高等院校教材

微型计算机原理与接口技术基础教程

谢瑞和 等编著

科学出版社
www.sciencep.com

高等院校教材

微型计算机原理与接口技术基础教程

谢瑞和等 编著

科学出版社

北京

内 容 简 介

本教材以奔腾系列微处理器为背景，全面讲授32位微型计算机的组成原理与接口技术，抛弃了传统教材中早已过时的陈旧内容与结构体系。首先讲述了奔腾系列微处理器的基本结构与CPU寄存器、PC主板结构、IA指令系统、汇编语言程序设计与开发调试；然后介绍了PC存储器与奔腾微处理器的总线信号和时序、Cache、基本I/O接口、中断技术、当代PC的串行接口、并行接口、USB接口与PCI扩展总线、键盘、显示器、ADC与DAC等用户接口电路设计；最后描述了奔腾系列PC的实模式、保护模式、MMX、超标量流水线、动态执行、分支预测、条件传送与特殊寄存器读/写等一系列增强技术。

全书集编著者20多年从事微机技术应用的教学、科研与写作经验，内容丰富、层次分明、语言精炼流畅，概念清晰，便于读者自学。本书可作为高等院校理工类相关专业“微机原理与接口技术”类课程的通用教材，对于广大的工程设计与科技人员也是一本值得推荐的好参考书。

图书在版编目(CIP)数据

微型计算机原理与接口技术基础教程/谢瑞和等编著．—北京：科学出版社，2005

高等院校教材

ISBN 7-03-015893-8

Ⅰ．微… Ⅱ．谢… 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(2005)第077033号

责任编辑：马长芳/责任校对：陈丽珠

责任印制：钱玉芬/封面设计：陈 静

科学出版社出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

新蕾印刷厂印刷

科学出版社发行 各地新华书店经销

*

2005年8月第 一 版 开本：B5(720×1000)

2005年8月第一次印刷 印张：22 3/4

印数：1—3 000 字数：440 000

定价：30.00元

（如有印装质量问题，我社负责调换〈环伟〉）

前　言

本书试图以当代 Pentium 系列替代传统的 8086 体系组织教学，抛弃了同类教材中近 20 年来基本维持不变的陈旧内容与结构体系，在内容的选材与安排上颇具特色。全书以奔腾系列为主线来介绍微型计算机的组成原理与接口技术，根据由浅入深的原则，按微型计算机原理基础、基本接口技术与增强特性三个阶段组织教材。

基础部分主要讲述了奔腾系列微处理器的基本结构、CPU 寄存器、PC 主板结构、IA 指令系统、汇编语言程序设计与汇编器开发调试的基本方法。

接口部分首先系统地介绍了 PC 存储器与奔腾微处理器的总线信号和时序、Cache、基本 I/O 接口、中断系统、PCI 扩展总线标准；然后分章详细地叙述了当代 PC 的串行接口、并行接口、USB 接口以及键盘、显示器、ADC 与 DAC 接口电路的设计。考虑到 PC 的兼容特性与国内教学现状，对 ISA 扩展总线、8255A、8254、8259、8250 等可编程接口芯片的接口特性与应用范例也进行了较详细的讨论。

提高部分较全面地描述了奔腾系列微型计算机的操作模式，尤其对保护模式寻址进行了详细的解析；然后精辟地介绍了奔腾系列的 MMX、超标量流水线、动态执行、分支预测、条件传送与特殊寄存器读/写等一系列增强技术。

本书的教授学时为 48～64 学时，书中有部分章节标记为“*”号，根据专业与学时的差别，这些内容可以留给有兴趣的学生自学钻研。

本书集编著者 20 多年从事微机技术应用的教学、科研与写作经验，在内容层次、语言表达以及概念描述等诸方面都力争便于读者自学。其编写目标是试图用作高等院校理工类相关专业《微机原理与接口技术》课程的通用教材。

参加本书编著的有谢瑞和、杨明、翁虹、张士军、左冬红、董毅、马爱梅、王红、林卫、程世平等多年主讲微机原理与接口技术以及相关实验课的教师。周丽军、何博与张金等研究生为本书的例题程序和接口技术实例做了大量开发工作，科学出版社对本书的出版给予了全力支持，付出了辛勤的劳动，在此一并致谢。

由于我们组织 32 位微机原理与接口技术教学的时间还不长，积累的经验还不多，编著者个人学识很有限，书中难免会有不少错误或不妥之处，诚心地期待同行与读者批评赐教。

作　者

2005 年 5 月于华中科技大学

目　录

第 1 章　计算机运算基础

计算机是用来进行各种信息处理的工具，尽管这些被处理的信息千差万别，但它们都是以数据的形式由计算机来操作的，而且通常都是以二进制为基础的。本章简要地概述了计算机中使用的数制系统及其几种常用的编码，可能多数读者对这些内容并不陌生，但作为计算机的运算基础来温习一遍，有助于更好地理解计算机内部操作的机理，提高学习后续章节内容的效率。

1.1　无符号数

无论计算机如何发展，它的内部操作总是基于由“0”与“1”组成的二进制数制，换言之，计算机实质上只能识别出二进制的两个数位，只是由不同的二进制位串可以编制出由它执行的各种不同的复杂操作而已。

1.1.1　二进制与十六进制

1. 二进制数

二进制数的基数为 2，逢二进一，它的多项式表示为

$$b_n\times 2^n+ b_{n-1}\times 2^{n-1}+\cdots+b_1\times 2^1+b_0\times 2^0+ b_{-1}\times 2^{-1}+b_{-2}\times 2^{-2}+\cdots$$

二进制整数部分的位权从小到大依次为 $2^0,2^1,2^2,2^3,2^4,2^5,2^6,2^7,\cdots$，亦即为十进制的 1,2,4,8,16,32,64,128,256,…。二进制小数部分的位权从大至小依次为 $2^{-1},2^{-2},2^{-3},2^{-4},\cdots$，亦即为十进制的 1/2,1/4,1/8,1/16,…。它的系数只有“0”与“1”两个数字，例如

$$(1101.11)_2=1\times 2^3+1\times 2^2+0\times 2^1+1\times 2^0+1\times 2^{-1}+1\times 2^{-2}$$

可见，它等于十进制的 13.75 。

为区别起见，规定二进制数以英文字母“B”(Binary)为后缀标记，如 1103.11B。

为方便起见，在二进制中经常使用 K (Kilo)、M (Mega)、G (Giga)等作为计量单位，应注意它们同十进制表示的数有所差异。

$$1\mathrm{K} = 2^{10} = 1\ 024$$

$$1\mathrm{M} = 2^{20} = 1\ 048\ 576$$

$$1\mathrm{G} = 2^{30} = 1\ 073\ 741\ 824$$

2. 十六进制数

十六进制数的基数是16，逢十六进一，它的多项式表示法为

$$h_n \times 16^n + h_{n-1} \times 16^{n-1} + \cdots + h_1 \times 16^1 + h_0 \times 16^0 + h_{-1} \times 16^{-1} + h_{-2} \times 16^{-2} + \cdots$$

十六进制整数部分的位权从小到大依次为 $16^0,16^1,16^2,16^3,16^4,\cdots$，亦即十进制的1,16,256,4096,65536,…。二进制小数部分的位权从大到小依次为 $16^{-1},16^{-2},16^{-3},16^{-4},\cdots$，亦即十进制的1/16,1/256,1/4096,1/65536,…。它的系数有16种数字，前10个为十进制数字0～9，后六个为英文字母A,B,C,D,E,F，分别为十进制数的11,12,13,14,15。例如

$$(89AB.4)_{16}=8\times16^3+9\times16^2+10\times16^1+11\times16^0+4\times16^{-1}$$

可见，它等于十进制的35 243.25。

为区别起见，规定十六进制数以英文字母“H”(Hexadecimal)为后缀标记，如89AB.4H。必须注意的是，为了与非数字含义的字符串相区别，当十六进制数的最高位为A～F中任何一个数字时，应在最前面加一位数字“0”，例如，ABCD3456H是非法的，而应改写为0ABCD3456H；当最高位为数字1～9时，则默认为是数据而非字符串。例如，9FEDCH是合法的。

顺便说明的是，在微型计算机的数据表示格式中，不带后缀标记或者带后缀“D”的数将默认为十进制数，例如$(254)_{10}$记为254D或254。

表1.1列举了部分二进制数、十六进制数与十进制数。这里将所有的位都用来表示数值，所以是无正负符号位的数。

表1.1　无符号二进制、十进制和十六进制的对照

二进制	十进制	十六进制	二进制	十进制	十六进制	二进制	十进制	十六进制
00000000B	0	00H	00001000B	8	08H	00010000B	16	10H
00000001B	1	01H	00001001B	9	09H	00010001B	17	11H
00000010B	2	02H	00001010B	10	0AH	00010010B	18	12H
00000011B	3	03H	00001011B	11	0BH	00010011B	19	13H
00000100B	4	04H	00001100B	12	0CH	00010100B	20	14H
00000101B	5	05H	00001101B	13	0DH	00010101B	21	15H
00000110B	6	06H	00001110B	14	0EH	00010110B	22	16H
00000111B	7	07H	00001111B	15	0FH	00010111B	23	17H

在计算机中将8位二进制数称为一个字节(Byte)，在数据的存储与处理中往往以字节为基本单元。因此，

对于8位无符号的二进制数，所能表示的十进制数范围是0～255。

对于 16 位无符号的二进制数，所能表示的十进制数范围是 0～65535。

对于 32 位无符号的二进制数，所能表示的十进制数范围是 0～4294967295。

1.1.2　数制转换

上述二进制数与十六进制数的介绍中就已经描述了将它们转换成十进制数的方法，这种方法也适用于其他任何非十进制数，亦即只需将非十进制数以多项式的形式展开表示，然后计算它们的十进制之和即可将其转换为十进制数。

下面讨论十进制数转换成非十进制数以及二进制数同十六进制数之间的相互转换。

1. 十进制数转换为非十进制数

将十进制数整数部分转换成其他进制，只需将整数除以该进制的基数，取其余数按“后高前低”的顺序排列作为转换结果。将十进数小数部分转换成其他进制，则应将该进制的基数乘以十进制数小数部分，然后取乘积的整数按“前高后低”的顺序排列作为转换结果。

(1) 十进制数转换为二进制数，采用除 2 取余法，即将十进制整数除以 2，得到一个商数和余数，再将商数除以 2 又得到一个商数和余数，如此继续除下去直至除尽为止。以最后所得的商数“1”作为最高位，将各次所得的余数按“后得先排”的顺序写下来即得二进制转换结果。

【例 1.1】 100＝(?)B

转换过程见图 1.1。

除数	被除数/商数	余数	
2	100		
2	50	0	
2	25	0	最低位 ↑
2	12	1	
2	6	0	
2	3	0	次高位
	1	1	

图 1.1　除 2 取余法

结果得 100＝1100100B。

(2) 同样的方法可将十进制数转换成十六进制数，采用除 16 取余法，即将十进制整数除以 16，得到一个商数和余数，再将商数除以 16 又得到一个商数和余

数，如此继续除下去直至除尽为止。以最后所得的商数作为最高位，将各次所得的余数按“后得先排”的顺序写下来即得十六进制转换结果。

【例 1.2】 35243＝(?)H

转换过程见图 1.2。

结果得 35243＝89ABH。

除数	被除数/商数	余数	
16	35243		最低位
16	2202	11	↑
16	137	10	↑
	8	9	次高位

图 1.2　除 16 取余法

(3) 将十进数小数部分转换成二进制小数，采用乘 2 取整法，即将十进制纯小数乘以 2，摘除乘积中的整数后保留小数部分再乘 2，如此继续下去直至乘积小数部分为零或者得到要求的精度为止。将各次摘取的整数依先后顺序写出来即为转换的二进制纯小数结果。

【例 1.3】 0.1875＝(?)B

转换过程见图 1.3。

	积的整数部分	被乘数/积的小数部分	乘数
		0.1875	2
高位 ↓	0	0.375	2
	0	0.75	2
低位	1	0.5	2
	1	0.0	

图 1.3　乘 2 取整法

结果得 0.1875＝0.0011B。

(4) 将十进数小数部分转换为十六进制小数，采用乘 16 取整法，即将十进制纯小数乘以 16，摘除乘积中的整数后保留小数部分再乘 16，如此继续下去直至乘积小数部分为零或者得到要求的精度为止。将各次摘取的整数依先后顺序写出来即为转换的二进制纯小数结果。

【例 1.4】 0.78125＝(?)H

转换过程见图 1.4。

	积的整数部分	被乘数/积的小数部分	乘数
高位		0.78125	16
	12	0.5	16
低位	8	0.0	

图 1.4　乘 16 取整法

结果得 0.78125＝0.C8H。

2. 十六进制数与二进制数之间的转换

由于一位十六进制数实际上是 4 位二进制数，所以两者之间的转换是轻而易举的。将二进制数转换成十六进制数时，以小数点为界，整数部分自右至左每 4 位一组，最高位部分不足 4 位时在左边补 0，每组用对应的一位十六进制数表示；而小数部分则自左至右每 4 位一组，最低位部分不足 4 位时在右边补 0，每组用对应的一位十六进制数表示。例如

10 1101 1010 0013.0110 11 B

＝ 2　D　A　3 . 6　C H

注意，不要将十六进制数小数点最后一位“C”误写为“3”。

由此可见，引入十六进制数的主要目的是为了免除书写与阅读一长串二进制数码的麻烦，克服容易出错之弊病，计算机本身并不需要做转换运算。

如果要将十六进制数转换成二进制数，只需将十六进制数的每一位用对应的 4 位二进制数表示并依原顺序排列即可。例如

5　F　8　4 .　E　4 H

＝ 0101 1111 1000 0100.1110 0100 B

1.2　常用的编码

计算机总是以量化的数据形式来进行操作的，但它处理的对象不一定都是“数”，例如我们现在阅读的就是“字符”。因此采用一些标准的二进制编码来表示部分常用的对象或信息就会方便得多。这里介绍最常用的 BCD 码与 ASCII 码。

1.2.1　BCD 码

人们已经习惯了十进制数，而且计算机的原始数据多数也是十进制的，但十进制数不能直接在计算机中进行处理，必须用二进制为它编码，这样就产生了二进制

编码的十进制数，简称 BCD(Binary Coded Decimal)码。

BCD 码用 4 位二进制数表示一位十进制数，但这 4 位二进制数中可表示的 16 个数码中有 6 个数码是多余的，应该抛弃，可以使用不同的方法来处理这些数码，因而产生了各种不同的 BCD 码，但最通用的是 8421 BCD 码，它是将十六进制数的 A～F 放弃不用。例如

89 的 BCD 码为 1000 1001

105 的 BCD 码为 0001 0000 0101

2004 的 BCD 码为 0010 0000 0000 0100

可见，BCD 码是很容易编制的，而且用它来表示十进制数就很直观，但是一定要区别于二进制数，两者表征的数值完全不同，例如

$(0010000000000001.1001)_{BCD}=2001.9$

0010000000000001.1001B＝8193.5625

以上用 4 位二进制表示 1 位十进制的编码称为紧(压)缩型 BCD 码(Packed BCD)，此时用 8 位二进制就能表示 2 位十进制。将一个字节分成高半字节与低半字节，各表示一位十进制数，CPU 通过对一个专门针对半字节的辅助进位的调整就可对 BCD 编码数据直接进行运算。

如果用 8 位二进制来表示 1 位十进制，则称为非紧(压)缩型 BCD 码(Unpacked BCD)，此时它的高 4 位全为 0。例如，86 的紧缩型 BCD 码是 1000 0110B，它的非压缩型 BCD 码是 0000 1000 0000 0110B。

1.2.2　ASCII 码与奇偶校验

计算机不仅仅只对数据进行运算，而且还要处理其他各种事务，因而它应能识别字母与各种字符，目前普遍使用 ASCII 码来表示字母与字符。

ASCII(American Standard Code for Information Interchange)码，即美国标准信息交换码，这是一种 7 位的二进制编码，可表示 128 个可打印字符码和控制码，其中包括英文大小写字母与数字 0～9。表 1.2 列出了标准 ASCII 码及其表示方法。

由于存储器的基本单元为 8 位一个字节，故通常还是用一个字节来表示 ASCII 码，并认为最高位 b_7 恒为零，于是 0～9 的 ASCII 码为 30H～39H，大写英文字母 A～Z 的 ASCII 码为 41H～5AH，小写英文字母 a～z 的 ASCII 码为 61H～7AH 等。

对于 0～9 的数字而言，ASCII 码是将非压缩 BCD 码的高半字节变为“3”，其转换简单而方便，例如 18 的非压缩 BCD 码是 0000 0001 0000 1000，而它的 ASCII 码是 0011 0001 0011 1000。因此某些科技文献又称非压缩 BCD 码为 ASCII BCD 码。