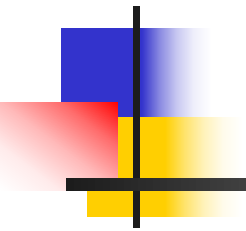


第三章

8086/8088 CPU的指令系统

3.1 8086/8088 CPU指令格式

OPRD ：泛指各种类型操作数(Operand)，如 JMP OPRD；这里的 OPRD 可以是偏移地址

MEM ：主要指存储器操作数(Memory)，但在许多情况下，常用 OPRD 替代 MEM

REG ：寄存器类型操作数(Register)

DST ：目标操作数(Destination)

SRC ：源操作数(Source)

LABEL ：程序标号，它的数值就是某指令的偏移地址

DISP8 或 DISP16 ：8 位或 16 位偏移量（Displacement），或用符号地址来表示

DATA ：8 位或 16 位立即数

3.1 8086/8088 CPU指令格式

PORT ： 输入输出端口（Port）,也可直接用数字或表达式来表示

EA ： 有效地址（Effective Address）

（ ） ： 表示寄存器的内容，如寄存器 AX 中的内容是 34H，可表为（AX）=34H

[ ] ： 表示存储器的内容，如存储器地址为 1000H 的内容是 6AH，即[1000H]=6AH

CPU 指令的二进制码的表述格式通常如下（方括号表示其中的内容可选）：

操作码 [操作数] ……，[操作数]

对应的汇编指令的表述格式通常如下：

指令助记符 [操作数]，…… [操作数] ；注释

3.1 8086/8088 CPU指令格式

表 3-1 8086/8088 指令的一般格式

操作码			寻址方式编码			操作数 1		操作数 2	
1 字节			1 字节			低位字节	高位字节	低位字节	高位字节
7~2	1	0	7 6	5 4 3	2 1 0	偏移量或操	偏移量或操	立即数低	立即数高
OP	D	W	MOD	REG	R/M	作数低字节	作数高字节	位字节	位字节

3.2 8086/8088 CPU指令的寻址方式

3.2.1 立即寻址

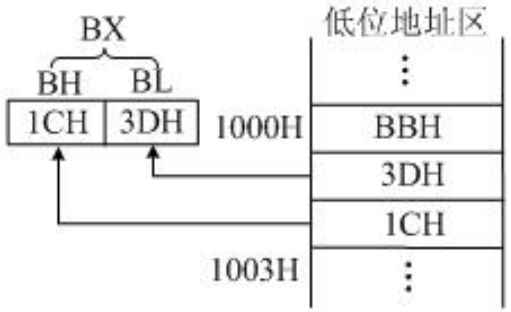


图 3-1 指令执行过程

```
MOV   AX，DATA        ；DATA 可以是段名
MOV   BX，C           ；C 可以是定义过的常量
MOV   CX，8+6-E2      ；8+6-E2 为表达式，E2 是常数
```

3.2 8086/8088 CPU指令的寻址方式

3.2.2 寄存器寻址

```
MOV   AL，CH ；将 CX 寄存器的高 8 位 CH 的内容送至 AX 的低 8 位 AL 中，CH 中的内容不变
MOV   DS，AX ；将源操作数 AX 寄存器中的内容送至目的操作数 DS 中，AX 中的内容不变
INC   CX     ；将 CX 寄存器中的内容加 1，仍然送回 CX 中
ADD   DX，CX ；将寄存器 DX 和 CX 中的 16 位数相加后送 DX，CX 中的内容不变
```

3.2 8086/8088 CPU指令的寻址方式

3.2.3 存储器寻址

EA= 基址 + 变址 + 位移量

$$EA = \left\{\begin{matrix}BX\\BP\end{matrix}\right\} + \left\{\begin{matrix}SI\\DI\end{matrix}\right\} + \left\{\begin{matrix}8位\\16位\end{matrix}\right\} \quad 偏移量 \}$$

物理地址(PA)=（段寄存器）*16 + 有效地址（EA） ；这里十进制 16 即 10H

3.2 8086/8088 CPU指令的寻址方式

1. 直接寻址

【例 3-1】设数据段寄存器 (DS)=3000H，数据 5678H 在此段的偏移地址是 2000H。这意味着 78H 的偏移地址是 2000H，56H 的偏移地址是 2001H。用图示显示以下指令：

MOV AX，[2000H] ；此指令的指令代码是：A10020H

的执行过程。

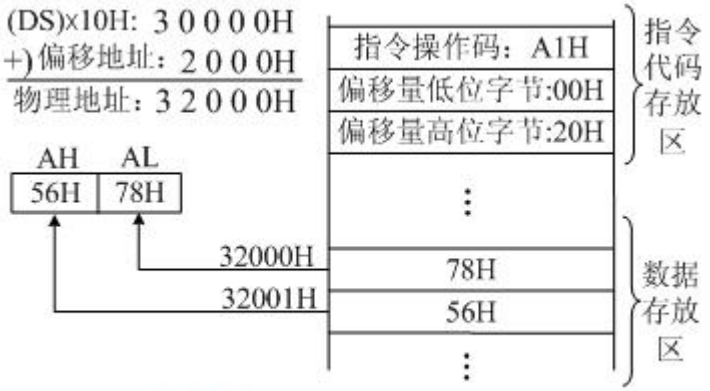


图 3-2 例 3-1 的执行过程

3.2 8086/8088 CPU指令的寻址方式

2. 寄存器间接寻址

① 寄存器SI、DI和BX的间接寻址

【例 3-2】 对于寄存器间接寻址的存储器操作数传送指令：

MOV AX，[SI]　　；此指令的指令代码是：8B04H

其源操作数物理地址的算式是： PA＝(DS)×10H＋(SI)

若假设(DS)＝5000H，(SI)＝3000H，

执行过程如图 3-3 所示,执行结果(AX)＝B6A5H。

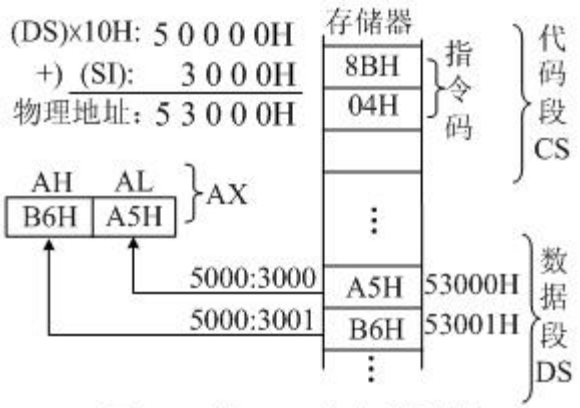


图 3-3 例 3-2 的执行过程

② 寄存器BP间接寻址

3.2 8086/8088 CPU指令的寻址方式

3. 寄存器相对寻址

① 若用**SI**、**DI**和**BX**作为变址寄存器，则段寄存器为数据段。

② 若用**BP**作为变址，段寄存器**SS**的堆栈段内容默认为寻址段。

3.2 8086/8088 CPU指令的寻址方式

4. 基址加变址寻址

```
MOV   AX，[BX][SI]
MOV   AX，[BX][DI]
MOV   AX，[BP][SI]
MOV   AX，[BP][DI]
```

```
MOV   AX，[BX+SI]
MOV   AX，[BX+DI]
MOV   AX，[BP+SI]
MOV   AX，[BP+DI]
```

【例 3-3】对于指令 MOV　AX,[BP][SI],假设(SS)＝5000H,(SI)＝3000H,(BP)＝4000H。另设此指令所在存储单元首地址是（18A6:0100），即(CS)=18A6H，此条指令所在存储器的代码段偏移地址是 100H。指令 MOV　AX，[BP][SI] 的代码是 8B02H。于是此指令的操作数的有效地址 EA=3000H+4000H=7000H。此指令表示将物理地址 PA=50000H+EA=57000H 为首地址的一个字的内容移送到存储器 AX 中。具体执行过程如图 3-4 所示，执行结果(AX)＝A354H。

3.2 8086/8088 CPU指令的寻址方式

4. 基址加变址寻址

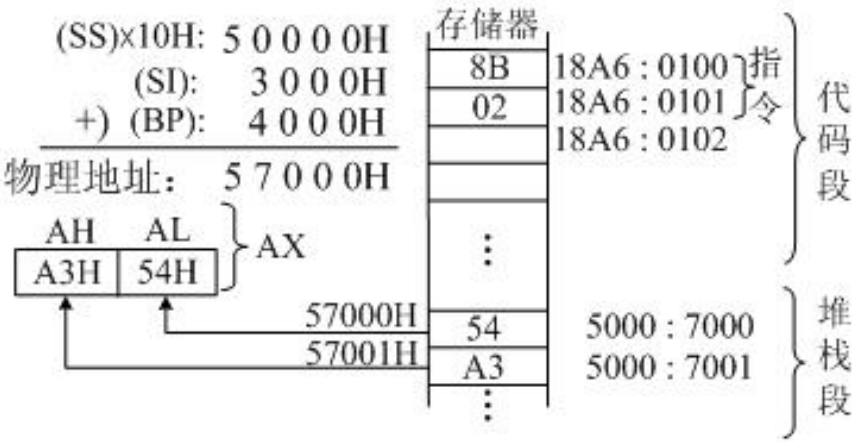


图 3-4 例 3-3 指令的执行过程

3.2 8086/8088 CPU指令的寻址方式

5. 相对基址加变址寻址

【例 3-4】对于指令： MOV AX，127H[BX][SI] ，计算其物理地址(操作数的首地址)。此指令也可表为 MOV AX，[BX+SI+0127H] 。此指令的代码是 8B802701。

这里设常数偏移量 CNDT＝0127H，(BX)＝0C345H，(SI)＝2500H，(DS)＝4000H；并设此指令所在存储单元首地址是（13A6:0100），即(CS)=13A6H，此条指令所在存储器的代码段偏移地址是 100H。于是操作数的物理地址是：

PA＝(DS)×10H＋EA=(DS)×10H + CNDT + (BX) + (SI)

＝40000H＋127H＋C345H＋2500H＝4E96CH

执行过程如图 3-5 所示，执行结果(AX)＝3DB2H。

3.2 8086/8088 CPU指令的寻址方式

5. 相对基址加变址寻址

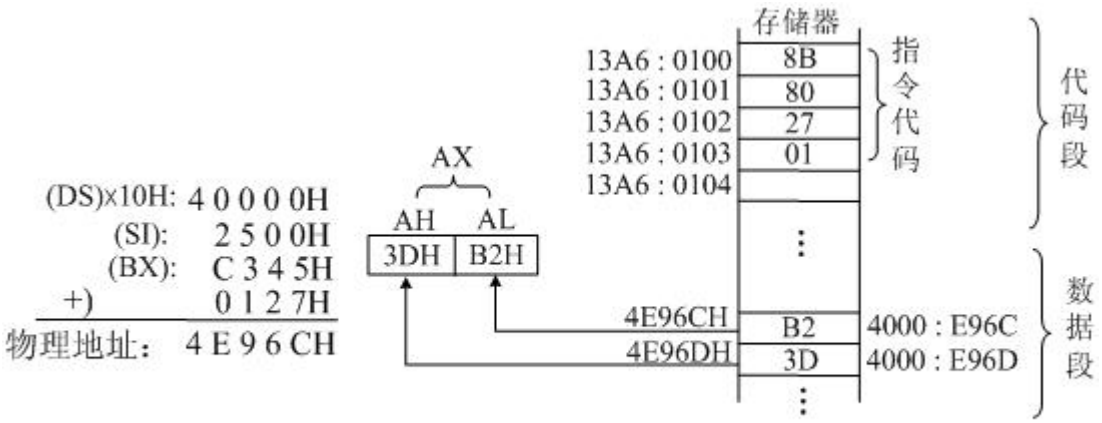


图 3-5 例 3-4 指令的执行过程

```
MOV   AX，[BX＋CNDT][SI]
MOV   AX，[BX＋SI＋CNDT]
MOV   AX，CNDT [BX][SI]
```

3.2 8086/8088 CPU指令的寻址方式

3.2.4 端口寻址

1. 直接寻址

```
IN   AL，n
```

2. 间接寻址

```
OUT   DX，AL
```

3.3 DEBUG的使用方法

3.3.1 DEBUG使用方法

1. 启动与退出DEBUG

2. DEBUG命令

3.输入汇编指令或程序的命令A

4. 反汇编内存中的程序

5. 设置断点全速运行内存中的程序

3.3 DEBUG的使用方法

3.3.1 DEBUG使用方法

6.单步运行内存中的程序

7. 查看与修改寄存器内容

8. 查看与修改存储器内容

9. 内存块复制与填数

10. 数据块输出与输入

11. 程序段落输出与输入

3.3 DEBUG的使用方法

3.3.2 DEBUG使用示例

【例 3-5】首先根据 3.3.1 节进入 DEBUG 界面

```
-R                              ；键入 R 后回车，首先了解此时所有寄存器的原始内容
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=1814  ES=1814  SS=1814  CS=1814  IP=0100   NV UP EI PL NZ NA PO NC
-R BX                           ；键入“R  BX”后回车，了解寄存器 BX 的内容
BX 0000                         ；BX 的初始内容是 0000H
:0200                           ；修改 BX 的内容为 200H 后回车
-R SI                           ；深色文字是键入的内容
SI 0000
:0320                           ；修改 SI 的内容为 320H，回车
-R DS
DS 1814
:1820                           ；修改 DS 的内容为 1820H，回车
-E 1820:622                     ；了解地址为 1820:622 存储器单元的数据。回车
1820:0622  00.AB                ；显示出 1820:622 单元的数据为 00H，键入 AB 更改为 ABH。回车
```

3.3 DEBUG的使用方法

3.3.2 DEBUG使用示例

```
- E 1820:623                    ；了解地址为 1820:623 存储器单元的数据。回车
1820:0623  00.CD                ；显示出 1820:623 单元的数据为 00H，键入 CD 更改为 CDH。回车
- D 1820:622  62F               ；了解地址为 1820:622 至 1820:62F 存储器单元的数据。回车
1820:0620      AB CD 00 00 00 00-00 00 00 00 00 00 00 00      ；指定存储器单元数据显示
- A 200                         ；准备于 CS 指定段的 200H 地址处开始输入汇编指令。回车
1814:0200 MOV AX, [BX+SI+102]  ；键入指令 MOV AX, [BX+SI+102H]
1814:0204                       ；不准备输入其它指令了，回车
- U 200 204                     ；反汇编地址为 1814:200 至 1814:204 单元的指令码
1814:0200 8B800201      MOV     AX,[BX+SI+0102]     ；反汇编显示，指令码是 8B800201H
1814:0204 0000          ADD     [BX+SI],AL
- T=200                         ；单步执行地址为 200H 单元开始的一条指令，并显示执行结果
AX=CDAB  BX=0200  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0320  DI=0000
DS=1820  ES=1814  SS=1814  CS=1814  IP=0204  NV UP EI PL NZ NA PO NC
1814:0204 0000  ADD  [BX+SI],AL  DS:0520=68  ；注意，AX 的内容变为 0CDABH
```

3.4 指令系统

3.4.1 数据传送类指令

1. 基本传送指令

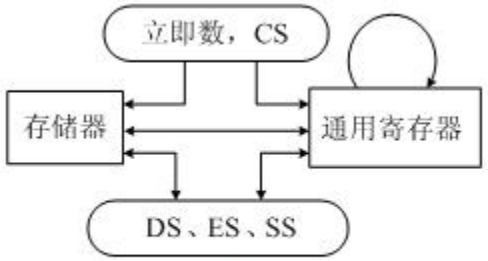


图 3-6 数据传送规则图

```
MOV   DST，SRC        ；   指令功能是   DST ← SRC

MOV   BX，CX          ；字传送指令，功能：BX←(CX)
MOV   AH，BL          ；字节传送指令，功能：AH←(BL)
MOV   [1200H]，BL     ；字节传送指令，默认段地址在 DS 中，功能：DS:[1200H] ←(BL)
```

【例 3-6】若把段内地址偏移量为 B3 的存储单元中的内容传送至同一段内的地址为 A3 的存储单元中去，一条 MOV 指令是不能直接完成这个传送任务的，但可通过寄存器来间接完成传送，例如：

```
MOV   AL，[00B3]
MOV   [00A3]，AL
```

段寄存器间数据的传送也可以采用类似方法。

3.4 指令系统

3.4.1 数据传送类指令

2. 交换指令　　XCHG　DST，SRC　　；　(DST) ↔ (SRC)

```
XCHG   AX，BX          ；寄存器 AX 和 BX 内的数据进行交换: (AX) ↔ (BX)
XCHG   AX，[3456H]     ；与数据段首地址为 3456H 的存储器单元的数据进行交换，即：
                       ；  (AX) ↔[(DS) : 3456H]
XCHG   [DI+BX]，AX     ；AX 与数据段首地址为(DI)+(BX)的存储器单元的数据进行交换，即：
                       ；[(DS) : (DI)+(BX)] ↔ (AX)
XCHG   [BP]，AL        ；AL 与堆栈段偏移地址为(BP)的存储器单元的数据进行交换，即：
                       ；[(SS) : (BP)] ↔ (AL)
```

3.4 指令系统

3.4.1 数据传送类指令

3. 查表指令(换码指令)

XLAT ；指令功能：AL ← [(DS) : (BX+AL)]

【例 3-7】假设(BX)=0040H，(AL)=0FH，(DS)=4000H，存储器中数据存储格式如图 3-7 所示，执行指令 XLAT 后，把物理地址:

PA=40000H+0040H+0FH=4004FH

的内容送 AL 寄存器，所以指令执行后(AL)=C8H。

图中显示 BX 中的偏移量指向表格的顶部地址。

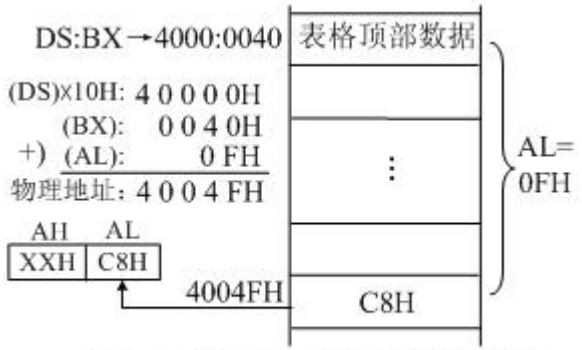


图 3-7 例 3-11 指令的执行过程

3.4 指令系统

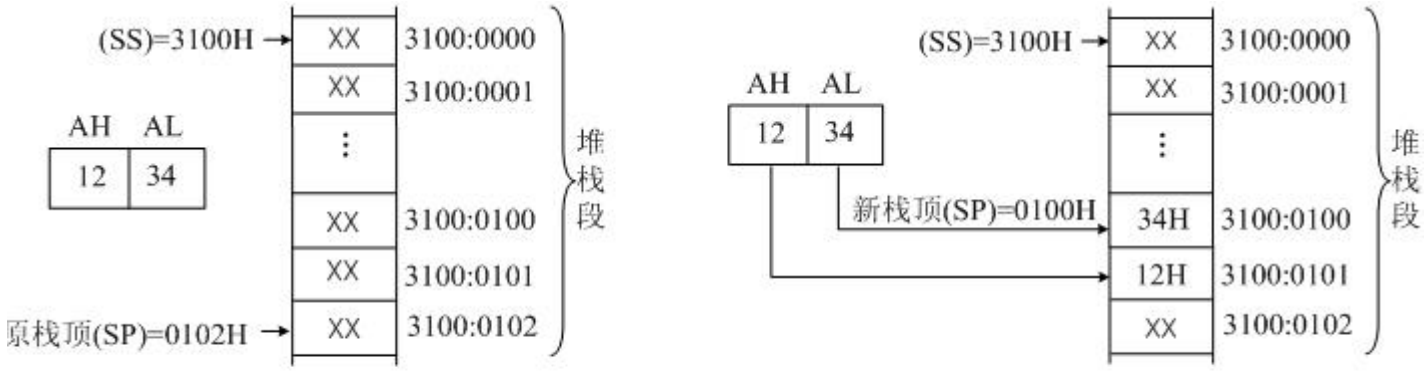
3.4.1 数据传送类指令

4. 堆栈传送指令

【例 3-8】 已知(SS)=3100H，原栈顶(SP)=0102H，(AX)=1234H。执行入栈指令：

PUSH AX

入栈指令执行操作如图 3-8 所示。最后新的 SP 指针是 100H，即(SP)=100H。



(a) 执行指令前的存储器情况　　(b) 执行指令后的存储器情况

图 3-8 例 3-8 指令的执行过程

3.4 指令系统

3.4.1 数据传送类指令

5. 地址传送指令

① 取偏移地址指令 LEA 的格式如下：

```
LEA   DST，SRC                ；指令执行操作：DST←SRC 的偏移地址
```

【例 3-9】执行指令： LEA AX，[BX＋DI＋0BCH]

设: (BX)＝123H，(DI)＝100AH，则指令执行后(AX)＝123H+100AH+0BCH=11E9H

② 全地址指针传送指令 LDS 的格式如下：

```
LDS   DST，SRC                ；指令执行操作：DST← SRC，DS← SRC＋2
LES   DST，SRC                ；指令执行操作：DST← SRC，ES← SRC＋2
```

3.4 指令系统

3.4.1 数据传送类指令

5. 地址传送指令

【例 3-10】 设某双字存储单元的段地址是 1817H（即当前 DS 的内容是 1817H），偏移地址为 3200H，此存储单元的双字数据为 12345678H，即此 32 位数据的首地址和对应的内部的数据分别是：(1817:3200)=78H、(1817:3200)=56H、...，则:

```
LDS   SI，[3200H]        ；执行指令后 (DS)=1234H，(SI)=5678H
LES   DI，[3200H]        ；执行指令后 (ES)=1234H，(DI)=5678H
```

3.4 指令系统

3.4.1 数据传送类指令

6. 标志传送指令

```
LAHF        ; AH←F_L ，即将标志寄存器低 8 位送 AH。
SAHF        ; F_L←(AH) ，即将 AH 的内容送标志寄存器低 8 位
PUSHF       ; 将标志寄存器内容压栈
POPF        ; 将栈顶内容送标志寄存器
```

3.4 指令系统

3.4.1 数据传送类指令

7. 输入/输出指令

① 输入指令IN

```
IN   AL，n      ；AL←[n]  ，将端口地址为 n 的 I/O 口上的 8 位数据输入到 AL
IN   AX，n      ；AH←[n+1]，AL← [n] ，将端口地址为 n 的 I/O 口上的 16 位数据输入 AX
IN   AL，DX     ；AL← [DX] ，将 DX 内容作端口地址的 I/O 口上的 8 位数据输入到 AL
IN   AX，DX     ；AH←[DX+1]，AL←[DX]，将 DX 内容指定地址端口的 16 位数据输入到 AX
```

② 输出指令OUT

```
OUT   n，AL      ；AL→[n]，将 AL 的 8 位数据输出到端口地址为 n 的 i、I/O 端口上
OUT   n，AX      ；AH→[n+1]，AL→[n]，将 AX 的 16 位数据输出到端口地址为 n 的端口上
OUT   DX，AL     ；AL→[DX]，将 AL 的 8 位数据输出到以 DX 指定地址的 I/O 端口上
OUT   DX，AX     ；AH→[DX+1]，  AL→[DX]
```

3.4 指令系统

3.4.2 算术运算指令

1. 加法指令

① 普通加法指令**ADD**

```
ADD   DST，SRC      ；执行操作：DST←DST+SRC
```

【例 3-11】

```
ADD   AL，34H       ；寄存器 AL 的内容与 8 位立即数 34H 相加
ADD   AX，2040H     ；寄存器 AX 的内容与立即数 2040H 相加
ADD   [BX+SI]，AX   ；通过间址，将存储器中的 16 位数据与寄存器相加
```

3.4 指令系统

3.4.2 算术运算指令

1. 加法指令

② 带进位的加法指令**ADC**

```
ADC   DST，SRC          ；执行操作：DST ← DST＋SRC＋CF ， CF 是进位标志位
```

```
【例 3-12】 ADC   AX，DX            ；寄存器与寄存器带进位相加，结果进入 AX 中
            ADC   CX，1000H         ；寄存器与立即数带进位相加，结果进入 CX 中
            ADC   AX，COUNT[SI]     ；AX ←(AX)+[COUNT+(SI)]寄存器与存储器带进位相加
```

3.4 指令系统

3.4.2 算术运算指令

1. 加法指令

③ 加1指令INC的指令

```
INC   DST        ;   执行操作：DST←DST+1
```

【例 3-13】

```
INC   BX                  ；将 16 位寄存器 BX 中的内容加 1
INC   AL                  ；将 8 位寄存器 AL 中的内容加 1
INC   BYTE   PTR [BX]     ；以 BX 的内容指定的存储器单元中的字节加 1
```

3.4 指令系统

③ 加1指令INC的指令

【例 3-14】 试完成两个四字节数的相加。

设此两个四字节的数已分别放在自 FIRST 和 SECOND 开始的存储区中，每个数占四个存储单元。存放时，最低字节在地址最低处，相加结果存放在地址以 THIRD 为首的存储区中。主要程序段如下：

```
LEA   SI，FIRST
MOV   AX，[SI]          ；把 FIRST 所指数据区的低二位字节送入 AX 中
LEA   DI，SECOND
ADD   AX，[DI]          ；将两个低位字节相加后送 AX
LEA   BX，THIRD
MOV   [BX]，AX          ；将相加结果送到地址以 THIRD 位首的存储区中
MOV   AX，[SI＋2]       ；从 FIRST 所指数据区取高二位字节送入 AX 中
ADC   AX，[DI＋2]       ；将两个高位字节带进位相加后送 AX
MOV   [BX＋2]，AX       ；存放结果
```

3.4 指令系统

3.4.2 算术运算指令

2. 减法指令

① 普通减法指令**SUB**

```
SUB   DST，SRC          ；执行操作；DST← DST－SRC
```

```
【例 3-15】  SUB   AX，CX                ；  AX← (AX)－(CX) ，16 位减法操作
             SUB   AL，10H               ；  AL← (AL)－10H  ，8 位减法操作
             SUB   WORD  PTR [DI]，BX    ；     [DI] ← [DI]－(BX) ，16 位减法操作
```

3.4 指令系统

3.4.2 算术运算指令

2. 减法指令

② 带借位的减法指令**SBB**

```
SBB   DST，SRC                    ；   执行操作：DST←DST－SRC－CF
```

```
【例 3-16】  SBB   AX，2030H      ；AX←(AX)－2030H－CF ，16 位带借位减法操作
             SBB   BL，CL         ；  BL←(BL)－(CL)－CF   ，8 位带借位减法操作
             SBB   [SI]，AL       ；  [SI]←[SI]－(AL)－CF ，8 位带借位减法间接寻址操作
             SBB   AX，DATA[SI]   ；AX←(AX)－[(SI)+DATA]－CF ，16 位间接寻址操作
```

3.4 指令系统

3.4.2 算术运算指令

2. 减法指令

③ 与**INC**的功能相反，**DEC**是减**1**指令

```
        DEC   DST                    ；执行操作；DST←(DST)－1
```

```
【例 3-17】DEC   CX                  ;16 位寄存器减 1
        DEC   AL                     ;8 位寄存器减 1
        DEC   WORD PTR [BP][DI]      ；即 DEC   WORD PTR[BP+DI]，字操作，存储器减 1
        DEC   BYTE PTR [BX]          ；存储器减 1，字节操作
```

3.4 指令系统

3.4.2 算术运算指令

2. 减法指令

④ 求补指令**NEG**

```
NEG   DST                    ;  执行操作：DST ← 0－(DST)
```

⑤ 比较指令**CMP**

```
CMP   DST，SRC               ；指令操作：DST－SRC  → 影响标志位
```

【例 3-18】

```
CMP   AL，CL                 ；8 位寄存器间的内容比较
CMP   AX，CX                 ；16 位寄存器间的内容比较
CMP   WORD PTR [BX]，AX      ；16 位寄存器与存储器内容比较
CMP   [BX＋102H]，SI         ；间接寻址存储器内容与寄存器内容比较
```

3.4 指令系统

3.4.2 算术运算指令

2. 减法指令

⑤ 比较指令**CMP**

【例 3-19】 若(BL)= 87H，考察执行指令 CMP BL，9BH 后的结果。
用 DEBUG 可以证明，执行此指令后标志位的情况是：
CF=CY=1、OF=NV=0、SF=NG=1、ZF=NZ=1、PF=PO=0。
若将 87H 和 9BH 看成是无符号数，
则(BL) < 9BH，有借位，故 CF=CY；结果非 0：ZF=NZ;
若将 87H 和 9BH 看成是有符号数，
则二数皆为负值，87H= –121，9BH= –101。
仍然有(BL)<9BH，OF=NV、SF=NG，OF≠SF。

3.4 指令系统

3.4.2 算术运算指令

3. 乘法指令

```
MUL    SRC                    ；无符号数乘法指令
IMUL   SRC                    ；有符号数乘法指令
```

【例 3-20】 试求执行指令 MUL BL 与 IMUL BL 后的乘积，及 CF 和 OF 的值。

在此设 (AL)＝34H，(BL)＝DFH，

AX＝(AL)×(BL)＝ 34H×0DFH＝2D4CH，由于 AH 不为 0，

故 CF＝1，OF＝1。

若为有符号数，执行指令 IMUL BL 后：

AX＝(AL)×(BL)＝ 34H×0DFH=52×(－33)= －1716=－(6B4H)=F94CH,

CF＝1，OF＝1。

利用 DEGUG 也能得到相同结果，这时显示 CF=CY，OF=OV。

3.4 指令系统

3.4.2 算术运算指令

4. 除法指令

DIV SRC

IDIV SRC

【例 3-21】 设 (AX)＝520AH，(BL)＝9CH，求执行除法指令 DIV BL 后的结果；

若设(AX)＝FC0AH，(BL)＝56H，求执行除法指令 IDIV BL 后的结果。

最后用 DEBUG 验证。

此例中，对于指令 DIV BL，都是无符号数相除，

则 520AH÷9CH＝86H…62H，即(AL)＝86H，(AH)=62H，于是(AX)=6286H。

对于有符号数的除法指令 IDIV BL，先用绝对值相除，再加符号位。

FC0AH 的绝对值是 0－FC0AH＝3F6H，3F6H÷56H＝BH…44H。

商和余数的符号都为负，所以

(AL)＝-BH=F5H，(AH)＝-44H=BCH，(AX)＝BCF5H。

3.4 指令系统

3.4.2 算术运算指令

4. 除法指令

CBW　　CWD

【例 3-22】 完成二有符号数相除的程序设计，并在 DEBUG 上调试。

设被除数、除数都为 16 位数，并分别存放在 BUFFER 开始的连续 4 个存储器单元中，从 BUFFER 地址单元开始存放的 4 个字节数据分别是：0ECH、6AH、32H、0F2H。要求将相除的商和余数存放在后继 4 个单元中。

即计算 6AECH÷F232H，并标出存储单元中的放置计算结果情况。参考程序如下：

```
MOV   BX，OFFSET BUFFER     ；取偏移地址
MOV   AX，[BX]              ；取被除数, (AX)=6AECH
   CWD                      ；除数进行扩展，被除数要扩展为 32 位，于是(DX)=0000H
IDIV  [BX+2]                ；两个数相除: 00006AECH÷F232H=(AX,DX)=FFF90A4AH
MOV   [BX+4]，AX            ；商存入内存，（AX）=FFF9H，等于-7
MOV   [BX+6]，DX            ；余数存入内存，（DX）=0A4AH,等于+2634
```

3.4 指令系统

3.4.2 算术运算指令

5. 十进制调整指令

(1) 加法的非压缩BCD码调整指令AAA

【例 3-23】 用非压缩 BCD 码实现 35H＋8H，结果送 AX。

设(AX)=0305H。

```
ADD   AL，08H          ；加法指令执行结果：(AH)＝03H，(AL)＝0DH
AAA                    ；AAA 指令执行结果：(AX)＝0403H
```

3.4 指令系统

3.4.2 算术运算指令

5. 十进制调整指令

（2）减法的非压缩BCD码调整指令AAS

【例 3-24】 用非压缩 BCD 码实现 75H＋6H，结果送 AX。

```
MOV   AX，0705H
SUB   AL，06H          ；指令执行结果：AH＝07H，AL＝FFH
AAS                    ；指令执行结果：AH＝06H，AL＝09H
```

3.4 指令系统

```
-A 100                                  ；用汇编命令 A 输入程序
13B3:0100 MOV AX,0705                   ；键入指令 MOV AX，0705H   [回车]
13B3:0103 SUB AL,6
13B3:0105 AAS
-T=100  3                                   ；连续 3 个单步运行程序
AX=0705  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=13B3  ES=13B3  SS=13B3  CS=13B3  IP=0103   NV UP EI PL NZ NA PO NC
13B3:0103 2C06          SUB     AL,06
                                        ；第一条指令运行后，AX=0705
AX=07FF  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=13B3  ES=13B3  SS=13B3  CS=13B3  IP=0105   NV UP EI NG NZ AC PE CY
13B3:0105 3F            AAS
                                       ;运行减法指令后 AH=07，AL=FF，AC=1
AX=0609  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=13B3  ES=13B3  SS=13B3  CS=13B3  IP=0106   NV UP EI PL NZ AC PE CY
                                       ；运行 AAS 减法调整指令后 AX=0609 ,CY=1
```

3.4 指令系统

3.4.2 算术运算指令

5. 十进制调整指令

（3）乘法的非压缩BCD码调整指令AAM

【例 3-25】 用非压缩 BCD 码完成 5H×8H，结果送 AX。设(AL)=05H，(BL)=08H。

```
MUL   BL                    ；指令执行结果：AX＝0028H
AAM                         ；指令执行结果：AX＝0400H
```

(4) 除法的非压缩BCD码调整指令AAD

【例 3-26】 设(AX)=0103H，(BX)=6，用非压缩 BCD 码完成 13÷6，结果送 AX。

```
AAD                         ；AX＝000DH
DIV   BL                    ；商 AL＝02H，余数 AH=01H，AX＝0102H
```

3.4 指令系统

3.4.2 算术运算指令

5. 十进制调整指令

(5) 加法的压缩BCD码调整指令DAA

【例 3-27】 设(AL)=74H，(BL)=99H，用压缩 BCD 码完成 74+99，结果送 AL。

```
ADD   AL，BL        ；加法结果是：（AL）＝0DH，CF=CY=1，AF=NA=0
DAA                 ；调整后，(AL)＝73H，  CF=CY=1，AF=AC=1
```

(6) 减法的压缩BCD码调整指令DAS

3.4 指令系统

3.4.3逻辑运算指令

1. 按位逻辑相与指令AND

```
AND   DST，SRC          ；执行操作：DST←DST ∧ SRC，∧是按位逻辑与操作符。
```

【例 3-28】若(BL)= 0F0H，[DI]=0F0FH，考察以下指令执行的结果：

```
AND   AL，BL          ；对 AL 内的低 4 位全部清 0，同时保持高 4 位不变
AND   DX，00FFH       ；对 DX 内的高 8 位全部清 0，同时保持低 8 位不变
AND   CX，[DI]        ；对 CX 内的部分 4 位值清 0
```

3.4 指令系统

3.4.3逻辑运算指令

2. 按位逻辑相或指令OR

OR DST，SRC ；执行操作：DST←DST∨SRC，∨是按位逻辑或操作符。

3. 按位逻辑取非指令NOT

NOT DST ；执行操作是按位逻辑取反，将结果送回 DST。

4. 按位逻辑异或指令XOR

XOR DST，SRC ；执行操作：DST←DST ⊕ SRC，⊕是按位逻辑异或操作符。

3.4 指令系统

3.4.3逻辑运算指令

5. 测试指令TEST

```
TEST  DST，SRC          ；执行操作：DST←DST∧SRC，DST 内容不变。
```

【例 3-29】 测试 AL 中的第 0、1、2 位是否为 0。

```
MOV   AL，11001000B
TEST  AL，00000111B     ；执行指令后 AL 的内容不变，但标志 ZF=1
```

3.4 指令系统

3.4.4 移位指令

1. 非循环移位指令

① 算术左移**SAL**和逻辑左移**SHL**指令

```
SAL   DST，cnt      ；算术左移指令，
SHL   DST，cnt      ；逻辑左移指令
```

图 3-9 SHL 和 SAL 操作示意图

② 算术右移指令**SAR**

```
SAR   DST，cnt
```

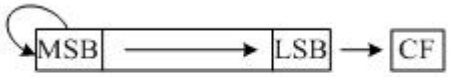


图 3-10 SAR 指令操作示意图

3.4 指令系统

3.4.4 移位指令

③ 逻辑右移指令SHR

SHR DST，cnt

图 3-11 SHR 指令操作示意图

【例 3-30】 设执行移位指令前，(BX)=5A99H，(CL)=4，(AL)=85H，(DI)=200H，存储器数据段以首地址为 200H 单元内存有字：1234H。分析执行以下指令后的结果。

```
SHL   BX，CL                ；执行此指令后 (BX)=A990H, CF=1
SAR   AL，  1               ；执行此指令后 (AL)=11000010B=C2H，CF=1
SHR   WORD  PTR [DI]，CL    ；执行此指令后 原来的 1234H 变为 0123H，CF=0
```

对于一个数，左移或右移 1 位，只要移位后的数未超出一个字节或一个字所能表达的范围，则相当于原来的数乘以或除以 2。

【例 3-31】试使用移位指令完成计算：94ABH÷2H 和 7A4H×8H

```
SHR   AX，1          ；94ABH=38059，执行指令后：(AX)=4A55H=19029,
SHL   AX，  CL       ；执行指令前：(AX)=7A4H，(CL)=3，执行指令后：(AX)=3D20H
```

3.4 指令系统

3.4.4 移位指令

2. 循环移位指令

① 循环左移指令**ROL**　　ROL　DST，cnt

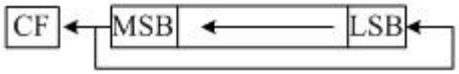


图 3-12 ROL 指令操作示意图

② 循环右移指令**ROR**　　ROR　DST，cnt

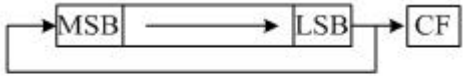


图 3-13 ROR 指令操作示意图

3.4 指令系统

3.4.4 移位指令

2. 循环移位指令

③ 带进位循环左移指令**RCL**　　RCL　DST，cnt

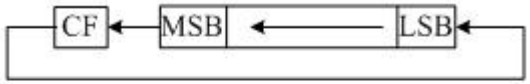


图 3-14 RCL 指令操作示意图

④ 带进位循环右移指令**RCR**　　RCR　DST，cnt

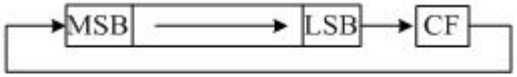


图 3-15 RCR 指令操作示意图

3.4 指令系统

3.4.4 移位指令

2. 循环移位指令

【例 3-32】 设执行指令前，(BL)=A9H，(CL)=2，(AL)=6BH，(DI)=200H，存储器数据段地址为 200H 单元的内容是 98H，CF=1。用 DEBUG 验证并分析执行以下指令后的结果。

```
ROL   BL，CL                ；执行此指令后 (BL) =10100110B=A6H, CF=0
ROR   AL，  1               ；执行此指令后 (AL) =10110101B=B5H，CF=1
RCL   AL，  1               ；执行此指令后 (AL) =11010110B=D7H，CF=0
RCR   BYTE  PTR [DI]，CL    ；执行此指令后 原来的 98H 变为 66H，CF=0
```

3.4 指令系统

3.4.5 串操作指令

1. 串操作指令

（1）串传送指令MOVS的指令格式

```
MOVSB  /  MOVSW          ；字节传送指令 / 字传送指令
```

【例 3-33】试用串传送指令编写的程序把 DS 数据段的偏移地址为 2000H 开始的 200 个字节的内容顺序传送到 ES 附加段的偏移地址为 3000H 开始的单元中。

设已确定好了 DS 和 ES 的段地址内容，程序段如下：

```
        MOV   SI，2000H      ；SI←源串首址指针
        MOV   DI，3000H      ；DI←目的串首址指针
        MOV   CX，200        ；CX←字节串长度
        CLD                  ；清 0 DF，使地址指针递增
RND:    MOVSB                ；传送一个字节数据
        DEC   CX             ；字节串长度减 1
        JNZ   RND            ；测试 CX，若 CX≠0 转 RND 处继续进行传送，否则停止
```

3.4 指令系统

3.4.5 串操作指令

1. 串操作指令

(2) 串比较指令CMPS的指令

```
CMPSB  /  CMPSW        ；字节串比较指令 / 字串比较指令
```

(3) 串搜索指令SCAS的指令

```
SCASB  /  SCASW        ；字节串搜索指令 / 字串搜索指令
```

(4) 读取串指令LODS的指令

```
LODSB  /  LODSW                ；字节读取串指令 / 字读取串指令
```

3.4 指令系统

3.4.5 串操作指令

1. 串操作指令

(5) 存串指令STOS的指令

```
STOSB  /  STOSW      ;  字节存串指令 /  字存存串指令
```

【例 3-34】把扩展段 0200H 开始的 256 个内存单元清零。程序段如下：

```
      CLD                    ; 清 0 方向标志位 DF，使 DI 作增量操作
      MOV  DI，200H          ; DI 存储器的首地址。假设已对 ES 设好扩展段的段地址
      MOV  CX，128           ; CX←字串长度
      MOV  AX，0             ; 清 AX 累加器
RND :  STOSW                       ; 传送一个字 0
      DEC  CX
      JNZ  RND                ; 若 CX≠0 转 LP 处
```

3.4 指令系统

3.4.5 串操作指令

2. 重复前缀指令

(1) 使用重复前缀REP的指令

```
REP   MOVSB     /     REP   MOVSW      ；重复字节传送 / 重复字传送指令
REP   LODSB     /     REP   LODSW      ；重复字节读取串 / 重复字读取串指令
REP   STOSB     /     REP   STOSW      ；重复字节存串 / 重复字存串指令
```

(2) 相等重复前缀REPE/REPZ 的指令

```
REPE   CMPSB     /     REPE   CMPSW     ；重复字节比较指令 / 重复字比较指令
REPE   SCASB     /     REPE   SCASW     ；重复字节搜索指令 / 重复字搜索指令
```

3.4 指令系统

3.4.5 串操作指令

2. 重复前缀指令

(3) 不相等重复前缀REPNE/REPNZ 的指令

```
REPNE   CMPSB     /     REPNE   CMPSW      ；重复字节比较指令 / 重复字比较指令
REPNE   SCASB     /     REPNE   SCASW      ；重复字节搜索指令 / 重复字搜索指令
```

3. 重复前缀指令用法注意点

3.4 指令系统

3.4.5 串操作指令

3. 重复前缀指令用法注意点

【例 3-35】 在数据段中有一字符串，长度为 20，要求把它们传送到附加段中的一个缓冲区中，其中源串存放在数据段中从符号地址 AS 开始的存储区域内，每个字符占一个字节；AD 为附加段中用以存放字符串区域的首地址。完成此任务的程序段如下：

```
LEA   SI，AS          ；置源串偏移地址
LEA   DI，AD          ；置目的串偏移地址
MOV   CX，20          ；置串长度计数值
CLD                   ；方向标志复位，地址作增量计数
REP   MOVSB           ；字符串传送
```

3.4 指令系统

3.4.5 串操作指令

3．重复前缀指令用法注意点

【例 3-36】 在附加段中有一个字符串，存放在以符号地址 AD 开始的区域中，长度为20，要求在该字符串中搜索空格符(ASCII 码为 20H)。完成此任务的程序段如下：

```
LEA    DI，AD          ；装入目的串偏移地址
MOV    AL，20H         ；装入搜索关键字节
MOV    CX，20          ；装入字符串长度
CLD                    ；方向标志复位，地址作增量计数。(DF=0=UP；DF=1=DN)
REPNE  SCASB           ；字符搜索
```

上述程序段执行之后，DI 的内容即为相匹配字符(ASCII 码的 20H)的下一个字符的地址，CX 中是剩下的还未比较的字符个数。若字符串中没有所要搜索的关键字节，则查完后，(CX)=0，即退出重复操作状态。

3.4 指令系统

3.4.5 串操作指令

【例 3-37】 比较两个字符串数据，如果两个字符串都相等，则把 RESULT 单元置为 00H；如果有一个不相等，则把 RESULT 置为 FFH。第一个字符串放在 Sl 开始的单元中，第二个字符串放在 S2 开始的单元中，字符串长度为 100 个字节，程序段如下：

```
        PUSH   DS
        POP    ES                ；两字符串都在数据段，故设(ES)=(DS)
        LEA    SI，S1            ；SI 指向第一个字符串
        LEA    DI，S2            ；DI 指向第二个字符串
        MOV    CX，0064H         ；CX 字符串长度作计数值
        CLD                      ；地址指针递增
        REPZ   CMPSB             ；串比较，循环
        JNZ    A1                ；若有一个不等，则转 A1，置 0FFH 标志
        MOV    AL，0             ；两个串都相等，置 00H 标志
        JMP    A2
A1：    MOV    AL，0FFH
A2：    MOV    RESULT，AL
```

3.4 指令系统

3.4.6 控制转移指令

1. 无条件转移指令

```
JMP   OPRD                          ; OPRD 是转移的目的地址
```

(1) 无条件转移指令JMP的直接转移类型

```
JMP   SHORT OPRD                    ; IP 改变范围是 IP+8 位偏移量

    JMP   NEAR   PTR   OPRD         ; IP 改变范围是 IP－16 位偏移量
或  JMP    OPRD                     ; NEAR 可省略

JMP   FAR   PTR   OPRD      ; IP＝所转移的段内偏移量 OPRD，CS＝OPRD 所在的段地址.
```

3.4 指令系统

3.4.6 控制转移指令

1. 无条件转移指令

(2) 无条件转移指令JMP的段内间接转移

```
JMP   WORD PTR OPRD                    ; IP=(EA),
```

```
JMP   WORD   PTR [BX]      ; IP=[(DS)×10H+(BX)] ，指令默认寄存器 BX 存放偏移地址
JMP   WORD   PTR BX        ; IP=(BX) ，注意，指令默认 BX 寄存器
JMP   DWORD PTR OPRD       ; IP=(EA)，CS=(EA+2)
```

3.4 指令系统

3.4.6 控制转移指令

2. 子程序调用和返回指令

（1）段内调用的CALL指令

```
CALL   LABEL        ; 直接给出被调用子程序的程序标号 LABEL，即偏移地址
CALL   REG          ;  即由寄存器 REG 间接给出 16 位转跳地址
CALL   MEM          ; 通过间接寻址，由存储器间接给出 16 位转跳地址
```

【例 3-38】考察以下指令的功能：

```
CALL   [700H]       ; 存储单元偏移地址 700H 和 701H 的内容分别送 IP 的低 8 位和高 8 位
CALL   AX           ; AX 的内容直接送 IP
CALL   NEAR  PTR  INTV ; 程序标号 INTV 的偏移地址送 IP
```

3.4 指令系统

3.4.6 控制转移指令

2. 子程序调用和返回指令

（2）段间调用的CALL指令

```
CALL   LABEL      ; 直接给出 32 位被调用子程序的标号 LABEL，包括此子程序的所在段地址
CALL   MEM        ; 通过间接寻址，由存储器间接给出 32 位转跳地址
```

```
CALL   DWORD   PTR [DI]          ; SP←(SP)－4, 间接寻址：IP←[DI] , CS←[(DI)+2]
CALL   1815H : 700H              ; SP←(SP)－4, IP←700H , CS←1815H
CALL   FAR   PTR   INTV          ; SP←(SP)－4, IP←INTV 的偏移地址 , CS←INTV 的段地址
```

(3) 子程序返回指令RET

3.4 指令系统

3. 条件转移指令

表 3-2　条件转移指令表

汇编指令格式	功能与操作方式
标志位转移指令	
JZ/JE OPRD，或 JNZ/JNE　OPRD	结果为 0(ZF=1)，或结果不为 0(ZF=0)转移
JS　OPRD，或 JNS OPRD	结果为负数(SF=1)，或结果为正数(SF=0)转移
JP/JPE OPRD，或 JNP/JPO　OPRD	结果奇偶校验为偶(PF=1)，或结果奇偶校验为奇(PF=0)转移
JO　OPRD，或 JNO　OPRD	结果溢出(OF=1)，或结果不溢出(OF=0)转移
JC　OPRD，或 JNC　OPRD	结果有进位/借位(CF=1)，或结果无进位/借位(CF=0)转移
不带符号数比较转移指令	
JA　OPRD，或 JNBE　OPRD	高于，或不低于等于(CF)转移
JAE　OPRD，或 JNB　OPRD	高于等于，或不低于(CF=0)转移
JB　OPRD，或 JNA　OPRD	小于或不大于等于(CF=1)转移
JBE　OPRD，或 JNA　OPRD	小于等于或不大于(CF∨ZF=1)转移
带符号数比较转移指令	
JG　OPRD，或 JNLE　OPRD	高于或不低于等于((SF⊕OF)∨ZF=0)转移
JGE　OPRD，或 JNL　OPRD	高于等于或不低于(SF⊕OF=0)转移
JL　OPRD，或 JNGE　OPRD	小于或不大于等于(SF⊕OF=1)转移
JLE　OPRD，或 JNG　OPRD	小于等于或不大于((SF⊕OF)∨ZF=1)转移
测试转移指令	
JCXZ　OPRD	CX＝0 时转移

3.4 指令系统

3.4.6 控制转移指令

4. 循环控制指令

表 3-3 循环指令表

指令格式	执行操作
LOOP OPRD	CX=CX－1；若 CX≠0，则循环，否则跳出循环
LOOPNZ OPRD，LOOPNE OPRD	CX=CX－1;若 CX≠0 且 ZF=0，则循环，否则跳出循环
LOOPZ OPRD，LOOPE OPRD	CX=CX－1;若 CX≠0 且 ZF=1，则循环，否则跳出循环

3.4 指令系统

【例 3-40】 有一首地址为 ARY 的 20 个字的数组，试编写程序段，求出该数组的内容之和(不考虑溢出)，并把结果存入 TOTAL 中。程序段如下：

```
        MOV   CX，20          ；设计数器初值
        MOV   AX，0           ；累加器 AX 置初值 0
        MOV   SI，AX          ；间接寻址地址指针 SI 设初值为 0
START：ADD   AX，ARY[SI]     ；此指令等同于 ADD  AX，[ARY+(SI)]
        ADD   SI，2           ；修改间址寄存器地址指针值(是字操作，因此加 2)
        LOOP  START           ；若 CX≠0，返回 START，重复
        MOV   TOTAL，AX       ；存结果
```

3.4 指令系统

【例 3-41】 有一字符串，存放在 STR 的内存区域中，字符串的字节长度为 12。要求在字符串中查找空格(ASCII 码为 20H)，若找到，则继续查找空格符号，否则转到 NOFOUND 去执行。程序段如下：

```
        MOV   CX，12          ；设计数器初值
        LEA    SI，STR
        MOV   AL，20H         ；空格的 ASCII 码送 AL 作比较用
        INC    SI             ；地址指针加 1
NEXT:   CMP   AL，[SI]        ；比较是否空格，如果是，则标志位 ZF=ZR=1
        LOOPNZ   NEXT         ；若 ZF=1，表示找到空格符，则继续找空格符号
                              ；否则，若 CX=0，完成循环后执行以下指令
        JNZ   NOTFOUND        ；这时若 ZF=0，转 NOTFOUND 执行程序，否则执行下面的程序。
        …
NOTFOUND ：…
```

3.4 指令系统

【例 3-42】 设有一个数据区的起始地址是 BLK1，共存放有 100 个字节的带符号数，试编程把该区域内的正数和负数分离开，分别送入同一数据段内的两个缓冲区内。正数存放在 PDATA 开始的单元中，负数存放在 NDATA 开始的单元中。

```
        MOV   SI，OFFSET  BLK1      ；使地址指针 SI 指向数据区首地址 BLK1
        MOV   DI，OFFSET  PDATA     ；使地址指针 DI 指向正数数据区首地址 PDATA
        MOV   BX，OFFSRT  NDATA     ；使地址指针 BX 指向负数数据区首地址 NDATA
        MOV   CX，100               ;计数初值
        CLD                         ；清方向标志位 DF，使增量计数
LP1 ：  LODSB                       ；以 SI 为地址指针，将数据送 AL
        TEST  AL，80H               ；判断正、负数，ZF=0，为负数，否则为正数
        JNZ   MINUS                 ；负数，转移至 MINUS
        STOSB                       ；否则为正，存此数据至以 DI 为地址指针的存储区
        JMP   CNTU                  ；无条件转移到 CNTU
MINUS：XCHG   BX，DI                ；将负数地址指针送 DI，因为指令 STOSB 的指针默认 DI
        STOSB                       ；以 DI 指针存负数
        XCHG  BX，DI                ；再换回正数指针
CNTU：  DEC   CX                    ；计数器内容减 1
        JNZ   LP1                   ；计数器若不为 0，转跳 LP1 继续数据处理
        HLT
```

3.4 指令系统

【例 3-43】 从 BLK 开始有一 100 个字节的数据串，在该数据串中查找关键字对应的数值 20H。在该数据串中，如搜索到 20H，则将其所在地址保存在 P1 单元中。如果无关键字则使 P1 单元清零。程序段如下：

```
        PUSH   DS
        POP    ES                    ；辅助段和数据段在同一段中
        MOV    DI，OFFSET   BLK      ；将字串首地址送地址指针 DI
        MOV    CX，64H               ；置计数值 100
        MOV    AL，20H               ；送关键数值 20H
        CLD                          ；置增量地址计数
        REPNZ   SCASB                ；在字串中重复搜寻，若搜到系统数字则结束
        JZ      FOUND                ；ZF=1,表示发现关键字，转跳至 FOUND
        JMP     DONE1                ；ZF=0，未发现关键字，转跳 DONE1
FOUND:  DEC     DI                   ；将此关键字的地址指针减 1，回到原指针。
        MOV    [P1]，DI              ；将关键字地址送入 P1 地址的单元中
        JMP     DONE
DONE1:  MOV    BX，OFFSET   P1       ；将单元地址 P1 送 BX
        MOV    [BX]，0000H           ；P1 单元清 0
DONE:   HLT
```

3.4 指令系统

3.4.6 控制转移指令

5. 中断指令

(1) 中断指令INT的格式

INT n

【例 3-44】 分析了解执行指令 INT 6 后的情况。假设指令 INT 6 的当前地址是：(1835H:0102H)，它的下一条指令的地址是(1835H:0104H)。这就是说在执行此中断指令前，(CS)=1835H，(IP)=0102H；并设此时(SS)=1837H，(SP)=3000H，标志寄存器(FR)=3087H。

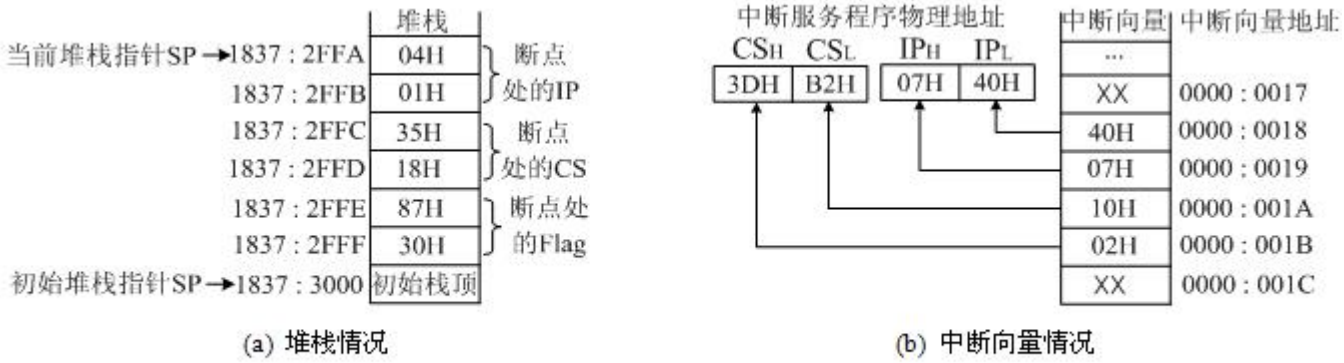


图 3-16 执行 INT 6 指令时堆栈与中断向量情况

3.4 指令系统

3.4.6 控制转移指令

5. 中断指令

(2) 溢出中断指令**INTO**的格式

(3) 中断返回指令**IRET**的格式

3.4 指令系统

3.4.7 处理器控制指令

表 3.4 处理器控制指令

汇编指令助记符		指令功能
标志操作指令	STC	进位标志 CF 置 1
	CLC	进位标志 CF 置 0
	CMC	进位标志 CF 取反
	STD	方向标志 DF 置 1
	CLD	方向标志 DF 置 0
	STI	中断允许标志 IF 置 1
	CLI	中断允许标志 IF 置 0
空操作	NOP	空操作
外同步指令	WAIT	等待 TEST 信号有效
	ESC	交权给外部处理机
	LOOK	在下一条指令期间封锁总线
暂停指令	HLT	暂停直至中断或复位