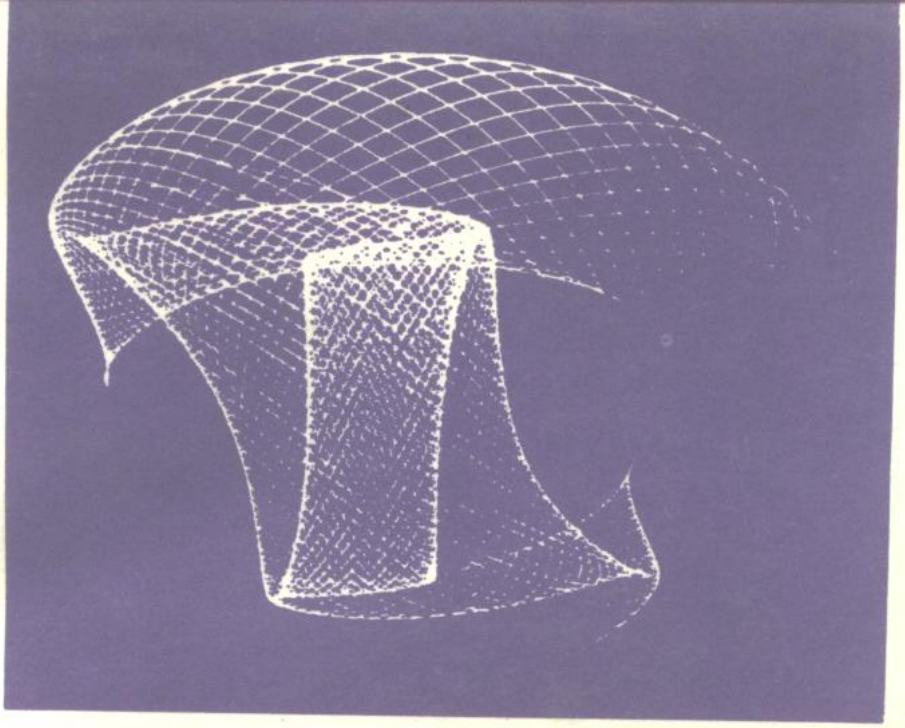

# 微型计算机实用技巧

主　编：牛超群　周景璞
副主编：王国强　王海山　苏振东

海洋出版社

# 微型计算机实用技巧

主　编：牛超群　周景璞
副主编：王国强　王海山　苏振东

[illegible]出版社
[illegible]年·北京

(京)新登字 087 号

## 内 容 提 要

本书针对微型机应用中的有关问题，精选了应用技巧 200 例，共分 8 大类。本书的主要特点是实用性，兼顾系统性。在每个应用技巧中都尽可能附源程序清单或图示，使读者阅后即可操作使用，解决实际问题；同时每类应用技巧都能自成体系，覆盖了主要应用领域的典型问题。

本书特别适用于微型计算机应用人员之用，亦可供大中专学生、教师阅读参考。

主　编：牛超群　周景璞
副主编：王国强　王海山　苏振东
编　委：（按姓氏笔划为序）
于广平　王铁军　李贞子
宋考平　邢川生　张文波
张建立　陆秀军　周东风
贺志刚　郝玉宝　高淑荣
鹿志文

**微型计算机实用技巧**
主　编：牛超群　周景璞
副主编：王国强　王海山　苏振东
责任编辑：刘 博

海洋出版社出版（北京市复兴门外大街 1 号）
新华书店北京发行所发行　通县宏飞 印刷厂印刷
开本：787×1092　1/16 印张：25.375　字数：700 千字
1993 年 1 月第一版　1993 年 1 月第一次印刷
印数：1—4500 册

ISBN 7—5027—2501—6/TP・72　定价：18.00 元

# 前　言

目前，微型计算机已经应用到国民经济和社会生活的各个领域，并发挥着越来越重要的作用。在微型计算机应用过程中，已经开发了一大批应用成果，积累了大量的应用经验，创造了许多好的应用方法、技巧。这些应用技巧对进一步开展微型机应用无疑是宝贵的财富。有句俗话"窗户纸，一捅就破"，在这里，应用技巧就是起到"捅"的作用。但由于应用人员所掌握的应用技巧多是零散的，不系统的，使用起来还不能得心应手。为使微型计算机发挥更大作用，我们在大量应用技巧中，精选而成这本《微型计算机实用技巧》。

本书共精选应用技巧 200 个，分为 8 类。本书在内容分类和编排上，力求包含针对微型机应用各方面典型问题的技巧，使每类技巧都能自成体系；同时尽量增强可操作性，使读者阅后就可解决实际问题。愿本书成为广大计算机应用人员的得力工具，使计算机应用工作在已有基础上得到更大发展。

在本书的编纂过程中，参考收录了一些国内有关报刊公开发表的文章，在此谨致以衷心的感谢。同时对支持本书编写的张克勤、王荣华、刘景霞、邵刚等许多同志表示感谢！

由于微型计算机应用范围很广，应用技巧很多，本书采集的仅是其中的几朵浪花，加之编者水平有限，书中定有不少谬误与不妥之处，敬请读者批评指正。

编　者

1992 年 4 月于大庆

# 目　录

**第一部分： DOS 实用技巧**





此为试读，需要完整PDF请访问：www.ertongbook.com

**第四部分： 数据库管理系统实用技巧**

**第五部分： 用户界面实用设计技巧**

**第六部分： 打印实用技巧**



**第七部分： 加密与解密实用技巧**

**第八部分： 其他**

# 第一部分： DOS 实用技巧

## 1.1 ATTRIB 的妙用

ATTRIB 命令可以说是 DOS 命令集中一座尚待开发的“金矿”。举例来说，ATTRIB 可以将不得不一次处理一个的文件组合成批处理文件。假设一个程序需要处理若干文件但又不能使用 *.* 方式(如打印俄文字符的程序 PRRUS. COM。标准的方法是建立一个批处理文件为程序如 PRRUS 调用目录中的每个文件)。ATTRIB 可以自动生成这种批文件。

首先，将要运行的程序拷贝一份并命名为 A. COM 或 A. EXE。接着，把要处理的所有文件拷入一个临时子目录(不是严格要求这样，仅是一种安全措施)。然后键入命令 attrib+a-r *.*(或 *.TXT，或其他)。这一步为每个文件打开文档位。再键入 attrib *.* > otp. bat，建立一个名为 OTP. BAT 的批处理文件。当然也可以用其他名字。该批文件将包括类似如下的命令行：

```
A   C:\path\filename   1
A   C:\path\filename   2
```

这里‘A’代表第一个 ATTRIB 命令打开的文档标志。由于已经把执行程序命名为‘A’，不用打入单个命令，只需运行这个建立的批处理文件就可处理所有的文件了。

## 1.2 反动态跟踪的几种方法

为了防止别人窃取、仿制自己的程序，就必须阻止他人对程序进行静态分析和动态跟踪。其主要途径是：

(1) 通过对程序指令和解密程序本身的加密，使程序指令(也称之为明文)变成密文数据，这样反汇编出来的是一些不可理解的符号。而达到防止静态分析的目的。在程序运行时，由一段解密程序逐一恢复原来的指令。(2)主要是根据 DEBUG 程序需要调用 INT 1，INT 3，INT 0，INT 4 等中断，在程序中破坏这些中断向量，使其不能跟踪程序指令。

1. 预备知识

(1) DOS 系统有关的中断调用

| 中断号 | 向量地址 | 中断名称 |
|---|---|---|
| INT 0 | 0:00～0:03 | 被零除 |
| INT 1 | 0:04～0:07 | 单步执行 |
| INT 2 | 0:08～0:0B | 不可屏蔽中断 |
| INT 3 | 0:0C～0:0F | 断点 |
| INT 4 | 0:10～0:13 | 溢出 |

DEBUG 调试程序的单步执行命令是通过调用 INT 1 实现。设置断点命令是通过调用 INT 3 实现。当除数为零进行除法时，系统自动调用 INT 0。如果运算发生溢出，且后有 INT 0 指令，系统调用 INT 4 执行，如果没有溢出，INT 0 相当于空操作。

(2) 中断屏蔽寄存器(口地址： 21H)位：

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| lpt1 | 软盘 | 硬盘 | com2 | com1 | I/O | 键盘 | 时钟 |



此为试读，需要完整PDF请访问：www.ertongbook.com

其中，com1，com2：异步通信口；lpt1：打印机

如果第 i 位为 0，则允许该设备中断；为 1，则不允许该设备中断，在该设备被禁止中断后，就失去了作用。

(3) 标志寄存器

| 位号： | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 功能： | — | NT | IO | PL | OF | DF | IF | TF | SF | ZF | — | AF | — | PF | — | CF |

CF(carry flag)：进位标志位；

PF(parity flag)：奇偶校验标志位；

AF(auxilary flag)：辅助进位标志位；

ZF(zero flag)：零标志位；

SF(sign flag)：符号标志位；

TF (trap flag)：陷井标志位，该位是为 DEBUG 单步执行命令而设置的标志位，当该位置 1，紧接着执行一条指令后，便产生一个内部中断，以便你能逐条指令地查看你的程序，所有指令代码都不影响该位；

IF(interrupt—enalbe flag)：中断使能标志位；

DF(direction flag)：方向标志位；

OF(over—flow flag)：溢出标志位。

2. 反动态跟踪的方法

(1) 由于 DEBUG 程序的单步执行和设置断点命令是通过调用 INT1、INT3 实现的，如果在程序中破坏了这两个中断向量，就不能进一步跟踪程序；

(2) 通过对中断屏蔽寄存器的第二位置 1，禁止键盘产生硬件中断，使键盘不起作用；

(3) 利用解密程序本身作为解密数据的因子，来对下一段的密文进行解密。如果用修改解密程序的方法来避开反动态跟踪指令，就会破坏解密数据因子，从而破坏了解密数据；

程序 1

```
1188 :2B9   NOP
     :2BA   CS:
     :2BB   MOV   [0000],AX
     :2BE   CS:
     :2BF   MOV   [0002],DS
     :2C3   CS:
     :2C4   MOV   [0004],ES
     :2C8   PUSH  DS
     :2C9   XOR   AX,AX
     :2CB   PUSH  AX
     :2CC   POP   DS  ;DS=0
     :2CD   SUB   SP,+02
     :2D0   POP   SI  ;SI=0,这是由程序初始状态决定的
     :2D8   MOV   AX,CS
     :2DA   MOV   ES,AX;ES=CS
     :2DC   MOV   AX,2E8F
     :2DF   PUSH  AX;2E8F 进栈,后面将会用到这个数据
```

```
:2E0   MOV    DI,0006 ;DS=0   ES=CS,DS:0 即 0:0 的 10 个字
:2E3   MOV    CX,000A ;传送到 CS:0 处
:2E6   REPZ;将原始的 INT0,INT1,INT2,INT3,
:2E7   MOVSW, …,等中断向量保存,
:2E8   PUSH   CS
:2E9   POP    DS; DS=CS
:2EA    OR    AL,02 ;读中断屏蔽寄存器
:2EC    MOV   [1A],AL ;送到 1A 内存单元
:2EF    OR    AL,02;置键盘中断位为 1
:2F1    OUT   21,AL;禁止键盘中断
:2F3    MOV   ES,CX ;ES=0,因为在前面的程序已将 CX;清零
:2F5    MOV   BX,0C ;INT3 的中断向量地址偏移
:2F8    MOV   SI,2B9 ;指向解密数据因子,包括了本程序段
:2FB    MOV   DI,326;指向密文件码地址
:2FE    MOV   CX,2879
:301    NOP
:302    LODSB ;取解密数据因子
:303    MOV   AH,AL
:305    ADD   AX,BX
:307    MOV   DX,ES
:309    ADD   AX,DX
:30B    ADD   AL,AH;计算出解密数据
:30D    ES:
:30E    MOV   [BX],AX ;破坏 INT3 的中断向量(地址在 0.0C)
:310    XOR   [DI],AL;解密
:312    CS:
:313    MOV   AL,[1A]
:316    OR    AL,02
:318    OUT   21,AL;禁止键盘产生中断
:31A    INC   DI
:31B    CMP   SI,326;解密数据因子从 2B9-326 之间
:31F    JB    324
:321    MOV   SI,2B9
:324    LOOP   302 ;循环 2879 次
:326 ……
……
```

(4) 利用地址翻转技术来迷惑破译者,在你不注意时,置好中断向量的内容。所谓地址翻转就是段寄存器只有 16 位,如果运算时超过了 16 位,就会产生溢出位,地址发生了翻转;

例如:如果 DS=FBCC,SI=4340,那么 DS:[SI]的段地址是:

FBCC+434=10000=0000,偏移是:00 即 DS:[SI]=FBCC:[4343]=10000:[00]=0000:00

(5) 利用中断向量指向自身,而中断向量本身恰好又是一段指令的方法,来迷惑、欺骗破译者,

即使是有经验的程序员也很难识破；

例如：我们把中断向量值 FBCC：4340 送到中断向量地址为 FBCC：4340 处(即 FBCC：4340＝10000：00＝0：00)即送到 INT0 的中断向量地址处，而 4340，FBCC 恰好又是如下一段指令(在内存中的顺序是 40 43 cc fb)：

```
0:0 40 INC AX
0:01 43 INC BX
0:02 CC INT 3
0:03 FB STI
```

(6) 在程序中将除数置零，然后执行除法运算，从而产生了被零除中断 INT0，操作系统自动调用 INT0 中断处理程序，使你无法跟踪；

程序 2

```
1188 :326 MOV SI,OFBCC
     :329 MOV DS,SI ;DS=FBCC
     :32B MOV SI,434C ;SI=434C
     :32E POP AX ;AX=2E8F,因为上次压栈的值是 2E8F
     :32F MOV [SI],AX ;AX 送到 FBCC:434C=0:0C 即 INT3;偏移处
     :331 MOV DI,SI ;DI=SI=434C
     :333 ADD SI,+2;SI=434E
     :336 MOV [SI],CS ;DS 送到 FBCC:434E=0:0E 即 INT3;段址处
     :338 MOV BX,DS
     :33A SUB BX,B442
     :33E SUB BX,SI
     :340 SUB BX,-8
     :343 MOV CL,04;CL=04
     :345 SHL BX,CL
     :347 ADD BX,8;计算出的 BX 值是 4348
     :34A MOV AX,CS;AX=FBCC
     :34C MOV SI,BX;SI=4348
     :34E MOV [SI-02],CS ;CS 送到 FBCC:4346=:06 即;INT1 段址处
     :351 ADD BX,E763;BX=2AAB
     :355 MOV [SI-4],BX ;BX=2AAB 送到 FBCC:4344=0:04;
                    即 INT1 偏移处,此时 INT1;的中断向量为 CS:2AAB
     :358 MOV [SI-6],DS;DS=FBCC 送到 FBCC:4342=0:02,即 INT0 段址处
     :35B SUB SI,8 ;SI=4340
     :35E MOV [SI],SI ;SI=4340 送到 FBCC:4340=0:0;
                      即 INT0 偏移处,使 INT0 的中断向量指向自身
     :360 MOV BX,CX;BX=4
     :362 XCHG BH,BL ;BL=0,BH=4
     :364 MOV SI,2B9;解密数据因子地址
     :367 MOV CX,0EA ;循环计数
     :36A NOP
     :36B PUSH DS
```

```
:36C PUSH CS
:36D POP DS;DS=CS
:36E POP ES;ES=FBCC
:36F MOV DX,0000;DX=0,以后程序要用的解密数据
:372 LODSB ;取解密数据因子
:373 MOV [01B],AL
:376 XOR AH,AH ;AH=0
:378 DIV BH ;当 BH=0 时,就会产生被零除 INT0 中断,这时,你就不能继续跟踪
:37A ADD DX,AX
:37C ES:
:37D DEC WORF PTR [DI];ES:[DI]=FBCC:434C=0:0,即 INT3 的中断向量的每一
     次减一
:37F MOV BL,[01B]
:383 LOOP 372
:385 MOV SI,2B9
:388 MOV CX,00EA
:38B DEC BH ;BH 减一,为产生被零除中断
:38D JNS 372
:38F MOV SI,08B5
:392 PUSH DS
:393 POP ES
.............
```

①首先，在 1188:32E～1188:336 处置好地址为 FBCC:434C=0000:0C 即 INT3 的中断向量内容,其值为 CS:2E8F;

②然后，在 1188:358～1188:35E 处置好地址为 FBCC:4340=0000:0 即 INT0 的中断向量内容，其值为 FBCC:4340。使 INT0 的中断向量指向自身，而 FBCC:4340(在内存中的顺序是 4043CCFB)是下列指令：

```
INC AX
INC BX
INT 3
CLI
```

当产生被零除中断、系统调用 INT0 时，就等于执行上述指令；

③在 1188: 372～1188:38d 这层循环中。当循环第 4 次时，在:378 处产生被零除中断，在产生被零除之前，我们要记下 AX,BX,DX 的值,因为后面的程序要用这个值进行解密；

④执行 INT0 中断程序(即 INC AX, INC BX,INT 3,CLI)时，在 AX、BX 的内容加一之后，执行 I NT3 的中断程序。INT3 的中断向量内容最先被置成 CS:2E8F,但在 372－:383 这层循环的:37D 处，INT3 的中断向量的偏移内容每次被减 1。这样产生被零除中断时,INT3 的中断向量的偏移内容实际值为 2E8F－4＊EA＝2AE7,即在执行 INC AX、INC BX 之后，程序转到 CS:2AE7 处执行。

程序 3

入口参数:AL＝91 BL＝CC DX＝8B89 DI＝434C DS＝CS ES＝FBCC

```
1188 :2AE7 ADD SP,+06 ;弹出调用 INT3 时的入栈,使其不返回调用程序
```

```
;2AEA MOV [2B37],BL ;BL=0CC 恰好是INT3的指令代码,即将在1188:2B37处的
  指令修改成INT3.
;2AEE MOV BP,SP ;BP=SP 保留栈顶指针
;2AF0 SUB BL,AL ;BL=0CC-091=03B
;2AF2 CLD
;2AF3 PUSH DS
;2AF4 PUSH ES
;2AF5 PUSH DI
;2AF6 MOV AX,ES;AX=FBCC
;2AF8 SUB AX,0AFD
;2AFB MOV ES,AX ;ES=FCF
;2AFD ADD AX,0251 ;AX=F320
;2B00 MOV DI,AX;DI=F320
;2B02 STOSW
;2B03 ES:
;2B04 MOV [DI],ES ;把值FBCC:F320送到地址是FBCC:F320=00:10即INT4的中
  断向量地址处,使INT4的中断向量指向自身,而FBCC:F320又是AND BL,DH、
  IRET 指令代码.
;2B06 POP DI;DI=434C
;2B07 POP ES ;ES=FBCC
;2B08 MOV DH,0C5
;2B0A ADD DH,BL
;2B0C MOV BL,DH
;2B0E MOV SI,2AE7;解密数据因子的地址
;2B11 LODSB
;2B12 ADD BL,AL
;2B14 INT0 ;当溢出时调用溢出中断INT4,否则为空指令
;2B15 CS:
;2B16 MOV AL,[1A]
;2B19 OR AL,02
;2B1B OUT 21,AL;禁止键盘产生中断
;2B1D ES:
;2B1E MOV WORD PTR [DI],2AB1 ;置FBCC:
      ;434C=0:0即INT3的中断向量为CS:2AB1
;2B22 CMP SI,2B37
;2B26 JB 2B11 ;以上的程序是为了计算解密数据BL
;2B28 MOV SI,2AB1 ;被解密代码的地址
;2B2B MOV CX,002A
;2B2E LODSB
;2B2F XOR AL,BL
;2B31 MOV [SI-01],AL ;解密
;2B34 LOOP 2B2E
```

```
:2B36 POP DX
:2B37 IRET;此处在程序开始时被置成 INT3
:2B38 ............
```

(7) 程序设置状态寄存器陷井标志位 TF,产生陷井中断。操作系统自动调用 INT1。当该位置1时,紧接着执行一条指令后便产生一个内部中断,以便你能逐条查看程序。标志位 TF 是为系统而设计的,一般禁止编程者使用,所有的指令也不影响该标志位。所以就很容易疏忽这一点而被引入歧途。

程序 4

```
入口参数:DI=434C DX=4B49 ES=FBCC
1188 :2A81 STI
     :2A82 SUB DI,+0C;DI=434C
     :2A85 MOV BX,ES ;BX=FBCC
     :2A87 MOV SI,2AAB ;被解密代码的首址
     :2A8A MOV CX,003C
     :2A8D NOP
     :2A8E LODSB
     :2A8F XOR AL,BL ;解密
     :2A91 CS:
     :2A95 MOV AL,[1A]
     :2A98 OR AL,02;禁止键盘产生中断
     :2A9A OUT 21,AL
     :2A9C LOOP 2A8E
     :2A9E MOV [BP-02],BX ;BP=SP+6,且 BX=FBCC.BX 的内容送入堆栈
     :2AA1 MOV [BP-04],BX ;BX 的内容送堆栈
     :2AA4 ES: ;ES=FBCC DI=4340
     :2AA5 MOV AX,[DI];取 FBCC:4340=00;0 即 INT0 的偏移地址值 4340 送到 AX
     :2AA7 MOV [BP-06],AX ;AX 的内容送堆栈
     :2AAA IRET
     :2AAB ..........
```

说明:在执行中断调用,系统依次自动压入标志寄存器、段地址寄存器、返回时执行的指令地址。2A9E-:2AAA 段指令看似修改调用该程序(即调用 INT3)时,压入的三个堆栈,以便 IRET 能够返回到 INT0 中断服务程序中去。其实不然,它在 1188:2A9E 处置好了要弹出到状态寄存器的值,此值为:

BX=FBCC=1111101111001100

其 TF 位正好为 1,程序中这一点做得非常隐蔽。

(李晓辉)

### 1.3 VDISK 的妙用

DOS 3.0 以上版本提供了一个名为 VDISK.SYS 的虚拟磁盘驱动程序,从而使用户能自由地按自己需要建立适当大小的 RAM 虚拟盘。使用中发现,2.X 的 DOS 版本利用 3.0 所提供的驱动程序也可以使用 RAM 虚拟盘。

对双软驱无硬盘系统的用户,在从应用程序中退出时常会碰到"Insert disk with COMMAND



此为试读,需要完整PDF请访问: www.ertongbook.com