MOV

传送指令

格式：MOV dest，src ；dest← src

这里的 dest 为目的的操作数，src 为源操作数，以下类同。

功能：MOV 指令用于将一个操作数从存储器传送到寄存器，或从寄存器传送到存储器，或从寄存器传送到寄存器，也可以将一个立即数存入寄存器或存储单元，但不能用于存储器与存储器之间，以及段寄存器之间的数据传送。

MOV EAX，EBX；EAX←（EBX）（32 位双字）

MOV AX，BX ；AX←（BX）（16 位字）

MOV AL，BL ；AL←（BL）（8 位字节）

MOV EAX，25 ；EAX←25（32 位立即数）

MOV EAX , [500]；EAX ←DS：500H 单元中双字

注意：使用 MOV 指令时，不能向段寄存器传送立即数。如要在段寄存器中设置新值，必须先将立即数送入通用寄存器或存储单元，再由寄存器或存储单元送入段寄存器。但是，不允许向 CS 寄存器或（E）IP 寄存器进行任何形式的数据传送。另外，传送指令的两个操作数的位数必须相同。

ADD

加法指令

格式：ADD dest，src；dest←（dest）+（src）

要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器、存储单元或立即数。

功能：ADD 指令将两个操作数相加。相加结果取代第一操作数。

注意：scr 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行算术运算。例如：

ADD AX，BX ；AX←（AX）+（BX）

ADD EBX，ECX ；EBX←（EBX）+（ECX）

ADD EXTMEM，AX ；EXTMEM←（EXTMEM）+（AX）

ADD EXTMEM，23 ；EXTMEM←（EXTMEM）+23

SUB

减法指令

格式：SUB dest，src；dest←（dest）-（src）

要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器、存储单元或立即数。

功能：SUP 指令从第一操作数中减去第二操作数并将结果取代第一操作数。

注意：src 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行算术运

算。例如：

SUB AX，BX　　；AX←（AX）-（BX）

SBU EAX，EBX　　；EAX←（EAX）-（EBX）

ADC

带进位加法指令

格式：ADC　dest，src；dest←（dest）+（src）+（CF）

要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器、存储单元或立即数。

功能：ADC 指令先计算两操作之和。若进位 CF 为 1，则将结果加 1 送第一操作数的存放单元。

注意：src 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行算术运算。例如：

ADC AL，4

ADC AX，298

ADC EBX，11223344H

ADC NUMBER，78043810H

SBB

带借位减法

格式：SBB dest，src　　；dest←（dest）-（src）-（CF）

要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器、存储单元或立即数。

功能：SBB 指令从第一操作数减去第二操作数，如果 CF 为 1 则再减去 1，送第一操作数的存放单元。

注意：src 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行算术运算。例如：

SBB AX，BX

SBB EAX，EBX

SBB ECX，0ABCD1234H

CMP

比较指令

格式：CMP dest，src；（dest）—（src）

要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器、存储单元或立即数。

功能：CMP 指令从第一操作数减去第二操作数，相减结果不送回，只用于置 FLAGS 中

标志位。
注意：src 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行比较。
例如：

```
CMP AL，5
JZ    ABC        ；若 AL=5 则转 ABC，否则顺序执行
      …
ABC：
      …
```

INC and DEC

加一和减一指令
加 1 指令 INC 和减 1 指令 DEC 都是单操作数指令，源操作数单元也用作目的操作数单元。
格式：INC　reg/mem　；reg/mem←（reg）/（mem）+1
　　DEC　reg/mem　；reg/mem←（reg）/（mem）—1
要求：reg 为 8 位、16 位、32 位通用寄存器，mem 为 8 位、16 位、32 位存储单元。
功能：INC 指令使操作数加 1。它影响 SF、OF、ZF、AF 和 PF 位，但不影响进位标志 CF。DEC 指令从操作数中减 1，它影响 SF、OF、ZF、AF 和 PF 位，但不影响 CF 标志位。
例如：

```
INC AL                    ；AL←（AL）+1
INC EBX                   ；EBX←（EBX）+1
INC BYTE PTP MEMLOC       ；MEMLOC←（BEMLOC）+1
DEC BX                    ；BX←（BX）-1
DEC EAX                   ；EAX←（EAX）-1
```

MUL and IMUL

乘法指令
格式 1：MUL　src
　　IMUL　src
MUL 为不还符号乘法指令，IMUL 为带符号指令。
要求：src 原操作数可为 8 位、16 位、32 位通用寄存器或存储单元或立即数。
功能：寄存器 AL、AX 或 EAX 存放其中一个操作数并保存乘积的低半部分，另一个操作数 src 来自通用寄存器或存储单元或立即数，乘积的高半部分在 8 位时放在 AH，在 16 位乘 16 位时放 EAX 的高 16 位，且同时放在 DX 中（为了和 8086 兼容），在 32 位乘 32 位时则放在 EBX 中。

格式 2：IMUL dest，src
要求：dest 目的操作数可为 8 位、16 位、32 位通用寄存器，src 原操作数可为 8 位、16

位、32 位通用寄存器或存储单元或立即数。
功能：任何一个字节寄存器、字寄存器或双字寄存器的第一操作数与第二个来自寄存器或存储器的同样宽度的操作相乘，乘积放在第一个操作数所在的寄存器中。

注意：src 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行算术运算。

格式 3：IMUL dest，src，im
要求：im 可为 8 位、16 位、32 位立即数，dest 目的操作数可为 8 位、16 位、32 位通用寄存器，src 第二操作数可为 8 位、16 位、32 位通用寄存器或存储单元。
功能：im 立即数乘一个放在寄存器或存储器中的第二操作数，结果放在第一操作数指定的寄存器中。

注意：在 IMUL 指令格式 2 及格式 3 中，由于被乘数、乘数和乘积的长度一样，所以，有时会溢出。遇溢出时，溢出部分抛弃，且溢出标志 OF 置 1。
例如：

```
MOV   AL，NUMBER1
MUL   NUMBER2                    ；乘积存放在 AX 中
MOV   AX，NUMBER1
IMUL NUMBER2；乘积的高 16 位在 DX 中，同时放在 EAX 的高 16 位中，低 16 位放在 AX 中
MOV   EAX，12345678H
IMUL NUMBER                      ；乘积高 32 位在 EDX 中，低 32 位在 EAX 中
IMUL EDX，ECX                    ；用 EDX 中的内容乘以 ECX 中的内容，结果在 EDX 中
IMUL EDX，MEM_DWORD              ；用 EDX 中的内容和内存中双字相乘，结果在 EDX 中
IMUL EDX，MEM_DWORD，20          ；将存储器中双字乘 20，结果放送 EDX
```

DIV and IDIV

除法指令
格式：DIV   src
　　IDIV   src
要求：src 源操作数可为 8 位、16 位、32 位通用寄存器或存储单元，DIV 为无符号除法指令，IDIV 为带符号除法指令。
功能：用 AX、DX+AX 或者 EDX+EAX 存放 16 位，32 位或者 64 位被除数，除数的长度为被除数的 1/2，可放在寄存器或存储器中。指令执行后，商放在原存放被除数的寄存器的低半部分，余数放在高半部分。
注意：除法有一种特殊情况，比如被除数为 1000，放在 AX 中，除以 2，结果商为 500，应放在 AL 中；余数为 0，应放在 AH 中。此时，500 超过了 AL 的最大范围 256，会产生 0 号中断。
例如：

```
MOV   AX，NUMBER
IDIV   DIVSR   ；商在 AL 中，余数在 AH 中
```

MOV　DX，NUMBER_MSW
MOV　AX，NUMBER_LSW
IDIV　DIVSR　；被除数的高 16 位在 DX 中，低 16 位在 AX 中，除数为 16 位。相除后，商在 AX 中，余数在 DX 中
MOV　EDX，NUMBER_MSB
MOV　EAX，NUMBER_LSB
IDIV　DIVSR ；被除数的高 32 位在 EDX 中，低 32 位在 EAX 中，除数为 32 位，其结果商在 EAX 中，除数在 EDX 中。

AND、OR、XOR、NOT and TEST

逻辑运算指令
包括 AND、OR、XOR、NOT 和 TEST 指令。
格式：AND　dest，src　　；dest←（dest）^（src）
　　OR　　dest，src　　；dest←（dest）ˇ（src）
　　XOR　　dest，src　　；dest←（dest）⊕（src）
　　NOT　　reg/mem　　；reg/mem ←（FFFF）FFFF—（reg）/（mem）
　　TEST　dest，src　　；（dest）^（src）
要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器、存储单元或立即数。Reg 为 8 位、16 位、32 位通用寄存器，mem 为 8 位、16 位、32 位存储单元。
功能：AND、OR 和 XOR 指令按位逻辑“与”、“或”和“异或”。这些指令使用操作数的下列组合：两个寄存器操作数、一个通用寄存器操作数和一个存储器操作数、一个立即数与一个通用寄存器操作数或一个通用寄存器操作数。TEST 指令将两个操作数按应“与”，只置标志不存结果。NOT 指令是一元运算指令，用于将指定操作数的各位反应，其操作数取自寄存器或存储器。
注意：src 源操作数和 dest 目的操作数不能同时为存储单元。段寄存器不能进行逻辑运算。
例如：

```
NOT　EBX
AND　AX，0F0F0H
OR　　AX，CX
```

TEST 指令的使用举例如下：

```
TEST　AL，00010000B
JZ　　ABC　　　　；AL 中位 4 为 0 则转 ABC，否则顺序执行
…
ABC：
…
```

XCHG

交换指令

格式：XCHG dest，src ；dest←→src

要求：dest 为 8 位、16 位或 32 位通用寄存器或存储单元，src 为 8 位、16 位或 32 位通用寄存器或存储单元。

功能：XCHG 指令用于交换两个操作数。这条指令实际上起到了三条 MOV 指令的作用。指令中的两个操作数可以是两个寄存器操作数或一个寄存器与一个存储器操作数。当一个操作数是存储器操作数时，XCHG 指令自动地激活 LOOK 信号，这一特性在多处理器访问共享临界资源时非常有用。

LEA

取数据指令

格式：LEA r16/r32，mem

要求：r16 为一个 16 位通用寄存器，r32 为一个 32 位通用寄存器，mem 为存储单元。不是将存储单元的内容传送给通用寄存器。该指令能够用来进行乘加运算。因此，它是 Pentium/80486/80386 指令系统中最有效的指令之一。

例如：

```
LEA   BX，[SI]                    ；BX←（SI）
LEA   ECX，[EDX][43ESI]DOLLAR
```

设：EDX=34H，ESI=52H，DOLLAR=7289H，则该指令将有效地址 7405H（=34H14352H+7289H）传送给 ECX。

在很多情况下，LEA 指令可用以相应的源操作数为立即数的 MOV 指令代替。

例如：

```
LEA   BX，VARWORD
```

与

```
MOV   BX，OFFSET VARWORD
```

这两条指令的执行效果是完全一样的。区别在于后者是伪指令，由编译程序在编译时赋值；而前者在执行时赋值。

PUSH and POP

格式：PUSH src

POP dest

功能：PUSH 指令用于压入存储器操作数、寄存器操作数或立即数。例如：

```
PUSH AX
PUSH EBX
PUSH 12340000H
PUSH FS
PUSH GS
```

POP 指令与 PUSH 指令相反，将栈顶的数据弹出到通用寄存器或存储器中。例如：

```
POP AX
POP M_ADD
POP FS
POP GS
```

注意：当压堆栈的是 8 位数且操作宽度属性是 16 位或 32 位时，应对 8 位数进行带符号扩展（扩展到相应的位数）。

JMP

无条件转移指令

格式：JMP label

JMP reg/m16

功能：JMP 指令无条件地将控制转移到指令的指定的目标地址。根据目标地址相对于转移指令的位置，转移可分为短转移、段内转移和段间转移。

短（SHORT）转移属于相对转移，指在段内的短距离（-128~127）的转移。这种转移容易因使用不慎超出范围而出错。

段内（NEAR）转移指 CS 中段值和（E）IP 值都发生改变的转移。在这种情况下，处理器转到另一段中的某一地址去执行，因此，JMP 指令中要同时给出段选择符和偏移值。

转移指令的寻址方式分为直接寻址和寄存器、存储器间接寻址方式。

直接寻址在指令中直接给出转移地址。转移可以发生在段内，也可以发生在段间。

间接寻址在指令中给出的寄存器或存储单元中存放着转移的目标地址。因寄存器的长度与（E）IP 相同（存放偏移值），故采用寄存器间址的转移只可能是段内转移如下所示：

JMP NEAR PTR TABLE[EBX]；为段内转移

JMP FAR PTR TABLE[EBX] ；为段间转移

在 16 位寻址方式下，可以用 WORD 和 DWORD 来区分不带标号的转移是段内容转移还是段间转移。例如：

JMP WORD PTR[BX] ；为段内转移

JMP DWORD PTR[EBX] ；为段间转移

在 32 位寻址方式下，可由 DWORD 和 FWORD 来区别不带标号的转移是何种转移。例如：

JMP DWORD PTR[EBX]；为段内转移

JMP FWORD PTR[EBX]；为段间转移

下面给出在 16 位寻址方式下段内转移的程序段，其功能是转移到 ENTRY 去执行。

```
        DRVTBL   LABEL WORD
        DW       DRV$INIT
        DW       MEDI$CHK
        …
        DW       ENTRY
        …
ENTRY：
         …
         MOV      AX，SEG DRVTBL
         MOV      DS，AX
         MOV      SI，NUMBER 为命令表中 ENTRY 序号的 2 倍
         JMP      NEAR PTR DRVTBL[ST]
```

上述程序段中的后两条指令也可以改为：

```
         MOV      AX，NUMBER
```

```
MOV     SI，OFFSET DRVTBL
ADD     SI，AX
JMP     WORD PTR[SI]
```

对上述程序，还可以用寄存器间址的形式改为：

```
MOV     BX，OFFSET ENTRY
JMP     BX
```

Pentium/80486/80386 的指令与 8086/8088 相比，除了有 32 位寻址方式外，更重要的是在保护方式下，当执行“段”间 JMP 指令时，若“段”值（选择符）所对应的描述符是任务门或 TSS，将发生任务转移，并要进行严格的保护性检查。

JE/JZ, JA/JNBE, JAE/JNB, JB/JNAE, JBE/JNA

条件转移指令

JA/JNBE　label；高于/不低于等于，条件满足则转移到 label 指字的地址。

JAE/JNB　label；高于等于/不低于，条件满足则转移到 label 指字的地址。

JB/JNAE　label；低于/不高于等于，条件满足则转移到 label 指字的地址。

JBE/JNA　label；低于等于/不高于，条件满足则转移到 label 指字的地址。

JE/JZ　　label；等于/为 0，条件满足则转移到 label 指字的地址。