个指示灯，循环点亮，瞬间只有一个灯亮。

③观察实验结果，验证程序是否正确。

1.4　实验原理图

1.5 实验步骤

①连线说明：注意：JP51 对应于 P1.0，P1.1，……P1.7 共用到了 8 个引脚

A3 区：JP51	——	F5 区：JP65

②编写程序或将参考程序中 3 个空白的地方填完整。

③实验结果：通过 F5 区的 LED 指示灯（8 个指示灯轮流点亮），观察实验的输出结果是否正确。

1.6 参考程序

```
        ORG     0000H
        LJMP    START       ;在 0000H 处放一条跳转指令，跳到 START 处

        ORG     0100H
START:  MOV     SP,#60H
        MOV     A,#0FFH
        CLR     C
START1: RL      A           ;连同符号位 C 进行循环左移
        MOV     P1,A
        ACALL   Delay
        SJMP    START1
Delay:  MOV     R5,#2       ;延时
Delay1: MOV     R6,#0
Delay2: MOV     R7,#0
        DJNZ    R7,$
        DJNZ    R6,Delay2   ;将 R6 中的数减 1，不为 0 就跳转
        DJNZ    R5,Delay1
        RET

        END
```

如果读者使用星研集成环境软件，请考虑以下问题？

1、运行程序前，打开变量窗;

2、使用单步进入命令，运行到第六行后，运行过程中变量窗有何变化？将鼠标停留在 A、SP 上一秒后，出现什么？,它与变量窗使用场合的区别？

3、第九行是调用延时子程序，如何进入延时子程序（使用单步进入命令）？如何将延时子程序一下子运行完毕（使用单步命令；也可以将光标移到下一行，使用运行到光标处命令;）？单步进入命令与单步命令有何区别？

4、运行几次后，在第十行设置一个断点，使用全速断点命令运行几次，观

察运行结果，它与运行到光标处命令有何区别？

5、Delay 是一个延时子程序，改变延时常数，使用全速运行命令，显示发生了什么变化？

6、观察寄存器，有哪几种方法？

1）在工作区窗的通用寄存器标签视中；

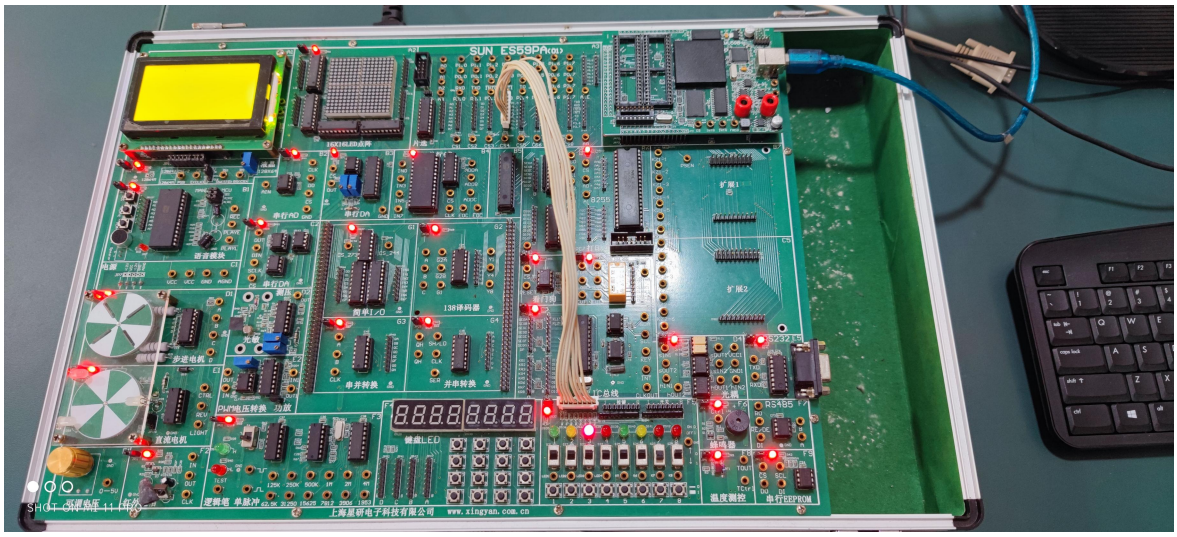
2）变量窗

3）鼠标停留在寄存器上

4）观察窗

5）寄存器窗

1.7 实验结果



2　实验二 74HC138 译码器实验

2.1　实验目的与要求

①掌握 74HC138 译码器的工作原理，熟悉 74HC138 译码器的具体运用连接方法，了解 74HC138 是如何译码的。

②认真预习本节实验内容，尝试自行编写程序，填写实验报告

2.2　实验设备

SUN 系列实验仪一套、PC 机一台

2.3　实验内容

①编写程序：使用单片机的 P1.0、P1.1、P1.2 控制 74HC138 的数据输入端，通过译码产生 8 选 1 个选通信号，轮流点亮 8 个 LED 指示灯。

②运行程序，验证译码的正确性。

2.4　实验原理图

2.5　实验步骤

①连线说明：注意：此处只用到了 P1.0，P1.1， P1.2 共 3 个引脚，这和上个实验不同的。

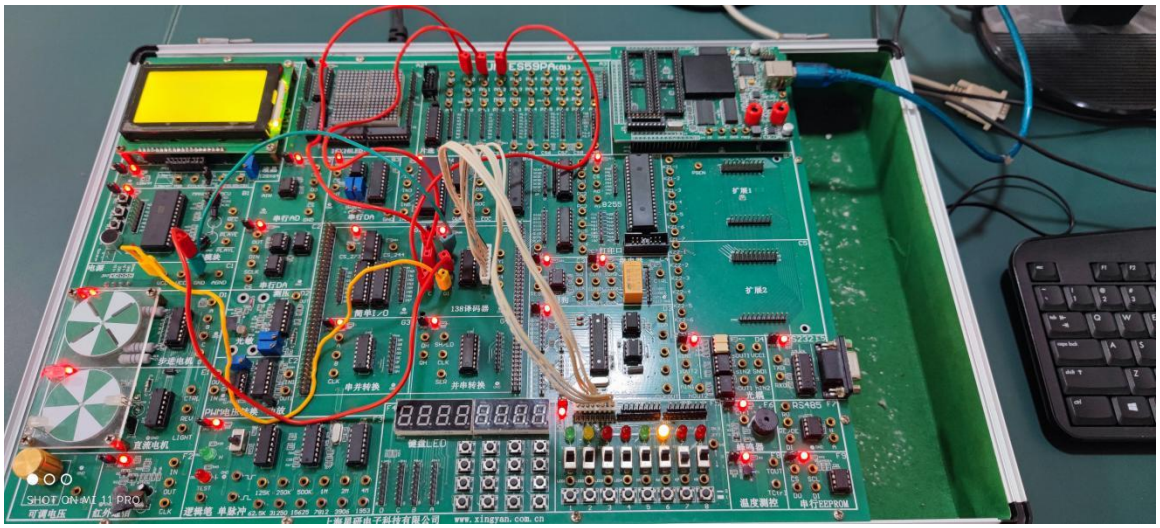
G2 区：A、B、C	——	A3 区：**P1.0、P1.1、P1.2**
G2 区：G1、G2A、G2B	——	C1 区：VCC、GND、GND
G2 区：JP35	——	F5 区：JP65（LED 指示灯）

②调试程序，查看运行结果是否正确。

2.6 演示程序

```
138 译码器实验（跑马灯），P1.0--A, P1.1---B, P1.2--C, /G2B--GND, /G2A--GND
            ORG     0000H
            LJMP    START

            ORG     0100H
START:      MOV     SP,#60H
            CLR     A               ;使累加器 A 清 0
START1:     MOV     P1,A            ;初值，第一次 0 位 LED 亮
            LCALL   DLTIME          ;调用延迟子程序 DLTIME
            INC     A
            CLR     ACC.3           ;使 ACC 的第 3 位为 0
            SJMP    START1
DLTIME:     MOV     R5,#20
DLTIME1:    MOV     R6,#100
DLTIME2:    MOV     R7,#100
            DJNZ    R7,$
            DJNZ    R6,DLTIME2
            DJNZ    R5,DLTIME1
            RET

            END
```

2.7 实验结果



2.8 实验扩展及思考

在单片机系统中，74HC138 通常用来产生片选信号，请读者考虑一下，应如何处理？

答：取三根底地址线，接到 38 译码器的输入端，译码产生，8 个使能控制信号。

3 实验三 PWM 电压转换实验（选做）

3.1 实验目的

①了解 PWM 电压转换原理

②掌握单片机控制的 PWM 电压转换

3.2 实验设备

SUN 系列实验仪一套、PC 机一台

3.3 实验内容

①PWM 电压转换原理：

(1)将一定频率的输入信号转换为直流电；

(2)通过调节输入信号占空比调节输出的直流电电压，输出电压随着占空比增大而减小

②实验过程

(1)输入 15kHZ 左右的方波，经 LM358 进行 PWM 电压转换，输出直流电，驱动直流电机

(2)通过按键调整占空比来改变 PWM 输出电压，直流电机的转速会随之变化

3.4 实验原理图

3.5 实验步骤

①连线说明：

E2 区：IN	——	A3 区：P1.2，方波输入
E2 区：OUT	——	E2 区：IN1
E2 区：OUT1	——	E1 区：CTRL，直流电机电源输入
A3 区：JP51	——	F5 区：JP74

②通过 F5 区的 1、2 键调整占空比来改变 PWM 输出电压，直流电机的转速会随之变化：

1 号键减少占空比；2 号键增加占空比

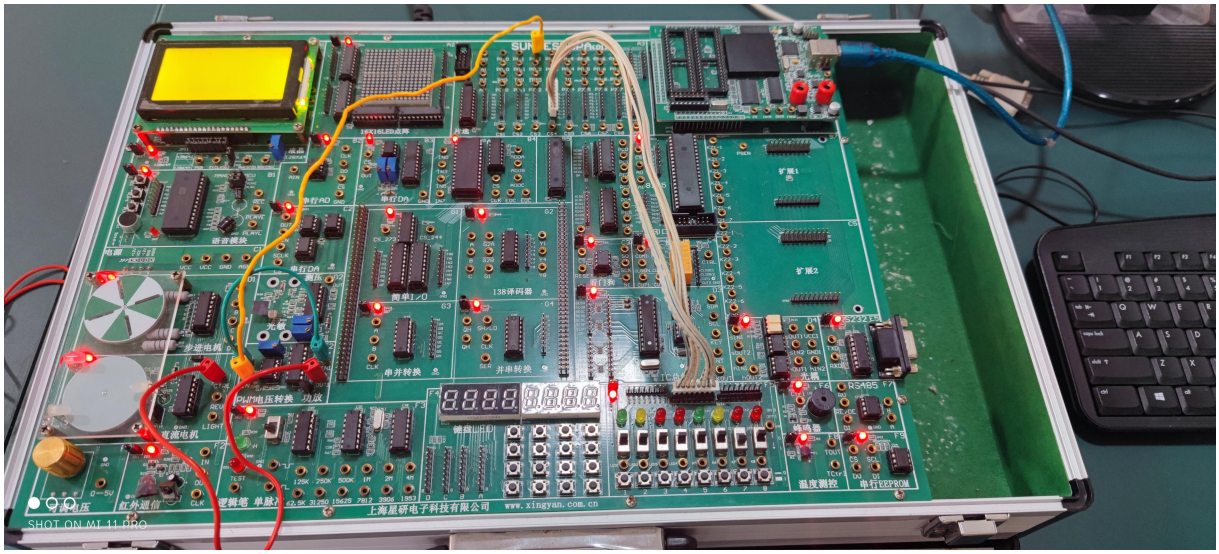
3.6 演示程序

```
IN          BIT     P1.2          ;PWM 方波输入,定义 IN 代表 P1.2
PWM_LOW     DATA    30H           ;低电平时间
PWM_HIGH    DATA    31H          ;高电平时间,控制频率在 15kHZ 左右
periods     EQU     0E0H          ;周期 64us

            ORG     0000H
```

```
                LJMP        START

                ORG         000BH
                LJMP        iTIMER0

                ORG         0100H
START:          MOV         SP,#60H
                MOV         PWM_LOW,#periods
                MOV         PWM_HIGH,#periods
                MOV         TH0,#periods
                MOV         TL0,#periods
                MOV         TMOD,#02H
                SETB        EA                  ; 使 EA 置 1
                SETB        ET0
                SETB        TR0
START1:         ACALL       ScanKey
                JNZ         Key1
Key0:           MOV         A,PWM_HIGH          ; 增加占空比
                CJNE        A,#0FBH,Key0_1
                SJMP        START1          ;大于这个值，对定时中断已反应不过来
Key0_1:         INC         PWM_HIGH
                DEC         PWM_LOW             ;低电平时间减 1，从而增加占空
                                                ;比
                SJMP        START1
Key1:           MOV         A,PWM_LOW           ;减少占空比
                CJNE        A,#0FBH,Key1_1
                SJMP        START1          ;大于这个值，对定时中断已反应不过来
Key1_1:         INC         PWM_LOW             ;低电平时间加 1，从而减少占空
                                                ;比
                DEC         PWM_HIGH
                SJMP        START1
iTIMER0:        JBC         IN,iTIMER0_1
                MOV         TL0,PWM_HIGH
                SETB        IN
                RETI
iTIMER0_1:      MOV         TL0,PWM_LOW
                NOP
                RETI
ScanKey:        JNB         P1.0, ScanKey1      ;键扫描
                JB          P1.1, ScanKey
ScanKey1:       ACALL       Delay20ms           ;消抖动
                ACALL       Delay20ms
                JNB         P1.0, ScanKey2
                JB          P1.1, ScanKey
```

```
                MOV     A,#1
                SJMP    ScanKey3
ScanKey2:       CLR     A
ScanKey3:       JNB     P1.0,$
                JNB     P1.1,$
                RET
Delay20ms:      MOV     R6,#10
Delay1:         MOV     R7,#100
                DJNZ    R7,$
                DJNZ    R6,Delay1
                RET

                END
```

3.7 实验结果



3.8 实验扩展及思考

①改变 PWM 的输入频率，使用示波器观看 LM358 的输出，由此加深对 PWM 电压转换的了解。

4　实验四 8255 控制交通灯实验

4.1　实验目的与要求

①了解 8255 芯片的工作原理，熟悉其初始化编程方法以及输入、输出程序设计技巧。学会使用 8255 并行接口芯片实现各种控制功能，如本实验（控制交通灯）等。

②熟悉 8255 内部结构和与单片机的接口逻辑，熟悉 8255 芯片的 3 种工作方式以及控制字格式。

③认真预习本节实验内容，尝试自行编写程序，填写实验报告。

4.2　实验设备

SUN 系列实验仪一套、PC 机一台

4.3　实验内容

①编写程序：使用 8255 的 PA0..2、PA4..6 控制 LED 指示灯，实现交通灯功能。

②连接线路验证 8255 的功能，熟悉它的使用方法。

4.4　实验原理图

4.5 实验步骤

①连线说明：

B6 区：CS、A0、A1	——	A3 区：CS1、A0、A1
B6 区：JP56（PA 口）	——	F5 区：JP65

②观察实验结果，是否能看到模拟的交通灯控制过程。

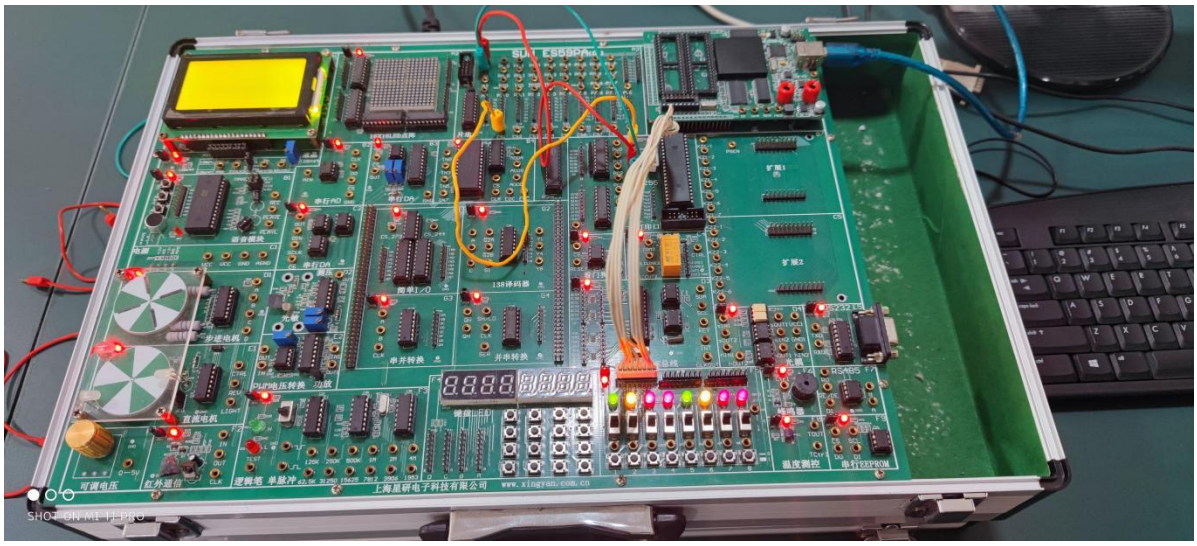
4.6 演示程序

```
COM_ADD     XDATA    0F003H
PA_ADD      XDATA    0F000H
PB_ADD      XDATA    0F001H
PC_ADD      XDATA    0F002H
            ORG      0000H
            LJMP     STAR
            ORG      0100H
STAR:       MOV      SP,#60H
            MOV      DPTR,#COM_ADD
            MOV      A,#80H          ;PA、PB、PC 为基本输出模式
            MOVX     @DPTR,A
            MOV      DPTR,#PA_ADD    ;灯全熄灭
```

```
        MOV     A,#0FFH
        MOVX    @DPTR,A         ;将累加器 A 的值送到片外 RAM 中
START1: MOV     A,#37H
        MOVC    A,@A+PC
        MOVX    @DPTR,A         ;东西绿灯，南北红灯
        LCALL   DL5S            ;调用延时 5S 的子程序
        MOV     R4,#6
START2: MOV     A,#30H
        MOVC    A,@A+PC
        MOVX    @DPTR,A         ;东西绿灯闪烁，南北红灯
        ACALL   DL500ms
        MOV     A,#29H
        MOVC    A,@A+PC
        MOVX    @DPTR,A
        ACALL   DL500ms
        DJNZ    R4,START2
        MOV     A,#23H          ;东西黄灯亮，南北红灯
        MOVC    A,@A+PC
        MOVX    @DPTR,A
        ACALL   DL3S
        MOV     A,#1EH          ;东西红灯，南北绿灯
        MOVC    A,@A+PC
        MOVX    @DPTR,A
        ACALL   DL5S
        MOV     R4,#6
START3: MOV     A,#17H          ;东西红灯，南北绿灯闪烁
        MOVC    A,@A+PC
        MOVX    @DPTR,A
        ACALL   DL500ms
        MOV     A,#10H
        MOVC    A,@A+PC
        MOVX    @DPTR,A
        ACALL   DL500ms
        DJNZ    R4,START3
        MOV     A,#0AH          ;东西红灯，南北黄灯亮
        MOVC    A,@A+PC
        MOVX    @DPTR,A
        ACALL   DL3S
        SJMP    START1          ;跳转到 START1 处
        DB      01111110B       ;东西绿灯，南北红灯
        DB      11111110B       ;东西绿灯闪烁，南北红灯
        DB      10111110B       ;东西黄灯亮，南北红灯
        DB      11011011B       ;东西红灯，南北绿灯
        DB      11011111B       ;东西红灯，南北绿灯闪烁
```

```
                DB          11011101B       ;东西红灯，南北黄灯亮
DL500ms:        MOV         R5,#25
DL500ms1:       MOV         R6,#100
DL500ms2:       MOV         R7,#100
                DJNZ        R7,$
                DJNZ        R6,DL500ms2     ;将 R6 减 1，不为 0 就跳转。
                DJNZ        R5,DL500ms1
                RET
DL3S:           MOV         R4,#6
DL3S1:          LCALL       DL500ms
                DJNZ        R4,DL5S1
                RET
DL5S:           MOV         R4,#10
DL5S1:          LCALL       DL500ms
                DJNZ        R4,DL5S1
                RET

                END
```

4.7 实验结果



4.8 实验扩展及思考

①如何对 8255 的 PC 口进行位操作？

答:8255 控制字 D7=1 时，D6~DO 为口模式控制，D7=O 时控制字为 PC 口的位操作模式，如控制字=00H，PCO 复位；控制字=01H，PCO 置位;控制字=OEH，PC7 复位;控制字=OFH，PC7 置位；其中 D6~D4 没定义，D3~D1 从 000B~111B

分别指定 PCO~PC7 脚，DO 位为相应的电平高低，0 为低电平，1 为高电平。

4.9 实验总结

通过本次试验，我了解了 8255 芯片的工作原理，熟悉其初始化编程方法以及输入、输出程序设计技巧。学会使用 8255 并行接口芯片实现控制交通灯。熟悉 8255 内部结构和与 8088 的接口逻辑，熟悉 8255 芯片的 3 种工作方式以及控制字格式。

5　实验五 键盘、数码块实验（选做）

5.1　实验目的与要求

①进一步掌握 8255 的设计、编程方法。

②掌握矩阵键盘的扫描方法

③掌握动态扫描数码块的方法

④认真预习，做好实验前的准备工作，填写实验报告

5.2　实验设备

SUN 系列实验仪一套、PC 机一台

5.3　实验内容

①编写程序：扫描键盘，如有按键，键号显示于数码管。

②连接线路，验证 8255 的功能，熟悉它的使用方法。

5.4　实验原理图

5.5　实验步骤

①连线说明：

B6 区：CS、A0、A1	——	A3 区：CS1、A0、A1
B6 区：JP53(PB 口)、JP75(B)、JP79(C)、JP52(PC 口)	——	F4 区：A、B、C、D

②运行程序，观察实验结果（任意按下 F4 区 4X4 键盘几个键，它上面的 8 个 LED 显示器会将按键的编码从左至右依次显示出来），可依此验证对程序的正确性。

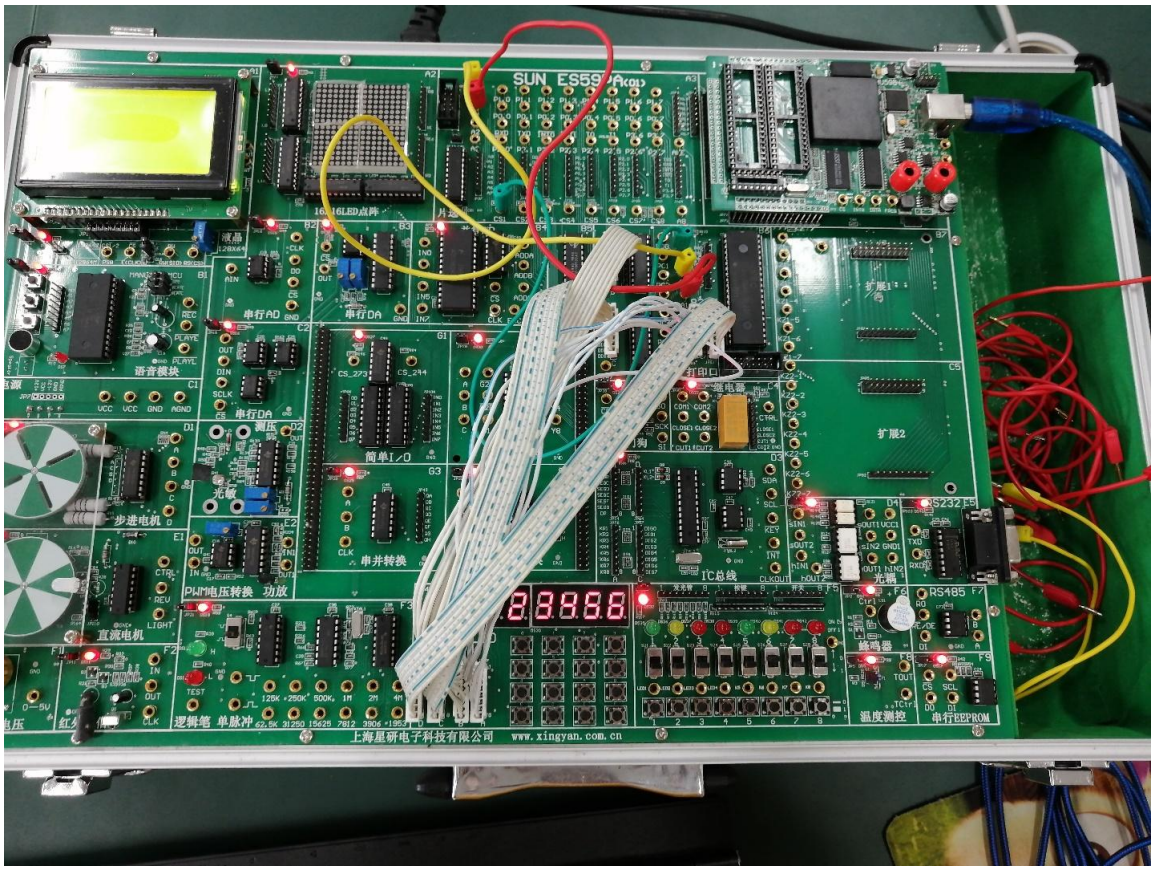
5.6 演示程序

```
#define u8    unsigned char
xdata u8 COM_8255 _at_ 0xF003;
xdata u8 PA_8255 _at_ 0xF000;
xdata u8 PB_8255 _at_ 0xF001;
xdata u8 PC_8255 _at_ 0xF002;

u8 buffer[8];                    //8 个字节显示缓冲区

void DL1()
{
    u8 i,j;
    i = 0x2;
    do
    {
        j = 250;
        while(--j)
        {;}
    }while(--i);
}

code const u8 SegArray[] =
{0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90,0x88,0x83,0xc6,0xa1,0x86,0
x8e,0xff};
void DIR()
{
    u8 i = 0xfe;
    u8* pBuffer = buffer;
    while(i != 0xff)
    {
        PA_8255 = SegArray[*pBuffer++];    //段数据->8255 PA 口
        PB_8255 = i;                       //扫描模式->8255 PB 口
        DL1();                             //延迟 1ms
        i = ((i << 1) | 0x1);
```

```
        PB_8255 = 0xff;
    }
}

u8 AllKey()
{
    PB_8255 = 0x0;                      //全"0"->扫描口
    return ~PC_8255 & 0x3;              //读键状态, 取低二位
}

u8 keyi()
{
    u8 i,j;
    while (1)
    {
        if (AllKey() == 0)              //调用判有无闭合键子程序
        {
            DIR();
            DIR();                      //调用显示子程序,延迟 6ms
            continue;
        }
        DIR();
        DIR();
        if (AllKey() == 0)              //调用判有无闭合键子程序
        {
            DIR();
            continue;
        }
        i = 0xfe;
        j = 0;
        while(i != 0xff)
        {
            PB_8255 = i;
            if ((PC_8255 & 0x1) == 0)
            {                                   //0 行有键闭合
                break;
            }
            else if ((PC_8255 & 0x2) == 0)
            {                                   //1 行有键闭合
                j += 8;
                break;
            }
            j++;                        //列计数器加 1
            i = ((i<<1) | 1);
```

```
        }
        if (i == 0xff)                          //完成一次扫描,没有键按下
            continue;
        do
        {
            DIR();
        }while(AllKey() != 0);                  //判断释放否
        return j;                               //键号
    }
}

void main()
{
    u8 i;
    COM_8255 = 0x89;                            //PA、PB 输出，PC 输入
    for (i = 0; i < 8; i++)
        buffer[i] = 0x10;                       //0x10-消隐
    DIR();
    while(1)
    {
        for (i = 0; i < 8; i++)
            buffer[i] = keyi();
    }
}
```

5.7 实验结果



5.8 实验扩展及思考

①显示程序中延时函数起什么作用？如何调节数码块亮度？

答：在每段程序后面，都有一个延时和消影。消影的主要作用是避免下段代码受到干扰，而延时的作用是区分显示和消影，避免无法观察的数字的现象。调节数码块亮度：增大电压。