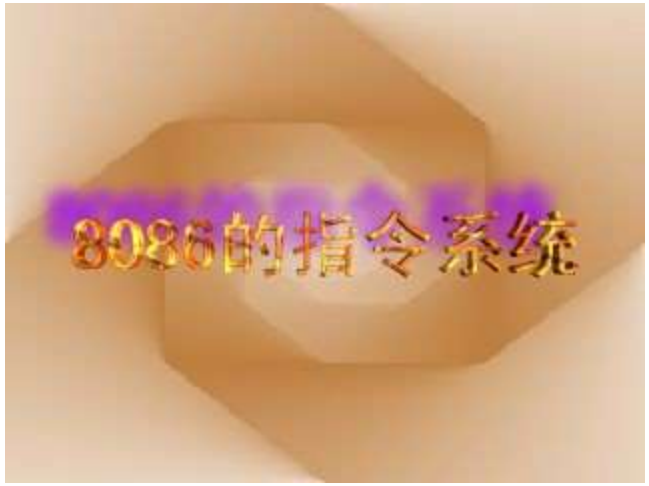
教学提示

全面而准确地理解每条指令的功能和应用，是编写汇编语言程序的关键



逐个展开指令

3.3 数据传送类指令

- 数据传送是计算机中最基本、最重要的一种操作
- 传送指令也是最常使用的一类指令
- 传送指令把数据从一个位置传送到另一个位置
- 除标志寄存器传送指令外，均不影响标志位
- 重点掌握

MOV XCHG XLAT PUSH POP LEA

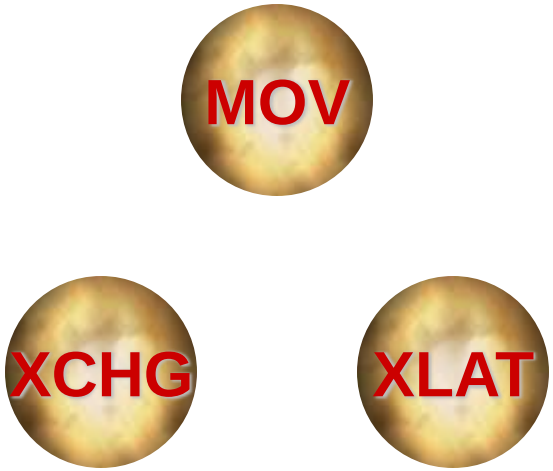
3.3.1 通用数据传送指令

- 提供方便灵活的通用传送操作
- 有3条指令

MOV

XCHG

XLAT

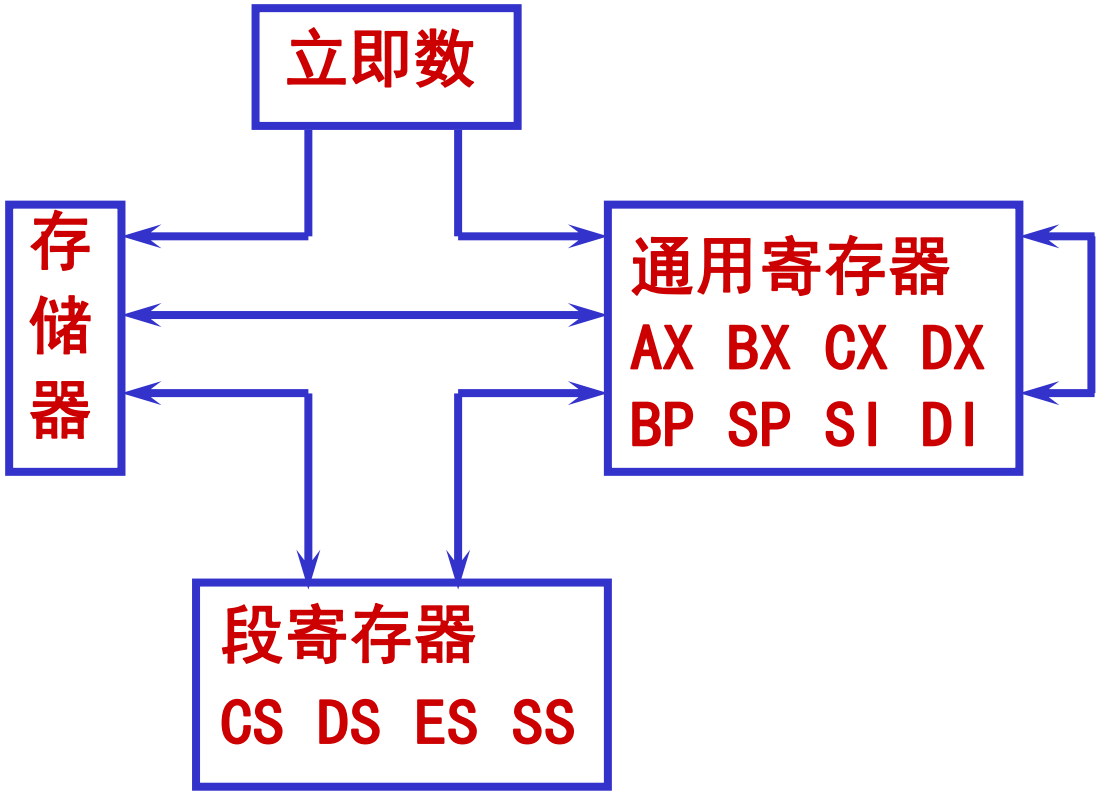


传送指令MOV（move）

- 把一个字节或字的操作数从源地址传送至目的地址

```
MOV reg/mem,imm                    例题3.1
MOV reg/mem/seg,reg                例题3.2
MOV reg/seg,mem                    例题3.3
MOV reg/mem,seg                    例题3.5
                ; 段寄存器送寄存器或主存
```

MOV指令传送功能



MOV也并非任意传送

非法传送种种

- 两个操作数的类型不一致 示例
  - 例如源操作数是字节，而目的操作数是字；或相反
- 两个操作数不能都是存储器 示例
  - 传送指令很灵活，但主存之间的直接传送却不允许
- 段寄存器的操作有一些限制 示例
  - 段寄存器属专用寄存器，对他们的操作能力有限

交换指令XCHG（exchange）

- 把两个地方的数据进行互换

```
XCHG reg,reg/mem

                    ; reg ↔ reg/mem
```

- 寄存器与寄存器之间对换数据 例题3.6
- 寄存器与存储器之间对换数据 例题3.7
- 不能在存储器与存储器之间对换数据

换码指令XLAT（translate）

- 将BX指定的缓冲区中、AL指定的位移处的一个字节数据取出赋给AL

```
XLAT                    ; al←ds:[bx+al]
```

例题3.8

- 换码指令执行前：

  在主存建立一个字节量表格，内含要转换成的目的代码

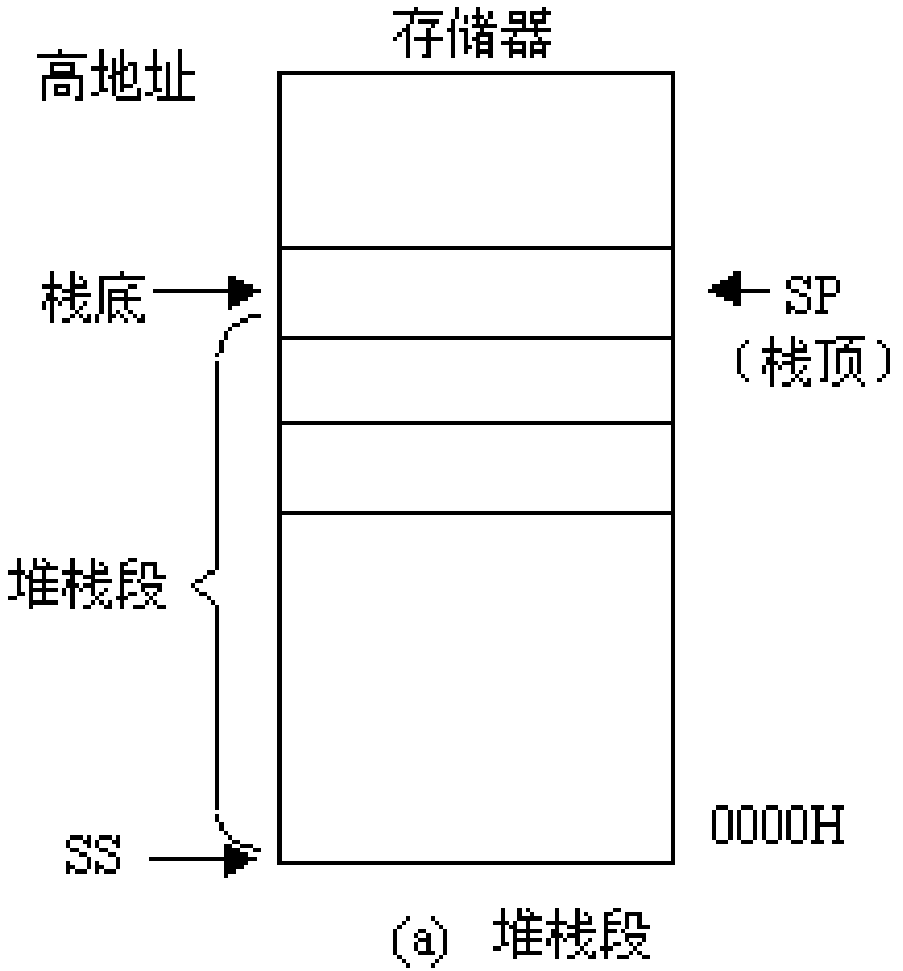
  表格首地址存放于BX，AL存放相对表格首地址的位移量

- 换码指令执行后：

  将AL寄存器的内容转换为目标代码

3.3.2 堆栈操作指令

- 堆栈是一个“后进先出FILO”（或说“先进后出FILO”）的主存区域，位于堆栈段中；SS段寄存器记录其段地址
- 堆栈只有一个出口，即当前栈顶；用堆栈指针寄存器SP指定
- 栈顶是地址较小的一端（低端），栈底不变



堆栈的操作

- 堆栈只有两种基本操作：进栈和出栈，对应两条指令PUSH和POP

PUSH

；进栈指令先使堆栈指针SP减2，然后把一个字操作数存入堆栈顶部

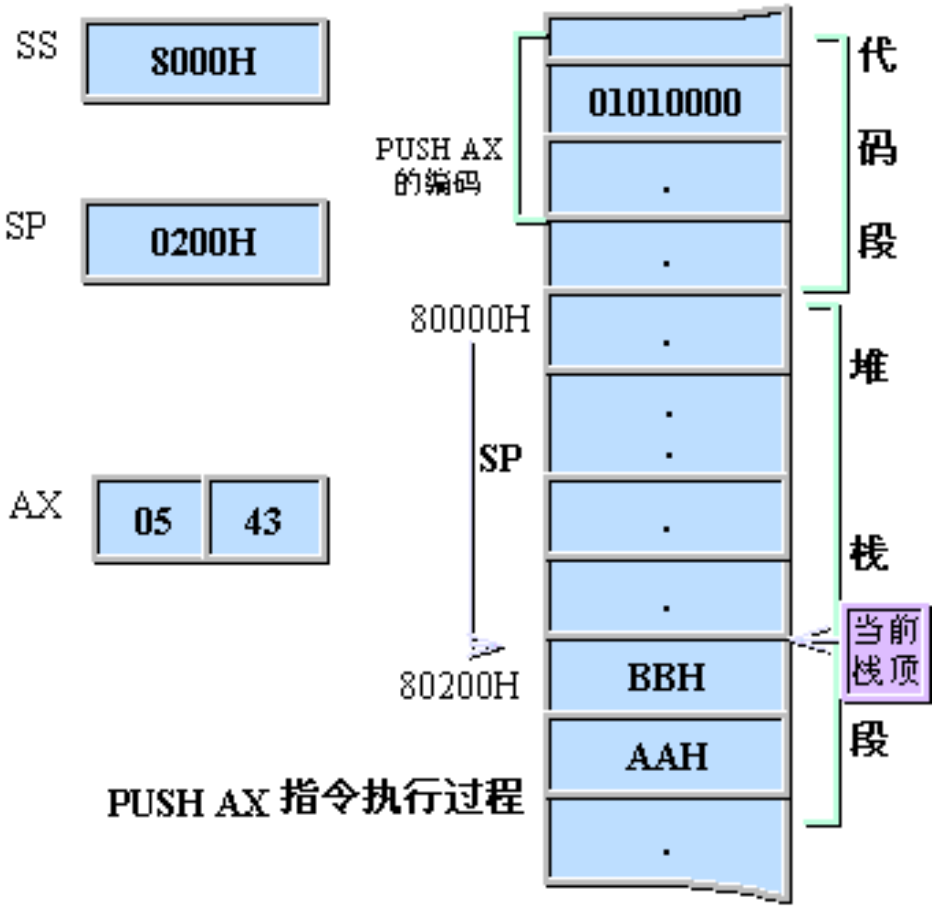
POP

；出栈指令把栈顶的一个字传送至指定的目的操作数，然后堆栈指针SP加2

进栈指令PUSH


```
PUSH r16/m16/seg
; SP←SP－2
; SS:[SP]←r16/m16/seg
```

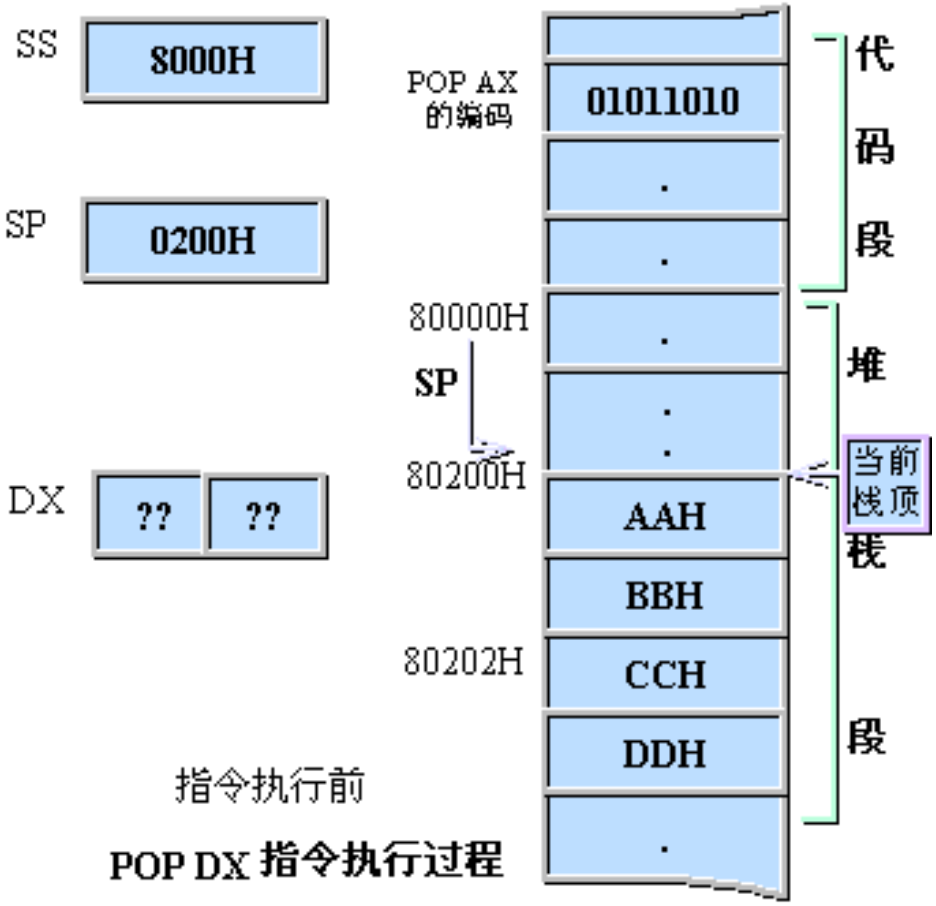
```
push ax
push [2000h]
```



出栈指令POP


```
POP r16/m16/seg
; r16/m16/seg←SS:[SP]
; SP←SP+2
```

```
pop ax
pop [2000h]
```



堆栈的特点

- 堆栈操作的单位是字，进栈和出栈只对字量
- 字量数据从栈顶压入和弹出时，都是低地址字节送低字节，高地址字节送高字节
- 堆栈操作遵循先进后出原则，但可用存储器寻址方式随机存取堆栈中的数据
- 堆栈常用来
  - 临时存放数据
  - 传递参数
  - 保存和恢复寄存器

例题3.11

3.3.3 标志寄存器传送指令

- 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

标志低字节进出AH指令

LAHF

；AH←FLAGS的低字节

- LAHF指令将标志寄存器的低字节送寄存器AH
- SF/ZF/AF/PF/CF状态标志位分别送入AH的第7/6/4/2/0位，而AH的第5/3/1位任意

SAHF

；FLAGS的低字节←AH

- SAHF将AH寄存器内容送FLAGS的低字节
- 用AH的第7/6/4/2/0位相应设置SF/ZF/AF/PF/CF标志

标志寄存器进出堆栈指令

PUSHF

; SP←SP－2

; SS:[SP]←FLAGS

- PUSHF指令将标志寄存器的内容压入堆栈，同时栈顶指针SP减2

POPF

; FLAGS←SS:[SP]

; SP←SP＋2

- POPF指令将栈顶字单元内容送标志寄存器，同时栈顶指针SP加2

例题3.12

3.3.4 地址传送指令

- 地址传送指令将存储器单元的逻辑地址送至指定的寄存器
  - 有效地址传送指令 **LEA**
  - 指针传送指令 **LDS和LES**
- 注意不是获取存储器单元的内容

有效地址传送指令LEA（load EA）

- 将存储器操作数的有效地址传送至指定的16位寄存器中

```
LEA r16,mem
          ; r16←mem的有效地址EA
```

例题3.13

指针传送指令

LDS r16,mem

; r16←mem,

; DS←mem+2

- LDS指令将主存中mem指定的字送至r16，并将mem的下一字送DS寄存器

LES r16,mem

; r16←mem,

; ES←mem+2

- LES指令将主存中mem指定的字送至r16，并将mem的下一字送ES寄存器

例题3.14

3.3.5 输入输出指令

- 8086通过输入输出指令与外设进行数据交换；呈现给程序员的外设是端口（Port）即I/O地址
- 8086用于寻址外设端口的地址线为16条，端口最多为2^{16}＝65536（64K）个，端口号为0000H～FFFFH
- 每个端口用于传送一个字节的外设数据

输入输出寻址方式

- 8086的端口有64K个，无需分段，设计有两种寻址方式
  - 直接寻址：只用于寻址00H～FFH前256个端口，操作数i8表示端口号
  - 间接寻址：可用于寻址全部64K个端口，DX寄存器的值就是端口号
- 对大于FFH的端口只能采用间接寻址方式

输入指令IN

- 将外设数据传送给CPU内的AL/AX

演示

```
IN AL,i8
; 字节输入：AL←I/O端口（i8直接寻址）
IN AL,DX
; 字节输入：AL←I/O端口（DX间接寻址）
IN AX,i8
; 字输入：AX←I/O端口（i8直接寻址）
IN AX,DX
; 字输入：AX←I/O端口（DX间接寻址）
```

例题3.15

输出指令OUT

- 将CPU内的AL/AX数据传送给外设

演示

```
OUT i8,AL
; 字节输出：I/O端口←AL（i8直接寻址）
OUT DX,AL
; 字节输出：I/O端口←AL（DX间接寻址）
OUT i8,AX
; 字输出：I/O端口←AX（i8直接寻址）
OUT DX,AX
; 字输出：I/O端口←AX（DX间接寻址）
```

例题3.16

3.4 算术运算类指令

- 四则运算是计算机经常进行的一种操作。算术运算指令实现二进制（和十进制）数据的四则运算
- 请注意算术运算类指令对标志的影响
  - 掌握：ADD/ADC/INC、SUB/SBB/DEC/NEG/CMP
  - 熟悉：MUL/IMUL、DIV/IDIV
  - 理解：CBW/CWD、DAA/DAS、 AAA/AAS/AAM/AAD

加法指令ADD

- ADD指令将源与目的操作数相加，结果送到目的操作数
- ADD指令按状态标志的定义相应设置

例题3.17

```
ADD reg,imm/reg/mem
                ; reg←reg＋imm/reg/mem
ADD mem,imm/reg
                ; mem←mem＋imm/reg
```

带进位加法指令ADC

- ADC指令将源与目的操作数相加，再加上进位CF标志，结果送到目的操作数
- ADC指令按状态标志的定义相应设置
- ADC指令主要与ADD配合，实现多精度加法运算

例题3.18

```
ADC reg,imm/reg/mem
          ; reg←reg＋imm/reg/mem＋CF
ADC mem,imm/reg
          ; mem←mem＋imm/reg＋CF
```

增量指令INC（increment）

- INC指令对操作数加1（增量）
- INC指令不影响进位CF标志，按定义设置其他状态标志

```
INC reg/mem
; reg/mem←reg/mem+1
```

```
inc bx
inc byte ptr [bx]
```

减法指令SUB（subtract）

- SUB指令将目的操作数减去源操作数，结果送到目的操作数
- SUB指令按照定义相应设置状态标志

例题3.17

```
SUB reg,imm/reg/mem
                ; reg←reg－imm/reg/mem
SUB mem,imm/reg
                ; mem←mem－imm/reg
```

带借位减法指令SBB

- SBB指令将目的操作数减去源操作数，再减去借位CF（进位），结果送到目的操作数。
- SBB指令按照定义相应设置状态标志
- SBB指令主要与SUB配合，实现多精度减法运算

例题3.18

```
SBB reg,imm/reg/mem
          ; reg←reg－imm/reg/mem－CF
SBB mem,imm/reg
          ; mem←mem－imm/reg－CF
```

减量指令DEC（decrement）

- DEC指令对操作数减1（减量）
- DEC指令不影响进位CF标志，按定义设置其他状态标志

```
DEC reg/mem
; reg/mem←reg/mem－1
```

- INC指令和DEC指令都是单操作数指令
- 主要用于对计数器和地址指针的调整

求补指令NEG（negative）

- NEG指令对操作数执行求补运算：用零减去操作数，然后结果返回操作数
- 求补运算也可以表达成：将操作数按位取反后加1
- NEG指令对标志的影响与用零作减法的SUB指令一样

```
NEG reg/mem
; reg/mem←0－reg/mem
```

例题3.19

比较指令CMP（compare）

- CMP指令将目的操作数减去源操作数，按照定义相应设置状态标志
- CMP指令执行的功能与SUB指令，但结果不回送目的操作数

例题3.20

```
CMP reg,imm/reg/mem
                ; reg－imm/reg/mem
CMP mem,imm/reg
                ; mem－imm/reg
```

3.4.3 乘法指令

```
MUL r8/m8
; 无符号字节乘法
; AX←AL×r8/m8
MUL r16/m16
; 无符号字乘法
; DX.AX←AX×r16/m16
```

```
IMUL r8/m8
; 有符号字节乘法
; AX←AL×r8/m8
IMUL r16/m16
; 有符号字乘法
; DX.AX←AX×r16/m16
```

说明

例题3.21

3.4.4 除法指令

说明

DIV r8/m8 ；无符号字节除法：

AL←AX÷r8/m8的商，Ah←AX÷r8/m8的余数

DIV r16/m16 ；无符号字除法：

；AX←DX.AX÷r16/m16的商，DX←DX.AX÷r16/m16的余数

例题3.22

IDIV r8/m8 ；有符号字节除法：

AL←AX÷r8/m8的商，Ah←AX÷r8/m8的余数

IDIV r16/m16 ；有符号字除法：

；AX←DX.AX÷r16/m16的商，DX←DX.AX÷r16/m16的余数

3.4.5 符号扩展指令

不影响标志位

```
CBW；AL的符号扩展至AH
；如AL的最高有效位是0，则AH＝00
；AL的最高有效位为1，则AH＝FFH。AL不变
```

```
CWD；AX的符号扩展至DX
；如AX的最高有效位是0，则DX＝00
；AX的最高有效位为1，则DX＝FFFFH。AX不变
```

- 什么是符号扩展

例题3.23

- 符号扩展指令常用于获得倍长的数据

例题3.24

3.4.6 十进制调整指令

- 十进制数调整指令对二进制运算的结果进行十进制调整，以得到十进制的运算结果
- 分成压缩BCD码和非压缩BCD码调整

- 压缩BCD码就是通常的8421码；它用4个二进制位表示一个十进制位，一个字节可以表示两个十进制位，即00～99

- 非压缩BCD码用8个二进制位表示一个十进制位，只用低4个二进制位表示一个十进制位0～9，高4位任意，通常默认为0

BCD码（Binary Coded Decimal）

- 二进制编码的十进制数：一位十进制数用4位二进制编码来表示
- 8086支持压缩BCD码和非压缩BCD码的调整运算

真值	8	64
二进制编码	08H	40H
压缩BCD码	08H	64H
非压缩BCD码	08H	0604H

压缩BCD码加、减调整指令

（ADD AL,i8/r8/m8）

（ADC AL,i8/r8/m8）

DAA

例题3.25a

；AL←将AL的加和调整为压缩BCD码

（SUB AL,i8/r8/m8）

（SBB AL,i8/r8/m8）

DAS

例题3.25b

；AL←将AL的减差调整为压缩BCD码

- 使用**DAA**或**DAS**指令前，应先执行以**AL**为目的操作数的加法或减法指令
- **DAA**和**DAS**指令对**OF**标志无定义，按结果影响其他标志，例如**CF**反映压缩**BCD**码相加或减的进位或借位状态

例题3.26

非压缩BCD码加、减调整指令

（ADD AL,i8/r8/m8）

（ADC AL,i8/r8/m8）

AAA

例题3.27a

；AL←将AL的加和调整为非压缩BCD码

；AH←AH＋调整的进位

（SUB AL,i8/r8/m8）

（SBB AL,i8/r8/m8）

AAS

例题3.27b

；AL←将AL的减差调整为非压缩BCD码

；AH←AH－调整的借位

- 使用**AAA**或**AAS**指令前，应先执行以**AL**为目的操作数的加法或减法指令
- **AAA**和**AAS**指令在调整中产生了进位或借位，则AH要加上进位或减去借位，同时**CF=AF=1**，否则**CF=AF=0**；它们对其他标志无定义

非压缩BCD码乘、除调整指令

例题3.27c

```
（MUL r8/m8）
AAM
；AX←将AX的乘积调整为非压缩BCD码
```

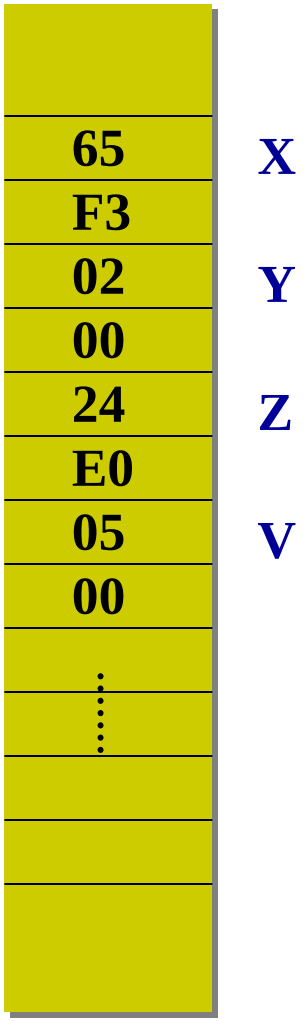
例题3.27d

```
AAD
；AX←将AX中非压缩BCD码扩展成二进制数
（DIV r8/m8）
```

- **AAM指令跟在字节乘MUL之后，将乘积调整为非压缩BCD码**
- **AAD指令跟在字节除DIV之前，先将非压缩BCD码的被除数调整为二进制数**
- **AAM和AAD指令根据结果设置SF、ZF和PF，但对OF、CF和AF无定义**

习题3.19

- 设X、Y、Z、V均为16位带符号数，分别存放在X、Y、Z、V存储单元中，阅读如下程序段，得出它的运算公式，并说明运算结果存于何处？



习题3.19：算术运算1

```
mov  ax,X
imul  Y              ; DX.AX=X×Y
mov  cx,ax
mov  bx,dx           ; BX.CX=X×Y
mov  ax,Z
cwd
add  cx,ax
adc  bx,dx           ; BX.CX=X×Y+Z
```

习题3.19：算术运算2

```
sub cx,540
sbb bx,0
; BX.CX=X×Y+Z-540
mov ax,V
cwd
sub ax,cx
sbb dx,bx
; DX.AX=V-(X×Y+Z-540)
idiv X
; DX.AX=(V-(X×Y+Z-540))÷X
```

第3章 教学要求（2）

1. 熟悉8086的基本参数、堆栈工作原理、指令对标志的影响、符号扩展的含义、压缩和非压缩BCD的格式
2. 掌握基本指令：MOV／XCHG／XLAT、PUSH／POP、LEA；CLC／STC／STC、CLD／STD；ADD／ADC／INC、SUB／SBB／DEC／CMP／NEG、CBW／CWD

第3章 教学要求（3）

3. 熟悉特色指令：IN / OUT ；CLI / STI；MUL / IMUL、DIV / IDIV、DAA / DAS、AAA / AAS
4. 了解不常使用的指令：LAHF / SAHF / PUSHF / POPF、LDS / LES；AAM / AAD；


课间休息