第3章
指令系统（II）

3.3.5 累加器专用传送指令

- 有输入输出指令IN/OUT和换码指令XLAT
- 该类指令只和累加器AL或AX有关，而与AH无关。
- 输入输出指令用来完成I/O端口与累加器之间的数据传送，指令中给出I/O端口的地址值。
- 若I/O端口是16位地址，则IN、OUT指令中的地址要用DX间接寻址，绝不能在指令中直接出现16位I/O地址，而8位I/O地址可直接写在输入/输出指令中。

1、IN 输入指令(Input)

- 指令格式：

（1）IN AL，端口地址

或 IN AX，端口地址

（2）IN AL，DX；端口地址存放在DX寄存器中

或 IN AX，DX

- 指令功能：从8位端口读入一个字节到AL寄存器，或从16位端口读一个字到AX寄存器。

16位端口由两个地址连续的8位端口组成，从16位端口输入时，先将给定端口中的字节送进AL，再把端口地址加1，然后将该端口中的字节读入AH。

注： 输入输出端口号，范围0~255(00~FFH)

【例】

```
IN   AL, 0F1H;

IN   AX, 80H;


MOV     DX, 310H;
IN   AL, DX
```

2、OUT 输入指令(Output)

- 指令格式:

（1）OUT 端口地址，AL

或 OUT 端口地址，AX

（2）OUT DX，AL；DX = 端口地址

或 OUT DX，AX

- 指令功能：将AL中的一个字节写到一个8位端口，或把AX中的一个字写到一个16位端口。

同样，对16位端口进行输出操作时，也是对两个连续的8位端口继续输出操作。

【例】

```
OUT     85H, AL;

MOV     DX, 300H;
OUT     DX, AX;
```

判断：OUT　3FFH, AL;

3、XLAT 表转换指令(Table Lookup-Translation)

- 指令格式：XLAT
- 执行操作：(AL) ← ((BX) + (AL))
- 标志位：不影响
- 指令功能：通过AL和BX寄存器进行表格查询，即：将累加器AL中的一个字节转换为内存表格中的数据，表格的偏移地址由BX与AL内容之和确定。

❑XLAT指令使用步骤：

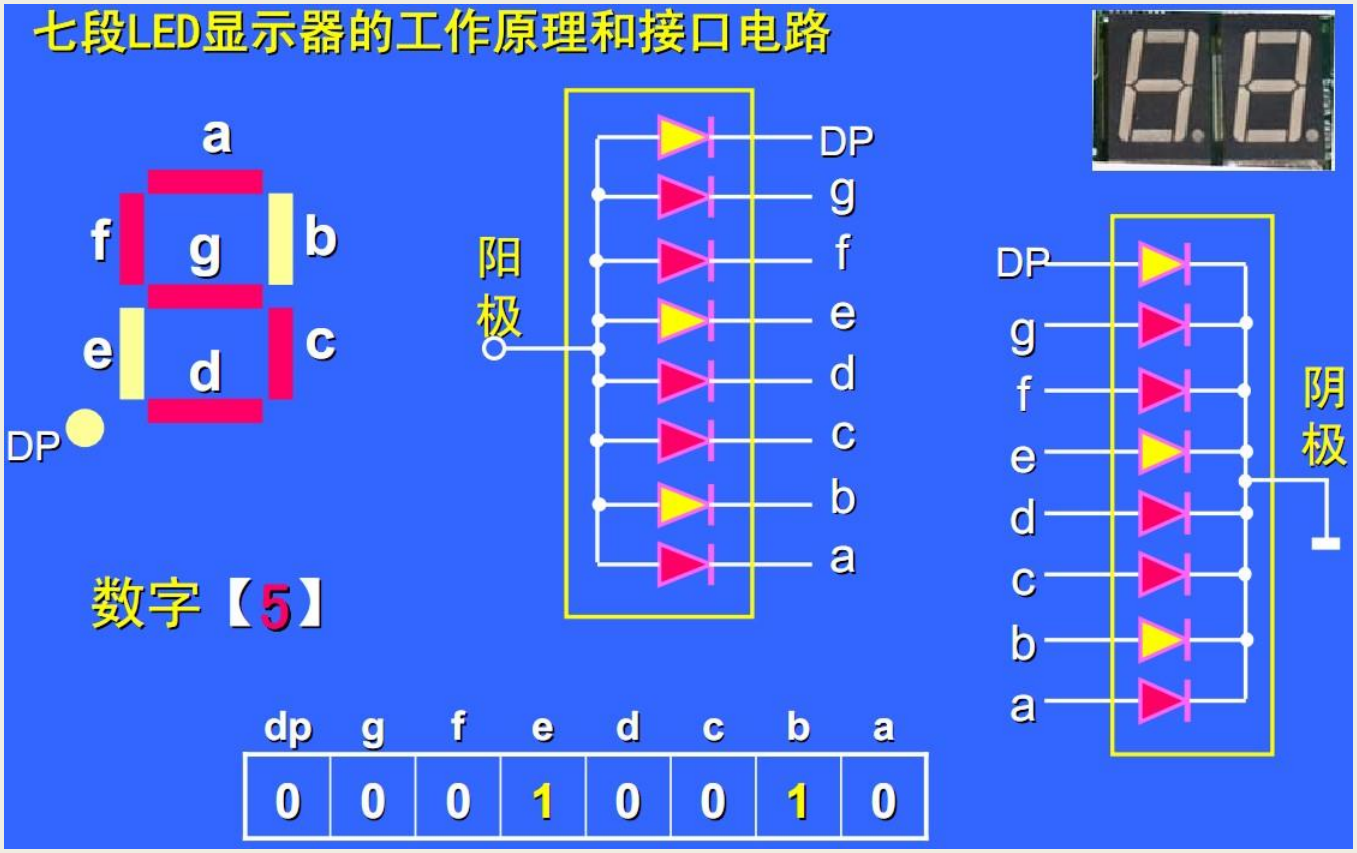
A. 使用之前必须先建立一个表格，表格中的内容是所需要转换的代码；

B. 将转换表的起始地址装入→BX寄存器；

C. 欲查的某项与表头地址的位移量→AL，即表格最多包含256个字节；

D. 执行XLAT指令后，根据位移量从表中查到转换后的代码值→AL寄存器中。

【例】若十进制数字0~9的LED七段码对照表如表所示，试用XLAT指令求数字5的七段码值。



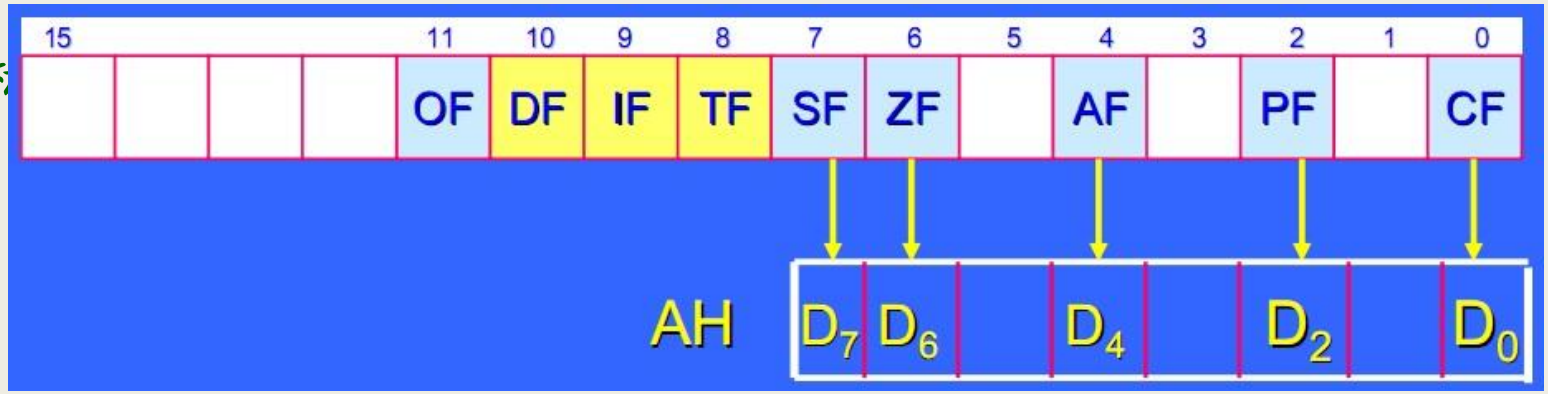
十进制数字	七段显示码	十进制数字	七段显示码
0	40H	5	12H
1	79H	6	02H
2	24H	7	78H
3	30H	8	00H
4	19H	9	18H

3.3.6 标识寄存器传送指令

- 标志寄存器传送指令用来传送标志寄存器FLAGS的内容，方便进行对各个标志位的直接操作
- 有2对4条指令
  - ✓ 低8位传送：LAHF和SAHF
  - ✓ 16位传送：PUSHF和POPF

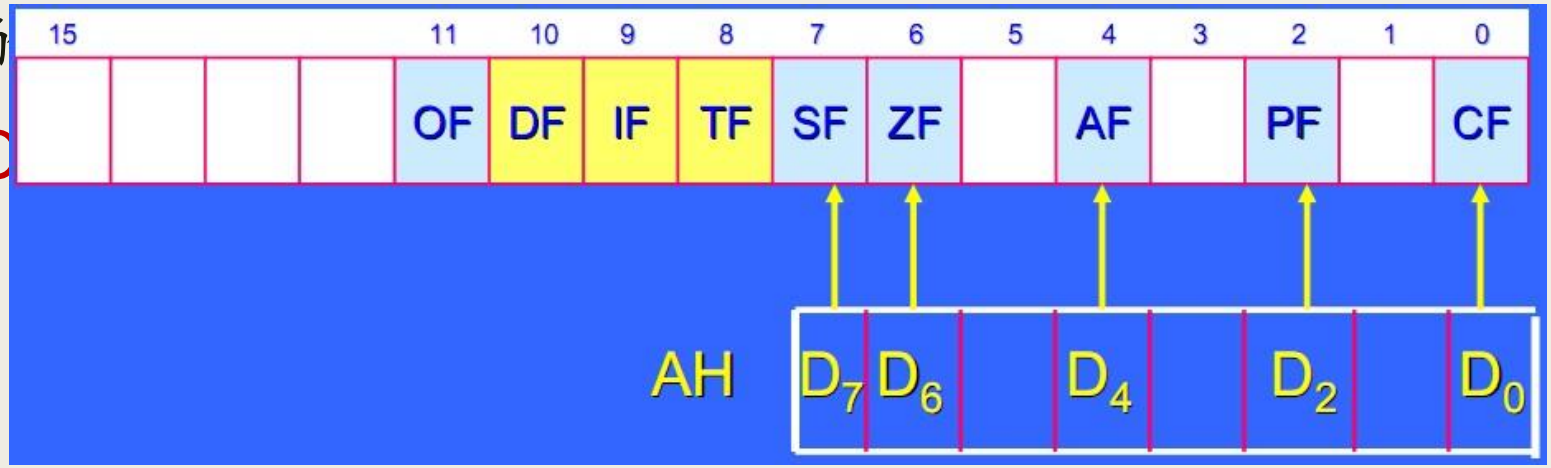
1、LAHF 标志送到AH指令(Load AH from Flag)

- 指令格式：LAHF
- 指令功能：将标志寄存器中低8位传送到AH中，包括5个状态标志SF、ZF、AF、PF、CF，其对应的位是第7、6、4、2和0，而5、3、1位没有定义。
- 标



2、SAHF AH送标志寄存器(Store AH into Flags)

- 指令格式：SAHF
- 指令功能：把AH内容存入标志寄存器。这条指令与LAHF的操作相反，它把寄存器AH中的7、6、4、2、0位传送到标志寄存器的



3、PUSHF 标志入栈指令(Push Flag onto Stack)

该指令不影响标志位

- 指令格式：PUSHF
- 指令功能：把整个标志寄存器的内容压入堆栈；同时修改堆栈指针，使SP ← SP－2；

4、POPF 标志出栈指令(Pop Flag off Stack)

该指令影响标志位

- 指令格式：POPF
- 指令功能：把当前堆栈指针SP所指的一个字弹入标志寄存器FLAGS；并修改堆栈指针，使SP ← SP＋2。

注意：

- 要成对使用PUSHF和POPF，可对标志寄存器进行保存和恢复。
- 常用在：过程(子程序)调用，中断服务程序，对主程序的状态(即标志位)进行保护。
- 也可用来改变追踪标志TF。在8086指令系统中没有直接改变TF(D8位)的指令。

【课后习题13】若SS = 1000H，SP = 1000H，AX = 1234H，BX = 5678H，Flag = 2103H，试说明执行指令：

```
PUSH        BX
PUSH        AX
PUSHF
POP         CX
```

之后，SP=？SS=？CX=？并画图指出栈中各单元的内容。

【课后习题18】指出下列8086指令中哪些是错误的，错在何处?

(1)	MOV DL，BX
(2)	MOV DS，2000H
(3)	MOV [DI]，[SI]
(4)	MOV AX，[BX][BP]
(5)	XCHG CX，[2400H]
(6)	PUSH DH
(7)	IN AH，DX
(8)	OUT 01F0H，AX

3.4 算术运算类指令

- 算术运算指令可处理4种类型的数:
  - 无符号二进制整数
  - 带符号二进制整数
  - 无符号压缩十进制整数(Packed Decimal)
  - 无符号非压缩十进制整数(Unpacked Decimal)

一个8位二进制数可看成4种不同类型的数，所表示的数值亦不同。

数的表示

- 二进制数：可以是8位或16位，若是带符号数，则用补码表示。
- 压缩十进制数：一个字节中存放两个BCD码十进制数。
- 非压缩十进制数：一个字节的低半字节存放十进制数，高半字节为全零。

【例】对十进制数字58：

压缩十进制数表示：只需一个字节，即0101 1000B；

非压缩十进制数表示：需两个字节，即0000 0101B 和0000 1000B。

算术运算类指令

8086/8088指令系统提供：

- 加、减、乘、除运算指令：处理无符号或带符号的8位/16位二进制数的算术运算；
- 调整操作指令：进行压缩的或非压缩的十进制数的算术运算；
- 加法和减法运算指令：带符号数和无符号数的加法和减法的运算可以用同一条加法或减法指令来完成。
- 乘法和除法运算：分别设置无符号数和带符号数的乘、除法指令。

绝大部分算术运算指令都影响状态标志位。

加　法	
ADD	加法
ADC	带进位的加法
INC	增量
AAA	加法的ASCII调整
DAA	加法的十进制调整
除　法	
DIV	无符号数除法
IDIV	整数除法
AAD	除法的ASCII调整
CBW	把字节转换成字
CWD	把字转换成双字

减　法	
SUB	减法
SBB	带借位的减法
DEC	减量
NEG	取负
CMP	比较
AAS	减法的ASCII调整
DAS	减法的十进制调整
乘　法	
MUL	无符号数乘法
IMUL	整数乘法
AAM	乘法的ASCII调整

1、ADD加法指令(Addition)

- 指令格式： ADD 目的，源
- 指令功能：目的 ← 源十目的
- 标志位：根据相加结果，自动对6个状态标志OF、SF、ZF、AF、PF、CF进行置位/复位

对标志的解释(人为决定):

1) 两个加数都看成无符号数时，运算结果为9AH，即十进制数154。在这种情况下，SF和OF都没有意义，我们只关心ZF和CF标志，在BCD码运算或奇偶校验时才考虑AF或PF标志。
2) 两个加数都当成带符号数时，符号标志SF和溢出标志OF很重要，而进位标志CF没有意义。
3) 带符号数能表示的范围-128~+127，而本例中，两个正数94和60相加，其和为154，由于154超过了范围，即产生了溢出，OF＝1

ADD指令的操作数搭配共有如下6中方式

【例】

```
ADD     AX，BX;
                                    (AX)←(AX)+(BX)
ADD     SI，1234H;
                                    (SI)←(SI)+1234H
ADD     AL，[SI+BX]   ;
                                    (AL)←(AL)+((SI)+(BX))
ADD     BUF[DI]，DX   ;
                                    (BUF+(DI))←(BUF+(DI))+(DX)
ADD     BUF，56 ;
                                    (BUF)←(BUF)+56H
ADD     AL，BUF[BX] ;
                                    ←(AL)←(AL)+(BUF+(BX))
```

【课后习题5】若AX = 98ABH，BX = A8BCH。求执行指令ADD AX，BX后，AX与BX中的内容。并指出SF、ZF、AF、PF、CF和OF的状态

2、ADC带进位的加法指令(Addition with Carry)

- 指令格式： ADC 目的，源
- 指令功能：目的←源十目的十CF
- 标志位：根据相加结果，自动对6个状态标志OF、SF、ZF、AF、PF、CF进行置位/复位
- 注意：
  - 源操作数可以是寄存器、存储器、立即数；
    目的操作数：只能用寄存器、存储单元；
  - 源、目的操作数不能同时为存储器，且类型必须一致，均为字节或字；

3、INC增量指令(Increment)

- 指令格式：INC　目的
- 指令功能：目的←目的十1
- 标志位：只影响状态标志OF、SF、ZF、AF、PF，不影响CF标志
- 操作数的要求：通用寄存器、内存。
- 注意：
  - 这条指令主要用在循环程序中，对地址指针和循环计数器等进行修改；
  - 该指令只有一个操作数时，如果要使内存单元的内容增1，程序中必须说明该存储单元是字还是字节。

【例】

```
INC  BL;
INC  CX;
INC  [SI];


INC  BYTE  PTR[SI];
INC  WORD  PTR[SI] ;
```

4、AAA加法的ASCII调整指令(ASCII Adjust for Addition)

- 指令格式：AAA
- 指令功能：在用ADD或ADC指令对两个非压缩十进制数或ASCII码表示的十进制数作加法后，运算结果已存在AL 中，用此指令将AL寄存器中的运算结果调整为一位非压缩十进制数，仍保留在AL中。如果AF＝1，表示向高位有进位，则进到AH寄存器中。
- 注意：

非压缩十进制数的高4位为全0，低4位为十进制数字0~9。【例】将9表示成0000 1001

- 调整过程：

若AL 低4位 ＞ 9或AF ＝ 1，则：

①AL ← AL + 6

②用 与(∧) 操作将AL高4位清0

③AF置1，CF置1，AH ← AH + 1

否则，仅将AL寄存器的高4位清0。

- 标志位：会根据运算结果影响CF和AF，但对OF、PF、ZF、SF未定义。

5、DAA加法的十进制调整指令(Decimal Adjust for Addition)

- 指令格式：DAA
- 指令功能：将两个压缩BCD数相加后的结果调整为正确的压缩BCD数。
- 标志位：会根据运算结果影响CF、AF、PF、ZF、SF，但对OF未定义。
- 注意：

相加后的结果必须在AL 中，才能使用DAA指令。

❑ 调整过程:

①若AL的低4位 ＞ 9或AF ＝ 1，则

AL ← AL + 6，对低4位进行调整;

②若此时AL的高4位 ＞ 9或CF ＝ 1，则

AL←AL + 60H，对高4位进行调整，并使CF置1，否则CF置0。

3.4.2 减法指令

1、SUB减法指令(Subraction)

- 指令格式：SUB　　目的，源
- 指令功能：目的←目的－源
- 标志位：根据相减结果，自动对6个状态标志OF、SF、ZF、AF、PF、CF进行置位/复位

【例】

SUB BX，CX ；

(BX)←(BX)-(CX)

SUB SI，1234H；

(SI)←(SI)-1234H

SUB AL，[BP+BX] ；

(AL)←(AL)-((BP)+(BX))

SUB WORD PTR[DI]，1000H ；

对数据段中由DI指定的相邻两个存储单元中的16位数，减去1000H

SUB BUF，56H ；

(BUF)←(BUF)-56H

SUB AX，2000H；

(AX)←(AX)-2000H

2、SBB带借位的减法指令(Subtract with Borrow)

- 指令格式：SBB　　目的，源
- 指令功能：目的←目的 - 源 - CF　(SBB主要用于多字节减法中)
- 标志位：该指令会根据运算结果影响 OF、SF、ZF、AF、PF、CF进行置位/复位

【例】

SBB AX，2030H；　　(AX)←(AX)-2030H-CF

SBB DX，BX；　　(DX)←(DX)-(BX)-CF

SBB DX，[BX+20H]；

(DX)←(DX)-((BX)+20H)-CF

3、DEC减量指令(Decrement)

- 指令格式：DEC　目的
- 指令功能：目的←目的-1
- 标志位：DEC指操作对CF没有影响，而对OF、SF、ZF、AF、PF等会产生影响。

【例】

```
DEC   BX;

DEC   WORD PTR[BP] ;
```

4、NEG取负指令(Negate)

- 指令格式：NEG　目的
- 指令功能：目的←0－目的，即对目的操作数取负
- 标志位：该指令影响标志 OF、SF、ZF、AF、PF、CF

【例】

```
NEG  AX;

NEG   BYTE PTR[BX] ;
```

5、CMP比较指令(Compare)

- 指令格式：CMP　目的，源
- 指令功能：目的－源，结果不回送到目的操作数中，仅反映在标志位上。
- 标志位：与减法指令SUB相同，可根据运算结果影响OF、SF、ZF、AF、PF、CF等标志。
- 用途：用在比较两个数大小又不破坏原操作数的场合。

【例】

```
CMP     AX，BX;
CMP     BX，3000H;
CMP     BUF[SI]，AX;
```

减法指令小结：

(上述五种指令都做减法运算)

1)对于双操作数指令(SUB、SBB、CMP)：

源操作数可以是寄存器、存储器或立即数；

目的操作数可以是寄存器、存储器，但不能为立即数；

两个操作数不能同时为存储器。

2)对于单操作数指令(DEC、NEG)：

目的操作数可以是寄存器、存储器，但不能为立即数；

如果是存储器操作数，还必须说明其类型是字节还是字。

3) 运算之后

除DEC指令不影响CF标志外，其它均影响OF、SF、ZF、AF、PF和CF标志。

4) 在减法操作后

如果源操作数大于目的操作数，需要借位时，进位/借位标志CF将被置1。

6、AAS减法的ASCII调整指令(ASCII Adjust for Subtraction)

- 指令格式：AAS
- 指令功能：将AL寄存器中的运算结果调整为正确的非压缩十进制数之差，仍保留在AL 中。
- 标志位：影响CF和AF，对OF、PF、ZF、SF无定义。
- 使用前提：用SUB或SBB指令对两个非压缩十进制数或ASCII码表示的十进制数作减法，运算结果已存在AL。

❑ 调整过程:

若AL寄存器的低4位＞9或AF＝1，则:

①AL←AL－6，AF置1

②将AL寄存器高4位清零

③AH←AH－l，CF置l

否则，不需要调整

7、DAS减法的十进制调整指令(Decimal Adjust for Subtracion)

- 指令格式： DAS
- 指令功能：将AL 中两个压缩BCD数相减后的结果调整为正确的压缩BCD数。高4位和低4位分别进行调整。
- 标志位：会根据运算结果影响CF、AF、PF、ZF、SF，但对OF未定义。
- 使用前提：在两个压缩十进制数用SUB或SBB相减后，结果在AL 中。

- 调整过程：

①如果AL寄存器的低4位＞9或AF＝1，

则： AL ← AL－6，AF置1

②如果此时AL高半字节＞9或标志位CF＝I，

则：AL ← AL－60H，CF置1

【课后习题8】若AL=75H，BL=48H。

(1)求执行指令

ADD AL，BL

DAA

后AL=？标志CF=？AF=？并说明BCD码调整情况。

(2)若执行指令SUB AL，BL与DAS后，情况又如何?

3.4.3 乘法指令

1、MUL无符号数乘法指令(Multiply)

指令格式：MUL 源

指令功能：把源操作数和累加器中的数都当成无符号数，然后将两个数相乘，操作数可以是字节或字。

源操作数是一个字节：原操作数与累加器AL 中的内容相乘，乘积为双倍长的16位数，高8位送到AH，低8位送AL 。即：AX ← AL × 源

源操作数是一个字：原操作数与累加器AX的内容相乘，结果为32位数，高位字放在DX寄存器中，低位字放在AX寄存器中。即：

(DX，AX) ← AX × 源

- 操作数要求：可以是寄存器、存储单元，但不能是立即数；源操作数是存储单元时，必须在操作数前加B或W说明是字节还是字。
- MUL指令的操作数搭配有4种

2、IMUL整数乘法指令(Integer Multiply)

- 指令格式：IMUL 源
- 指令功能：把源操作数和累加器中的数都作为带符号数，进行相乘。

▲存放结果的方式与MUL相同：

源操作数为字节：与AL相乘，双倍长结果送到AX 中；

源操作数为字：与AX相乘，双倍长结果送到DX和AX 中，最后给乘积赋予正确的符号。

▲对标志位的影响：

- 乘积的高半部分不是低半部分的符号扩展(不是全0或全1)，则高位部分为有效位，表示它是积的一部分，于是置

  CF＝OF＝1;

- 结果的高半部分为全0或全1，表明它仅包含了符号位，那么使

  CF ＝ OF＝0。

- AF、PF、SF和ZF不定。

3、AAM乘法的ASCII调整指令(ASCII Adjust for Multiply)

- 指令格式：AAM
- 指令功能：对AL 中的两个非压缩十进制数相乘的乘积进行十进制数的调整，在AX 中得到正确的非压缩十进制数的乘积，高位在AH 中，低位在AL 中。
- 调整过程:

把AL寄存器内容除以10，商放在AH 中，余数在AL中。即:

AH←AL/10所得的商

AL←AH/10所得的余数

- 标志位：会影响PF、ZF、SF，但对OF、CF和AF无定义。
- 注意：
  - 两个ASCII码数相乘之前，先屏蔽掉每个数字的高半字节，使每个字节包含一个非压缩十进制数(BCD数)，再用MUL指令相乘，乘积放到AL寄存器中，然后用AAM指令进行调整。
  - 8086的指令系统中，十进制乘法运算不允许采用压缩十进制数，故调整指令仅此一条。

3.4.4 除法指令

1、DIV无符号数除法指令(Division，unsigned)

- 指令格式：DIV　源
- 指令功能：对两个无符号二进制数进行除法操作。

源操作数为字节：

16位被除数必须放在AX 中，8位除数为源操作数。

AL ← AX ／源(字节)的商

AH ← AX ／源(字节)的余数

若被除数只有8位，必须把它放在AL 中，并将AH清0。

源操作数为字:

32位被除数在(DX，AX)中，16位除数作源操作数。

AX ←(DX，AX) ／源(字)的商

DX ←(DX，AX) ／源(字)的余数

若被除数、除数都是16位，则将16位被除数送到AX 中，再将DX清0。

- 注意:

①源操作数可以是寄存器、存储单元;

②DIV指令执行后，所有标志位均无定义。

2、IDIV整数除法指令(Integer Division)

- 指令格式：IDIV 源
- 指令功能：对两个带符号二进制数进行除法操作(也称为带符号数除法)
- 注意：
  - 操作与DIV相同；
  - 商和余数都是带符号数，且规定余数的符号和被除数的相同；
  - 指令执行后，所有标志位均无定义。
  - 无论对(DIV)还是(IDIV)，都要注意溢出问题。

字节操作时:

被除数的高8位绝对值大于除数的绝对值，则产生溢出。

(对于无符号数，允许最大商为FFH；对于带符号数，允许商的范围为-127~+127，或-81H~+7FH);

字操作时:

被除数的高16位绝对值大于除数的绝对值，则产生溢出。

(对于无符号数，允许最大商为FFFFH；对于带符号数，允许商的范围为-32767~+32767，或-8001H~7FFFH)。

2、IDIV整数除法指令(Integer Division)

带符号数除法指令，字节操作时要求被除数为16位，字操作时要求被除数为32位；

如果被除数不满足这个条件，不能简单地将高位置0，而应该用符号扩展指令(Sign Extension)将被除数转换成除法指令所要求的格式。

3、AAD除法的ASCII调整指令(ASCII Adjust for Division)

- 指令格式：AAD
- 指令功能：在做除法前，把BCD码转换成二进制数。
- 标志位：会影响PF、ZF、SF，但对OF、CF和AF无定义。
- 调整过程：

  AL←AH ×10＋AL;

  AH←00

- 注意：AX 中的两位非压缩格式的BCD数除以一个非压缩的BCD数前，要先用AAD指令把AX 中的被除数调整成二进制数，并存到AL 中，才能用DIV指令进行运算。

【课后习题9】设X、Y、R、S、Z均为16位无符号数的变量。按已给定的表达式Z(X+Y)/(R-S)→Z，有程序如下，试在空格处填入适当的指令（注：在加减过程中均无进位和借位）。

4、CBW把字节转换为字指令(Convert Byte to Word)

- 指令格式：CBW
- 指令功能：把寄存器AL 中字节的符号位扩充到AH的所有位。（AH被称为AL的符号扩充）

5、CWD把字转换成双字指令(Convert Word to Double Word)

- 指令格式：CWD
- 指令功能：把AX 中字的符号值扩充到DX寄存器的所有位。
- 扩展方法：

若AX的D_{15} ＝0，则DX ←0000H；

若AX的D_{15} ＝1，则DX ←FFFFH。

- 注意：CWD指令执行后，也不影响标志位。