第四讲 如何写汇编语言机器指令语句

```
START:  MOV   AX，DATA
        MOV   DS，AX ;数据段首址赋值
        MOV   CX，50
        MOV   AX，0
        MOV   BX，1
NEXT:   ADD   AX，BUF[BX]
        INC   BX
        INC   BX
        DEC   CX
        JNZ   NEXT
        CLC
        MOV   SUM，AX
        MOV   AH,4CH
        INT   21H
```

典型汇编语言机器指令语句形式：

（1）无操作数：操作码

（2）单操作数：操作码　操作数

（3）双操作数：

操作码　操作数 1，操作数 2

目的　源

一般格式：　[标号：] [指令前缀] 助记符 [操作数] [；注释]

写汇编语言的指令语句需要掌握的两大问题：

1、操作数如何表达

2、操作码有哪些

一、寻址方式（第 2 章）

问题：如何找到某人？先确定其地址。

1. 路上遇到了该人。　立即得到（地址和人）。
2. 打电话问该人在哪里。　直接获得。
3. 问其他知情者。　间接获得。

每个知情者可能只知道一部分，要问几个知情者。　组合方式。

结论：客观上存在几种可能的渠道获得某人的住址，然后找到该人。

因此，计算机的指令也可以通过几种方式得到操作数的存放地址，然后取到操作数。

寻址方式：寻找操作数（源操作数或目的操作数）存放地址的方式。

问：　　ADD　AX，100　　的寻址方式？

（针对　源操作数或目的操作数）

计算机内的操作数的可以存放在三种不同的部件中：

第一种：CPU 内的寄存器

第二种：存储器（内存）

第三种：IO 设备的端口中

寻址方式四要素：格式、操作数所在部件、地址值和限制条件

1. 寄存器寻址，　　INC　　AX
2. 寄存器间接寻址，MOV　DI，　[BX]
3. 变址寻址，　　MOV　AR[SI]，DX
4. 基址加变址寻址，MOV　DX，AR[BX][SI]
5. 立即寻址，　　MOV　AX，10
6. 直接寻址。　　MOV　AX，DS:[10H]

这六种寻址方式决定了常用汇编指令的语法格式。重要吧！

- 寄存器寻址

（1）在指令中的使用格式：R

（R 是 CPU 内的寄存器名）

例：　INC　　BL

　　ADD　　EAX，EBX　　（类型）

（2）部件：操作数在指令指明的 CPU 内的寄存器 R 中

（3）地址：指令所指明的寄存器代表的地址就是操作数的存放地址（段？）

（4）限制条件：R 是任意 8 位、16 位、32 位寄存器，个别指令的限制除外

特点：存取速度快，经常存取的操作数采用之。

● 基址加变址寻址

（1）在指令中的使用格式：

[BR+IR*F+V]

或：V[BR][IR*F]

或：V[BR+IR*F]

例如：MOV　AX，8[BX][SI]

MOV　EAX，BUF[EDI*2+EBP]

（2）部件：操作数存放在存储器中。

（3）地址：偏移地址 EA 是指令中指定的基址寄存器 BR 的内容、变址寄存器 IR 的内容乘以比例因子、位移量 V 三项相加之和。

EA=（BR）+ （IR）*F + V

（EA 的计算过程为有符号数计算，结果为无符号数）

（4）限制条件：

（A）F=1，2，4，8。

（B）位移量 V 为一个有符号数或代数表达式，有效位数≤R 的位数。当 V 为变量及其表达式时，决定整个操作数的类型和段寄存器。

（C）没有变量时，默认段寄存器由基址寄存器 BR 决定。

当使用 16 位寄存器时：能使用哪些寄存器？如何确定 BR、IR？默认段寄存器？：

- BR = BX、BP 之一；IR = SI，DI 之一；F 只能为 1。
- 当 BR = BX 时，默认段寄存器为数据段寄存器 DS；
- 当 BR = BP 时，默认段寄存器为堆栈段寄存器 SS。

当使用 32 位寄存器时，能使用哪些寄存器？如何确定 BR、IR？默认段寄存器？

- BR=任一 32 位通用寄存器；
- IR=除 ESP 之外的任一 32 位通用寄存器

（BR 和 IR 可以相同，如[EAX+EAX*4+7]）。

- ①未带比例因子的寄存器是 BR；
  ②当比例因子为 1 并省略时，常见的是放在前面的寄存器为 BR。
- 当 BR 为 ESP、EBP 时，默认段寄存器用 SS；
  其它用 DS。

例 1： MOV AX，8[BX][SI]

源操作数地址： V=8，BR= BX（决定了段寄存器为 DS），IR=SI，F=1

执行前：（AX）=45H，（BX）=30H，（SI）=20H，

(WORD PTR) DS:([0058H]) =0099H。 （二维地址）

执行： EA=（基址寄存器）+（变址寄存器）+ V

= 30H + 20H + 8H = 0058H

DS:([0058H])= 0099H → AX

…	XXH	DS
0058H	99H	
0059H	00H	
005AH	12H	

执行后：（AX）=99H，（BX）、（SI）、DS:([0058H]）未变。

例 2： MOV EAX，-6[EDI*2][EBP]

源操作数地址：V= - 6，BR=EBP（决定了段寄存器为 SS），IR=EDI

特点：与高级语言中的二维数组对应，

int COUSE[i][j] ⟷ V[BR][IR*F]

寻址方式可以归纳为三大类寻址方式：

（1）寄存器方式 R （操作数在寄存器中）

（2）存贮器方式 （操作数在存储器中）

①寄存器间接方式 [R]

②变址方式 V[R*F] 常用于表指针（一维数组）

③基址加变址方式 V[IR*F][BR] 矩阵运算（二维数组）

（含寄存器便于存取一片连续单元）

④直接方式 [n] 或 变量名（表达式）

（3）立即方式 n （操作数跟在指令后面即在代码段中）

赋初值

存贮器方式一般形式:（基址加变址方式）

EA = （基址寄存器）+（变址寄存器）* 比例因子 + 位移量

省掉任何一个/两个就向其它形式转化

双操作数指令中，源和目的操作数不能同为存储器寻址方式

例：MOV BYTE PTR[SI], [DI] ; ERROR

ADD [EBX+EDI*4+10], COUNT ;ERROR COUNT 为字变量

只能是以下几种组合：

(1)寄存器对寄存器。

(2)寄存器与存储器。

(3)源操作数为立即寻址，目的操作数为寄存器或存储器之一。

存贮器寻址方式确定段寄存器的优先级从高到低的依据为：

段超越前缀（跨段前缀） “段寄存器名:”、

变量、

默认基址寄存器、

默认间址寄存器。

例：

```
MOV AX, [SI]            ; 使用 DS 段寄存器
MOV AX, [BP+SI]         ; 使用 SS 段寄存器
MOV AX, SUM[BP+SI]      ; 使用 DS 段寄存器（SUM 是 DS 段内变量）
MOV AX, CS: SUM[BP+SI]  ; 使用 CS 段寄存器
```

二、机器指令及其学习方法（P56：3.2 节）

（一）指令系统的分类

1. 数据传送指令
2. 算术运算指令
3. 位操作指令

4.串操作指令：MOVS、CMPS、SCAS、LODS、STOS（第五章）

5.程序控制指令（第四章）

转移指令：无条件转移指令 JMP

有条件转移指令 JNE、JG、JL、JGE、JLE ……

循环指令：LOOP

子程序调用及返回指令：CALL、RET

中断调用及返回指令： INT、IRET

6.处理机控制指令

标志的操作指令：CLC、CMC、STC；STD、CLD；STI、CLI。

其它指令：HLT、NOP、WAIT。

（二）学习指南 》》》 通过讲解示例指令，掌握：

（1）这些指令的用法

功能、符号、使用格式、特殊规定、影响哪些标志位、

指令字节长度、机器周期

（2）学习一条新指令的方法。 附录 2，P304

1. 数据传送指令

一般数据传送指令： MOV、MOVSX、MOVZX、XCHG、XLAT

堆栈操作指令： PUSH、PUSHA、PUSHAD、POP、POPA、POPAD

标志传送命令： PUSHF、PUSHFD、POPF、POPFD、SAHF、LAHF

地址传送指令： LEA、LDS、LES 、LFS、LGS、LSS

I/O 指令： IN、OUT（第六章）

（除 POPF、POPFD、SAHF 指令外，其他不影响标志位）

（1）一般传送指令

功能：实现数据传送，（OPS）→OPD （字节、字或双字）

符号：MOV

格式：MOV OPD, OPS

特殊规定：P58 传输途径示意图。

①不能实现存储单元之间的数据传送，

即 OPS、OPD 不能同时采用存储器寻址方式，如要传递，需以寄存器作桥梁才能完成。

例：需要将字变量 BUF0 中的内容传送至 BUF1 中，用以下方式：

```
MOV   AX , BUF0
MOV   BUF1 , AX
```

② 立即数不能传递至段寄存器中；不能向 CS 送数据；IP/EIP 不能在任何语句中出现。

如“MOV DS , 1000H”“MOV CS , AX”、“MOV AX，IP”均错，“MOV DS , AX”正确。

③ OPS 和 OPD 必须同时为字节、字、双字（即：类型相同），

如“MOV AX , CL”则为错误语句。

(2)其他需要注意的问题

传送偏移地址指令 LEA OPD，OPS，例如：LEA BX, DS:[10H]

与 MOV BUF，OFFSET MINDS 区别

又例：LEA EAX, [EDX+EDI*2+3]

后面自学。

2. 算术运算指令

加运算指令：ADD、ADC、INC

减运算指令：SUB、SBB、DEC、NEG、CMP

乘运算指令：IMUL、MUL

除运算指令：IDIV、DIV

符号扩展： CBW、CWD、CWDE、CDQ

(1)加指令 ADD

格式： ADD OPD，OPS

功能：(OPD)＋(OPS)→OPD

即将目的操作数与源操作数相加，结果存入目的地址中，而源地址中的内容并不改变。

ADC ：(OPD)＋(OPS)+ CF →OPD

（2）比较指令 CMP

格式：CMP OPD，OPS

功能：(OPD)-(OPS) ↛ OPD

（3）注意乘、除法指令的特殊规定

3. 位操作指令

逻辑运算指令：NOT、AND、TEST、OR、XOR、BT

移位指令：SHL/SAL、SHR、SAR；ROL、ROR、RCL、RCR

SHLD、SHRD

(1)逻辑运算指令

① 逻辑乘指令 AND

格式：AND OPD, OPS

功能：(OPD)∧(OPS)—>OPD

……

例：AND AX, OFH

执行前：(AX)=0FBBAH

执行后：(AX)=0AH

该指令主要用来在目的操作数中清除与源操作数置 0 的对应位，因此可用来将存储器或寄存器中不需要的部分清 0。将需要部分分离出来。

② 测试指令 TEST

格式：**TEST　OPD，OPS**

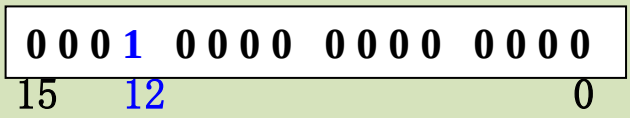
功能：**(OPD)∧(OPS)**

该指令主要用来检测目的操作数中某一位或某几位是否同时为 0，然后根据测试结果置 OF、CF、SF、ZF 位，后面往往跟着转移指令，根据测试结果确定转移方向。

需要测试哪一位或哪几位，则将源操作数中相应这几位置 1，其余置 0。

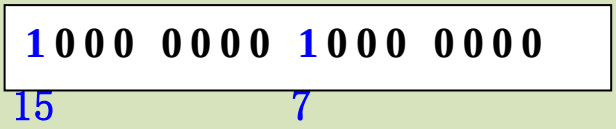
例如：要测试 AX 中第 12 位是否为 0，为 0 转 L 则要使用如下指令：

```
TEST AX, 1000H
JZ   L
```



如果要同时测试第 15 位和第 7 位是否同时为 0，为 0 转 L

```
TEST AX, 8080H
JZ L
```



这三条逻辑指令用法的选择：

a) 如果要将目的操作数中某些位清 0，用 AND

b) 如果要将目的操作数中某些位置 1，用 OR

c) 用来测试目的操作数中某一位或某几位是否为 0 或 1，而结果不能改变，用 TEST。

d) 操作数自身相或、相与结果不变。但影响 ZF、SF、PF，使 CF、OF 清零。可利用此判断操作数是否为 0。

(2)移位指令

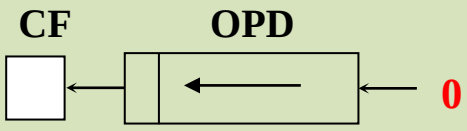
（相关操作数地址、方向、补入数、移出数、移给谁、移位次数）

①算术移位指令

a. 算术左移和逻辑左移指令 SAL/SHL

格式：SHL OPD,OPS 或 SAL OPD,OPS

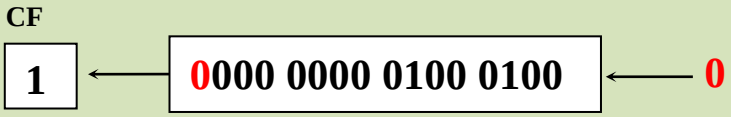
功能：将 OPD 的内容向左移动 OPS 指定的位数，其中 OPS 只能是常数 n (n<31)或 CL，CL 内容即为移动次数，移动方式为：



OPD 不能是立即数。

例如：SAL AX, 1

执行前：(AX)=0044H，CF=1



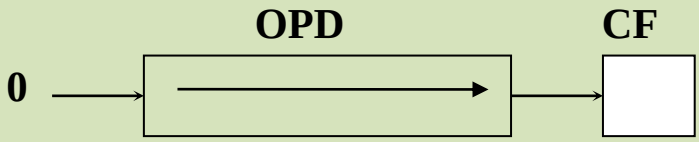
结果：CF=0，(AX)=0088H

乘 2 的效果。

b. 逻辑右移指令 SHR

格式：SHR OPD，OPS

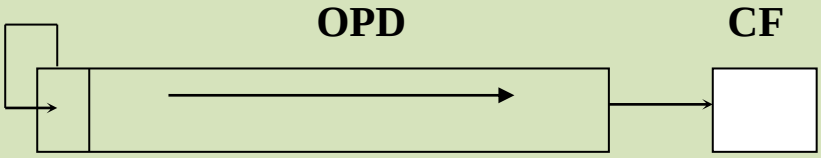
功能：（同样要修改 CF）



c. 算术右移指令 SAR

格式：SAR OPD, OPS

功能：将 OPD 中的操作数移动 OPS 所指定的次数。



在移动过程中符号位保持不变。

注意：算术移位等价于进行 2^n 的乘除运算。

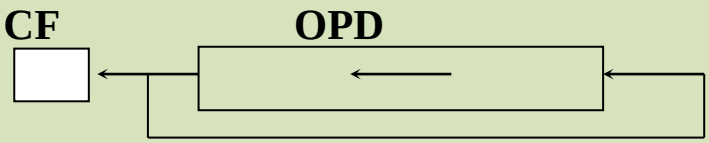
对有符号数来说，只有算术右移指令可保证正确，其他指令有可能溢出。

②循环移位指令

a. 循环左移指令 ROL

格式：ROL OPD, OPS

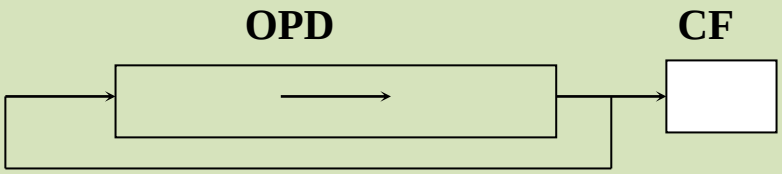
功能：



b. 循环右移指令 ROR

格式：ROR OPD，OPS

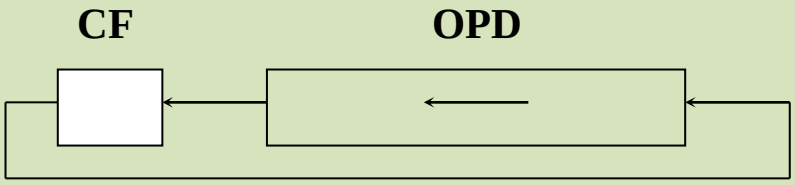
功能：



c. 带 C 位的循环左移指令

格式：RCL OPD，OPS

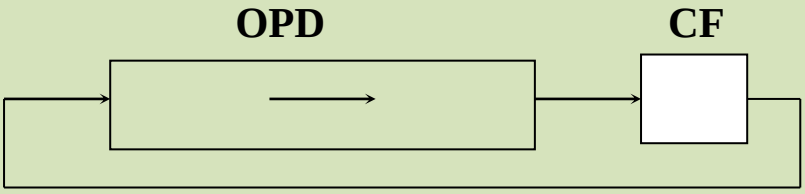
功能：



d. 带 C 位的循环右移指令 RCR

格式：RCR OPD，OPS

功能：



}