高等学校计算机教材

汇编语言程序设计

主　编　倪蕴涛
副主编　宫　兵　李峻灵

Assembler Language Programming

東北林業大學出版社

高等学校计算机教材

汇编语言程序设计

主　编　倪蕴涛
副主编　宫　兵　李峻灵

东北林业大学出版社

图书在版编目（CIP）数据

汇编语言程序设计/倪蕴涛主编．—哈尔滨：东北林业大学出版社，2007.4
ISBN 978-7-81131-007-8

Ⅰ．汇…　Ⅱ．倪…　Ⅲ．汇编语言—程序设计　Ⅳ．TP313

中国版本图书馆 CIP 数据核字（2007）第 067060 号

责任编辑：杨秋华
封面设计：彭　宇

NEFUP

汇编语言程序设计
Huibian Yuyan Chengxu Sheji
主　编　倪蕴涛
副主编　宫　兵　李峻灵

东北林业大学出版社出版发行
（哈尔滨市和兴路 26 号）
哈尔滨市工大节能印刷厂印装
开本787×1092　1/16　印张13.25　字数300千字
2007 年 4 月 第 1 版　2007 年 4 月 第 1 次 印 刷
印数 1—1 000 册
ISBN 978-7-81131-007-8
TP·78　定价：26.00 元

内 容 简 介

汇编语言既是高等学校计算机相关专业学生必修的专业课程之一，也是学习其他后续课程的基础和重要工具。因此，学习和掌握汇编语言知识，对于加深对计算机专业知识的理解、掌握和应用具有重要的作用。本书主要研究了 IBM-PC 及其兼容机汇编语言的基本知识和程序设计方法。本书共九章，内容包括微型计算机的基础知识和硬件结构、汇编语言的寻址方式和指令系统、宏汇编语言及其程序设计方法、输入/输出程序设计方法及文件的基本操作、汇编语言的上机及调试方法等。本书每章包含丰富的例题和习题。本书对知识的阐述简明扼要、内容完整并且语言精炼。本书计划教学时数为 60 学时。

本书可以作为高等学校计算机专业本科教材使用，也可以作为高等职业技术学院计算机相关专业的教材，以及从事计算机相关专业的技术人员的参考资料。

前　言

汇编语言程序设计是计算机及其相关专业的重要专业基础课程，是计算机研究与应用，特别是计算机控制领域必修的核心课程之一。深入学习和掌握汇编语言不仅对学习其他后续课程具有重要的作用，同时对理解整个计算机体系结构及内部工作过程具有很大的帮助。

本书以 8086 微处理器为主，重点介绍了指令系统及汇编语言程序设计的基本方法及汇编语言的高级程序设计方法，并初步介绍了 80386 微处理器的内部结构及程序设计方法。

由于汇编语言相对其他语言较难理解和掌握，因此本书通过大量的实例，由浅入深、循序渐进贯穿整个知识主线，以便于读者的学习和掌握。

全书共九章。第一章到第二章主要介绍了汇编语言的基本知识和 8086 微处理器的内部结构；第三章到第五章主要介绍汇编语言的寻址方式、指令系统及宏汇编语言的基本要求，为进行程序设计打下基础；第六章到第八章是汇编语言程序设计的核心部分，主要研究各种程序设计方法、DOS 功能调用及 BIOS 中断调用和计算机人机接口的程序设计。

本书可以作为高等学校计算机专业本科教材使用，也可以作为高等职业技术学院计算机相关专业的教材以及从事计算机相关专业的技术人员的参考资料。

本书由哈尔滨师范大学倪蕴涛、李峻灵和黑龙江大学宫兵共同编写，全书由倪蕴涛组织编写、审阅定稿。同时得到了其他院校老师的帮助，在此表示感谢。

由于编者水平有限，书中难免有不妥或错误之处，敬请计算机界同仁及广大读者批评指正。

编　　者

2006 年 3 月

目　录

第一章 基础知识

本章重点

- 计算机语言的发展及分类。
- 汇编语言的特点和应用领域。

语言是人与人之间进行交流的重要媒介。人与计算机之间同样也需要语言进行信息交流，汇编语言作为一种计算机语言，即充当着人与计算机之间进行信息交流的媒介的作用，而且汇编语言是唯一能够利用计算机的硬件资源、面向机器的计算机语言。

第一节 汇编语言

一、计算机语言的分类

按照计算机语言的发展历程，目前所使用的计算机语言可以分为3类：机器语言、汇编语言和高级语言。

20世纪80年代初，在软件设计思想上又发生了一次革命，其成果就是面向对象的程序设计。在此之前的高级语言，设计思想主要是面向过程的，即程序的执行是流水线式的，在一个模块被执行完成前，人们不能干别的事，也无法动态地改变程序的执行方向。因此从软件设计思想上计算机语言又可以分为面向任务和面向对象的语言。

机器语言：众所周知，计算机的所有动作都是在指令的控制下完成的。而能够直接控制计算机完成指定动作的即为机器指令。通常一条机器指令是由二进制数0和1组成的具有一定顺序的代码序列。不同顺序的二进制代码序列代表的机器指令也不同。一条机器指令的一般形式为：

操作码	地址码

操作码和地址码合在一起所组成的二进制序列构成一条完整的机器指令。其中，操作码部分指出该指令要做什么样的操作，即指出了运算的种类，如算术运算、逻辑运算和移位运算及地址运算等；而地址码部分则指出参与运算的源操作数和目的操作数的存放位置。

由于机器指令能够和计算机直接交换信息，因此机器指令也成为面向机器的语言。例如，完成两个数据100和256相加的功能，8086CPU的代码序列如下：

10111000011001000000000

00000101000000000000001

10100011000000000100000

对应的十六进制形式表达为：**B86400050001A30020H**。

汇编语言：汇编语言为较早发明的一种计算机程序设计语言，在计算机程序设计领域，特别是在与计算机硬件操作相关的领域得到了广泛的应用。

因为机器指令全部由二进制代码组成，因而编写程序过程相当繁琐，而且给程序阅读和程序调试带来很多麻烦，不易于理解和记忆。为了解决机器语言的这些缺点，人们发明了助记符语言——汇编语言。即用代表一定意义的符号来表示机器指令的操作码和操作数，并且遵循一定的语法规则编写的计算机语言成为汇编语言。由于汇编语言比机器语言更加容易记忆和被人们接受，如减法指令用 **SUB**、数据传送用 **MOV** 等，因此汇编语言也更接近于机器语言。

用汇编语言助记符编写的具有一定功能的汇编语言指令集合称为汇编语言源程序。通常源程序是不能被计算机理解并直接执行的，必须通过翻译程序将汇编语言源程序翻译成机器指令序列之后才能够执行，这种将汇编语言源程序翻译成目标程序的过程称为汇编。汇编语言源程序通过翻译转换之后形成的程序称为目标程序，该目标程序虽已具备机器语言的形式，但还不能执行，必须通过链接程序将目标程序转换为可执行文件之后才能执行，具体过程如图 **1 - 1** 所示。

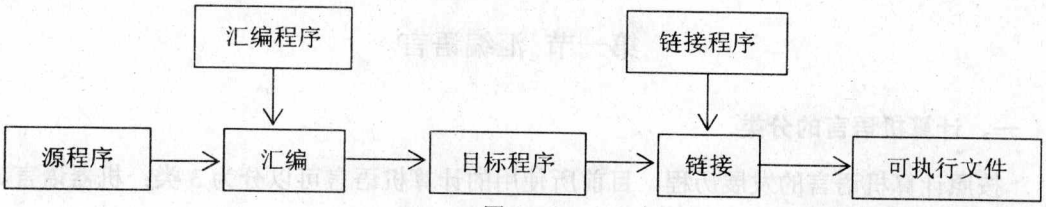

图 1 - 1

高级语言：该语言是更接近于人类自然语言的计算机语言，因此称为高级语言，如 **C** 语言、**Fortran** 语言等。相对人的思维来讲，高级语言更加容易被人接受和掌握。但并不是说高级语言最好，每一种语言都有各自的应用领域和优势。

二、汇编语言的特点

相对机器语言来讲，汇编语言具有可读性强、便于记忆和理解等优点。由于汇编语言使用了助记符语言，因此汇编语言源程序易于理解，具有较强的目的性，几乎每一条指令都具有完整的意义和操作目的。例如指令：**MOV AX，10** 即是将立即数 **10** 传送给累加器 **AX**。

相对其他高级计算机语言来讲，汇编语言具有控制能力强、程序代码段、执行速度快等优点。当今所使用的高级语言和面向对象的计算机语言在数据运算和事件处理相对汇编语言具有较强的优势，然而用高级语言编写的程序在翻译成机器语言后对计算机硬件要求很高，程序在执行过程当中要求存储空间过大，代码冗长，执行速度慢。特别是在相关接口设备驱动以及与输入输出设备进行通信时，汇编语言以代码精简和高效率体现出高级语言不可比拟的优势。因此在设备驱动领域和通信领域经常使用高级语言和汇编语言交叉编译，充分发挥各自的优点。

三、汇编语言的应用

如上所述，汇编语言经常用于编写设备驱动程序和一些适时控制领域，比如操作系统的内核的编写、声卡和显卡的驱动程序等。由于汇编语言是计算机能够提供给用户的最快和最有效的语言，也是能够利用计算机所有硬件特性并且能够直接控制硬件的唯一

语言，因而在对于程序的空间和时间要求很高的场合，汇编语言是必不可少的。至于很多需要直接控制硬件的应用场合，则汇编语言更是体现出它的优势。

第二节 一个完整的汇编语言程序

【例1-1】利用汇编语言完成在显示器上输出字符串“Harbin Normal University”。程序清单如下：

```
STACK   SEGMENT   STACK
        DB   200  DUP（0）
STACK   ENDS
DATA    SEGMENT
STR     DB  'Harbin Normal University $'
DATA    ENDS
CODE    SEGMENT
        ASSUME CS：CODE，DS：DATA，SS：STACK
BEGIN：
        MOV  AX，DATA
        MOV  DS，AX
        LEA  DX，STR
        MOV  AH，9
        INT  21H
EXIT：  MOV  AH，4CH
        INT 21H
CODE    ENDS
        END  BEGIN
```

习题一

1．什么是汇编语言？

2．解释什么是汇编语言源程序、目标程序？如何将汇编语言源程序制作成为可执行文件？

3．同高级语言及机器语言相比较，汇编语言存在哪些优点和缺点？

第二章 微型计算机的内部结构

本章重点

- 8086CPU 的内部结构及各寄存器的基本功能。
- 存储器的存储机理。
- 存储器物理地址的形成及段的划分。
- 数据的机内表示方法。

由于汇编语言是一种接近机器语言的计算机低级语言，因此学习相关的计算机硬件知识，特别是计算机 CPU 的内部结构以及各部件之间的关系，对深入学习和熟练运用汇编语言具有积极的作用。

微型计算机主要包括微处理器（CPU）、主存储器和输入输出接口以及系统总线组成，系统结构如图 2 - 1 所示。

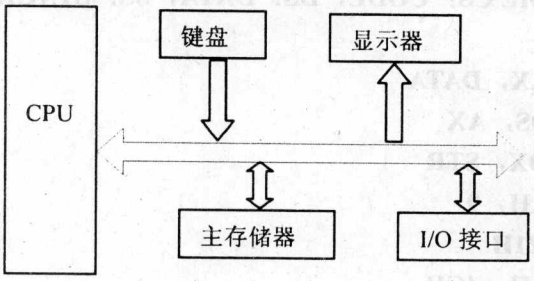

图 2 - 1

CPU 是整个计算机系统的核心，主要用于算术和逻辑运算以及实现对整个计算机系统的控制。主存储器作为高速的数据存储部件主要用来保存程序和数据。输入输出接口用来完成 CPU 与存储器或者外部设备之间的数据交换。系统总线主要在各个部件之间作为提供通信的媒介。

第一节 8086 微处理器的内部结构

INTEL8086 是 INTEL 公司推出的 16 位微处理器，按功能可分为两部分：执行部件（EU）和总线接口部件（BIU），具体结构如图 2 - 2 所示。

一、执行部件

执行部分 EU 负责指令的执行，并进行算术逻辑运算等。EU 从 BIU 中的指令队列中取得指令。它由一个算术逻辑单元(ALU)、一组通用寄存器和标志寄存器组成，它们均是 16 位的寄存器。执行部件中含有 8 个 16 位的寄存器，这些寄存器属于 CPU 的专用寄存器，按其用途可将它们分成两组：数据寄存器组和指示器变址寄存器组。

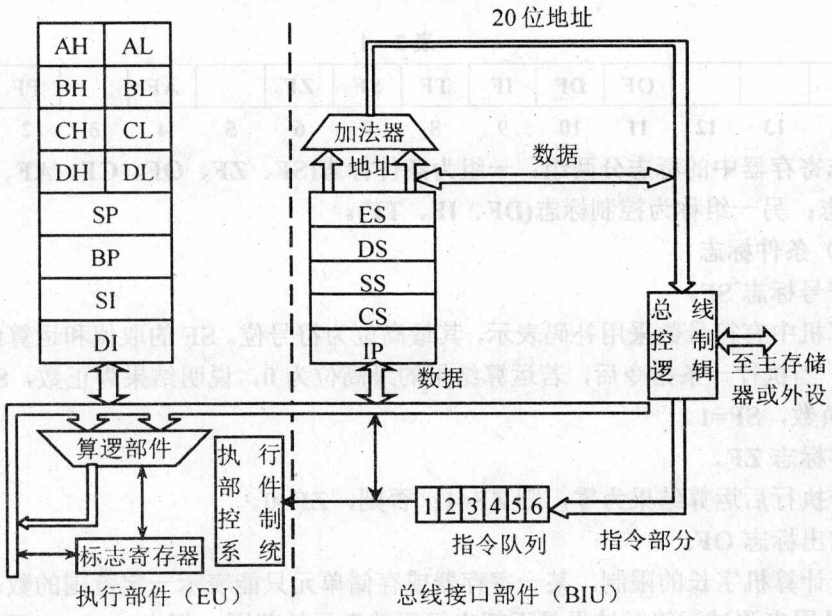

图 2－2

1．数据寄存器组(AX、BX、CX、DX)

数据寄存器主要用来保存操作数或运算结果等信息。它们的存在减少了为存取操作数所需访问总线和主存的时间，加快了机器的运行速度，其中 AX 称为累加器，BX 称为基址寄存器，CX 称为计数寄存器，DX 称为数据寄存器。它们既可以作为 16 位寄存器使用，又可按高 8 位和低 8 位作为 8 位寄存器使用，因此，又将这 4 个寄存器分为两组：High 组(AH、BH、CH、DH)和 Low 组(AL、BL、CL、DL)。除此之外，这些寄存器还有其他的用途。

2．指示器变址寄存器组(SI、DI、SP、BP)

这 4 个寄存器均为 16 位寄存器，它们一般用来存放操作数的偏移地址，作为指示器或变址寄存器。

SP 称为堆栈指示器（堆栈指针），用来指出当前堆栈段中栈顶的偏移地址。

BP 称为对堆栈操作的基址寄存器。用来堆栈段中某一存储单元的偏移地址。

SI 和 DI 除了作为一般指示器和变址寄存器外，在串操作指令中，SI 往往被规定用来作为传送源操作数的地址指示器，而 DI 被规定用来作为传送目的操作数的地址指示器，因此，SI 称为源变址寄存器，DI 称为目的变址寄存器。

当然，指示器变址寄存器也可以作为数据寄存器使用。

3．标志寄存器

标志寄存器(Flags)用来保存在一条指令执行之后，CPU 所处状态的信息及运算结果的特征或反映 CPU 的控制信息等。8086 的标志寄存器为一个 16 位的寄存器，其中设置了 9 个标志如表 2－1 所示。

表 2-1

				OF	DF	IF	TF	SF	ZF		AF		PF		CF
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

标志寄存器中的标志分两组：一组为条件标志(SF、ZF、OF、CF、AF、PF)，也叫状态标志；另一组称为控制标志(DF、IF、TF)。

（1）条件标志

①符号标志 SF。

计算机中有符号数采用补码表示，其最高位为符号位。SF 的取值和运算结果的最高位相同。当执行一条指令后，若运算结果的最高位为 0，说明结果为正数，SF=0；否则结果为负数，SF=1。

②零标志 ZF。

指令执行后运算结果为零，则 ZF=1；否则，ZF=0。

③溢出标志 OF。

由于计算机字长的限制，某一寄存器或存储单元只能表示一定范围的数。因此，溢出标志是用来测试运算时结果是否超出机器数表示的范围。超出 OF=1；否则 OF=0。对于溢出的判断可以根据符号位的值来实现，比如两个正数的和结果却为负值，或者两个负数相加运算所得的结果为正数，都说明运算的结果超出了机器字长的表示范围，因此都产生了溢出。

④进位标志 CF。

运算时运算结果的最高位是否产生进位或借位，有则 CF=1；否则 CF=0。当执行一个加法运算时，若最高位向上产生进位，或当执行一个减法运算时，最高位向上产生借位时，CF=1；否则 CF=0。

⑤辅助进位标志 AF。

进行加法或减法运算时，第 3 位向第 4 位产生进位(加法)或借位(减法)，则 AF=1；否则 AF=0。辅助进位标志主要用于 BCD 码的加减法运算中作为是否需要进行十进制调整的条件依据。

⑥奇偶标志 PF。

当运算结果中(低 8 位)中“1”的个数为偶数位时，PF=1；否则 PF=0。该位主要用于数据传输过程中检测可能出现的数据传输错误。

（2）控制标志

①方向标志 DF。

方向标志主要用于串操作，当 DF=0 时，串操作指令按照由低地址向高地址进行正向处理；当 DF=1 时，串操作指令按照由高地址向低地址进行反向处理。该标志位可以通过指令 STD 指令将 DF 置 1，也可以通过 CLD 指令将 DF 置 0。

②中断允许标志 IF。

当 IF=1 时，CPU 开中断，即 CPU 相应外设的中断请求；当 IF=0 时，CPU 关中断，即 CPU 不响应外设的中断请求。该标志位可以通过指令 STI 指令将 IF 置 1，也可以通过 CLI 指令将 IF 置 0。

③跟踪标志 TF。

当 TF=1 时，CPU 处于单步工作方式，即每执行完一条指令后，CPU 自动产生类型为 1 的中断，使程序单步执行；当 TF=0 时，则 CPU 处于连续工作方式。该标志主要用于程序的调试过程中。

二、标志寄存器操作指令

8086 微处理器专门为标志寄存器设置了以下几条指令。

1．标志寄存器入栈指令

格式：PUSHF

功能：将标志寄存器的内容压入堆栈。此指令通常可以作为间接读取标志寄存器的内容。

例如：假设（FLAGS）=0B85H，执行指令 PUSHF。

执行后，系统自动将 0B85H 推入堆栈段保存起来。

2．标志寄存器出栈指令

格式：POPF

功能：将堆栈段中当前栈顶的一个字的内容弹出到标志寄存器当中，此指令可以间接向标志寄存器赋值。

【例 2－1】

```
MOV  AX，0B58H；将立即数 0B58H 传送给累加器 AX
PUSH  AX；将（AX）的内容推入堆栈
POPF；将堆栈段中当前栈顶的内容弹出到表示寄存器
```

三、总线接口部件(BIU)

总线接口部件 BIU 包括一组 16 位的段寄存器（CS、DS、ES、SS）、一个指令指示器（IP）、指令队列(8086 长 6 个字节，8088 长 4 个字节)、地址加法器和总线控制器等。BIU 根据执行部件 EU 的请求，完成 CPU 与存储器或 I/O 设备之间的数据传送。在 EU 执行指令的过程中，BIU 根据需要从存储器中预先取一些指令，保存到指令队列中。如果 EU 执行一条转移指令，使程序发生转移，那么存放在指令队列中的预先取得的指令就不再有用，BIU 会根据 EU 的指示从新的地址重新开始取指令。

由于 EU 所提供的存储器地址是 16 位的，而 8086 访问 1M(220)存储空间却需要 20 位地址，为了形成这 20 位地址，在 BIU 中设立了 4 个段寄存器(CS 代码段，DS 数据段，ES 附加段，SS 堆栈段)。CPU 在当前任意时刻可以直接访问 4 个存储段：一个当前代码段、一个当前数据段、一个当前附加段、一个当前堆栈段，每个段最大可达 64K 字节。

指令指示器(IP)，它总是存放着下一次将从主存中取出指令的偏移地址(简称 EA)，其值为该指令到所在段段首址的字节距离。IP 的内容有总线接口部件自动修改，当执行顺序指令时，IP 中存放的是下一条即将取出的指令的偏移地址；当发生中断或者程序调用时，总线接口部件会自动保存当前的 IP 地址值，同时将新的转移地址送入 IP 中，

实现程序的自动跳转。

地址加法器是 8086 微处理器中专用的特殊加法运算部件，根据执行部件提供的 16 位偏移地址和当前段寄存器中提供的 16 位段地址作为两个操作数，通过地址加法器进行移位加法运算，计算出 CPU 进行存取操作所需要的 20 位物理地址。

第二节 主存储器

一、主存储器

计算机的存储系统由主存储器(也称为内存储器)和辅助存储器(也称为外存储器)组成，存储器是用来存放程序和数据的物理部件。

主存储器是计算机的重要组成部分，用户的程序和数据都存放在存储器当中。主存储器(简称主存)的基本存储单元是位，它能容纳一个二进制的 0 和 1。整个主存由许多存储位构成，这些存储位每 8 位组合成一个字节（Byte），每相邻的 2 个字节组成一个字(Word)。为了区别这些不同的字节(或字)存储单元，每一单元都被指定一个编号，称为此单元的物理地址(简称 PA)。PC 机的主存是按字节编址的，即以字节作为最小单位。假定主存容量为 1M 字节，则它的最低地址为 00000H，最高地址为 0FFFFFH。

主存储器的读取规则："高高低低"规则，即高地址对应高字节，低地址对应低字节。而在字类型数据中，由于字是由 2 个相邻的字节构成，因此规定字的地址为 2 个字节中地址相对较小的那一个表示。

【例 2－2】图 2－3 为主存储器部分存储单元的存储状态，试从中读取数据。

如图所示，字节单元(00000)的内容为 12H，字单元(00000)的内容为(00001，00000)=9812H，字单元(00001)的内容为(00002，00001)=0AC98H。

注意：如果十六进制数的第一个数字为字符 A～F，需在此十六进制数前加一数字 0，以便与其他同名的变量名及寄存器名相区别。

00000	12
00001	98
00002	AC
00003	47
00004	09
00005	78
00006	23
00007	9A

图 2－3

二、段的划分和段的重叠

8086CPU 有 20 根地址线，最大寻址 2^{20}=1MB。而 8086CPU 的字长为 16 位，直接寻址 2^{16}=64KB，无法寻址 1MB。为此，8086 采用了存储器地址分段的方法，实现对整个 1MB 物理存储空间的寻址操作。

将整个存储器分成许多逻辑段，每个逻辑段的容量最多为 64KB，允许它们在整个存储器空间定义。各个逻辑段可以紧密相连，也可以有重叠。对于任何一个物理地址来说，可以唯一地被包含在 1 个逻辑段中，也可以被包含在多个相互重叠的逻辑段中，只要能得到它所在段的首地址和段内的相对地址，就可以对它进行访问。

因为 8086CPU 的段寄存器只有 4 个，因此某一时刻，当前段只能有 4 个，即当前数据段、当前代码段、当前堆栈段和当前扩展数据段。但绝不意味着 1MB 的存储空间只能定义 4 个段。若一个段按照最大的定义空间 64KB 来计算，1MB 的存储空间至少可以定义 16 个 64KB 大小的逻辑段。如图 2－4 所示。

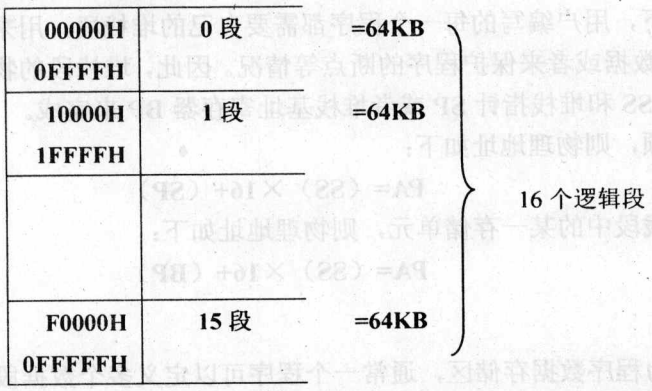

图 2－4

三、存储器物理地址的形成

物理地址（PA）是由段地址与偏移地址共同决定的。段地址来自于段寄存器（CS、DS、ES、SS），是 16 位的地址；偏移地址由（IP、SP、SI、DI）等寄存器提供。可以用逻辑地址将物理地址表示为物理地址=段地址：偏移地址的形式。由 16 位的段地址及 16 位的偏移地址计算物理地址的表达式如下：

物理地址=段地址×16+偏移地址

相应的功能通过 CPU 提供的段地址和偏移地址，由总线接口部件完成，对应的硬件结构如图 2－5 所示。

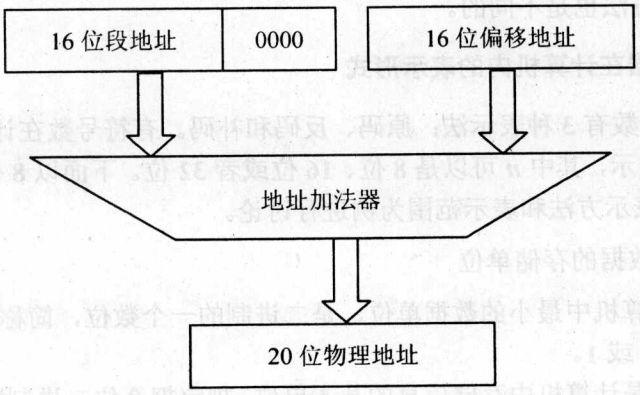

图 2－5

1．代码段

代码段作为程序代码存储区，它的物理地址是由当前代码段寄存器 CS 和指令指示器 IP 共同按照以上公式实现的：

PA=（CS）×16+（IP）

【例 2－3】假设当前代码段寄存器（CS）=1000H，指令指示器（IP）=2000H，则当前代码段的物理地址起始处为 1000H×16+2000H=12000H。

2．堆栈段

通常情况下，用户编写的每一个程序都需要自己的堆栈段，用来存放程序运行过程中产生的中间数据或者来保护程序的断点等情况。因此，堆栈段的物理地址可以由当前堆栈段寄存器 SS 和堆栈指针 SP 或者堆栈基址寄存器 BP 来完成。

若访问栈顶，则物理地址如下：

PA=（SS）×16+（SP）

若访问堆栈段中的某一存储单元，则物理地址如下：

PA=（SS）×16+（BP）

3．数据段

数据段作为程序数据存储区，通常一个程序可以定义多个数据段以及扩展数据段。而当前数据段的物理地址由数据段寄存器DS和某一存储单元的偏移地址EA共同完成，具体形式如下：

PA=（DS）×16+（EA）

而当前扩展数据段的物理地址由数据段寄存器 ES 和某一存储单元的偏移地址 EA 共同完成，具体形式如下：

PA=（ES）×16+（EA）

第三节 数据在计算机内的表示形式

计算机中的数据通常分为数值型数据和非数值型数据。因此，不同类型的数据在计算机内部的表示方法也是不同的。

一、数值型数据在计算机内的表示形式

计算机内定点数有 3 种表示法：原码、反码和补码。有符号数在计算机中一律采用 *n* 位二进制补码表示，其中 *n* 可以是 8 位、16 位或者 32 位。下面以 8 位和 16 位有符号数在计算机中的表示方法和表示范围为例进行讨论。

1．计算机中数据的存储单位

位（Bit）：计算机中最小的数据单位，是二进制的一个数位，简称位（比特），1 位二进制数取值为 0 或 1。

字节（Byte）：是计算机中存储信息的基本单位，规定把 8 位二进制数称为 1 个字节，单位是 B（1B＝8Bit），常用的存储信息时容量与字节有关的单位换算如下：