普通高等教育“十一五”国家级规划教材

微型计算机原理与接口技术

WEIXING JISUANJI YUANLI YU JIEKOU JISHU

第5版

周荷琴　冯焕清／编著

技术大学出版社

• 普通高等教育"十一五"国家级规划教材
中国科学院指定考研参考书

微型计算机原理与接口技术

• 第5版 •

周荷琴　冯焕清　　编著

中国科学技术大学出版社

• 合　肥 •

内 容 简 介

本书是为中国科学技术大学工科电子类专业本科生学习“微型计算机原理与系统”课程编写的教材，是作者在参考了国内外大量文献、资料的基础上，吸取各家之长，并结合多年教学和应用研究的经验，精心组织编写而成的。全书内容丰富，图文并茂，讲述深入浅出，通俗易懂，并附有大量的实例和习题，部分习题还给出了解题提示，既可用作教材，也适合于自学，被列入“普通高等教育‘十一五’国家级规划教材”和“中国科学院指定考研参考书”。

全书15章，内容安排上注重系统性、先进性和实用性。前5章是基础部分，主要介绍8086微型机系统的组成原理、体系结构、指令系统、汇编语言程序设计方法以及存储器的原理和电路设计。第6～12章讨论接口和总线技术，包括中断、DMA和I/O接口以及几个典型的大规模集成电路接口芯片(8255A、8253/8254、8259A、8251A、8237A)，A/D和D/A以及总线技术也被纳入其中。最后3章介绍高档微型机的工作原理，其中第13章包括32位微处理器的结构和工作模式、寄存器组成、保护模式下的内存管理、保护模式下的中断和异常以及任务切换等内容，第14章介绍32位机新增指令、浮点数、SIMD技术和指令系统，并给出了许多编程实例，第15章简要介绍了PC/XT机的系统结构，主要对32位微型机的结构以及64位CPU和智能多核处理器进行了讨论，并概要阐述了64位机的系统结构和技术特点。

本书可作为高等学校电子类专业和其他相近相关专业本科教育的教材，也可作为从事微型计算机系统设计和应用等相关科技工作者的参考书。

图书在版编目(CIP)数据

微型计算机原理与接口技术/周荷琴，冯焕清编著. —5版. —合肥：中国科学技术大学出版社，2013.3

普通高等教育“十一五”国家级规划教材

ISBN 978-7-312-03191-5

Ⅰ.微… Ⅱ.①周… ②冯… Ⅲ.①微型计算机—理论—高等学校—教材 ②微型计算机—接口技术—高等学校—教材 Ⅳ.TP36

中国版本图书馆CIP数据核字(2013)第030916号

责任编辑：张善金

出 版 者：中国科学技术大学出版社

地　　址：安徽省合肥市金寨路96号　　**邮编**：230026

网　　址：http://press.ustc.edu.cn

电　　话：发行部　0551-63606086-8810

印 刷 者：合肥学苑印务有限公司

发 行 者：中国科学技术大学出版社

经 销 者：全国新华书店

开　　本：787mm×1092mm　1/16

印　　张：33

字　　数：860千

版　　次：1996年9月第1版　2013年3月第5版

印　　次：2013年3月第30次印刷

印　　数：277 001—288 000册

定　　价：42.00元

前　言

本书是为中国科学技术大学工科电子类本科生学习“微型计算机原理与系统”课程编写的教材，被列入“普通高等教育‘十一五’国家级规划教材”和“中国科学院指定考研参考书”。自1996年出版以来，本书被很多高校选用，得到了广大读者和同行老师的热情支持，并反馈回不少宝贵的意见和建议，在此谨表示感谢！为适应微型计算机技术飞速发展的形势和教育教学改革不断深化的需要，我们再次对原书进行了全面的修订。

自20世纪70年代第一代微型计算机问世以来，计算机技术以惊人的速度发展，涌现了数十个品种几百个型号的微处理器，数据宽度从8位、16位、32位发展到了64位，处理器芯片的CPU核心发展到了双核乃至4核、6核和8核，当前微型计算机的发展已经进入了智能多核时代。希望通过这次修订，本书能更系统归纳和清晰展示已经发展了40多年的计算机高新技术，能更深入浅出地讲清楚那些看似深奥的计算机知识，从而真正有助于教师们的课堂教学和学生们的课后阅读。通过努力我们顺利完成了修订工作，新版将以全新的面貌与读者见面。

全书共分15章，内容包括3部分：第1～5章是基础部分，仍以8086为主要对象，包括绪论、8086 CPU、寻址方式、指令系统、汇编语言程序设计和存储器。第6～12章讨论了接口和总线技术，包括中断、DMA和I/O接口以及8255A、8253/8254、8259A、8251A、8237A等典型的可编程接口芯片，A/D和D/A以及总线技术也被纳入其中。第13～15章介绍了高档微型机的原理，包括32位微型机的工作原理、指令系统与程序设计以及64位CPU和智能多核处理器，也包括16位、32位和64位机的系统结构和技术特点。

本次修订的主要内容包括：

1. 对第1、2、4、5、8、12、15章的全部内容以及第6、13、14章的部分内容重新进行了组织和编写，尽可能删繁就简，循序渐进，将现今最新的技术包含进来，表述上没有冗余，文字简练顺畅，全书一气呵成。

2. 存储器一章不仅包含了最新的存储器技术，还对目前在嵌入式系统领域广泛应用的串行EEPROM作了较详细的介绍；考虑到Cache技术在现代CPU技术中的地位越来越重要，对其工作原理的讨论也比较深入。

3. 将I/O接口技术与并行接口芯片8255合并形成了第6章，并将几个简单接口芯片的介绍前移到第2章。

4. 总线技术一章对已淘汰的总线标准只做了简单回顾，重点讨论了当前流行的系统总线PCI、PCI Express以及串行总线USB和IEEE 1394。

5. 第13章32位微型计算机原理，对保护模式下的中断和异常进行了较详细的描述，并增加了任务切换的内容。

6. 第14章32位微型计算机的指令系统与程序设计，在原有的32位微型计算机新增指令和程序设计的基础上，增加了IEEE标准754浮点数的表示方法、奔腾处理器的SIMD技术、SIMD指令系统、SIMD程序设计实例等内容。

7. 第15章在原有PC/XT机以及32位微型计算机的系统结构基础上，增加了64位CPU和多核处理器的内容，并着重介绍了64位微型机的系统结构、芯片组和主板，内容涉及最新的i7、i5和i3智能多核处理器。

8. 对第4章和第14章中的16位和32位微型计算机的所有汇编语言程序实例都进行了精心设计，并全部通过了上机验证。

9. 对原书保留章节中的绝大部分插图进行了修正。

10. 修改了原书保留章节中的部分内容和习题，并为许多习题增加了解题的提示信息，读者根据例题和提示信息，不难得出正确答案。

吴秀清教授为本书此前的版本做出了重要贡献，特此感谢！

在本书撰写过程中，青年教师李峰、何力以及研究生潘剑锋、刘冰啸、乔赫元、袁非牛、王鹏、刘勃、刘学亮、王恒良、郭永刚、赵何、黄庆华、卢胜、陈立群、梅涛、武海澄、陈功等在资料的收集、例题的验证、程序的调试、插图的绘制、多媒体课件的制作等方面做了许多工作，并对书中的内容提出了不少有益的建议，在此一并表示衷心的感谢！

此外，在本书编写过程中，我们查阅、参考了大量国内外相关的书籍和文献以及网上资料，在此特向这些文献的作者表示深切的感谢！

由于作者水平有限，错误和不当之处在所难免，敬请读者批评指正，以便日后再版时予以修正。

编　　者

2013年1月于合肥

目　录

第1章 绪 论

本章首先介绍计算机中数的表示方法，然后就计算机的基本结构、大致的工作过程和计算机软件等内容展开讨论，接着阐述微型计算机的结构和微型计算机系统，使大家对 CPU、存储器、接口等概念有一个初步的了解，最后简要讲述计算机和微型计算机的发展概况。

1.1 计算机中数的表示方法

1.1.1 进位计数制

进位计数制是指用一组固定的数字符号和特定的规则来表示数的方法。在日常生活中，数多用 10 进制来表示，这是大家所熟悉的。有时也用别的进制来表示数，例如，表示时间的时、分、秒之间是 60 进制，而小时与天之间为 24 进制。在微型计算机中，则采用只有 0 和 1 两个数字的二进制来表示数。这是因为计算机是一种电子设备，由大量只能识别电信号的逻辑部件组成，可以用电平的高低、开关的通断、晶体管的导通和截止来表示数字 0 和 1，运算规则简单，使用方便可靠。人们经常使用的字母、符号和图形等，在计算机中也一律用二进制编码来表示。但二进制数的数位太长，不易书写和记忆，而人们又习惯于使用 10 进制数，因此在计算机领域里使用多种进位制来表示数，常用的有二进制、10 进制和 16 进制的表示方法。

1. 10 进制数(Decimal)

10 进制数具有 10 个不同的数字符号 0～9，其基数为 10，各位的权值为 10^i，其实际值可按权展开后相加获得。在 10 进制数字后面可以加后缀 D，表示该数是 10 进制数，但 D 通常省略不写。例如，10 进制数

$$347D=347=3\times10^2+4\times10^1+7\times10^0$$

2. 二进制数(Binary)

它只有 0 和 1 两个数字，其基数为 2，各位的权值为 2^i。表示二进制数时，后面必须加后缀 B。例如，二进制数

$$10110B=1\times2^4+0\times2^3+1\times2^2+1\times2^1+0\times2^0=22$$

3. 16 进制数(Hexadecimal)

它由 0～9、A、B、C、D、E、F 共 16 个数字和字母组成，其基数为 16，各位的权值为 16^i。表示 16 进制数时，后面必须加后缀 H。字母 A～F 分别表示 10 进制数的 10～15。例如，16 进制数

$$3A0FH=3\times16^3+10\times16^2+0\times16^1+15\times16^0=3\times4096+10\times256+15=14863$$

每个 16 进制数字都可用 4 位二进制数来表示，见表 1.1，如 0AH＝1010B，0FH＝

1111B。由于16进制数与二进制数之间转换很方便，数位长度又只有二进制数的1/4，因此，在编写汇编语言程序和打印程序清单时，广泛使用16进制数。在计算机中表示存储器的地址和存放的数据时，通常也都用16进制数来表示。

表1.1　10进制、二进制、16进制数、BCD码的关系

10进制数	二进制数	16进制数	BCD码
0	0000	0	0000
1	0001	1	0001
2	0010	2	0010
3	0011	3	0011
4	0100	4	0100
5	0101	5	0101
6	0110	6	0110
7	0111	7	0111
8	1000	8	1000
9	1001	9	1001
10	1010	A	0001　0000
11	1011	B	0001　0001
12	1100	C	0001　0010
13	1101	D	0001　0011
14	1110	E	0001　0100
15	1111	F	0001　0101

4. 8进制数(Octal)

8进制数由0～7共8个数字组成，其基数为8，各位的权值为8^i。表示8进制数时，后面必须加后缀O或Q。例如，8进制数

$$753Q=7\times8^2+5\times8^1+3\times8^0=491$$

每位8进制数可以由3位二进制数组成，例如，627Q=110 010 111B，它与二进制数之间的转换十分方便。在20世纪70～80年代使用的小型计算机系统上，普遍采用8进制数编写汇编语言程序和打印程序清单。随着微型计算机技术的普及，16进制计数法被用户广泛采用，微机中已不再采用8进制计数法了。

5. 不同进制数之间的转换

由上面的例子可以看到，将二进制或16进制数转换成10进制数时，只要将当前位数值乘以该位的权值，再将各部分相加即可。如果要将10进制数转换成二进制数就有些麻烦，可以采用除以2求余数的方法来得到。

例1.1　将10进制数25转换成二进制数，先用2去除，得到的商为12，第一个余数$K_0=1$，此位为结果的最低位。再将12除以2，得到第二个余数$K_1=0$……如此不断进行下去，直到商等于0为止，最后一次得到的余数为结果的最高位。这样，可以得到25=11001B。

$$
\begin{array}{r|l}
2 & 25 \\
\hline
2 & 12 \quad \text{余数}=1(K_0) \quad \text{最低位} \\
\hline
2 & 6 \quad \text{余数}=0(K_1) \\
\hline
2 & 3 \quad \text{余数}=0(K_2) \\
\hline
2 & 1 \quad \text{余数}=1(K_3) \\
\hline
 & 0 \quad \text{余数}=1(K_4) \quad \text{最高位}
\end{array}
$$

根据表1.1,将二进制数转换成16进制数就很容易。例如,1000 1010B=8AH。这里要说明的是由于二进制数的数位较长,不便于阅读,所以本书在后面书写二进制数时,从最低位起,每隔4个二进制数,前面加一个空格。但当二进制数在程序中出现时,这个空格必须除去,否则送到计算机中进行运算时就不正确了。

例如,把二进制数1011010011011001B写成1011 0100 1101 1001B,它等于B4D9H。

6. 位、字节、字和字长等数据单位表示

位(Bit):在计算机中,二进制数的每一位(0或1)是组成二进制信息的最小单位,称为1个比特,简称位,它是数字系统和计算机中信息存储、处理和传送的最小单位。

字节(Byte):8个二进制信息组成的一个单位称为1个字节,1 Byte=8 Bit。

字(Word):一个字由16位二进制数也即两个字节组成。一个16位的字D_{15}~D_0可分为高字节和低字节两个部分,其中D_{15}~D_8为高字节,D_7~D_0为低字节。

字长(Word Length):计算机中还用字长这个术语来表示数据,字长决定了计算机内部一次可以处理的二进制代码的位数,它取决于计算机内部的运算器、通用寄存器和数据总线的位数。根据计算机的字长不同,可将计算机分为8位机、16位机、32位机和64位机等不同的机型。显然,计算机的字长越长,一次能同时传送和处理的数据就越多,运算速度越快,精度也越高,但制造工艺就越复杂。

1.1.2 二进制编码

在计算机中,数都采用二进制的形式来表示,因此各种数字、英文字母、运算符号等,都要采用若干特定的二进制码的组合来表示,这就是二进制编码。最常用的编码有BCD码和ASCII码两种。

1. 8421 BCD码(Binary Coded Decimal)

虽然计算机中的数都采用二进制表示,但是二进制数不仅数位长,而且不直观,所以常采用BCD码来表示。BCD码有0,1,2,…,9共10个不同的数字符号,也是逢10进1,但它的每一位数字都用4位二进制来表示,所以称为二进制码的BCD数,它是一种很直观的编码。4位二进制数可表示0000~1111共16种码,取前10个码作为BCD码。由于它4位的权值分别是8、4、2、1,因此也称为8421 BCD码。

例1.2 用8421 BCD码表示10进制数327。

$$(327)_{10}=(0011\quad 0010\quad 0111)_{\text{BCD码}}$$

反之,由BCD码也很容易求得它表示的10进制数。

例1.3 $(1001\quad 0101\quad 1000)_{\text{BCD码}}=958$

10进制数、二进制数、16进制数和2-10进制BCD码之间的关系如表1.1所示。

BCD码既照顾了人们使用10进制数的习惯，又考虑了计算机的特点，确实很好，但运算起来有点麻烦。由于计算机中有专门的调整电路对它自动进行处理，所以实际应用中经常采用这种编码。

2. ASCII码

除了10进制数外，各种数字、字母及字符也必须用二进制编码来表示后，计算机才能进行处理。最常用的是ASCII码，即美国标准信息交换码(American Standard Code for Information Interchange)。它用7位代码(00～7FH)来表示计算机中存储的字母、数字及符号，共可表示128个字符。如数字0～9，字母A～Z，a～z等，其中数字0～9的ASCII码为30H～39H，字母A～Z的ASCII码为41H～5AH。键盘键入的数、字母以及送到CRT显示的字符都必须用ASCII码表示。ASCII码表见附录B。一些控制字符也列入表中，它们在计算机中实现某些控制功能，如CR、LF和BEL分别表示回车、换行和响铃，STX、ETX、ENQ等用于串行异步通信。

1.1.3 带符号数的表示方法

上面介绍的二进制数没有提到符号问题，因此是一种无符号数。但在计算机中，数有正、负之分，怎么来表示符号呢？通常用最高位作为符号位。若一个数的长度为8位(D_7～D_0)，则用D_7位作符号位，$D_7=1$，表示是负数；$D_7=0$，表示是正数。

例1.4 0101 1101B=+93

1101 1101B=−93

连同符号位在一起作为一个数称为机器数，它表示的实际数值称为机器数的真值。上面两个式子中，等式左边的为机器数，右边的为真值。超过8位的数可以用16位二进制数(D_{15}～D_0)来表示，这时，$D_{15}=1$，表示负数；$D_{15}=0$，表示正数。

为了运算方便，机器数通常有三种表示方法：原码、反码和补码。

1. 原码

正数的符号位用0表示，负数用1表示，其余位为数值，这种表示方法称为原码。

例1.5 X=+105，$[X]_{原}$=0110 1001B

X=−105，$[X]_{原}$=1110 1001B

原码简单易懂，与真值的换算也很方便，但若要进行两个异号数相加或者两个同号数相减的运算，就要做减法操作。然而在一般的计算机中是没有减法运算部件的，减法运算也要用加法部件实现，所以要引进反码和补码。

2. 反码

正数的反码与原码相同，最高位为符号位，用0表示，其余位为数值。

例1.6 $[+4]_{反}$=0000 0100B

$[+31]_{反}$=0001 1111B

$[+127]_{反}$=0111 1111B （最大值）

负数的反码为它的正数按位取反，即连同符号位一同取反。

例1.7 $[-4]_{反}$=1111 1011B

$[-31]_{反}$=1110 0000B

$[-127]_{反}=1000\ 0000B$ （最小值）

所以，8位二进制数表示的反码范围为$-127\sim+127$。当带符号数用反码表示时，最高位为符号位，当它为正数时，后7位为真正的值，当它为负数时，后7位要取反后才能得到真正的值。

例1.8 $[X]_{反}=1001\ 0100B$

$[X]_{真值}=-[110\ 1011]=-107$

3. 补码

正数的补码表示与原码相同，最高位为符号位，用0表示，其余位为真值，负数的补码最高位为1，数值部分则由它的反码再加1形成。

例1.9 $[+4]_{原}=0000\ 0100=[+4]_{反}=[+4]_{补}$

$[-4]_{原}=1000\ 0100B$

$[-4]_{反}=1111\ 1011B$ （正数按位取反）

$[-4]_{补}=1111\ 1100B$ （反码+1）

$[+127]_{原}=0111\ 1111B=[+127]_{反}=[+127]_{补}$

$[-127]_{原}=1111\ 1111B$

$[-127]_{反}=1000\ 0000B$

$[-127]_{补}=1000\ 0001B$

$[-128]_{补}=1000\ 0000B$

8位二进制数能表示的补码范围为$-128\sim+127$，可以推算出16位二进制数能表示的二进制补码的范围为$-32768\sim+32767$。当带符号数用补码表示时，最高位是符号位，当符号位是0时，表示正数，后7位为其真正的数；当符号位是1时，表示负数，要将后7位的最低位减1，求得反码，再按位取反，才能得到真正的数(真数)。

例1.10 若已知$[X]_{补}=1001\ 0100B$，求X的反码和原码。

$[X]_{反}=[X]_{补}-1=1001\ 0100B-1=1001\ 0011B$

$[X]_{原}=1110\ 1100B$

$X=-110\ 1100B=-(64+32+8+4)_{10}=-108$

引进补码后，可以将补码连同符号位一起看作一个数，各位的权都是2^i，但最高位为0时，表示正数，为1时表示负数，其余为数值。

例1.11 $[-128]_{补}=1000\ 0000B=-128+0$，最高位的权为$2^7=128$，同样，

$[-4]_{补}=1111\ 1100B$

$=-128+(64+32+16+8+4)=-4$

$[+4]_{补}=0000\ 0100B=4$

所以，任何一个数用补码表示后，都可以看成“连同符号位的数”。符号位也一起参加运算，一个数要减去另一个数时，只要加上其补码即可。

例1.12 做减法运算，求$7-19=?$ 可以用$7+[-19]_{补}$来完成。

$[7]_{补}=0000\ 0111B$

$[+19]_{补}=0001\ 0011B$

$[-19]_{补}=1110\ 1101B$

$$\begin{array}{rl} & 0000\ 0111\mathrm{B} \quad [7]_{补} \\ + & 1110\ 1101\mathrm{B} \quad [-19]_{补} \\ \hline & 1111\ 0100\mathrm{B} = \mathrm{F4H}(和) \end{array}$$

所以，和的补码＝F4H＝1111 0100B

和的反码＝F3H＝1111 0011B

和的原码＝1000 1100B，其真值为－12

可见，7＋(－19)＝－12，答案正确。

前面介绍的数虽有正、负之分，但都是整数，实际应用时会遇到实数，即带小数点的数。实数通常用浮点数来表示，我们将在第 14 章详细介绍浮点数，并举实例来说明在 32 位机中如何用浮点数来编写程序。

1.2 计算机的基本结构

世界上第一台通用可编程计算机 ENIAC(Electronic Numerical Integrator and Calculator)于 1946 年由美国宾夕法尼亚大学研制成功，它使用了 17000 多个电子管和超过 500 英里长的导线，这台庞大的计算机重量超过 30 吨，每秒只能执行 10 万次运算，但它推动世界进入了电子计算机时代。ENIAC 采用重新连接线路的方法实现编程，编程时需要对大约 6000 多个开关进行仔细的机械定位，并用转插线把选定的各个控制部件互连起来构成程序序列，这很像早期的电话接线总机，需要许多工人花几天时间才能完成，效率很低。另外，电子管的功耗大、寿命低，维护起来很麻烦。

为替代重新连接线路实现编程，产生了用于控制计算机的计算机语言，称为机器语言(Machine Language)，它由一条条 1 和 0 组成的二进制代码构成，这些代码被称为指令(Instruction)，告诉计算机要执行哪些运算和操作。这种二进制代码以指令组的形式存储在计算机中，称为程序(Program)。这种方法比通过重新连接机器线路进行编程的方法有效，但编程时要涉及许多代码，开发过程仍然非常耗时。数学家冯·诺依曼(John Von Neumann)首先开发出能接收指令，并可将指令存储到存储器中的系统，为了纪念他，常将这种计算机称为冯·诺依曼结构的机器。

半个多世纪以来，计算机技术不断发展，相继出现了各种类型的计算机，包括小型、中型、大型、巨型计算机以及广泛使用的微型计算机，它们的规模不同，性能和用途各异，但就其结构而言，都是冯·诺依曼计算机结构的延续和发展。

1.2.1 计算机的基本结构

1. 计算机的基本组成

冯·诺依曼计算机的基本组成框图如图 1.1 所示。它主要由 5 个部分组成，各部分的基本功能如下：

·存储器　用来存放原始数据、中间结果以及为使计算机能自动进行运算而编制的程序，它们均以二进制的形式存放在存储器中。

·运算器　用来执行算术运算(加、减、乘、除)、逻辑运算(与、或、非、异或)和移位等操