大学计算机基础教育规划教材

微型计算机原理及接口技术

李伯成 编著

清华大学出版社

大学计算机基础教育规划教材

微型计算机原理及接口技术

李伯成 编著

清华大学出版社
北京

内容简介

本书介绍微型计算机各主要组成部分的工作原理和工程上的实现方法。内容上强调基本概念，以及分析问题和解决问题的方法。在说明一些常用的典型接口芯片的基础上，重点介绍利用这些概念和方法设计常见外设的接口。通过本书的学习，读者能够独立地设计一个小的微型计算机系统。

本书适合作为高等院校非计算机专业的教材，也可供其他技术人员参考。

图书在版编目(CIP)数据

微型计算机原理及接口技术/李伯成编著. 北京：清华大学出版社，2005.1
(大学计算机基础教育规划教材)
ISBN 7-302-09769-0

Ⅰ.微… Ⅱ.李… Ⅲ.①微型计算机－理论－高等学校－教材 ②微型计算机－接口－高等学校－教材 Ⅳ.TP36

中国版本图书馆 CIP 数据核字(2004)第 107108 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦
http://www.tup.com.cn　　**邮　　编**：100084
社 总 机：010-62770175　　**客户服务**：010-62776969

组稿编辑：张　民
文稿编辑：霍志国
封面设计：孟繁聪
印 刷 者：北京市昌平环球印刷厂
装 订 者：三河市李旗庄少明装订厂
发 行 者：新华书店总店北京发行所
开　　本：185×260　**印张**：21　**字数**：494 千字
版　　次：2005 年 1 月第 1 版　2005 年 1 月第1 次印刷
书　　号：ISBN 7-302-09769-0/TP・6746
印　　数：1～5000
定　　价：26.00 元

序

大学计算机基础教育规划教材

进入21世纪，社会信息化不断向纵深发展，各行各业的信息化进程不断加速。我国的高等教育也进入了一个新的历史发展时期，尤其是高校的计算机基础教育，正在步入更加科学、更加合理、更加符合21世纪高校人才培养目标的新阶段。

为了进一步推动高校计算机基础教育的发展，教育部高等学校非计算机专业计算机基础课程教学指导分委员会近期提出了《关于进一步加强高校计算机基础教学的几点意见》(以下简称《意见》)。《意见》针对计算机基础教学的现状与发展，提出了计算机基础教学改革的指导思想；按照分类、分层次组织教学的思路，《意见》的附件提出了计算机基础课教学内容的知识结构与课程设置。《意见》认为，计算机基础教学的典型核心课程包括：大学计算机基础、计算机程序设计基础、计算机硬件技术基础(微机原理与接口、单片机原理与应用)、数据库技术与应用、多媒体技术与应用、网络技术与应用。附件中介绍了上述六门核心课程的主要内容，这为今后的课程建设及教材编写提供了重要的依据。在下一步计算机课程规划工作中，建议各校采用“1＋X”的方案，即：“大学计算机基础”＋ 若干必修／选修课程。

教材是实现教学要求的重要保证。为了更好地促进高校计算机基础教育的改革，我们组织了国内部分高校教师进行了深入的讨论和研究，根据《意见》中的相关课程教学基本要求组织编写了这套“大学计算机基础教育规划教材”。

本套教材的特点如下：

(1) 体系完整，内容先进，符合大学非计算机专业学生的特点，注重应用，强调实践。

(2) 教材的作者来自全国各个高校，都是教育部高等学校非计算机专业计算机基础课程教学指导委员会推荐的专家、教授和教学骨干。

(3) 注重立体化教材的建设，除主教材外，还配有多媒体电子教案、习题与实验指导，以及教学网站和教学资源库等。

(4) 注重案例教材和实验教材的建设，适应教师指导下的学生自主学习的教学模式。

(5) 及时更新版本，力图反映计算机技术的新发展。

本套教材将随着高校计算机基础教育的发展不断调整，希望各位专家、教师和读者不吝提出宝贵的意见和建议，我们将根据大家的意见不断改进本套教材的组织、编写工作，为我国的计算机基础教育的教材建设和人才培养做出更大的贡献。

《大学计算机基础教育规划教材》丛书主编

教育部高等学校非计算机专业计算机基础课程教学指导分委员会主任委员

冯博琴

2004 年 8 月

前言

微型计算机已经广泛应用于各行各业，促进了社会的发展和进步。作为工程技术人员，必须很好地掌握微型计算机的概念与技术。本书是为高等院校理工科专业教学及一般工程技术人员学习微型计算机而编写的。

如何学习并掌握微型计算机的有关知识，并用所学知识解决具体的工程问题，本书在编写过程中予以特别的关注。技术发展异常迅速，就硬件处理器而言，有各种类型的通用CPU、单片微型计算机、数字信号处理器(DSP)、片上系统(SOC)，以及专用处理器芯片，而且新的处理器芯片还在不断地涌现。通常认为，可以采取从特殊到一般的学习方法，即选择某一种典型的处理器(CPU、单片机或DSP)，认真学习并掌握其中的基本概念和基本方法。一种典型的处理器学好了，再遇到其他型号的处理器必能很容易掌握它们。这是因为它们的基本概念、基本思路和基本方法都是相同的，共性的东西是非常多的。同时，为了能在有限的时间里把基本问题描述清楚，应选择比较简单的处理器去解释复杂的概念(太复杂的处理器不太适于时间较短的课堂教学)。为此，本书以80x86为对象进行分析与描述。

本书是为理工科非计算机专业的学生编写的教材。为适应学生未来工作的需要，同时也为保证本书的体系完整，便于后面章节的讲述，在内容上将汇编语言程序设计作为第2章。全书教学实施需70学时左右。如果已经熟悉书前的预备知识，可越过这部分内容直接从第1章讲起。最后一章的内容比较繁杂，课堂讲授有一定困难，可概要说明，让学生自己阅读，慢慢加以体会。

本书的目的在于培养工程思维能力，在描述清楚基本概念的基础上，侧重于解决具体工程应用问题。要求读者能利用所学的基本概念，提出解决工程问题的思路和方法，提高分析具体工程问题和解决问题的能力。

本书由李伯成编著。在编写本书的过程中，力求以简明扼要的语言，重点突出地描述基本概念，并且在内容中融入作者教学和科研工作经验。由于水平及时间所限，错误不当之处在所难免，敬请读者批评指正。

作者

2004年7月

微型计算机原理及接口技术 目录

第 0 章　预备知识 …… 1

第 1 章　微处理器及总线 …… 8

1.1　微型计算机的基本结构 …… 8
1.1.1　微型计算机的组成及各部分的功能 …… 8
1.1.2　微型计算机的工作过程 …… 11
1.2　8088(86)CPU …… 12
1.2.1　概述 …… 12
1.2.2　8088 CPU 引线及其功能 …… 13
1.2.3　8086 CPU 引线 …… 18
1.2.4　8088 CPU 的内部结构 …… 19
1.2.5　存储器寻址 …… 22
1.2.6　8088 CPU 的工作时序 …… 24
1.3　系统总线的形成 …… 27
1.3.1　几种常用的芯片 …… 27
1.3.2　最小模式下的系统总线形成 …… 29
1.3.3　最大模式下的系统总线形成 …… 30
1.3.4　8086 的系统总线形成 …… 30
1.4　总线及其驱动 …… 31
1.4.1　总线概述 …… 31
1.4.2　内总线 …… 32
1.4.3　外总线 …… 33
1.4.4　总线驱动与控制 …… 34
习题 …… 41

第 2 章　指令系统及汇编语言程序设计 …… 43

2.1　8088(86)的寻址方式 …… 43
2.1.1　决定操作数地址的寻址方式 …… 43
2.1.2　决定转移地址的寻址方式 …… 46
2.2　8088(86)的指令系统 …… 48

2.2.1 传送指令 …… 48
2.2.2 算术指令 …… 52
2.2.3 逻辑运算和移位指令 …… 58
2.2.4 串操作指令 …… 63
2.2.5 程序控制指令 …… 66
2.2.6 处理器控制指令 …… 70
2.2.7 输入输出指令 …… 71
2.3 汇编语言 …… 72
2.3.1 汇编语言的语句格式 …… 73
2.3.2 常数 …… 74
2.3.3 伪指令 …… 75
2.3.4 汇编语言的运算符 …… 80
2.3.5 汇编语言源程序的结构 …… 82
2.4 汇编语言程序设计 …… 83
2.4.1 程序设计概述 …… 84
2.4.2 基本的程序设计方法 …… 84
2.4.3 汇编语言程序举例 …… 93
2.4.4 汇编语言程序的查错与调试 …… 98
习题 …… 100

第3章 存储器系统 …… 102

3.1 概述 …… 102
3.1.1 存储器的分类 …… 102
3.1.2 存储器的主要性能指标 …… 103
3.2 读写存储器 …… 104
3.2.1 静态读写存储器 …… 105
3.2.2 动态存储器 …… 113
3.3 只读存储器 …… 118
3.3.1 EPROM …… 118
3.3.2 EEPROM(E^2PROM) …… 122
3.4 多端口存储器 …… 129
3.4.1 双端口存储器 …… 129
3.4.2 先进先出存储器 …… 131
3.5 存储系统 …… 133
3.5.1 存储器的层次结构 …… 133
3.5.2 高速缓存 …… 134
3.5.3 虚拟存储器 …… 138
3.5.4 光盘概述 …… 141

习题…… 143

第 4 章 输入输出技术…… 145

4.1 概述 …… 145
4.1.1 外设接口的编址方式…… 145
4.1.2 外设接口的基本模型…… 146
4.2 程序控制输入输出 …… 147
4.2.1 无条件传送方式…… 147
4.2.2 查询方式…… 149
4.3 中断方式 …… 153
4.3.1 中断的基本概念…… 153
4.3.2 8086(88)的中断系统…… 158
4.3.3 中断控制器 8259 …… 163
4.4 直接存储器存取 …… 176
4.4.1 DMA 的一般过程 …… 176
4.4.2 DMA 控制器 8237 …… 177
习题…… 192

第 5 章 常用接口芯片及应用…… 193

5.1 简单接口 …… 193
5.1.1 三态门…… 193
5.1.2 锁存器…… 193
5.1.3 带有三态门输出的锁存器…… 193
5.2 可编程并行接口 8255 …… 196
5.2.1 8255 的引线及内部结构 …… 196
5.2.2 8255 的工作方式 …… 197
5.2.3 控制字及状态字…… 203
5.2.4 8255 的寻址及连接 …… 205
5.2.5 初始化及应用…… 205
5.3 可编程定时器 8253 …… 207
5.3.1 8253 的引线功能及内部结构 …… 207
5.3.2 8253 的工作方式 …… 208
5.3.3 8253 的控制字 …… 211
5.3.4 8253 的寻址及连接 …… 212
5.3.5 初始化及应用…… 213
5.4 可编程串行接口 8250 …… 215
5.4.1 概述…… 215
5.4.2 串行接口 8250 …… 216

5.4.3 串行通信总线 RS-232C …… 228
5.5 键盘接口 …… 230
5.5.1 概述 …… 230
5.5.2 矩阵键盘的基本结构 …… 230
5.5.3 非编码矩阵键盘接口的实现 …… 233
5.5.4 专用键盘接口芯片 …… 237
5.6 打印机接口 …… 237
5.6.1 打印机接口总线 …… 237
5.6.2 串行接口电路及驱动程序 …… 238
5.6.3 并行接口电路及驱动程序 …… 240
5.7 显示器接口 …… 243
5.7.1 七段数码显示器 …… 243
5.7.2 LED 接口电路 …… 243
5.8 光电隔离输入输出接口 …… 246
5.8.1 隔离的概念及意义 …… 246
5.8.2 光电耦合器件 …… 247
5.8.3 光电耦合器件的应用 …… 249
5.9 数模(D/A)变换器接口 …… 252
5.9.1 D/A 和 A/D 在控制系统中的地位 …… 252
5.9.2 D/A 变换器的基本原理 …… 253
5.9.3 典型的 D/A 变换器芯片 …… 255
5.10 模数(A/D)变换器接口 …… 259
5.10.1 A/D 变换器的主要技术指标 …… 260
5.10.2 典型的 A/D 变换器芯片 …… 262
5.10.3 A/D 变换器应用实例 …… 265
5.10.4 A/D 接口的调试 …… 271
5.11 电机接口 …… 272
5.11.1 直流电机接口 …… 272
5.11.2 步进电机接口 …… 275
习题 …… 282

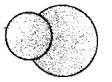
第 6 章 Pentium 处理器 …… 287

6.1 80x86 的发展过程 …… 287
6.2 Pentium 处理器引线及内部寄存器 …… 289
6.2.1 Pentium 100 的引线 …… 289
6.2.2 Pentium 100 的内部寄存器 …… 291
6.3 特权级与描述符 …… 295
6.3.1 特权级 …… 295

6.3.2 保护措施…… 297
6.3.3 描述符…… 298
6.4 工作模式 …… 302
6.4.1 实地址模式…… 302
6.4.2 保护模式…… 303
6.4.3 虚拟 8086 模式 …… 304
6.4.4 系统管理模式…… 305
6.5 中断和异常 …… 306
6.5.1 分类…… 306
6.5.2 中断或异常的响应过程…… 307
6.6 程序转移与任务的切换 …… 310
6.6.1 任务状态段…… 311
6.6.2 任务与描述符…… 312
6.6.3 控制转移的分类…… 313
6.6.4 任务内的控制转移…… 314
6.6.5 任务间的切换…… 317
6.7 其他有关问题 …… 319
6.7.1 寻址方式和指令系统…… 319
6.7.2 实地址模式到保护模式的切换…… 320
习题…… 321

参考文献…… 323

第0章



预备知识

首先介绍要用到的一些预备知识。如果已经熟悉了这些知识，则可以跳过这一部分。

1. 数与数制

(1) 十进制记数法

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。正是由于表示数量的符号有10个，故称为十进制记数法。

例如，十进制数3758可用上述法则来表示：

$$(3758)_{10}=3\times10^3+7\times10^2+5\times10^1+8\times10^0$$

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可以用上述法则表示：

$$(275.368)_{10}=2\times10^2+7\times10^1+5\times10^0+3\times10^{-1}+6\times10^{-2}+8\times10^{-3}$$

(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，即0和1。二进制数中的任何一个0或1称为比特(bit)。

例如，二进制数110101可以表示为

$$(110101)_2=1\times2^5+1\times2^4+0\times2^3+1\times2^2+0\times2^1+1\times2^0$$

(3) 二进制数与十进制数的相互转换

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

$$\begin{aligned}(101101.11)_2&=1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}\\&=(45.75)_{10}\end{aligned}$$

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，分别处理整数部分和小数部分。一个十进制整数的二进制转换方法就是“除2取余”；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。

(4) 八进制记数法

在八进制记数中，用 0,1,2,…,7 这 8 个符号来表示数量，无论多大的数，都是用这 8 个符号的组合来表示的。

(5) 十六进制记数法

在十六进制记数中，用 0,1,2,…,9 和 A,B,…,F 这 16 个符号来表示数量，即表示数量的符号有 16 个，故称为十六进制记数法。

例如，十六进制数 E5D7.A3 可以表示为

$$(E5D7.A3)_{16}=E\times16^3+5\times16^2+D\times16^1+7\times16^0+A\times16^{-1}+3\times16^{-2}$$

同前所述，一个十进制数可以转换成十六进制数，其方法为十进制数的整数部分"除十六取余"，十进制数的小数部分则采用"乘十六取整"。由于 1 位十六进制数可以用 4 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 4 位二进制数为一组，将每一组用相应的 1 位十六进制数来表示，即可得到正确的十六进制数。

2. 算术逻辑运算

(1) 二进制加法

二进制加法与十进制加法相类似。不同的是，二进制加法中是"逢二进一"，其法则如下：

0+0=0

1+0=1

0+1=1

1+1=0　　并进位

(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"。

(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

(4) 二进制除法

二进制除法是乘法的逆运算，其方法与十进制除法是一样的，而且二进制数仅由 0 和

1 构成，做起来更简单。

(5) 二进制与

二进制与又称逻辑乘，其法则如下：

$0 \wedge 0=0$；$0 \wedge 1=0$；$1 \wedge 0=0$；$1 \wedge 1=1$

(6) 二进制或

二进制或又称逻辑加，法则如下：

$0 \vee 0=0$；$0 \vee 1=1$；$1 \vee 0=1$；$1 \vee 1=1$

(7) 二进制异或

二进制异或的法则如下：

$0 \forall 0=0$；$0 \forall 1=1$；$1 \forall 0=1$；$1 \forall 1=0$

3. 符号数的表示方法

表示一个带符号的二进制数有 4 种方法。

(1) 原码法

原码法的规则就是符号连上数值，符号放在最高位，且用 0 表示正数，用 1 表示负数，其后跟着数值。

例如，8 位二进制符号数$(+45)_{10}$和$(-45)_{10}$，可以表示如下：

$$(+45)_{10}=(0\quad 0101101)_2$$

符号位　数值

$$(-45)_{10}=(1\quad 0101101)_2$$

符号位　数值

(2) 反码法

在计算机的早期，曾采用反码法来表示带符号的数。对于正数，其反码与其原码相同。例如：

$$(+45)_{10}=(00101101)_2$$

也就是说，正数用符号位与数值凑到一起来表示。

对于负数，用相应正数的原码各位取反来表示，包括将符号位取反。取反的含义就是将 0 变为 1，将 1 变为 0。例如，$(-45)_{10}$的反码表示就是将上面$(+45)_{10}$的二进制数各位取反：

$$(-45)_{10}=(11010010)_2$$

(3) 补码法

在计算机中，符号数是用补码(对 2 的补码)来表示的。用补码法表示带符号数的法则：正数的表示方法与原码法和反码法一样；负数的表示方法为该负数的反码加 1。

例如，$(+4)_{10}$的补码表示为$(00000100)_2$，而$(-4)_{10}$用补码表示时，可先求其反码表示$(11111011)_2$，然后再在其最低位加 1，变为$(11111100)_2$。这就是$(-4)_{10}$的补码表示，即$(-4)_{10}=(11111100)_2$。

(4) 移码法

在计算机的浮点数表示中会用到移码。移码可以理解为：在补码的基础上偏移多少数值。偏移的数值可以人为定义。例如，对 n 位整数来说，经常采用偏移量为 2^{n-1}。若令 n 为 8，则偏移量为 2^7，即 128。也就是说，在补码的基础上加上 128 便成为移码。

例如，+7 的补码为 00000111，则 +7 的移码为 1000111。同样，−7 的补码为 11111001，那么，−7 的移码便是 01111001。

可见，在偏移 2^{n-1} 的情况下，只要将补码的符号位取反便可获得相应的移码。

4. 补码的运算

补码加减法的运算法则为

$[X+Y]_{补}=[X]_{补}+[Y]_{补}$

$[-X]_{补}=-[X]_{补}$

$[X-Y]_{补}=[X]_{补}+[-Y]_{补}=[X]_{补}-[Y]_{补}$

由这些法则可知，和的补码可用补码求和；差的补码可将减数求补再与被减数相加，即在补码情况下，利用加法器可做减法运算。

在计算机中，一般都不设置专门的减法电路。遇到两个数相减时，处理器就自动地将减数取补，然后将被减数和减数的补码相加来完成减法运算。例如，$(69)_{10}-(26)_{10}=?$ 可以写成 $(69)_{10}+(-26)_{10}$。利用 $(69)_{10}$ 的原码和 $(-26)_{10}$ 的补码相加，即可以得到正确的结果。

注意，当两个同符号的数相加(或者是异符号码数相减)时，若相加(或相减)的结果超出了规定的数值范围，则会发生溢出。一旦发生溢出，其结果肯定是错误的。例如，两个带符号的数 $(01000001)_2$(十进制数+65)与 $(01000011)_2$(十进制数+67)相加：

$$\begin{array}{rr} & 01000001 \\ + & 01000011 \\ \hline & 10000100 \end{array}$$

可以看到，两个正数相加的结果为一负数，结果显然是荒谬的。产生错误的原因就是溢出。

再分析两个负数 $(10001000)_2$ 和 $(11101110)_2$ 的相加情况：

$$\begin{array}{rr} & 10001000 \\ + & 11101110 \\ \hline & 01110110 \end{array}$$

上面为两负数相加的情况，自己可做解释。

5. 数的定点表示和浮点表示

(1) 数的定点表示法

所谓定点数就是小数点固定不变的数。小数点的位置通常有两种约定：即定点整数(相当于小数点在最低有效位之后)和定点小数(相当于小数点在最高有效位之前)。

如前所述，对于表示带符号数，符号总是放在最高位。若表示带符号数的字长为 n 位，则定点整数原码、补码的表示范围分别如下：

定点整数原码的表示范围：$-(2^{n-1}-1)\sim+(2^{n-1}-1)$

定点整数补码的表示范围：$-(2^{n-1})\sim+(2^{n-1}-1)$

若用 n 位字长表示小数，则定点小数原码、补码的表示范围分别如下：

定点小数原码的表示范围：$-(1-2^{-(n-1)})\sim+(1-2^{-(n-1)})$

定点小数补码的表示范围：$-1\sim+(1-2^{-(n-1)})$

(2) 数的浮点表示法

定点数表示比较简单，只能是纯整数和纯小数。能表示的数值范围也比较小，运算中很容易因超出范围而溢出。为克服这些缺点而引入数的浮点表示法。

在十进制数中，一个数可以写成多种表示形式。例如，83.125 可写成 $10^2\times0.83125$，$10^3\times0.083125$，$10^4\times0.0083125$ 等。同样，一个二进制数，也可以写成多种表示形式。例如，二进制数 1011.10101 可以写成 $2^4\times0.101110101$，$2^5\times0.0101110101$，$2^6\times0.00101110101$ 等。可以看出，一个二进制数能够用一种普遍的形式来表示：

$$2^E\times F$$

其中 E 称作阶码，F 称作尾数。通常把用阶码和尾数表示的数叫做浮点数，这种表示数的方法称为浮点表示法。

在浮点表示法中，阶码通常为带符号的整数，尾数为带符号的纯小数。浮点数的一种表示格式如下：

数符	阶码	尾　数

很明显，浮点数的表示不是惟一的。当小数点的位置改变时，阶码也随着相应改变，可以用多种形式来表示同一个数。

浮点数能表示的数值范围主要由阶码决定，表示数值的精度则主要由尾数来决定。为了充分利用尾数来表示更多的有效数字，通常均采用规格化浮点数。规格化就是将尾数限定在小于 1 且大于等于 0.5 之间。

当尾数用补码表示时，若尾数 $M\geqslant0$，则尾数规格化应为 $M=0.1xxx\cdots x$。其中 x 可为 0，也可为 1。

若尾数 $M<0$，规格化满足 $[-1/2]_{补}>[M]_{补}\geqslant[-1]_{补}$，则尾数规格化应为 $M=1.0xxx\cdots x$。其中 x 可为 0，也可为 1。

(3) 工业标准 IEEE 754

IEEE 754 是由 IEEE 制定的有关浮点数的工业标准，它被广泛采用。该标准的表示形式如下：

$$(-1)^S2^E(b_{0\diamond}\,b_1b_2b_3\cdots b_{P-1})$$

其中，$(-1)^S$ 为该浮点数的数符，当 S 为 0 时表示为正数，S 为 1 时为负数；E 为指数，用移码表示；$(b_{0\diamond}\,b_1b_2b_3\cdots b_{P-1})$ 为尾数，共 P 位，用原码表示。

目前，计算机中使用的 3 种形式的 IEEE 754 浮点数格式如表 0.1 所示。

在 IEEE 754 标准中，特别要说明的就是尾数在规格化时的处理。也就是说在规格化的过程中必须使 b_0 为 1，而且小数应当在◇位置上，是隐含的。规格化时将 b_0 去掉，也是隐含的。在使用时应注意到这种情况。

表 0.1 3 种形式的 IEEE 754 浮点数格式

参 数	单精度浮点数	双精度浮点数	扩充精度浮点数
浮点数字长	32	64	80
尾数长度 P	23	52	64
符号位 S	1	1	1
指数长度 E	8	11	15
最大指数	+127	+1023	+16383
最小指数	−126	−1022	−16382
指数偏移量	+127	+1023	+16383
可表示的实数范围	$10^{-38}\sim10^{38}$	$10^{-308}\sim10^{308}$	$10^{-4932}\sim10^{4932}$

为了说明 IEEE 754 浮点数的应用，举例如下：

现欲利用 IEEE 754 标准将数 176.0625 表示为单精度浮点数。首先将该十进制数转换成二进制数：

$$(176.0625)_{10}=(10110000.0001)_2$$

对上面二进制数规格化：

$$10110000.0001=1.\diamond01100000001\times2^7$$

这就保证了使 b_0 为 1，而且小数在◇位置上。将 b_0 去掉并扩展为单精度浮点数所规定的 23 位尾数：01100000001000000000000。

再来求取阶码。现在指数为 7，而单精度浮点数规定指数的偏移量为 127(注意，不是前面移码描述中所提到的 128)，即在指数 7 上加 127。那么，$E=7+127=134$。则指数的移码表示为 10000110。

最后，得到$(176.0625)_{10}$的单精度浮点数表示：

0 10000110 01100000001000000000000

6. BCD 码

转换十进制数为其等值的二进制数称作编码。前面提到的二进制数称作纯二进制码。计算机只能识别用高、低电平表示的 0 或 1。对计算机来说，用纯二进制码是十分方便的。但人们则更习惯于十进制数。为此，发明了用二进制编码来表示的十进制数，有以下两种表示方法。

(1) 8421 码

BCD 码中的 8421 码是用 4 位二进制数表示 1 位十进制数，它们的对应关系如表 0.2 所示。