第3章 微处理器的指令系统

【学习目标】

8086/8088 CPU 的指令系统是 Intel 80x86 系列 CPU 共同的基础，其后续高型号微处理器的指令系统都是在此基础上新增了一些指令逐步扩充形成的。同时，它也是目前应用范围最广的一种指令系统。因此，本章将重点讨论 8086/8088 CPU 的指令系统。最后，将指出 Intel 80x86 系列 CPU 指令集的一些问题，并介绍几种扩展指令集的实用知识。

通过本章对 8086/8088 CPU 寻址方式和指令系统的学习，应该掌握汇编语言程序设计所需要的汇编语言和编写程序段的基础知识。

【学习要求】

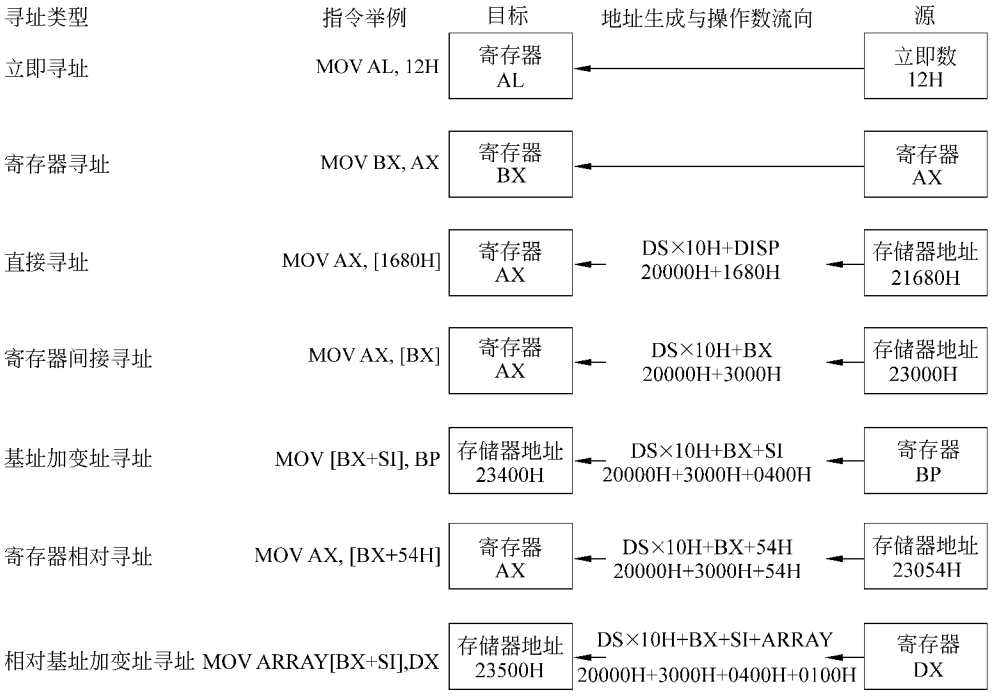
- 在理解与掌握各种寻址方式的基础上，着重掌握存储器寻址的各种寻址方式。
- 应熟练掌握 4 类数据传送指令。难点是 XLAT、IN、OUT 指令。
- 学习算术运算类指令中的难点是带符号乘、除指令与十进制指令。
- 学习逻辑运算和移位循环类指令时，要着重理解 CL 的设置和进位位的处理。
- 学习串操作类指令时，着重理解重复前缀的使用。
- 学习程序控制类指令时，着重理解条件转移的条件及测试条件。
- 理解指令集的发展趋势，了解几种流行的指令集：MMX、SSE、SSE2、SSE3 和 3DNow!。

3.1 8086/8088 的寻址方式

指令格式包括操作码和操作数(或地址)两部分，根据操作码所指定的功能去寻找操作数所在地址的方式就是寻址方式。要熟悉指令的操作首先要了解寻址方式。8086/8088 的寻址方式分为两种不同的类型：数据寻址方式和程序存储器寻址方式。前者是寻址操作数地址，后者是寻址程序地址(在代码段中)。

3.1.1 数据寻址方式

数据寻址方式有多种，图 3-1 给出了各种数据寻址方式的类型、指令举例以及存储器地址生成方法与数据流向，所有操作数的流向都是由源到目标，即它们在指令汇编语言格式的操作数区域中都是规定由右到左。源和目标可以是寄存器或存储器，但不能同时为存储器（除个别串操作指令 MOVS 外）。下面将分别对各种寻址方式给予更详细的说明。



注：BX=3000H, SI=0400H, ARRAY=0100H, DS=2000H

图 3-1　8086/8088 数据寻址方式

1. 立即寻址

立即寻址是将立即数传送到目标寄存器或存储器中。操作数就在指令中，当执行指令时，CPU 直接从紧跟着指令代码的后续地址单元中（经队列缓冲器）取得该立即数，而不必执行总线周期。立即数可以是 8 位，也可以是 16 位；并规定只能是整数类型的源操作数。这种寻址主要用来给寄存器赋初值，指令执行速度快。表 3-1 列出了各种立即数寻址的 MOV 指令。

2. 寄存器寻址

寄存器寻址是最通用的数据寻址方式。其操作数就放在 CPU 的寄存器中，而寄存器名在指令中指出。对 16 位操作数来说，寄存器可以为 8 个 16 位通用寄存器，而对 8 位操作数来说，寄存器只能为AH、AL、BH、BL、CH、CL、DH、DL。在一条指令中，源操作数或/和

表 3-1　使用立即寻址的 MOV 指令示例

汇 编 语 句	长度/位	操　　作
MOV AH,4CH	8	把 4CH 传送到 AH 中
MOV AX,1234H	16	把 1234H 传送到 AX 中
MOV DI,0	16	把 0000H 传送到 DI 中
MOV CL,100	8	把 100(64H)传送到 CL
MOV AI,'A'	8	把 ASCII 码 A(41H)传送到 AL 中
MOV AX,'AB'	16	把 ASCII 码 BA*(4241H)传送到 AX 中
MOV CL,10101101B	8	把二进制数 10101101 传送到 CL 中
MOV WORD PTR [SI], 6180H	16	把立即数 6180H 传送到数据段由 SI 和 SI+1 所指的两存储单元中

注：*'AB'在内存中的数据结构为 ASCII 码 BA。

目的操作数都可以采用寄存器寻址方式。这种寻址的指令长度短，操作数就在 CPU 内部进行，不需要使用总线周期，所以执行速度快。注意，使用时源与目标操作数应有相同的数据类型长度。

表 3-2 列出了各种寄存器寻址的 MOV 指令。注意，代码段寄存器不能用 MOV 指令来改变，因为若只改变 CS 而 IP 为未知数，则下一条指令的地址将是不确定的，这可能引起系统运行的紊乱。

表 3-2　使用寄存器寻址的 MOV 指令示例

汇 编 语 句	长度/位	操　　作
MOV AL,BL	8	把 BL 复制到 AL 中
MOV BH,BL	8	把 BL 复制到 BH 中
MOV CX,AX	16	把 AX 复制到 CX 中
MOV SP,BP	16	把 BP 复制到 SP 中
MOV DI,SI	16	把 SI 复制到 DI 中
MOV AX,ES	16	把 ES 复制到 AX 中

下面将讨论属于存储器寻址的各种寻找方式。指令系统中采用的复杂的寻址方式主要是针对存储器操作数而言的。当 CPU 寻找存储器操作数时，必须先经总线接口单元 BIU 的总线控制逻辑电路进行存取。当执行单元 EU 需要读写位于存储器的操作数时，应根据指令给出的寻址方式，由 EU 先计算出操作数地址的偏移量(即有效地址 EA)，并将它送给 BIU，同时请求 BIU 执行一个总线周期，BIU 将某个段寄存器的内容左移 4 位，加上由 EU 送来的偏移量形成一个 20 位的物理地址，然后执行总线周期，读写指令所需的操作数。8086/8088 CPU 所寻址的操作数地址的有效地址 EA，是一个无符号的 16 位地址码，表示操作数所在段的首地址与操作数地址之间的字节距离。所以，它实际上是一个相对地址。EA 的值由汇编程序根据指令所采用的寻址方式自动计算得出。计算 EA 的通式为：

EA＝基址值(BX 或 BP)＋变址值(SI 或 DI)＋位移量 DISP

3. 直接数据寻址

直接数据寻址有两种基本形式：直接寻址和位移寻址。

(1) 直接寻址

直接寻址简单、直观，其含义是指令中以位移量方式直接给出存储器操作数的偏移地址，即有效地址 EA＝DISP。这种寻址方式的指令执行速度快，用于存储单元与 AL、AX 之间的 MOV 指令。

(2) 位移寻址

位移寻址也以位移量方式直接给出存储器操作数的偏移地址，但适合于几乎所有将数据从存储单元传送到寄存器的指令。

以上两种方式都是把位移量加到默认的数据段地址或其他段地址上形成的。表 3-3 列出了使用 AX、AL 的直接寻址指令示例；表 3-4 列出了使用位移量的直接数据寻址的示例。

表 3-3　使用 AX，AL 的直接寻址指令示例

汇 编 语 句	长度/位	操　　作
MOV AX,[1680H]*	16	把数据段存储器地址 1680H 和 1681H 两单元的字内容复制到 AX 中
MOV AX,NUMBER	16	把数据段存储器地址 NUMBER 中的字内容复制到 AX 中
MOV TWO,AL	8	把 AL 的字节内容复制到数据段存储单元 TWO 中
MOV ES:[3000H],AX	16	把 AX 的字内容复制到附加数据段存储单元 3000H 中
MOV AX,DATA	16	把数据段存储单元 DATA 的字内容复制到 AX 中

注：＊汇编语言中很少采用绝对偏移地址，通常采用符号地址。

表 3-4　使用位移量的直接数据寻址指令示例

汇 编 语 句	长度/位	操　　作
MOV CL,COW	8	把数据段存储单元 COW 的内容(字节)复制到 CL 中
MOV ES,NUMBER	16	把数据段存储器地址 NUMBER 中的内容(字)复制到 ES 中
MOV CX,DATA2	16	把数据段存储单元 DATA2 中的内容(字)复制到 CX 中
MOV DATA3,BP	16	把基址指针寄存器 BP 的内容复制到数据段存储单元 DATA3 中
MOV DI,SUM	16	把数据段存储单元 SUM 的字内容复制到 DI 中
MOV NUMBER,SP	16	把 SP 的内容复制到数据段存储单元 NUMBER 中

位移寻址与直接寻址的操作相同，只是它的指令为 4B 长而不是 3B 长。

【例 3-1】 MOV CL,[2000H]指令与 MOV AL,[2000H]指令的操作相同，但 MOV CL,[2000H]指令为 4B 长，而 MOV AL,[2000H]指令为 3B 长。

4. 寄存器间接寻址

寄存器间接寻址的操作数一定是在存储器中，而存储单元的有效地址 EA 则由寄存器保存，这些寄存器是基址寄存器 BX、基址指针寄存器 BP、变址寄存器 SI 和 DI 之一或它们的某种组合。书写指令时，这些寄存器带有方括号[]。

【例 3-2】 设 BX＝3000H，DS＝2000H，当执行 MOV AX,[BX]指令后，则数据段存储

单元为23000H处的字内容将被复制AX中，即23000H的内容送到AL，23001H的内容送到AH。指令中的方括号[]在汇编语言中表示间接寻址。表3-5给出了寄存器间接寻址的指令示例。

表3-5　寄存器间接寻址的指令示例

汇编语句	长度/位	操　作
MOV AL,[BX]	8	把数据段中以BX作为有效地址的存储单元的内容(字节)复制到AL中
MOV [SI],BL	8	把寄存器BL的内容复制到数据段以SI作为有效地址的存储单元
MOV CX,[DX]	16	把数据段由DX寻址的存储单元的内容(字)复制到CX中
MOV [BP],CL*	8	把寄存器CL的内容复制到堆栈段以BP作为有效地址的存储单元中
MOV [SI],[BX]	—	除数据串操作指令外，不允许由存储器到存储器的传送

注：*系统把由BP寻址的数据默认为在堆栈段中，其他间接寻址方式均默认为数据段。

当使用BX、DI和SI寻址存储器时，寄存器间接寻址或任何其他寻址方式都默认使用数据段，而使用基址指针寄存器BP寻址存储器时，则默认使用堆栈段。

在使用寄存器间接寻址时，要注意在某些情况下，要求用指定的类型运算伪指令BYTE PTR、WORD PTR或DWORD PTR来规定传送数据的长度。

【例3-3】 MOV AL,[SI]指令的书写格式是对的，因为汇编程序能够清楚地根据AL来判明[SI]是指定存储器数据为字节传送类型。

【例3-4】 MOV [SI],6AH指令的书写格式是模糊的。因为，汇编程序不能根据立即数6AH确定[SI]存储单元的数据类型的长度。如果将此指令书写成MOV BYTE PYR [SI],6AH，则汇编程序就能清楚地判明SI所寻址的存储单元为字节类型。

5. 基址加变址寻址

基址加变址寻址类似于间接寻址，它也是间接地寻址存储器数据。其操作数的有效地址EA是一个基址寄存器(BX或BP)的内容与一个变址寄存器(SI或DI)的内容之和。

【例3-5】 MOV [BX+SI],CL指令是将寄存器CL中的字节内容复制到数据段中由BX加SI寻址的存储单元中。

在使用基址加变址寻址时，通常用基址寄存器保持存储器数组的起始地址，而变址寄存器保持数组元素的相对位置。如果是用BP寄存器寻址堆栈段存储器数组，则由BP寄存器和变址寄存器两者生成有效地址。

【例3-6】 当执行指令MOV DX,[BP+SI]时，若BP=2000H，SI=0300H，SS=1000H，则指令执行后，将把堆栈段中12300H单元的字数据传送到DX寄存器。表3-6给出了基址加变址寻址的指令示例。

6. 寄存器相对寻址

寄存器相对寻址是带有位移量DISP的基址或变址寄存器(BX、BP或DI、SI)寻址。

【例3-7】 在MOV AX,[SI+4000H]指令中，假设SI=0500H，DS=2000H，则指令执行时，微处理器按段加偏移寻址机制得到的有效地址为EA=SI+4000H=4500H，再加上

表 3-6　基址加变址寻址的指令示例

汇 编 语 句	长度/位	操　　作
MOV CL,[BX+SI]	8	把以 BX+SI 作为有效地址的数据段存储单元的内容(字节)复制到 CL
MOV CX,[BP+DI]	16	把以 BP+DI 作为有效地址的堆栈段存储单元内的内容(字)复制到 CX
MOV [BX+DI],SP	16	把 SP 的内容(字)存入以 BX+DI 作为有效地址的数据段存储单元
MOV [BP+SI],CH	8	把寄存器 CH 的内容(字节)存入以 BP+SI 作为有效地址的堆栈段存储单元
MOV [AX+BX],CX	16	把 CX 中的内容(字)存入以 AX+BX 作为有效地址的数据段存储单元

DS×10H=20000H,生成所寻址的存储器物理地址为 24500H,于是,指令执行后将把数据段存储单元 24500H 中的字内容送到 AX。表 3-7 给出了寄存器相对寻址的指令示例。

表 3-7　寄存器相对寻址的指令示例

汇 编 语 句	长度/位	操　　作
MOV CL,[SI+200H]	8	把以 SI+200H 作为有效地址的数据段存储单元的字节内容装入 CL
MOV ARRAY[DI],BL	8	把 BL 中的字节内容存入以 ARRAY+DI 作为有效地址的数据段存储单元
MOV LIST[DI+3],AX	16	把 AX 的字内容存入以 LIST+DI+3 之和作为有效地址的数据段存储单元
MOV AX,ARRAY[BX]	16	把数据段中以 ARRAY+BX 作为有效地址的字内容装入 AX
MOV SI,[AL+12H]	16	把以 AL+12H 作为有效地址的数据段存储单元的字内容装入 SI

7. 相对基址加变址寻址

相对基址加变址寻址是用基址、变址与位移量 3 个分量之和形成有效地址的寻址方式。

【例 3-8】 在 MOV AX,[BX+DI+200H]指令中,设 BX=0100H,DI=0300H,DS=4000H。当指令执行时,先计算出有效地址为 EA=BX+DI+200H=0600H,指令运行后,将把数据段存储单元 40600H 中的字内容装入 AX。表 3-8 给出了相对基址加变址寻址的指令示例。

相对基址加变址寻址方式一般很少使用,通常用来寻址存储器的二维数组数据。

【例 3-9】 存储器中有一个文件 FILE 包含 A、B、C、D 4 个记录,每个记录又包含 10 个元素,如果要求将其中存储在单元 RECA 中的记录 A 的元素 0 复制到记录 D 的元素 4,这时,可以用位移量寻址文件,用基址寄存器 BX 寻址记录,而用变址寄存器 DI 寻址记录中的元素。程序段如下。

```
MOV BX,OFFSET RECA          ;寻址记录 A 的存储单元 RECA
MOV DI,0                    ;寻址单元 0
```

```
MOV AL,FILE[BX+DI]          ;取出记录 A 的元素 0
MOV BX,OFFSET RECD          ;寻址记录 D 的存储单元 RECD
MOV DI,4                    ;寻址单元 4
MOV FILE[BX+DI],AL          ;复制到记录 D 的元素 4 中
```

表 3-8　相对基址加变址寻址的指令示例

汇 编 语 句	长度/位	操　　作
MOV BL,[BX+SI+100H]	8	把以 BX+SI+100H 作为有效地址的数据段存储单元的字节内容装入 BL
MOV AX,ARRAY[BX+DI]	16	把以 ARRAY+BX+DI 之和作为有效地址的数据段存储单元的字内容装入 AX
MOV LIST[BP+DI],BX	16	把 BX 的字内容存入以 LIST+BP+DI 之和作为有效地址的堆栈段存储单元
MOV AL,LIST[BX+DI]	8	把以 LIST+BX+DI 之和作为有效地址的数据段存储单元的字节内容装入 AL
MOV FILE[BP+DI+2],DL	8	把 DL 存入以 BP+DI+2 之和作为有效地址的堆栈段存储单元

3.1.2　程序存储器寻址方式

程序存储器寻址方式即转移类指令(转移指令 JMP 和调用指令 CALL)的寻址方式。这种寻址方式最终是要确定一条指令的地址。

在 8086/8088 系统中,由于存储器采用分段结构,所以转移类指令有段内转移和段间转移之分。所有的条件转移指令只允许实现段内转移,而且是段内短转移,即只允许转移的地址范围在－128～＋127 字节内,由指令中直接给出 8 位地址位移量。对于无条件转移和调用指令又可分为段内短转移、段内直接转移、段内间接转移、段间直接转移和段间间接转移 5 种寻址方式。

3.1.3　堆栈存储器寻址方式

表 3-9 列出了可以使用的一些 PUSH 和 POP 指令的示例。

表 3-9　PUSH 和 POP 指令的示例

汇 编 语 句	操　　作
PUSHF	把标志寄存器 FLAGS 的内容复制到堆栈中
POPF	把从堆栈弹出的一个字装入标志寄存器 FLAGS
PUSH DS	把 DS 的内容复制到堆栈中
PUSH 12ABH	把 12ABH 压入堆栈
POP CS	非法操作
PUSH WORD PTR[BX]	把数据段中由 BX 寻址的存储单元内的字复制到堆栈中
PUSHA	把通用寄存器 AX、CX、DX、BX、SP、BP、DI、SI 的内容复制到堆栈中
POPA	从堆栈中弹出数据并顺序装入 SI、DI、BP、SP、BX、DX、CX、AX 中

3.1.4 其他寻址方式

1. 串操作指令寻址方式

数据串(或称字符串)指令不能使用正常的存储器寻址方式来存取数据串指令中使用的操作数。执行数据串指令时,源串操作数第1个字节或字的有效地址应存放在源变址寄存器SI中(不允许修改),目标串操作数第1个字节或字的有效地址应存放在目标变址寄存器DI中(不允许修改)。在重复串操作时,8086/8088能自动修改SI和DI的内容,以使它们能指向后面的字节或字。因指令中不必给出SI或DI的编码,故串操作指令采用的是隐含寻址方式。

2. I/O端口寻址方式

在8086/8088指令系统中,输入输出指令对I/O端口的寻址可采用直接或间接两种方式。

(1) 直接端口寻址

这种寻址方式端口地址以8位立即数方式在指令中直接给出。例如,IN AL,n指令是将端口号为8位立即数n的端口地址中的字节操作数输入到AL,它所寻址的端口号只能在0～255范围内。

(2) 间接端口寻址

这种寻址方式类似于寄存器间接寻址,16位的I/O端口地址在DX寄存器中,即通过DX间接寻址,故可寻址的端口号为0～65 535。例如,OUT DX,AL指令是将AL的字节内容输出到由DX指出的端口中去。

下面将详细讨论8086/8088的指令系统。8086/8088的指令按功能可分为6类:数据传送、算术运算、逻辑运算、串操作、程序控制和CPU控制。

3.2 数据传送类指令

数据传送类指令可完成寄存器与寄存器之间、寄存器与存储器之间以及寄存器与I/O端口之间的字节或字传送,除了SAHF和POPF指令对标志位有影响外,这类指令所具有的共同特点是不影响标志寄存器的内容。

3.2.1 通用数据传送指令

通用数据传送指令包括基本的传送指令MOV,堆栈操作指令PUSH和POP,数据交换指令XCHG与字节翻译指令XLAT。

1. 基本的传送指令

MOV d,s;d←s

指令功能：将由源 s 指定的源操作数送到目标 d。

前面已介绍过 MOV 指令的使用例子。注意：源操作数可以是 8/16 位寄存器、存储器中的某个字节/字或者是 8/16 位立即数；目标操作数不允许为立即数，其他同源操作数。而且两者不能同时为存储器操作数。

MOV 指令可实现的数据传送类型可归纳为以下 7 种。

(1) MOV mem/reg1,mem/reg2

由 mem/reg2 所指定的存储单元或寄存器中的 8 位数据或 16 位数据传送到由 mem/reg1 所指定的存储单元或寄存器中，但不允许从存储器传送到存储器。这种双操作数指令中，必须有一个操作数是寄存器。例如，表 3-2～表 3-8 中所列的各种指令示例。

(2) MOV mem/reg,data

将 8 位或 16 位立即数 data 传送到由 mem/reg 所指定的存储单元或寄存器中。例如，表 3-1 所列的各种指令示例。

(3) MOV reg,data

将 8 位或 16 位立即数 data 传送到由 reg 所指定的寄存器中。

(4) MOV ac,mem

将存储单元中的 8 位或 16 位数据传送到累加器 ac 中。

(5) MOV mem,ac

将累加器 AL(8 位)或 AX(16 位)中的数据传送到由 mem 所指定的存储单元中。

(6) MOV mem/reg,segreg

将由 segreg 所指定的段寄存器(CS、DS、SS 或 ES)的内容传送到由 mem/reg 所指定的存储单元或寄存器中。

(7) MOV segreg,mem/reg

允许将由 mem/reg 指定的存储单元或寄存器中的 16 位数据传送到由 segreg 所指定的段寄存器(但代码段寄存器 CS 除外)中。

【例 3-10】 MOV DS,AX 指令是对的；MOV CS,AX 指令是错的。

注意，MOV 指令不能直接实现从存储器到存储器之间的数据传送，但可以通过寄存器作为中转站来完成这种传送。

【例 3-11】 MOV [SI],[BX]指令是错的；而用以下两条指令是对的。

```
MOV AX,[BX]
MOV [SI],AX
```

【例 3-12】 要将数据段存储单元 ARRAY1 中的 8 位数据传送到存储单元 ARRAY2 中，用 MOV ARRAY2,ARRAY1 指令是错的；而用以下两条指令则可以完成该操作。

```
MOV AL,ARRAY1
MOV ARRAY2,AL
```

2. 堆栈操作指令

(1) PUSH s

字压入堆栈指令，允许将源操作数 s(16 位)压入堆栈。

(2) POP d

字弹出堆栈指令，允许将堆栈中当前栈顶两相邻单元的数据字弹出到 d。

PUSH 和 POP 是两条成对使用的进栈与出栈指令，其中，s 和 d 可以是 16 位寄存器或存储器两相邻单元，以保证堆栈按字操作。

【例 3-13】 设当前 CS＝1000H，IP＝0030H，SS＝2000H，SP＝0040H，BX＝2340H，则 PUSH BX 指令的操作过程如图 3-2 所示。

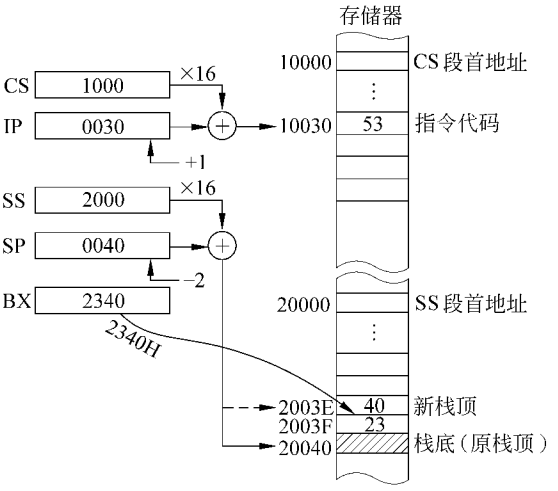


图 3-2　PUSH BX 指令的操作过程

该进栈指令执行时，堆栈指针被修改为 SP－2→SP，使之指向新栈顶 2003EH，同时将 BX 中的数据字 2340H 压入栈内 2003FH 与 2003EH 两单元中。

【例 3-14】 设当前 CS＝1000H，IP＝0020H，SS＝1600H，SP＝004CH，则 POP CX 指令执行时，将当前栈顶两相邻单元 1604CH 与 1604DH 中的数据字弹出并传送到 CX 中，同时修改堆栈指针，SP＋2→SP，使之指向新栈顶 1604EH。

PUSH 和 POP 两条指令可用来保存并恢复现场数据。由于堆栈中的内容是按 LIFO（后进先出）的次序进行传送的，因此，保存内容和恢复内容时，需按照对称的次序执行一系列压入指令和弹出指令。

【例 3-15】 若在一段子程序开头需要这样保存寄存器的内容：

```
PUSH AX
PUSH BX
PUSH DI
PUSH SI
```

则由子程序返回前，应该如下一一对应地恢复寄存器的内容：

```
POP SI
POP DI
POP BX
POP AX
```

使用堆栈指令时应该注意：