21世纪 高等学校本科系列教材

总主编 吴中福

# 汇编语言程序设计

(8)

雷金辉 主编

杨志翔 李心一 副主编

重庆大学出版社

# 汇编语言程序设计

雷金辉　主　编
杨志翔　李心一　副主编

重庆大学出版社

## 内 容 提 要

本书为计算机科学与技术专业本科系列教材之一，主要阐述了 8086/8088，80X86 汇编语言程序设计的方法与技术。全书共分 10 章：第 1、2 章主要以汇编语言所需要的数值表示及类型、CPU 结构等基础知识入手，介绍汇编语言的有关概念、程序结构及上机过程，讲解了汇编语言寻址方式；第 3、4 章主要讲解汇编语言的指令系统和伪操作指令，说明了汇编语言程序格式；第 5 章叙述分支、循环、子程序等基本程序结构及其程序设计技术；第 6 章讲述高级宏汇编技术；第 7 章简要叙述 I/O 程序设计和中断技术，包括 BIOS 和 DOS 系统功能调用的使用方法；第 8 章实例介绍简单和复杂应用程序设计，第 9 章是包括汇编语言和高级语言程序模块相连接在内的通讯技术；第 10 章简要叙述 80386 和 Pentium 程序设计。同时将上机实验内容有意识地融入各章内，很好地把理论、实践及练习联系在一起。

本书通俗易懂，其理论与实践性较强，适宜作高等院校教材，也适用于初学者学习和掌握汇编语言程序设计技术。另外，还可供使用汇编语言技术的工程技术人员参考。

**图书在版编目(CIP)数据**

汇编语言程序设计/雷金辉主编．—重庆：重庆大学出版社．2001.9

计算机科学与技术专业本科系列教材

ISBN 7-5624-2350-4

Ⅰ.汇… Ⅱ.雷… Ⅲ.汇编语言—程序设计—高等学校—教材 Ⅳ.TP313

中国版本图书馆 CIP 数据核字(2001)第 064144 号

**汇编语言程序设计**

雷金辉 主 编

杨志翔 李心一 副主编

责任编辑 彭 宁

*

重庆大学出版社出版发行

新 华 书 店 经 销

重庆大学建大印刷厂印刷

*

开本：787×1092 1/16 印张：19.25 字数：480 千字

2001 年 9 月第 1 版 2001 年 9 月第 1 次印刷

印数：1—5 000

ISBN7-5624-2350-4/TP·306 定价：25.00 元

# 前言

汇编语言是一门低级语言。它能充分利用计算机的硬件资源且能进行有效的直接控制，同时利用它来深刻理解和掌握计算机深层结构和许多专业应用知识，即使在计算机业高速发展的今天，也起着独特的、甚至是不可替代的作用。

“汇编语言程序设计”是计算机专业一门重要的基础课程，是必修的核心课程之一，是“操作系统”和“微机原理与接口技术”等其他核心课程的必要先修课。该课程主要采用目前国内广泛使用的IBM PC为基本机型，让学生从基础知识入手，通过循序渐进的学习系统理论、上机实践、习题练习三位一体相结合，达到让学生理解与掌握程序设计技术，熟悉上机操作和程序调试；通过程序设计技巧与方法讲解及80386、Pentium程序设计介绍，培养学生综合地分析问题、解决问题的能力，在掌握知识的深度与广度上有更进一步的提高。

全书共分10章：第1、2章主要以汇编语言所需要的数值表示及类型、CPU结构等基础知识入手，介绍汇编语言的有关概念、程序结构及上机过程，讲解了汇编语言寻址方式；第3、4章主要讲解汇编语言的指令系统、伪操作指令，说明了汇编语言程序格式；第5章叙述分支、循环、子程序等基本程序结构及其程序设计技术；第6章讲述高级宏汇编技术；第7章简要叙述I/O程序设计和中断技术，包括BIOS和DOS系统功能调用的使用方法；第8章实例介绍简单和复杂应用程序设计，第9章是包括汇编语言和高级语言程序模块相连接在内的通讯技术；第10章简要叙述80386和Pentium程序设计。同时将上机实验内容有意识地融入各章内，以便很好地把理论、实践及练习联系在一起。

本书通俗易懂，其理论与实践性较强，书中例题内容均通过上机实际运行正确。本书适宜作高等院校教材，也适用于初学者学习和掌握汇编语言程序设计技术。另外，也可供使用汇编语言技术的工程技术人员参考。

本课程的参考学时数为60。

本书第6、9章由杨志翔编写，第3、4章由王崇国编写，第2、5章

由杨颖编写，第1、8章由李心一编写，第7、10章及附录等内容由雷金辉编写，全书由雷金辉统稿。在编写和审定中，许多同仁提出了很多的宝贵意见和建议，在此一并表示感谢。

限于编者的学识水平，书中如有错误或不妥之处，敬请广大同行及读者指正。

编　者

2001年6月

# 目录

# 第 1 章
# 基础知识

本章主要介绍在学习汇编语言之前所需的基础知识，通过对数据的表示与类型、微机的组成、CPU 的结构等内容的学习，使我们对计算机的组成和内部结构、计算机所采用的数制及数与字符的表示有一个大致的了解。

## 1.1 数据的表示与类型

### 1.1.1 数制系统

使用微处理器需要了解二进制，八进制，十进制和十六进制数的一些知识，若对此不熟悉，本节将提供一些背景知识，并介绍进位计数制之间的转换。

**(1)数制**

我们知道，一个十进制数有两个主要特点：一是有 10 个不同的数码，即 0～9 十个数字；二是逢“十”进位，即基数为 10。因此，对于一个数，同一个数字在不同的位置代表的数值是不同的。如，在 999.99 中小数点左面的第一位的 9 代表个位，就是它本身的数值 9；小数点左面的第二位的 9 代表十位，它的值为 $9\times10^1$；左面的第三位的 9 代表百位，它的值为 $9\times10^2$；而小数点右面的第一位它的值就为 $9\times10^{-1}$；右面的第二位它的值就为 $9\times10^{-2}$。所以，这个数可以写成：

$$999.99=9\times10^2+9\times10^1+9\times10^0+9\times10^{-1}+9\times10^{-2}$$

一般地说，任意一个十进制数 A，都可表示为：

$$A=A_{n-1}\times10^{n-1}+A_{n-2}\times10^{n-2}+\cdots+A_1\times10^1+A_0\times10^0+$$

$$A_{-1}\times10^{-1}+\cdots+A_{-m}\times10^{-m}=\sum_{i=n-1}^{-m}A_i\times10^i$$

其中 i 表示数的某一位；$A_i$ 表示第 i 位的数码，它可以是 0～9 中的任一个，由具体的数 A 确定；m 和 n 为正整数，n 为小数点左面的位数，m 为小数点右面的位数；10 为基数。

对于二进制数，与十进制数类似，也有两个主要特点：一是有 2 个不同的数码，即 0 和 1；二是逢“2”进位，即基数为 2。对于一个数，不同的数码在不同的数位所代表的值也是不同的。如 11011.101：

$$(11011.101)=1\times2^4+1\times2^3+1\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-3}=(27.625)_{10}$$

一般地说，任意一个二进制数 B，都可表示为：

$$B = B_{n-1} \times 2^{n-1} + B_{n-2} \times 2^{n-2} + \cdots + B_1 \times 2^1 + B_0 \times 2^0 +$$

$$B_{-1} \times 2^{-1} + \cdots + B_{-m} \times 2^{-m} = \sum_{i=n-1}^{-m} B_i \times 2^i$$

其中 $B_i$ 只能取 1 或 0，由具体的数 B 确定；m 和 n 为正整数，n 为小数点左面的位数，m 为小数点右面的位数；2 为基数。

类似地，八进制数，其基数为 8，包含 8 个数字：0～7；十六进制数，其基数为 16，由 16 个数字组成：0～9，A～F。

由此可推：N 进制数也具有这两个特点，即：一是有 N 个不同的数码，即 N 个数字组成；二是逢“N”进位，即基数为 N。因此，它的每一个数位 i，对应一个固定的值 $N^i$，$N^i$ 就称为该位的“权”，小数点左面各位的权依次是基数 N 的正次幂；而小数点右面各位的权依次是基数 N 的负次幂。一般地，N 进制数 Y 可表示为：

$$Y = \pm \sum_{i=n-1}^{-m} Y_i \times N^i$$

其中 $Y_i$ 是 i 位的数码，由具体的数 Y 确定；m 和 n 为正整数，n 为小数点左面的位数，m 为小数点右面的位数；N 为基数；$N^i$ 为权。

**例** 1.1.1　假设要将八进制数 $125.7_8$ 转换成十进制，据上述公式展开可得：

$(125.7)_8 = 1 \times 8^2 + 2 \times 8^1 + 5 \times 8^0 + 7 \times 8^{-1} = (85.875)_{10}$

每一种数制都有相应的运算规则，这与数制的定义和特点是分不开的。如二进制数，其基本运算规则是：

$$\begin{pmatrix} 0+0=0 \\ 0+1=1 \\ 1+0=1 \\ 1+1=10 \\ 1+1+1=11 \end{pmatrix} \qquad \begin{pmatrix} 0-0=0 \\ 0-1=1(\text{借位}) \\ 1-0=1 \\ 1-1=0 \\ 0-1-1=0(\text{借位}) \end{pmatrix}$$

计算机本身只认识二进制数，而在计算机上我们常用十进制、二进制、八进制和十六进制。下面学习数的转换及其他方面。

**(2)N 进制转换成十进制**

根据上述 N 进制的定义与特点，可得 N 进制转换成十进制的方法：首先将 N 进制数 Y 按权展开，然后利用十进制数的运算规则进行运算，所得结果即为十进制数。这种方法也称位权展开法。

**例** 1.1.2　将十六进制数 3AB.11 转换成十进制，按上述公式展开可得：

$(3AB.11)_{16} = 3 \times 16^2 + A \times 16^1 + B \times 16^0 + 1 \times 16^{-1} + 1 \times 16^{-2} =$　采用十进制数运算规则

$3 \times 16^2 + 10 \times 16^1 + 11 \times 16^0 + 1 \times 16^{-1} + 1 \times 16^{-2} =$　运算，将 A 转换为 10，

$(939.0664)_{10}$　B 转换为 11

**例** 1.1.3　将二进制数 10101.11 转换成十进制，按上述公式展开可得：

$(10101.11)_2 = 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = (21.75)_{10}$

(3)十进制数转换成 N 进制数

由十进制数转换成 N 进制数要比 N 进制数转换成十进制数困难。需要分成整数部分和小数部分两个部分来转换:整数部分采用除基数来转换,而小数部分则采用乘基数来转换。

1)十进制整数部分的转换

把十进制整数转换成 N 进制数时,要用基数 N 来除,每次都要保留余数。转换过程算法如下:

a. 用基数 N 来除十进制整数部分。

b. 保留余数(最先得到的余数是最低位数字)。

c. 重复步骤(a)和(b)直到商为 0。

**例** 1.1.4　要把一个十进制数 10 转换成二进制数,要用 2 做除法。商是 5,余数为 0。首先得到的余数放在个位上,在本例中为 0。接着用 5 除以 2。所得商是 2,余数是 1。这个 1 是权为 $2^1$ 的那位对应的值。继续做除法,直到商为 0 为止。从底至上结果写作$(1010)_2$,转换过程是:

2) 10　余数 = 0
2) 5　余数 = 1
2) 2　余数 = 0
2) 1　余数 = 1
0

**例** 1.1.5　给出由十进制数 10 转换成八进制数,用基数 8 来除的过程。结果是八进制的 12。转换过程是:

8) 10　余数 = 2
8) 1　余数 = 1
0

从十进制数到十六进制数是用除 16 来完成。余数的范围是 0 ~ 15。从 10 到 15 的余数转换成十六进制的字母 A 到 F。

**例** 1.1.6　十进制数 109 转换成十六进制,结果为 6D。

16) 109　余数 = 13(D)
16) 6　余数 = 6
0

2)十进制小数部分的转换

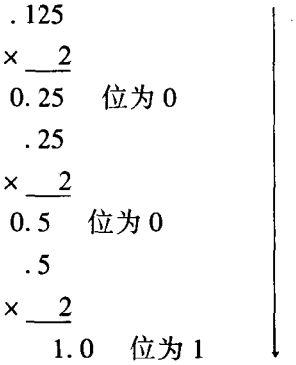
转换小数部分需要用转换的目标数制的基数 N 做乘法得到。例如,要把一个十进制小数转换成二进制小数,要用 2 来乘。乘法完成后,结果的整数部分保留起来作为结果的一位,小数部分继续用基数 2 来乘。当小数部分为 0 时,整个过程也就结束了。注意有些数永远乘不尽,即 0 永远不会成为余数。十进制小数部分转换算法如下:

a. 用基数 N 来乘十进制小数部分。

b. 保留结果的整数部分(0 也包括在内)作为结果的一位。注意第一个得到的数字紧跟着写在基数小数点后面。

c. 用步骤(b)的小数部分反复使用步骤(a)和(b),直到步骤(b)的小数部分为 0。

**例** 1.1.7　假设十进制 .125 要转换成二进制，结果是二进制小数 0.001。




小数部分为 0

同样的方法也适用于把十进制数转换成 N 进制数。

**例** 1.1.8　把例 1.1.7 中的十进制数 .125 转换成八进制数，结果为 0.1。

.125
× 8
1.0　位为　1

小数部分为 0

**例** 1.1.9　将十进制小数 .046875 转换成十六进制数，结果为 0.0C。

.046875
× 16
0.75　位为 0
.75
× 16
12.0　位为 12，转换为数码 C

小数部分为 0

**(4)二进制、十六进制、八进制数之间的相互转换**

表 1.1 给出了二进制数、八进制数及十六进制数的编码。由表可以看出，二进制数、八进制数及十六进制数之间有一个明显的关系，即：一位十六进制数可由 4 位二进制数惟一表示；而一位八进制数可由 3 位二进制数惟一表示。

因此，二进制数、八进制数及十六进制数之间可以相互表示和随意转换。其转换方法如下：

二进制数⇔十六进制数：对二进制数，首先以小数点为基准，向左右两边划分，每四位一组，最左和最右边不够四位的补零；然后把每一组用相应的一位十六进制数代替，即得十六进制数。对十六进制数，首先将每一位十六进制数用相应的四位二进制数代替；然后将最左边多余的零去掉(注意有效数字之间的零不能去掉)，即得二进制数。

二进制数⇔八进制数：对二进制数，首先以小数点为基准，向左右两边划分，每三位一组，最左和最右边不够三位的补零；然后把每一组用相应的一位八进制数代替，即得八进制数。对八进制数，首先将每一位八进制数用相应的三位二进制数代替；然后将最左边多余的零去掉(注意有效数字之间的零不能去掉)，即得二进制数。

十六进制数⇔八进制数：通过十六进制数⇔二进制数⇔八进制数这样的方式来转换。

**例 1.1.10**　十六进制数 2AC 转换成二进制数，再转换为八进制数。

$(2AC)_{16}$ = 0010 1010 1100 =

$(1010101100)_2$ =

0010 1010 1100 =

$(1254)_8$

**例 1.1.11**　二进制数 1000111101.111 转换成十六进制数。

$(1000111101.111)_2 = 0010\ 0011\ 1101.1110 = (23d.e)_{16}$

**表 1.1　二进制编码的十六进制(BCH)码**

| 二进制 | 八进制 | 十六进制 | 二进制 | 八进制 | 十六进制 |
|---|---|---|---|---|---|
| 0000 | 0 | 0 | 1001 | 11 | 9 |
| 0001 | 1 | 1 | 1010 | 12 | A |
| 0010 | 2 | 2 | 1011 | 13 | B |
| 0011 | 3 | 3 | 1100 | 14 | C |
| 0100 | 4 | 4 | 1101 | 15 | D |
| 0101 | 5 | 5 | 1110 | 16 | E |
| 0110 | 6 | 6 | 1111 | 17 | F |
| 0111 | 7 | 7 | 0001 0000 | 20 | 10 |
| 1000 | 10 | 8 | 0001 0001 | 21 | 11 |

**(5)原码、补码和反码**

原码由符号与真值来表示。符号位于数据的最高位，正号用 0 表示，负号用 1 表示。真值用数的二进制表示。

例如：$(-12)_{10}$，$(12)_{10}$ 分别为 1 1100(1 代表符号为负，1100 代表真值)与 0 1100(0 代表符号为负，1100 代表真值)。

在计算机中，为了 CPU 在进行运算时的简单以及在对数的编码时的惟一性，对数据采用了补码的形式来存储。而为了得到补码，引入了反码。

对于正数，其反码和补码与原码相同。

对于负数，反码是在原码的基础上通过对真值按位取反，即 0 变为 1，1 变为 0。补码是在反码的基础上加 1 得到。

**例 1.1.12**　求一个 8 位的二进制负数 11001100 的反码。

最高位的符号 1 保持不变；1001100 按位取反，得 0110011。故其反码为 10110011。

**例 1.1.13**　求负数 -65 的补码。

第一步，求原码，得：1 1000001

第二步，求反码，得：1 0111110

第三步，求补码，得：1 0111110

\+ 1

1 0111111

### 1.1.2 计算机中的数据表示

一般来说,“数据”有数值数据及非数值数据。数值数据以带符号和无符号整数,以及浮点数(实数),BCD 形式出现。非数值数据常以 ASCII 等形式出现。其他数据表示格式(如余 3 码)由于并不常见,这里就不再赘述。

**(1)非数值数据的表示(ASCII 码)**

ASCII 码(美国标准信息交换码)是目前普遍采用的、比较通用的字符二进制编码。标准 ASCII 码是一个 7 位二进制编码,它的第 8 位即最高位在某些系统中用于保存奇偶校验位。如果 ASCII 码是用在打印机上的,若用于字母数字打印则最高位是 0,用于图形打印则最高位是 1。而在个人计算机上最左面那位(第 8 位)放置一个逻辑 1 可以选择一个扩展 ASCII 码集。扩展的 ASCII 字符存储一些异国字母以及标点,希腊字符,算术字符,画框字符以及其他的特殊字符。注意扩展字符可能随打印机而异。表 1.2 是 ASCII 码表。

**表 1.2 7 位二进制编码的 ASCII 码表**

| | X0 | X1 | X2 | X3 | X4 | X5 | X6 | X7 | X8 | X9 | XA | XB | XC | XD | XE | XF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0X | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEI | BS | HT | LF | VT | FF | CR | SO | SI |
| 1X | DLE | DCl | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US |
| 2X | SP | ! | “ | # | $ | % | & | ‘ | ( | ) | * | + | , | - | . | / |
| 3X | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4X | @ | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
| 5X | P | Q | R | S | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | - |
| 6X | ` | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o |
| 7X | p | q | r | s | t | u | v | w | x | y | z | { | \| | } | ~ | DEL |

ASCII 控制字符,同样也在表 1.2 中列出,它们在计算机系统中实施控制功能,包括清屏,退格,换行等。为了从计算机键盘上输入控制编码,当敲入一字母时,要同时按 Ctrl 键。为了得到控制代码 0lH,敲入 Ctrl + A;敲入 Ctrl + B 得到控制代码 02H 等。注意这时在 DOS 提示符下控制代码出现在屏幕上,Ctrl + A 显示为^A,Ctrl + B 显示为^B,等等。同时还要注意在大多数当今的键盘上,回车代码(CR)是回车键。CR 的目的在于把光标或打印头返回到最左边。另外一个在许多程序中出现的代码是换行代码(LF),负责把光标下移一行。

为了用表 1.2 把字母,数字或控制字符转换成 ASCII 字符,首先找到用于转换的字母,数字代码。接着,找寻十六进制 ASCII 码的第一位数字。然后找第二位。例如,大写字母 A 的 ASCII 码是 4X + X1 = 41H,而小写字母 d 的 ASCII 码是 6X + X4 = 64H。

**(2)数值数据的表示**

1)一般表示

正如前一小节所介绍的,在计算机中,数大多采用二进制表示。如对无符号整数,采用二进制原码表示,对有符号整数,采用二进制补码表示,对实数,常采用浮点数格式或指数格式来表示。实数表示格式在此不再介绍,读者可参阅其他资料。

2)BCD 码表示

以二进制来表示的十进制数称 BCD 码。二进制编码的十进制(BCD)信息以压缩或非压缩

形式存储。压缩 BCD 数据以每字节两位数字方式存储，非压缩 BCD 数据以每字节一位数字形式存储。BCD 数字的范围由 $0000_2$ 到 $1001_2$，或者十进制的 0 ~ 9，非压缩 BCD 编码数据经常由键盘返回，而压缩 BCD 数据经常在一些指令中使用，包括微处理器指令系统中的 BCD 加法和减法指令。

表 1.3 给出一些十进制数转换成压缩和非压缩 BCD 形式的例子。BCD 数据主要应用于简单算术运算的设备。如果一个系统需要复杂的算术运算，BCD 码很少被使用，因为没有实施复

**表 1.3　压缩和非压缩 BCD 数据**

| 十进制 | 压　缩 | 非 压 缩 |
|---|---|---|
| 12 | 0001 0010 | 0000 0001 0000 0010 |
| 632 | 0000 0110 0011 0010 | 0000 0110 0000 0011 0000 0010 |
| 910 | 0000 1001 0001 0000 | 0000 1001 0000 0001 0000 0000 |

杂 BCD 算术运算的简单且有效的方法。

**(3)数据类型**

计算机存取的以二进制位表示的信息位数一般是 8 的倍数。

1)字节数据

一个字节由 8 个二进制位组成。字节数据以无符号整数或带符号整数形式存储。图 1.1 列出了字节整数的无符号和符号形式。这些形式的区别是最左位是数位还是符号位。无符号

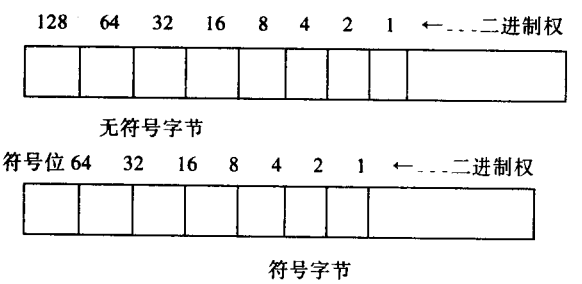



图 1.1　无符号和符号字节，并列出每一二进制位的权

整数在值域上范围是从 00H ~ FFH(0 ~ 255)。带符号整数的值域范围是 - 128 到 + 127。

2)字数据

一个字(16 位)由两个字节的数据组成。一个带符号字和一个无符号字的惟一区别是最左位是代表数位还是符号位。如果是数位，则其数的范围为 0 ~ 65 535；如果是符号位，则其数的范围为 - 32 768 ~ + 32 767。正如字节带符号数一样，带符号字也以补码形式表示一个负数。

3)N 字节数据

由 N 个字节组成。常有双字，四字，十字节数据等。其带符号数和无符号数的惟一区别是最左位是代表数位还是符号位。如果是数位，则其数的范围为 $0 \sim 2^{N*8}-1$；如果是符号位，则其数的范围为 $-2^{(N*8-1)} \sim +2^{(N*8-1)}-1$。带符号数也以补码形式表示一个负数。

4)字符串

字符串是指由字符构成的一个线性数组。通常每个字符用一个字节表示，但有时每个字

符也可用一个字或一个双字来表示。

## 1.2 微型计算机的组成

### 1.2.1 几个基本定义

微处理器、微型计算机和微型计算机系统，这是三个含义不同但又有着密切依存关系的基本概念。

**(1)微处理器**

微处理器简称 μP 或 MP(Microprocessor)，是指由一片或几片大规模集成电路组成的具有运算器和控制器功能的中央处理部件，又称为微处理机。它本身并不等于微型计算机，而只是其中央处理器。通常，在微型计算机中直接用 CPU 表示微处理器。

**(2)微型计算机**

微型计算机(Microcomputer)，简称 μC 或 MC，是指以微处理器为核心，配以存储器、输入/输出接口电路等部件，并由系统总线(地址总线，数据总线，控制总线)相联系的计算机(又称主机或微电脑)。当把微处理器、存储器和输入/输出接口电路统一组装在一块或多块电路板上或集成在单片芯片上，则分别称之为单板、多板或单片微型计算机。

**(3)微型计算机系统**

微型计算机系统(Microcomputer system)，简称 μCS 或 MCS，是指以微型计算机为中心，配以相应的外围设备、电源和辅助电路(统称硬件)以及管理指挥微型计算机工作的系统软件所

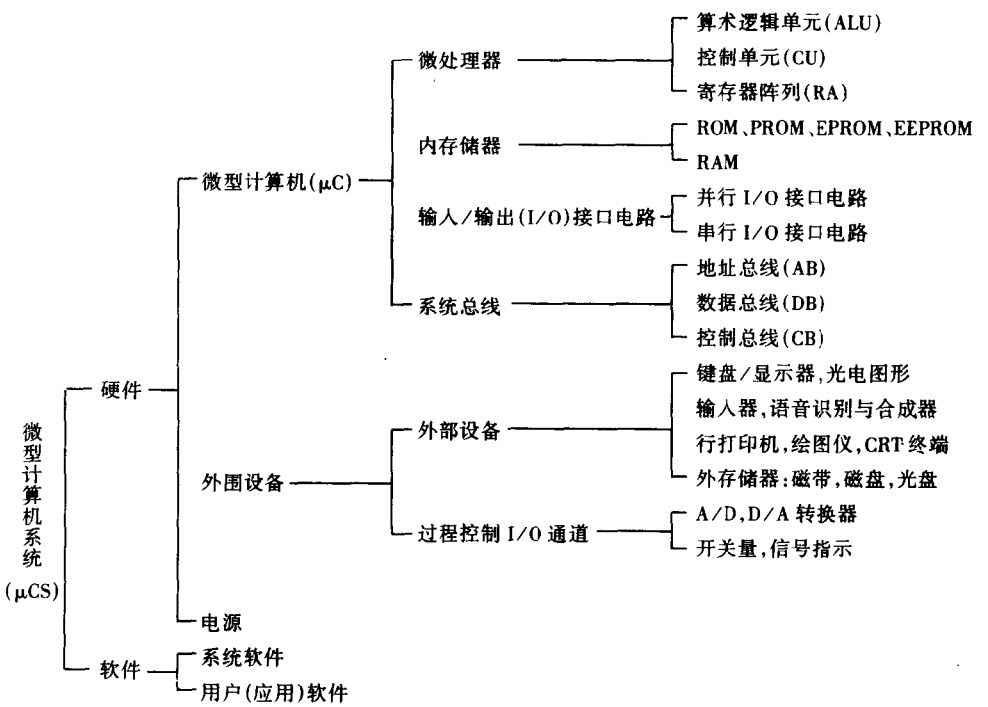



图 1.2 微机系统组成

构成的系统。

以上三者的含义及相互关系如图 1.2 所示。

### 1.2.2　微型计算机系统的组成

微型计算机系统与任何其他计算机系统一样，由硬件和软件两个主要部分组成。随着技术进步，外围设备在不断增加新成员。

**(1)硬件**

微机硬件系统的组成如图 1.3 所示。图中，微处理器是微机的运算、控制中心，用来实现算术、逻辑运算，并对全机进行控制。存储器(简称主存或内存)用来存储程序或数据。输入/输出(I/O)芯片是微机与输入输出设备之间的接口。

微机硬件系统结构：

所谓微机硬件系统结构系指按照总体布局的设计要求将各部件构成某个系统的连接方式。一种典型的微机硬件系统结构如图 1.4 所示。图中，用系统总线将各个部件连接起来。

系统总线是用来传送信息的公共导线，所有的信息都通过总线传送。通常，根据所传送信息的内容与作用不同，可将总线分为 3 类：数据总线 DB(Data Bus)，地址总线 AB(Address

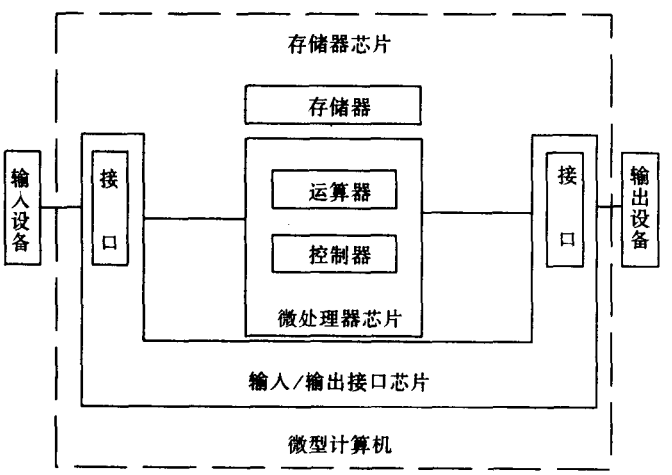



图 1.3　微机硬件系统组成框图

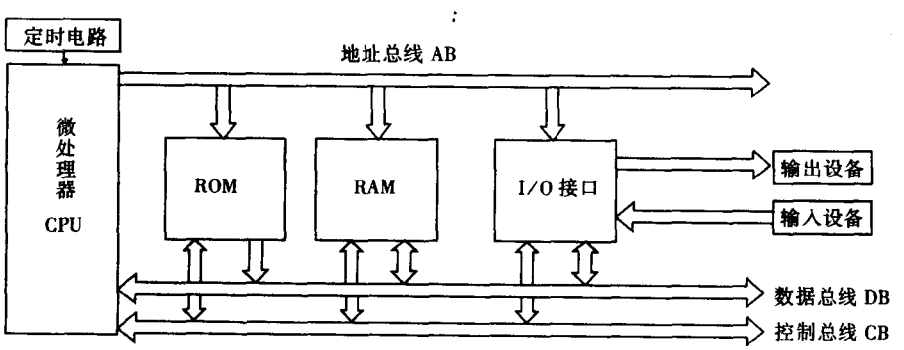



图 1.4　典型的微机硬件系统结构