★ 21 世纪高等学校电子信息类系列教材

《微型计算机原理与应用(第二版)》

学习指导及习题集

许录平　楼顺天　王丽娟　编著

西安电子科技大学出版社
2001

内 容 简 介

本书是与西安电子科技大学出版社出版的《微型计算机原理与应用(第二版)》(王永山等编著)教材配套的学习指导及习题集。该书在章节安排上与教材相一致，其内容包括学习要点、典型例题和习题，并在附录中给出了常用并行接口芯片 8255A、部分习题的参考答案及一份课程模拟试题。

该书可作为“微机原理与系统设计”、“微机原理与程序设计”等相关课程的参考书，也可作为研究生入学考试复习指导书。

《微型计算机原理与应用(第二版)》学习指导及习题集

许录平　楼顺天　王丽娟　编著

策　　划　夏大平

责任编辑　戚文艳

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)8227828　　邮　编　710071

http：//www.xduph.com　　E-mail：xdupfxb@pub.xaonline.com

经　　销　新华书店

印　　刷　高陵县印刷厂

版　　次　2001 年 9 月第 1 版　2001 年 9 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印张 6.5

字　　数　149 千字

印　　数　1～6 000 册

定　　价　8.00 元

ISBN 7-5606-0398-X/TP・0516

前　言

《微型计算机原理与应用》一书自1991年12月由西安电子科技大学出版社出版至今9年多来，已发行了6万余册。该书已被全国近20所高校长期使用，并得到了广大授课教师、学生及从事计算机软、硬件开发的科研人员的一致好评。作为全国高等学校电子信息类规划教材，该书经修订于1999年12月出版第二版。为便于教学及配合新版教材，根据多年的实际教学经验，编写了这本学习指导及习题集。书中采用与教材章节一致的内容编排方式，并遵循原教材面向教学和面向应用相结合的指导思想，除了对各章的学习要点进行总结外，以典型例题的形式给学生学习上以详细指导，使学生对所学内容有比较全面的了解。给出与教材内容相配合的习题（其中部分习题取自教材的第一版）和部分习题的参考答案，供学生练习使用。考虑到科技工作者的实际需要，增加了有关8255A的内容介绍。附上一份课程模拟试题，帮助学生检验学习效果。

本书第1～5章和附录A由楼顺天编写，第7～9章由许录平编写，第6、10、11章由王丽娟编写。全书由许录平统稿。

本书在立项、编写和定稿过程中，承蒙教材主编王永山教授提出了许多宝贵意见，在此表示诚挚的感谢。由于我们水平有限，书中难免有错误和不妥之处，殷切希望广大读者批评指正。

编　者

2001年5月

第 1 章　微型计算机系统概述

要求熟悉微型计算机系统的硬件组成和基本工作方法，以及微型计算机的软件和操作系统。了解计算机的硬件组成结构、Intel 微处理器的主要成员、系统总线的概念。理解微型计算机的基本操作过程以及指令、程序等基本概念。理解操作系统的重要作用，掌握 DOS 基本命令的使用。

1.1　习　　题

1. 简述微型计算机系统的组成。
2. 简述计算机软件的分类及操作系统的作用。
3. CPU 是什么？写出 Intel 微处理器的家族成员。
4. 写出 10 条以上常用的 DOS 操作命令。

第 2 章　计算机中的数制和码制

要求掌握计算机中数制和码制的基础知识，主要包括各种进制数的表示法及相互转换、二进制数的运算、有符号二进制数的表示方法及运算时的溢出问题、实数的二进制表示法、BCD 编码和 ASCII 字符代码等内容。重点掌握各种进制数的表示及相互转换、有符号数的补码表示及补码运算。

2.1 学 习 要 点

1. 任意进制数的表示

任意一个数 N 可表示成 p 进制数：

$$(N)_p = \sum_{i=-m}^{n-1} k_i p^i$$

其中，$k_i=0, 1, \cdots, p-1$，数 N 表示成 m 位小数和 n 位整数。

2. 数制之间的转换

十进制数到任意进制数(设为 p 进制数)的转换规则：

(1) 整数部分：N 除以 p 取余数。

(2) 纯小数部分：N 乘以 p 取整数。

任意进制数(设为 p 进制数)到十进制数的转换规则：按权展开。

3. 二进制数的算术运算

二进制数的运算规则：逢二进一，借一当二。

4. 二进制数的逻辑运算

二进制数的逻辑运算有四种：AND(与)、OR(或)、NOT(非)、XOR(异或)运算，其特点是按位运算，即本位的运算结果不会对其它位产生任何影响。

5. 有符号数的补码表示

对于任意一个有符号数 N，在机器字长能表示的范围内，可分两步得到补码：

(1) 取 N 的绝对值。

(2) 如果 N 为负数，则对其绝对值中的每一位(包括符号位)取反，并在最低位加 1。这样就得到了有符号数 N 的补码。

6. 补码运算规则

补码运算规则：

(1) $[x+y]_{补}=[x]_{补}+[y]_{补}$ (mod 2^n)

(2) $[x-y]_{补}=[x]_{补}-[y]_{补}$ (mod 2^n)

(3) $[x-y]_{补}=[x]_{补}+[-y]_{补}$ (mod 2^n)

(4) $[x]_{补}-[y]_{补}=[x]_{补}+[-y]_{补}$ (mod 2^n)

一般称已知$[y]_{补}$求得$[-y]_{补}$的过程为变补或求负。其规则为

$$[-y]_{补}=\overline{[y]_{补}}+1$$

这可以直接采用 NEG 指令完成。

7. BCD 编码

用 4 位二进制数表示 1 位十进制数，这种表示方法称为 BCD(编)码。最常用的编码方法是采用 4 位二进制数的前 10 种组合来表示 0～9，这种编码方案称为 8421BCD 码。

当让计算机处理 BCD 码时，应对计算结果进行适当的修正。对加法运算应采用“加 6 修正”，对减法运算应采用“减 6 修正”，其规则总结如下：

(1) 两个 BCD 码位相加(相减)无进(借)位时，如果结果小于或等于 9，则该位不需要修正；如果结果大于 9，则该位进行加 6(减 6)修正。

(2) 两个 BCD 码位相加有进(借)位，则该位进行加 6(减 6)修正。

(3) 低位修正结果使得高位大于 9，则高位进行加 6(减 6)修正。

8. 常用字符的 ASCII 码

数字 0～9：30H～39H；字母 A～Z：41H～5AH；字母 a～z：61H～7AH；空格：20H；回车(CR)：0DH；换行(LF)：0AH；换码(ESC)：1BH。

2.2 习　　题

1. 将下列十进制数转换成二进制数：

(1) 49

(2) 73.8125

(3) 79.75

2. 将下列二进制数转换成十六进制数：

(1) 101101B　　(2) 1101001011B

(3) 1111111111111101B　　(4) 100000010101B

(5) 1111111B　　(6) 10000000001B

3. 将十六进制数转换成二进制数和十进制数：

(1) FAH　　(2) 5BH

(3) 78A1H　　(4) FFFFH

4. 将下列十进制数转换成十六进制数：

(1) 39

(2) 299.34375

(3) 54.5625

5. 将下列二进制数转换成十进制数：

(1) 10110.101B

(2) 10010010.001B

(3) 11010.1101B

6. 计算(按原进制运算)：

(1) 10001101B+11010B　　(2) 10111B+11100101B

(3) 1011110B−1110B　　(4) 124AH+78FH

(5) 5673H+123H　　(6) 1000H−F5CH

7. 已知a=1011B，b=11001B，c=100110B，按二进制数完成下列运算，并用十进制数运算检查计算结果：

(1) a+b　　(2) c−a−b

(3) a×b　　(4) c/b

8. 已知a=00111000B，b=11000111B，计算下列逻辑运算：

(1) a AND b　　(2) a OR b

(3) a XOR b　　(4) NOT a

9. 设机器字长为8位，写出下列各数的原码和补码：

(1) +1010101B　　(2) −1010101B

(3) +1111111B　　(4) −1111111B

(5) +1000000B　　(6) −1000000B

10. 写出下列十进制数的二进制补码(设机器字长为8位)：

(1) 15　　(2) −1

(3) 117　　(4) 0

(5) −15　　(6) 127

(7) −128　　(8) 80

11. 设机器字长为8位，先将下列各数表示成二进制补码，然后按补码进行运算，并用十进制数运算进行检验：

(1) 87−73　　(2) 87+(−73)

(3) 87−(−73)　　(4) (−87)+73

(5) (−87)−73　　(6) (−87)−(−73)

12. 已知a、b、c、d为二进制补码：a=00110010B，b=01001010B，c=11101001B，d=10111010B，计算：

(1) a+b　　(2) a+c

(3) c+b　　(4) c+d

(5) a−b　　(6) c−a

(7) d−c　　(8) a+d−c

13. 设下列四组为 8 位二进制补码表示的十六进制数，计算 a+b 和 a−b，并判断其结果是否溢出：

(1) a=37H，b=57H　　(2) a=0B7H，b=0D7H

(3) a=0F7H，b=0D7H　　(4) a=37H，b=0C7H

14. 求下列组合 BCD 数的二进制数和十六进制数：

(1) 3251　　(2) 12907

(3) ABCD　　(4) abcd

15. 将下列算式中的十进制数表示成相应的组合 BCD 码并进行运算，然后用加 6 或减 6 修正其结果：

(1) 38+42　　(2) 56+77

(3) 99+88　　(4) 34+69

(5) 38−42　　(6) 77−56

(7) 15−76　　(8) 89−23

16. 将下列字符串表示成相应的 ASCII 码(用十六进制数表示)：

(1) Example 1　　(2) XiDian University

(3) −108.652　　(4) How are you?

(5) Computer　　(6) Internet Web

17. 将下列字符串表示成相应的 ASCII 码(用十六进制数表示)：

(1) Hello　　(2) 123<CR>456 (注：<CR>表示回车)

(3) ASCII　　(4) The number is 2315

第 3 章　微机系统中的微处理器

要求了解微型计算机系统中的核心部件微处理器(CPU)。通过了解 CPU 的内部和外部结构，理解微处理器级总线(地址总线、数据总线和控制总线)的概念；通过学习 CPU 的功能结构，掌握 CPU 中两个独立单元(执行单元 EU 和总线接口单元 BIU)的并行执行过程；通过学习 8086 的寄存器结构，掌握汇编语言程序设计所需要的 14 个寄存器；通过介绍 8086 的存储器组织与分段、I/O 端口地址空间等基本知识，了解 8086 CPU 与外围电路的关系。重点掌握数据的 8 种基本寻址方式和转移地址的 4 种寻址方式。

3.1 学 习 要 点

1. 微处理器的内部结构

从微处理器(也称中央处理单元，即 CPU)的内部结构，可以了解 CPU 的工作过程，这对掌握汇编语言的编程是很有好处的。

典型的微处理器内部结构可分成 4 个组成部分：

(1) 算术逻辑运算单元(ALU)：CPU 的核心，完成所有的算术和逻辑运算操作。

(2) 工作寄存器：用于暂存寻址信息和计算中间结果。

(3) 控制器：CPU 的“指挥中心”。在它的控制下，CPU 才能完成指令的读入、寄存、译码和执行。

(4) I/O 控制逻辑：处理 CPU 的 I/O 操作。

区分下列这些名词解析：程序计数器(PC，Program Counter)、指令寄存器(IR，Instruction Register)、指令译码器(ID，Instruction Decode)、控制逻辑部件、堆栈指针(SP，Stack Pointer)、处理器状态字(PSW，Processor State Word)。

2. 微处理器的外部结构

CPU 的引脚信号通过逻辑部件的处理和组合，构成了系统总线：

(1) 数据总线(16 位，注意，对 8088 CPU 而言，只有 8 位)：用于传送信息。

(2) 地址总线(20 位)：用于传送地址码，可寻址 $2^{20}=1$ MB 空间。

(3) 控制总线(16 条)：用于控制用户设计的各个逻辑部件。

在以 8086/8088 CPU 构成的系统中，存储器地址空间与端口地址空间分开，采用两个独立的地址空间：存储单元地址采用 A_0～A_{19}编址，端口地址采用 A_0～A_{15}编址。

3. 微处理器的功能结构

在功能上，8086/8088 CPU 由两个独立的逻辑单元组成：执行单元(EU)和总线接口单元(BIU)。EU 用于完成指令所要求的运算操作；而 BIU 用于完成指令地址计算和 CPU 通过系统总线访问存储器时的地址计算，也即由逻辑地址计算出物理地址。这两个单元是独立、并行执行的。

4. 寄存器结构

8086/8088 CPU 内部有 14 个 16 位的寄存器，它们可分为三组：通用寄存器(8 个)，段寄存器(4 个)和控制寄存器(2 个)。

5. 物理地址与逻辑地址

逻辑地址的表示形式为“段地址：偏移地址”，其相应的物理地址为：段地址×10H＋偏移地址。例如，0800：01A0 的物理地址为：0800H×10H＋01A0H＝081A0H。

6. 控制寄存器

控制寄存器有 2 个(16 位)：IP(Instruction Pointer)指令指针和 PSW(Processor State Word)微处理器状态字。

IP 相当于程序计数器 PC，用于保存下一条要执行指令的段内偏移地址。

PSW 中定义了 9 个标志位。其中，状态标志位 CF、AF、ZF、SF、OF 和 PF 用于表示上一次 CPU 运算操作的状态，控制标志位 DF、IF 和 TF 用于控制 CPU 的后续操作。

7. 存储器分段的基本概念

从存储器的最低端开始，每 64 KB 构成一个段，但每个能被 16 整除的地址都可以是一个新段的开始，因此，段与段之间是互相覆盖的。相邻两个段之间相差 16 个单元。

为了指示一个存储单元，除了需要指定偏移地址之外，还需要指出其段地址。任何一个汇编语言程序，虽然可以包含任意多个段，但当前使用的段只有 4 个，分别用 CS、DS、ES 和 SS 来指示。

8. 字存储格式

由于 8086 CPU 有 16 条数据总线，因此在理想情况下，一个总线周期可以读写一个字，但这要求字的存储是对准的，即低位字节存放于偶地址，相应的高位字节存放于相邻的奇地址。对于未对准存储的字需要 2 个总线周期进行读写。应该注意，对于 8088 CPU，由于它只有 8 位的数据总线，一个字无论怎样都需要 2 个总线周期。

对于 I/O 端口，也有类似的结论。

9. 寻址方式

在指令中，用于说明操作数所在地址的方法，称为寻址方式。寻址方式又可以分成数

据的寻址方式(最常用的有 8 种)和转移地址的寻址方式(4 种)。

3.2 典型例题

例 3.1 有一块 120 个字的存储区域，其起始地址为 625A：234D，写出这个存储区域首末单元的物理地址。

解 存储区域的字节数为

$$2\times120=240=0F0H$$

首地址为

$$625AH\times10H+234DH=648EDH$$

末地址为

$$648EDH+0F0H-1=649DCH$$

或者 $625AH\times10H+(234DH+0F0H)-1=625A0H+243DH-1=649DCH$

例 3.2 两个十六进制数 7825H 和 5A1FH 分别相加和相减后，求运算结果及各标志位的值。

解 7825H+5A1FH=0D244H，AF=1，CF=0，ZF=0，SF=1，OF=1(当将 7825H 和 5A1FH 看作有符号数时，两个正数相加得到一个负数，结果显然是错误的。实际上，在运算过程中，次高位产生了进位而最高位没有产生进位，故运算产生溢出)，PF=1(因为在 44H 中包含有偶数个 1)。

7825H−5A1FH=1E06H，AF=1，CF=0，ZF=0，SF=0，OF=0，PF=1。

5A1FH−7825H=0E1FAH，AF=0，CF=1，ZF=0，SF=1，OF=0，PF=1。

3.3 习　题

1. 微处理器内部结构由哪几部分组成？阐述各部分的主要功能。

2. 微处理器级总线有哪几类？各类总线有什么作用？

3. 为什么地址总线是单向的，而数据总线是双向的？

4. 8086/8088 微处理器内部有哪些寄存器？其主要作用是什么？

5. 如果某微处理器有 20 条地址总线和 16 条数据总线：

(1) 假定存储器地址空间与 I/O 地址空间是分开的，则存储器地址空间有多大？

(2) 数据总线上传送的有符号整数的范围有多大？

6. 将十六进制数 62A0H 与下列各数相加，求出其结果及标志位 CF、AF、SF、ZF、OF 和 PF 的值：

(1) 1234H　　(2) 4321H

(3) CFA0H　　(4) 9D60H

7. 从下列各数中减去 4AE0H，求出其结果及标志位 CF、AF、SF、ZF、OF 和 PF 的值：

(1) 1234H　　(2) 5D90H

(3) 9090H　　(4) EA04H

8. 什么是逻辑地址？什么是物理地址？它们之间的关系如何？

9. 写出下列存储器地址的段地址、偏移地址和物理地址：

(1) 2134：10A0

(2) 1FA0：0A1F

(3) 267A：B876

10. 给定一个数据的有效地址为 2359H，并且(DS)＝490BH，求该数据的物理地址。

11. 在一个程序段开始执行之前，(CS)＝0A7F0H，(IP)＝2B40H，求该程序段的第一个字的物理地址。

12. 下列操作可使用哪些寄存器？

(1) 加法和减法　　(2) 循环计数

(3) 乘法和除法　　(4) 保存段地址

(5) 表示运算结果的特征　　(6) 指令地址

(7) 从堆栈中取数的地址

13. IBM PC 有哪些寄存器可用来指示存储器的地址？

14. 设(BX)＝637DH，(SI)＝2A9BH，位移量＝0C237H，(DS)＝3100H，求下列寻址方式产生的有效地址和物理地址：

(1) 直接寻址　　(2) 用 BX 的寄存器间接寻址

(3) 用 BX 的寄存器相对寻址　　(4) 用 BX 和 SI 的基址变址寻址

(5) 用 BX 和 SI 的基址变址且相对寻址

15. 若(CS)＝5200H，物理转移地址为 5A238H，那么(CS)变成 7800H，物理转移地址为多少？

16. 设(CS)＝0200H，(IP)＝2BC0H，位移量＝5119H，(BX)＝1200H，(DS)＝212AH，(224A0H)＝0600H，(275B9H)＝098AH。求使用下列寻址方式时的转移地址：

(1) 段内直接寻址方式；

(2) 使用 BX 的寄存器寻址的段内间接寻址方式；

(3) 使用 BX 的寄存器相对寻址的段内间接寻址方式。

17. 将下列两组的词汇和说明关联起来：

(1) CPU　　(2) EU

(3) BIU　　(4) IP

(5) SP　　(6) 存储器

(7) 堆栈　　(8) 指令

(9) 状态标志　　(10) 控制标志

(11) 段寄存器　　(12) 物理地址

(13) 汇编语言　　(14) 机器语言

(15) 汇编程序　　(16) 连接程序

(17) 目标码　　(18) 伪指令

A. 保存当前栈顶地址的寄存器

B. 指示下一条要执行指令的地址
C. 总线接口部件，实现执行部件所需要的所有总线操作
D. 分析并控制指令执行的部件
E. 存储程序、数据等信息的记忆装置，PC 机有 RAM 和 ROM 两种
F. 以后进先出方式工作的存储器空间
G. 把汇编语言程序翻译成机器语言程序的系统程序
H. 惟一代表存储器空间中的每个字节单元的地址
I. 能被计算机直接识别的语言
J. 用指令的助记符、符号地址、标号等符号书写程序的语言
K. 把若干个模块连接起来成为可执行文件的系统程序
L. 保存各逻辑段的起始地址的寄存器
M. 控制操作的标志，PC 机有三位：DF、IF、TF
N. 记录指令操作结果的标志，PC 机有六位：OF、SF、ZF、AF、PF、CF
O. 执行部件，由算术逻辑单元(ALU)和寄存器组等组成
P. 由汇编程序在汇编过程中执行的指令
Q. 告诉 CPU 要执行的操作，在程序运行时执行
R. 机器语言代码

第 4 章　汇编语言程序设计基本方法

全面掌握 8086、8088 CPU 指令系统的使用，包括指令的功能、寻址方式及其书写格式、对标志位的影响、使用注意事项。掌握汇编程序设计所必需的伪指令，并由此构成汇编程序的完整结构。掌握变量、常量及伪指令的使用和一些常用的基本程序设计方法。在分支程序设计中，要特别注意每个分支的完整性和分支条件的合理使用；在循环程序设计中，掌握循环程序的基本结构，特别注意应避免出现死循环；在子程序设计中，着重掌握参数的各种传递方式及其实现，对堆栈这种特殊的存储区域进行了详细的描述，切实掌握堆栈的使用。宏指令与字符串操作是汇编语言设计中的两个难点，教材中对此也作了详细的介绍，要求掌握正确使用宏指令和字符串操作指令。

教材中简要介绍了 DOS 功能调用的使用方法和常用的一些 DOS 功能，要求能熟练使用 INT 21H 的 01，02，09，0AH，4CH 号等功能。

4.1 学 习 要 点

1. 基本概念

掌握机器语言程序、机器语言编程、汇编语言程序、汇编语言程序设计等基本概念并熟悉汇编、连接等过程，能够进行汇编语言的程序设计。

2. 汇编语言指令

在汇编语言程序设计中，有三类指令：指令、伪指令和宏指令。

(1) 指令。它对应于二进制代码指令，汇编后形成一条机器语言指令，指示 CPU 进行各种操作。它在程序执行时才得到执行。

(2) 伪指令。它只告诉汇编程序(MASM. EXE)应如何汇编，而本身并不形成机器语言指令。它在源程序汇编的过程中得到执行。

(3) 宏指令。这是用户自己定义的指令。它由一组指令、伪指令构成，并在汇编过程中进行宏展开。它也是一种伪指令，也没有对应的机器语言指令。实际上，可将一组程序段用一个宏指令名来表示。

3. 汇编语言程序设计的一般步骤

(1) 分析问题；(2) 确定算法；(3) 编写程序；(4) 调试程序；(5) 编写说明。

4. 汇编语言的语句格式

汇编语言的语句由四部分组成：

[名称：] 操作助记符 操作数 [；注释]

[名称] 操作助记符 操作数 [；注释]

其中，第一种为指令格式，其名称部分表示标号；第二种为伪指令格式，其名称部分表示变量名。在指令中，操作数至多可以有两个，而伪指令中操作数可以有多个。在操作数之间，用逗号(，)间隔。

5. 标号

标号主要用于表示一条指令的位置，以便转移、循环、子程序调用等指令能够转移到这条指令所在的位置。标号具有三个属性：段地址(用于指示标号所在的段)、偏移地址(用于指示标号在段内的偏移地址)和类型。

6. 变量

变量用于保存程序中要用到的可变的量，这些量可由程序来改变。变量具有五个属性：段地址(用于指示变量所在的段)、偏移地址(用于指示变量在段内的偏移地址)、类型、长度和大小。

7. 属性操作符

无论是标号，还是变量，通过属性操作符得到的值为立即数，因此，应将它看作是一个无类型的量。

8. PTR 操作符

PTR 操作符可用来暂时改变变量或标号的类型。在双操作数指令中，当两个操作数都不指定类型(类型不定)，或者当两个操作数的类型不一致时，都会发生错误，这时可采用 PTR 操作符暂时改变变量的类型。PTR 操作符的使用格式为：

类型 PTR 表达式

注意，PTR 操作符只在本行起作用。例如，MOV AL，BYTE PTR VAR1(VAR1 为字变量)。

9. 伪指令 DW，DD 的特殊用法

变量名 1 DW 标号(或变量名 2)±常数

变量名 3 DD 标号(或变量名 4)±常数

定义的＜变量名 1＞为字型地址指针，其内容为＜标号±常数＞或＜变量名 2±常数＞的段内偏移地址；定义的＜变量名 3＞为双字型地址指针，其内容为＜标号±常数＞或＜变

量名 4±常数>的段内偏移地址和段地址，例如：

```
AD1 DB 100 DUP(? )          ；设变量 AD1 逻辑地址为 0100：2157
AD2 DW AD1                  ；变量 AD2 内容为 2157H
AD3 DD AD1                  ；变量 AD3 内容为 2157H，0100H
```

10. EQU 和“=”伪指令

利用 EQU 和“=”伪指令，可以用有意义的标识符代替表达式或常数，这样能够方便用户进行程序设计。

名称　EQU　表达式

名称 = 常数

利用“=”伪指令定义的名称还可以重复定义。

11. MOV 指令传送图

MOV 指令可在立即数、通用寄存器、段寄存器、存储器之间传送数据，其传送路径可参见教材的图 4.3。

需要特别注意的是，利用 MOV 指令不能直接传送的路径有 5 条：

(1) 立即数→段寄存器；

(2) 存储单元→存储单元；

(3) 段寄存器→段寄存器；

(4) 其它→CS；

(5) 其它→立即数。

除最后两条路径外，前三条路径可分两步实现。例如，要将立即数 12A6H 传送到段寄存器 DS，应分两步：

```
MOV AX，12A6H
MOV DS，AX
```

需要说明的是，MOV 指令的这种传送路径也适用于其它的双操作数指令，如 ADD，ADC，SUB，SBB 等指令。

12. 操作数类型

对于一个操作数的类型，下列几点值得特别注意：

(1) 立即数是无类型的；

(2) 不含变量名的直接寻址、寄存器间接寻址、寄存器相对寻址、基址变址寻址、基址变址且相对寻址的操作数为无类型；

(3) 利用 PTR 操作符可暂时改变存储单元的类型。

对于双操作数指令，两个操作数的类型必须匹配：

(1) 两者都指定了类型，则必须一致，否则指令出错(类型不一致)；

(2) 两者之一指定了类型，则一般指令无错；

(3) 两者都无类型，则指令出错(类型不定)。

13. BP 寄存器

当 BP 寄存器用作为基址寄存器时，其默认的段寄存器为 SS，这与其它的基址变址寄存器(BX、SI、DI)不同。但是，当指令中包含有变量名时，其默认的段地址与变量所在的段地址相同。例如，假设变量 VAR1 所在的段用 DS 来指示，则 MOV AX，[BP+4]的段寄存器为 SS，而指令 MOV AL，VAR1[BP+2]的段寄存器为 DS。

14. 堆栈

堆栈是一块特殊的存储器区域，这块区域以先进后出(FILO，First In Last Out)的方式工作，系统为此提供了特殊的指针 SP 和段寄存器 SS。

15. 两数的比较

在两数大小比较时，我们总是说，“CF 是对无符号数而言的，而 OF 是对有符号数而言的”，对这句话的正确理解并不容易。

在对两数执行算术运算或比较指令时，计算机并不区分操作数是无符号数还是有符号数，它总是将操作数当作二进制数进行运算，并按照无符号数的运算规则设置 CF，按照有符号数的运算规则设置 OF。

两数运算的结果是否超出范围，取决于实际参加运算的两个操作数。当参加运算的两个操作数为无符号数时，则可根据 CF 确定运算结果是否溢出；当参加运算的两个操作数为有符号数时，则应根据 OF 确定运算结果是否溢出。

类似的结论可推广到两数的大小比较。对两个无符号数，其大小可直接根据 CF 来确定；但对两个有符号数，其大小关系应根据 OF 与 SF 共同确定。

16. 符号扩展指令的正确用法

符号扩展指令 CBW 可将 AL 中的符号位扩展到 AH 中，即：当 AL 的 D7＝0 时，AH＝00H；当 D7＝1 时，AH＝0FFH。这样，AL 经符号扩展指令 CBW 扩展后得到一个字。同样，AX 经符号扩展指令 CWD 扩展后得到一个双字。

应该注意一点：符号扩展指令适用于有符号数，反言之，有符号数可以并且只能采用符号扩展指令将一个字节(字)扩展成一个字(双字)。而无符号数绝对不能采用符号扩展指令，当需要将一个字节(字)扩展成一个字(双字)时，只需将高位直接清零。

17. 逻辑运算指令

逻辑运算指令是位操作指令，某一位的运算结果不会影响其它位的运算。因此，有 CF＝0，AF＝0，OF＝0，其它标志位(SF、ZF 和 PF)视运算结果而定。特别值得一提的是，NOT 指令对 6 个标志位都没有影响。

通过逻辑指令可以完成某些特殊的操作：

(1) 寄存器清零：

```
XOR AX, AX              ; (AX)←0
```

(2) 小写字母的 ASCII 码变成大写字母的 ASCII 码(设 AL 中存放字母的 ASCII 码)：