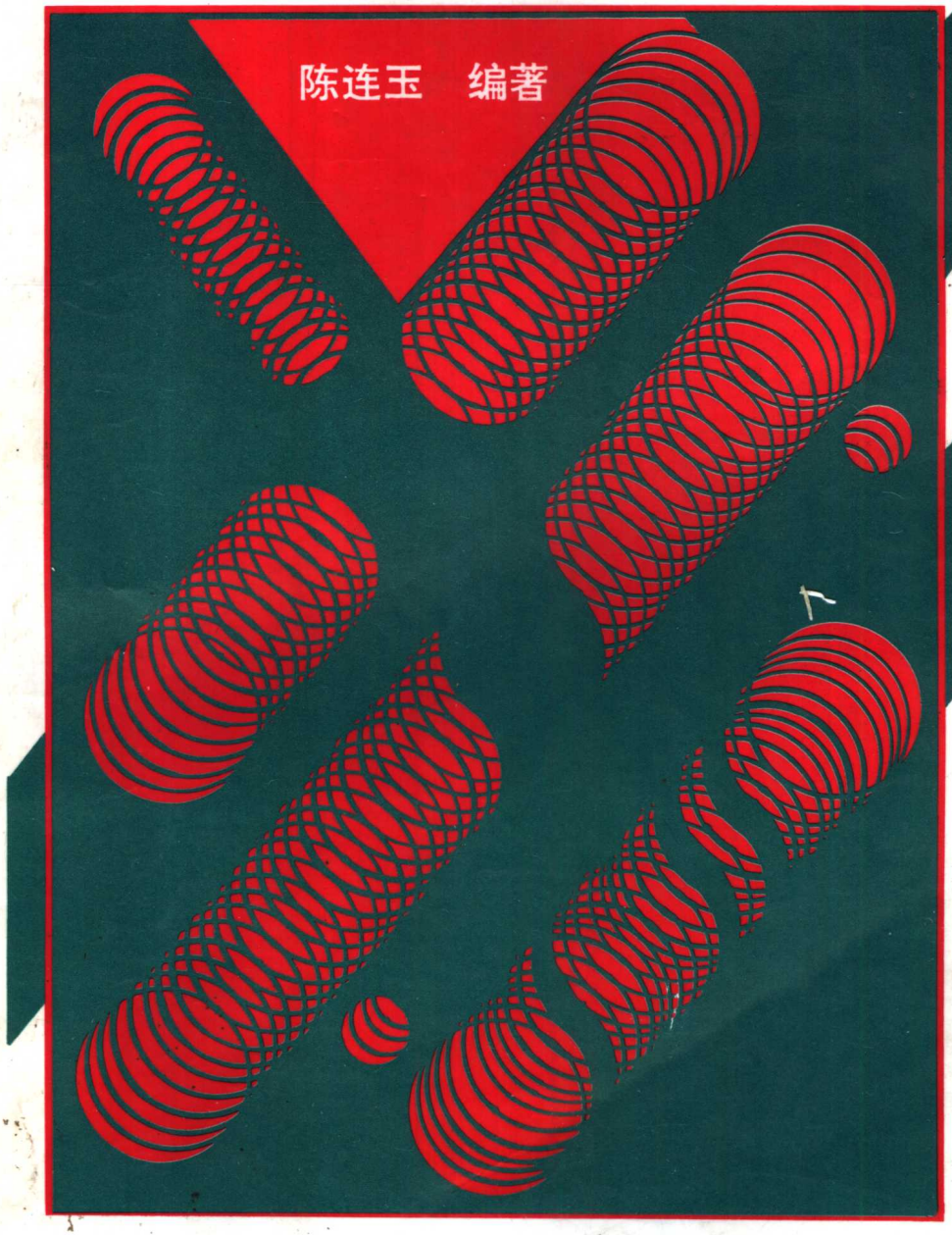
fei ji suan ji zhuan yei xi lie jiao cai

非计算机专业系列教材

汇编语言简明教程

HUI BIAN YU YAN JIAN MING JIAO CHENG



大连理工大学出版社

非计算机专业系列教材

汇编语言简明教程

陈连玉　编著

大连理工大学出版社

(辽)新登字 16 号

内容提要

本书是作者在多年教学实践的基础上编写而成的。书中针对 IBM-PC 系列微型机上使用的汇编语言系统，着重介绍了该语言的基本概念、基本指令和各种程序设计方法，并配有一定数量的习题。

本书可作为高校非计算机专业、计算机类各专业(低学时要求)、大专、中等专业学校师生以及工程技术人员学习汇编语言程序设计的教材和辅修课教材，也可以作为成人自学考试的教材和参考书。

汇编语言简明教程

陈连玉 编著

* * *

大连理工大学出版社出版发行
(大连市凌水河 邮政编码 116024)
大连理工大学印刷厂印刷

* * *

开本：787×1092 1/16 印张：14 字数：328 千字
1996 年 6 月第 1 版 1996 年 6 月第 1 次印刷
印数：1—5000 册

* * *

责任编辑：郭学满 刘新锋 责任校对：寸 土
封面设计：孙宝福

* * *

ISBN 7-5611-1151-7
TP · 106 定价：12.00 元

前 言

在人类即将步入信息社会的今天，外语和计算机已成为各类工程技术人员不可缺少的工具，有人曾形象地称其为工程技术人员的两条腿，可见电子计算机的重要地位。

汇编语言程序设计是计算机类各专业的主要技术基础课程之一，也是非计算机专业人员学习计算机应用知识的必修课。因此掌握汇编语言对于研制系统软件、系统软件的二次开发、使用计算机完成实时控制等都有着十分重要的作用。本书从认识论的基本原则出发，力求做到由浅入深、突出重点、分解难点、方便记忆、便于自学，理论与实践并重。书中既介绍汇编语言的基本概念、基本指令，又介绍它们的具体使用——程序设计的各种结构及其实现方法；既重视程序设计的技巧、风格，又重视上机实践。由于汇编语言是实践性很强的一门课程，只有通过大量的上机训练，调试一定数量的程序，才能较好地掌握所学的内容。希望读者千万不要忽视这一点。在某种意义上说，计算机本身就是一位好老师，其前提就是熟练地掌握上机操作的各种步骤和程序的调试运行。做到这一点，程序编制得正确与否，一上机就能检验出来。

本书共分十一章。第一章从程序设计的角度介绍微型机的结构，包括微处理器、存储器、内存储器的地址分段、指令的寻址方式。第二章介绍汇编语言，包括常量、变量、各种运算符、表达式、指令格式和各种伪指令。考虑到系统性，这部分单设一章，讲授时可穿插到各章中去，例如，讲过子程序后，再介绍宏指令较好，便于两者的比较。第三章介绍顺序结构程序设计。第四章介绍分枝结构程序设计。第五章介绍循环结构程序设计。第六章介绍子程序。第七章介绍输入输出程序设计。第八章介绍中断。第九章介绍汇编语言在图形方面的应用。第十章介绍 80286/80386 系统。第十一章介绍上机操作过程。各章之后有一定数量的习题。

作者在编写本书过程中得到了大连理工大学计算机系张华同志的大力帮助，她为本书的出版付出了辛勤的劳动，在此表示衷心的感谢！由于时间紧迫，书中谬误之处一定不少，敬请各位读者提出宝贵意见。

编者

1996 年 1 月于大连

目　录

第一章 IBM-PC 微型机的结构

微型计算机的基本结构如图 1.1 所示。它主要包括这样几大部分：微处理器、内存储器、输入/输出设备、输入/输出接口和总线。

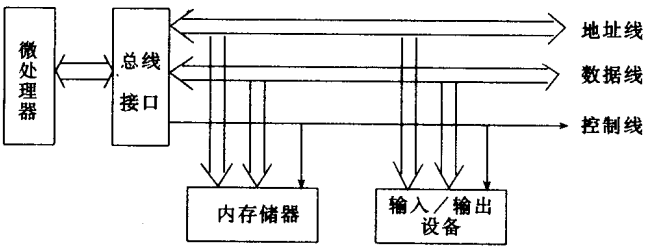


图1.1 微型计算机的基本结构

第一节 微处理器

IBM-PC 微型计算机上使用的微处理器(也称为微处理机)是 INTEL 公司生产的 8088 微处理器，它相当于一般计算机中的中央处理机(CPU)，是整个机器的核心部件，其内部结构有两大部分，一部分是可执行部件 EU，另一部分是总线接口部件 BIU。其结构如图 1.2 所示。

由图 1.2 可以看出，8088 微处理器能够完成 16 位数据信息的传输，但是它和外部总线，比如同内存储器进行数据交换时，是以字节为单位进行的，因此通常被称为 8 位微处理器，也被称为准 16 位微处理器，因为其中的累加器、运算器、寄存器都可以处理 16 位的信息。下面分别讨论微处理器的两大部件。

一、可执行部件 EU

可执行部件由算术逻辑部件 ALU、通用寄存器、标志寄存器和控制逻辑电路组成，完成各种数据信息的加工、处理以及有效地址的计算。

1. 算术逻辑部件 ALU 和控制逻辑电路

算术逻辑部件完成数据信息的加工和各种运算。控制逻辑电路负责有关指令的译码、解释执行，完成各种规定的动作。与此同时，根据操作的情况，请求接口部件接通总线，进行数据传递；还可根据指令的执行情况，向总线接口部件 BIU 发出各种控制信号和请求信号等。

2. 通用寄存器

8086/8088 微处理器的可执行部件中设有 8 个 16 位的通用寄存器，用于完成进行算术运算、逻辑运算的 8 位操作数或者 16 位操作数的存放。8 个通用寄存器分为两组，一组是数据寄存器，另一组为指针和变址寄存器。数据寄存器有 AX，BX，CX 和 DX，其中每一个都可

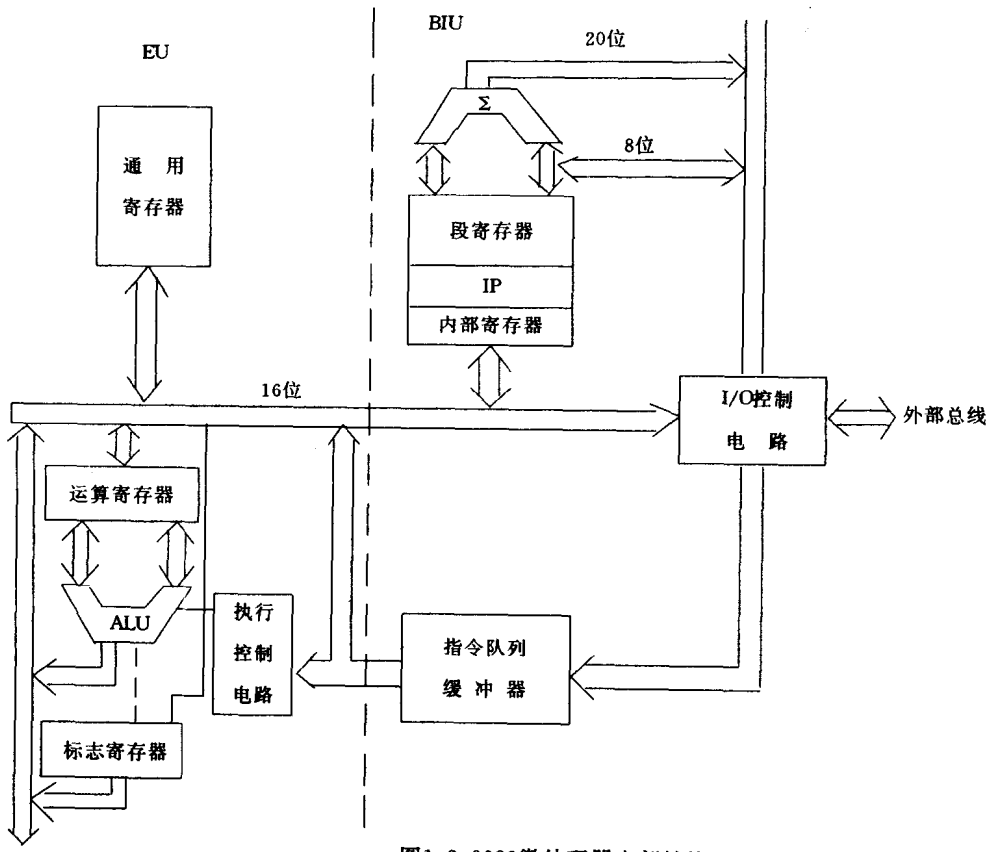


图1.2 8088微处理器内部结构

以存放16位二进制信息。由于这4个数据寄存器的每一个又可以作为两个8位寄存器使用，即高8位作为一个寄存器，低8位作为另一个寄存器，所以也可以称为H,L寄存器。例如，AX可以作为16位寄存器，而AH,AL分别作为八位寄存器。它们都可以根据用户的要求存放算术、逻辑运算的操作数。指针和变址寄存器有BP,SP,SI和DI，其中BP称为基础指针(或基址寄存器)，SP称为堆栈指针(或堆栈指示器)，SI称为源变址寄存器，DI称为目的变址寄存器。它们主要用于操作数有效地址的确定。通用寄存器示意图如图1.3所示。

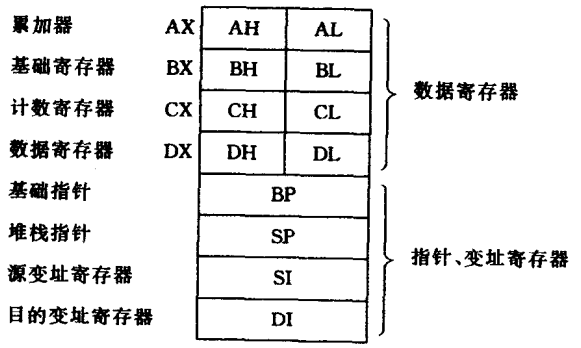


图1.3 通用寄存器

3. 标志寄存器

标志寄存器亦称为程序状态字(PSW)。它反映算术、逻辑运算等操作后的结果特征以及程序设置的状态信息。该寄存器共有 16 位,但只使用其中的 9 位,最高四位和第 1,3,5 位不用,通常为 0。使用的各位的名称及含义如图 1.4 所示。

图 1.4 标志寄存器

第 0 位为进位标志位 C(Carry),它反映算术运算中最高位是否有进位(加法)或者借位(减法)的情况以及移位操作后,操作数某一位的相应值。当加法运算时,如果最高位有进位,则该位为 1,若没有进位,该位为 0。当减法运算时,如果最高位有借位,则该位置 1,否则,该位置 0。通常对某一位置 1,也称为置位,而对某一位置 0 也称为复位。

第 1 位不用,通常为 0。

第 2 位为奇偶标志位 P(Parity),它反映某一操作完成后,其结果操作数低八位中含有 1 的个数是奇数还是偶数。若为偶数,此位置位,否则复位。通常用于传递数据时错误检查。

第 3 位不用,通常为 0。

第 4 位为辅助进位标志位 A(Auxitiary Carry),亦称半进位标志位。在字节运算中,它反映低四位是否向高四位进位或借位的情况,即一字节中的第 3 位是否向第四位进位(加法)或借位(减法)的情况。若有进位或借位则此位置位,否则此位复位。

第 5 位不用,通常为 0。

第 6 位是零标志位 Z(Zero),它体现某一操作的结果是否为零,如果为零则此位置 1,若结果非零(不为 0)则此位置 0。

第 7 位为运算结果的符号标志位 S(Sign),它反映操作结果的最高位的状态,即其值等于结果最高位的值。由于 8086/8088 汇编语言中,对有符号的数采用补码表示,因此这一标志位反映有符号数的符号,如果其值为 0,表示为正数,其值为 1,表示为负数。

第 8 位是追踪标志位 T(Trap),此标志位是为调试程序设置的,当此位为 1 时,进入追踪方式(或称单步方式)。在这种方式下,CPU 每执行一条指令后,产生一个内部中断,允许用户进行各种检查。

第 9 位为中断开放标志位 I(Interupt—enable),它体现 CPU 是否允许接受中断请求信号,若此标志位置 1,则允许接受中断请求,否则禁止中断请求。

第 10 位为方向标志位 D(Direction),在字符串操作时,它反映变址器中的内容是自动增值还是减值。如果此标志位为 0,则为增值;否则为减值。

第 11 位是溢出标志位 O(Overflow),在算术运算中,它反映有符号数运算的结果是否超出了寄存器所能表示数的范围,如果超出,则此标志位置位,否则复位。

二、总线接口部件 BIU

如果将 8088 微处理器的可执行部件看作一般计算机的运算器,而它的总线接口部件类

似于一般计算机的控制器。它具有以下功能：保存即将执行的指令，将稳定的指令状态提供给可执行部件；计算下一条执行指令的地址，控制程序执行的顺序，保证程序的正确运行；控制并完成通用寄存器和内存储器、I/O 设备间的信息交换等。由图 1.2 可见 BIU 由段寄存器、指令指针寄存器、地址加法器、指令队列缓冲器和 I/O 控制电路组成。其中指令物理地址的形成由程序段寄存器、指令指针寄存器在地址加法器中确定；从内存储器中取出指令的暂存由指令队列缓冲器完成，而 I/O 电路完成微处理器和内存储器或 I/O 设备的信息交换。

1. 指令指针寄存器 IP

指令指针寄存器又称为指令指示器、指令计数器、程序指示器、程序计数器等。它是一个 16 位寄存器，能存放 2 个字节的信息，其中存放的内容是将要执行的指令的首字节在程序段内的地址偏移量，它控制着程序指令执行的顺序。

2. 段地址寄存器

8086/8088 微处理器的 BIU 中共有 4 个 16 位的段地址寄存器 CS，DS，SS，ES。

其中程序段寄存器 CS 用于存放程序所在内存储器中地址单元的段地址，数据段寄存器 DS 用于存放数据所在内存储器中地址单元的段地址；堆栈段寄存器 SS 则用于存放堆栈段的段地址；而附加段寄存器 ES 通常是用于存放操作数或缓冲区信息段的段地址，一般是对字符串操作时用作目的操作数的段地址。

3. 地址加法器

地址加法器的功能是将某个段寄存器的内容及相应的段内偏移量作适当的处理以形成操作数或指令所在内存储器中的物理地址。具体实现过程见图 1.5。

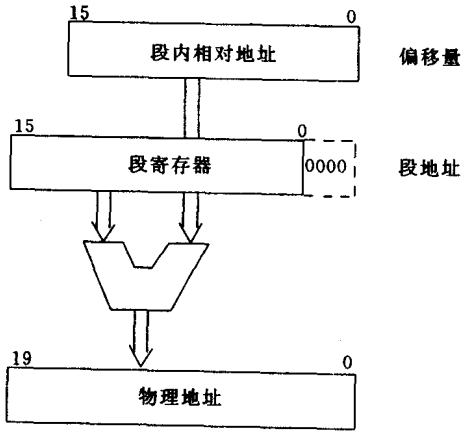


图1.5 物理地址的形成

4. 指令队列缓冲器

8088 微处理器的指令队列缓冲器是一组暂存指令的寄存器，亦可称为指令栈，由 4 个 8 位的寄存器组成。因此它可以存放四个字节的指令信息。

第二节 存储器

存储器是存放信息的记忆装置。分主存储器(或内存储器)和辅助存储器(或外存储器)两种。辅助存储器通常是利用磁性介质的磁化技术存取信息。例如,软磁盘、磁鼓、磁带、硬磁盘等。主存储器是计算机的核心部件,它类似于人的大脑,可以存放和记忆各种信息,例如:程序指令、数据、程序运行中需要的各种参数,指令执行的中间结果、最后结果等。因此有时亦将存储器称为记忆装置。主存储器是由一系列存储元件构成的。IBM-PC 微型计算机上使用的主存储器是半导体集成电路存储器。这种存储器又分为只读存储器(ROM)和随机存储器(RAM)。只读存储器只允许从中读取信息,而不允许写入信息,随机存储器则既能读出信息,又能写入信息。读出和写入也可以使用术语取出和存入来描述,因此读/写和取/存是同义语。ROM 通常用于存放系统提供的标准程序,例如 BASIC 解释程序、监控程序等。RAM 是提供给用户使用的存储区,用户可以根据需要存放程序、数据、运算结果等各种信息。主存储器中存放信息的基本存储单元通常是字节,每个字节存放 8 位二进制代码信息。对于存储器以字节作为基本存储单元的主存储器来说,信息的存取是以字节为基本单位,即 8 位二进制代码是作为一个整体从存储器中读出或向其写入的。也就是说取存信息,遵循"整存整取"的原则。此外对主存储器存储单元的读/写还应注意以下两点:读出信息时,除整体取出外,原存储单元中的信息内容仍然存在,丝毫没有被改变,所以从存储单元中取信息时有"取之不尽"之称。向存储单元中存入信息时,除了整体写入外,还破坏了原来存储单元中的信息,即新写入的信息替代了原有的信息。概括起来称之为"一冲就掉"。将存储单元的信息读写原则综合一下就是"存取信息,整存整取,取之不尽,一冲就掉"。

第三节 内存储器的地址分段

8086/8088 微处理器有 20 位的地址线,寻址范围可以达到 1MB(2^{20}),因此,在 IBM-PC 微型计算机上可以配有多达 1 兆字节的内存储器。由于每个字节作为内存储器的基本存储单元被分配一个地址号,因此 1MB(2^{20})的内存储器的编址范围是 00000H～FFFFFH,相当于十进制数的 0～1048575 共 1048576 个地址号。如前所述,8086/8088 微处理器的寄存器最多是 16 位的,要想形成 20 位的地址值是通过总线接口部件中的地址加法器将段寄存器(CS,DS,SS,ES 之一)的内容左移四位(相当于乘 16)同段内偏移量相加而得到。其中段内偏移量可能是立即数,可能是寄存器的值,也可能是立即数和寄存器的组合值。

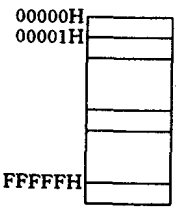


图 1.6 内存储器编址

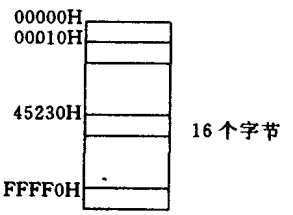


图 1.7 可能的段开始地址

由于段寄存器是16位寄存器，所能表示数的范围是0000H～FFFFH，形成地址的过程又将其乘以16(左移4位)，所以段寄存器的内容实际上相当于将内存储器每16个字节分成一段的段首址，即0000H～FFFFH分别与内存储器的00000H，00010H，…FFFF0H的地址号相对应。但是由于16个字节的分段容量太小，所以8086/8088微处理器提供0～64K的段内偏移量。因此对于内存储器某个地址号的确定就有多种段地址和偏移量的组合形式，其最大组合数应为4K。例如，对于内存储器地址为1FFFFH的单元，其段地址和段内偏移量的组合有：段地址为10000H时，偏移量为FFFFH；段地址为10010H时，偏移量为FFEFH；段地址为10020H时，偏移量为FFDFH；…；段地址为1FFF0H时，偏移量为000FH。图1.6和图1.7分别绘出了主存储器的编址和可能的段地址。如果按最大偏移量64K作为分段的依据，内存地址空间可划分成16段，此时段地址值为00000H，10000H，20000H，…，F0000H，对应的段寄存器值为0000H～F000H。也可以理解为16个段的段地址0～16由段寄存器的最高4位来表明，而段内偏移量则由段寄存器的内容的低12位乘以16的值与偏移量的和。图1.8示出了这种分段。8086/8088微处理器的四个段寄存器，在程序设计中的分工体现在对内存储器有关存储区的访问。

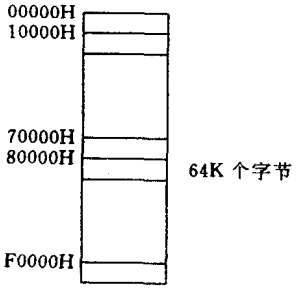


图1.8 将内存分成16段

1. 访问程序区

使用8086/8088汇编语言指令编写的程序存放在内存储器的某一区域中，该区域称为程序区。对程序区的访问，就是要确定程序中某条指令所在内存储器中的地址，从而取出并执行该指令。某指令在内存储器中的物理地址是由代码段寄存器CS和指令指针寄存器IP的内容确定的，即代码段寄存器的内容乘以16加上IP的内容，就是指令的物理地址。见图1.9(a)。顺便提一下，段寄存器CS的内容始终是当前执行的程序段地址，IP则指向程序段内即将执行的指令。

2. 访问数据区

内存储器中存放数据的区域称为数据区。为了确定程序中指令的某个操作数所在内存储器中数据区的存储单元地址，通常使用数据段寄存器DS存放数据区的段地址，而段内偏移量由某种组合的有效地址值指定。换句话说数据段寄存器DS的值乘以16与某种组合方式确定的有效地址值(EA)相加，其和就是操作数所在内存数据区的地址。对有些程序指令，例如字符串操作，有时要涉及到内存储器中两个数据块中的数据，因此需要确定源操作数地址和目的操作数的地址。8086/8088汇编语言规定：字符串操作的源操作数地址由数据段寄存器DS和源变址器SI来确定；而目的操作数地址则由附加段寄存器ES和目的变址器DI来确定。其物理地址的计算均为相应段寄存器的内容乘以16的积再与变址器的内容相加。对数据区的访问示意图见图1.9(c)和(d)。

3. 访问堆栈区

程序设计中常常使用堆栈区保存现场信息，子程序的返回地址等。对堆栈区信息的访问使用堆栈段寄存器SS和堆栈指针SP。即将堆栈段寄存器SS的内容乘以16再加上SP的内容以确定内存储器中20位的物理地址。寻址过程见图1.9(b)。

值得注意的是四个段寄存器的内容初始化和修改方式是不尽相同的。其中代码段寄存

器 CS 只能通过转移指令(JMP)、调用指令(CALL)、返回指令(RET)、中断指令(INT)和中断返回指令(IRET)等改变。而其余段寄存器 DS,ES,SS 则可以通过寄存器传送指令赋值或修改。

综上所述,由于段内偏移量的最大值为 64KB,如果在设计程序时,程序区、数据区和堆栈区的长度均小于或者等于 64KB,则只需要在程序开始指定段寄存器 CS,DS,SS 的值,尔后不再考虑段寄存器,程序就能正常地在各区内工作。尤其是当程序中所使用的程序区、数据区和堆栈区的信息总长度小于或等于 64KB 时,在程序一开始甚至可以将 CS,DS 和 ES 赋予相同的值,程序亦能正常运行。如果某个区的信息量超过 64KB 时,程序中将要涉及段寄存器内容的变动。对于数据段寄存器 DS 的内容只要赋予不同的值,即可实现不同数据段的访问;而对于代码段寄存器 CS 的内容变动则可以实现程序段的再定位。

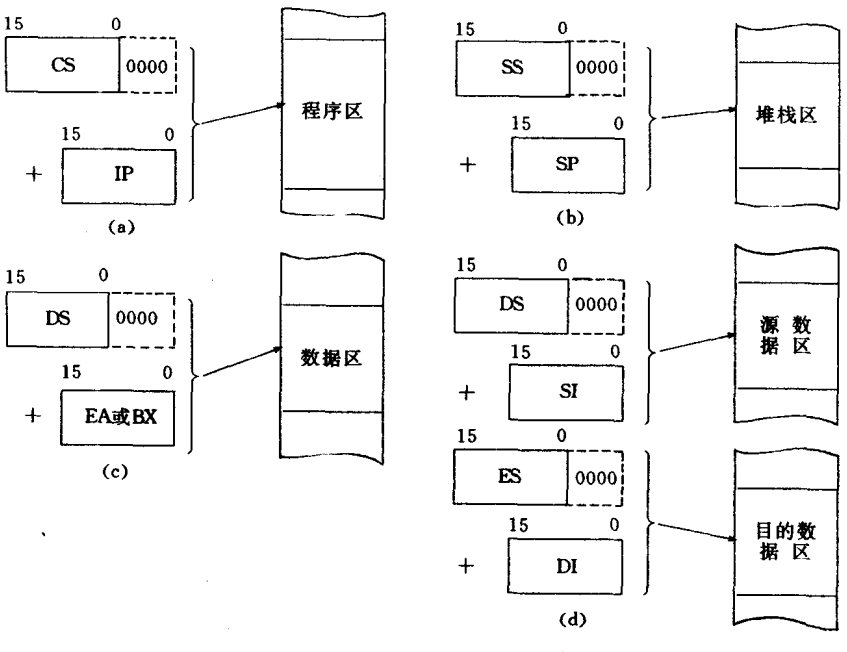


图1.9 对各区的访问示意图

第四节 指令的寻址方式

在 IBM-PC 微型计算机上,8086/8088 汇编语言指令按其书写格式主要有两部分,一部分是操作码部分,指出进行何种操作;另一部分是操作数地址或操作数部分,指出参与操作的对象或操作对象的所在地。有时操作数地址也称为地址码。操作码部分是使用助记符表示的,而操作数地址部分则是数字以及相应符号的组合形式,操作数为数字。

寻址方式就是在指令中提供操作数或操作数地址的方法。8086/8088 汇编指令中数据的寻址方式共有三大类,9 种寻址方式。

一、立即数寻址

在双操作数汇编语言指令中，开始部分是操作码的助记符，接着是一个空格，之后是逗号(,)分隔的两个操作数地址(或操作数)。逗号之前的操作数地址称为目的操作数地址，逗号之后的操作数地址(或操作数)称为源操作数地址或源操作数。如果源操作数是以常数形式给出时，则称为立即数寻址。指令中直接给出的常数可以是1字节8位二进制数(2位16进制数)，也可以是双字节的16位二进制数(4位16进制数)。值得注意的是立即数不能出现在目的操作数地址处。这种寻址方式的指令通常是给某个通用寄存器、内存储器的字节或字单元赋值。

例如：MOV CX,5000H 表示将常数5000H送入寄存器CX中。

再如：MOV CL,－50H 表示将补码表示的－50H送入寄存器CL中，即将B0H送入CL中。

二、寄存器寻址

如果汇编语言指令中的某个操作数地址是以寄存器助记符的形式给出的，此操作数地址的确定称为寄存器寻址。

例如：MOV CX,5000H 其目的操作数地址为寄存器CX，所以目的操作数的寻址方式为寄存器寻址。

再如：MOV AX,DX 源操作数地址和目的操作数地址的确定均为寄存器，因此，源和目的操作数的寻址方式均为寄存器寻址。

三、存储器寻址

为了更好地理解和描述存储器寻址，首先介绍一下有效地址的概念。所谓有效地址(EA)就是总线接口部件通过计算得到的相对某个段寄存器内容(通常是CS或DS)的偏移量。换言之，就是操作数所在段的首地址到该操作数地址之间的距离。有效地址是16位无符号数，其最大值为FFFFH，即十进制数65535，有效地址的范围是0～64K－1。

存储器寻址方式就是确定字节/字操作数所在的字节地址或字地址的方法。在汇编语言指令的书写中，字节/字操作数的地址是以某种组合形式给出的。前面讲过内存单元的物理地址是由段寄存器的内容左移四位和段内偏移量相加而确定的。一旦段寄存器的内容一定，只须确定段内偏移量即可，这个段内偏移量又是刚刚定义的有效地址EA。所以在某种意义上讲，存储器寻址方式就是确定有效地址EA的方式。根据组合方式的不同，存储器寻址方式共有7种类型。现分述如下：

1. 直接寻址

同立即数寻址相类似，直接寻址方式就是在指令中直接给出有效地址EA。其方式又有两种，一种是以4位16进制数的形式给出，但要将其括在方括号内；另一种是以一个变量标号的形式给出的。无论是哪一种形式，其机器码指令中表示EA的段内偏移量均为以2个字节形式存放在指令的操作码之后，EA的高8位存放在高地址单元中，低8位存放在低地址单元中。例如：

```
MOV AX,DS:[1000H]
MOV AX,TABLE
```

两条指令中的源操作数的寻址方式即为存储器寻址方式下的直接寻址，其物理地址的

确定均为数据段寄存器 DS 的内容左移 4 位再加上 EA 的值。第一条指令的 EA 等于 1000H，第二条指令的 EA 是标号变量 TABLE 所代表的地址单元相对于该段的首地址的位移量。

2. 基址寻址

指令中某个操作数有效地址 EA 存放在基础寄存器 BX 或者基址指针 BP 中时称为基址寻址。此时，书写汇编指令时，BX 或 BP 要用方括号括起来，以便区别于寄存器寻址。当使用[BX]时，其隐含的段寄存器为 DS；当使用[BP]时，其隐含的段寄存器为 SS。存储器物理地址的确定是由段寄存器的内容左移 4 位与 BX 或 BP 的内容相加之和。例如：

```
MOV AX,[BX]
MOV BX,[BP]
ADD [BX],DX
```

其中第一、二条指令的源单元，第三条指令的目的单元均为基址寻址。

3. 变址寻址

指令中某个操作数的有效地址 EA 是以括在方括号中的 SI 或 DI 两个变址寄存器之一的形式给出的寻址方式。内存储器物理地址的确定同基址寻址方式基本相同，两者的区别在于基址寻址中使用的是寄存器 BX 或 BP，而在变址寻址中使用的是变址寄存器 SI 或 DI。其隐含的段寄存器为 DS。

4. 相对变址寻址

这种寻址方式是变址寻址的扩展，反过来也可以说变址寻址为相对变址寻址的特殊形式。在相对变址寻址方式下，某个操作数有效地址值在指令中的表述除了含有带方括号的变址寄存器 SI 或者 DI 外，还有一个相对位移量，这个相对位移量可以是标号变量，16 位或 8 位数字。如果是 8 位数字，可以是正数，亦可为负数，均以补码形式存于存储单元中，若为负数，在进行地址计算时，首先要进行符号扩充，就是说在最高位(符号位)之前扩充 8 位数，每一位的值均和符号位相同，扩充后形成 16 位数。其操作数物理地址的确定是由段寄存器的内容乘以 16 后加上变址器的内容和相对位移量的值。例如：

```
MOV AX,VALUE[DI]
MOV AX,4000H[DI]
MOV AX,0FFH[DI]
```

三条指令的源单元均为相对变址寻址。

5. 相对基址寻址

相对基址寻址和相对变址寻址基本类似，所不同的只是变址器换成了基础寄存器 BX 或基址指针寄存器 BP。如果使用 BX，则隐含的段寄存器为 DS；若使用 BP，则隐含的段寄存器是 SS。在相对基址寻址方式下，汇编指令的书写格式有下列几种：

```
MOV AL,[BX]+4   ;标准格式
MOV AL,4[BX]    ;相对位移量在前
MOV AL,[BX+4]   ;相对位移量在括号中
```

以上 3 条指令尽管源操作数有效地址的书写格式不一样，但是都代表同一种寻址方式，而且所确定的有效地址是相同的。也就是 3 条指令的功能完全一样，它们是等价表达方式。