DEBUG 应用实例

黄焕如　（江西拖拉机发动机厂微机室）

摘要:*DEBUG 是 DOS 系统中重要的文件之一,也是目前应用最广泛的调试程序。本文结合一般微机用户经常遇到的问题,将 DEBUG 中 19 个命令归类介绍,并列举了一些实例,能使初学者在短时间内熟悉和掌握。*

DEBUG 提供了一个可控制的检测环境,使用户能监视和控制程序和执行,直接对可执行文件进行修改和补充,而不必重新汇编源程序,允许装入、修改或显示任何文件和执行目标文件。在 DOS 状态下,为启动 DEBUG 可执行格式:

[驱动器][路径]DEBUG[驱动器][路径][文件名[扩展名]][参数 1][参数 2]

进入 DEBUG 后,寄存器的值初始化如下:段寄存器 CS、DS、ES、SS 设置在空闲内存的底部;指令寄存器 IP 置 100H;SP 设置中该段末尾或装入程序暂存部分的底部中位置较低的部分;其余寄存器 BP、SI、DI、AX、BX、CX、DX 置零,如果用 DEBUG 打开某一文件,则 BX 和 CX 含该文件总字节数,BX 为高位,CX 为低位;标志寄存器设置为:NV UP EIOL NZ NA PO NC ;在代码段中,缺省的驱动器传输地址 DTA 为 CS:80H。如果由 DEBUG 装入 EXE 文件,则 DEBUG 将重新定位,并将段寄存器、堆栈指针和指令指针设置成文件中规定的值,DS 和 EX 将程序段前缀指向最低可利用的段上,BX 和 CX 指出文件的大小。如果由 DEBUG 装入 HEX 文件,则该文件应是 16 进制格式,将立即转换成可执行文件。

DEBUG 共有 19 个命令,为便于用户学习和掌握,按其功能和操作归纳为以下几类,结合实例介绍如下:

1.编写和修改汇编程序

A(汇编)　格式: A[地址]

功能: 将汇编语句直接写入内存,不需编译和链接。

N(命名)　格式: N[驱动器][路径]文件名[扩展名][参数]

功能: 将文件FCB格式化到CS:5C和6C中,供L、W 使用。

R(寄存器)格式: R[寄存器名]

功能: 显示修改寄存器内容,显示当前寄存器和下一条将执行的指令。

[示例一]将文件属性改为隐含(A、R、N、W、Q 命令)

利用 DEBUG 编写一个小型文件,将文件属型改为隐含,可排除 DOS 的 DIR 命令的查找。

```
C>DEBUG
-A 100
 0100 MOV CL,3
 0102 MOV CH,0
 0104 MOV BL,[80]
 0108 MOV BH,0
 010A MOV BYTE PTR[BX+81],0
 010F MOV DX,82
 0112 MOV AX,4301
 0115 INT 21
 0117 JB 11B
 0119 INT 20
 011B MOV DX,0124
 011E MOV AH,9
 0120 INT 21
 0122 INT 20
 0124 DB 'ERROR!',24H
-RCX
 CX 0000
 :2E
-N HMOD.COM
-W
-Q
```

[示例二]修改 DISKCOPY 命令(A、R、W、Q 命令)

DISKCOPY 命令是 DOS 的一条外部命令,其功能是把源驱动器中软盘的内容拷贝到目标驱动器中盘上,也即所谓的整盘拷贝。该命令的缺点是仅仅适用于拷贝单盘(或多盘)单份的情况,而拷贝单盘(或多盘)多份时就不太方便, 例如需要将某一软盘复制成十张相同的盘,利用该命令的原功能,必须读十次盘写十次盘,这不仅浪费

时间,而且也加剧了驱动器内磁头的磨损。由于在第一次读盘时,原盘的内容已经读入内存,如果能在拷贝完成后自行选择是否再读盘或直接将内存中的内容再写盘,就既能适应于拷贝单盘(或多盘)单份的情况,又可适用于拷贝单盘(或多盘)多份的情况。具体修改方法:

```
C>DEBUG DISKCOPY.COM(以 VER 2.1 为例)
 -RCX
  CX OA10 (查原文件字节数)
  :
 -A0240
  0240 JMP 0B11 (转新加的程序)
  0243 NOP
  0244 NOP
  0245 NOP
 -A0B11
  0B11 MOV DX,0B40
  0B14 MOV AH,09
  0B16 INT 21 (显示是否读盘揭示)
  0B18 MOV AH,07
  0B1A INT 21 (键盘输入)
  0B1C AND AL,5F (转为大写字母)
  0B1E CMP AL,59 (是否为 Y?)
  0B20 JZ 0B28
  0B22 CMP AL 4E (是否为 N?)
  0B24 JZ 0B34
  0B26 JMP 0B18
  0B28 AND WORD PTR [01AA],1C00
  0B2E JMP 01D7
  0B31 JMP 0248
  0B34 MOV WORD PTR [01AA],050C
  0B3A JMP 0208
  0B3D JMP 0248
  0B40 DB 0A,0D,'Rdad another diskette?(Y/N)
$'
 -RCX
  CX 0B60
 -W
 -Q
```

经过以上修改后的 DISKCOPY 命令,当拷贝软盘工作完成后,出现提示:

Copy another (Y/N)?

如键入 N 就退到 DOS 揭示符下,如键入 Y 则出现是否读盘的选择:

Read another diskette(Y/N)?

如键入 N 就不再读盘,而直接将内存中信息写入插在当前驱动器的软盘下,如键入 Y 就重新读盘,实际上执行原程序的读写命令。

[示例三]备份和修复硬盘主引导扇区(A、R、N、W、Q 命令)

硬盘自举失败或被病毒侵蚀往往是硬盘主引导扇区被破坏,因此为硬盘主引导扇区备份或者从同类型的硬盘获取主引导扇区来修复硬盘是很有必要的。由于硬盘主引导扇区不属于 DOS 管辖范围,DEBUG 和 DOS 功能调用无法直接对其读写,通常用 BIOS 的 INT13H 进行硬盘绝对读写,一般可利用 DEBUG 直接编写小程序来解决这一问题。

备份硬盘主引导扇区程序:

```
C>DEBUG
 -A 100
  0100 MOV AX,201
  0103 MOV BX,200
  0106 MOV CX,1
  0109 MOV DX,80
  010C INT 13
  010E MOV AL,0
  0110 MOV BX,200
  0113 MOV CX,1
  0116 MOV DX,2CE
  0119 INT 26
  011B INT 20
  011D
 -RCX
  CX 0000
  :1D
 -NA:RBOOT.COM
 -W
 -Q
```

恢复硬盘主引导扇区程序:

```
C>DEBUG
 -A 100
  0100 MOV AX,0
  0102 MOV BX,200
  0105 MOV CX,1
  0108 MOV DX,2CE
  010B INT 25
  010D MOV AL,301
  0110 MOV BX,200
  0113 MOV CX,1
  0116 MOV DX,80
  0119 INT 13
```

```
011B INT 20
011D
-RCX
CX 0000
:1D
-NA:RBOOT.COM
-W
-Q
```

[示例四]压缩汉字库节省内存(R、W、Q 命令)

对于内存较小的微机来说需缩小汉字库才能运行诸如 FoxBASE 等较大的软件,所谓压缩是指将字库中不常用的汉字部分去掉,对于常用的 16 点阵汉字库来说,每个汉字或字符占 32 字节,每区 94 个字,8-15 共 8 个区为空,一般可使用 L=(终止区号-8)×94×32 来计算字库长度,然后转换成 16 进制。一级字库占 55 区,二级字库占 32 区,且常用字均在一级字库中。

```
C>DEBUG CCLIB
-R CX
CX:XXXX
:输入 L 的低四位
-R BX
BX:XXXX
:输入 L 的高位
-W
-Q
```

2.读写磁盘绝对扇区

L(装入) 格式: L[地址[驱动器起始扇区扇区数]]

功能: 用于装入文件或磁盘扇区。

W(写入) 格式: W[地址[驱动器起始扇区扇区数]]

功能: 用于装入文件或磁盘扇区。

[示例五]修复硬盘 DOS 引导记录(L、W 命令)

利用 DEBUG 读取、转储、重写 DOS 分区引导程序,应注意相应的硬盘类型和 DOS 版本。

首先启动正常的硬盘,并在 A 驱动器中插入一张已格式化的盘,然后执行:

```
C>DEBUG
-L 100 2 0 1 (装入硬盘 C 的引导记录)
-W 100 0 60 1 (写入软盘保存)
```

如果硬盘出现引导扇区的故障,可用软盘启动,在 A 驱器中插入引导记录的备份盘,再利用 DEBUG 恢复正确的引导记录。

```
-L 100 0 60 1 (装入软盘引导记录备份)
-W 100 2 0 1 (复盖硬盘 C 引导扇区)
```

值得注意的是,L 和 W 命令读写磁盘时,扇区数应小于或等于 80H 扇区,尤其在写硬盘引导扇区时应特别注意,否则可能产生意想不到的后果。例如:

```
-L 100 2 0 100 (注意扇区数已超过 80H)
(其他操作)
-W 100 2 0 100 (将使得后 20H 扇区的数据重新从内存开始处安装,复盖了开始 20H 扇区数据,严重破坏了硬盘主引导扇区、DOS 引导扇区等)
```

3.程序跟踪调试

T(跟踪) 格式: T[=起始地址][步数]

功能: 用于跟踪执行指令。

G(执行) 格式: G[=起始地址][断点地址[断点地址…]]

功能: 用于执行程序,直至结束或到达指定断点地址。

P(步进) 格式: P[=起始地址][步数]

功能: 使执行子程序调用、循环指令、中断、重复字符串指令后,发出 P 命令,以便回到下一条指令。

[示例六]利用 BIOS 执行低格式化硬盘 (G、A 命令)

当硬盘出现故障时,有时需要对硬盘进行低格式化,利用 BIOS 固化序来低格式化硬盘是一个简单有效的好方法。由于各种机型的入口地址不同,以下仅以 PC/XT 及其兼容机为例,介绍如下:

```
C>DEBUG
-G=C800:0005
```

如果用上述方法不能成功,也可用 INT13H 中断的 7 号子功能来对硬盘低格式化。

```
C>DEBUG
-A 100
0100 MOV AX,0703 (7H 号子功能,交错数为 3)
0103 MOV CX,0001 (0 道 0 扇区始)
0106 MOV DX,0083 (C 盘 0 磁头始)
0109 INT 13
010B INT 3
-G
```

低格式化完成后,再使用 FDISK 对硬盘分区,然后执行高级格式化命令即可。

DEBUG 动态调试跟踪命令主要由 T、P、G 等命令组成,由于这些命令的功能存在一些缺陷,同时针对

DEBUG 的反动态跟踪技术和工具日趋成熟,使得仅仅利用 DEBUG 对大多数软件的解密很难实现,因此改进 DEBUG 调试跟踪功能的各种技术和方法应运而生,限于编幅不再详叙。

4.输入输出端口

I(输入) 格式: I[入口地址]

功能: 从指定端口输入并显示一个字节。

O(输出) 格式: O[入地址][字节]

功能: 将字节发送到指定端口。

[示例七]从串行口 1 读入一字节,并显示其值。(I 命令)

```
C>DEBUG
-I 3F8
```

[示例八]屏蔽全部中断并封锁键盘。(O 命令)

本例将屏蔽字 FFH 送往 8259 中断控制器的中断屏蔽寄存器,将屏蔽所有中断,包括键盘不再有反应,必须重新启动机器。

```
C>DEBUG
-0 21 FF
```

5.显示内存及源代码

D(转储) 格式: D[地址]或[范围]

功能: 用于显示内存中的内容。

U(反汇编) 格式: U[地址]或[范围]

功能: 用于反汇编内存中的指令。

[示例九]DEBUG 汉化(U、E、W、Q 命令)

西文软件的汉化主要指输入和输出两方面,DEBUG 的输入是通过调用 DOS 中断 AH=0AH 实现,本身就可以接收汉字,不需修改,汉化的关键是输出汉字。以 Ver3.1 版本为例,经分析有两处子程序与此有关:

0406H 处 and al,7f 将大于 7f 的字符屏蔽掉了;056D 处将英文中不可打印字符用点显示(.),反汇编原程序如下:

```
C>DEBUG
-U 056D
 056D LODSB
 056E CMP AL,7F
 0570 JAE 0576
 0572 CMP AL,20
 0574 JAE 0578
 0576 MOV AL,2E
 0578 STOSB
 0579 LOOP 056D
```

显然仅仅需要将 0406 和 056D 处修改成空指令即可。不同的版本汉化的地址不同,例如 Ver 3.3 版只需作如下修改:-E OAFE 90 90 90 90 -W -Q

6.检索修改内存

S(检索) 格式: S范围字符

功能: 用于检索范围地址中含字符的地址,找出匹配单元并显示地址。

C(比较) 格式: C 范围字符

功能: 用于比较内存中两数据块的内容。

E(改写) 格式: E 地址[字符]

功能: 用指定字符替换从指定地址开始的内容,或者按顺序方式显示、修改字节。

F(填充) 格式: F 范围字符

功能: 用指定字符填写指定地址内存中的单元。

M(移动) 格式: M 范围地址

功能: 用范围指定内存单元内容传送到指定地址开始的单元中。

[示例十]BASIC 语言加 P 参数存盘文件的解密(E、R、N、W、Q 命令)

在使用 BASIC 语言,用 P 参数存盘时,该文件只能执行而不能列表,或者说文件已经被加密。这是由于 BASIC 解释程序对源程序译码时,在工作区内设置了一个"软开关"标志,它将以是否置位来决定是否响应列表的请求。因此可采用这样的破译方法,先将需要解密的源程序从磁盘中读入内存,然后再读入一个非 P 参数存盘的文件,使"软开关:标志复位,再进行列表操作。为通用起见,可利用 DEBUG 编制一个两字节的短文件,为加 P 参数存盘文件解密,具体步骤如下:

```
C>DEBUG
-E 0100 FF FF
-RCX
 CX 0000
 :0002
-NPPP.BAS
-W
-Q
C>BASICA
 LOAD"文件名"(加 P 参数存盘)
```

LOAD"PPP"
LIST

[示例十一]WS 工作参数的修改(E、W、Q 命令)

中文 WorkStar 是目前得到广泛使用的字处理软件之一,其短小精悍、操作方便而很受用户欢迎。WS 的工作参数区设置在 200H-400H 内,例如:偏移地址(下同)0248 为每屏显示行数、0249 为每行显示列数、0284 为菜单揭示区色彩(0-7F)、028B 为编辑文本区色彩(0-7F)、02DC 为系统磁盘号(1、2、3、表示 A、B、C 盘)、0360 为帮助级别(0-3)、0363 为目录显示状态(00 关闭、FF 为显示)、0390 为默认编辑方式(00 为 D、FF 为 N)等。知道了工作参数在内存中的地址,很容易利用 DEBUG 的 E 命令修改工作参数,例如将 WS 设置成每屏显示行数为 15 行、字符颜色为绿色:

C>DEBUG WS.COM
-E0248 OF
-E0248 02 (0,1,…7 为黑,蓝,绿,青,红,棕,黄,白 8,9,…F 为加强色)
-W
-Q

7.计算和退出

H(运算) 格式: H 值 1 值 2

功能: 计算两个数的和差

[示例十二]计算 16 进制数的和差(H、Q 命令)

在程序设计中,当需要计算地址的累加值时,可直接利用 DEBUG 的 H 命令,如果需要计算的数值很多,可结合使用 DOS 的管道命令。例如:

C>COPY CON LSO.TXT
H 值 1 值 2
H 值 3 值 4
……
Q
C>TYPE LSO.TXT|DEBUG>LS1.TXT (在 LS1.TXT 文件中含有全部计算结果)

Q(退出) 格式: Q

功能: 退出DEBUG.