

## 2.4 算术运算指令

- ◇ 算术运算指令用来执行二进制的算术运算：加、减、乘、除。
- ◇ 这类指令会根据运算结果影响状态标志，有时要利用某些标志才能得到正确的结果，使用他们时请留心有关状态标志。
- ◇ 重点掌握
  - ◆ 加法指令：ADD、ADC、INC
  - ◆ 减法指令：SUB、SBB、DEC、CMP、NEG

## 2.4.1 加法和减法指令

- ◇ 加法指令: **ADD**, **ADC**和**INC**
- ◇ 减法指令: **SUB**, **SBB**, **DEC**, **NEG**和**CMP**
- ◇ 他们分别执行字或字节的加法和减法运算，除**INC**和**DEC**不影响**CF**标志外，其他按定义影响全部状态标志位
- ◇ 操作数组合：
  - ◆ 运算指令助记符 **reg, imm/reg/mem**
  - ◆ 运算指令助记符 **mem, imm/reg**

# 1. 加和减指令



## ◇ ADD dest,src

； 加法： $\text{dest} \leftarrow \text{dest} + \text{src}$

； **ADD**指令把两操作数相加，和存放为目的操作数

---

## ◇ SUB dest,src

； 减法： $\text{dest} \leftarrow \text{dest} - \text{src}$

； **SUB**指令用目的操作数减去源操作数，差送到目的操作数

## 例: 加法

`mov ax,7348h` ; AX=7348H

`add al,27h`

`add ax,3fffh`

;  $AL = 48H + 27H = 6FH$ ,  $AX = 736FH$

;  $OF = 0$ ,  $SF = 0$ ,  $ZF = 0$ ,  $PF = 1$ ,  $CF = 0$

;  $AX = 736FH + 3FFFH = B36EH$

;  $OF = 1$ ,  $SF = 1$ ,  $ZF = 0$ ,  $PF = 0$ ,  $CF = 0$

## 例: 減法

**sub ah,0f0h** ;承接上页程序, 指令执行前AH=0B3H

; AH=B3H-F0H=C3H, AX=C36EH

; OF=0, SF=1, ZF=0, PF=1, CF=1

**mov word ptr[200h],0ef00h**

; [200H]=EF00H, 标志不变

**sub [200h],ax**

; [200H]=EF00H-C36EH=2B92H

; OF=0, SF=0, ZF=0, PF=0, CF=0

**sub si,si** ; SI=0

; OF=0, SF=0, ZF=1, PF=1, CF=0

## 2. 带进位加和减指令

### ◇ **ADC** dest,src

； 加法： $\text{dest} \leftarrow \text{dest} + \text{src} + \text{CF}$

； **ADC**指令除完成**ADD**加法运算外，还要加上进位**CF**，结果送到目的操作数

---

### ◇ **SBB** dest,src

； 减法： $\text{dest} \leftarrow \text{dest} - \text{src} - \text{CF}$

； **SBB**指令除完成**SUB**减法运算外，还要减去借位**CF**，结果送到目的操作数

# 无符号双字加法和减法

例：编程计算下面表达式的值：

8234 7856H + 1234 8998H - 8000 4491H

mov ax,7856h	;AX=7856H
mov dx,8234h	;DX=8234H
add ax,8998h	;AX=01EEH, CF=1
adc dx,1234h	;DX=9469H, CF=0
sub ax,4491h	;AX=BD5DH, CF=1
sbb dx,8000h	;DX=1468H, CF=0

**DX.AX=8234 7856H+1234 8998H-8000 4491H  
=1468 BD5DH**

### 3. 比较指令CMP (compare)

**CMP** dest,src

- ； 做减法运算： $\text{dest} - \text{src}$
- ； **CMP**指令将目的操作数减去源操作数，但差值不回送目的操作数

比较指令通过减法运算影响状态标志，用于比较两个操作数的大小关系

```
cmp ax,bx  
cmp al,100
```



## 4. 增量和减量指令

**INC** reg/mem

； 增量（加1）： $\text{reg/mem} \leftarrow \text{reg/mem} + 1$

**DEC** reg/mem

； 减量（减1）： $\text{reg/mem} \leftarrow \text{reg/mem} - 1$

✧ **INC**指令和**DEC**指令是单操作数指令

✧ **INC**和**DEC**不影响**CF**标志

```
inc si          ; si ← si + 1  
dec byte ptr [si] ; [si] ← [si] - 1
```

## 5. 求补指令NEG (negative)

**NEG** reg/mem

;  $\text{reg/mem} \leftarrow 0 - \text{reg/mem}$

- ◇ **NEG**指令对操作数执行求补运算，即用零减去操作数，差存回操作数
- ◇ **NEG**指令对标志的影响与用零作减法的**SUB**指令一样，**NEG**指令是一个单操作数指令

## 例: 求补运算

mov ax,0ff64h

neg al ;  $AL = 0 - 64H = 9CH$ ,  $AX = FF9CH$   
;  $OF = 0$ ,  $SF = 1$ ,  $ZF = 0$ ,  $PF = 1$ ,  $CF = 1$

sub al,9dh ;  $AL = 9CH - 9DH = FFH$ ,  $AX = FFFFH$   
;  $OF = 0$ ,  $SF = 1$ ,  $ZF = 0$ ,  $PF = 1$ ,  $CF = 1$

neg ax ;  $AX = 0 - FFFFH = 0001H$   
;  $OF = 0$ ,  $SF = 0$ ,  $ZF = 0$ ,  $PF = 0$ ,  $CF = 1$

dec al ;  $AL = 01H - 1 = 0$ ,  $AX = 0000H$   
;  $OF = 0$ ,  $SF = 0$ ,  $ZF = 1$ ,  $PF = 1$ ,  $CF = 1$

neg ax ;  $AX = 0 - 0 = 0$   
;  $OF = 0$ ,  $SF = 0$ ,  $ZF = 1$ ,  $PF = 1$ ,  $CF = 0$

## 2.4.3 乘法指令

- ✧ 乘法运算分为无符号数乘法和有符号数乘法，各有相应的指令。
- ✧ 两个**8**位二进制数相乘，积为**16**位二进制数；
- ✧ 两个**16**位二进制数相乘，积为**32**位二进制数。

Next

## 2.4.3 乘法指令

指令格式：

✧ 无符号数乘法

**MUL SRC**;  $(AX) \leftarrow (AL) \times (SRC)$  8位数乘法

$(DX, AX) \leftarrow (AX) \times (SRC)$  16位数乘法

✧ 有符号数乘法

**IMUL SRC**; 操作同上，但是操作数为带符号数

注意：

✧ 源操作数不允许使用立即数寻址方式。

Next

## 2.4.3 乘法指令

- ✧ 乘法指令只影响**CF**和**OF**标志，对其他标志位无影响（状态不定）。
- ✧ 对于**MUL**指令，如果字节型数据相乘之积（**AH**）=0或字数据相乘之积（**DX**）=0，则**CF=OF=0**，否则**CF=OF=1**；
- ✧ 对于**IMUL**指令，如果字节数据相乘之积**AH**或字数据相乘之积**DX**的内容是低一半的符号扩展，则**CF=OF=0**，否则**CF=OF=1**。

### 2.4.3 乘法指令 举例

例：若把(AL)=0FEH视为无符号数，其数值为254，视为有符号数，则数值为-2。

把(BH)=0AH视为无符号数，其数值为10，视为有符号数，数值为+10。

✧执行指令 **MUL BH** 后 (AX) = 09ECH,  
CF=OF=1

✧执行指令 **IMUL BH** 后 (AX) = FFECH,  
CF=OF=0

## 4. 除法指令

✧除法运算分为无符号数除法和有符号数除法，各有相应的指令。

✧规则：

✧当除数是8位二进制数时，要求被除数是16位二进制数；

✧当除数是16位二进制数时，要求被除数是32位二进制数。



## 4. 除法指令

指令格式：

无符号数除法： **DIV SRC**

$(AL) \leftarrow (AX) / (SRC)$  8位二进制数除法的商

$(AH) \leftarrow (AX) / (SRC)$  8位二进制数除法的余数

或

$(AX) \leftarrow (DX, AX) / (SRC)$  16位二进制数除法的商

$(DX) \leftarrow (DX, AX) / (SRC)$  16位二进制数除法的余数

有符号数除法： **IDIV SRC**

；操作同上，但是操作数为带符号数

## 4. 除法指令

### 注意：

- ◇ 当除数是字节数据时，被除数必须放在**AX**中；
- ◇ 当除数是字数据时，被除数必须放在**DX, AX**中。
- ◇ 除法指令对状态标志无定义（状态不定）。
- ◇ 若除数为**0**或除法运算结果超出规定的范围时，将产生**0**号中断。
- ◇ **8086/8088**规定**IDIV**指令运算结果余数的符号与被除数相同。
- ◇ 有符号数除法运算中，当被除数位数不够时，则需将被除数扩展到所需的位数。**8086/8088**设有带符号数扩展指令。

# 符号扩展指令

指令格式:

✧ 字节扩展到字 **CBW**

；将寄存器 **AL** 中的符号位扩展到寄存器 **AH**

✧ 字扩展到双字 **CWD**

；将寄存器 **AX** 中的符号位扩展到寄存器 **DX**

这两条指令不影响标志位。

## 符号扩展指令应用举例

例：编程计算表达式  $(-125) \times 10 \div 300$  的值。

MOV AL,83H ; (AL)  $\leftarrow$  (-125)

MOV BL,10

IMUL BL ; (AX)  $\leftarrow$  (-125)  $\times$  10

CWD ; (AX) 符号扩展到 (DX)

MOV BX,300

IDIV BX

运算结果：AX= FFFCH=-4，余数DX=FFCEH=-50



十进制调整指令  
请同学们自学

## 5. 十进制调整指令

**BCD**码是一种用二进制编码的十进制数，又称为二—十进制数。

8086/8088中**BCD**码分为两种形式：

- ①压缩的**BCD**码；
- ②非压缩的**BCD**码，它的低四位是**BCD**码，高四位没有意义。

## 5. 十进制调整指令

- ✧ 由于**BCD**码是四位二进制编码，四位二进制数共有**16**个编码，**BCD**码只用其中的**10**个，其余没用的编码称为**无效码**。
- ✧ **BCD**码运算结果进入或跳过无效码区时，都会出现错误。为了得到正确结果，必须进行调整。
- ✧ **8086/8088**针对压缩**BCD**码和非压缩**BCD**码，分别设有两组十进制调整指令，其调整方法略有不同。

## (1) 压缩BCD码十进制调整指令

指令格式:

◇ 加法十进制调整:

DAA; (AL) ← 把AL中的和调整到压缩BCD码格式

◇ 减法十进制调整:

DAS; (AL) ← 把AL中的差调整到压缩BCD码格式

调整方法是:

① 累加器AL低4位大于9或辅助进位标志位AF = 1, 则累加器AL加06H修正。

② 累加器AL高4位大于9或进位标志位CY = 1, 则累加器AL加60H修正。

③ 累加器AL高4位大于等于9, 低4位大于9, 则累加器AL进行加66H修正。



[例]进行BCD码加法运算59+68=127

$$\begin{array}{r}
 \begin{array}{rcl}
 59 & 0101 & 1001 \\
 +) & 68 & 0110 \ 1000 \\
 \hline
 127 & 1100 & 0001
 \end{array} \\
 +) & & 0110 \ 0110 \\
 \hline
 1 & 0010 & 0111
 \end{array}$$

注意：

- ✧ 压缩BCD码加法或减法十进制调整指令必须用在ADD (ADC) 或SUB (SBB) 指令之后，调整结果对标志OF无影响，对其他状态标志位均有影响。
- ✧ 减法十进制调整方法与加法十进制调整类同，只是将加6变为减6操作。

## (2) 非压缩BCD码十进制调整指令

指令格式

加法十进制调整

AAA ; (AL) &0FH > 9, 或AF=1则  
(AL)  $\leftarrow$  (AL) + 6  
(AF)  $\leftarrow$  1, (CF)  $\leftarrow$  (AF)  
(AH)  $\leftarrow$  (AH) + 1, (AL)  $\leftarrow$  (AL) &0FH

减法十进制调整

AAS ; (AL) &0FH > 9, 或AF=1则  
(AL)  $\leftarrow$  (AL) - 6  
(AF)  $\leftarrow$  1, (CF)  $\leftarrow$  (AF)  
(AH)  $\leftarrow$  (AH) - 1, (AL)  $\leftarrow$  (AL) &0FH

## (2) 非压缩BCD码十进制调整指令

### 指令格式

#### 乘法十进制调整

**AAM** ;  $(AH) \leftarrow (AL) / 0AH, (AL) \leftarrow (AL) \% 0AH$

#### 除法十进制调整

**AAD** ;  $(AL) \leftarrow 10 \times (AH) + (AL), (AH) \leftarrow 0$



注意：

- ①非压缩BCD码加法十进制调整指令必须用在ADD (ADC) 或SUB (SBB) 指令之后，结果影响标志位AF和CF，对其他标志位均无定义。
- ②非压缩BCD码乘法十进制调整指令必须用在MUL指令之后，结果影响标志位SF、ZF和PF，对AF、CF和OF标志位均无影响。
- ③非压缩BCD码除法十进制调整指令的应用与乘法不同，AAD指令必须用在DIV指令之前，先将AX中的非压缩BCD码被除数调整为二进制数（仍在AX中），再进行除法运算，使商和余数也是非压缩BCD码，结果影响标志位SF、ZF和PF，对AF、CF和OF标志位均无影响。




[例] 已知:  $(AX) = 0234H$ ,  $(BH) = 38H$

指令: `ADD AL, BH`

`AAA`

- ◇ 指令执行前, 寄存器AL和BH的内容分别为4和8的ASCII码。
- ◇ 第一条指令执行后,  $(AL) = 6CH$ ,  $AF = 0$ ;
- ◇ 第二条指令进行十进制调整, 结果为:  $(AX) = 0302H$ ,  $AF = 1$ ,  $CF = 1$ 。

- 
- ◇ DAA: Decimal Adjust for Addition
  - ◇ DAS: Decimal Adjust for Subtraction
  - ◇ AAA: ASCII Adjust for Addition
  - ◇ AAS: ASCII Adjust for Subtraction
  - ◇ AAM: ASCII Adjust for Multiplication
  - ◇ AAD: ASCII Adjust for Division

