# 寻址方式

1. 指令组成

【标号】：指令助记符 【目的操作数】，【源操作数】；【注释】

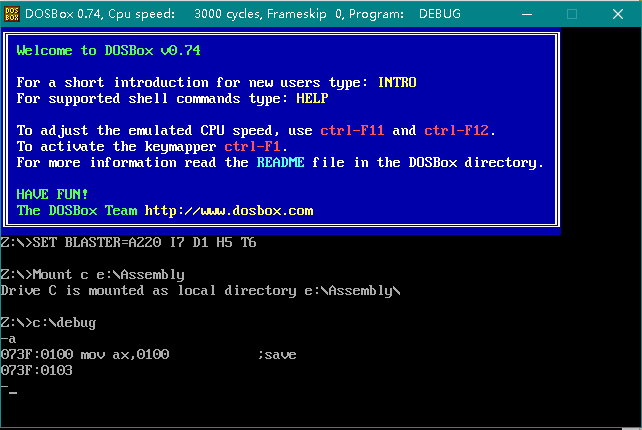
（见图2.1.1）

标号表示该指令在主存中的逻辑地址。

每个指令助记符就代表一种指令。

目的和源操作数表示参与操作的对象。

注释是对该指令或程序段功能的说明。



2.1.1

二、指令中的操作数

1、每种指令的操作码：

①用一个**唯一**的助记符表示（指令功能的英文缩写）；

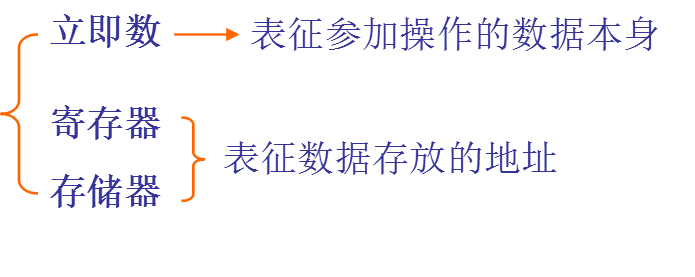
②对应着机器指令的一个二进制编码。

2、指令中的操作数：

①可以是一个具体的数值（立即数）；

②可以是存放数据的寄存器（寄存器）；

③或指明数据在主存位置的存储器地址（内存单元）。

主要分为以下三种：

1. 立即数操作数

立即数本身是参加操作的数据，可以是8位或16位，**只能作为源操**

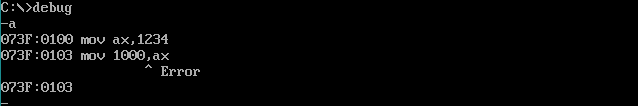
**作数**，即立即数**无法成为目标操作数（见图2.1.1）**。

如：

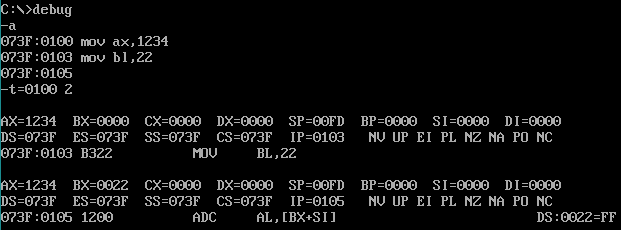
MOV AX,1234H

MOV BL,22H

（此处H在debug指令输入时，无需添加，自动默认为十六进制数）



2.2.1 无法成为目标操作数



2.2.2

1. 寄存器操作数

参加操作的操作数在CPU的通用寄存器中。

AH、AL、BH、BL、CH、CL、DH、DL

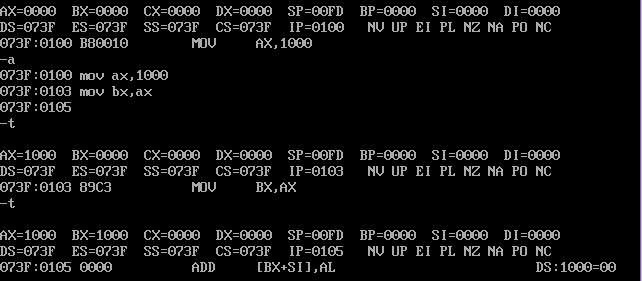
AX、BX、CX、DX、SI、DI、BP、SP

CS、DS、SS、ES

如：

MOV AX,1000

MOV BX,AX



2.2.3

1. 存储器操作数

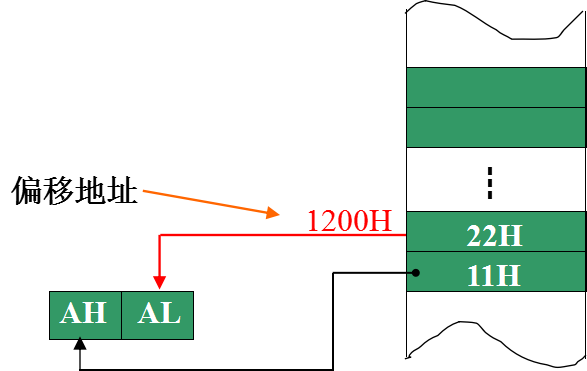
参加运算的数存放在存储器的某一个或某两个单元中。

表现形式：[立即数或寄存器]

**需注意，[ ]中的内容是存放所寻找数据的单元的偏移地址。**

如：

MOV AX,[1200]



2.2.4

1. 寻址方式

8086处理器的寻址方式笼统地可以分为：**三类七种**。

寻址方式

直接寻址方式

寄存器间接寻址方式

寄存器相对寻址方式

基址变址寻址方式

相对基址变址寻址方式

寄存器寻址方式

操作数存放于CPU的某个内部寄存器中，较常用。

立即数寻址方式

操作数直接存放在机器代码中，紧跟在操作码之后。

仅适合于源操作数。

存储器寻址方式

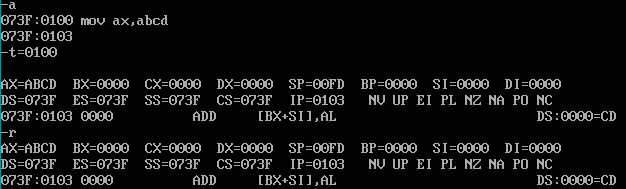
操作数存放才除代码段以外的存储区中，五种方式，五种偏移地址（EA）计算方法。

1、立即数寻址

指令中的源操作数是立即数，即源操作数是参加操作的数据本身。

特点是执行速度快，主要给寄存器赋值，但不能直接给段寄存器 （DS,CS,SS,ES）赋值。

例：MOV AX，abcd



2.3.1

1. 寄存器寻址

参加操作的操作数在CPU的通用寄存器中。

AH、AL、BH、BL、CH、CL、DH、DL

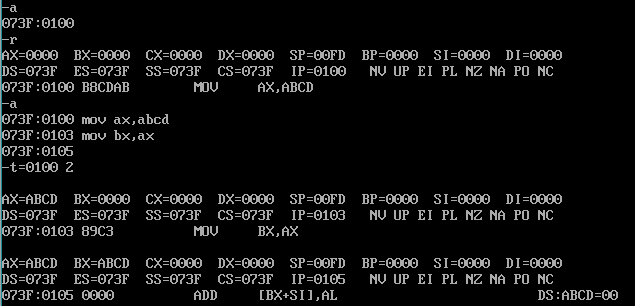
AX、BX、CX、DX、SI、DI、BP、SP

CS、DS、SS、ES

例：

MOV AX,abcd ;立即数寻址

MOV BX,AX ;寄存器寻址

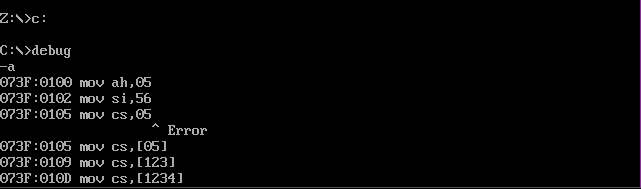


2.3.2

**需注意，如果要想操作数存放在4个寄存器中，则需要用中括号**

**将操作数括起来，其他寄存器则不用。**

如：



2.3.3

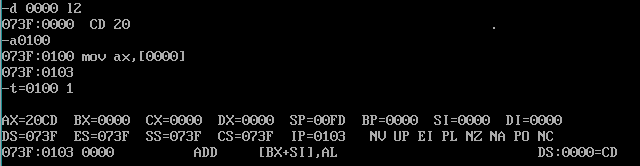
1. 存储器寻址方式

①直接寻址

指令中直接给出操作数的偏移地址

例：

MOV AX,[0000]



2.3.4

直接寻址方式下，操作数的段地址默认为数据段(DS)，但允

许段重设，即由指令定义段。

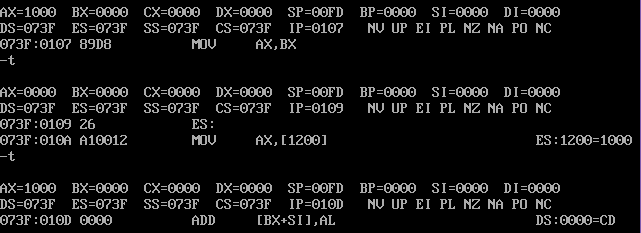
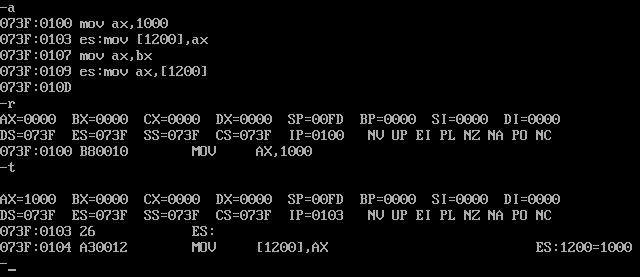
例：

MOV AX,1000

ES:MOV [1200],AX ;注意段超越格式

MOV AX,BX

ES:MOV AX,[1200]



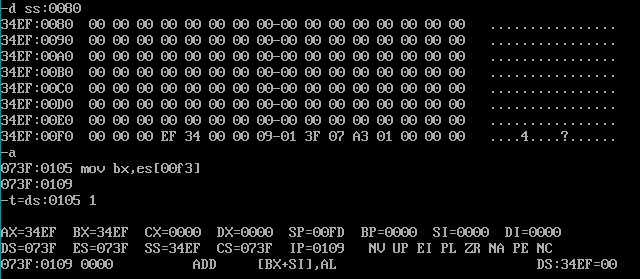
2.3.5

段超越的格式亦可如下所示

例：

MOV BX,ES[00F3]

②寄存器间接寻址



2.3.6

参与操作的操作数存放在内存中，其偏移地址为指令中的寄存器的 内容（简例可返回至存储器操作数处查看，图见2.2.4）。

存放偏移地址的寄存器称为**间址寄存器**，它们是：**BX,BP,SI,DI**。

操作数的段地址（数据处于哪个段）取决于选择哪一个间址寄存器。

**BX，SI，DI 默认在数据段（DS）**

**BP 默认在堆栈段（SS）**

例：

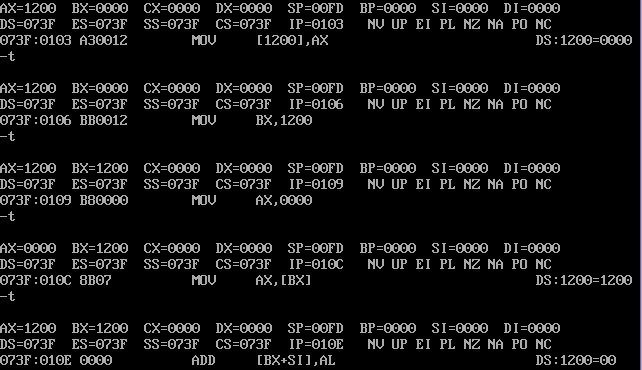
MOV AX,1200

MOV [1200],AX

MOV BX,1200

MOV AX,0

MOV AX,[BX]



2.3.6

③寄存器间接相对寻址

操作数的偏移地址为寄存器的内容加上一个**位移量。**

**需注意的是存放偏移地址的寄存器只能是间址寄存器 （BX,BP,SI,DI）,所以不能出现类似的指令，如：[ax+200] , 200[ax],这样将会发生错误。**

例：

MOV AX,1200

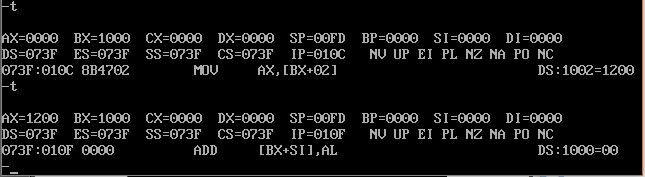
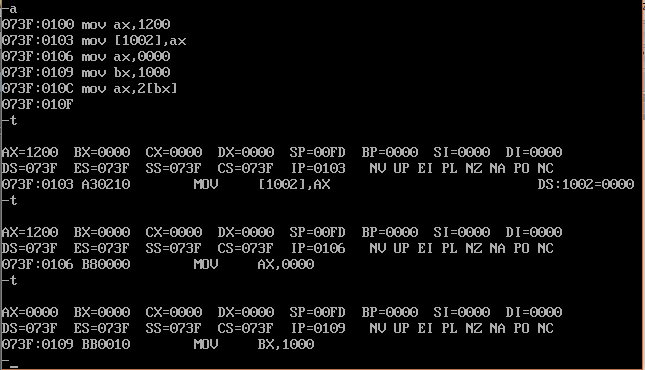
MOV [1002],AX

MOV AX,0000

MOV BX,1000

MOV AX,2[BX]

④基址-变址寻址



2.3.7

操作数的偏移地址为

一个基址寄存器的内容 + 一个变址寄存器的内容；

操作数的段地址由选择的基址寄存器决定

基址寄存器为BX，默认在数据段（DS）

基址寄存器为BP，默认在堆栈段（SS）

变址寄存器为SI，DI

**需注意：mov ax,[bx][bp] error不能两个基址寄存器放一起**

**mov ax,[si][di] error不能两个变址寄存器放一起**

**mov ax,[bp][si] correct**

基址变址寻址方式与相对寻址方式一样，主要用于一维数组操作。

例：

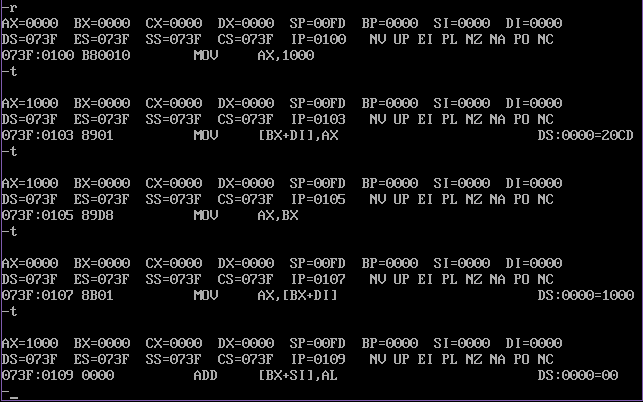
MOV AX,1000

MOV [BX][DI],AX

MOV AX,BX

MOV AX,[BX][DI]

⑤基址-变址相对寻址



2.3.7

操作数的偏移地址为：

基址寄存器内容+变址寄存器内容+位移量

或：BX/BP（基址寄存器）＋SI/DI（变址寄存器）＋位移量

操作数的段地址由选择的基址寄存器（BX对应DS，BP对应SS）决定。

（基址变址相对寻址方式主要用于二维表格操作）

**注意点同基址-变址寻址中的注意点。**

例：

MOV AX,[BX+SI]

MOV BX,0100

MOV SI,2

MOV AX,2[BX][SI]（此处，意味：2+AX+SI，等同于[ax+si+2]）

（此处示例见图2.3.8）

⑥隐含寻址

因指令而异，如mul指令：

MUL BL ; 相当于ax=al×bl

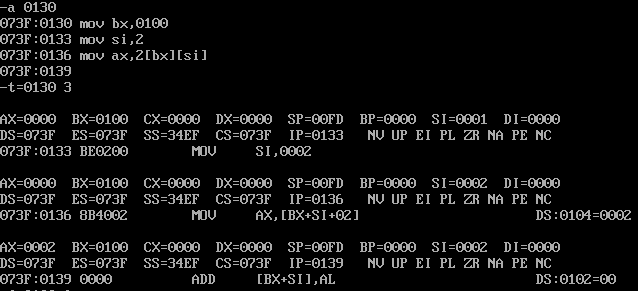
例：

MOV BL,2

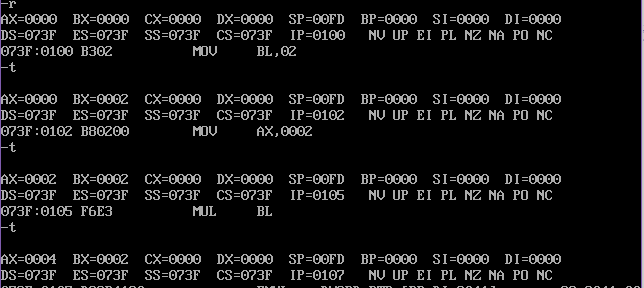
MOV AX,2

MUL BL

（此处示例见图2.3.9）



2.3.8



2.3.9

# 第三章 指令系统

**一、指令与指令系统**

指令长度

多操作数指令：三操作数及以上

单操作数指令：操作码 操作数

双操作数指令：操作码 操作数,操作数

零操作数指令：操作码

指令的执行速度：存储器>立即数>寄存器

**二、IA-32指令系统**

1、数据传送类指令

数据传送

算术运算

逻辑运算和移位

串操作

程序控制

处理器控制

（1）通用数据传送

特点：该类指令的执行对标志位（FLAGS）不产生影响！！

（所有指令的格式均不做文字叙述，图例中都有。）

一般数据传送指令

注意点：

1. 两操作数字长必须相同；
2. 不能同时为存储器操作数([立即数或寄存器])；不能同时为段寄存器(DS,ES,SS)；源操作数是立即数时，目标操作数不能是段寄存器；
3. IP和CS不作为目标操作数，FLAGS一般也不作为操作数在指令中出现。

若要改变段寄存器，先将数据存入非段寄存器，再送入段寄存器。

堆栈操作指令

注意点：

1. 指令的操作数必须是16位的；
2. 操作数可以是寄存器或存储器两单元，但不能是立即数；
3. 不能从栈顶弹出一个字给CS（允许PUSH CS）；
4. PUSH和POP指令在程序中一般成对出现；
5. PUSH指令的操作方向是从高地址向低地址，而POP指令的操作正好相反；
6. 堆栈指令遵循先进后出的原则，可用于交换两操作数的位置，也可用于数据的暂时存放，后面遇到时再做了解。

交换指令

注意点：

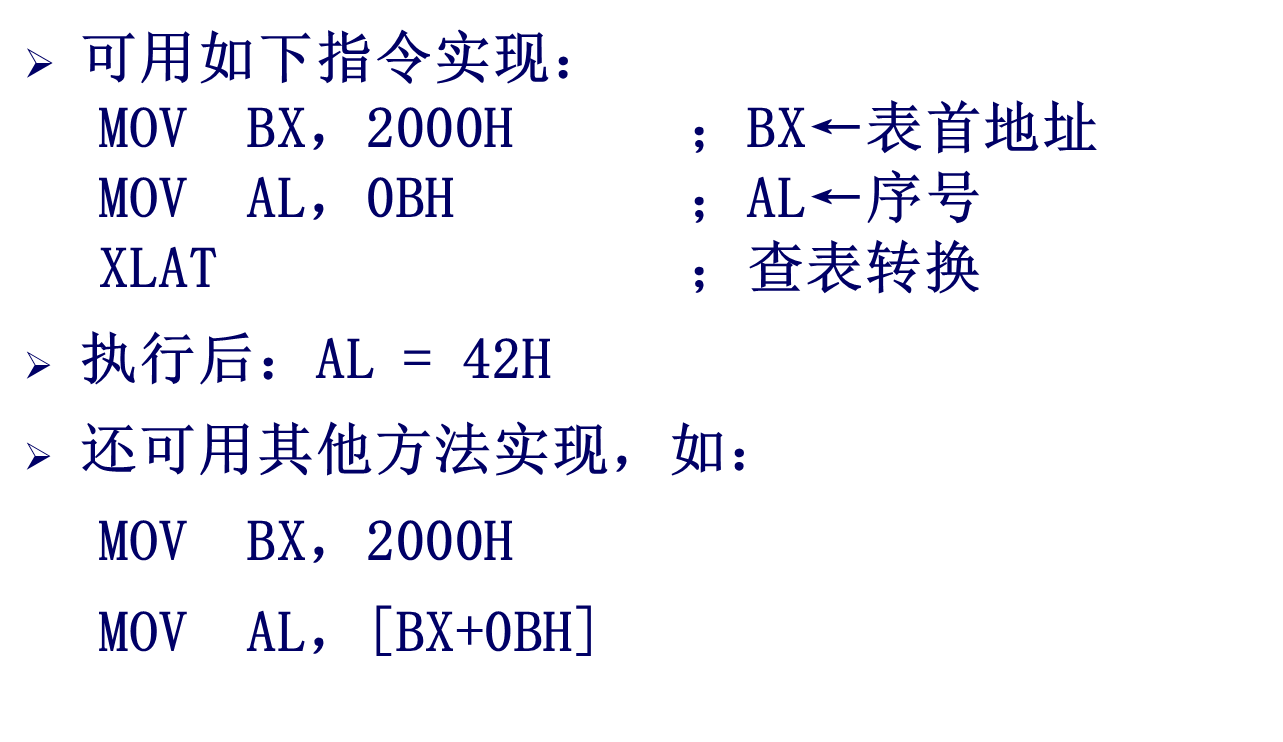
1. 两操作数必须有一个是寄存器操作数；
2. 不允许使用段寄存器；
3. 两个操作数字长要一致。

查表指令

说明：

1. BX存放“表格”的首地址，AL内容为在表内位移量，如：查找表中

第三行，则AL存入3即可。BX+AL即所寻元素的偏移地址；

1. 最大长度不超过256字节；
2. 其本身蕴含隐藏寻址，将BX+AL的内容存入AL中，多用于后期大汇

编编程，此处不做累述；

字位扩展指令

说明：

1. 将有符号数的符号位扩展到高位；
2. 指令为零操作数指令，采用隐含寻址，隐含的操作数为AH及DX(个

人理解)；

1. 无符号数的扩展规则为在高位补零。

CBW（CWD）：

将AL(AX)扩展到AH(DX)；

最高位为1，AH(DX)=FF(FF)；

最高位为0，AH(DX)=00(00)。

（2）地址传送指令

取偏移地址指令LEA（大多用于大汇编程序编写）

具体见图例(源操作数MEM必须是一个存储器操作数)

（3）标志位操作指令

LAHF：

将SF\ZF\AF\PF\CF依次对应AH的第7、6、4、2、0位，送入AH中。

（对标志位本身无影响）

SAHF：

作用与LAHF相反，并改变标志位。

（4）PUSHF POPF

PUSHF（POPF）：

将现有状态压入（弹出）栈，PUSHF不影响标志位，POPF改变标志位。

**2、算术运算指令**

**（1）加法运算指令**

**普通加法指令ADD**

**注意点：**

1. **对六个标志位都产生影响；**
2. **不能对段寄存器进行运算，不能同时为存储器操作数。**

**带进位加法指令ADC**

**注意点：使用前要先将CF清零。**

**自加加法指令INC**

**注意点：**

1. **常用于修改地址指针；**
2. **除了不影响CF标志位，其余标志位依旧受其影响。**

**（2）减法指令**

**SUB对应ADD，SBB对应ADC，DEC对应INC，要求与对应计算方法皆相同，此处不再做累述。**

**求补指令NEG**

**注意点：**

**其真实操作过程即用0减去操作数，该过程相当于对该操作数求补码。**

**0000 H-1234 H=EDCC H**

**00 H-FB H=0000 0000B-1111 1011B=0000 0101B=05H**

**比较指令CMP**

**其本质是源操作数-目标操作数，但不影响操作数本身，仅影响标志位。**

**注意点：**

1. **若是无符号数比较，只看CF；**

**跳转的判断条件**

**JNC/JC**

**若 AX > BX CF=0**

**若 AX < BX CF=1**

1. **若是有符号数比较，由OF和SF同时决定；**

**跳转的判断条件**

**JNL/JL或JGE/JNGE**

OF和SF状态相同 AX > BX

OF和SF状态不同 AX < BX

1. **ZF=1，二数相同；ZF=0，二数不同。**

**（3）乘法指令**

**无符号数乘法指令：MUL**

**注意点：**

1. **该操作指令是隐含操作数，字节数隐藏AX，16位数隐藏DX:AX；**
2. **两操作数字长必须相等（见杂记），不能是立即数。**

**带符号数乘法指令：IMUL**

**注意点：**

1. **两数皆是有符号数；**
2. **若乘积的高半部分是第半部分的符号位的扩展，CF=OF=0，，否则CF=OF=1。**

若将两个寄存器中内容看做无符号数，则应使用指令：MUL CL指令执行后（AX）=10DEH，因AH中的结果不为零，故CF=OF=1。

若将两操作数看做有符号数，则二者相乘应采用有符号数的乘法指令，即：IMUL CL。指令执行后AX=FFDEH=-34，则AH中内容为AL中的符号扩展，故CF=OF=0.

（4）除法指令

无符号除法指令：DIV

注意点：

1. 被除数的字长必须为除数字长的2倍，若不够，则用字位扩展指令来扩展其位数；
2. **除法指令不影响标志位。**
3. **字节操作：AL存放商，AH存放余数；**
4. **字操作：AX存放商，DX存放余数。**

**有符号除法指令与之类似，不再累述。**

**（5）BCD码运算调整指令**

|  |  |  |
| --- | --- | --- |
| **是否压缩 算法类型** | **压缩** | **非压缩** |
| **加法** | **DAA** | **AAA** |
| **减法** | **DAS** | **AAS** |
| **乘法** | **AAM** |  |
| **除法** | **AAD** |  |

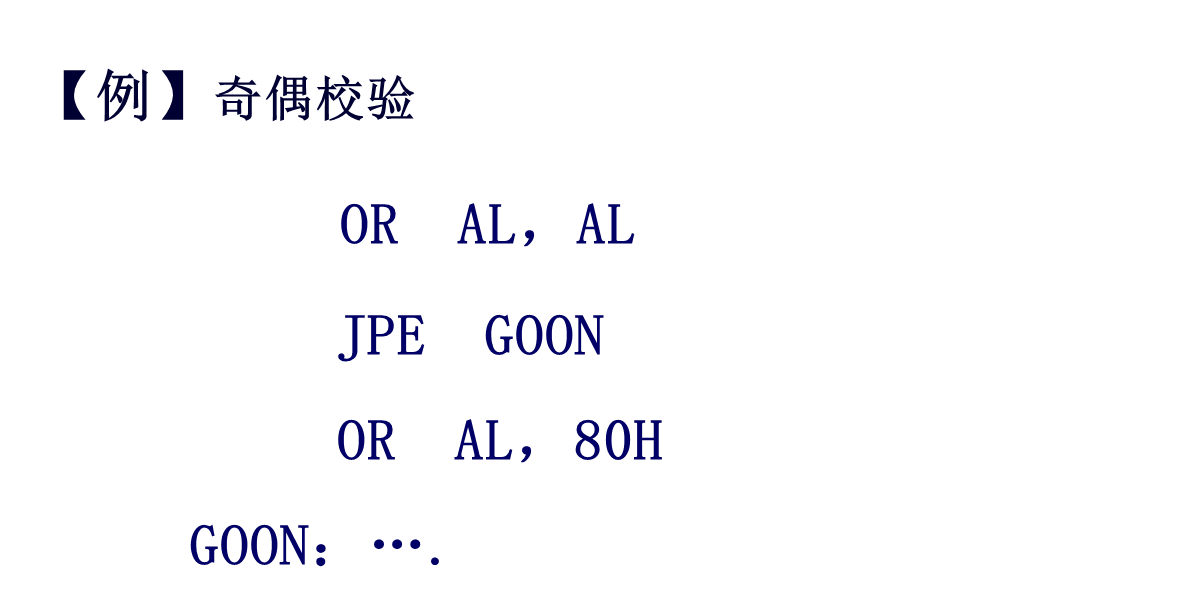
**注意点：**

1. BCD调整指令不能单独使用，必须跟在算术运算指令之后（除法例外）;
2. 均为隐含寻址方式，隐含的操作数是AL或AL、AH。

**3、逻辑运算与位移**

**（1）与(或)指令AND（OR）**

**注意点：**

1. **当且仅当二者都是“1”（“0”），才为“1”（“0“）；一方是“0”（“1“），全是“0”（“1“）；**
2. **目标操作数不能是立即数；**
3. **可实现某些位保持不变，某些位清零（置一）的作用；**
4. **AND(OR) AX,AX不改变操作数，使CF和OF清零。**

**（2）非指令NOT**

**注意点：**

1. **操作数取反送回原地址；**
2. **操作数不能是立即数；**
3. **对标志位无影响。**

**（3）异或指令XOR**

**特殊用法：将寄存器自身异或，可使寄存器清零。**

**（4）测试指令TEST**

**注意点：执行AND的过程，但结果不送回，只影响标志位，通常用于测试某些位的状态。**

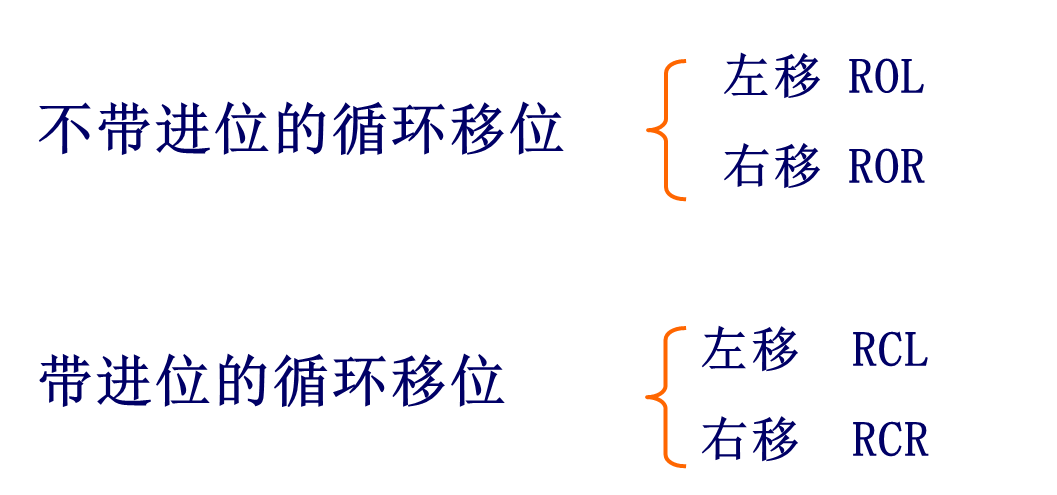
**（5）非循环移位指令**

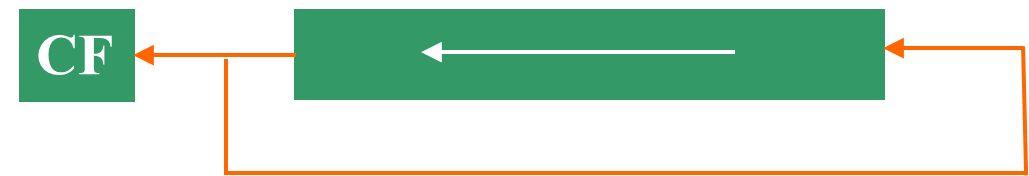
**逻辑左(右)移SHL(SHR)**

**算术左(右)移SAL(SAR)**

**注意点：**

1. **左移二进制乘法，右移二进制除法；**
2. **单次移位，操作数为1，多次移位，将操作数存入cl，再移位；**
3. OF=1对SHL指令不表示左移后溢出，而对SAL指令表示移位后超出了符号数的表示范围。
4. **执行SHR时，高位补零；执行SAR时，高位保持不变。**

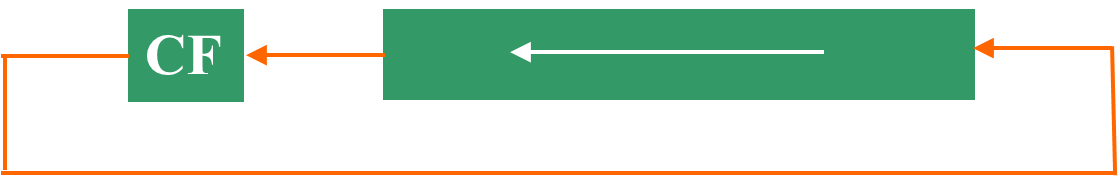
**（6）循环移位指令**



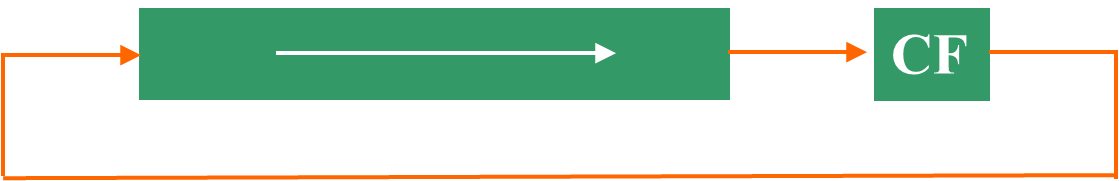
ROL



ROR



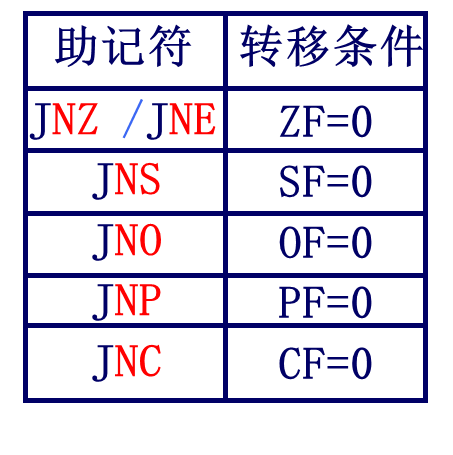
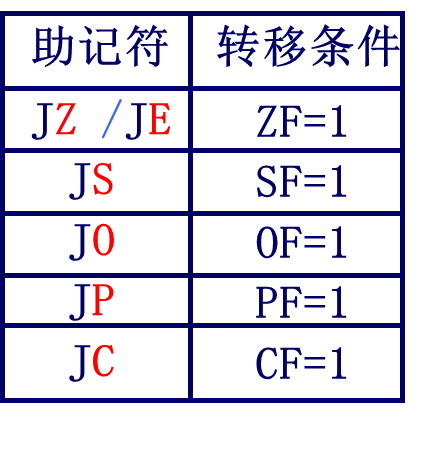
RCL

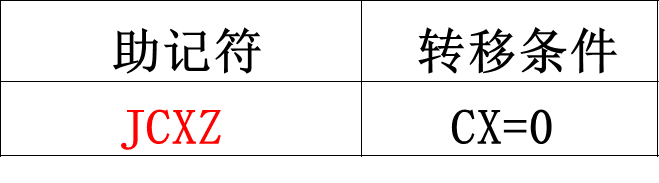


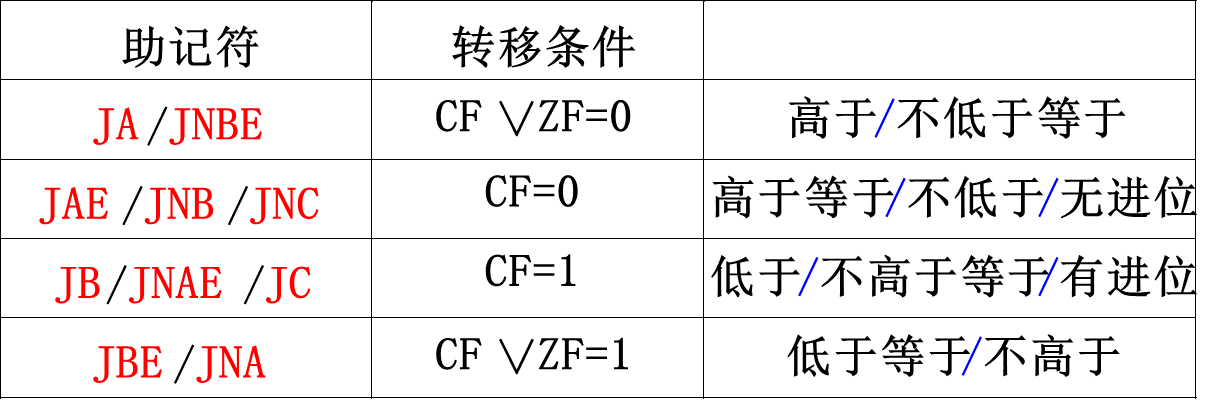
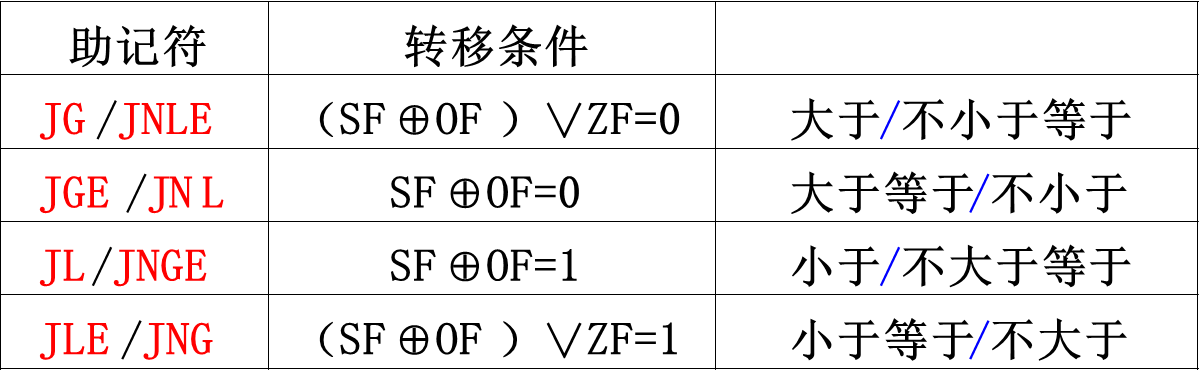
RCR

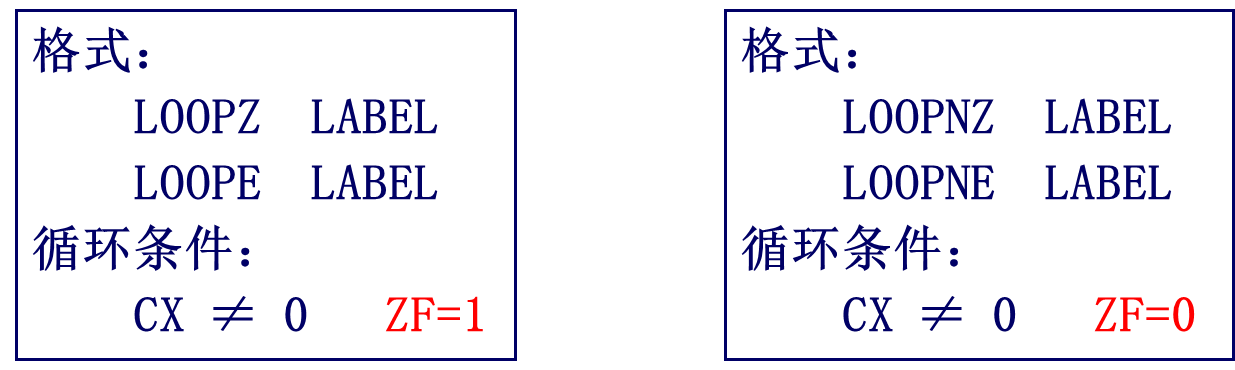
**4、程序控制指令**

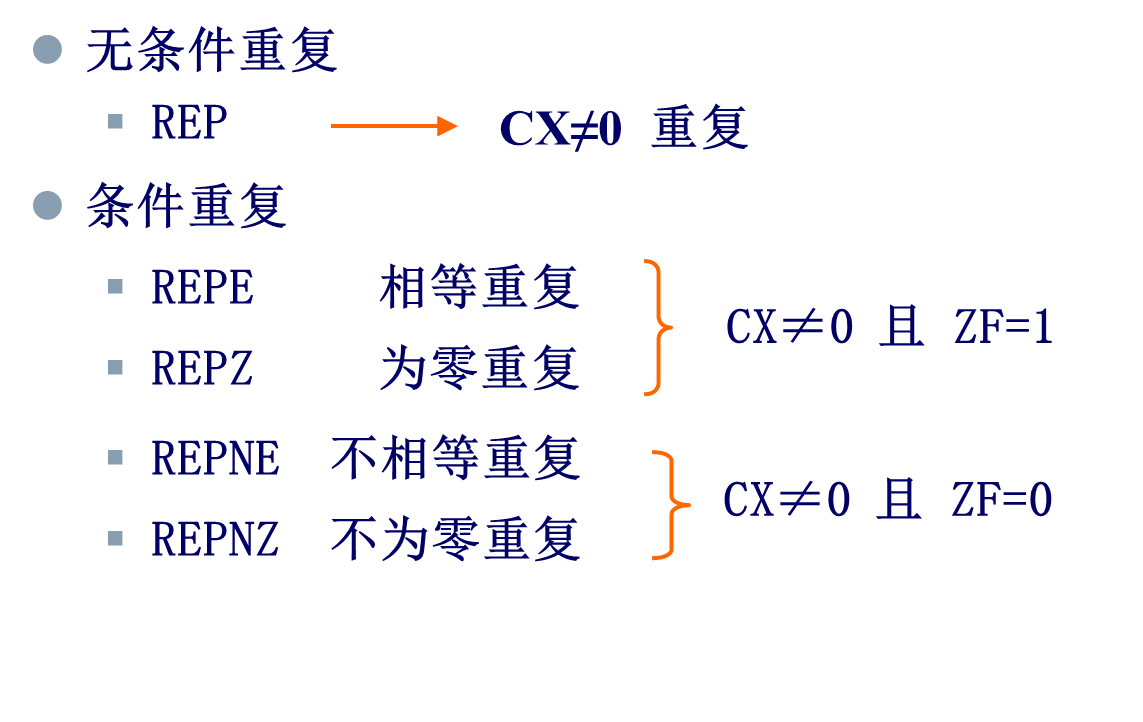
**转移指令**

** A.以单个状态标志作为转移条件**

** B.以CX的值为0作为转移条件**

** C.以两个无符号数比较的结果作为转移条件**

** D.条件循环**

**5、串操作指令**

**存放及改变方向**

**源串存放在DS，偏移地址由SI指定；**

**目标串必须在ES，偏移地址由DI指定；**

**DF=0，往高地址方向；**

**DF=1，往低地址方向。**

**串传送指令**

**MOVSB(W) 地址取默认值**

**REP MOVSB(W) 重复将DS:SI内的内容传送到ES:DI，传送次数根据CX内的值来决定。**

**串比较指令**

**CMPS OPRD1,OPRD2**

**CMPSB(W) 可用于检查两处数据的传送是否出现不同。**

**串扫描指令**

**SCAS PROD**

**SCASB(W) 可用于搜索一段存储内与AL(AX)内容相同的数据**

**例：**

**MOV AL,$**

**REPNZ SCASB ;没找到$就继续循环**

**串装入指令**

**LODS OPRD**

**LODSB(W) 将DS:SI的字（节）传送到AL(AX)中去，可以用于之后传送到DL处，在显示器上显示。**

**串存储指令**

**STOS OPRD**

**STOSB(W) 将AL(AX)传送到ES:DI处，可将某个区域内的值置为相同。**

# 杂记

**一、状态标志**

1、进位标志CF

NC：0，表示无进位

CY：1，表示有进位

2、溢出标志OF

NV：0，表示无进位

CY：1，表示有进位

3、符号标志SF

PL：0，表示正数

NG：1，表示负数

1. 零位标志ZF

NZ：0，表示结果不为零

ZR：1，表示结果为零

1. 奇偶标志PF

PO：0，表示结果中最低字节里为1的个数为奇数

PE：1，表示结果中最低字节里为1的个数为偶数

**二、算法指令**

1、两数相乘，如果字节数不等，并不影响运行，譬如：

**MOV AX,1025H**

**MOV BL,03H**

**MUL BL**

在该过程中，执行的是BL\*AL，即25H\*03H，执行完毕后，AL\*BL->AX，因此，将006F直接填入AX。

如果是MUL BX，那么结果就是306F。

2、两数相除，商存低位，余数存高位

**三、汇编语言程序设计**

1、OFFSET 取后面的偏移地址

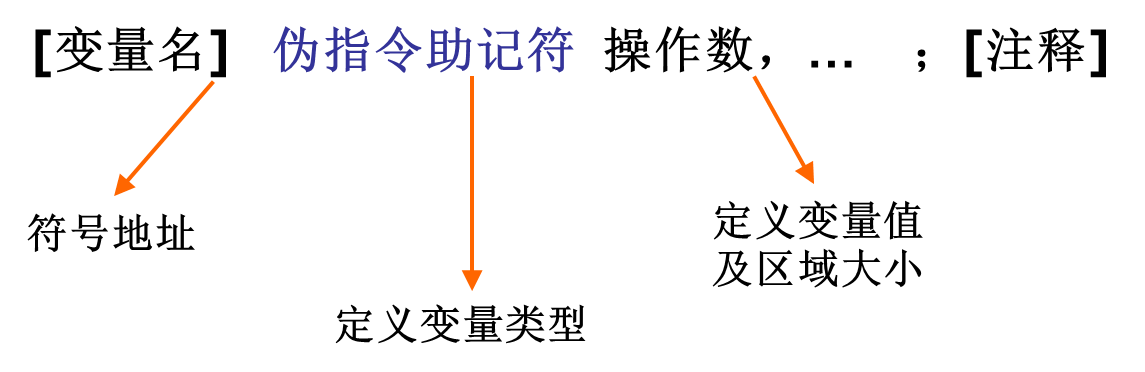
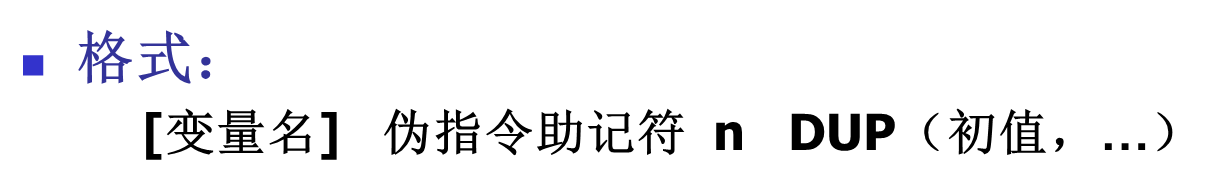
2、SEG 取后面的段地址

3、TYPE：取变量的类型

LENGTH：取定变量的区间长度

SIZE：取所定变量的区间的字节数

SIZE=TYPE \* LENGTH

1. DB 字节型定义 AL,AH
2. DW 字型定义 AX,BX
3. DUP 将某个数据填入某个地方N次
4. 单字符输入：01H，输入内容存放在AL中。
5. 单字符显示输出：02H，将DL 中的字符放在显示器上显示。
6. 0DH,0AH,'$'：回车、换行、字符串结束符，通常用于字符串的输出。
7. 字符串输入：0AH
8. 定义要输入的字符串的缓存空间

A1 DB 30 DUP(“$”)

1. 格式控制

MOV AH,0AH

LEA DX,A1

INT 21H

此时即可输入字符串，并在显示器上保留。

1. 字符串输出：09H，同单字符显示。
2. 返回操作系统：4CH
3. 以上指令均存入AL中。

# 附录

1. **寄存器操作数的表达**

r8——任意一个8位通用寄存器

AH AL BH BL CH CL DH DL

r16——任意一个16位通用寄存器

AX BX CX DX SI DI BP SP

reg——代表r8或r16

seg——段寄存器

CS DS ES SS

1. **存储器操作数的表达**

m8——一个8位存储器操作数单元（所有主存寻址方式）

m16——一个16位存储器操作数单元（所有主存寻址方式）

mem——代表m8或m16

**3、立即数的表达**

i8——一个8位立即数

i16——一个16位立即数

imm——代表i8或i16

dest——目的操作数

src——源操作数