第3章 寻址方式

3.1 数据寻址方式

3.2 指令寻址方式

3.3 跨段的有关问题

3.4 实模式与保护模式

3.5 综合举例

什么是寻址方式？

- ✓通常，执行一条带有操作数的指令需要指明两个问题，一是进行什么操作？二是操作数在那里？
- ✓我们将寻找数据和指令存放地址的方式称为寻扯方式。
- ✓操作数在计算机运行过程中可能存储的物理位置有指令、CPU、存储器、接口等；不同的存储位置所采用的寻址方式不同。

3.1 数据寻址方式

✓数据寻址方式即寻找操作数地址的方式，在8086/80286中只能使用16位寻址，而80386及其后继机型则既可用16位寻址，也可用32位寻址，

✓无论多少位寻址实质都是寻找操作数的物理地址。

✓物理地址是由段地址和偏移地址两部分组成，段地址存放在相应的段寄存器中，偏移地址存储在相应的地址寄存器中，偏移地址又称有效地址。

地址寄存器：IP、SP、BX和BP等

有效地址(偏移地址)的组成

指令、CPU?

如果一个操作数存放在**存储器**中，那么有效地址可以由以下四部分组成。

✓1. 位移量(displacement)是存放在指令中的一个8位、16位和32位的数，但它不是立即数，而是一个地址。

✓2. 基址(base)是存放在基址寄存器中的内容。它是有效地址中的基址部分，通常用来指向数据段中数组或字符串的首地址。

✓3. 变址(index)是存放在变址寄存器中的内容。它通常用来访问数组中的某个元素或字符串中的某个字符。

有效地址(偏移地址)的组成

✓4. 比例因子是386及其后继机型中新增加寻址方式中的一个术语，其值可为1，2，4或8。在寻址中，可用变址寄存器的内容乘以比例因子来取得变址值。这类寻址方式对访问元素长度为2，4，8字节的数组特别有用。

？

有效地址的计算方法可以用下式表示：

✓EA = 基址 + (变址×比例因子) + 位移量（X）

✓这四个成分中，除比例因子是固定值外，其他三个成分都可正可负，以保证指针移动的灵活性。

3.1.1 16位寻址

- ✓16位寻址方式是指操作数的偏移地址由16位二进制数组成，段地址保存在DS、ES、SS、CS中，20位物理地址是由段地址左移4位二进制加上操作数的偏移地址形成，最大寻址空间为1MB。

1. 立即寻址

在该寻址方式中，操作数直接存放在指令内，且紧跟在指令操作码之后，指令码和操作数都存放在代码段中。

- ✓汇编格式：n
- ✓操作数位置：指令
- ✓功能：指令码下一单元的内容即为操作数n，操作数存放在指令中。

n立即数，第一章例子

立即寻址例题

例3.1 MOV AX，2000H

- ✓说明：MOV是双操作数指令，功能是实现数据传送。该指令的目的操作数是AX，源操作数是2000H，反汇编代码为B80020，其中指令长度为3个字节，立即数2000H占两个字节，且紧跟在指令操作码之后存放在代码段之中。

B8?

- ✓执行前：（AX）＝0000H
- ✓执行后：（AX）＝2000H。

用途：寄存器或存储器赋初值，也可与寄存器或存储器操作数进行算术逻辑运算

优点：简单、快速

2. 直接寻址

在该寻址方式中，指令直接给出了操作数的偏移地址，操作数存放在存储器内，操作数的偏移地址与操作码一起存放在指令中，且紧跟在指令码之后。要想存取操作数必须首先生成物理地址，操作数物理地址是由段地址和偏移地址EA相加而成，默认段为数据段。

书上图

- ✓汇编格式：[n]或含有变量的地址表达式
- ✓操作数位置：存储器
- ✓功能：n或含有变量的地址表达式的值是操作数的偏移地址EA，且存放在指令的下一个字单元中，但操作数存放在存储器中。

直接寻址例题

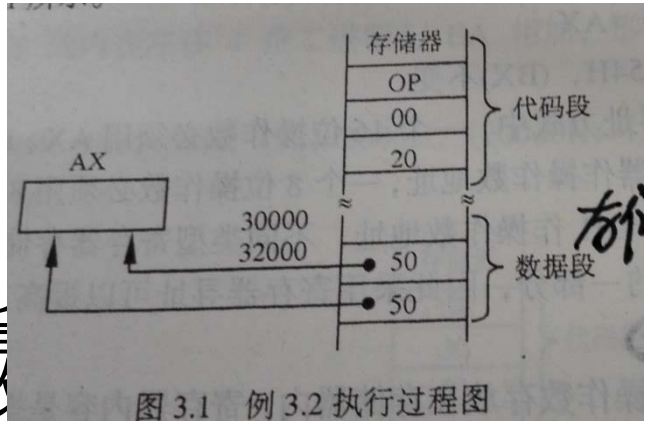


图 3.1　例 3.2 执行过程图

例3.2 MOV AX，[2000H]

✓说明：源操作数采用直接寻址方式，指
容2000H为源操作数的EA，操作数存放

✓执行前：(DS)＝3000H， 执行后：(AX)＝5050H

例3.3 MOV AX，BUFA

✓说明：源操作数采用直接寻址方式，指令码下一字单元的内容为变量BUFA的EA，操作数存放在存储器变量BUFA所指向的存储单元中。指令功能是将字类型变量BUFA的内容送入AX之中。

✓执行前：假定BUFA在数据段中定义，EA＝1000H，(DS)＝2000H，(AX)＝1122H，(21000)＝1259H，

✓执行后：（AX）＝1259H

例3.4

3. 寄存器寻址

在该寻址方式中，寄存器名是操作数的符号地址，寄存器内容是指令所需的操作数。

✓汇编格式：R（R是寄存器名）

✓操作数位置：寄存器

✓功能：寄存器R的内容即为操作数

例3.5 INC AX

✓说明：INC为加1指令操作符，单操作数指令，目的操作数采用寄存器寻址，AX的内容即为目的操作数。

✓执行前：(AX)=3344H

✓执　行：(AX)＋1→AX

✓执行后：(AX)=3345H

例3.1-3.4

寄存器寻址例题

例3.6 DEC CX

✓说明：DEC为减1指令操作符，单操作数指令，寄存器CX为目的操作数地址，CX的内容为目的操作数。

✓执行前：(CX)=78H，

✓执　行：(CX)-1→CX

✓执行后：(CX)=77H

注意：16位操作数、8位操作数用不同寄存器；优点：速度快，提高效率

立即寻址、直接寻址和寄存器寻址那个最快、最慢？

例3.7 ADD AX，BX

✓说明：ADD是双操作数指令，功能是实现加法运算，AX为目的操作数地址，AX的内容为目的操作数，BX为源操作数地址，BX的内容为源操作数。

✓执行前：(AX)=1234H，(BX)＝5620H，即

✓执　行：(AX)＋(BX) →AX

✓执行后：(AX)＝6854H，(BX)不变。

4. 寄存器间接寻址

✓在该寻址方式中，操作数存放在存储器内，寄存器内容是操作数的偏移地址EA，要想寻找操作数必须首先生成操作数的物理地址。

✓能够用作间接寻址的寄存器只能是寄存器BX、BP、SI、DI其中之一。操作数所在段的段地址8086处理器有默认规定，即若寄存器BX、SI或DI作间接寻址，则操作数存放在数据段中；若寄存器BP作间接寻址，则操作数存放在堆栈段中。

?

✓汇编格式：[R]

✓操作数位置：存储器　书中图(有问题?)

✓功能：寄存器R的内容为操作数的偏移地址EA，

寄存器间接寻址例题

例3.8 ADD AX，[SI]

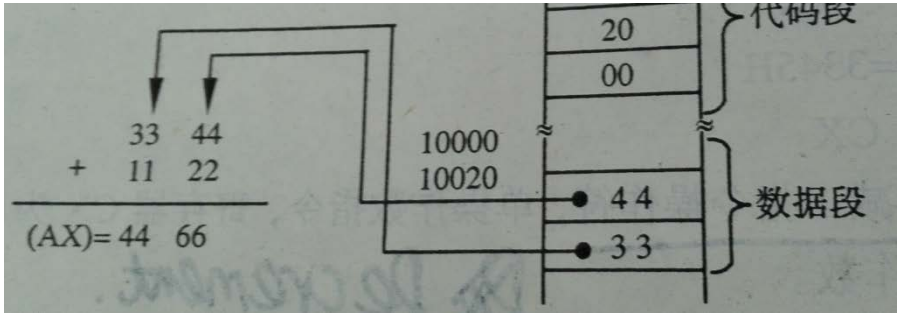
✓说明：ADD是双操作数指令，功能是实现加法运算，AX是目的操作数地址，寻址方式是寄存器寻址方式，[SI]是源操作数地址，寻址方式采用寄存器间接寻址，操作数在存储器内，默认在数据段中。

✓执行前：(AX)＝1122H，(SI)=20H，(DS)=1000H，(10020H)=3344H

✓执行：(10020H)+(AX)→AX

✓执行后：(AX)=4466H，(SI)、(DS)、(10020H)内容不变。

✓执行过程如图3.2所示。



寄存器间接寻址例题

例3.9 SUB AX，[BP]

✓说明：SUB是双操作数指令，功能是实现减法运算，目的操作数地址是AX，源操作数的偏移地址EA为BP的内容。由于源操作数用BP作间接寻址，所以源操作数存储在堆栈段中，寄存器SS的内容左移4位二进制与EA相加，形成源操作数的物理地址PA。

✓执行前：(AX)=3344H，(BP)＝30H，(SS)=2000H，(20030H)=1234H

✓执行：(20030H) →AX

✓执行后：(AX)=2110H，(BP)、(SS)、(20030H)不变。

图3.3

5. 寄存器相对寻址

- ✓在该寻址方式中，操作数存放在**存储器**中，而操作数的**偏移地址EA**是指令中指定的**寄存器的内容**与指令中给出的**位移量相加**之和，与寄存器间接寻址方式相比，偏移地址EA的生成**增加了一个相对位移量X**。
- ✓汇编格式：**X[R]**
- ✓操作数位置：存储器
- ✓功能：R的内容与X相加之和为操作数的偏移地址EA。
- ✓其中，X表示位移量，R表示寄存器名。寄存器**只能选择**SI、DI、BP、BX其中之一。操作数所在段的段地址遵循8086处理器的**默认规定**。

?

寄存器相对寻址例题

例3.10 MOV AX，5[BX]

✓说明：源操作数采用寄存器相对寻址，**X相对量**的值为5，**BX作间接寻址寄存器**，所以源操作数的EA=[BX]+5，**数据段寄存器DS**的内容左移4位二进制与EA相加，形成源操作数的物理地址PA。**目的操作数**采用寄存器寻址，操作数地址为AX。

✓**执行前**：(AX)＝0000H，(BX)＝1BH，(DS)＝2000H，(20020H)=3789H。

✓**执　行**：　(20020H)→AX

✓**执行后**：(AX)=3789H，(BX)、(DS)、(20010H)内容不变。

图3.4

寄存器相对寻址例题

例3.11 MOV 6[BP]，BX

✓说明：**目的操作数采用寄存器相对寻址**，偏移地址EA＝[BP]+6。由于BP用作间接寻址寄存器，所以**堆栈段寄存器SS**的内容左移4位二进制与EA相加，形成目的操作数的物理地址PA。源操作数采用寄存器寻址，操作数地址为BX。

✓**执行前**：(BX)＝4000H，(BP)＝30H，(SS)＝2000H，(20036H)=0000H。

✓**执　行**：　(BX)→20036H

✓**执行后**：(20036H)=4000H，(BX)、(BP)、(SS)未变。

图3.5

访问数组中任一元素：X+[R]

6. 基址加变址寻址

- ✓在该寻址方式中，操作数存放在存储器里，操作数的偏移地址EA是由指令中指定的基址寄存器内容、变址寄存器内容相加之和组成。

- ✓汇编格式：[BR+IR]
- ✓操作数位置：存储器
- ✓功能: BR的内容加上IR的内容之和即是操作数的偏移地址EA。
- ✓其中，BR表示基址寄存器，只能选用BX、BP之一；IR表示变址寄存器，只能选用SI、DI之一。

书中图

注意：

- ✓基址寄存器选用BX还是BP，决定了是从**堆栈段中还是从数据段**中获取操作数。
- ✓也就是说，若选用**BP作为基址寄存器**，操作数应保存在**堆栈段**中，段寄存器SS的内容左移4位二进制与偏移地址EA相加，形成操作数的物理地址PA
- ✓若选用**BX作为基址寄存器**，操作数应保存在**数据段**中，段寄存器DS的内容左移4位二进制与偏移地址EA相加，形成操作数的物理地址PA。

基址加变址寻址例题

例3.12 MOV AX，[BX][SI] (**等价形式**[BX＋SI])

✓说明：目的操作数地址是AX，源操作数偏移地址EA＝[BX]＋[SI]，其中，基址寄存器选用了**BX**，变址寄存器选用了SI。由于源操作数选用BX作基址寄存器，所以其物理地址PA是由**数据段寄存器DS**的内容左移4位二进制与偏移地址EA相加形成。

✓**执行前**：(AX)=45H，(BX)＝30H，(SI)＝20H，(DS)=1000H，(10050H)=9988H。

✓**执　行**：(10050H)→AX

✓**执行后**：(AX)=9988H，(BX)、(SI)、(DS)、(I0050H)不变。

基址加变址寻址例题

例3.13 MOV [BP][DI]，BX (**等价形式** [BP＋DI])

- ✓ 说明：源操作数地址是BX，目的操作数偏移地址EA＝[BP]＋[DI]，基址寄存器选用了**BP**，变址寄存器选用了DI。由于目的操作数选用BP作基址寄存器，所以其物理地址PA由**堆栈段寄存器SS**的内容左移4位二进制与偏移地址EA相加形成。

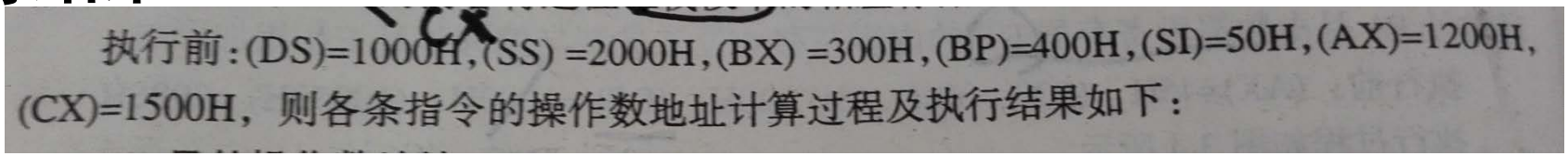
- ✓ **执行前**：(BX)=3344H，(BP)＝10H，(DI)＝40H，(SS)=4000H，(40050H)=0000H。
- ✓ **执　行**：(BX)→40050H
- ✓ **执行后**：(40050H)=3344H，(BX)、(BP)、(DI)、(SS)均不变。

7. 相对基址加变址寻址

- ✓ 在该寻址方式中，操作数存放在存储器里，操作数的偏移地址EA是由指令中指定的基址寄存器内容、变址寄存器内容及位移量X三项相加之和组成。
- ✓ 汇编格式：X[BR+IR]
- ✓ 操作数位置：存储器
- ✓ 功能: BR的内容加上IR的内容，再加上位移量X，所得之和是操作数的偏移地址EA。
- ✓ 其中，X表示位移量，BR表示基址寄存器，只能选用BX、BP之一；IR表示变址寄存器，只能选用SI、DI之一。
- ✓ 同样在这里，基址寄存器选用BX还是BP，决定了是从堆栈段中还是从数据段中获取操作数。也就是说，若选用BP作为基址寄存器，操作数在堆栈段中，若选用BX作为基址寄存器，则操作数在数据段中，操作数物理地址的生成与基址加变址寻址方式相同。

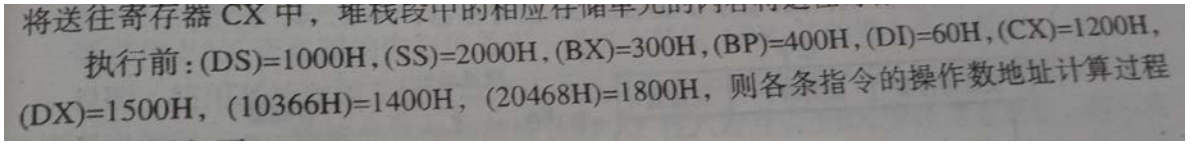
相对基址加变址寻址例题

例3.14 计算下列2条指令中操作数的地址并指出执行结果。


执行前：(DS)=1000H，(SS) =2000H，(BX) =300H，(BP)=400H，(SI)=50H，(AX)=1200H，
(CX)=1500H，则各条指令的操作数地址计算过程及执行结果如下：

1. MOV 2 [BX+SI]，AX
2. MOV 4 [BP+SI]，CX

例3.15： 计算下列2条指令中操作数的地址并指出执行结果。


将送往寄存器 CX 中，堆栈段中的相应存储单元的内容送往寄存器 DX 中。
执行前：(DS)=1000H，(SS)=2000H，(BX)=300H，(BP)=400H，(DI)=60H，(CX)=1200H，
(DX)=1500H，(10366H)=1400H，(20468H)=1800H，则各条指令的操作数地址计算过程

1. MOV CX， 6 [BX+DI]
2. MOV DX， 8 [BP+DI]，

数组列数！

3.1.2 32位寻址

✓**纯32位寻址**是32位微处理器在**保护模式**下的一种寻址方式，需要提供32位偏移地址。

✓80x86系列从**80386**起就把机器字长从16位增加到32位，相应的16位寄存器也扩展为32位寄存器(EAX，EBX，ECX，EDX，ESP，EBP，EDI，ESI)，这些扩展后的寄存器**既可保存数据也可保存地址**，统称为**通用寄存器**。

- ✓386及其后继机型除提供32位寻址外，**还兼容了16位寻址**。在使用32位寻址时，32位通用寄存器可以作为基址或变址寄存器使用。也就是说，允许32位通用寄存器作**指针寄存器用**。
- ✓**在实模式下，这两种寻址方式可同时使用**，但段的大小被限制在**64KB之内**，这样段内的偏移地址范围应为0000-FFFFH，所以在把32位通用寄存器用作指针寄存器时，应该注意它们的高16位应为0。
- ✓本节内容是指在**实模式下的32位寻址**，如果32位指针寄存器高16位不为0，则需要在保护模式下寻址。

8. 相对比例变址寻址方式

- ✓在该寻址方式中，操作数的有效地址是**变址寄存器的内容**乘以指令中指定的**比例因子**再加上**相对位移量**之和，所以偏移地址EA由三种成分组成，它允许使用**除ESP以外的32位通用寄存器**，默认段通常为**数据段**。
- ✓汇编格式：X[ER×N]
- ✓操作数位置：存储器
- ✓功能：EA＝(ER)×N+X，ER表示扩展后的32位寄存器，N＝2，4，8。
- ✓这种寻址方式与寄存器相对寻址方式相比，**增加了比例因子**。其优点在于：对于元素大小为2，4，8字节的数组而言，可以**在变址寄存器中给出数组元素下标**，而由寻址方式控制直接使用**比例因子**把下标转换为数组元素的偏移地址起始值。

16位：
SI、DI？
取决于基址寄存器

例3.16

9. 基址比例变址寻址方式

- ✓在该寻址方式中，操作数的偏移地址EA是**变址寄存器**的内容乘以**比例因子**再加上**基址寄存器**的内容之和，默认段通常为**数据段**。

- ✓汇编格式：[ER][ER×N]
- ✓操作数位置：存储器
- ✓功能：EA＝(ER)+(ER)×N，ER表示扩展后的32位寄存器，N＝2，4，8。
- ✓这种寻址方式与基址变址寻址方式相比，增加了**比例因子**，其优点是对数组元素能够更加方便的访问

例3.17

10. 相对基址比例变址寻址方式

✓在该寻址方式中，操作数的有效地址是**变址寄存器**的内容乘以**比例因子**，加上**基址寄存器**的内容，再加上**位移量**之和，所以有效地址由四种成分组成，默认段通常为**数据段**。

✓汇编格式：X[ER][ER×N]
✓操作数位置：存储器
✓功能：EA＝(ER)+(ER)×N＋X，ER表示扩展后的32位寄存器，N＝2，4，8。
✓这种寻址方式与相对基址变址寻址方式相比增加了比例因子，因此对元素为2，4，8字节的**二维数组**的处理将更加方便。

注意：

- ✓上面三种（8，9，10）寻址方式均与**比例因子**有关，这些寻址方式只能用在80386及其后继机型中，8086/80286不支持这三种寻址方式。

EA公式；10种方式是特例

3.2 指令寻址方式

- ✓**指令寻址方式**即寻找程序指令**入口地址**的方式，也就是当一条指令执行时，如何确定下一条将要执行指令的入口地址。

- ✓前面对专用**寄存器IP**已经讲过，它是专门用来存放下一条将要执行的指令**入口地址**的寄存器，指令的流向是由IP指针来确定的。

- ✓一条指令入口地址的表示形式为**CS：IP**，根据CS和IP的内容，指令寻址方式可分为以下四种：**段内直接、段内间接、段间直接、段间间接**。

3.2.1 段内直接寻址

- ✓**段内直接寻址**是指**转移指令**语句与该指令执行后将跳转到的**转向指令语句**在**同一代码段**内，转向指令的**有效地址**在转移指令语句中**直接给出**，常用的是**标号**形式。
- ✓该指令寻址方式的**实质**是CS值不变，IP改变，转移指令中**符号地址的值**即为转向的有效地址。
- ✓它类似于**数据寻址**方式中的**直接寻址**。

✓转移指令常用JMP汇编指令，它是无条件转移指令，相当于C语言的GOTO语句，该指令语句的汇编格式为：

JMP <指令入口地址CS：IP>

✓其中，段内跳转CS可以省略。

✓例如，JMP LABEL（标号）

✓该语句的功能是指令流无条件跳转到标号为LABEL的指令的入口处。

✓段内直接寻址转移指令的汇编格式可以表示为：

JMP <标号>

```
code1  segment
      ……
      jmp  next
      ……
Next：……
      ……
code1  ends
```

指令功能：标号值即为转向指令的EA。

反汇编：

```
C:\BIN>debug string.exe
-u
0B46:0000 B8440B        MOV  AX, 0B44
0B46:0003 8ED8          MOV  DS, AX
0B46:0005 8D160000      LEA  DX, [0000]
0B46:0009 EB04          JMP  000F
0B46:000B B409          MOV  AH, 09
0B46:000D CD21          INT  21
0B46:000F B44C          MOV  AH, 4C
0B46:0011 CD21          INT  21
```

3.2.2 段内间接寻址

- ✓段内间接寻址是指转移指令语句与执行后将跳转到的转向指令语句在同一代码段内，转向指令的有效地址在转移指令语句中间接给出，常用的是一个寄存器或是一个存储单元。
- ✓该指令寻址方式的实质是CS值不变，IP改变，转向指令的有效地址存储在寄存器或存储单元中。
- ✓这个寄存器或存储单元的内容可以用数据寻址方式中除立即数以外的任何一种寻址方式取得，所得到的转向指令的有效地址用来取代IP寄存器的内容。

?

此时用R/[R]与数据寻址时有何不同！

段内间接寻址转移指令的汇编格式

✓汇编格式1：JMP BX ？？

✓指令功能：BX寄存器内容为转向指令的EA。

✓汇编格式2： JMP WORD PTR[BP+TABLE]

?✓指令功能：数据段中EA＝(BP)+TAABLE的存储单元字内容为转向指令的EA。

✓其中，WORD PTR为属性操作符，用以指出其后的寻址方式所取得的转向地址是一个字类型的有效地址，也就是说它是一种段内转移。

✓以上两种寻址方式均为段内转移，所以直接把求得的转向指令的有效地址送到IP寄存器即可

书上例子

3.2.3 段间直接寻址

- ✓段间直接寻址是指转移指令语句与执行后跳转到的转向指令语句不在同一代码段内，转向指令语句的段地址和偏移地址在转移指令中直接给出，所以转移指令语句应直接提供4字节地址内容。
- ✓只要用转移指令中指定的偏移地址取代IP寄存器的内容，用段地址取代CS寄存器的内容，就完成了从一个段到另一个段的转移操作。

段间直接转移指令的汇编格式可表示为：

✓汇编格式1： JMP FAR PTR NEXT

```
code2  segment
        ……
 next:  ......
        ……
code2  ends
code1  segment
      ……
 jmp  far ptr next
      ……
code1  ends
```

✓指令功能：NEXT为另一个段内转向语句的符号地址，FAR PTR是表示段间转移的操作符。即：

✓说明：指令执行后，指令指针IP将指向code2代码段中next标识的语句，CS变为code2代码段的段地址，IP变为next标号值。

3.2.4 段间间接寻址

- ✓段间间接寻址不但指转移指令语句与执行后跳转到的转向指令语句不在同一代码段内，而且转向指令的段地址和偏移地址也被存放在存储器中间接提供，用存储器中的相继两个字单元的内容来取代IP和CS寄存器中的原始内容，以达到段间转移的目的。
- ✓存储单元的地址由转移指令指定，可以通过数据寻址方式中存储器寻址取得。

这种指令的汇编格式可表示为：

✓汇编格式：JMP FAR PTR X[R]

✓指令功能：X[R]为**寄存器相对寻址**方式，操作数在存储器内，FAR PTR为双字段间操作符，转移指令应取双字内容填充段间转向指令的CS：IP。

3.3 跨段的有关问题

寄存器直接寻址？？

- ✓8086的存储器是分段使用的。通常，若选用寄存器BP作为寄存器间接寻址、变址寻址或基址加变址寻址时，则操作数在当前堆栈段中，操作数的物理地址PA由堆栈段寄存器SS的内容左移4位二进制与偏移地址EA相加形成
- ✓否则，操作数在当前数据段，操作数的物理地址PA则由数据段寄存器DS的内容左移4位二进制与偏移地址EA相加形成。
- ✓这是8086的基本约定，即默认状态。当要否定默认状态，到非约定段寻找操作数时，必须采用跨段前缀运算符“：”来指明操作数所在段的段寄存器。

跨段前缀符“ ：”

- ✓跨段前缀符“：”用于临时给变量、标号或地址表达式指定一个段属性，且只在所出现的语句中有效，它并不改变地址表达方式的偏移地址和类型属性。
- ✓汇编格式：段寄存器名：地址表达式，
- ✓或：段名：地址表达式
- ✓功能：冒号“：”之前的段名或段寄存器指明了操作数所在段。

例3.27(1) MOV AX，DS：[BP]

(2) MOV BX，ES：[BX] 书上；错误

(3) MOV CX，SS：[SI]

(4) MOV DX，SS：[DI]

PA计算：

上述4条指令的源操作数物理地址分别计算如下：

✓(1)：PA=（DS）左移4位+[BP]

✓(2)：PA=（ES）左移4位+[BX]

✓(3)：PA=（SS）左移4位+[DI]

✓(4)：PA=（SS）左移4位+[DI]

间接寻址？

变址寻址？

基址加变址寻址？

表3.1

3.4 实模式与保护模式

- ✓实模式即实地址访问模式，它是Intel 80x86 CPU处理器的一种操作模式。实模式的最大寻址能力是2^{20}＝1MB，可访问内存空间包括物理内存和BIOS-ROM。
- ✓*实模式下处理器没有硬件级的内存保护概念和多道任务的工作模式。*8086和8088只能工作于实模式，而80286及以上的处理器可工作于实模式或者保护模式下。
- ✓但是为了向下兼容，80286及以后的X86系列兼容处理器仍然是开机启动时工作在实模式下。

PPT25：实模式（16、32），保护（纯32）
80286:16位；保护？？

- ✓在实模式存储器寻址中，程序员只要在程序中给出段地址和偏移地址，机器就会自动用段地址左移4位二进制再加上偏移地址的方法，求得所选存储单元的物理地址，从而取得相应存储单元的内容。

- ✓因此，程序员在编程时并未直接指定所选存储单元的物理地址，而是给出了一个逻辑地址（即段地址：偏移地址），是机器自动用某种方法来取得所选存储单元的物理地址的。

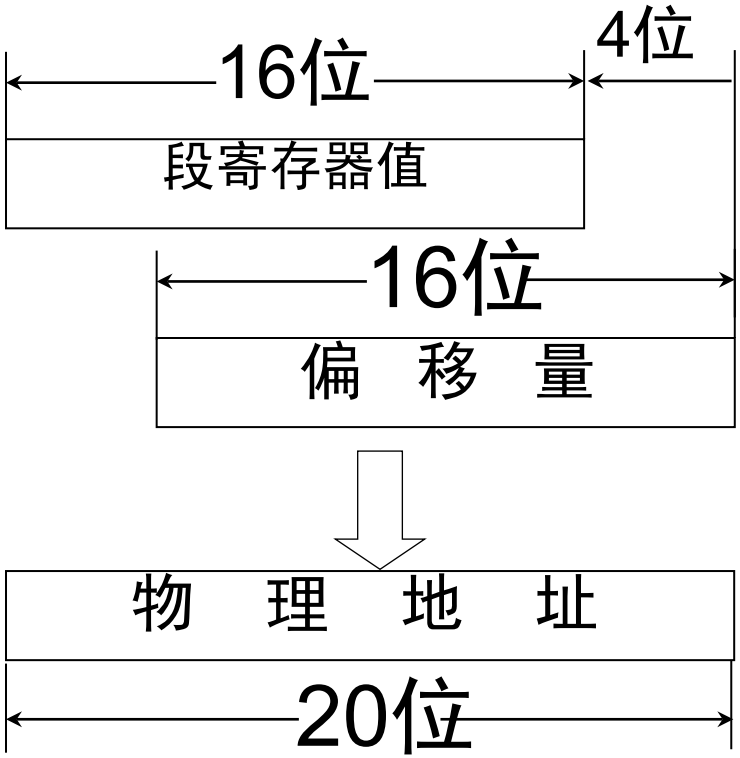


图3.8　实模式存储器寻址示意图

保护模式

- ✓保护模式又称作**虚拟地址保护模式**。尽管在Intel 80286已经提出了虚地址保护模式，但它并没有完全地实现80386及以后的保护模式功能，实质上还是实模式，是在实模式上模拟的保护模式，真正的32位地址出现在Intel 80386上。
- ✓保护模式本身是80286及其兼容处理器序列之后产生的一种操作模式，为提高系统的多道任务和系统的稳定性它具有许多特性设计。

注意：

- ✓保护模式的特性是*阻止*被其他任务或系统内核破坏已经不健全的*程序的运行*，保护模式也有对硬件的支持，例如中断运行程序，移动工作进程文档到另一个进程和置空多任务的保护功能。

说明：

- ✓在保护模式存储器寻址中，仍然要求程序员在程序中指定逻辑地址，只是机器采用另一种比较复杂、或者说比较间接的方法来求得相应的物理地址。
- ✓因此，对程序员编程来说，并未增加复杂性。
- ✓*在保护模式下，逻辑地址是由选择器和偏移地址两部分组成*，选择器存放在段寄存器中，但它不能直接表示段基地址，而由操作系统通过*地址转换*取得段基地址，再和偏移地址相加，从而求得所选存储单元的物理地址。

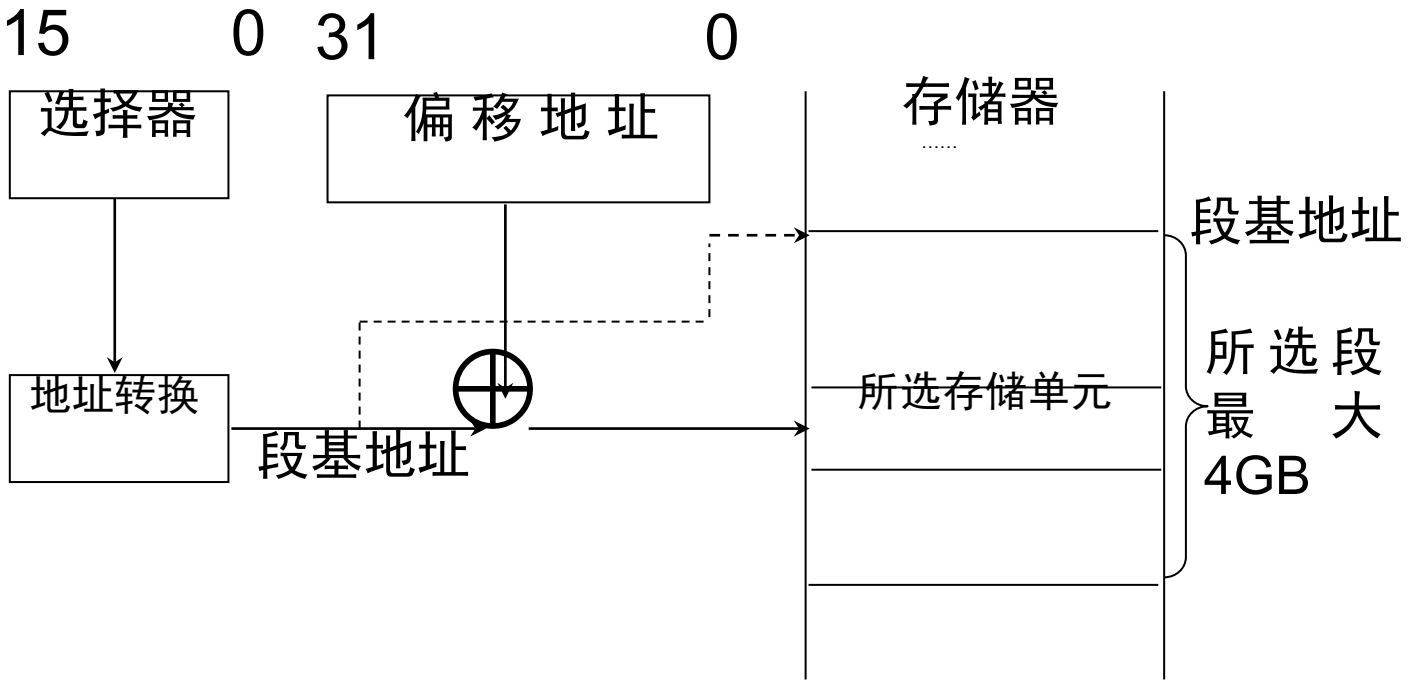


图3.9　保护模式存储器寻址示意图

主要区别：

✓由此可见，实模式和保护模式的主要区别是寻址能力不同，实模式下寻址能力为1MB，而保护模式下寻址能力可以扩展到4GB。

3.5 综合举例

综合例题1：

✓编程实现把buf1存储空间的内容移动到buf2存储空间中去。