DEBUG命令大全

启动 Debug，它是可用于测试和调试 MS-DOS 可执行文件的程序。

Debug [[drive:][path] filename [parameters]]

参数

[drive:][path] filename 指定要测试的可执行文件的位置和名称。

parameters 指定要测试的可执行文件所需要的任何命令行信息。

说明

使用 Debug 命令但不指定要测试的文件

如果使用没有位置和文件名的 Debug 命令，然后键入所有的 Debug 命令以响应 Debug 提示符，连字符 (-)。

Debug 命令

以下是 Debug 命令列表：

? 显示 Debug 命令列表。

a 汇编 8086/8087/8088 记忆码。

c 比较内存的两个部分。

d 显示部分内存的内容。

e 从指定地址开始，将数据输入到内存。

f 使用指定值填充一段内存。

g 运行在内存中的可执行文件。

h 执行十六进制运算。

i 显示来自特定端口的 1 字节值。

l 将文件或磁盘扇区内容加载到内存。

m 复制内存块中的内容

/n 为 l 或 w 命令指定文件，或者指定正在测试的文件的参数。

o 向输出端口发送 1 个字节的值。

p 执行循环、重复的字符串指令、软件中断或子例程。

q 停止 Debug 会话。

r 显示或改变一个或多个寄存器。

s 在部分内存中搜索一个或多个字节值的模式。

t 执行一条指令，然后显示所有寄存器的内容、所有标志的状态和 Debug 下一步要执行的指令的解码形式。

u 反汇编字节并显示相应的原语句。

w 将被测试文件写入磁盘。

xa 分配扩展内存。

xd 释放扩展内存。

xm 映射扩展内存页。

xs 显示扩展内存的状态。

- 分隔命令参数

所有 Debug 命令都接受参数，除了 q 命令之外。可以用逗号或空格分隔参数，但是只有在两个十六进制值之间才需要这些分隔符。因此，以下命令等价：

dcs:100 110

d cs:100 110

d,cs:100,110

- 指定有效地址项

Debug 命令中的 address 参数指定内存位置。Address 是一个包含字母段记录的二位名称或一个四位字段地址加上一个偏移量。可以忽略段寄存器或段地址。a，g，l，t，u 和 w 命令的默认段是 CS。所有其他命令的默认段是 DS。所有数值均为十六进制格式。

有效地址如下：

CS:0100

04BA:0100

在段名和偏移量之间要有冒号。

- 指定有效范围项

Debug 命令中的 range 参数指定了内存的范围。可以为 range 选择两种格式：起始地址和结束地址，或者起始地址和长度范围（由 l 表示）。

例如，下面的两个语法都可以指定从 CS:100 开始的 16 字节范围：

cs:100 10f

cs:100 l 10

++

Debug 子命令

选择 Debug 命令以获得详细信息。

Debug:A（汇编）

Debug:C（比较）

Debug（转储）

Debug:E（键入）

Debug:F（填充）

Debug:G（转向）

Debug:H（十六进制）

Debug:I（输入）

Debug:L（加载）

Debug:M（移动）

Debug:N（名称）

Debug:O（输出）

Debug:P（执行）

Debug:Q（退出）

Debug:r（寄存器）

Debug:s（搜索）

Debug:T（跟踪）

Debug:U（反汇编）

Debug:W（写入）

Debug:XA（分配扩展内存）

Debug:XD（取消分配扩展内存）

Debug:XM（映射扩展内存页）

Debug:XS（显示扩展内存状态）

**********Debug子命令**********

Debug:A（汇编）

直接将 8086/8087/8088 记忆码合并到内存。

该命令从汇编语言语句创建可执行的机器码。所有数值都是十六进制格式，必须按一到四个字符输入这些数值。在引用的操作代码（操作码）前指定前缀记忆码。

a [address]

参数

address

指定键入汇编语言指令的位置。对 address 使用十六进制值，并键入不以“h”字符结尾的每个值。如果不指定地址，a 将在它上次停止处开始汇编。

有关将数据输入到指定字节中的信息，请单击“相关主题”列表中的 Debug E（键入）。

有关反汇编字节的信息，请单击“相关主题”列表中的 Debug U（反汇编）。

范例

a 命令支持所有形式的间接注册命令，如下例所示：

add bx,34[bp+2].[si-1]

pop [bp+di]

push [si])

还支持所有操作码同义词，如下例所示：

loopz 100

loope 100

ja 200

jnbe 200

对于 8087 操作码，必须指定 wait 或 fwait 前缀，如下例所示：

fwait fadd st,st(3) ; this line assembles

; an fwait prefix

说明

使用记忆码

段的替代记忆码为 cs:、ds:、es: 和 ss:。远程返回的记忆码是 retf。字符串处理的记忆码必须明确声明字符串大小。例如，使用 movsw 可以移动 16 位的字串，使用 mov***（文字因故被系统屏蔽）***（文字因故被系统屏蔽） 可以移动 8 位字节串。

汇编跳转和调用

汇编程序根据字节替换自动将短、近和远的跳转及调用汇编到目标地址。通过使用 near 或 far 前缀可以替代这样的跳转或调用，如下例所示：

-a0100:0500

0100:0500 jmp 502 ; a 2-byte short jump

0100:0502 jmp near 505 ; a 3-byte near jump

0100:0505 jmp far 50a ; a 5-byte far jump

可以将 near 前缀缩写为 ne。

区分字和字节内存位置

当某个操作数可以引用某个字内存位置或者字节内存位置时，必须用前缀 word ptr 或者前缀 byte ptr 指定数据类型。可接受的缩写分别是 wo 和 by。以下范例显示两种格式：

dec wo [si]

neg byte ptr [128]

指定操作数

Debug 使用包括在中括号 ([ ]) 的操作数引用内存地址的习惯用法。这是因为另一方面 Debug 不能区分立即操作数和内存地址的操作数。以下范例显示两种格式：

mov ax,21 ; load AX with 21h

mov ax,[21] ; load AX with the

; contents of

; memory location 21h

使用伪指令

使用 a 命令提供两个常用的伪指令：db 操作码，将字节值直接汇编到内存，dw 操作码，将字值直接汇编到内存。以下是两个伪指令的范例：

db 1,2,3,4,"THIS IS AN EXAMPLE"

db THIS IS A QUOTATION MARK:"

db "THIS IS A QUOTATION MARK:"

dw 1000,2000,3000,"BACH"

++

Debug:C（比较）

比较内存的两个部分。

c range address

参数

range

指定要比较的内存第一个区域的起始和结束地址，或起始地址和长度。有关有效的 range 值的信息，请单击“相关主题”列表中的“Debug 说明”。

address

指定要比较的第二个内存区域的起始地址。有关有效 address 值的信息，请单击“相关主题”列表中的“Debug 说明”。

++

范例

以下命令具有相同效果：

c100,10f 300

c100l10 300

每个命令都对 100h 到 10Fh 的内存数据块与 300h 到 30Fh 的内存数据块进行比较。

Debug 响应前面的命令并显示如下信息（假定 DS = 197F）：

197F:0100 4D E4 197F:0300

197F:0101 67 99 197F:0301

197F:0102 A3 27 197F:0302

197F:0103 35 F3 197F:0303

197F:0104 97 BD 197F:0304

197F:0105 04 35 197F:0305

197F:0107 76 71 197F:0307

197F:0108 E6 11 197F:0308

197F:0109 19 2C 197F:0309

197F:010A 80 0A 197F:030A

197F:010B 36 7F 197F:030B

197F:010C BE 22 197F:030C

197F:010D 83 93 197F:030D

197F:010E 49 77 197F:030E

197F:010F 4F 8A 197F:030F

注意列表中缺少地址 197F:0106 和 197F:0306。这表明那些地址中的值是相同的。

++

说明

如果 range 和 address 内存区域相同，Debug 将不显示任何内容而直接返回到 Debug 提示符。如果有差异，Debug 将按如下格式显示：

address1 byte1 byte2 addess2

++++

Debug（转储）

显示一定范围内存地址的内容。

d [range]

参数

range

指定要显示其内容的内存区域的起始和结束地址，或起始地址和长度。有关有效的 range 值的信息，请单击“相关主题”列表中的“Debug 说明”。如果不指定 range，Debug 程序将从以前 d 命令中所指定的地址范围的末尾开始显示 128 个字节的内容。

有关显示寄存器内容的信息，请单击“相关主题”列表中的 Debug R（寄存器）。

++

范例

假定键入以下命令：

dcs:100 10f

Debug 按以下格式显示范围中的内容：

04BA:0100 54 4F 4D 00 53 41 57 59-45 52 00 00 00 00 00 00 TOM.SAWYER……

如果在没有参数的情况下键入 d 命令，Debug 按以前范例中所描述的内容来编排显示格式。显示的每行以比前一行的地址大 16 个字节（如果是显示 40 列的屏幕，则为 8 个字节）的地址开头。

对于后面键入的每个不带参数的 d 命令，Debug 将紧接在最后显示的命令后立即显示字节内容。

如果键入以下命令，Debug 将从 CS:100 开始显示 20h 个字节的内容：

dcs:100 l 20

如果键入以下命令，Debug 将显示范围从 CS 段的 100h 到 115h 中所有字节的内容：

dcs:100 115

++

说明

当使用 d 命令时，Debug 以两个部分显示内存内容：十六进制部分（每个字节的值都用十六进制格式表示）和 ASCII 码部分（每个字节的值都用 ASCII 码字符表示)。每个非打印字符在显示的 ASCII 部分由句号 (.) 表示。每个显示行显示 16 字节的内容，第 8 字节和第 9 字节之间有一个连字符。每个显示行从 16 字节的边界上开始。

++

Debug:E（键入）

将数据输入到内存中指定的地址。

可以按十六进制或 ASCII 格式键入数据。以前存储在指定位置的任何数据全部丢失。

e address

参数

address

指定输入数据的第一个内存位置。

list

指定要输入到内存的连续字节中的数据。

有关集成记忆码的信息，请单击“相关主题”列表中的 Debug A（汇编）。

有关显示内存部分内容的信息，请单击“相关主题”列表中的 Debug D （转储）。

++

范例

假定键入以下命令：

ecs:100

Debug 按下面的格式显示第一个字节的内容：

04BA:0100 EB.

要将该值更改为 41，请在插入点键入 41，如下所示：

04BA:0100 EB.41_

可以用一个 e 命令键入连续的字节值。在键入新值后按 SPACEBAR（空格键），而不是按 ENTER 键。Debug 显示下一个值。在此范例中，如果按三次 SPACEBAR（空格键），Debug 将显示下面的值：

04BA:0100 EB.41 10. 00. BC._

要将十六进制值 BC 更改为 42，请在插入点键入 42，如下所示：

04BA:0100 EB.41 10. 00. BC.42_

假定决定值 10 应该是 6F。要纠正该值，请按 HYPHEN 键两次以返回到地址 0101（值 10）。Debug 显示以下内容：

04BA:0100 EB.41 10. 00. BC.42-

04BA:0102 00.-

04BA:0101 10._

在插入点键入 6f 更改值，如下所示：

04BA:0101 10.6f_

按 ENTER 停止 e 命令并返回到 Debug 提示符下。

以下是字符串项的范例：

eds:100 "This is the text example"

该字符串将从 DS:100 开始填充 24 个字节。

++

说明

使用 address 参数

如果在没有指定可选的 list 参数的值情况下指定 address 的值，Debug 将显示地址和内容，在下一行重复地址，并等待您的输入。此时，您可以执行下列操作之一：

• 替换字节值。为此，请在当前值后键入新值。如果您键入的值不是有效的十六进制值，或该值包含两个以上的数字，则 Debug 不会回显无效或额外的字符。

• 进入下一个字节。为此，请按 SPACEBAR（空格键）。要更改该字节中的值，请在当前值后键入新值。如果按 SPACEBAR（空格键）时，移动超过了 8 位界限，Debug 程序将显示新的一行并在行首显示新地址。

• 返回到前一个字节。为此，请按 HYPHEN 键 (-)。可以反复按 HYPHEN 键 (-) 向后移动超过多个字节。在按 HYPHEN 时，Debug 开始新行并显示当前地址和字节值。

• 停止执行 e 命令。为此，请按 ENTER 键。在任何字节位置都可以按 ENTER。

使用 list 参数

如果指定 list 参数的值，随后的 e 命令将使用列表中的值替换现有的字节值。如果发生错误，将不更改任何字节值。

List 值可以是十六进制字节或字符串。使用空格、逗号或制表符来分隔值。必须将字符串包括在单或双引号中。

++++

Debug:F（填充）

使用指定的值填充指定内存区域中的地址。

可以指定十六进制或 ASCII 格式表示的数据。任何以前存储在指定位置的数据将会丢失。

f range list

参数

range

指定要填充内存区域的起始和结束地址，或起始地址和长度。