DEBUG 命令及应用

DEBUG是一个调试软件，共有19条命令，具有指令语句输入、指令单步执行、程序设断点执行、寄存器或存储器显示和赋值、磁盘文件和扇区读写等功能。为操作人员提供了可以跟踪、测试程序的环境和条件。

DEBUG命令分类

- 指令输入、汇编及反汇编

  DEBUG是DOS系统代码调试软件，由它输入的汇编指令程序，属于半汇编语言形式，不能接受汇编语言标号和多数伪指令。

  A：输入指令并完成汇编

  U：对目标代码反汇编。U不区分目标代码是数据还是程序，一律按指令代码反汇编输出。

- 显示、修改寄存器和存储器单元内容

  R：显示和修改寄存器内容

  D：显示存储器单元内容

  E，F：修改存储器单元内容

- 执行程序

T，P：跟踪程序执行；　G：连续或设断点执行程序

- 数据块比较、搜索及移动

C：比较两个内存数据块；　S：在数据块中进行搜索

M：实现数据移动

- 磁盘文件或扇区的读/写

N：定义文件名；　W：写磁盘文件或扇区

L：读磁盘文件或扇区

- 算术运算、输入输出

H：实现两个16进制数加减运算

I：从指定端口读数据；　O：把字节数据写到端口

- 其它

Q：退出DEBUG状态返回到DOS状态

W：存盘

DEBUG命令应用说明

- 所有**DEBUG**命令均是一个字母，后面带有或不带有参数。
- 命令和参数可用大写、小写或大小写混合方式输入。
- 命令和参数可以用空格或“，”分隔。

  例如：

  -D CS:100 110

  -D CS:100, 110

  -D, CS:100 110

等效。

- DEBUG默认输出输入数据或地址是16进制数，不加后缀H。
- 每个命令有ENTER键结束。
- Ctrl+Break键或Ctrl+C键可中断一个命令的执行返回到“-”提示符。
- Ctrl+Num LOCK键可暂停屏幕上卷，按任意键继续。
- DEBUG检查输入指令语法错误。例如：

```
-D CS:100. 1000
                ^Error
-A
0CE8:0100 mov ax 2000
                  ^ Error
```

- **DEBUG**允许使用**DOS**手册介绍的控制键和编辑键。
- 汇编语言语法说明：

1）前缀助记符必须在相关指令之前输入，且可分行输入。

2）段跨越助记符是**CS：**、**ES：**、**DS：**、**SS：**。

3）远返回（段间返回）助记符是**RETF**。

4）汇编命令根据转移目标地址位移量，能自动汇编短、近或远的转移和调用指令。

5）支持所有形式的寄存器间接寻址。

ADD BX，54[BP+2][SI-9]

POP [BP+DI]

6）支持所有同义词的指令码

DEBUG主要命令

- A（Assemble）汇编

  A [内存地址]：从内存指定地址开始输入汇编指令，并汇编成机器码。若其后的地址参数缺省，则从上一次汇编命令最后指令的下一个字节单元开始输入，否则，从CS：0100H单元开始输入。默认段地址是CS。

- U(Unassemble)反汇编

  U[起始地址]；　　U[起始地址　终止地址]

  U[起始地址　L字节数]

  对内存目标代码反汇编。屏幕显示地址、目标代码及汇编符号指令，L后面的参数是反汇编的字节数。

  若U后面的参数缺省，则从上一次U命令最后一条指令的下一个地址开始反汇编，否则，从CS：0100H地址开始反汇编。若起始地址只含偏移地址，则默认段地址是CS。

- D (Dump)显示存储器

  D［起始地址］

  D［起始地址　终止地址］； D［起始地址，L 字节数］

  显示内存指定单元的内容。显示屏幕的左半部分是十六进制形式，一个字节两位数；右半部分是对应的ASCII码字符，不能被打印的字符用“.”表示。左半部分“-”号用来区分8个字节区。

不指定段地址时，自动显示DS的内容。如果只指定首地址，则显示从首地址开始的80个字节的内容。如果完全没有指定地址，则显示上一个D命令显示的最后一个单元后的内容。

- E（Enter）修改寄存器

  E 地址[值表]

  修改存储器单元内容。若值表参数缺省，则只显示、修改一个字节内容；若只含偏移地址，则默认DS。

  值表中每项占一个字节，各项由空格或“，”分隔，值表用来替换存储器单元原来的内容。

  输入E命令后，显示该单元地址和内容，等待用户修改，可按下列两种方法操作：

（1）按空格键：完成一个字节的显示和修改，且显示下一个字节地址和内容等待修改。

（2）按“-”键：完成一个字节的显示和修改，且显示前一个字节地址和内容等待修改。

按回车终止E命令，结束修改。

例 **e 140b:000e f3'x'**

则用两个字节的**f3**、**'X'**内容代替相应的内容两个存储单元的内容。

- R（Register）显示或修改寄存器

R[寄存器名]

显示或修改指定寄存器内容，若缺省，则显示所有CPU寄存器内容和由CS：IP指示的地址及指令。

- G（GO）执行程序

G[起始地址]

G[起始地址　断点地址1，断点地址2，…]

从起始地址开始执行程序，执行到断点地址时停止，同时显示所有寄存器和标志寄存器内容，并指出下一条要执行的指令地址和指令。

若命令地址缺省，则从CS：IP开始执行。默认段地址是CS。

当设有断点时，程序执行到第一个断点地址（顺序执行，遇到最小断点地址）停下，且显示当前所有寄存器内容。断点最多设十个，且与顺序无关。

例 —r

例 —g 0009

(假设CS的值为：0c1c)

- T (Trace)跟踪命令

T[=起始地址]

T[=起始地址　指令条数]

从起始地址开始，逐条跟踪指令的执行，若在命令参数中给出了指令条数，则执行完这些指令后停下，否则执行一条指令就停下。执行时，屏幕显示每条指令执行后各寄存器内容。

例　－t=0003　3

若起始地址缺省，则从CS：IP开始执行，或接上次T命令的下条指令执行。

一些常见的出错信息

1、Register already defined 汇编内部出现逻辑错误
2、Unknown symbol type 在符号语句的类型中，有些不能识别的东西
3、Symbol is multi-defined 重复定义一个符号
4、Symbol not defined 符号没有定义
5、Syntax error 语句的语法与任何可识别的语法不匹配
6、Symbol is reserved word 企图非法使用一个汇编程序的保留字(例：定义add为一变量)
7、Not proper align/combine type SEGMENT参数不正确
8、One operand must be const 加法指令的非法使用
9、Operands must be same or 1 abs 减法指令的非法使用
10、Already have base register 试图重复基地址

11、Illegal size for item 引用的项的长度是非法的，（如：双字移位）

12、Illegal register value 指定的寄存器值不能放入“reg” 字段

13、Must be AX or AL 某些指令只能用AX或AL

14、Improper use of segment reg 段寄存器使用不合法（如：mov ds,0）

15、Division by 0 or overflow 给出一个用0作除数的表达式

16、Value is out of range 数值大于需要使用的

17、CS register illegal usage 试图非法使用CS寄存器

18、DUP is too large for linker
DUP嵌套太长，以至于从连接程序不能得到所要的记录