|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 指令格式 | 解释 | 操作数要求 | 标志位影响 | 其他 |
| 传送类指令 | 基本不影响 | 标志位 |  |  |
| Mov d，s |  | **所有双操作数指令都只能有一个内存操作数**；mov指令不能对段寄存器传送立即数，不能段寄存器之间传送，且两个操作数必须类型相同。 | 无 |  |
| Xchg d，s |  | 不能立即数和段寄存器，最多一个内存操作数 | 无 |  |
| Lahf | 将标志寄存器低8位传到ah保存 | 无操作数指令，隐含ah 标志寄存器 | 无 | 用于暂时保护标志寄存器的低8位 |
| Sahf | 将ah传动到标志寄存器低8位 | 同上 | 影响标志寄存器低8位，  Sf zf af pf cf | 将先前保存在ah的标志位恢复到标志寄存器 |
| Push s | Sp-2后将源操作数传送到新栈顶 | 单操作数寻址，必须为16位寄存器或内存单元，隐含指向的栈顶内存单元作为目的操作数 | 无 | 堆栈元素都是两字节的 |
| Pop s | 先把栈顶传送到目的，再sp+2 | 目的操作数只能时字类型（16位） | 无 | 入栈出栈操作遵循后进先出 |
| Pushf | 先sp-2，再将16位标志寄存器数据传送栈顶 | 隐含标志寄存器和栈顶内存单元 | 无 | 暂时保护标志寄存器 |
| Popf | 和上面相反 | 隐含二者同上 | 所有标志位 | 恢复 |
| Lea d，s | 将源操作数有效地址（ea）传到目的操作数 | 源操作数只能是内存单元（只有它有有效地址）目的操作数只能是16位通用寄存器 | 无 |  |
| Lds d，s  Les d，s | 与lea有所区别，将内存单元中的数据内容作为地址，把该地址和该地址＋2的数据内容存到目的操作数和ds或es |  | 无 | **详见书p59** |
| 算术运算类 | 指令 |  |  |  |
| Add d，s | 两者相加结果保存到d | 双操，操作数类型可字节，可字，源操可立即数 | 所有状态标志位 | “半加” |
| Adc d，s | Cf加在最低位 | 双操，操作数类型可字节，可字，源操可立即数 | 所有状态标志位 | 全加  长操作数 |
| Inc d | 加一 | 单操，不能立即数  可字/字节的通用寄存器/内存单元 | Sf zf af pf  Of（只有从01111111变1000000置1，其余均0） | 计数，下标 |
| Sub d，s | d-s | 双操，操作数类型可字节，可字，源操可立即数 | 所有状态标志位  **判断cf，要用减法来判断，不能用补码加法来判断** | 和加法一样有二义性，不确定是无符号数运算还是补码运算 |
| Sbb d，s | 最低位-cf | 双操，操作数类型可字节，可字，源操可立即数 | 所有状态标志位  **判断cf，要用减法来判断，不能用补码加法来判断** |  |
| Dec d | 减一 | 单操，不能立即数  可字/字节的通用寄存器/内存单元 | Sf zf af pf Of **（无cf）** |  |
| Neg d | 1. d   求相反数 | 操作数常解释为补码，**单操，不能立即数**，可字/字节的通用寄存器/内存单元（**不超过16位** | 所有状态标志位 | 注意溢出  -128  -32768 |
| Cmp d，s | d-s  同sub但不保存结果 |  | 所有状态标志位 |  |

无符号数：

Cf=0 d>=s

Cf=1 d<s

补码：

Of=0 sf=0 d>s

Of=1 sf=1 d>=s

Of=0 sf=1 d<s

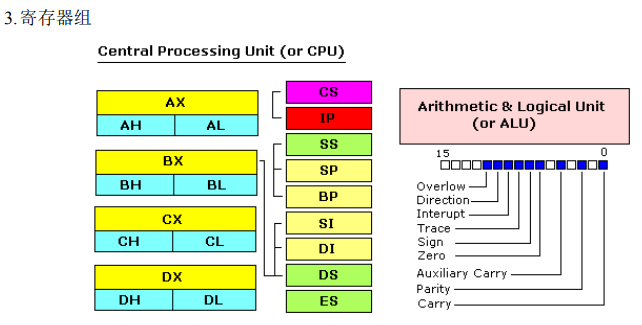
Of=1 sf=0 d<s

综上，sf=of则d>=s否则d<s

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 位操作类指 | 令 | 以二进制位为单位 | 来操作数据 |  |
| And d，s | 按位与 |  | **Cf=of=0**  Af不确定  Sf zf pf 按算术指令解释 | 取位模板（位清零），需要则置一，不需置零 |
| Or d，s |  |  | **Cf=of=0**  Af不确定  Sf zf pf 按算术指令解释 | 置位模板，要则放1，其余放0不影响 |
| Xor d，s |  |  | **Cf=of=0**  Af不确定  Sf zf pf 按算术指令解释 | 位变反操作，要变反则设1，保持不变则设0 |
| Not d | 按位取反 | 单操 | **不影响任何标志位** |  |
| Test d，s | 类似cmp |  | **Cf=of=0**  Af不确定  Sf zf pf 按算术指令解释 | 通过zf判断所分离二进制位取值**P68** |
| Sal d，c | 算术左移  最后移出入cf，右补0 | 操作数认为补码 | 所有状态标志位，af不确定。**Count=1时，of才有意义** | 相当于乘以2 |
| Sar d，c | 算术右移  最后移出入cf，左补符号位 | 操作数认为补码 | 所有状态标志位，af不确定。**Of始终为0** | 补码除以2 |
| Shl d，c | 逻辑左移  最后移出入cf，右补0 | 针对无符号数 | 所有状态标志位，af不确定。**Count=1时，of才有意义** |  |
| Shr d，c | 逻辑右移  最后移出入cf，左补0 | 针对无符号数 | 所有状态标志位，af不确定。**Count=1时，of才有意义** |  |
| Rol d，c | 循环左移  Cf被赋从左移出的最后一位，此前按原顺序补右边 |  | 仅**cf** （最后移出的）count=1，**of**才有意义，移位前后最高位无变化则1 |  |
| Ror d，c | 循环右移  Cf被赋从右移出的最后一位，此前按原顺序补左边 |  | 仅**cf** （最后移出的）count=1，**of**才有意义，移位前后最高位无变化则1 |  |
| Rcl d，c | 带进位循环左移，cf看作最高位整体最后移出入cf，右补0 |  | **Cf**作为最高位  count=1，**of**才有意义，移位前后最高位（不是cf）无变化则1 |  |
| Rcr d，c | 带进位循环右移，cf看作最低位，整体最后移出入cf，左补符号位 |  | **Cf**作为最高位  count=1，**of**才有意义，移位前后最高位（不是cf）无变化则1 |  |
| 处理器控制 | 类指令 | 多为无操作数 | 指令 |  |
| CLC | Clear carry flag |  | Cf=0 | 常用于配合adc实现长操作数加法 |
| STC | Set carry flag |  | Cf=1 |  |
| CMC | Complement carry flag |  | Cf取反 |  |
| CLD | Clear direction flag |  | Df=0 |  |
| STD | Set direction flag |  | Df=1 |  |
| CLI | Clear interrupt-enable flag |  | If=0 |  |
| STI | Set interrupt-enable flag |  | If=1 |  |
| HLT | Cpu时钟信号输入切断，无法再继续执行所有时序过程 | 仅能通过reset复位信号使系统重新启动，才能恢复cpu |  |  |
| NOP | 无操作，空转三个节拍 |  |  | 可以用于延时等待 |

**地址加法器**

**寄存器组**



**标志位**

①进位标志位CF

**最高位**（字第十五位，字节第七位）进位或借位1，否则0；

无符号数；

②奇偶标志位PF

**低八位**偶数个1为1，奇数个1为0

③辅助进位标志位AF

第三位产生进位或借位为1，否则为0

无符号数；

④零值标志位ZF

结果为0为1；否则为0

⑤符号标志位SF

同运算结果最高位一致，1负0正

补码；

⑥溢出标志位OF

溢出1，否则0；

补码；

⑦单步跟踪标志位TF

单步模式1，连续执行0

⑧中断使能标志位IF

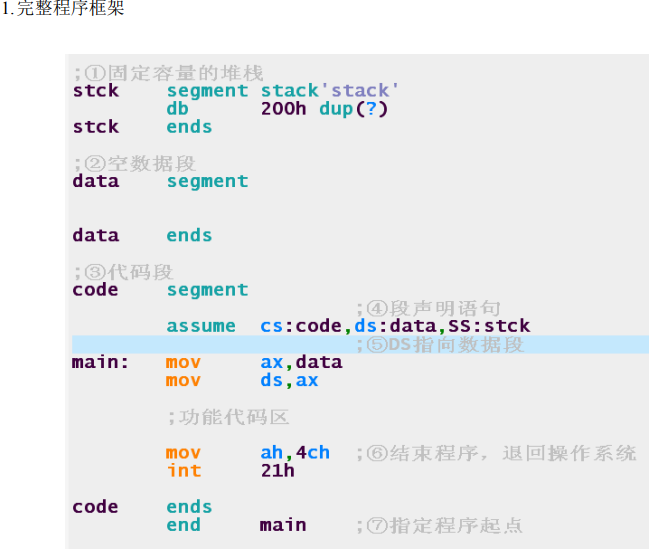
允许中断1

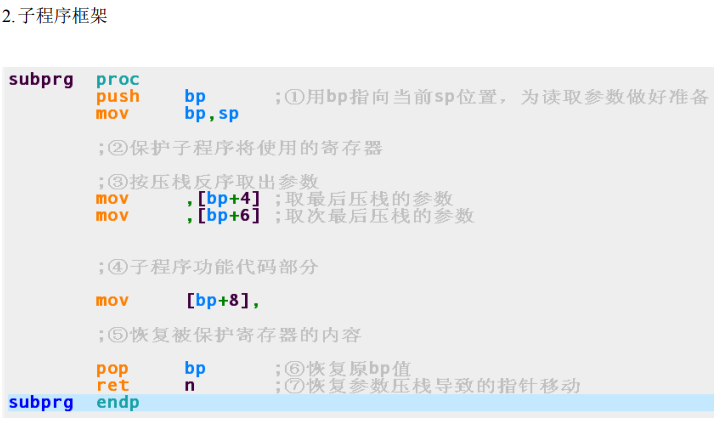
⑨方向标志位DF

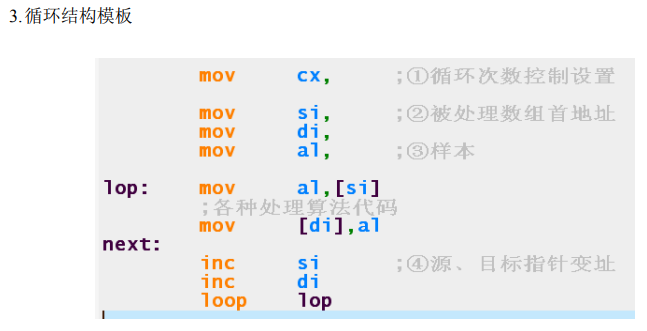
1地址递减0地址递增

**寄存器寻址方式**

**Chapter 6**







**1.常用debug指令**

**2.常量和变量定义语句**：

① EQU、=：

编译程序将符号常量定义语句的定义值作为立即数，取代源程序中所有其他位置的该符号常量；

② DB、DW、DD 及 DUP：

编译程序根据各变量的排列顺序和所占的字节数，计算出各变量的偏移量，以直接寻址方式或作为其他寻址方式中的位移量，取代源程序中所有其他位置的该变量符号。

3.其他伪指令

① $、ORG；

② SEG、OFFSET；

③ NEAR、FAR；BYTE、WORD、DWORD；

④ LABEL、THIS、TYPE、PTR

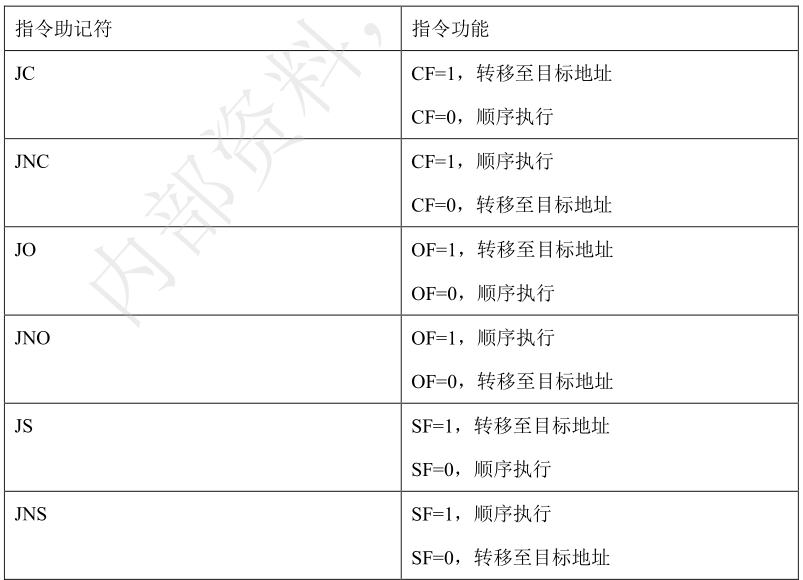
**一．条件转移指令**

1.无条件转移

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| JMP | 分为段内段间的直接和间接转移指令 | 标号或者16位的通用寄存器名称或者字类型的内存单元（段内间接转移） |  |  |

2.条件转移

（1）单标志位转移





（2）JCXZ指令 若（CX）=0，则转移到目标地址，（CX）不为0，则顺序执行。**在8086/8088指令系统中，该指令是唯一不以标志位为判断条件的条件转移指令**

（3）无符号数条件跳转指令 将最近一条执行的CMP（或者SUB）指令中的两个操作数解释为无符号数编码，以CMP指令所影响的CF、ZF标志位来综合判断被减数与减数的大小关系，从而根据判断结果来决定是否实施流程转移。**将CMP指令的操作数解释为无符号数**



（4）符号数条件跳转指令 **将CMP指令的操作数解释为补码**



二．**循环控制指令**

1. LOOP指令

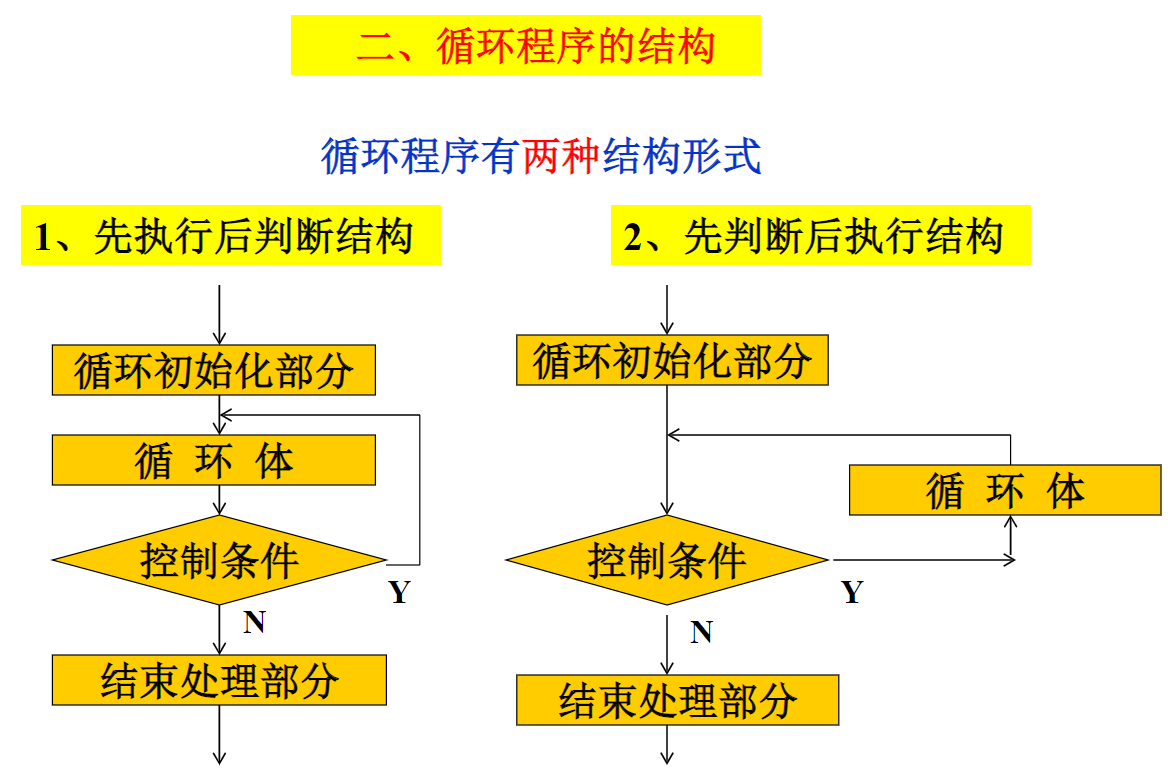
指令功能：首先（CX）-1=>CX，这个减法不影响标志位，然后如果（CX）不为0，则跳转至目标地址，否则顺序执行

1. LOOPZ/LOOPE

(CX)!=0并且ZF=1，则转移至目标地址

1. LOOPNZ/LOOPNE

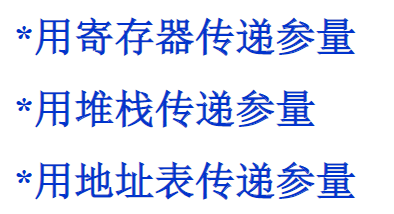
(CX)!=0并且ZF=0，则转移至目标地址



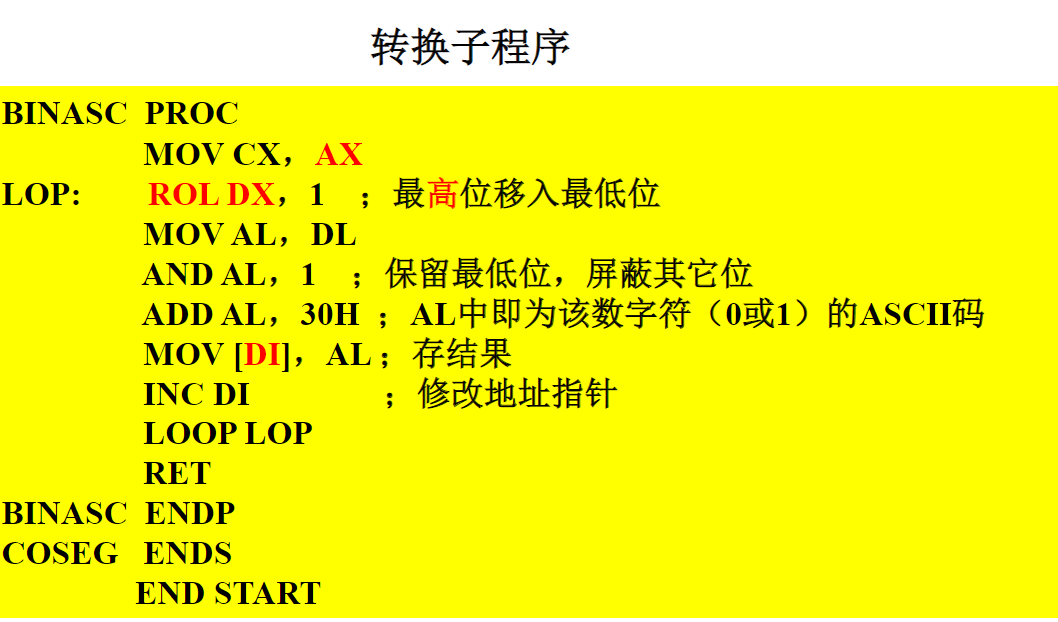
**C8**，

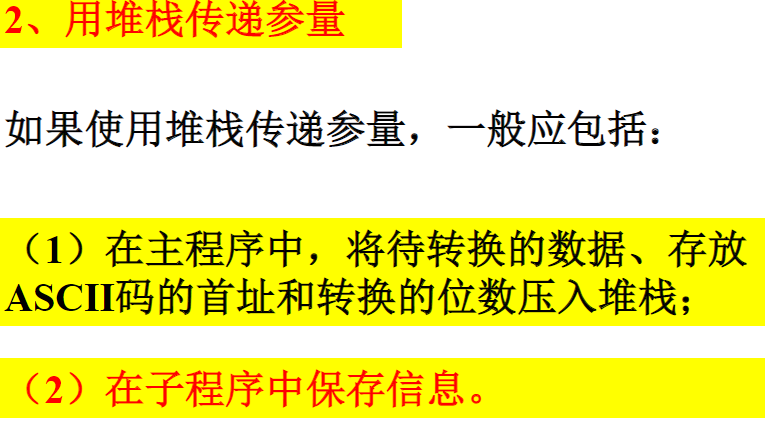
**系统调用与返回**

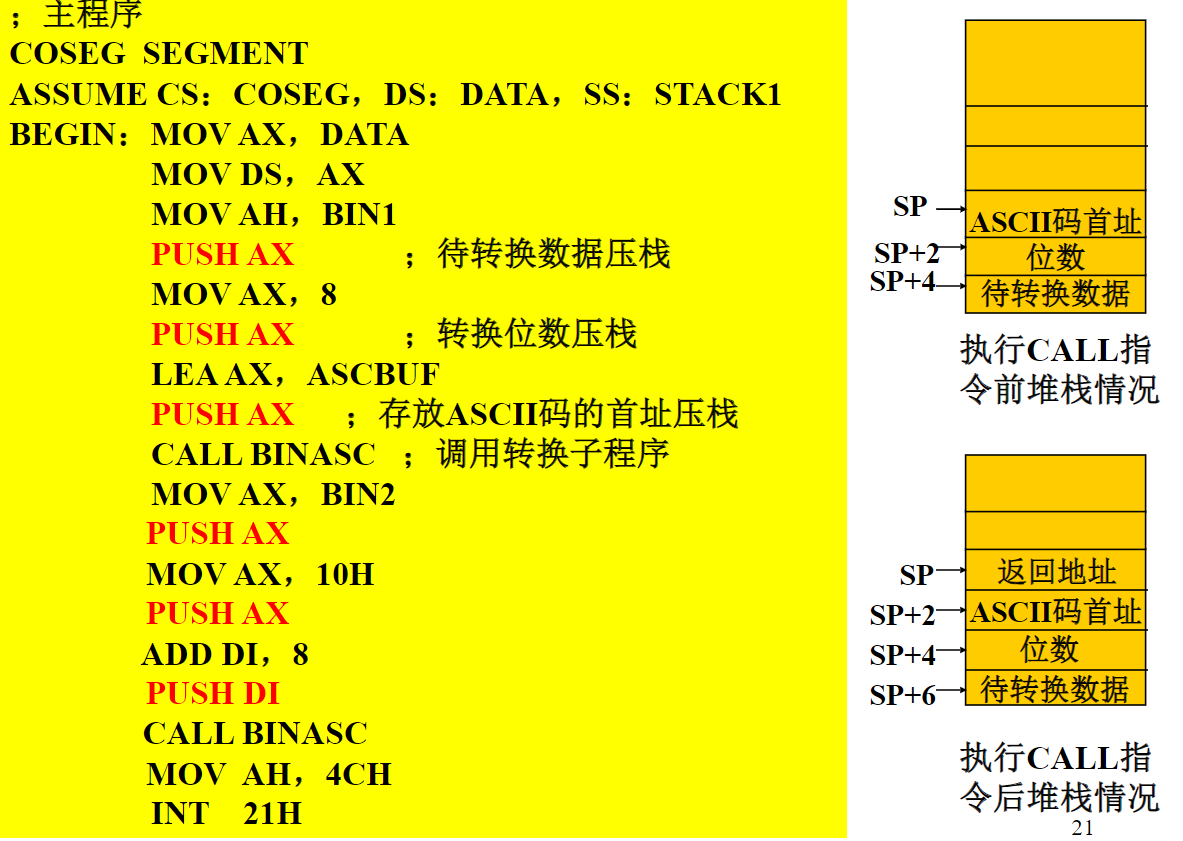
**参数传递**

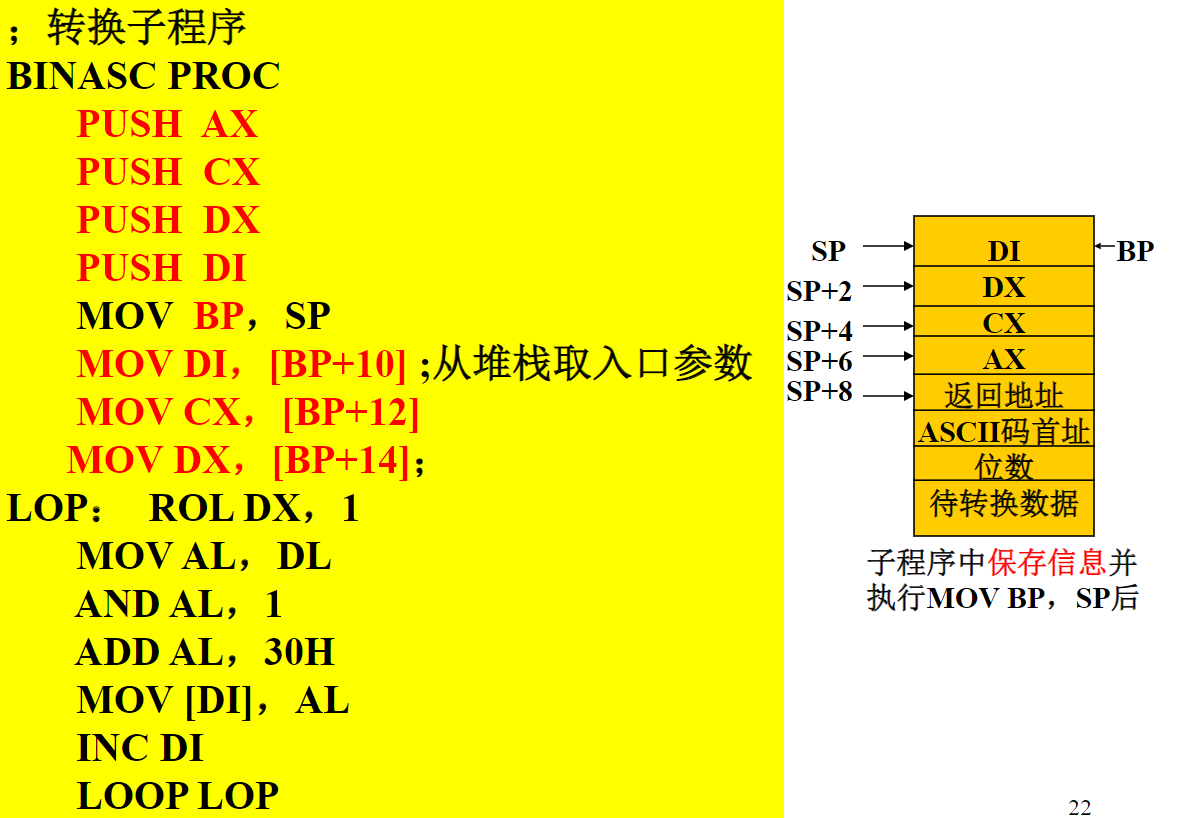


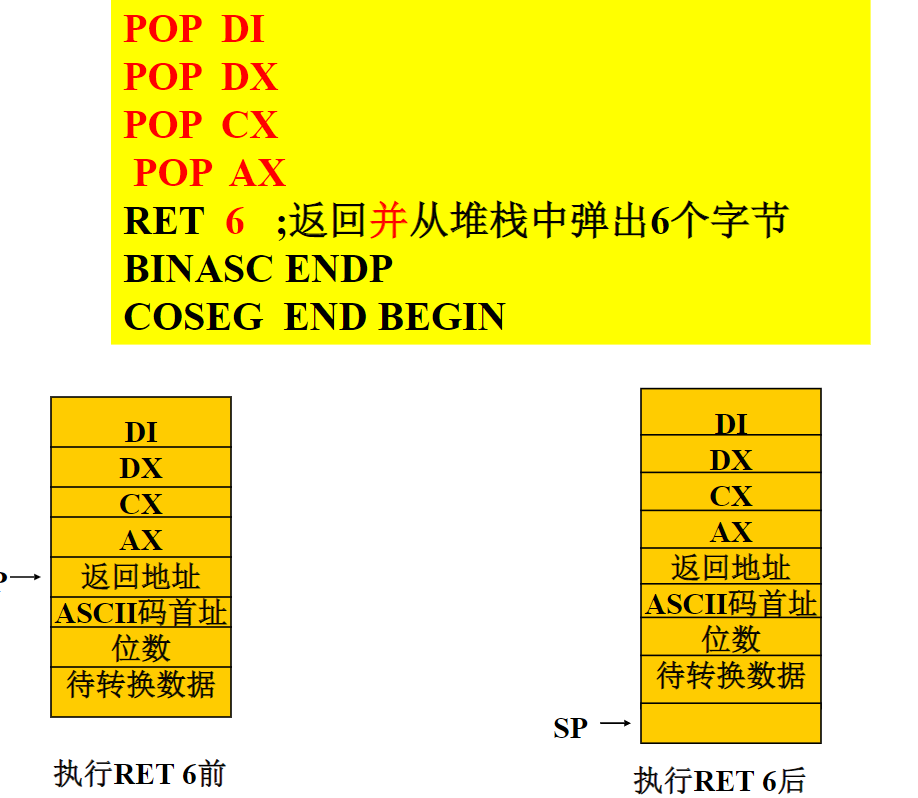


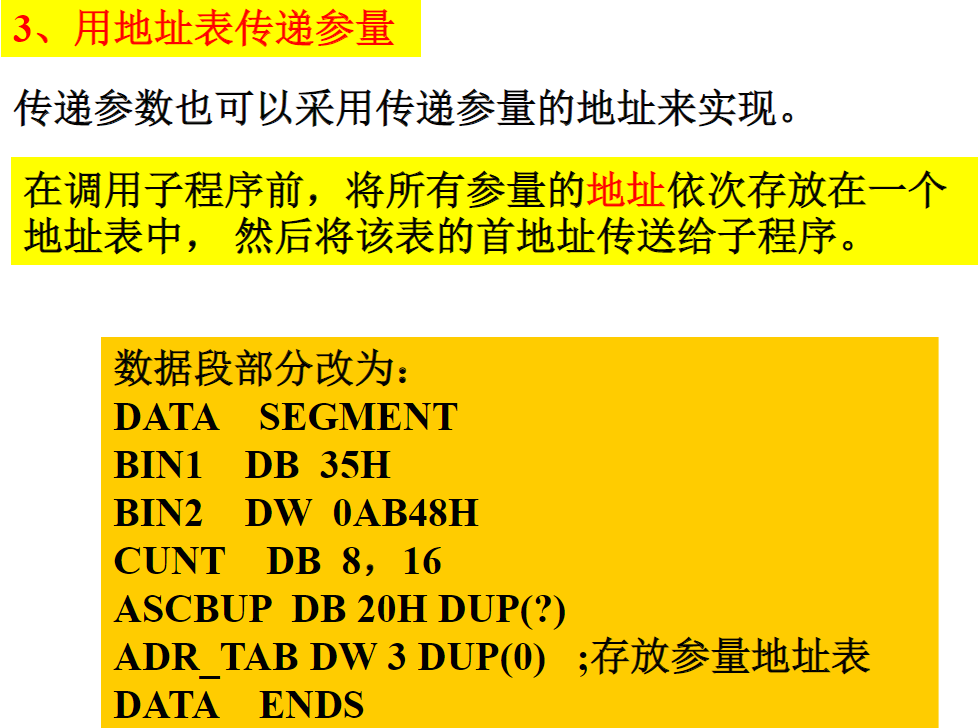


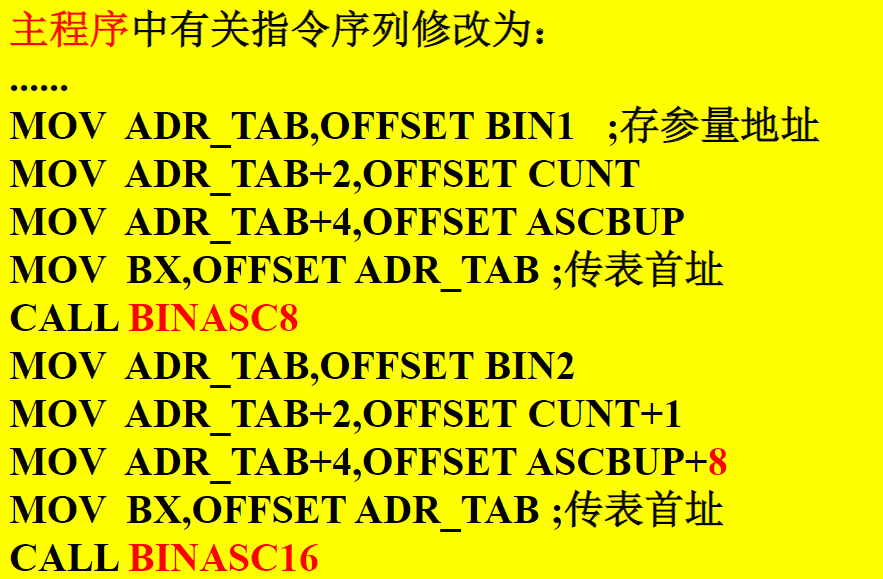






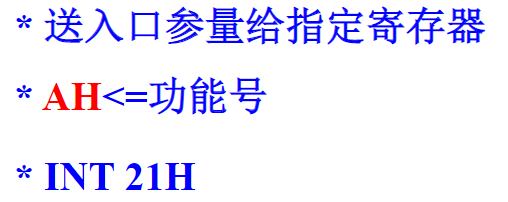








**DOS功能调用**



1. 单个字符输入

MOV AH,01H

INT 21H

功能，接受一个键盘输入字符，并保存到AL寄存器中。

入口参数：无 出口参数：（AL）=输入字符的ASCII码

1. 单个字符输出

MOV AH，02H

INT 21H

功能，在屏幕光标位置显示（DL）指示的字符

入口参数：（DL）=输出字符的ASCII码 出口参数：无

1. 字符串输入

MOV AH，0AH

MOV DX,缓冲区起始偏移量

INT 21H

功能，接受接盘字符串输入，输入的字符串保存在（DS）：（DX）+2起始的缓冲区，（DS）：（DX）指示的字节单元为入口参数，含义为字符串的最大字符数，（DS）：（DX）+1为实际从键盘接收的字符数。

入口参数 （DS）：（DX）=缓冲区首字节地址，（DS）：（DX）=最大字符数

出口参数：（（DS）：（DX）+1）=实际输入字符数

1. 字符串输出

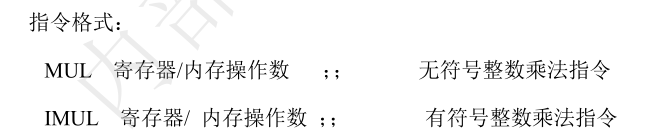
MOV AH,09H

MOV DX,字符串起始偏移量

INT 21H

功能：在屏幕显示逻辑地址（DS）:（DX）指向的字符串，直到遇见“$”结束。字符串必须以**“$”结尾**。

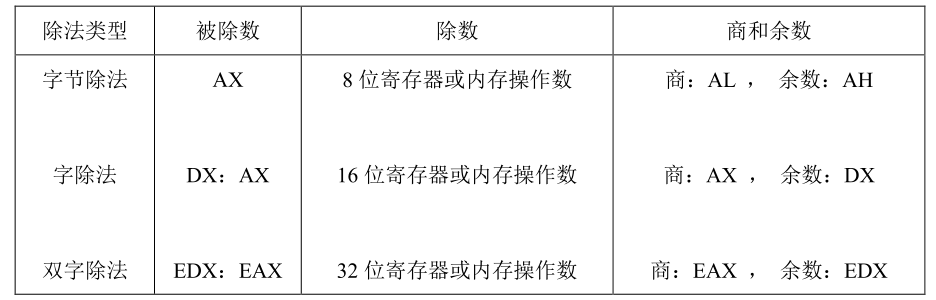
**Chapter9乘法**





**除法运算指令**

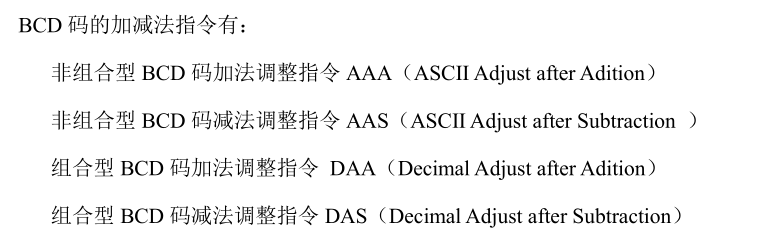




BCD码，用4个二进制位表示一个十进制数

非组合型BCD码，占用一个字节的低4位，高四位用0填充

组合型BCD码，由两位BCD码组成，占用一个字节，一个BCD码占用字节的低四位，一个BCD码占用高四位。



都是在加法指令（ADD，ADC）和减法指令（SUB，SBB）之后，对存放在AL中的加法减法结果进行调整