微机原理及接口技术

实 验 指 导 书

杨霞 王江波 张文革 李斌 编

长安大学电子与控制工程学院

自动化与交通控制工程实验中心

2009 年 9 月

前　言

本实验指导是为适应各大、中专院校开设微机原理及应用方面的课程需做大量软硬件实验的需要而编写的，供学生编程用。完成本实验指导中的实验，可使学生基本掌握 8086/8088 的结构原理、接口技术、程序设计技巧。手册中详细叙述了各实验的目的、内容，列出了接线图、程序框图和实验步骤。

主要学习内容为 80X86 语言实验环境配置、汇编源语言格式、输出字符、循环结构、子程序调用，以及加减乘除等指令操作、通用接口芯片的接口编程与使用。所有实验都是相互独立的，次序上也没有固定的先后关系，在使用本书进行教学时，教师可根据教学要求，选择相应实验。学习结束后，要求学生能够独立编写出综合加减乘除等指令，以及循环结构、子程序调用等程序控制程序、看懂一般接口芯片电路图。

目 录

实验一 清零程序

一、实验目的

掌握 8088 汇编语言程序设计和调试方法。

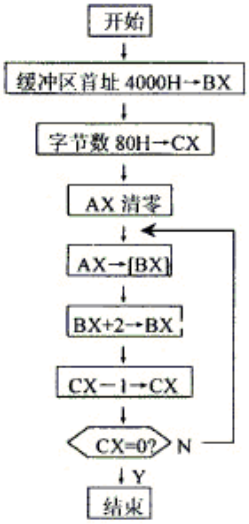
二、实验设备

STAR 系列实验仪一套、PC 机一台。

三、实验内容

把 RAM 区内 4000H-40FFH 单元的内容清零 。

四、程序框图



五、源程序清单

```
    .MODEL   TINY
    .STACK    100
    .DATA
    .CODE
    ORG 0100H
START:

    ;编写程序

    JMP $
    END START
```

六、实验步骤

手动修改 RAM 区内 4000H-40FFH 的内容，连续或单步方式运行程序，检查 4000-40FFH 内容 执行程序前后的变化。

七、思考

1、把 4000H-40FFH 中的内容改成 FF，如何修改程序。

2、把 4000H-40FFH 中的内容改成 00~FF，如何修改程序。

实验二 拆字程序

一、实验目的

掌握汇编语言设计和调试方法。

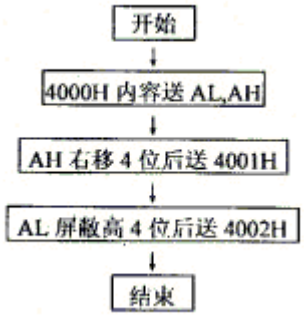
二、实验设备

STAR 系列实验仪一套、PC 机一台。

三、实验内容

把 4000H 单元的内容拆开，高位送 4001H 低位，低位送 4002H 低位，4001H、4002H 的高位清零，一般本程序用于把数据送显示缓冲区时用。

四、程序框图



五、源程序

```
    .MODEL TINY
    .STACK
    .DATA
    .CODE
START:

    ; 编写程序

    JMP $
```

六、实验步骤

手动修改 4000H 的内容，用连续或单步方式运行程序，检查 4000H-4002H 中内容变化情况。

七、思考

1. 如何用断点方式调试本程序。

2. 把 4000H、4001H 单元低位的内容合成一字送 4002H 单元。

实验三 数据区移动

一、实验目的

掌握 RAM 中的数据操作。

二、实验设备

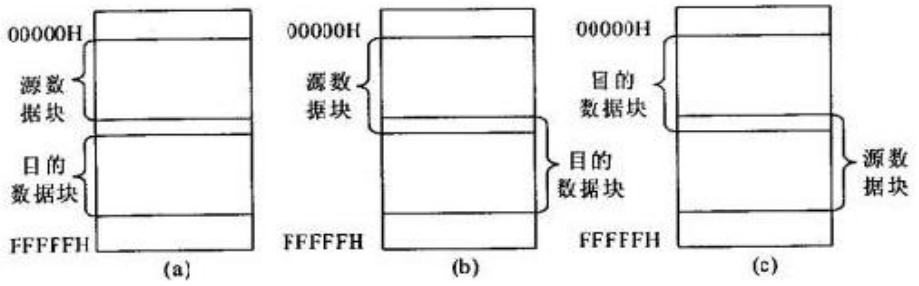
STAR 系列实验仪一套、PC 机一台。

三、实验内容

把 4100H 源 RAM 区首址内的 16 字节数据传送到 4200H 目标 RAM 区。

四、设计思想

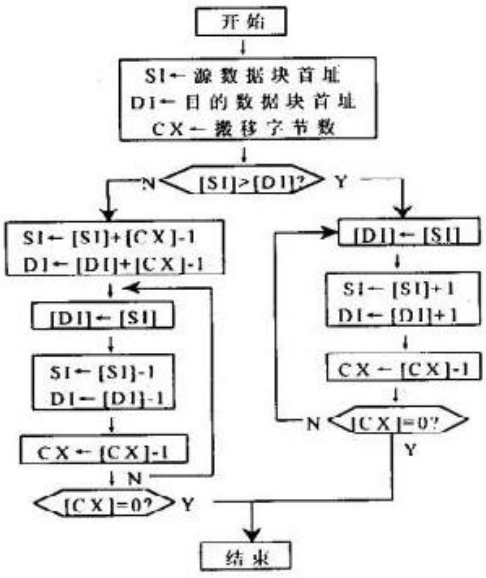
程序要求把内存中某一数据区(称为源数据块)传送到另一存储区(称为目的数据块)。源数据块和目的数据块在存储器中可能有三种情况，如下图：



对于两个数据块分离的情况，如图(a)，数据的传送从数据块的首址开始，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。

可以得出以下结论：当源数据首址＞目的块首址时，从数据块首址开始传送数据。 当源数据块首址＜目的块首址时，从数据块末地址开始传送数据。

五、程序框图



六、源程序清单

```
    .MODEL    TINY
.STACK
.DATA
.CODE
ORG 0100H
START0:

; 编写程序

JMP $
END START0
```

七、实验步骤

调试运行 3060 程序,检查 4100-410FH 中内容是否和 4200-420FH 中内容完全一致。

八、思考

1. 把 4200H 源 RAM 区首址内的 16 字节数据传送到 4100H 目标 RAM 区。
2. 把 4100H 源 RAM 区首址内的 16 字节数据传送到 410AH 目标 RAM 区。

实验五 多字节减法运算

一、实验目的

掌握 BCD 码、补码，熟悉多文件、多模块汇编语言程序设计的方法。

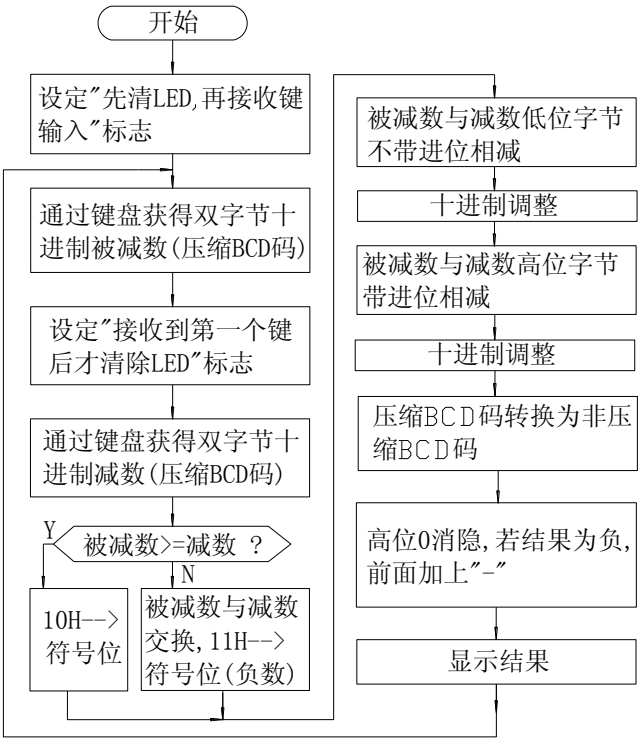
二、实验设备

STAR 系列实验仪一套、PC 机一台。

三、实验内容

从键盘上输入 4 位被减数、减数，实现双字节 BCD 码(四位数)的减法，结果显示在数码管上；进一步熟悉使用断点、单步进入、单步、运行到光标处、修改 PC 指针、全速运行等各种调试手段；熟悉查看特殊功能寄存器、CS 段、DS 段存贮器的各种方法。

四、程序框图



双字节十进制减法程序框图

五、实验步骤

1、连线说明：

E5 区 ： CLK	——	B2 区：2M
E5 区 ： CS	——	A3 区：CS5
E5 区 ： A0	——	A3 区：A0
E5 区 ： A、B、C、D	——	G5 区：A、B、C、D

2、在 G5 区的键盘上输入 4 位被减数、减数

3、结果显示在 G5 区的数码管上

六、源程序清单

```
                .MODEL          TINY
EXTRN           Display8:NEAR, GetBCDKey:NEAR     Display8显示BUFFER内连续的八个字符或
                                                  数字。;GetBCDKey得到双字节十进制数,
                                                  键值存放在DI所指的内存区（字型）
```

```
LED_TAB:    DB  03FH,06H,05BH,04FH,66H,6DH,7DH,07H
            DB  07FH,6FH,77H,7CH,39H,05EH,79H,71H
        .CODE
START:  MOV     AX,@DATA
        MOV     DS,AX
        MOV     ES,AX

 ;编写程序

        END     START
```

(2) 编写程序，控制 8 位数码管依次显示任意指定字符（0-F）；

```
        .MODEL TINY
        .STACK  100
        .MODEL  TINY
        .STACK  100
        .DATA
IO8255_Con  EQU     0F003H      ;CS3
IO8255_A    EQU     0F000H      ;A
IO8255_B    EQU     0F001H      ;B

LED_TAB:    DB  03FH,06H,05BH,04FH,66H,6DH,7DH,07H
            DB  07FH,6FH,77H,7CH,39H,05EH,79H,71H
        .CODE
START:  MOV     AX,@DATA
        MOV     DS,AX
        MOV     ES,AX

 ;编写程序

    ENDSTART
```

(3) 编写程序，控制 8 位数码管同时显示 0、1、……F:

```
.MODEL  TINY
.STACK  100
.MODEL  TINY
.STACK  100
.DATA
IO8255_Con  EQU0F003H       ;CS3
IO8255_A    EQU0F000H       ;A
IO8255_B    EQU0F001H       ;B

LED_TAB:    DB  03FH,06H,05BH,04FH,66H,6DH,7DH,07H
        DB  07FH,6FH,77H,7CH,39H,05EH,79H,71H
F1      DB  0
    .CODE
```

```
                .MODEL      TINY
;使用8253的计数器0,外接2Mhz,经26分频后,送给8251,产生4800bps
CTL_ADDR        EQU         0FF01H          ;控制字或状态字
DATA_ADDR       EQU         0FF00H          ;读写数据
W_8253_T0       EQU         0BF00H          ;计数器0地址
W_8253_C        EQU         0BF03H          ;控制字
                .STACK      100
                .DATA
Receive_Buffer  DB          10 DUP(0)       ;接受缓冲器
Send_Buffer     EQU         Receive_Buffer  ;发送缓冲器
                .CODE
START:          MOV         AX,@DATA
                MOV         DS,AX
                MOV         ES,AX
                NOP
                CALL        INIT_8253
                CALL        INIT_8251
START1:         MOV         CX,10
                CALL        Receive_Group
                MOV         CX,10
                CALL        Send_Group
                JMP         START1
INIT_8253       PROC        NEAR
                MOV         DX,W_8253_C
                MOV         AL,37H          ;定时器0，方式3
                OUT         DX,AL
                MOV         DX,W_8253_T0
                MOV         AL,26H          ;BCD码26(2000000/26)=16*4800
                OUT         DX,AL
                MOV         AL,0
                OUT         DX,AL
                RET
INIT_8253       ENDP
INIT_8251       PROC        NEAR
                CALL        RESET_8251
                MOV         DX,CTL_ADDR
                MOV         AL,7EH          ;波特率系数为16，8个数据位
                OUT         DX,AL           ;一个停止位，偶校验
                CALL        DLTIME          ;延时
                MOV         AL,15H          ;允许接收和发送发送数据，清错误标志
                OUT         DX,AL
                CALL        DLTIME
                RET
INIT_8251       ENDP
```

```
Reset_8251      PROC        NEAR
                MOV         DX,CTL_ADDR
                MOV         AL,0
                OUT         DX,AL           ;向控制口写入"0"
                CALL        DLTIME          ;延时，等待写操作完成
                OUT         DX,AL           ;向控制口写入"0"
                CALL        DLTIME          ;延时
                OUT         DX,AL           ;向控制口写入"0"
                CALL        DLTIME          ;延时
                MOV         AL,40H          ;向控制口写入复位字40H
                OUT         DX,AL
                CALL        DLTIME
                RET
Reset_8251      ENDP
;接受一组数据,CX--接受数目
Receive_Group   PROC        NEAR
                LEA         DI,Receive_Buffer
Receive_Group1: CALL        Receive_Byte
                STOSB
                LOOP        Receive_Group1
                RET
Receive_Group   ENDP
;接受一个字节
Receive_Byte    PROC        NEAR
                MOV         DX,CTL_ADDR
Receive_Byte1:  IN          AL,DX           ;读入状态
                TEST        AL,2
                JZ          Receive_Byte1   ;有数据吗?
                MOV         DX,DATA_ADDR    ;有
                IN          AL,DX
                RET
Receive_Byte    ENDP
;发送一组数据,CX--发送数目
Send_Group      PROC        NEAR
                LEA         SI,Send_Buffer
Send_Group1:    LODSB
                CALL        SendByte
                LOOP        Send_Group1
                RET
Send_Group      ENDP
;发送一个字节
Sendbyte        PROC        NEAR
                PUSH        AX
                MOV         DX,CTL_ADDR     ;读入状态
```

2、调试程序，查看运行结果是否正确

七、演示程序

```
                .MODEL   TINY
EXTRN           Display8:NEAR, SCAN_KEY:NEAR
IO8259_0        EQU      0F000H
IO8259_1        EQU      0F001H
Con_8253        EQU      0E003H
T0_8253         EQU      0E000H
IO8255_Con      EQU      0D003H              ;CS3
IO8255_PC       EQU      0D002H
                .STACK   100
                .DATA
StepControl     DB       0                   ;下一次送给步进电机的值
buffer          DB       8 DUP(0)            ;显示缓冲区，8个字节
buffer1         DB       8 DUP(0)            ;显示缓冲区，8个字节
SpeedNo         DB       0                   ;选择哪一级速度
StepDelay       DB       0                   ;转动一步后，延时常数
StartStepDelay  DB       0;若选择速度过快,延时由长到短,最终使用对应延时常数
StartStepDelay1 DB       0                   ;StartStepDelay
bFirst          DB       0                   ;有没有转动过步进电机
bClockwise      DB       0                ; =1 顺时针方向  =0 逆时针方向转动
bNeedDisplay    DB       0                   ;已转动一步，需要显示新步数
StepCount       DW       0                   ;需要转动的步数
StepDelayTab:   DB       250,125,83,62,50,42,36,32,28,25,22,21
.CODE
START:          MOV      AX,@DATA
                MOV      DS,AX
                MOV      ES,AX
                NOP
                MOV      bFirst,1            ;有没有转动过步进电机
                MOV      bClockwise,1        ;顺时针方向
                MOV      StepControl,33H     ;下一次送给步进电机的值
                MOV      SpeedNo,5           ;第五级速度
                CALL     Init8255
                CALL     Init8253
                CALL     Init8259
                CALL     WriIntver
                MOV      buffer,0            ;显示缓冲器初始化
                MOV      buffer+1,0
                MOV      buffer+2,0
                MOV      buffer+3,0
                MOV      buffer+4,10H
                MOV      AL,SpeedNo
                MOV      buffer+5,AL
```

```
                MOV     buffer+6,10H
                MOV     buffer+7,0
STAR2:          LEA     SI,buffer
                LEA     DI,buffer1
                MOV     CX,8
                REP     MOVSB
                LEA     SI,buffer1
                CALL    Display8
STAR3:          CALL    Scan_Key
                JB      STAR5
                CMP     bNeedDisplay,0
                JZ      STAR3
                MOV     bNeedDisplay,0
                CALL    Step_SUB_1
                JMP     STAR2
STAR5:          CLI                             ;终止步进电机转动
                CMP     AL,10
                JNB     STAR1
                MOV     AH,buffer+2
                MOV     buffer+3,AH
                MOV     AH,buffer+1
                MOV     buffer+2,AH
                MOV     AH,buffer
                MOV     buffer+1,AH
                MOV     buffer,AL
                JMP     STAR2
STAR1:          CMP     AL,14
                JNB     STAR3
                LEA     SI,DriverTab
                SUB     AL,10
                SHL     AL,1
                XOR     AH,AH
                MOV     BX,AX
                JMP     CS:[SI+BX]
DriverTab:      DW      Direction               ;转动方向
                DW      Speed_up                ;提高转速
                DW      Speed_Down              ;降低转速
                DW      Exec                    ;步进电机根据方向、转速、步数开始转动
Direction:      CMP     bClockwise,0
                JZ      Clockwise
                MOV     bClockwise,0
                MOV     buffer+7,1
AntiClockwise:  CMP     bFirst,0
                JZ      AntiClockwise1
```

```
                OUT     DX,AL
                MOV     AL,0FEH
                OUT     DX,AL
                RET
Init8259        ENDP
WriIntver       PROC    NEAR
                PUSH    ES
                MOV     AX,0
                MOV     ES,AX
                MOV     DI,20H
                LEA     AX,TIMER0
                STOSW
                MOV     AX,CS
                STOSW
                POP     ES
                RET
WriIntver       ENDP

                END     START
```

八、实验扩展及思考

1、怎样改变电机的转速？

2、通过实验找出电机转速的上限，如何能进一步提高最大转速？

3、怎样能使电机反转？

附录一　汇编语言的存储模型

模型	说明
TINY(微)	所有数据及代码装入同一个代码段内，此模型的程序按.COM 文件格式编写，要求程序从地址 0100H 处开始
SMALL(小)	这种模型包含两个段：一个 64KB 的数据段和一个 64KB 的代码段
MEDIUM(中)	这种模型包含一个 64KB 的数据段和任意多个代码段，以供大程序使用
COMPACT(压缩)	包含一个代码段和任意多个数据段
LARGE(大)	LARGE 模型允许多个代码段和数据段
HUGE（巨型）	允许数据段大于 64KB，其他与 LARGE 模型相同
FLAT（平展）	仅限于 MASM6.X 版本。FLAT 模型使用一个 512KB 的段来存储所有的代码和数据，应注意的是该模型主要用于 Windows NT 中

```
        RET
KeyScan     ENDP
;CY =1,有键,键值在 AL 中;CY=0,没有按键
GetKeyA     PROC    NEAR
        CALL    SCAN_KEY
        RET
GetKeyA     ENDP
;键值在 AL 中
GetKeyB     PROC    NEAR
        CALL    SCAN_KEY
        JNB GetKeyB
        RET
GetKeyB     ENDP
;BCD 码  ;F1 是否需要先清除显示
GetBCDKey   PROC    NEAR
        CMP CX,0
        JZ  GetBCDKey5
        CMP CX,9
        JNB GetBCDKey5
        PUSH    AX
        PUSH    BX
        PUSH    DX
        PUSHF
        MOV AX,CX
        CLC
        RCR AX,1
        DEC AX
        ADD DI,AX
        STD
        MOV AH,0
        CMP F1,0
        JZ  GetBCDKey1
        CALL    KeyScan
GetBCDKey1: PUSH    AX
        MOV AL,8
```

```
        CLC
        SUB AL,CL
        CALL    INIT8279_1  ;8279 初始化
        POP AX
        CMP F1,0
        JNZ GetBCDKey3
GetBCDKey2: CALL    KeyScan     ;扫描
GetBCDKey3: CMP AL,10
        JNB GetBCDKey2
        NOT AH
        PUSH    AX
        CMP AH,0
        JZ  GetBCDKey4
        ROR AL,4
        MOV ES:[DI],AL
        JMP GetBCDKey6
GetBCDKey4: OR  AL,ES:[DI]
        STOSB
GetBCDKey6: POP AX
        LEA BX,LED_TAB
        XLAT
        CALL    WRITE_DATA  ;显示输入值
        LOOP    GetBCDKey2
        POPF
        POP DX
        POP BX
        POP AX
GetBCDKey5: RET
GetBCDKey   ENDP
;显示以 SI 开始的 8 个数字（0～F)
DisPlay8    PROC    NEAR
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX
```

附录三　8279 命令功能一览表

<table>
<tr><th>D7　D6　D5</th><th>D4</th><th>D3</th><th>D2</th><th>D1</th><th>D0</th></tr>
<tr><td>命令类型</td><td colspan="5">命　　令　　内　　容</td></tr>
<tr><td rowspan="6">0　0　0
键盘/显示</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td rowspan="3">左端入口</td><td rowspan="3">8 字符显示</td><td colspan="2">双键锁定</td><td rowspan="3">内编码扫描</td></tr>
<tr><td>0</td><td>1</td></tr>
<tr><td colspan="2">N 键轮回</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td rowspan="3">右端入口</td><td rowspan="3">16 字符显示</td><td colspan="2">传感器矩阵</td><td rowspan="3">内译码扫描</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td colspan="2">选通输入扫描显示</td></tr>
<tr><td rowspan="2">0　0　1
时钟编程</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td colspan="5">分频系数取值为：2-31</td></tr>
<tr><td rowspan="2">0　1　0
读 FIFO/传感器 RAM</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>1-地址自加 1
0-地址不变</td><td>（不用）</td><td colspan="3">定 8279 中 FIFO 及传感器 RAM 的首地址
（000-111B 共 8 个单元）</td></tr>
<tr><td rowspan="2">0　1　1
读显示器 RAM 内容</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>1-地址自加 1
0-地址不变</td><td colspan="4">读显示器 RAM 内容
（0000-1111B 共 16 个单元）</td></tr>
<tr><td rowspan="2">1　0　0
写显示 RAM</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>1-地址自加 1
0-地址不变</td><td colspan="4">写显示器 RAM
（0000-1111B 共 16 个单元）</td></tr>
<tr><td rowspan="2">1　0　1
显示禁止/熄灭</td><td>X</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>（不用）</td><td>禁止写 A 组显示 RAM</td><td>禁止写 B 组显示 RAM</td><td>A 组熄灭</td><td>B 组熄灭</td></tr>
<tr><td rowspan="8">1　1　0
清除显示 RAM 和 FIFO</td><td>1</td><td>0</td><td>X</td><td>1</td><td>1</td></tr>
<tr><td rowspan="5">允许清除</td><td colspan="2">显示 RAM 全部清为 0</td><td rowspan="5">FIFO 成空状态，中断输出线复位，传感器 RAM 读出地址置 0</td><td rowspan="5">全部清除</td></tr>
<tr><td>1</td><td>0</td></tr>
<tr><td colspan="2">显示 RAM 置为 20H（A 组=0010B，B 组=0000B）</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td colspan="2">显示 RAM 置为 FFH</td></tr>
<tr><td>0</td><td>X</td><td>X</td><td>X</td><td>1</td></tr>
<tr><td colspan="5">按 D2、D3 决定的方式清除</td></tr>
<tr><td>0</td><td>X</td><td>X</td><td>X</td><td>0</td></tr>
<tr><td colspan="5">不清除</td></tr>
<tr><td rowspan="2">1　1　1
结束中断/出错方式设置</td><td>1</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td colspan="5">A.　在传感器方式，用此命令结束传感器 RAM 的中断请求。
B.　在键盘扫描 N 键轮回方式，用此命令设置特定错误方式。</td></tr>
</table>