第3章 chapter 3

寻址方式


E1(寻址方式引言)

计算机是通过执行指令序列来解决问题的,而指令序列由多条指令构成。汇编语言指令由指令助记符和操作数构成,指令助记符说明指令的功能;操作数由各种不同寻址方式提供。汇编语言指令格式为:

```
指令助记符 [操作数 1],[操作数 2],[操作数 3] [;注释]
```

执行指令过程中,操作数是指令主要处理对象。如何表示操作数或操作数所在位置是一个关键。指令中的操作数可以直接给出,也可以存放到寄存器或存储器中。寻找操作数或操作数位置的方法称为寻址方式。寻址方式可分为三大类,分别为立即寻址方式、寄存器寻址方式和存储器寻址方式。

学习目标:

- 了解寻址方式的概念。
- 掌握立即寻址方式的用法。
- 掌握寄存器寻址方式的用法。
- 掌握存储器寻址方式的用法。


E2(立即寻址)

3.1 立即寻址方式

操作数直接包含在指令中,紧跟操作码之后,作为指令的一部分存放在代码段中,称为立即寻址方式。取出指令同时将操作数取出,这样的操作数称为立即数。立即数类似于C语言中的常数,可以是8位或16位。如果为16位立即数,按小端方式(高高低低)规则进行存储。

源操作数为立即寻址方式的示例如下。

```
-A
073F:0100   MOV AL,78H        ;立即数 78H 传送至 AL
073F:0102   MOV BX,1234H      ;立即数 1234H 传送至 BX
```

通过反汇编命令U,可以查看以上两条指令的机器语言,如图3.1所示。

通过图3.1中显示的信息可知每条机器指令的功能。根据存放机器指令内存单元首

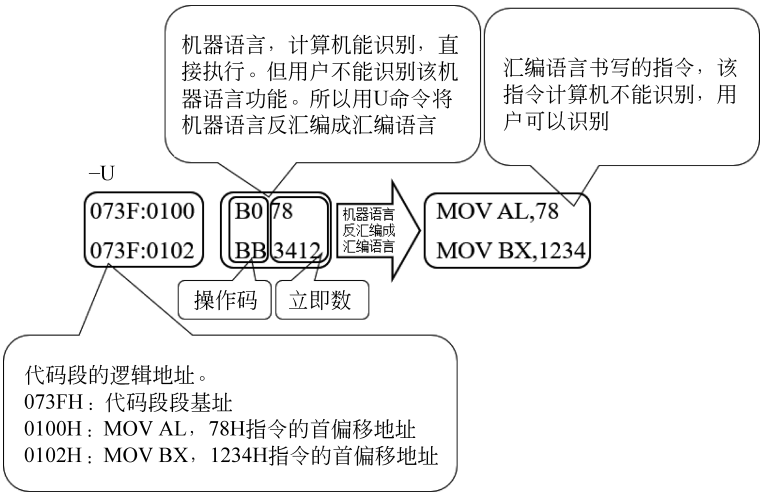

图 3.1 机器语言反汇编成汇编语言

偏移地址，画出机器指令内存示意图，如图 3.2 所示。

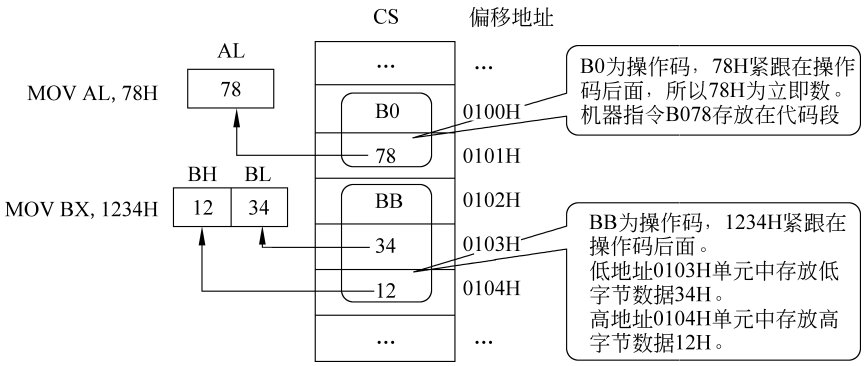

图 3.2 机器指令内存示意图

注意：

目的操作数不能为立即寻址，源操作数可为立即寻址。如下指令不合法：

```
-A
0B0E:0100 MOV 12H,AL
                ^ Error
0B0E:0100 MOV 1234,BX
                  ^ Error
```

如果是字符常数，需加上单引号或双引号。例如：

```
MOV AL,'A'
MOV CX,'AB'
```

或

```
MOV CX,"AB"
```


E3（寄存器寻址）

3.2 寄存器寻址方式

指令中所需操作数存放在CPU内部寄存器中，称为寄存器寻址方式。CPU内部可用的寄存器有8位寄存器和16位寄存器。

8位寄存器有：AL、AH、BL、BH、CL、CH、DL和DH。

16位寄存器有：AX、BX、CX、DX、SI、DI、SP、BP、ES、SS、CS和DS。

寄存器寻址方式可用于目的操作数，也可用于源操作数。

1. 目的操作数为寄存器寻址方式，源操作数为立即寻址方式

```
MOV AX,1278H          ;将立即数 1278H 传送至 AX
MOV CL,56H            ;将立即数 56H 传送至 CL
```

2. 目的操作数为存储器寻址方式，源操作数为寄存器寻址方式

```
MOV VAL,AL            ;将 AL 传送至符号地址 VAL 指向的字节单元
MOV BUF,CX            ;将 CX 传送至符号地址 BUF 指向的字单元
```

其中，VAL为字节类型变量，BUF为字类型变量。

3. 目的操作数和源操作数均为寄存器寻址方式

```
MOV AL,CH             ;将 CH 传送至 AL
MOV BX,DX             ;将 DX 传送至 BX
```

指令中操作数存放在寄存器中，执行指令过程会减少访问内存单元次数，采用寄存器寻址方式能提高指令执行速度。通常情况下，指令中操作数采用寄存器寻址方式，但是寄存器毕竟有限，寄存器寻址方式应根据实际问题来确定。

注意：

📖 两个操作数均为寄存器寻址方式，寄存器必须位数相同。如下指令不合法：

```
-A
0B0E:0100 MOV BX,AL
                  ^ Error
0B0E:0100 MOV CH,DX
                  ^ Error
```


E4（存储器寻址引言）

3.3 存储器寻址方式

指令中所需操作数存放在存储器的存储单元中，称为存储器寻址方式。每一个存储单元由唯一的一个物理地址标识，物理地址通过逻辑地址可求得。

编写指令时，指令中操作数所在存储单元的地址用偏移地址表示，CPU访问存储单元需通过物理地址来访问。根据存储单元偏移地址不同的提供方法，可将存储器寻址方

式分为 5 种，分别为直接寻址方式、寄存器间接寻址方式、寄存器相对寻址方式、基址加变址寻址方式和相对基址变址寻址方式。

3.3.1 直接寻址方式


E5(直接寻址)

指令中直接给出所需操作数偏移地址，该寻址方式称为直接寻址方式。偏移地址需用“[ ]”括起来，紧跟操作码之后，作为指令的一部分存放在代码段中。取指令时，将偏移地址一起取出，通过偏移地址和段基址计算出物理地址。CPU 根据物理地址指向的存储单元取出所需操作数。

一般情况下，直接寻址方式默认数据段 DS；如果使用其他段中的数据，则必须使用段超越前缀进行说明。

1. 源操作数为直接寻址方式

1）字操作

【例 3.1】 设（21000H）＝56H，（21001H）＝89H，（21002H）＝36H，（21003H）＝66H，DS＝2000H，执行 MOV CX，[1001H]后，CX 中数据为什么？

分析：MOV CX，[1001H]指令两个操作数寻址方式及功能分析如下。

源操作数：[1001H]为直接寻址方式，默认数据段 DS，EA＝1001H，PA＝DS×10H＋EA＝2000H×10H＋1001H＝21001H。

目的操作数：CX 为寄存器寻址方式。

功能：该指令将 21001H 指向的字单元中数据传送至 CX。直接寻址方式的寻址规则分为三步，如图 3.3 所示。

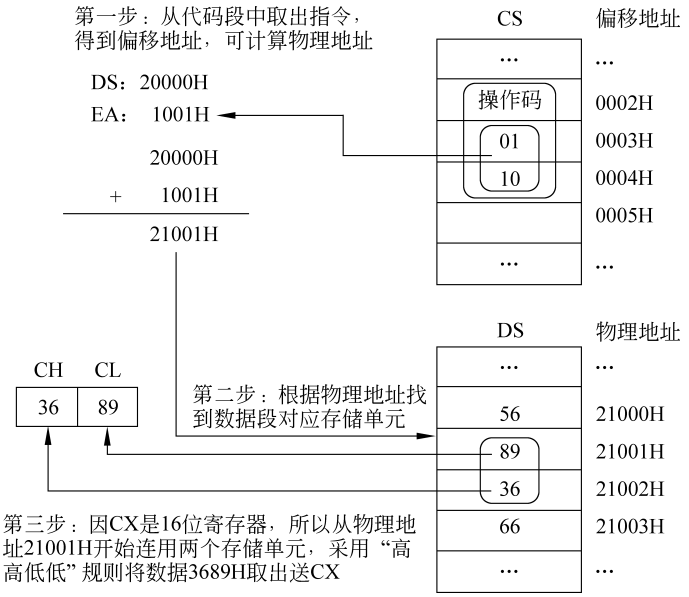

图 3.3 字操作、源操作数为直接寻址方式

2）字节操作

【例 3.2】 设 DS=2000H，(22000H)=90H，(22001H)=29H，(22002H)=16H，执行 MOV BL，[2002H]后，BL 中数据为什么？

分析：MOV BL，[2002H]指令两个操作数寻址方式及功能分析如下。

源操作数：[2002H]为直接寻址方式，默认数据段 DS，偏移地址 EA=2002H，物理地址 PA=DS×10H+EA=2000H×10H+2002H=22002H。

目的操作数：BL 为寄存器寻址方式。

功能：该指令将 22002H 指向的字节单元中数据取出传送至 BL。直接寻址方式的寻址规则分为三步，如图 3.4 所示。

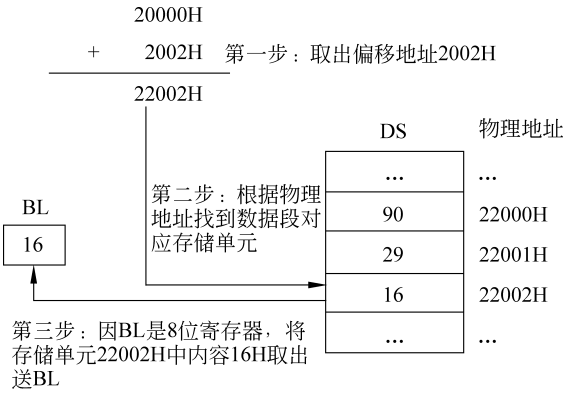

图 3.4 字节操作、源操作数为直接寻址方式

【例 3.3】 设 DS=2000H，ES=3000H，(22000H)=90H，(22001H)=29H，(22002H)=16H，(32000H)=19H，(32001H)=32H，(32002H)=06H，执行 MOV DH，ES：[2002H]后，DH 中数据为什么？

分析：MOV DH，ES：[2002H] 指令两个操作数寻址方式及功能分析如下。

源操作数：[2002H]为直接寻址方式，默认数据段 DS。因为采用段超越，所以使用附加段 ES。偏移地址 EA=2002H，物理地址 PA=ES×10H+EA=3000H×10H+2002H=32002H。

目的操作数：DH 为寄存器寻址方式。

功能：该指令将附加段 32002H 指向的字节单元中数据取出传送至 DH。直接寻址方式过程如图 3.5 所示。

2. 目的操作数为直接寻址方式

1）字操作

【例 3.4】 设 DS=3000H，(32000H)=90H，(32001H)=29H，(32002H)=16H，(32003H)=86H，AX=1030H，执行 MOV [2002H]，AX 后，目的操作数中数据为什么？

分析：MOV [2002H]，AX 指令两个操作数寻址方式及功能分析如下。

源操作数：AX 为寄存器寻址方式。

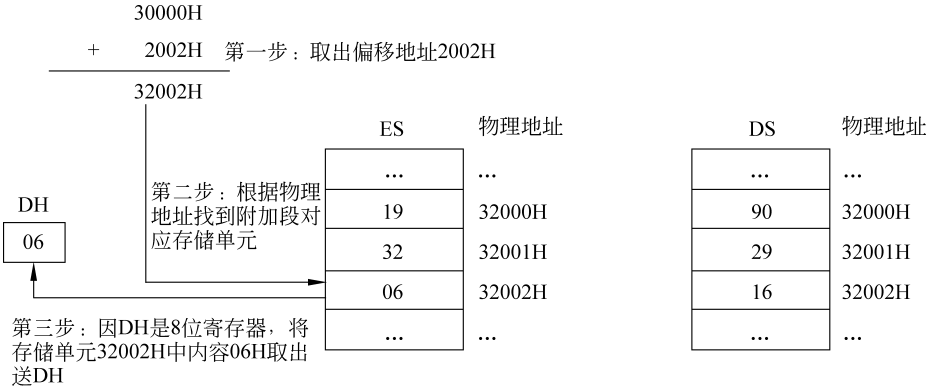

图 3.5 字节操作、源操作数采用段超越的直接寻址方式

目的操作数：[2002H]为直接寻址方式，默认数据段 DS，偏移地址 EA＝2002H，物理地址 PA＝DS×10H＋EA＝3000H×10H＋2002H＝32002H。

功能：该指令将 AX 中数据传送至 32002H 指向的字单元中。直接寻址方式过程如图 3.6 所示。

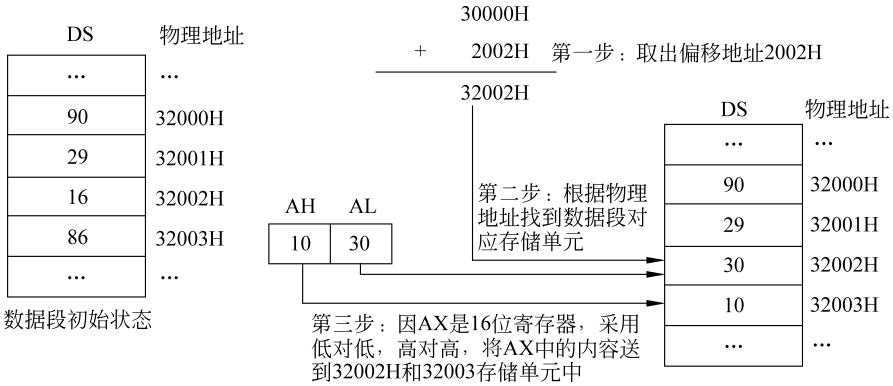

图 3.6 字操作、目的操作数为直接寻址方式

2）字节操作

【例 3.5】 设 DS＝3000H，(32000H)＝90H，(32001H)＝29H，(32002H)＝16H，(32003H)＝86H，CH＝10H，执行 MOV [2002H]，CH 后，目的操作数中数据为什么？

分析：MOV [2002H]，CH 指令两个操作数寻址方式及功能分析如下。

源操作数：CH 为寄存器寻址方式。

目的操作数：[2002H]为直接寻址方式，默认数据段 DS，偏移地址 EA＝2002H，物理地址 PA＝DS×10H＋EA＝3000H×10H＋2002H＝32002H。

功能：该指令将 CH 中数据传送至 32002H 指向的字节单元中。直接寻址方式过程如图 3.7 所示。

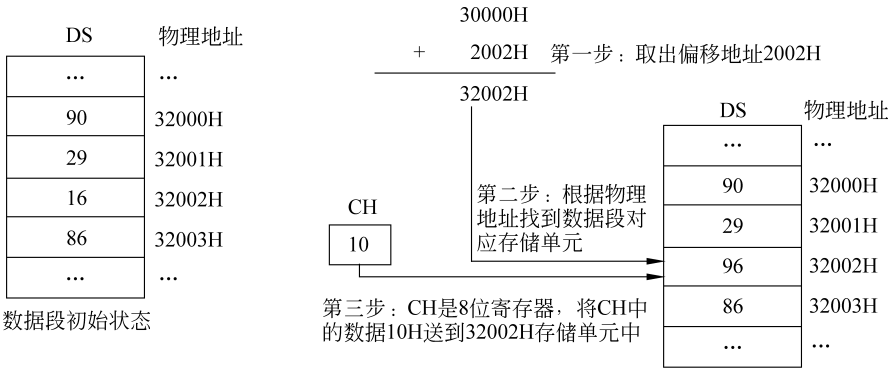

图 3.7　字节操作、目的操作数为直接寻址方式

注意：

直接寻址方式用于处理存储单元中的数据，其操作数是存储变量中的数据，该寻址方式可在 64KB 段内进行寻址。

立即寻址方式与直接寻址方式的区别如下。

```
MOV AX,2002H        ;将立即数 2002H 传送至 AX
MOV AX,[2002H]      ;将 DS 段 2002H 指向的字单元中数据传送至 AX
```


E6（寄存器间接寻址）

3.3.2　寄存器间接寻址方式

指令中的操作数存放在存储单元中，存储单元偏移地址存放在寄存器中，该寻址方式称为寄存器间接寻址。存放存储单元偏移地址的寄存器有 4 个，分别为 BX、BP、SI 和 DI。每个寄存器使用时必须用“[]”括起来。寄存器间接寻址方式若不使用段超越前缀，则规定如下。

偏移地址由 BX、SI 或 DI 寄存器指定，默认数据段 DS。物理地址计算方法：

$$PA=DS\times 10H+BX/SI/DI$$

偏移地址由 BP 寄存器指定，默认堆栈段 SS。物理地址计算方法：

$$PA=SS\times 10H+BP$$

1. 源操作数为寄存器间接寻址方式

字操作。

【例 3.6】 设 DS＝2000H，(22000H)＝90H，(22001H)＝29H，(22002H)＝16H，BX＝2001H，执行 MOV AX,[BX]后，AX 中数据为什么？

分析： MOV AX,[BX]指令两个操作数寻址方式及功能分析如下。

源操作数： [BX]为寄存器间接寻址方式，默认数据段 DS，EA＝2001H，PA＝DS×10H＋EA＝2000H×10H＋2001H＝22001H。

目的操作数： AX 为寄存器寻址方式。

功能： 该指令将 22001H 指向的字单元中数据传送至 AX。寄存器间接寻址方式的

寻址规则分为三步，如图 3.8 所示。

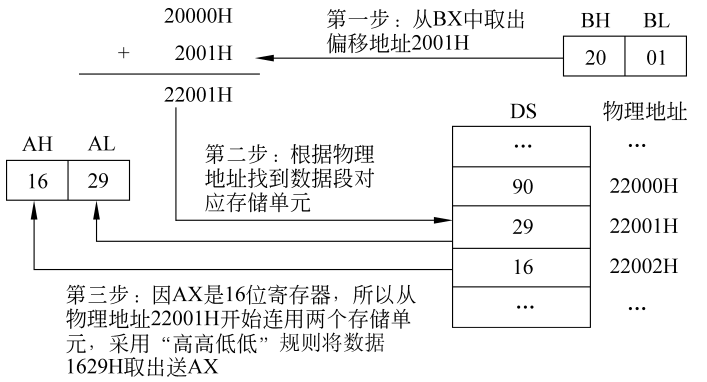

图 3.8 字操作、源操作数用 BX 作寄存器间接寻址方式

【例 3.7】 设 DS＝2000H，SS＝3000H，BP＝0001H，(22000H)＝90H，(22001H)＝29H，(22002H)＝16H，(30000H)＝69H，(30001H)＝09H，(30002H)＝03H，执行 MOV DX，[BP]后，DX 中数据为什么？

分析：MOV DX，[BP]指令两个操作数寻址方式及功能分析如下。

源操作数：[BP]为寄存器间接寻址方式，默认堆栈段 SS，EA＝0001H，PA＝SS×10H＋0001H＝3000H×10H＋0001H＝30001H。

目的操作数：DX 为寄存器寻址方式。

功能：该指令将 30001H 指向的字单元中数据传送至 DX。寄存器间接寻址方式的寻址规则分为三步，如图 3.9 所示。

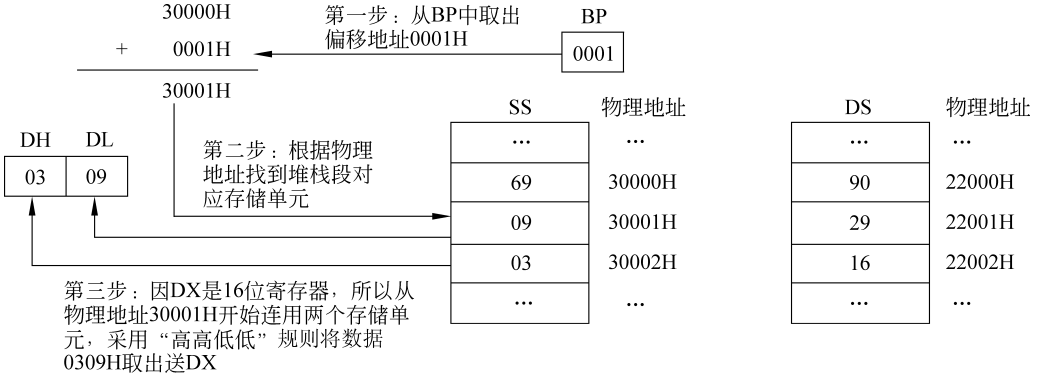

图 3.9 字操作、源操作数用 BP 作寄存器间接寻址方式

思考

1. 设 DS＝2000H，(22000H)＝90H，(22001H)＝29H，(22002H)＝16H，SI＝2002H，执行 MOV CH，[SI]后，CH 内容是什么？

2. 设 DS=3000H，SS=2000H，DI=0001H，(22000H)=90H，(20001H)=29H，(30001H)=09H，(30002H)=03H，执行 MOV DL，[DI]后，DL 内容是什么？

2. 目的操作数为寄存器间接寻址方式

【例 3.8】 设 ES=2000H，(23101H)=02H，(23102H)=82H，(23103H)=10H，SI=3103H，DL=05H，执行 MOV ES：[SI]，DL 后，目的操作数中数据为什么？

分析：MOV ES：[SI]，DL 指令两个操作数寻址方式及功能分析如下。

源操作数：DL 为寄存器寻址方式。

目的操作数：[SI]为寄存器间接寻址方式，EA=3103H，默认数据段 DS，因为采用段超越，所以使用附加段，PA=ES×10H+EA=2000H×10H+3103H=23103H。

功能：该指令将 DL 中数据传送至 23103H 指向的字节单元中。寄存器间接寻址方式的寻址规则分为三步，如图 3.10 所示。

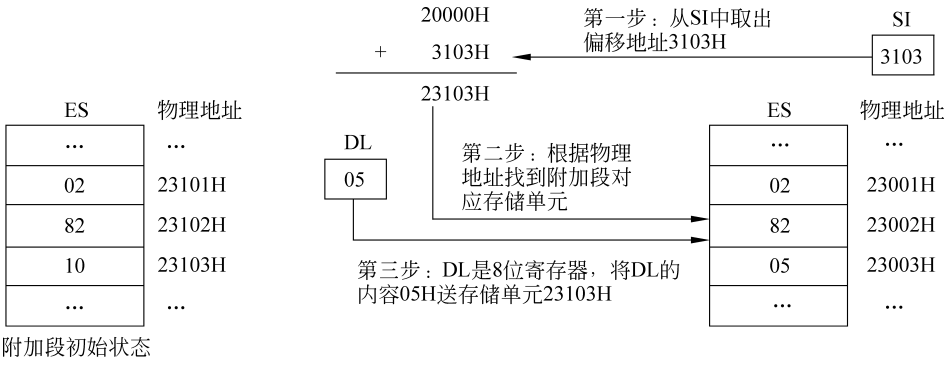

图 3.10 字节操作、目的操作数采用段超越用 SI 寄存器间接寻址方式

思考

设 ES=3000H，(23101H)=02H，(23102H)=82H，(23103H)=10H，BX=3101H，AX=0502H，DS=2000H，执行 MOV [BX]，AX 后，目的操作数数据为什么？

E7（寄存器相对寻址）

3.3.3 寄存器相对寻址方式

指令中操作数存放存储单元，存储单元偏移地址由寄存器（SI、DI、BX、BP）和一个 8 位或 16 位相对位移量之和决定，该寻址方式称为寄存器相对寻址方式。寄存器相对寻址方式，若不采用段超越，则规定如下。

使用 BX、SI 或 DI 寄存器，默认数据段 DS。物理地址计算方法如下。

PA=DS×10H+BX/SI/DI+8 位相对位移量/16 位相对位移量

- 使用 BP 寄存器，则默认堆栈段 SS。物理地址计算方法如下。

  PA＝SS×10H＋BP＋8 位相对位移量/16 位相对位移量

- 寄存器相对寻址书写方式有以下两种。

```
MOV AL,[BX]10H     或   MOV AL,[BX+10H]
MOV BX,01H[SI]     或   MOV BX,[01H+SI]
MOV [DI+123H],CH   或   MOV [DI]123H,CH
MOV [BP+03H],DX    或   MOV [BP]03H,DX
```

1. 源操作数为寄存器相对寻址方式

【例 3.9】 设 DS＝5000H，(52000H)＝90H，(52001H)＝78H，(52002H)＝56H，BX＝2000H，执行 MOV DX,[BX]01H 后，DX 中数据为什么？

分析：MOV DX,[BX]01H 指令两个操作数寻址方式及功能分析如下。

源操作数：[BX]01H 为寄存器相对寻址方式，默认数据段 DS，EA＝BX＋01H＝2000H＋01H＝2001H，PA＝DS×10H＋EA＝50000H＋2001H＝52001H。

目的操作数：DX 为寄存器寻址方式。

功能：该指令将 52001H 指向的字单元中数据传送至 DX。寄存器相对寻址方式的寻址规则分为三步，如图 3.11 所示。

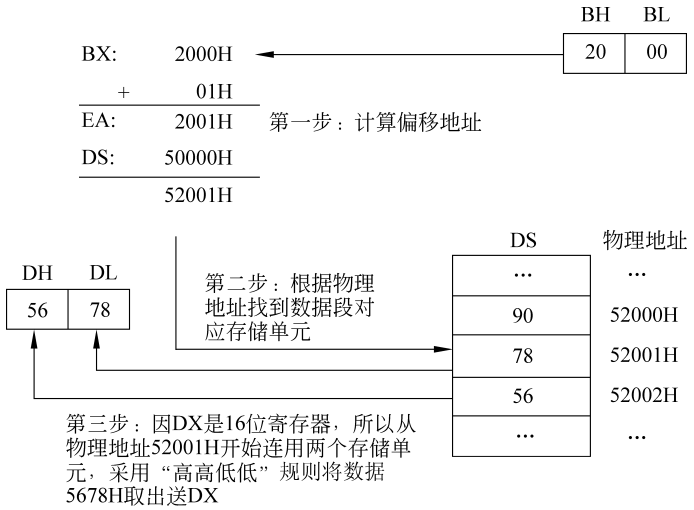

图 3.11 字操作、源操作数为寄存器相对寻址方式

思考

设 DS＝5000H，(52000H)＝90H，(52001H)＝78H，(52002H)＝56H，BX＝2000H，执行 MOV CL,[BX]02H 后，CL 的内容是什么？

2. 目的操作数为寄存器相对寻址方式

1）字操作

【例 3.10】 设 DS＝3000H，(31008H)＝90H，(31009H)＝78H，(3100AH)＝56H，DI＝1000H，BX＝9163H，执行 MOV [DI＋09H]，BX 后，目的操作数中数据为什么？

分析：MOV [DI＋09H]，BX 指令两个操作数寻址方式及功能分析如下。

源操作数：BX 为寄存器寻址方式。

目的操作数：[DI＋09H]为寄存器相对寻址方式，默认数据段 DS，EA＝DI＋09H＝1000H＋09H＝1009H，PA＝DS×10H＋EA＝30000H＋1009H＝31009H。

功能：该指令将 BX 中数据传送至 31009H 指向的字单元中。寄存器相对寻址方式的寻址规则分为三步，如图 3.12 所示。

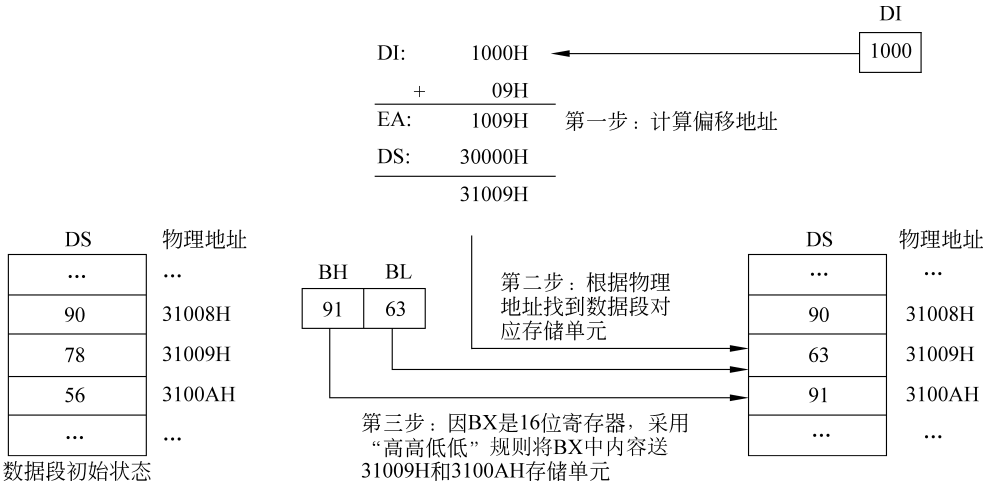

图 3.12　字操作、目的操作数为寄存器相对寻址方式

2）字节操作

【例 3.11】 设 ES＝2000H，(23010H)＝02H，(23011H)＝82H，(23012H)＝10H，BP＝3000H，DL＝05H，执行 MOV ES：[11H＋BP]，DL 后，目的操作数中数据为什么？

分析：MOV ES：[11H＋BP]，DL 指令两个操作数寻址方式及功能分析如下。

源操作数：DL 为寄存器寻址方式。

目的操作数：[11H＋BP]为寄存器相对寻址方式，默认堆栈段 SS，因为采用段超越，所以使用附加段 ES，EA＝11H＋BP＝10H＋3000H＝3011H，则 PA＝ES×10H＋EA＝2000H×10H＋3011H＝23011H。

功能：该指令将 DL 中数据传送至 23011H 指向的字节单元中。寄存器相对寻址方式的寻址规则分为三步，如图 3.13 所示。


E8（基址变址寻址）

3.3.4　基址变址寻址方式

指令中操作数存放在存储单元，存储单元偏移地址由基址寄存器（BX 或 BP）和变址

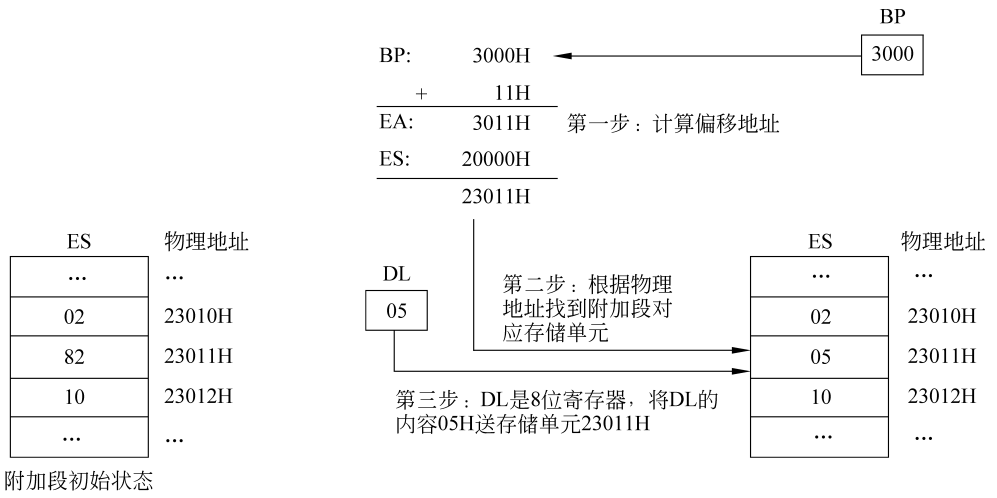

图 3.13 字节操作、目的操作数段超越采用寄存器相对寻址方式

寄存器(SI 或 DI)之和决定，这种寻址方式称为基址变址寻址方式。基址变址寻址方式，若不采用段超越，则规定如下。

- 使用 BX 和 SI 或 DI 寄存器，默认数据段 DS。物理地址计算方法如下。

$$PA=DS\times 10H+BX+SI/DI$$

- 使用 BP 和 SI 或 DI 寄存器，默认堆栈段 SS。物理地址计算方法如下。

$$PA=SS\times 10H+BP+SI/DI$$

注意：

- 基址变址寻址方式必须有一个基址寄存器和一个变址寄存器，不能同时为基址寄存器或变址寄存器。例如，以下指令不合法。

```
MOV AL,[BX+BP]
MOV [SI+DI],CX
```

- 基址变址寻址方式，如果不采用段超越，默认段以基址寄存器(BX、BP)为准。例如：MOV AL,[BP+SI]，BP 默认 SS 段，SI 默认 DS 段，以基址寄存器 BP 为准，该指令源操作数 PA=SS×10H+BP+SI。

- 基址变址寻址书写方式有以下两种。

```
MOV BX,[SI+BP]  或  MOV BX,[SI][BP]
MOV [DI+BX],CH  或  MOV [DI][BX],CH
```

1. 源操作数为基址变址寻址方式

【例 3.12】 设 DS=5000H，(52000H)=90H，(52001H)=78H，(52002H)=56H，BX=2000H，SI=0001H，执行 MOV DX,[BX][SI]后，DX 中数据为什么？

分析：MOV DX,[BX][SI]指令两个操作数寻址方式及功能分析如下。

源操作数：[BX][SI]为基址变址寻址方式，默认数据段 DS，EA＝BX＋SI＝2000H＋0001H＝2001H，PA＝DS×10H＋EA＝50000H＋2001H＝52001H。

目的操作数：DX 为寄存器寻址方式。

功能：该指令将 52001H 指向的字单元中数据传送至 DX。基址变址寻址方式的寻址规则分为三步，如图 3.14 所示。

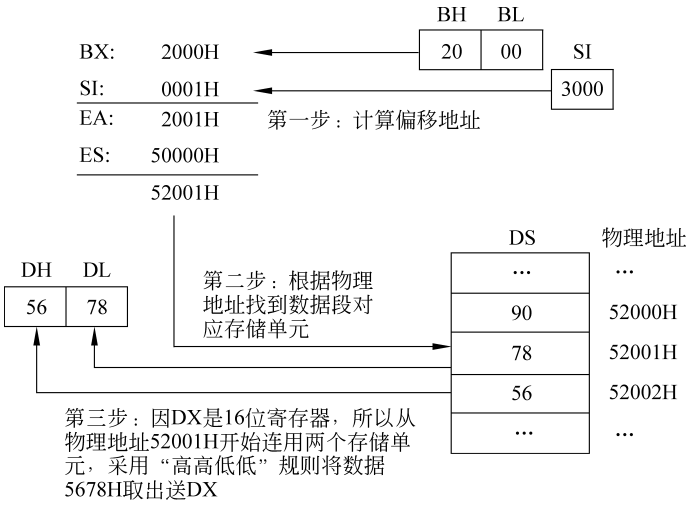

图 3.14　字操作、源操作数为基址加变址寻址方式

思考

设 DS＝5000H，(52000H)＝90H，(52001H)＝78H，(52002H)＝56H，BX＝2000H，DI＝0001H，执行 MOV CL，[BX][DI]后，CL 中内容为什么？

2. 目的操作数为基址变址寻址方式

【例 3.13】 设 SS＝3000H，(31008H)＝90H，(31009H)＝78H，(3100AH)＝56H，DI＝1000H，BP＝0009H，BX＝9086H，执行 MOV [DI＋BP]，BX 后，目的操作数中数据为什么？

分析：MOV [DI＋BP]，BX 指令中两个操作数寻址方式及功能分析如下。

源操作数：BX 为寄存器寻址方式。

目的操作数：[DI＋BP]为基址变址寻址方式，默认堆栈段 SS，EA＝DI＋BP＝1000H＋09H＝1009H，PA＝SS×10H＋EA＝30000H＋1009H＝31009H。

功能：该指令将 BX 中数据传送至 31009H 指向的字单元中。基址变址寻址方式的寻址规则分为三步，如图 3.15 所示。

思考

设 SS＝3000H，(31008H)＝90H，(31009H)＝78H，(3100AH)＝56H，SI＝1000H，BP＝0009H，BL＝90H，执行 MOV [SI＋BP]，BL 后，目的操作数中数据为什么？

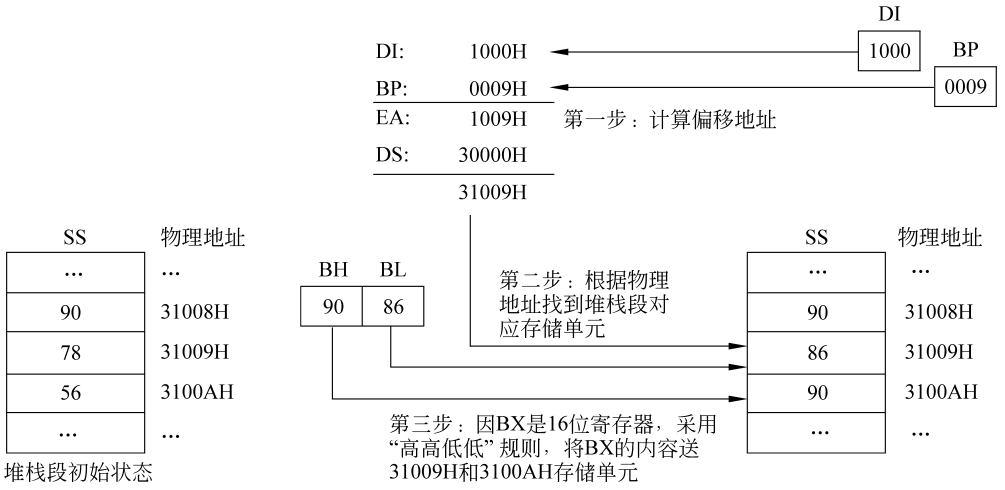

图 3.15 字操作、目的操作数为基址加变址寻址方式

3.3.5 相对基址变址寻址方式


E9（相对基址变址寻址）

指令中操作数存放在存储单元，存储单元偏移地址由基址寄存器（BX 或 BP）、变址寄存器（SI 或 DI）和一个 8 位或 16 位相对位移量之和决定，这种寻址方式称为相对基址变址寻址方式。相对基址变址寻址方式，若不采用段超越，则规定如下。

- 使用 BX 和 SI 或 DI 寄存器，默认数据段 DS。物理地址计算方法如下。

  PA＝DS×10H＋BX＋SI/DI＋8 位相对位移量/16 位相对位移量

- 使用 BP 和 SI 或 DI 寄存器，默认堆栈段 SS。物理地址计算方法如下。

  PA＝SS×10H＋BP＋SI/DI＋8 位相对位移量/16 位相对位移量

- 相对基址加变址寻址书写方式有以下两种。

```
MOV BX,[SI+BP+01H]   或   MOV BX,[SI][BP]01H
MOV 10H[DI][BX],CH   或   MOV [DI+BX+10H],CH
```

1. 源操作数为相对基址变址寻址方式

【例 3.14】 设 DS＝7000H，(72000H)＝90H，(72001H)＝26H，(72002H)＝77H，BX＝1000H，SI＝0001H，执行 MOV AX,[BX][SI]1000H 后，AX 中数据为什么？

分析：MOV AX,[BX][SI]1000H 指令中两个操作数寻址方式及功能分析如下。

源操作数：[BX][SI]1000H 为相对基址变址寻址方式，默认数据段 DS，EA＝BX＋SI＋1000H＝1000H＋0001H＋1000H＝2001H，PA＝DS×10H＋EA＝70000H＋2001H＝72001H。

目的操作数：AX 为寄存器寻址方式。

功能：该指令将 72001H 指向的字单元中数据传送至 AX。相对基址变址寻址方式的寻址规则分为三步，如图 3.16 所示。

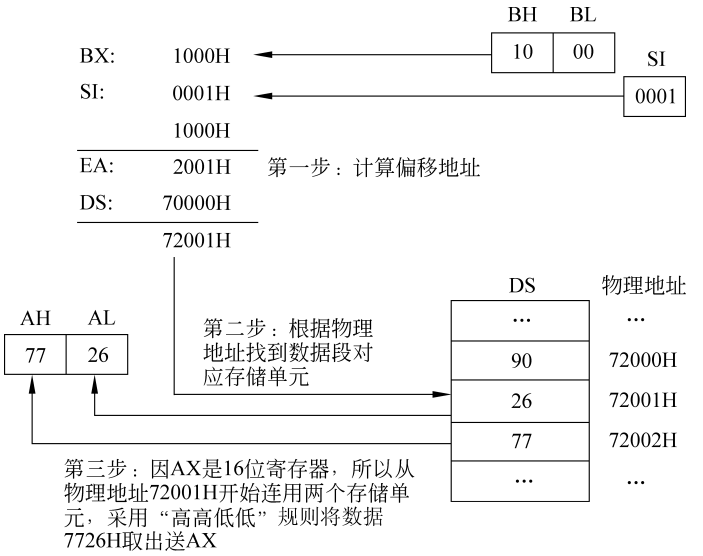

图 3.16　字操作、源操作数为相对基址变址寻址方式

思考

设 DS=7000H,(72000H)=90H,(72001H)=26H,(72002H)=77H,BX=1000H,DI=0001H,执行 MOV AH,[BX][DI]1000H 后,AH 的内容是什么?

2. 目的操作数为相对基址变址寻址方式

【例 3.15】 设 SS=2000H,(21008H)=90H,(21009H)=78H,(2100AH)=56H,DI=1000H,BP=0006H,BX=1209H,执行 MOV [0003H+DI+BP],BX 后,目的操作数中数据为什么?

分析：MOV [0003H+DI+BP],BX 指令中两个操作数寻址方式及功能分析如下。

源操作数：BX 为寄存器寻址方式。

目的操作数：[0003H+DI+BP]为相对基址变址寻址方式,默认堆栈段 SS,EA=DI+BP+0003H=1000H+0006H+0003H=1009H,PA=SS×10H+EA=20000H+1009H=21009H。

功能：该指令将 BX 中数据传送至 21009H 指向的字单元中。相对基址变址寻址方式的寻址规则分为三步,如图 3.17 所示。

思考

设 SS=2000H,(21008H)=90H,(21009H)=78H,DI=1000H,BP=0006H,BL=12H,执行 MOV [0003H+DI+BP],BL 后,目的操作数内容是什么?

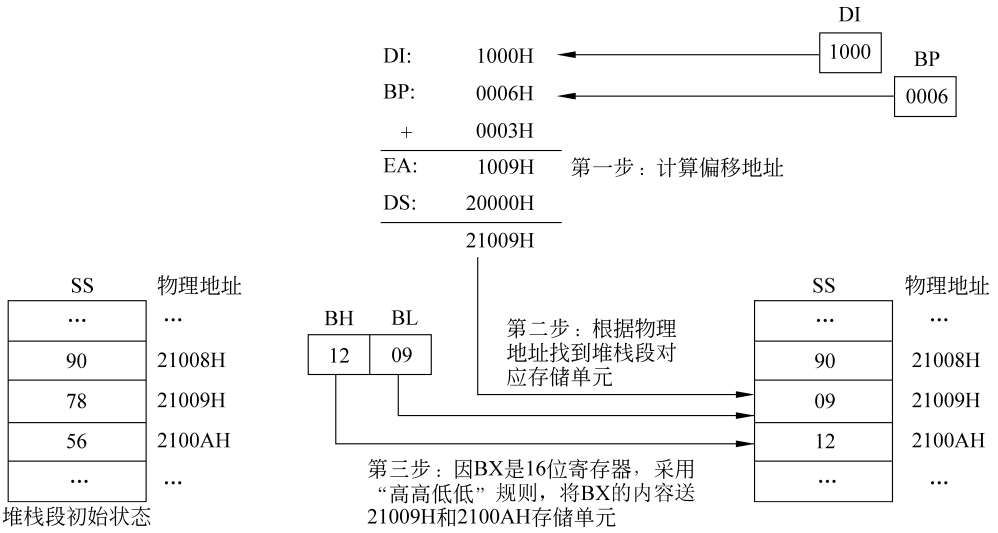

图 3.17 字操作、目的操作数为相对基址加变址寻址方式

3.4 常用 DEBUG 命令

大部分汇编语言源程序在调试阶段需用到调试程序进行调试。DEBUG 是专门为分析和调试汇编语言程序而设计的一种调试工具，它具有汇编、反汇编、显示和修改寄存器或内存等功能。通过单步执行、设置断点等方式为汇编语言程序设计者提供了非常有效的调试手段。

3.4.1 DEBUG 程序的启动


F1（debug 命令启动）

在 Windows 7(64 位)操作系统中使用 DEBUG.EXE(32 位)可执行程序，先安装 DOSBox 应用程序，再运行 DEBUG.EXE 程序。DOSBox 应用程序是一个 DOS 模拟程序，采用 SDL(Simple DirectMedia Layer) 库设计，SDL 是一套开放源代码跨平台多媒体开发库，使用 C 语言实现。所以 DOSBox 应用程序可以很方便地移植到其他平台。

下载 DOSBox 应用程序后，放到指定目录下。例如，本书中将下载的 DOSBox 存放到 E:\DOXBox 文件夹，如图 3.18 所示。双击 DOSBox.EXE 程序，单击 Next 按钮，最后单击 Install 按钮。安装好后，在桌面上会呈现如图 3.19 所示图标。双击该图标，启动 DOSBox 程序，出现两个界面，任意一个界面都不能关闭，如图 3.20 所示。

DEBUG.EXE 文件存放在 E:\MASM 文件夹，如图 3.21 所示。DOSBox 启动后，首先使用 MOUNT 挂载命令，将 E:\MASM 文件夹挂载到 DOSBox 虚拟盘符；挂载好后，需切换到 DOSBox 虚拟盘符，直接启动 DEBUG.EXE 文件，出现“_”提示符，说明 DEBUG 启动成功，如图 3.22 所示。

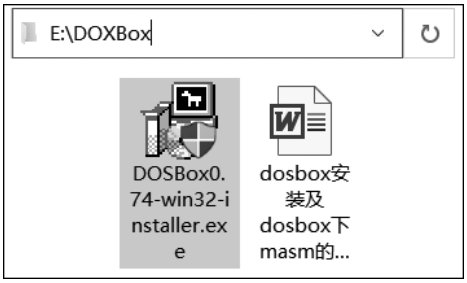

图 3.18　DOSBox 安装程序目录

图 3.19　DOSBox 图标

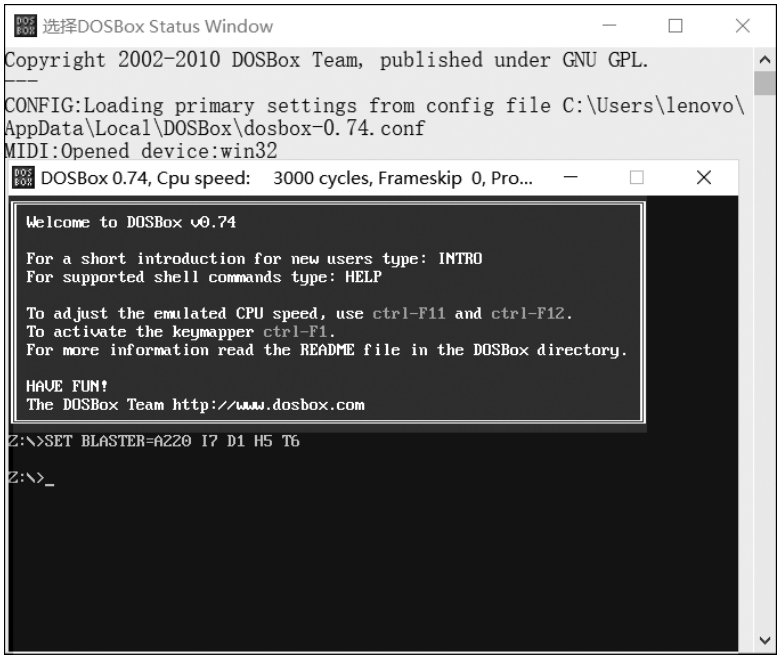

图 3.20　启动 DOSBox 界面

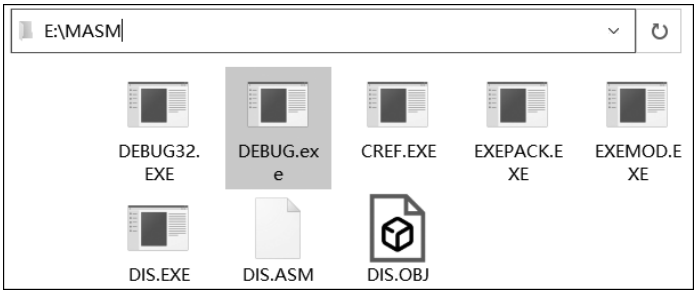

图 3.21　DEBUG.EXE 文件目录

图 3.22 DEBUG 程序启动方法

3.4.2 常用 DEBUG 调试命令及功能

常用 DEBUG 命令有 R、D、E、A、T、G、U 和 Q，均为一个字母，字母后可以跟一个或多个参数；字母和参数之间可用空格分隔，也可不用空格分隔；参数和参数之间必须用分隔符进行分隔，分隔符可以为空格、Tab 或逗号。

DEBUG 环境下注意：

- 所有数据都默认为十六进制数，而且不用字母“H”结尾。
- 不区分大小写。
- 如果使用地址时，格式为：

段寄存器名：偏移地址
段基址：偏移地址
偏移地址

- 如果有语法错误，显示错误行，并提示错误所在位置，如例 3.16 所示。

【例 3.16】

```
-UQ   1200;错误行
  ^ERROR
```

分析：反汇编命令为 U，所以该行有错误，“^”对应错误位置为 Q。

【例 3.17】

```
-U   198W
       ^ERROR
```

反汇编命令 U，后面加十六进制 16 位偏移地址，198W 中 W 是错误的数码，所以该行有错误，“^”对应错误位置为 W。


F2(R 命令)

1. R 命令——显示/修改寄存器内容

1）格式 1：

```
R
```

功能：显示 CPU 中所有寄存器。

【例 3.18】 显示所有寄存器。

```
-R
AX=1234  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100  NV UP EI PL NZ NA PO NC
073F:0100  0000  ADD [BX+SI],AL                          DS:0000=CD
```

说明：

AX、BX、CX 和 DX 可以当作 16 位寄存器使用，也可以当作 8 位寄存器使用，以 AX 寄存器为例说明：

AX 当作 16 位寄存器使用，AX 中数据为 1234。

AX 当作 8 位寄存器使用，分为高 8 位 AH=12 和低 8 位 AL=34，如图 3.23 所示。

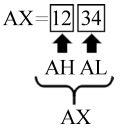

图 3.23 AX 寄存器

NV UP EI PL NZ NA PO NC 为标志状态信息。

073F:0100 0000 ADD [BX+SI],AL 表示下一条要执行指令为 ADD，目的操作数为[BX+SI]，偏移地址 BX+SI=0000，源操作数为 AL。DS：0000=CD，表示数据段偏移地址为 0000 存储单元中存放的数据为 CD。

2）格式 2：

```
R  16位寄存器名/段寄存器名
```

功能：显示或修改 CPU 中指定寄存器。

【例 3.19】 显示指定寄存器。

```
-R  AX
1234
:
```

说明：

冒号后可有两种操作：

若不修改 AX 寄存器中数据，则直接按 Enter 键。

若修改 AX 寄存器中数据，冒号后输入 1～4 位十六进制数，再按 Enter 键，AX 中数据被修改，再次通过 R 命令查询被修改的数据。例如：

```
-R  AX
1234
:9876
-R  AX     ;查看 AX 修改后的数据
```

```
9876
:
```

不能出现高 8 位寄存器名(AH、BH、CH 和 DH)或低 8 位寄存器名(AL、BL、CL 和 DL),如图 3.24 所示。

```
-R AL
br Error
-R CH
br Error
-R DL
br Error
-R CL
br Error
-R DH
br Error
```

图 3.24 显示 8 位寄存器错误信息

2. RF 命令——显示/修改标志寄存器内容

8086CPU 标志寄存器有 9 个标志位,跟踪标志位 TF 不能通过指令修改,所以 DEBUG 环境中可显示和修改 8 个标志位。8 个标志位顺序为 OF(溢出标志位)、DF(方向标志位)、IF(中断允许标志位)、SF(符号标志位)、ZF(零标志位)、AF(辅助进位标志位)、PF(奇偶标志位)、CF(进位标志位)。各个标志位含义如表 3.1 所示。

表 3.1 标志位含义

标 志 位	置 位	复 位
OF(溢出标志位)	OF=1	OF=0
	OV	NV
DF(方向标志位)	DF=1	DF=0
	DN	UP
IF(中断允许标志位)	IF=1	IF=0
	EI	DI
SF(符号标志位)	SF=1	SF=0
	NG	PL
ZF(零标志位)	ZF=1	ZF=0
	ZR	NZ
AF(辅助进位标志位)	AF=1	AF=0
	AC	NA
PF(奇偶标志位)	PF=1	PF=0
	PE	PO
CF(进位标志位)	CF=1	CF=0
	CY	NC

格式:

```
RF
```

功能:显示或修改标志寄存器。