



# 第三章 8086的指令系统

- ◆ 8086指令特点
- ◆ 8086的寻址方式
- ◆ 8086的指令格式及数据类型
- ◆ 8086的指令集





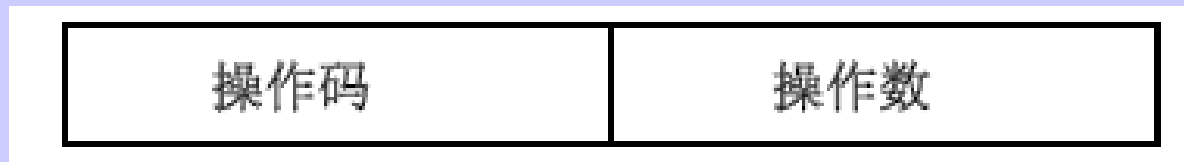
## 3.1 8086指令特点

- ◆ 1. 8086具有灵活的指令格式
- ◆ 2. 8086有较强的运算指令
- ◆ 3. 8086指令有极强的寻址能力
- ◆ 4. 8086指令有处理多种数据能力





### ■ 3. 2 8086的寻址方式



寻址方式就是指令中用于说明操作数所在地址的方法

**MOV AX, 1234H ;B8 34 12**

**MOV AX, [1234H] ;A1 34 12**





# 一、 8086的寻址方式说明

## 1. 有效地址EA (Effective Address)

当操作数在内存中时，指令的地址码（操作码）给出所访问的内存单元的逻辑地址。在寻址方式中，逻辑地址的形成是由多个分量组合而成，该组合地址又叫有效地址。







## 2、MOV数据传送指令

其格式为：

**MOV** 目的操作数，源操作数

- ◆ 目的操作数和源操作数均可采用不同的寻址方式，
- ◆ 两个操作数的类型必需一致。





## 二、寻址方式介绍

### 1. 立即寻址 (Immediate addressing)

操作数就在指令中，紧跟在操作码后面，作为指令一部分存放在内存的代码段中，这种操作数称为立即数。

例：

**MOV AX, 34EAH    B8 EA 34**

**MOV BL, 20H      B3 20**





## 2、寄存器寻址

### (Register addressing)

操作数在寄存器中，指令中源操作数和目的操作数都可用这种寻址方式。

例：

**MOV AL , BL** **88 D8**

**MOV AX , 1234H** **B8 34 12**

**MOV AL , AH** **88 E0**





### 3、直接寻址（Direct addressing）

当指令中的源操作数或目的操作数，采用直接给出被访问内存单元的逻辑地址时，这种寻址方式称直接寻址。

例：

**MOV AX , [3E4CH]      A1 4C 3E**

**MOV [1234H] , AL      A2 34 12**



## 4、寄存器间接寻址（**Register indirect addressing**）

内存单元的**逻辑偏移地址**通过寄存器间接给出。

例：

**MOV SI, 61A8H**

**MOV DX, [SI]**





## 5、基址/变址寻址 (Based/Indexed addressing)

这种寻址方式中提出位移量的概念，即在寄存器间接寻址给出的偏移地址上，加一相对位移量。位移量是一带符号的16位16进制数。当使用BX或BP寄存器时，称基址寻址；使用SI或DI寄存器时，称变址寻址。

例：

**MOV CX, 36H[BX]**

**MOV -20[BP], AL**





## 6、基址加变址寻址 (Based Indexed addressing)

它的EA是由三部分组成的，基址寄存器BX或BP的内容加上变址寄存器的内容再加位移量。物理地址由基址寄存器按规则选择段寄存器，也可以使用段超越。

例：

**MOV AX, 8AH[BX][SI]**

该例中  $EA = 8AH + BX + SI$

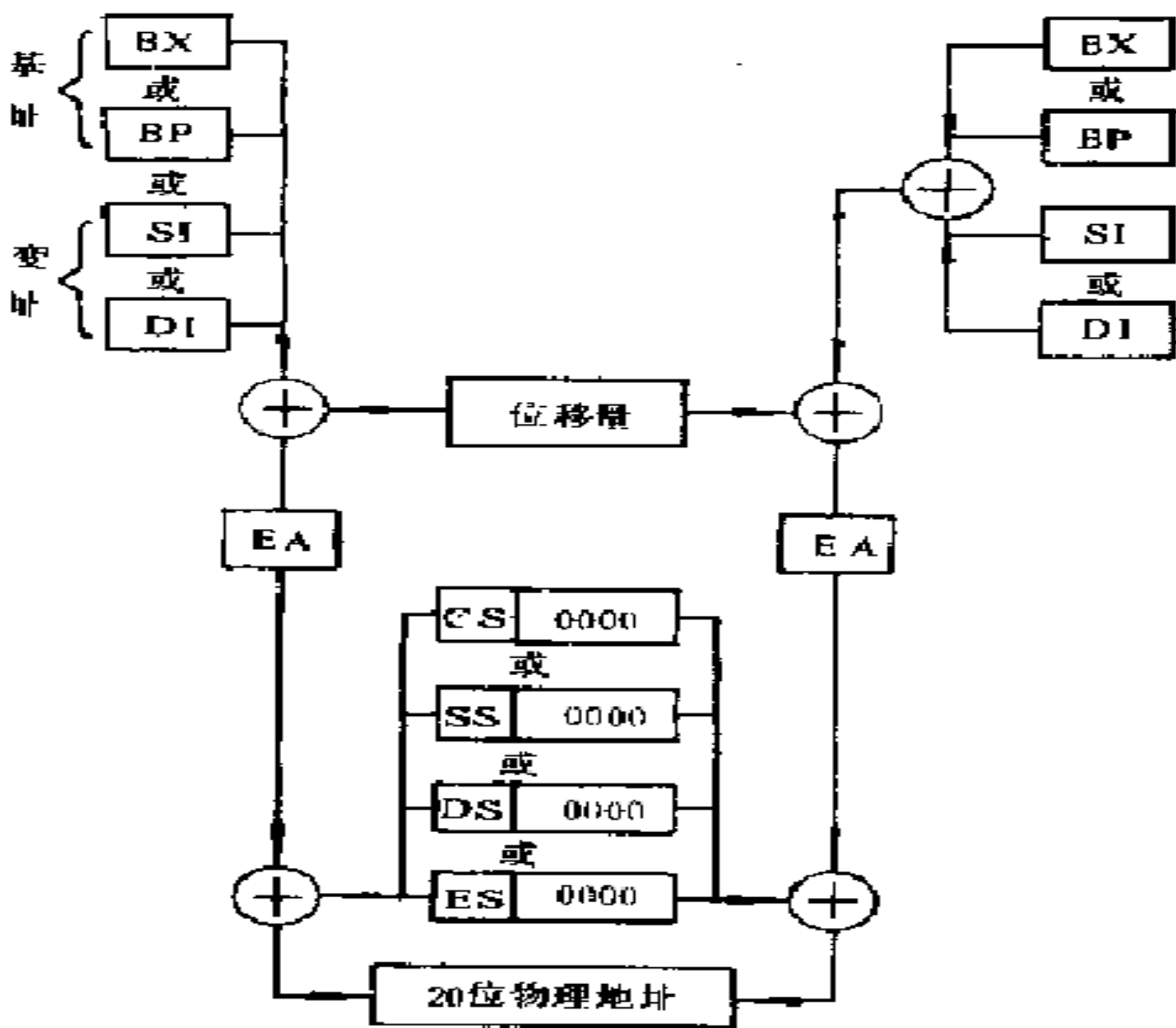
物理地址 =  $DS \times 10H + 8AH + BX + SI$





存储器存取方式	约定段	可超越使用的段	偏移量
取指令	<b>CS</b>	无	<b>IP</b>
堆栈操作	<b>SS</b>	无	<b>SP</b>
源字符串	<b>DS</b>	<b>CS, ES, SS</b>	<b>SI</b>
目的字符串	<b>ES</b>	无	<b>DI</b>
用 <b>BP</b> 作基址	<b>SS</b>	<b>CS, ES, DS</b>	有效地址
通用数据读写 ( <b>BP</b> 作基址除外)	<b>DS</b>	<b>CS, ES, SS</b>	有效地址







## 7、其他

### A、隐含寻址

在指令中没有明显的标出，而指定寄存器参加操作，称之为“隐含寻址”。

**DAA**

**MUL BL**





寄存器名	特殊用途	隐含性质
<b>AX, AL</b>	在输入输出指令中作数据寄存器用	不能隐含
	在乘法指令中存放被乘数或乘积，在除法指令中存放被除数或商	隐 含
<b>AH</b>	在LAHF指令中，作目标寄存器用	隐 含
<b>AL</b>	在十进制运算指令中作累加器用	隐 含
	在XLAT指令中作累加器用	隐 含
<b>BX</b>	在间接寻址中作基址寄存器用	不能隐含
	在XLAT指令中作基址寄存器用	隐 含
<b>CX</b>	在串操作指令和LOOP指令中作计数器用	隐 含
<b>CL</b>	在移位/循环移位指令中作移位次数计数器用	不能隐含
<b>DX</b>	在字乘法/除法指令中存放乘积高位或被除数高位或余数	隐 含
	在间接寻址的输入输出指令中作地址寄存器用	不能隐含
<b>SI</b>	在字符串运算指令中作源变址寄存器用	隐 含
	在间接寻址中作变址寄存器用	不能隐含
<b>DI</b>	在字符串运算指令中作目标变址寄存器用	隐 含
	在间接寻址中作变址寄存器用	不能隐含
<b>BP</b>	在间接寻址中作基址指针用	不能隐含
<b>SP</b>	在堆栈操作中作堆栈指针用	隐 含

## B、串寻址（String addressing）

串寻址方式仅在8086的串指令中使用。规定源操作数的逻辑地址为**DS:SI**；目的操作数的逻辑地址为**ES:DI**。当执行串指令的重复操作时，根据设定的方向标志**DF**，**SI**和**DI**会自动调整。



### c、I/O（输入/输出）端口寻址 (I/O port addressing)

当操作数在外部设备时，使用I/O指令。  
此时有两种不同的寻址方式访问I/O端口。

- (1) 直接端口寻址方式。
- (2) 采用DX寄存器间接寻址方式
- (3) 输入指令中目的操作数可为AL或AX；  
输出指令中源操作数可为AL或AX。

例：

IN AL , 25H	E5 25H
MOV DX , 3E4H	
OUT DX , AL	EE





### 3. 3 8086的指令格式及数据类型

操作码	操作数
-----	-----

指令由操作码和操作数（地址码）组成。8086的指令长度是可变的，一条指令一般由1—6个字节组成。





# 一、指令中的操作数

## 1、单操作数指令

指令助记符

指令的16进制代码

**INC AX**

**40H**

**INC BX**

**43H**



## 2、双操作数指令

指令助记符

**MOV AL , 04**

**MOV AX , 04**

指令的16进制代码

**B004H**

**B80400**







### 3、三个操作数指令

8086指令系统中，大多数指令中只有1—2个操作数，但也有少数指令中有3个操作数，不过有一操作数隐含在操作码中。

例： ADC    AX , BX

该指令完成操作数AX、BX和CF位相加。





## 二、指令中的数据类型

- ◆ 无符号数
- ◆ 带符号数
- ◆ ASCII码
- ◆ BCD数（压缩BCD和非压缩BCD）





## 3. 4 8086的指令集

8086指令系统按功能可分为6大类型:

- 1、 数据传输类
- 2、 算术运算类
- 3、 逻辑运算类
- 4、 串操作类
- 5、 程序控制类
- 6、 处理机控制类





# 一、数据传送指令

■ 数据传送指令又可以分成4种：

- 通用数据传送
- 输入/输出数据传送
- 目的地址传送
- 标志寄存器转送





# 指令的共同特点是：

1、除POP和SAHF指令外，这类指令的操作结果不会影响FR寄存器中的标志。

2、指令中有两个操作数，目的操作数和源操作数，其执行过程为：

目的操作数 ← 源操作数，

当指令中仅列出一个操作数时，另一操作数为隐含。



# 常用的符号

- 累加器 a

- ◆ ax,al

- 寄存器 r:

- ◆ ax,bx,cx,dx,si,di,sp,bp,

- ◆ al,ah,cl,ch,bl,bh,dl,dh

- 段寄存器seg:

- ◆ ds,es,ss,cs

- 内存 mem:

- ◆ [nn], [bx],[si],[di],[bp]

- ◆ [bx/bp+count], [si/di+count]

- ◆ [bx+si/di+count], [bp+si/di+count]

- 立即数 im





# 1、通用数据传送指令

## 1)、MOV 传送指令

指令格式为： MOV 目的，源

功能：目的 ←—— 源



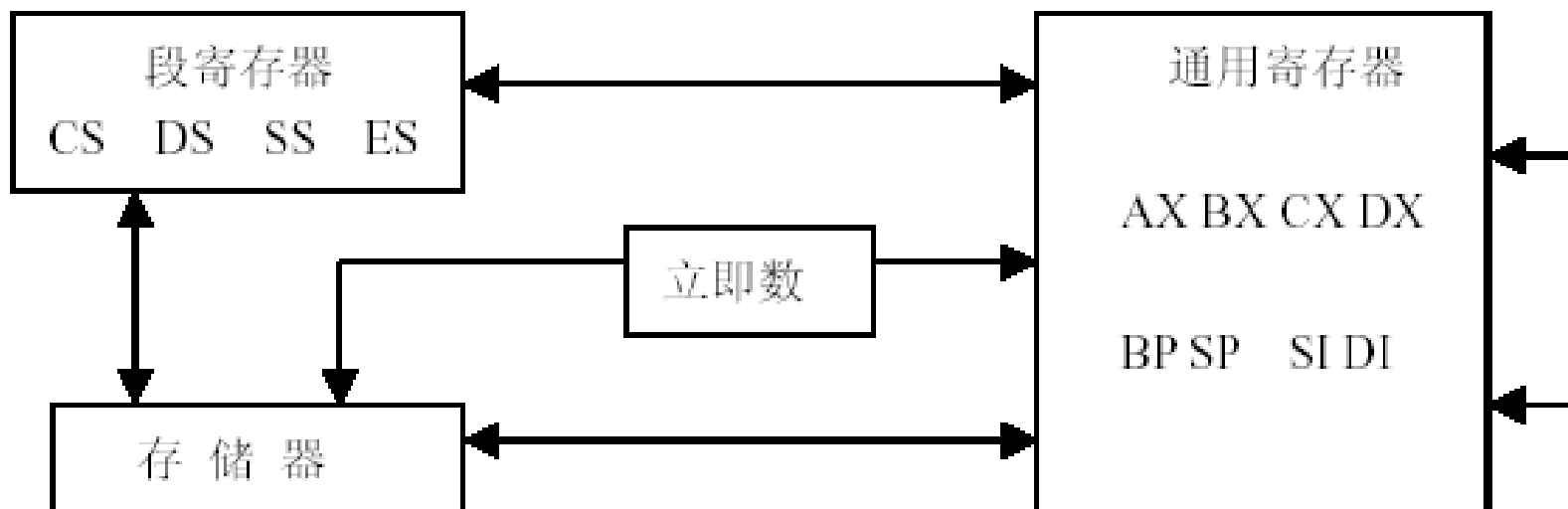


- `mov r,opr`
  - ◆ `Opr: r, mem,seg, im`
- `mov mem, opr`
  - ◆ `Opr: r, seg, im`
- `mov seg, opr`
  - ◆ `Opr: r, mem`

注意： 1、mem不能到mem  
2、seg做目的操作数时， 不包括cs。  
3、类型匹配







## 2) PUSH 进栈指令

指令格式为: **PUSH oprd**

**Oprd: r, mem, seg**

其操作过程是:

- a、**SP-2**, 指示堆栈中可以存放数据的位置
- b、存源操作数, 完成进栈操作。





### 3) POP 出栈指令

指令格式为: POP oped

Oped: r, mem, seg(不包括 CS!!!)

其操作过程是:

- a、将SS:SP所指示的栈顶处的两个字节的数据, 弹到目的操作数中;
- b、 $SP+2$ , 指示当前栈顶位置, 完成出栈操作。



## 4) XCHG 交换指令

指令格式为:

**XCHG 目的操作数 , 源操作数**

**XCHG R, OPRD**

**OPRD: R, MEM**

操作数不能为立即数;

源和目的不能同时为存储单元;

段寄存器不能作为操作数。





- 举例：  
把2000H单元的字和3000H单元互换





## 2、累加器专用传送指令

### 1) IN 输入指令

指令格式为:    **IN   AL , n**  
                  **IN   AX , n**  
                  **IN   AL , DX**  
                  **IN   AX , DX**

### 2) OUT 输出指令

指令格式为:    **OUT   n , AL**  
                  **OUT   n , AX**  
                  **OUT   DX , AL**  
                  **OUT   DX , AX**





- 3) **XLAT** 换码指令

指令格式为: **XLAT**

功能  $[bx + al] \rightarrow al$

例: 查表求  $n$  的平方。  $n:[0-9]$

1、将  $0-9$  的平方表建立在偏移地址为  $2000H$  的内存中, 如图。

2、查表





完成求5的平方指令序列为：

**MOV BX, 2000H** ;指向平方表的首地址

**MOV AL, 5** ; 将5换码成5的平方值

**XLAT** ; 查表，平方值在AL中



地址	平方值
2000H	00
2001H	01
...	...
2009H	81





### 3、目标地址传送指令

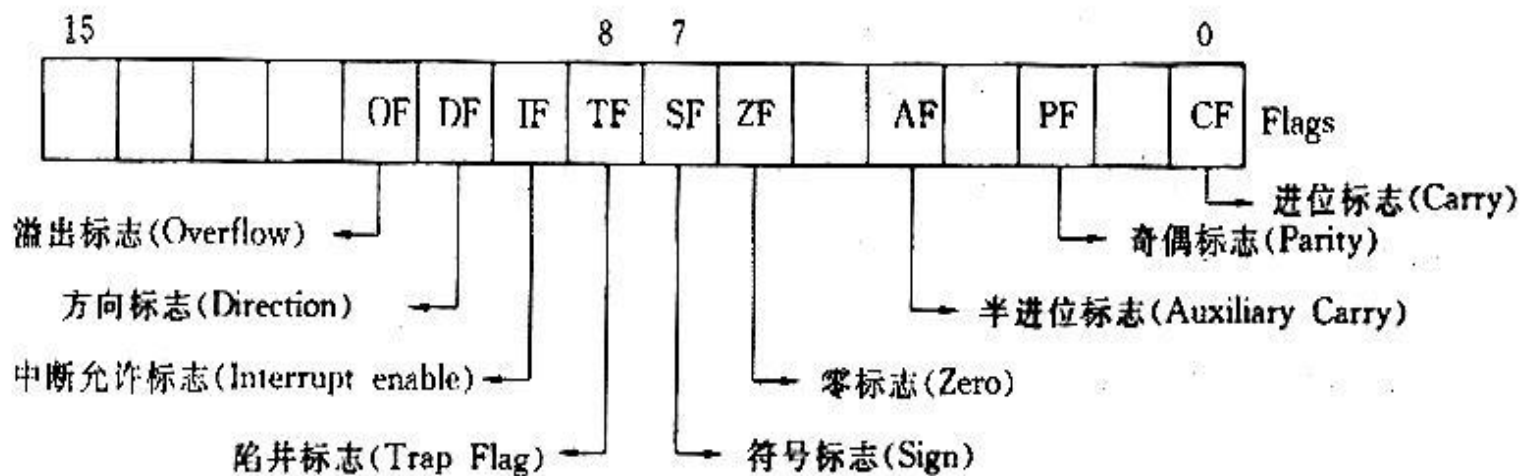
这类指令有：

- 1) **LEA** 有效地址传送到寄存器
- 2) **LDS** 装入一个新的物理地址
- 3) **LES** 装入一个新的物理地址



## 4、标志寄存器传送指令

- 1) **LAHF** AH 送 FR寄存器的低8位
- 2) **SAHF** FR寄存器的低8位送AH
- 3) **PUSHF** FR寄存器推入堆栈
- 4) **POPF** 从栈顶中弹出存入FR寄存器





- 举例：
- 1、把FR各位清零
  - 2、把TF置1，其他位不变





## 二、算术运算指令

### 1、算术加法指令

- 1) **ADD** 算术加法

指令功能：目的地 $\leftarrow$ 目的操作数+源操作数

- 格式

- **ADD R,OPRD**

**OPRD: R,MEM,IM**

- **ADD MEM, OPRD**

**OPRD:R,im**

算术指令影响标志位





## 二、算术运算指令

### 1、算术加法指令

- 2) **ADC** 带进位算术加法

指令功能：目的地 $\leftarrow$ 目的操作数+源操作数+CF

- 3) **INC** 加1指令

指令功能：目的地 $\leftarrow$ 目的操作数+1

- 4) **DAA** 对压缩BCD数加法操作的结果进行校正

指令功能：对AL寄存器的内容进行十进制调整

- 5) **AAA** 对非压缩BCD数加法操作的结果进行校正

指令功能：对AL寄存器的内容进行十进制调整

- ◆ 举例：多位的加法



## ◆ 2、算术减法指令

### 1) SUB 算术减法

指令功能：目的地  $\leftarrow$  目的操作数 - 源操作数

### 2) SBB 带进位算术减法

指令功能：目的地  $\leftarrow$  目的操作数 - 源操作数 - CF

### 3) DEC 减1指令

指令功能：目的地  $\leftarrow$  目的操作数 - 1

### 4) DAS 对压缩BCD数减法操作的结果进行校正

指令功能：对AL寄存器的内容进行十进制调整

### 5) AAS 对非压缩BCD数减法操作的结果进行校正

指令功能：对AL寄存器的内容进行十进制调整

### 6) CMP 比较指令

指令功能：两个操作数相减，不产生运算结果仅影响标志

### 7) NEG 取补指令

指令功能：0 - 目的操作数



### 3、算术乘法指令

#### 1) MUL 无符号数乘法

指令功能：完成两个操作数相乘

MUL OPRD;

AL\*OPRD->AX

AX\*OPRD->DX AX

OPRD: R, MEM

#### 2) IMUL 带符号数乘法

指令功能：完成两个操作数相乘

#### 3) AAM 非压缩BCD数乘法操作结果校正

指令功能：完成两个非压缩BCD数乘法结果的十进制数调整

举例：把扩展BCD转二进制数



## 4、算术除法指令

- **DIV** 无符号数除法

指令功能：完成两个操作数相除

**DIV OPRD**

$AX/OPRD \rightarrow AL$ :商,  $AH$ :余数

$DXAX/OPRD \rightarrow AX$ :商,  $DX$ :余数

- **IDIV** 带符号数除法

指令功能：完成两个操作数相除

另外还有：

**CBW** 带符号数字字节扩展、

**CWD** 带符号数字扩展、

**AAD** 非压缩BCD数除法校正

举例：把AL二进制数转为扩展BCD数







# 三、位操作指令

指令的共同点是：

- (1) 可以按二进制位进行操作；
- (2) 逻辑运算指令按逻辑门电路的运算规则，
- (3) 逻辑移位指令有左移和右移，移出的位都进入CF标志
- (4) 因移空位的补充方式不同有多种指令形式
- (5) 逻辑移位指令中，移动超过1次则用CL寄存器做计数器
- (6) 执行逻辑操作指令，CF均被清0





# 1、 逻辑运算指令

- 1) **NOT** 逻辑求反  
指令功能：将8位、16位寄存器或存储器内容求反
- 2) **AND** 逻辑与操作  
指令功能：将8位、16位寄存器或存储器内容和源操作数“与”
- 3) **OR** 逻辑或操作  
指令功能：将8位、16位寄存器或存储器内容和源操作数“或”
- 4) **XOR** 逻辑异或操作  
指令功能：将8位、16位寄存器或存储器内容和源操作数“异或”
- 5) **TEST** 测试指令  
指令功能：将8位、16位寄存器或存储器内容和源操作数“与”，不产生运算结果，仅影响状态标志





- AND 可以使指定位为0
- OR 可以使指定位为1
- XOR 可以使指定位为非





## 2、逻辑移位指令

### 1) SHL 逻辑左移

指令功能：将8位、16位寄存器或存储器内容左移，移空的位补0

**SHL OPRD, 1**

**OR**

**SHL OPRD, CL**

### 2) SAL 算术左移

指令功能：将8位、16位寄存器或存储器内容左移，移空的位补0

举例：扩展BCD码→BCD 码





## 2、逻辑移位指令

### 3) SHR 逻辑右移

指令功能：将8位、16位寄存器或存储器内容右移，移空的位补0

### 4) SAR 算术右移

指令功能：将8位、16位寄存器或存储器内容右移，移空的位由最高位补充。

举例：BCD 码→扩展BCD码





## 5) **ROL** 不带进位循环左移

指令功能：将8位、16位寄存器或存储器内容左移，移空的位由移出位补充。

## 6) **ROR** 不带进位循环右移

指令功能：将8位、16位寄存器或存储器内容右移，移空的位由移出位补充。





## 7) RCL 带进位循环左移

指令功能：将8位、16位寄存器或存储器内容左移，移空的位由CF位补充。

## 8) RCR 带进位循环右移

指令功能：将8位、16位寄存器或存储器内容右移，移空的位由CF位补充。

举例：倒序程序





## 四、串处理指令

- 源操作数指针 DS: SI,  
目的操作数指针 ES: DI
- 每操作一次SI,DI修改一次, 方向由DF控制
- 重复操作的退出
  - ◆ cx控制
  - ◆ 条件控制(ZF标志)







## 四、串处理指令

- 1、串传送指令 **MOVSB / MOVSW**
- 2、串比较指令 **CMPSB/COMPSW**
- 3、串搜索指令 **SCASB/SCASW**
- 4、串装入指令 **LODSB/LODSW**
- 5、串存储指令 **STOSB/STOSB**
- 6、指令前缀  
**REP、REPZ/REPE、REPNZ/REPNE**





# 五、程序控制转移指令

## 1、无条件转移指令

JMP 目标地址

### 1)、段内直接转移

JMP LABEL

$IP + \text{位移量} \rightarrow IP$

位移量 = LABEL指令地址 - JMP地址 - 3

### 2)、短转移指令

JMP LABEL

$IP + CBW(\text{位移量}) \rightarrow IP$

位移量 (8bit) = LABEL指令地址 - JMP地址 - 2





### 3)、段内间接转移

**JMP OPRD; OPRD:MEM**

offset OPRD  $\rightarrow$  IP

**OPRD:r, MEM**

**JMP [SI]**





### 3)、段间直接转移

JMP far LABEL

offset LABEL  $\rightarrow$  IP

seg LABEL  $\rightarrow$  cs

### 4)、段间间接转移

JMP OPRD; **OPRD:MEM**

offset OPRD  $\rightarrow$  IP

seg OPRD  $\rightarrow$  cs

JMP DWORD PTR [SI]



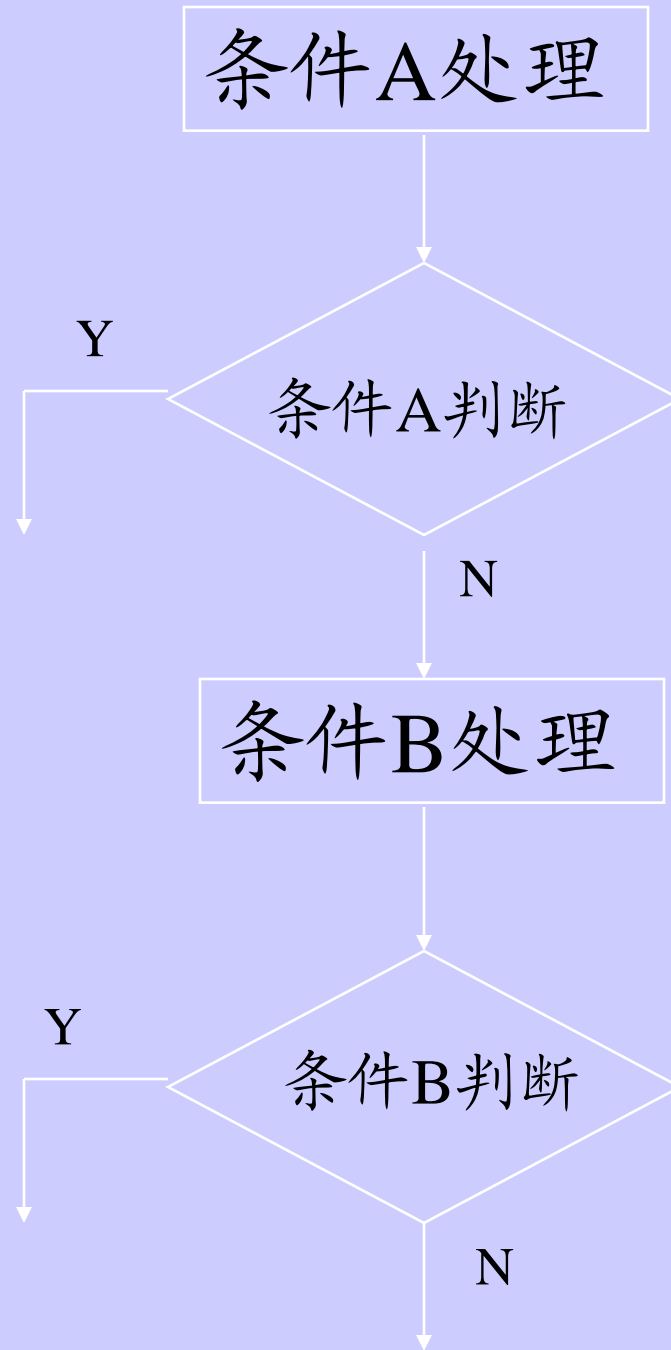


## 2、条件转移指令

### 1) 单条件转移指令

- |   |         |              |
|---|---------|--------------|
| ① | JC      | ;CF标志为1, 则转移 |
| ② | JNC     | ;CF标志为0, 则转移 |
| ③ | JE/JZ   | ;ZF标志为1, 则转移 |
| ④ | JNE/JNZ | ;ZF标志为0, 则转移 |
| ⑤ | JS      | ;SF标志为1, 则转移 |
| ⑥ | JNS     | ;SF标志为0, 则转移 |
| ⑦ | JO      | ;OF标志为1, 则转移 |
| ⑧ | JNO     | ;OF标志为0, 则转移 |
| ⑨ | JP/JPE  | ;PF标志为1, 则转移 |
| ⑩ | JNP/JPO | ;PF标志为0, 则转移 |







## 2) 用于无符号数的条件转移指令

① **JA/JNBE** ; 高于/不低于等于转移,  $CF \setminus ZF = 0$

② **JNA/JBE** ; 不高于/低于等于转移,  $CF \setminus ZF = 1$

③ **JB/JNAE** ; 低于/不高于等于转移,  $CF = 1$

④ **JNB/JAE** ; 不低于/高于等于转移,  $CF = 0$





### 3) 用于带符号数的条件转移指令

#### ① JG/JNLE

;大于/不小于等于转移,  $(SF \vee OF) \vee ZF=0$

#### ② JGE/JNL

;大于等于/不小于转移,  $(SF \vee OF) = 0$

#### ③ JL/JNGE

;小于/不大于等于转移,  $(SF \vee OF) = 1$

#### ④ JLE/JNG

;小于等于/不大于转移,  $(SF \vee OF) \vee ZF=1$







### 3、循环控制指令

#### 1) LOOP指令

指令格式： **LOOP** 目标地址

执行该指令，**CX**−1，若**CX**≠0，转移到目标地址，即：

**IP**←**IP**+8位位移量（带符号扩展到16位）。  
由于指令自动对**CX**寄存器做减1操作，故使用**LOOP**指令前，需将循环操作次数值赋给**CX**寄存器。

- 2) 另外还有：**LOOPZ**/**LOOPE**指令、**LOOPNZ**/**LOOPNE**指令、**JCXZ**指令。





## 4、子程序调用和返回指令

### 1) CALL指令

- 段内直接调用

例: **CALL LABEL**

a、 $IP \rightarrow \text{栈}$     b、 $IP + \text{位移量} \rightarrow IP$

- 段内间接调用

例: **CALL WORD PTR [SI]**

- 段间直接调用

例: **CALL FAR 目标地址**

- 段间间接调用

例: **CALL DWORD PTR [SI]**





## 2)、RET 子程序返回指令

和调用指令**CALL** 相对应的是返回指令**RET**。返回指令通常作为一个子程序或过程的最后一条指令，它用以返回到调用这个子程序的断点处。

关于**RET n** 指令，这条指令称为带参数返回指令。





## 5、中断指令和中断返回指令

- 1) **INT n** 中断指令
- 2) **INTO** 溢出中断指令
- 3) **IRET** 中断返回指令





## 六、处理器控制指令

### 1、标志控制指令

STC	； 使CF 置1
CLC	； 使CF 清0
CMC	； 使CF 取反
STD	； 使DF 置1
CLD	； 使DF 清0
STI	； 使IF置1
CLI	； 使IF 清0





## 2、外同步指令

- **HLT** 处理器暂停指令
- **WAIT** 等待指令
- **ESC** 换码指令/交权指令
- **LOCK** 总线封锁指令
- **NOP** 空操作指令