# 微机原理实验

# 指导书

商丘师范学院物理与信息工程系编

# 目　录

# 实验一 二进制多位加法运算实验

## 一、实验目的

（1）学习使用加法类运算指令编程及调试方法

（2）掌握加法类指令对状态标志位的影响

## 二、实验仪器

微机、微机原理接口实验仪

## 三、实验原理

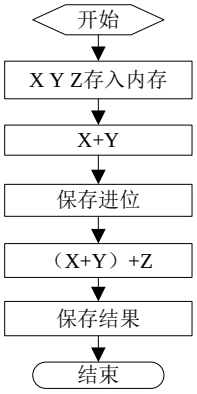
计算 X＋Y＋Z＝？（X、Y、Z 为 16 位二制数）。

由于本实验是三个十六位二制数相加运算，因此，当 X＋Y 时要考虑用 ADC 指令，把进位 C 加到结果的高 16 位中，当（X＋Y）＋Z 时，再把进位 C 加到结果的高 16 位中，本实验设定三个加数 0FFFFH，计算结果应为 2FFFDH。

数据 X、Y、Z 存放在内存 4000H~4005H 单元中。

运算结果保留在内存 4100H～4102H 单元中，其中 4102H 单元存放进位。

程序流程图：




实验程序：

```
CODE SEGMENT                ；程序段开始
        ASSUME CS:CODE      ；定义 CS 为程序段段寄存器
        ORG 2CA0H           ；程序从存储器地址 2CA0H 开始存放
START: CLC                  ；清除进位标志
        MOV SI，4000H       ；把数 4000H 存入 SI 寄存器
        MOV [SI]，0ffffH    ；被加数 0ffffH（X）存入 SI 指定的存储器单元（4000H、4001H）
        MOV [SI+2]，0ffffH  ；加数 0ffffH（Y）存入 SI+2 指定的存储器单元（4002H、4003H）
        MOV [SI+4]，0ffffH  ；加数 0ffffH（Z）存入 SI+4 指定的存储器单元（4004H、4005H）
        MOV AX，0000H       ；清除和的进位存储单元（4102H 单元=0）
        MOV [SI+102H]，AX   ；
        MOV AX，[SI]        ；从存储器中取出被加数（X），
        ADD AX，[SI+2]      ；从存储器中取出加数（Y），实现 X+Y，和在 AX 寄存器中，  进位在 CY 中
```

```
ADC [SI+102H]，0000    ；把进位存入 4102H 单元，[SI+102H]+0+CY
ADD AX，[SI+4]         ；（X+Y）+Z
MOV [SI+100H]，AX      ；X+Y+Z 和的低 16 位存入 4100H、4101H 中
ADC [SI+102H]，0000    ；（X+Y）+Z 和的进位存入 4102H 单元中
JMP $                  ；循环执行 JMP 指令，程序完成
CODE ENDS              ；程序段结束
END START               ；程序结束
```

## 四、实验步骤

（1）在 PC 机和实验系统联机状态下，运行该实验程序，可用鼠标左键单击菜单栏“文件”或工具栏“打开图标”，弹出“打开文件”的对话框，然后打开 8kAsm 文件夹，点击 S1.ASM 文件，单击“确定”即可装入源文件，再单击工具栏中编译，即可完成源文件自动编译、装载目标代码功能，再单击“调试”中“连续运行”或工具图标运行，即开始运行程序。

（2）运算结果保留在内存 4100H～4102H 单元中，点击 DICE-8086K 软件中存贮器 RAM 窗口，输入 RAM 的起始地址 4100、4101、4102，单元内容应为 FD、FF、02。

（3）连续运行程序，从程序数据存储器相应地址查看程序代码、运算数据、运算结果。

（4）修改运算数据 X=10000、Y=20000、Z=30000，单步运行程序，观察寄存器窗口、查看运行结果。

## 五、问题思考

（1）能不能直接把变量 X、Y、Z 放入内存？

（2）怎样修改运算数据？

（3）查看运行结果时应注意什么问题？

# 实验二 数码显示实验

## 一、实验目的

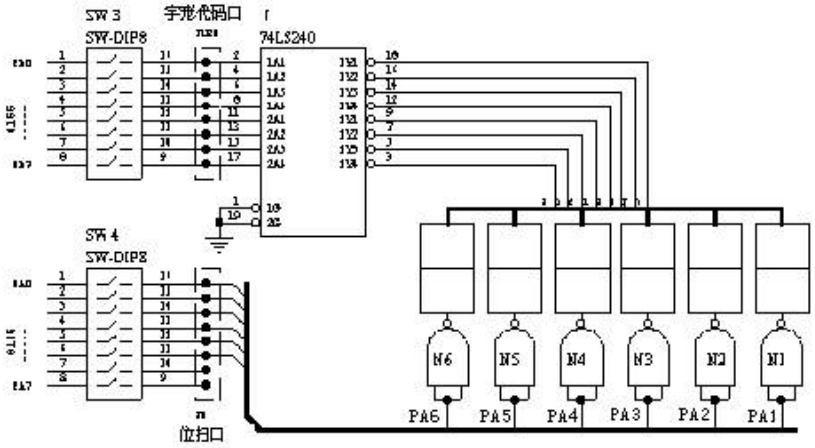
（1）了解 LED 数码管动态显示的工作原理及编程方法。

## 二、实验仪器

微机、微机原理接口实验仪

## 三、实验原理

实验箱数码显示电路如图所示，驱动数码管需要两个条件，一是通过字形代码端口输出字形代码，二是通过位型代码端口输出位型代码。

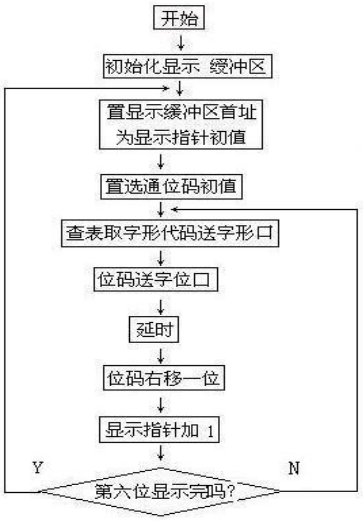



字形代码放在字形代码表中，使用 DB 指令建立表格，位型代码由右移指令实现。

显示数据由内存中的显示缓冲区提供，经过查表指令取出字形显示代码输出显示，

采用使用动态显示方式，循环输出显示代码。

程序流程图：



实验程序：

```
CODE    SEGMENT              ；程序段开始
ASSUME CS：CODE              ；定义 CS 为程序段段寄存器
        ORG 2DF0H            ；程序从存储器地址 2DF0H 开始存放
START： JMP START0           ；跳到程序开始处
  PA    EQU 0FF21H           ；把 PA 字符定义给字位口地址 0FF21H，使程序中的 PA 代表 0FF21H
  PB    EQU 0FF22H           ；把 PB 字符定义给字形口地址 0FF22H
  PC    EQU 0FF23H           ；把 PC 字符定义给键入口地址 0FF23H
  BUF   DB ?，?，?，?，?，?；在存储器中预定 6 个存储单元，作为 6 位数码管的显示缓冲区
data1:  db 0c0h,0f9h,0a4h,0b0h,99h,92h,82h,0f8h,80h,90h,88h,83h,0c6h,0a1h；定义显示字型码
        db 86h,8eh,0ffh,0ch,89h,0deh,0c7h,8ch,0f3h,0bfh,8FH
START0：CALL BUF1            ；程序开始，调用 BUF1 子程序
CON1:   CALL DISP            ；调用显示子程序
        JMP CON1             ；跳转到 CON1，进行主程序循环
DISP:   MOV AL，0FFH         ；把 0FFH 放入 AX 寄存器的低 8 位 AL 寄存器
        MOV DX，PA           ；字位口地址 0FF21H 存入 DX 寄存器
        OUT DX，AL           ；把 0FFH 送到字位口，准备把所有数码管熄灭
        MOV CL，0DFH         ；6 位数码管最高位的字位码存入 CL 寄存器
        MOV BX，OFFSET BUF；把显示缓冲区首地址存入 BX 寄存器
DIS1:   MOV AL，[BX]         ；从显示缓冲区中取出一个显示码存入 AX 的低 8 位 AL
        MOV AH，00H          ；AX 寄存器高八位清零
        PUSH BX              ；把 BX 寄存器的值存入堆栈存储器，以用于新的场合
        MOV BX，OFFSET DATA1；把字型码表的首地址存入 BX 寄存器
        ADD BX，AX           ；显示码+字型码表首地址=表内单元地址
        MOV AL，[BX]         ；查表，把某个显示码的字型码从存储器取出
        POP BX               ；恢复原来 BX 的值
        MOV DX，PB           ；字形口地址 0FF22H 存入 DX 寄存器
        OUT DX，AL           ；把字型码输出到字形口地址 0FF22H
        MOV AL，CL           ；把字位码存入 AL 寄存器
        MOV DX，PA           ；字位口地址 0FF21H 存入 DX 寄存器
        OUT DX，AL           ；把字位码输出到字位口地址 0FF21H
        PUSH CX              ；把 CX 寄存器的值存入堆栈存储器，以用于新的场合
DIS2:   MOV CX，00A0H        ；把数 00A0H 存入寄存器 CX，作为定时常数
        LOOP $               ；如果 CX—1≠0，重复执行指令 LOOP，延时
        POP CX               ；恢复原来 CX 的值
        CMP CL，0FEH         ；比较 CL 寄存器的值是否等于 0FEH
        JZ LX1               ；等于则跳转到 LX1 处，6 位数码管显示完毕
        INC BX               ；没有显示完就让 BX+1，指向显示缓冲区的下一位
        ROR CL，1            ；循环右移 CL 寄存器的值，把显示位形码修改成下一位
        JMP DIS1             ；跳转到 DIS1
   LX1: MOV AL，0FFH         ；熄灭所有数码管
        MOV DX，PB
        OUT DX，AL
        RET                  ；子程序返回
```

```
BUF1:    MOV BUF，0DH     ；把 DICE88 显示码存入显示缓冲区子程序
         MOV BUF+1，01H
         MOV BUF+2，0CH
         MOV BUF+3，0EH
         MOV BUF+4，08H
         MOV BUF+5，08H
         RET
;-------------------------------------------------------
         CODE ENDS        ；程序段结束
         END START        ；程序结束
```

## 四、实验步骤

（1）在 PC 机和实验系统联机状态下，运行该实验程序，可用鼠标左键单击菜单栏“文件”或工具栏“打开图标”，弹出“打开文件”的对话框，然后打开 8kAsm 文件夹，点击 S6.ASM 文件，单击“确定”即可装入源文件，再单击工具栏中编译，即可完成源文件自动编译、装载目标代码功能，再单击“调试”中“连续运行”或工具图标运行，即开始运行程序。

（2）数码管显示“DICE　88”字样。

（3）连续运行程序，从程序数据存储器相应地址查看程序代码、显示缓冲区、字型码表。

（4）修改参数，使数码管显示 2008 字样，单步运行程序，观察寄存器窗口、查看运行结果。

## 五、问题思考

（1）分析程序中 BX、DX 寄存器的作用。

（2）怎样控制数码管循环点亮？

# 实验三 数据块移动实验

## 一、实验目的

（1）了解内存中数据块移动方法

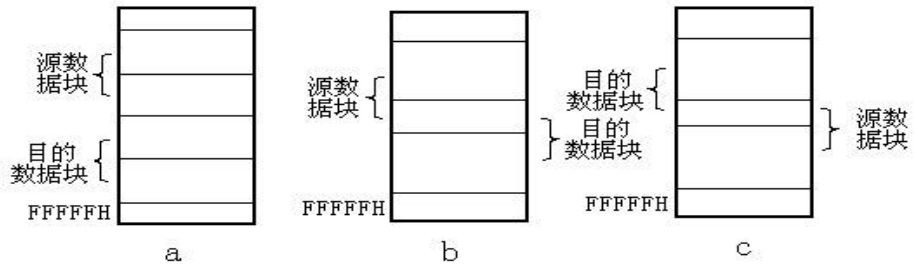
（2）掌握分支程序的设计方法

## 二、实验仪器

微机、微机原理接口实验仪

## 三、实验原理

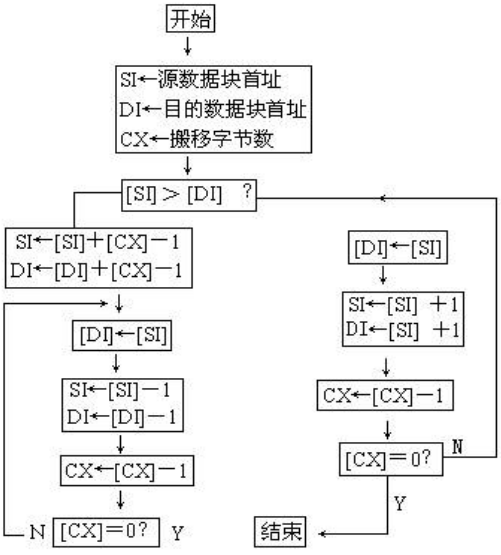
程序要求把内存中一数据区（称为源数据块）传送到内存另一数据区（称为目的数据块）。源数据块和目的数据块在存贮中可能有三种情况，如下图所示。




对于两个数据块分离的情况，如图（a），数据的传送从据块的首址开始，或者从数据块的末址开始均可。但对于有部分重叠的情况，则要加以分析，否则重叠部分会因“搬移”而遭破坏。

可以得出如下结论：当源数据块首址大于目的块首址时，从数据块首地址开始传送数据。当源数据块首址小于目的块首址时，从数据块末址开始传送数据。

程序流程图：




实验程序：

```
CODE SEGMENT                ；程序段开始
        ASSUME  CS：CODE    ；定义 CS 为程序段段寄存器
        ORG 2EF0H           ；程序从存储器地址 2EF0H 开始存放
START:  MOV CX，0100H       ；数据传送个数 0100H 存放 CX 寄存器
        MOV SI，4000H       ；源地址 4000H 存入源地址寄存器 SI
        MOV DI，4100H       ；目的地址 4100H 存入目的地址寄存器 DI
        CMP SI，DI          ；比较 SI 和 DI 的值的大小，判断
        JA FADR             ；SI>DI 则跳转到 FADR 处，从地址的头部开始传送数        据
        ADD SI，CX          ；SI＜DI 则地址的尾部开始传送数据，SI+CX 形成尾部源地址
        ADD DI，CX          ；DI+CX 形成尾部目的地址
        DEC SI              ；指向第一个数据源地址
        DEC DI              ；指向第一个数据目的地址
CON1:    MOV AL，[SI]       ；从源地址取出一个数据存入 AL 寄存器，地址减量传送
        MOV [DI]，AL        ；把数据存入目的地址
        DEC SI              ；减量修改源地址指针
        DEC  DI             ；减量修改目的地址指针
        DEC  CX             ；修改数据个数
        JNE  CON1           ；标志 ZF≠0（CX—1≠0）则跳转到 CON1，移动下一个数据
        JMP $               ；标志 ZF=0（CX—1=0）则传送结束
FADR:   MOV AL，[SI]        ；从源地址取出一个数据存入 AL 寄存器，地址增量传送
        MOV [DI]，AL        ；把数据存入目的地址
        INC SI              ；增量修改源地址指针
        INC DI              ；增量修改目的地址指针
        DEC CX              ；修改数据个数
        JNE FADR            ；标志 ZF≠0（CX—1≠0）则跳转到 CON1，移动下一个数据
        JMP $               ；标志 ZF=0（CX—1=0）则传送结束
CODE    ENDS                ；程序段结束
        END START           ；程序结束
```

## 四、实验步骤

(1)在源数据块 4000H～40FFH 中首址、末址几个单元，填入几个标志性字节。

(2)在 PC 机和实验系统联机状态下，运行该实验程序，可用鼠标左键单击菜单栏“文件”或工具栏“打开图标”，弹出“打开文件”的对话框，然后打开 8kAsm 文件夹，点击 S8.ASM 文件，单击“确定”即可装入源文件，再单击工具栏中编译，即可完成源文件自动编译、装载目标代码功能，再单击“调试”中“连续运行”或工具图标运行，即开始运行程序。

(3)复位 RST 键，查看目的数据块 4100H～41FFH 数据是否和源数据块 4000H～40FFH 单元相一致。

(4)修改目的地址为 4200H~42FFH，单步执行程序，并查看运行结果。

## 五、问题思考

(1) 程序是怎样实现循环控制的？

(2) 分析 SI、DI 寄存器的作用。

# 实验四 多分支程序实验

## 一、实验目的

（1）掌握程序散转的方法，实现程序的多分支转移

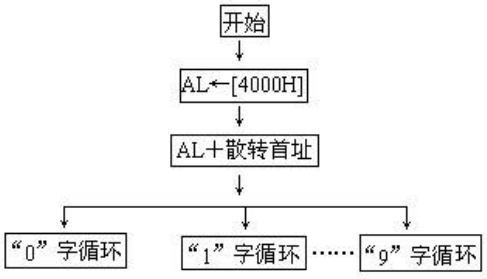
## 二、实验仪器

微机、微机原理接口实验仪

## 三、实验原理

多分支程序一般用于实现键盘功能的实现，当按键的键值被读入内存指定单元后，为实现键的功能需要根据键值跳往对应程序完成相应功能。方法是把每个程序的首地址先存放在一个表格中，利用键值查表把查得的值作为地址再利用跳转指令就可实现。

程序流程图：




实验程序：

```
CODE SEGMENT                    ；程序段开始
     ASSUME  CS:CODE            ；定义 CS 为程序段段寄存器
     ORG 2F40H                  ；程序从存储器地址 2F40H 开始存放
START:  JMP START0              ；程序开始
ADDR    DW  DP0,DP1,DP2,DP3,DP4,DP5,DP6,DP7,DP8,DP9；把程序段 DP0、DP1 等地址的值放入表格
START0: MOV SI，4000H           ；存储器 4000H 处存放分支代码（比如键盘送来的数字）
        MOV AL，[SI]            ；取出分支代码，存入 AL 寄存器
        SUB  AH，AH             ；清除 AH 寄存器的值，AH、AL 构成 AX
        SHL  AL，1              ；AL 寄存器的值左移一位
        MOV BX，OFFSET ADDR     ；把表格首地址放入 BX 寄存器
        ADD BX，AX              ；形成分支地址
        JMP [BX]                ；跳转到分支代码指定的程序段执行程序
DP0:    MOV BL，0c0H;DISP "0"   ；DP0 程序段，把显示 0 的字型码放入显示子程序入口参数 BL 寄存器
        JMP disp                ；跳转到显示子程序
DP1:    MOV BL，0f9 H ;DISP "1"
        JMP disp
DP2:    MOV BL，0a4 H ;DISP "2"
        JMP disp
```

```
DP3:    MOV BL，0b0 H ;DISP "3"
        JMP disp
DP4:    MOV BL，99 H ;DISP "4"
        JMP disp
DP5:    MOV BL，92 H ;DISP "5"
        JMP disp
DP6:    MOV BL，82 H ;DISP "6"
        JMP disp
DP7:    MOV BL，0f8 H ;DISP "7"
        JMP disp
DP8:    MOV BL，80 H ;DISP "8"
        JMP disp
DP9:    MOV BL，90 H ;DISP "9"
        JMP disp
;---------------------------------------
disp:   MOV AH，0DF H           ；把数码管最高位显示位形码放入 AH
disp0:  MOV DX，0ff22 H         ；把字形码端口地址放入 DX 寄存器
        MOV AL，BL              ；取出字型码
        OUT DX，AL              ；送出字形码
        MOV DX，0ff21H          ；把位形码端口地址放入 DX 寄存器
        MOV AL，AH              ；
        OUT DX，AL              ；送出位形码
        CALL   DLY              ；调用延时子程序
        ROR AH，01H             ；修改位形码，指向下一个数码管
        JMP   disp0             ；跳转到显示子程序开始，重复显示
DLY:    MOV CX，0001H           ；延时子程序，设置外层循环延时常数
dly1:   PUSH CX                 ；保存
        MOV CX，0ffffH          ；设置内层循环延时常数
disp1:  LOOP   disp1            ；循环判断（CX—1=0？）
        POP   CX                ；取出外层循环延时常数
        LOOP   dly1             ；判断外层循环次数
        RET                     ；返回
    CODE   NDS                  ；程序段结束
   END START                    ；程序结束
```

## 四、实验步骤

(1)在 4000H 单元写入 00，01，……09 中任一个数。

(2)在 PC 机和实验系统联机状态下，运行该实验程序，可用鼠标左键单击菜单栏“文件”或工具栏“打开图标”，弹出“打开文件”的对话框，然后打开 8kAsm 文件夹，点击 S9.ASM 文件，单击“确定”即可装入源文件，再单击工具栏中编译，即可完成源文件自动编译、装载目标代码功能，再单击“调试”中“连续运行”或工具图标运行，即开始运行程序。

(3)数码管应根据 4000H 单元中内容作不同的循环显示。

(4)修改程序，使数码管显示 A，并使延时程序延时加倍。

## 五、问题思考

(1) 为什么能看到数码管的循环点亮？
(2) 分析 SI、CX 寄存器的作用。

# 实验五 8255A 并行口实验

## 一、实验目的

（1）掌握通过 8255A 并行口传输数据的方法，以控制发光二极管的亮与灭

## 二、实验仪器

微机、微机原理接口实验仪

## 三、实验原理

⑴ 通过 8255A 控制发光二极管，PB4-PB7 对应黄灯，PC0-PC3 对应红灯，PC4-PC7 对应绿灯，以模拟交通路灯的管理。

⑵ 要完成本实验，必须先了解交通路灯的亮灭规律，设有一个十字路口 l、3 为南北方向，2、4 为东西方向，初始状态为四个路口的红灯全亮，之后，1、3 路口的绿灯亮，2、4 路口的红灯亮，1、3 路口方向通车。延时一段时间后，l、3 路口的绿灯熄灭，而 l、3 路口的黄灯开始闪烁，闪烁若干次以后，l、3 路口红灯亮，而同时 2、4 路口的绿灯亮，2、4 路口方向通车，延时一段时间后，2、4 路口的绿灯熄灭，而黄灯开始闪烁，闪烁若干次以后，再切换到 l、3 路口方向，之后，重复上述过程。

⑶ 程序中设定好 8255A 的工作模式及三个端口均工作在方式 0，并处于输出状态。

⑷ 各发光二极管共阳极，使其点亮应使 8255A 相应端口的位清 0。

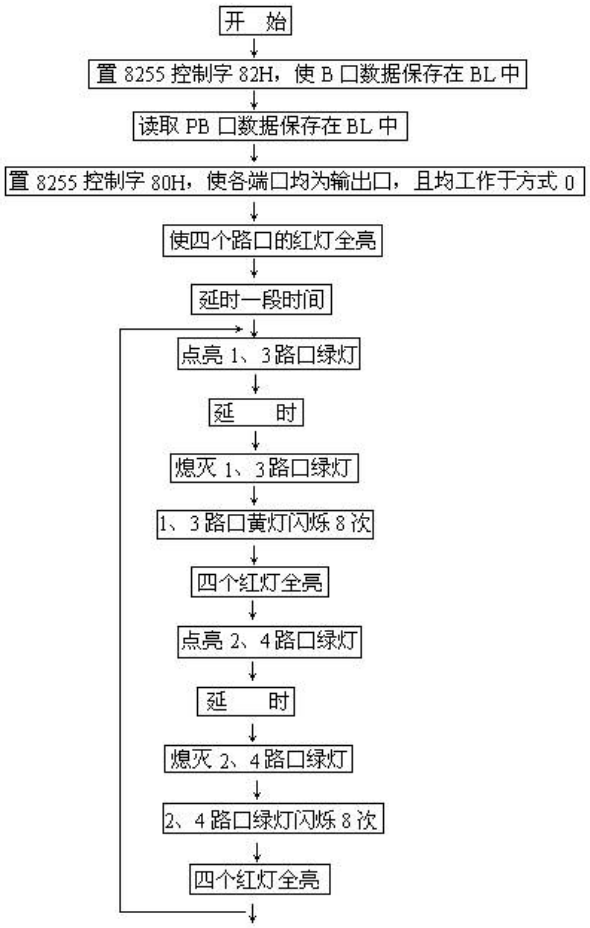
实验程序：

```
CODE        SEGMENT ;H8255-2.ASM
            ASSUME CS:CODE
IOCONPT     EQU 0FF2BH                      ; 8255 控制口地址
IOAPT       EQU 0FF28H                      ; 8255 PA 口地址
IOBPT       EQU 0FF29H                      ; 8255 PB 口地址，  高 4 位接黄灯
IOCPT       EQU 0FF2AH                      ; 8255 PC 口地址，低 4 位接红灯，高 4 位接绿灯
                ORG 11e0H
START:          MOV AL,82H                  ; 82H 控制字送 AL，
                MOV DX,IOCONPT              ; 8255 控制口地址送 DX 寄存器
                OUT DX,AL                   ; 送出控制字，PA、PC 输出，PB 输入
                MOV DX,IOBPT                ; PB 口地址送 DX
                IN AL,DX                    ; 从 PB 口取数据（PB 口低 4 位别处在用）
                MOV BYTE PTR DS:[0601H],AL  ; 保存到 0601H 单元
                MOV DX,IOCONPT              ; 取 8255 控制口地址
                MOV AL,80H                  ; 设置成 PA、PC、PB 输出
                OUT DX,AL
                MOV DX,IOBPT                ; 取 PB 口地址
                MOV AL,DS:[0601H]           ; 取 0601H 单元的数
                OR AL,0F0H                  ; 屏蔽低 4 位
                OUT DX,AL                   ; 送到 PB 口（恢复 PB 口低 4 位状态）
                MOV DX,IOCPT                ; 取 PC 口地址
                MOV AL,0F0H                 ; 把 0FH 送 PC 口，4 个红灯全亮
                OUT DX,AL
                CALL DELAY1                 ; 延时
```

```
IOLED0: MOV AL,10100101B        ; 1、3 路口绿灯亮，2、4 路口红灯亮
        MOV DX,IOCPT            ; 取 PC 口地址
        OUT DX,AL               ; 送出
        CALL DELAY1             ; 延时
        CALL DELAY1
        OR AL,0F0H              ; 熄灭 1、3 路口绿灯
        OUT DX,AL
        MOV CX,8H               ; 设置黄灯闪烁次数
IOLED1: MOV DX,IOBPT            ; 取 PB 口地址
        MOV AL,DS:[0601H]       ; 取 PB 口原来状态
        AND AL,10101111B        ; 点亮 1、3 路口黄灯，保持 PB 口低 4 位
        OUT DX,AL
        CALL DELAY2             ; 延时
        OR AL,01010000B         ; 熄灭 1、3 路口黄灯，保持 PB 口低 4 位
        OUT DX,AL               ;
        CALL DELAY2             ; 延时
        LOOP IOLED1             ; CX-1 不等于 0，到 IOLED1 循环
        MOV DX,IOCPT            ; 取 PC 口地址
        MOV AL,0F0H             ; 4 个红灯全亮
        OUT DX,AL
        CALL DELAY2             ; 延时
        MOV AL,01011010B        ; 2、4 路口绿灯亮，1、3 路口红灯亮
        OUT DX,AL
        CALL DELAY1             ; 延时
        CALL DELAY1
        OR AL,0F0H              ; 4 个红灯亮
        OUT DX,AL
        MOV CX,8H               ; 设置黄灯闪烁次数
IOLED2: MOV DX,IOBPT            ; 取 PB 口地址
        MOV AL,DS:[0601H]       ; 取 PB 口原来状态
        AND AL,01011111B        ; 点亮 2、4 路口黄灯，保持 PB 口低 4 位
        OUT DX,AL
        CALL DELAY2             ; 延时
        OR AL,10100000B         ; 熄灭 2、4 路口黄灯，保持 PB 口低 4 位
        OUT DX,AL
        CALL DELAY2             ; 延时
        LOOP IOLED2             ; CX-1 不等于 0，到 IOLED2 循环
        MOV DX,IOCPT            ; 取 PC 口地址
        MOV AL,0F0H             ; 4 个红灯全亮
        OUT DX,AL
        CALL DELAY2             ; 延时
        JMP IOLED0              ; 主循环
DELAY1: PUSH AX                 ; 延时子程序
        PUSH CX
        MOV CX,0030H            ; 设置延时常数
```

```
DELY2:   CALL DELAY2
         LOOP DELY2              ；CX-1 不等于 0，到 DELY2 循环
         POP CX
         POP AX
         RET
DELAY2:  PUSH CX                 ；短延时程序
         MOV CX,8000H            ；设置延时常数
DELA1:   LOOP DELA1              ；CX-1 不等于 0，到 DELY1 循环
         POP CX
         RET
CODE     ENDS
         END START
```

程序流程图：




## 四、实验步骤

（1）按图 6-4 连好实验线路

8255A　PC0－L3　PC1－L7　PC2－L11　PC3－L15　　（红灯）

PC4－L2　PC5－L6　PC6－L10　PC7－L14　　（绿灯）

PB4－L1　PB5－L5　PB6－L9　PB7－L13　　（黄灯）

（2）在联机状态下，加载 H8255-2.ASM 程序，全速运行程序，观察发光二极管工作状态。
（3）修改程序，使黄灯闪烁 16 次；车辆通过时间加倍。

## 五、问题思考

（1）为什么把 B 端口的值读进内存？
（2）分析短延时程序和长延时程序各对工作状态的影响。

# 实验六 8259 单级中断控制器实验

## 一、实验目的

（1）掌握 8259 中断控制器的接口方法

（2）学习 8259 中断控制器的应用编程

## 二、实验仪器

微机、微机原理接口实验仪

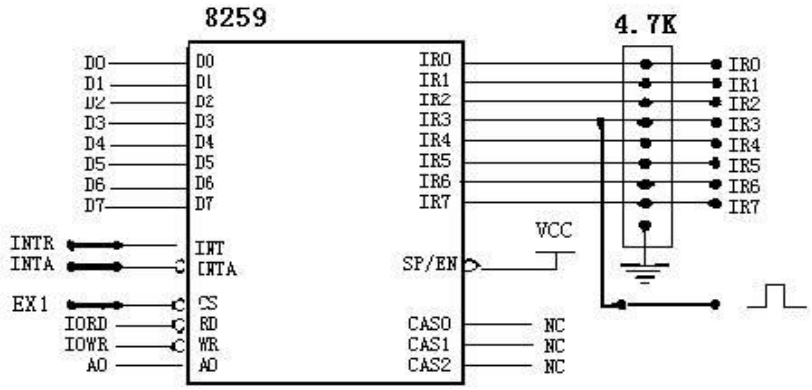
## 三、实验原理

⑴ 8259 芯片介绍

中断控制器 8259A 是专为控制优先级中断而设计的芯片。它将中断源优先级排队、辨别中断源以及提供中断矢量的电路集于一片中。因此无需附加任何电路，只需对 8259A 进行编程，就可以管理 8 级中断，并选择优先模式和中断请求方式。即中断结构可以由用户编程来设定。同时，在不需要增加其它电路的情况下，通过多片 8259A 的级联，能构成多达 64 级的矢量中断系统。

| 中断序号 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 变量地址 | 20H<br>23H | 24H<br>27H | 28H<br>2BH | 2CH<br>2FH | 30H<br>33H | 34H<br>37H | 38H<br>3BH | 3CH<br>3FH |

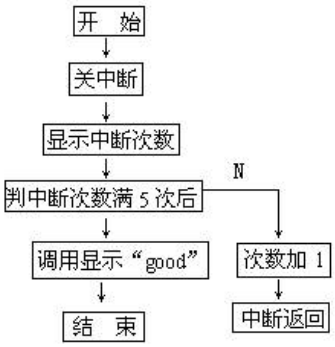
⑵ 本实验中使用 3 号中断源 IR3， “∩”插孔和 IR3 相连，中断方式为边沿触发方式，每拨二次 AN 开关产生一次中断，满 5 次中断，显示“8259——good”。如果中断源电平信号不符合规定要求，则自动转到 7 号中断，显示“Err”。




程序流程图：

主程序：


IR3 中断服务程序：




实验程序：

```
CODE        SEGMENT ;H8259.ASM
            ASSUME CS:CODE
INTPORT1    EQU 0060H          ;8259A 的命令字 ICW1 地址（8259A 的 CS 接到端口 EX1=60H 上）
INTPORT2    EQU 0061H          ; 8259A 的命令字 ICW2、ICW3、ICW4 地址
INTQ3       EQU INTREEUP3
INTQ7       EQU INTREEUP7
PA          EQU 0FF21H ;字位口  ; PA 口作为数码管位形口，地址是 0FF21H
PB          EQU 0FF22H ;字形口  ; PB 口作为数码管字形口，地址是 0FF22H
PC          EQU 0FF23H ;键入口
            ORG 12D0H
START:      JMP START0
BUF         DB ?,?,?,?,?,?      ; 预定显示缓冲区
intcnt      db ?                ; 预定中断次数单元
data1:      db 0c0h,0f9h,0a4h,0b0h,99h,92h,82h,0f8h,80h,90h,88h,83h,0c6h,0a1h
            db 86h,8eh,0ffh,0ch,89h,0deh,0c7h,8ch,0f3h,0bfh,8FH
START0:     CLD
            CALL BUF1          ; 把 8255-1 送入显示缓冲区
            CALL WRINTVER      ; 把中断向量设置到中断向量表

            MOV AL,13H         ; 把 13H 送命令字 ICW1，单片（不用 ICW3）、用 ICW4
            MOV DX,INTPORT1
            OUT DX,AL
            MOV AL,08H         ; 把 08H 送命令字 ICW2，中断向量从 08H 到 0FH 编码
            MOV DX,INTPORT2
            OUT DX,AL
            MOV AL,09H         ; 把 09H 送命令字 ICW4，缓冲方式、8088 系列
            OUT DX,AL
            MOV AL,0F7H        ; 把 0F7H 送命令字 OCW1，IR3 允许中断
            OUT DX,AL
            MOV intcnt,01H ;TIME=1      ; 设置中断次数单元
            STI                         ; 开中断
```

```
WATING:  CALL DISP ;DISP 8259-1        ；调用显示子程序
         JMP WATING                    ；循环等待中断

WRINTVER: MOV AX,0H                    ；设置中断向量子程序。赋予附加段段地址
          MOV ES,AX
          MOV DI,002CH                 ；段内偏移量赋初值
          LEA AX,INTQ3                 ；产生 INTQ3 段内偏移量地址放入 AX
          STOSW                        ；将 AX 的值写入到 ES：DI 指定的地址，同时 DI+2（写入中断
服务程序的偏移地址到中断向量表，设置 INTQ3 中断向量）
          MOV AX,0000h                 ；中断服务程序的段地址是 0
          STOSW                        ；将 AX 的值写入到 ES：DI 指定的地址，同时 DI+2（写入中断
服务程序的段地址到中断向量表，设置 INTQ3 中断向量）
          MOV DI,003CH
          LEA AX,INTQ7                 ；产生 INTQ7 段内偏移量地址地址放入 AX
          STOSW                        ；将 AX 的值写入到 ES：DI 指定的地址，同时 DI+2（写入中断
服务程序的偏移地址到中断向量表，设置 INTQ7 中断向量）
          MOV AX,0000h                 ；中断服务程序的段地址是 0
          STOSW                        ；将 AX 的值写入到 ES：DI 指定的地址，同时 DI+2（写入中断
服务程序的段地址到中断向量表；设置 INTQ3 中断向量）

          RET

INTREEUP3: CLI                         ；中断 3 子程序
           MOV AL,INTCNT               ；取中断次数
           CALL CONVERS                ；设置显示中断次数
           MOV BX,OFFSET BUF           ；取 BUF 显示缓冲区首地址段内偏移量（077BH）
           MOV AL,10H                  ；使数码管不显示的字符存入 AL
           MOV CX,05H                  ；循环次数，数码管的个数
 INTRE0:   MOV [BX],AL                 ；把显示码送入显示缓冲区 BX 指定的单元
           INC BX                      ；指向下一个单元
           LOOP INTRE0                 ；CX-1 不等于 0 则跳转，直到缓冲区写完
           MOV AL,20H                  ；把 20H 送到 OCW2，发出一般的中断结束命令
           MOV DX,INTPORT1
           OUT DX,AL
           ADD INTCNT,01H              ；增量中断次数单元
           CMP INTCNT,06H              ；是否够 5 次
           JNA INTRE2                  ；小于等于时转移到 INTRE2
           CALL BUF2 ;DISP:good        ；大于时调用把 GOOD 送入显示缓冲区子程序
INTRE1:    CALL DISP                   ；显示 GOOD
           JMP INTRE1                  ；死循环
CONVERS: AND AL,0FH                    ；屏蔽高 4 位
           MOV BX,offset buf           ；取显示缓冲区首地址段内偏移量
           MOV [BX+5],AL               ；把中断次数送缓冲区最后一位
```

```
          RET
INTRE2:   MOV AL,20H                ；把 20H 送到 OCW2，发出一般的中断结束命令
          MOV DX,INTPORT1
          OUT DX,AL
          STI                       ；开中断
          IRET                      ；中断返回

INTREEUP7: CLI                      ；中断 7 子程序
          MOV AL,20H                ；把 20H 送到 OCW2，发出一般的中断结束命令
          MOV DX,INTPORT1
          OUT DX,AL
          call buf3 ;disp:err       ；调用 Err 送入显示缓冲区子程序
INTRE3:   CALL DISP                 ；显示 Err
          JMP INTRE3                ；死循环

DISP:     MOV AL,0FFH               ；显示子程序，位码放 AL，0FFH 代表不显示
          MOV DX,PA                 ；位形口地址放 DX
          OUT DX,AL                 ；熄灭所以数码管
          MOV CL,0DFH               ；位形码放 CL
          MOV BX,OFFSET BUF         ；取显示缓冲区首地址段内偏移量
DIS1:     MOV AL,[BX]               ；从显示缓冲区取显示码
          MOV AH,00H
          PUSH BX
          MOV BX,OFFSET DATA1       ；取显示字型码首地址段内偏移量
          ADD BX,AX                 ；形成显示字形码地址
          MOV AL,[BX]               ；查表取字型码
          POP BX
          MOV DX,PB                 ；字形口地址放 DX
          OUT DX,AL                 ；送出字形码
          MOV AL,CL                 ；取位形码
          MOV DX,PA                 ；位形口地址放 DX
          OUT DX,AL                 ；送出位形码
          PUSH CX                   ；短延时准备
DIS2:     MOV CX,00A0H              ；延时常数放 CX
          LOOP $                    ；CX-1 不等于 0 则循环
          POP CX
          CMP CL,0FEH               ；是否显示完最后一位
          JZ LX1                    ；显示完跳转
          INC BX                    ；没完修改缓冲区偏移量，指向下一位
          ROR CL,1                  ；修改位形码
          JMP DIS1                  ；去显示下一位
LX1:      MOV AL,0FFH               ；
          MOV DX,PB                 ；熄灭所以数码管
```

```
        OUT DX,AL
        RET
BUF1:   MOV BUF,08H          ；把 8255-1 送入显示缓冲区子程序
        MOV BUF+1,02H
        MOV BUF+2,05H
        MOV BUF+3,09H
        MOV BUF+4,17H
        MOV BUF+5,01H
        RET
BUF2:   MOV BUF,09H          ；把 GOOD 送入显示缓冲区子程序
        MOV BUF+1,00H
        MOV BUF+2,00H
        MOV BUF+3,0dH
        MOV BUF+4,10H
        MOV BUF+5,10H
        RET
BUF3:   MOV BUF,0eH          ；把 Err 送入显示缓冲区子程序
        MOV BUF+1,18H
        MOV BUF+2,18H
        MOV BUF+3,10H
        MOV BUF+4,10H
        MOV BUF+5,10H
        RET
CODE    ENDS
        END START
```

## 四、实验步骤

（1）按图 6－6 连好实验线路图。

8259 的 INT 连 8088 的 INTR（Xl5）； 8259 的 INTA 连 8088 的 INTA（Xl2）； “∩”插孔和 8259 的 3 号中断 IR3 插孔相连，“∩”端初始为低电平； 8259 的 CS 端接 EX1； 连 JX4→JX17。

（2）在联机状态下，加载 H8259.ASM 程序，全速运行程序，系统显示 8259－1，。

（3）拨动 AN 开关按钮， 按满 5 次显示 good。

（4）修改程序，使开关搬动 10 次显示 2008。

## 五、问题思考

（1）显示缓冲区的作用是什么

（2）进入中断服务子程序由什么决定？