高校计算机教材丛书

微型计算机原理及应用

李伯成　侯伯亨　张毅坤　编著

COMPUTER

西安电子科技大学出版社

微型计算机原理及应用

微型计算机原理及应用

李伯成　侯伯亨　张毅坤　编著

西安电子科技大学出版社

1998

内 容 简 介

本书以 8086(8088)为对象，主要介绍微型计算机的基本结构、指令系统、汇编语言程序设计及基本的程序设计方法、内部存贮器、I/O 技术及典型的接口芯片的使用、微机应用系统设计及调试等。本书特别注意阐明基本概念、方法及工程实践问题。此外，书中还对 Pentium 处理器等做了简要介绍。

本书着眼于工程应用，通过学习为微机应用打下坚实的基础。书中内容简明扼要、深入浅出，融入了作者的经验和体会，便于阅读和学习。本书除可作为高校各专业的教材使用外，对一般工程技术人员也有较好的参考价值。

微型计算机原理及应用

李伯成　侯伯亨　张毅坤　编著

责任编辑　徐德源

西安电子科技大学出版社出版发行

西安市秦群印刷厂印刷

各地新华书店经销

开本 787×1092　1/16　印张：22　字数：518 千字

1998 年 5 月第 1 版　1998 年 5 月第 1 次印刷　印数：1 - 4 000

ISBN 7 - 5606 - 0581 - 8/TP・0292　　定价：22.00 元

序　言

随着技术的发展和进步，微型计算机的应用在各行各业中迅猛发展。它已成为每个专业技术人员必备的基础。本书是为高校各专业师生及一般科技人员学习微型计算机的需要而编写的。

如何学习微型计算机，使自己很快入门是经常困扰初学者的问题。而微机的发展，不管是硬件还是软件，真可谓日新月异，使人眼花缭乱。从通用的CPU 到单片机、数字信号处理器(DSP)、位片机、专用处理器等等，它们均由许多厂家进行生产，本身又有许多系列。我们认为可以从特殊到一般进行学习。即选择国内比较流行的某种型号的微型机(或单片机)，认真仔细地学好，建立正确的概念。只要进了门，再遇到其他类型的微型机你将很容易掌握它们。因为，尽管型号不同，但它们共性的东西是大量的。为此，我们以 8086(8088)为对象，为读者做深入分析和描述。

本书偏重于工程应用。因此，对于各种芯片(包括 CPU)，我们强调读者抓住其外部特性，能将它们用好就达到了目的。至于芯片内部的东西，以工程够用即可。读者也没有必要搞清楚那些大(或超大)规模集成电路芯片的内部细节。

微型计算机能够不知疲倦地一条指令接一条指令地执行程序，从而实现你所要求的各种功能。在学习本书时，请读者注意这门课的一些特点。

全书共分 8 章，从最基本的概念入手，引导读者逐步掌握微型机从硬件组成到软件编程的基本知识。通过本书的学习，使读者能初步掌握微型计算机组成原理和简单的应用。编写过程中力求重点突出，通俗易懂。在内容上做到简明扼要，深入浅出，便于各类人员阅读和学习。

本书的第 1、3 章由张毅坤编写，第 2、5、6、章由侯伯亨编写，第 4、7、8 章由李伯成编写。全书由李伯成主编并统稿。此书在编写过程中得到陕西省教委有关同志的支持和帮助，在此表示衷心的感谢。由于水平所限，加之时间匆促，错误和不当之处在所难免，敬请读者批评指正。

编者

1997 年 8 月于西安

目　录

第 1 章 预备知识

在本章里，主要介绍计算机中常用的数制与编码。尽管这些内容比较简单，但对于学习微型计算机原理来说，它们是必不可少的基础知识，故应当熟练掌握。

1.1 数与数制

1.1.1 十进制记数法

十进制记数法是使用最广、人们最熟悉的一种记数方式。

在十进制记数中，用 0，1，2，…，9 这 10 个符号来表示数量，无论多大的数，都是用这 10 个符号的组合来表示的。

在这种记数法中采用位值法则，即对每个数位赋予一定的位值，又称权。不同位的权是不一样的，上一位的权是下一位的 10 倍，通常我们所说的个位、十位、百位、千位等等就是说的权。个位的权是 10^0、十位的权是 10^1、百位的权是 10^2，依次类推。利用这种法则，就可以表示任意大小的数。例如，十进制数 3 758 可用上面的法则来表示：

$$(3758)_{10} = 3 \times 10^3 + 7 \times 10^2 + 5 \times 10^1 + 8 \times 10^0$$

式中的 10^3，10^2，10^1，10^0 分别表示各位的权。可见，表示一个数的值可用每位上的数乘以该位的权而后相加得到。

根据同样的法则，也可以表示十进制小数，小数点的右边各位的权为 10^{-1}，10^{-2}，10^{-3}，…。例如，十进制数 275.368 可以用上述法则写成：

$$(275.368)_{10} = 2 \times 10^2 + 7 \times 10^1 + 5 \times 10^0 + 3 \times 10^{-1} + 6 \times 10^{-2} + 8 \times 10^{-3}$$

1.1.2 二进制记数法

二进制记数法用来表示数量的符号只有两个，就是 0 和 1。二进制数中的任何一个 0 或 1 称为比特(bit)。

与十进制记数法类似，一个二进制数可利用位值记数法来表示，每一位具有不同的权，权的大小以 2 的幂来表示。例如，二进制数 110101 可以表示为

$$(110101)_2 = 1\times 2^5 + 1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0$$

从上式中可以看到，每一位的位置不同，它们的权也不同，从最低位开始，权分别是 2^0，2^1，2^2，2^3，…。

同理，上述方法能够用来表示二进制小数。在表示中所不同的是从小数点向右，每位的权分别是 2^{-1}，2^{-2}，2^{-3}，…。例如，二进制小数 0.101101 可以表示为

$$(0.101101)_2 = 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3} + 1\times 2^{-4} + 0\times 2^{-5} + 1\times 2^{-6}$$

1.1.3　二进制数与十进制数的相互转换

在微处理机的学习和应用中，经常要用到二进制数和十进制数的相互转换问题，读者应能熟练地完成它们之间的相互转换。

1. 二进制数转换成十进制数

如上所述，只要将二进制数的每一位乘上它的权然后加起来就可以求得二进制数的十进制数值。例如，二进制数 101101.11 换算成十进制数为

$$(101101.11)_2 = 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 1\times 2^{-1} + 1\times 2^{-2} = (45.75)_{10}$$

2. 十进制数转换成二进制数

十进制数转换为二进制数的方法分两步进行。

首先说明十进制整数的转换方法。十进制整数转换为二进制整数的法则就是“除 2 取余”。即对十进制整数连续除以 2，每次相除所得的余数就构成了所要转换的二进制数，而每次所得的整数商继续被 2 除，直到商为零为止，最后所得的余数就是所转换的二进制数的最高位。例如，欲将十进制数 175 转换为二进制数，其过程如下：

175÷2=87	余数为	1
87÷2=43		1
43÷2=21		1
21÷2=10		1
10÷2=5		0
5÷2=2		1
2÷2=1		0
1÷2=0		1

得到结果：$(175)_{10}=(10101111)_2$。

然后，再来说明十进制小数的转换过程。十进制小数转换为二进制小数的法则叫做“乘二取整”。就是将十进制小数连续乘 2，每乘一次取出乘积的整数部分上的 0 或 1，并将小数部分继续乘 2，直到相乘结果的小数部分为零或达到一定的精度为止，这时所取出的整数就构成了要转换的二进制小数。开始取得的整数为二进制的高位，最后取得的整数为二进制小数的最低位。例如，将十进制小数 0.718 75 转换成二进制小数，其过程如下：

0.718 75×2=1.437 5	整数部分	1
0.437 5 ×2=0.875		0

0.875　×2＝1.75　　1

0.75　　×2＝1.5　　1

0.5　　×2＝1.0　　1

于是，得到结果为：$(0.71875)_{10}=(0.10111)_2$。

综上所述，一个十进制整数的二进制转换方法就是“除 2 取余”；而一个十进制小数的二进制转换方法就是“乘 2 取整”。若一个十进制数既包含整数部分又包含小数部分，它的二进制转换就是将它的整数部分和小数部分用上述方法分别进行转换，最后将转换好的两部分结合在一起形成要转换的二进制数，例如，

$$(175.71875)_{10}=(10101111.10111)_2$$

1.1.4　八进制记数法

八进制记数法采用 0，1，2，…，7 这样 8 个符号来表示数量。同前面所说的十进制、二进制记数法一样，八进制数的不同位具有不同的权，权是用 8 的幂来表示的。例如，八进制数 372.01，根据各位的权不同可以写成：

$$(372.01)_8=3\times8^2+7\times8^1+2\times8^0+0\times8^{-1}+1\times8^{-2}$$

将上式中各位与其权相乘而后加到一起，就可以得到八进制数 372.01 的十进制数为

$$(372.01)_8=(250.015625)_{10}$$

这也表明了八进制数转换为十进制数的过程。

十进制数转换为八进制数的方法是：对于十进制整数采用“除 8 取余”的方法转换为八进制整数；对于十进制小数则采用“乘 8 取整”的方法转换为八进制小数。例如，将十进制数 194.468 75 转换成八进制数时，应将整数部分和小数部分分别转换，最后再合到一起就得到要转换的八进制数：

194÷8＝24　余数为　2　　0.468 75 ×8＝3.75　整数部分　3

24÷8＝3　　0　　0.75　　×8＝6.0　　6

3÷8＝0　　3

所以，

$$(194.46875)_{10}=(302.36)_8$$

二进制数转换成八进制数的方法就是从小数点起，把二进制数每三位分成一组，然后写出每一组的等值八进制数，顺序排列起来就得到所要求的八进制数。例如，将二进制数 11101111010.1011 转换为八进制数：

$$(011\ 101\ 111\ 010\ .\ 101\ 100)_2$$

$$(3\quad 5\quad 7\quad 2\ .\ 5\quad 4)_8$$

在转换过程中，为了构成三位一组，允许在最高位和最低位上补零。

依据同样的思想，即一位八进制数用三位二进制数表示，就可以直接将八进制数转换成二进制数。例如，将八进制数 712.46 转换为二进制数，其过程如下：

$$(7\quad 1\quad 2\ .\ 4\quad 6)_8$$

$$(111\ 001\ 010.100\ 110)_2$$

1.1.5 十六进制记数法

十六进制记数法是微处理机中最常用的一种数制。顾名思义，十六进制记数法采用 0，1，2，3…，9，A，B，C，D，E，F 这 16 个符号来表示数量。同样，一个十六进制数的每一位都有自己的权。权是由 16 的幂来表示的。这样，一个十六进制数就能够用各位与它们相应的权来表示。例如，十六进制数 E5D7.A3 可以表示为

$$(E5D7.A3)_{16} = E \times 16^3 + 5 \times 16^2 + D \times 16^1 + 7 \times 16^0 + A \times 16^{-1} + 3 \times 16^{-2}$$

同前所述，一个十进制数可以转换成十六进制数。其方法就是十进制的整数部分采用“除 16 取余”的方法得到十六进制数的整数部分，而十进制的小数部分则采用“乘 16 取整”的方法来获得。例如，将十进制数 47 632.781 25 转换成十六进制数，其转换过程分述如下。

整数部分：

47 632÷16=2 977	余数	0→16 进制数	0
2 977÷16=186		1→	1
186÷16=11		10→	A
11÷16=0		11→	B

小数部分：

0.781 25×16=12.5	整数	12→	C
0.5×16=8.0		8→	8

最后得到$(47632.78125)_{10}=(BA10.C8)_{16}$。

十六进制数转换为十进制数，由前面的叙述可以很方便地实现，只要将十六进制数各位与它们对应的权相乘，再加到一起就可得到。

由于一位十六进制数可以用四位二进制数来表示，因此二进制数与十六进制数的相互转换就比较容易。二进制数到十六进制数的转换是由小数点开始，每四位二进制数为一组，将每一组用相应的一位十六进制数来表示，即可得到正确的十六进制数，例如：

$$(1\ 1101\ 0100\ 1011\ 0111\ .\ 0101\ 1110\ 1010)_2$$
$$(1\quad D\quad 4\quad B\quad 7\quad .\quad 5\quad E\quad A)_{16}$$

相反，将十六进制数转换成二进制数，只要将十六进制数的每一位用其等值的四位二进制数代替，连在一起就得到了我们所需要的二进制数。例如，十六进制数 E7FF 转换成二进制数的过程如下：

$$(E7FF)_{16} = (1110\ 0111\ 1111\ 1111)_2$$
$$(E\quad 7\quad F\quad F)_{16}$$

到现在为止，我们介绍了几种常用的数制以及它们之间的相互转换。有一些转换做起来比较容易，也有一些比较麻烦，例如十六进制数与八进制数之间的转换以及它们与十进制数的相互转换等。遇到这种情况，可以直接进行转换，也可以通过二进制数进行中间转换。总之，读者比较熟练地掌握这些数制，对学好本书是有益的。

1.2　十进制数与字符的编码表示

1.2.1　BCD 码

转换十进制数为其等值的二进制数称之为编码。前面所提到的二进制数称为纯二进制码。微处理器只能识别用高低电平表示的 0 或 1。就其工作来说，纯二进制码用于微处理机是十分方便的，这一点后面还将进一步说明。但就人们的习惯来说，对十进制数更熟悉，用起来也很直观，遗憾的是微处理机却无法直接进行操作。为此，人们发明一种特殊的二进制编码，它兼有二进制和十进制记数的特点，既符合人们的习惯，计算机又能直接进行运算。人们将这种编码叫做二—十进制码，简称 BCD 码。

用二进制编码来表示十进制数的方案有多种。它们都是用 4 位二进制编码来表示一位十进制数的。经常使用的一种编码方案是所谓的 8421 码，其编码原则是每位十进制数用 4 位等值的二进制数来表示，从左到右各位二进制数的权为 8421，故名 8421 码。表 1.1 分别列出了十进制数、纯二进制码和 8421 BCD 码的对应关系。

从表 1.1 中可以看到，BCD 码仅仅利用了 4 位二进制编码的 10 种组合，而代表十进制数 10～15 的二进制编码，对于 BCD 码来说是非法的。很明显，BCD 码只利用了二进制中对应 0～9 的 10 种码组，而且只用这 10 种就已足够，剩下的 6 种二进制码组是不允许使用的。

表 1.1　BCD 码与其它数制的对应关系

十进制数	8421 BCD 码	纯二进制码
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 0
3	0 0 1 1	0 0 1 1
4	0 1 0 0	0 1 0 0
5	0 1 0 1	0 1 0 1
6	0 1 1 0	0 1 1 0
7	0 1 1 1	0 1 1 1
8	1 0 0 0	1 0 0 0
9	1 0 0 1	1 0 0 1
10	0 0 0 1 0 0 0 0	1 0 1 0
11	0 0 0 1 0 0 0 1	1 0 1 1
12	0 0 0 1 0 0 1 0	1 1 0 0

续表

十进制数	8421 BCD 码	纯二进制码
13	0 0 0 1 0 0 1 1	1 1 0 1
14	0 0 0 1 0 1 0 0	1 1 1 0
15	0 0 0 1 0 1 0 1	1 1 1 1

根据上述说明，一个十进制数，能够很方便地用 BCD 码来表示。例如，十进制数 834 用 BCD 码表示为

$$(834)_{10} = (1000\ 0011\ 0100)_{BCD}$$

为了避免与二进制编码混淆，在 BCD 码表示中，每位 BCD 码(4 位二进制数)写成一组，中间留有空隙，而且要标明此数为 BCD 数，如上例中所写的那样。

只要熟记十进制数 0～9 与 BCD 码的对应关系，则它们之间的相互转换是十分方便的。例如：

$$(0110\ 1001\ 0101.0010\ 0111\ 1001)_{BCD} = (695.279)_{10}$$

将每一位 BCD 码所表示的十进制数直接写出来，就得到相应的十进制数。反之，将十进制数用其相应的 BCD 码代替，就可获得相应的 BCD 码。

二进制数与 BCD 码的相互转换要略微麻烦一点，一般要通过一个向十进制转换的中间步骤，再由十进制数转换成所要求的编码。例如，要将二进制数 1011.01 转换成 BCD 码，则首先将它转换成十进制数 11.25，而后再将此十进制数转换成 BCD 码，即 $(0001\ 0001.0010\ 0101)_{BCD}$。

同样，由 BCD 码转换成二进制数亦然。

BCD 码在计算机中常用两种表示方法。一种如上所述，用 4 位二进制编码来表示一位十进制数，有时称这种表示为压缩十进制编码或压缩 BCD。这种表示法，用 8 位二进制数即可以表示两位十进制数，例如，$(78)_{10}=(0111\ 1000)_{BCD}$。另一种称为扩展 BCD，它利用 8 位二进制数表示一位十进制数，例如，$(78)_{10}=(00000111\ 00001000)_{BCD}$。

前一种方法利用率高，占用内存少；但后一种方法，在某些场合下又比较方便。

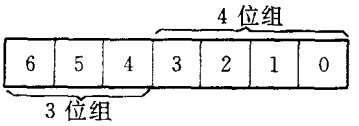
1.2.2 ASCII 码

ASCII 码是美国标准信息交换码的简称，现在为各国所广泛采用。

通常，ASCII 码由 7 位二进制编码来表示，用于微处理机与它的外部设备之间进行数据交换以及通过无线或有线进行数据传送。

7 位二进制编码共有 128 种不同的组合，用以表示 128 种不同的字符或功能。它们是十进制数字 0～9，大写和小写的 26 个英文字母，标点符号，一些特殊符号及由两三个大写英文字母组成的特殊控制功能字符。详细情况见附录 A。

代表上述字符或控制功能的 ASCII 码是由一个 4 位组和一个 3 位组构成的，形成 7 位二进制编码，其格式为



根据 ASCII 码的构成格式，可以很方便地从附录 A 中 ASCII 表查出每一个字符或特殊控制功能的编码。例如，大写英文字母 A，从表中查出其 3 位组为$(100)_2$，4 位组为$(0001)_2$，故构成字母 A 的 ASCII 编码为$(1000001)_2$或$(41)_{16}$。

除了上述的 7 位 ASCII 码外，还有简单的 6 位 ASCII 码。这种情况下，ASCII 码的构成格式由一个 2 位组和一个 4 位组构成。即低 4 位构成 4 位组，上述 3 位组变为 2 位组。

由于 6 位 ASCII 码仅有 64 种组合，因此，它只能表示 7 位 ASCII 表中所示的第 2、3、4、5 列字符。在一些简单应用的场合下，可以使用 6 位 ASCII 码。

有时，在 7 位 ASCII 码的基础上附加上一位放在最高位的左边，形成 8 位 ASCII 码，而新附加上的这一位是 8 位中的最高位。附加这一位常用于奇偶校验，用来表征数据传送过程中是否有一位出现了错误。

偶校验的含义是，包括这一位在内的 8 位二进制码中为 1 的位数之和为偶数。例如，字母 A 的 ASCII 码$(1000001)_2$变为 8 位 ASCII 表示时，应使 8 位中为 1 的各位之和为偶数，故偶校验位应为 0，于是具有偶校验位的 A 的 ASCII 码就变为$(01000001)_2$。

奇校验的含义是，包括校验位在内，所有为 1 的位数之和为奇数。根据此原则，具有奇校验位的 A 的 ASCII 码就是$(11000001)_2$。

1.3　二进制算术运算

1.3.1　二进制加法

二进制加法与十进制加法相类似，所不同的是，二进制加法中是“逢二进一”，其法则为

$$0+0=0$$
$$1+0=1$$
$$0+1=1$$
$$1+1=0 \text{ 并进位}$$

例如，两个二进制数相加：

$$\begin{array}{r} 10110101 \\ +\ 10001110 \\ \hline 101000011 \end{array}$$

1.3.2　二进制减法

在二进制减法中，同样有如下法则：

0－0＝0
1－0＝1
1－1＝0
0－1＝1 有借位

当不够减时需要借位，高位的 1 等于下一位的 2，即“借一当二”。例如，两个二进制数相减：

```
  10110100
- 01010111
----------
  01011101
```

1.3.3 二进制乘法

二进制乘法与十进制乘法是一样的。但因为二进制数只由 0 和 1 构成，因此，二进制乘法更简单。其法则如下：

0×0＝0
1×0＝0
0×1＝0
1×1＝1

例如，二进制数 1101.1 与 101.1 相乘：

```
    1101.1          被乘数(13.5)10
  ×  101.1          乘数  (5.5)10
---------
    1101 1
   11011
  00000
+ 11011
---------------
 1001010.0 1        乘积  (74.25)10
```

再看下面的例子：$(1011)_2 \times (1101)_2$。

```
     1011          被乘数(11)10
   × 1101          乘数  (13)10
---------
     1011
    0000
   1011
+  1011
---------------
 10001111          乘积  (143)10
```

从上述乘法运算中我们可以看到，从乘数的最低位起，凡遇到 1，就相当于在最终结果上加上一个被乘数，而遇到零则不加。但必须注意相加的位置，乘数最低位是 1，被乘数直接加在结果的最右边；而次低位是 1，应左移一位后再加；再次低位是 1，左移两位后再