普通高等教育“十二五”规划教材

接口技术

主　编　李长青
副主编　孙君顶　李泉溪

中国水利水电出版社
www.waterpub.com.cn

普通高等教育“十二五”规划教材

接口技术

主　编　李长青
副主编　孙君顶　李泉溪

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书介绍了微型计算机的基本原理与常用接口的设计方法。主要内容包括微处理器结构、指令系统、存储器接口、输入/输出接口、中断技术、常用可编程接口芯片及其应用、人机交互接口以及总线技术，最后以矿井监控系统内置信息传输接口卡与监控工作站设计为例，介绍接口技术综合应用实例。

本书可作为高等学校计算机科学与技术专业、通信工程专业、网络工程专业以及自动化专业或相关专业高年级学生以及研究生的“接口技术”课程教材，对从事微型计算机应用系统设计的科技人员也是一本有价值的参考书。

图书在版编目（CIP）数据

接口技术 / 李长青主编. -- 北京 : 中国水利水电出版社, 2014.8
普通高等教育“十二五”规划教材
ISBN 978-7-5170-2143-8

Ⅰ. ①接… Ⅱ. ①李… Ⅲ. ①单片微型计算机－接口－高等学校－教材 Ⅳ. ①TP368.147

中国版本图书馆CIP数据核字(2014)第128920号

书　　名	普通高等教育“十二五”规划教材 **接口技术**
作　　者	主编　李长青　　副主编　孙君顶　李泉溪
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路1号D座　100038） 网址：www.waterpub.com.cn E-mail：sales@waterpub.com.cn 电话：（010）68367658（发行部）
经　　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　　版	中国水利水电出版社微机排版中心
印　　刷	北京瑞斯通印务发展有限公司
规　　格	184mm×260mm　16开本　20.5印张　486千字
版　　次	2014年8月第1版　2014年8月第1次印刷
印　　数	0001—3000册
定　　价	**39.00**元

凡购买我社图书，如有缺页、倒页、脱页的，本社发行部负责调换

前　言

随着大规模集成电路制造技术的迅速发展，各类高性能的外围接口芯片随之出现，这就改变了过去一些微型计算机应用系统的设计思想，使我们设计出更高性能的微型计算机应用系统成为可能。接口技术是构成微型计算机应用系统的重要组成部分，其设计的合理与优越直接影响计算机应用系统的性能。为此，结合我们多年来从事微型计算机原理和接口应用技术的研究和教学经验，同时参考了国内外大量的文献资料，编写了《接口技术》这本教材。

本书主要从微型计算机系统设计与应用角度入手，淡化了微机原理部分的相关内容。本书较系统地介绍了微型计算机接口技术的相关内容和概念，在内容的安排上，注重系统性、实用性和先进性，以达到硬件上能够组成微机应用系统，软件上能够编程应用和系统开发的目的。在讲清基本原理的基础上，辅以实例介绍，尽可能做到理论联系实际，通俗易懂。这样使本教材更适宜于本科教学。

本教材共分 10 章，第 1 章介绍了微处理器运算基础与发展概况；第 2 章以 8086 CPU 为主讲述其基本结构和工作原理，然后简洁地介绍了 80386 CPU、Pentium CPU 及最新的酷睿微处理器；第 3 章介绍了 8086 汇编语言指令系统；第 4 章介绍了存储器接口技术；第 5 章介绍了输入输出接口技术；第 6 章介绍了中断技术；第 7 章介绍了常用的可编程接口芯片及其应用技术；第 8 章介绍了常用的人机接口技术；第 9 章介绍了总线技术；第 10 章以河南理工大学课题组开发的“KJ93 型矿井监控系统”为例，介绍接口技术的综合设计与应用。

本书由河南理工大学的李长青教授任主编、孙君顶与李泉溪教授任副主编。编写分工为：李长青编写第 1、7 章和 8.1 节，李泉溪编写第 2 章，孙君顶编写第 10 章和附录，李静编写第 4、5、9 章，刘静编写第 3 章，陈锋编写第 6 章及 8.2 和 8.3 节。全书由李长青和孙君顶教授进行修改、统稿和审核。

由于编者水平有限，错误与不妥之处，敬请读者与专家指正。

编　者

2014 年 3 月于河南理工大学

目 录

第 1 章　微型计算机概述

本章主要介绍微型计算机的数字运算基础以及基本概念，如计算机运算基础、微处理器、微型计算机、微型计算机系统、总线、微型计算机的发展和分类等。

众所周知，传统的电子数字计算机由五大部分组成，即运算器、控制器、存储器、输入设备和输出设备，如图 1.1 所示。

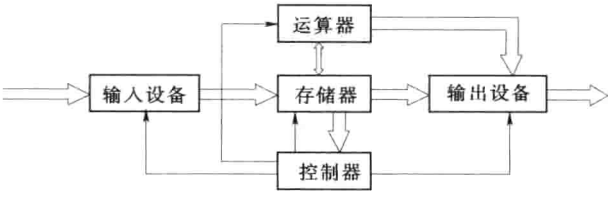

图 1.1　电子计算机的组成框图

其中：

运算器是能够完成各种运算（包括加、减、乘、除等算术运算、与或非等逻辑运算和比较等）的部件。

输入设备是用来将原始数据（包括程序）输入到存储器的部件；目前常用的输入设备有：键盘、鼠标、手写板、扫描仪、摄像头和麦克风等。

输出设备是用来将中间结果或最后结果输出的部件；目前常用的输出设备有：显示器和打印机等。

存储器是用来记录、存放原始数据（包括程序）、中间结果和运算结果的部件。

控制器是用来协调各部件工作的，发出控制命令的部件。

要想在图 1.1 所示的计算机中工作，首先需把原始数据和程序通过输入设备存入存储器里，然后操作计算机让其启动，所存的程序一条一条地被送到控制器，控制器根据每条指令的不同，发出不同的控制命令，自动进行运算，最后通过输出设备输出计算结果或将运算结果保存在存储器中。这就是迄今为止电子计算机所共同遵循的程序存储和程序控制的工作原理。这种原理称为冯·诺依曼型计算机原理。

在图 1.1 中，存在着两种信息，一种是数据，用双线表示，包括原始数据、中间结果、最终结果，以及表示程序的代码；另一种是控制命令，其流向用单线表示。不论是数据还是控制命令，在计算机中都用“0”和“1”表示的二进制数表示。

如图 1.1 所示的这五大部件是计算机的实体，统称为计算机的硬件（Hard ware）。其中存储器又分为内存储器和外存储器。外存储器、输入设备和输出设备统称为外部设备；运算器、控制器和内存储器合称为主机，而运算器和控制器两部分又合称为中央处理器 CPU（Center Processing Unit）。

1.1　计算机运算基础

计算机是用来进行各种数据运算与信息处理的工具，虽然被处理的信息千差万别，但它们都是以二进制数据的形式来进行运算操作的。本节简要地概述了计算机中使用的数制及其几种常用的编码。这些内容有助于更好地理解指令系统。

1.1.1　无符号数

不管计算机如何发展，它的内部操作运算数总是基于由“0”与“1”组成的二进制数，换言之，计算机实质上只能识别出二进制的“0”和“1”，只是由不同长度的二进制位的排列与组合可以编制出各种不同的复杂操作以及字符。

1.1.1.1　二进制数

二进制数的基数为 2，逢二进一，对于任意个具有 n 位整数 m 位小数的二进制数，它的多项式展开表示为：

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

$$=\sum_{i=-m}^{n-1}b_i\times 2^i$$

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、…亦即为十进制数的 1、2、4、8、16、32、64、128、256、…。

二进制小数部分的权位从大到小依次为 2^{-1}、2^{-2}、2^{-3}、2^{-4}、…亦即为十进制数的 1/2、1/4、1/8、1/16、…。

各位的系数只有“0”与“1”两种选择，例如：

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

求和计算后它等于十进制的 13.75。

科技文献中规定二进制数以英文字母“B”（Binary）为后缀标记，如 1101.11B。

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。

$1K=2^{10}=1024$

$1M=2^{20}=1048576$

$1G=2^{30}=1073741824$

1.1.1.2　十六进制数

十六进制数的基数是 16，逢十六进一，对于任意个具有 n 位整数 m 位小数的十六进制数，它的多项式表示为：

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

$$=\sum_{i=-m}^{n-1}h_i\times 16^i$$

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、…亦即为十进

制数的1、16、256、4096、65536、…。

十六进制小数部分的位权从大到小依次为16^{-1}、16^{-2}、16^{-3}、16^{-4}、…亦即为十进制数的1/16、1/256、1/4096、1/65536、…。它的位系数有16种数字，前10个为十进制数字0～9，后6个为英文字母A、B、C、D、E、F或者它们对应的小写字母，分别表示十进制数的10、11、12、13、14、15。例如：

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

它等于十进制的35243.25。

科技文献中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，当十六进制数的最高位为A～F中的任何一个字母数字时，为了与非数字的字符串相区别，规定应在最前面加一位数字“0”。例如，十六进制数ABCD3456H是非法的书写格式，而应改写为0ABCD3456H。当最高位为数字0～9时，则默认为是数字而非字符串。例如，十六进制数9FEDCH是合法的书写格式。

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标记“D”的数将默认为十进制数，例如$(245)_{10}$记为245D或245。

见表1.1列举了部分无符号二进制数、十进制数与十六进制数。这里所有的位都用来表示数值，是无正负符号位的数。

表1.1　　无符号二进制数、十进制数和十六进制数的对照

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），数据的存储与处理通常以字节为计算单元。两个字节组成的16位二进制数称为一个字（Word），4个字节组成的32位二进制数称为一个双字（Double Word），8个字节组成的64位二进制数称为一个四字（Quit Word）。总结如下：

对于8位无符号的二进制数，能够表示的十进制数范围是0～255；

对于16位无符号的二进制数，能够表示的十进制数范围是0～65535；

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

1.1.2　数值转换

在上述二进制数与十六进制数的介绍中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，亦即只需要将非十进制数以多项式的形式展开，然后计算它们的十进制数之和即可将其转化为十进制数。

下面讨论十进制数转换为非十进制数以及二进制数同十六进制数之间的相互转化。

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，只需要将该十进制数除以该进制的基数，将所得的商数再除以基数，直至不能整除为止，然后取每次相除所得的余数，按“后高前低”的顺序排列即为转换结果，即首次得到的余数为最低位系数，最后得到的余数是最高位系数。

将十进制数小数部分转换为其他进制，则应将该进制的基数乘以十进制数小数部分，取出乘积的小数部分再乘基数直至乘积为 0，或结果达到预定的精度，然后将每次乘积的整数按“前高后低”的顺序排列即为转换结果。

以下讨论几种常见的转换。

（1）十进制转换为二进制数，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，第一次的余数就是 b_0，再将商数除以 2 又得到一个商数和余数，第二次的余数就是 b_1，如此反复除下去直至不能整除为止。以最后所得的商数“1”作为最高位，将各次所得的余数按“后高前低”的顺序写下来即得到用二进制表示的结果。

【例 1.1】 100＝(?)B

转换过程见表 1.2。

表 1.2　　除 2 取余法

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

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，第一次的余数就是 h_0，再将商数除以 16 又得到一个商数和余数，第二次的余数就是 h_1，如此继续下去直至不能整除为止。以最后所得的商数作为最高位，将各次所得的余数按“后高前低”的顺序写下来即得用十六进制表示的结果。

【例 1.2】 35243＝(?)H。

转化过程见表 1.3。

表 1.3　　除 16 取余法

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

结果得 35243＝89ABH。

(3) 将十进制数小数部分转换成二进制小数，采用乘 2 取整法，即将十进制纯小数部分乘 2，提取乘积中的整数后保留小数部分再乘 2，第一次提取的整数就是 b_{-1}，如此继续下去直至乘积小数部分为零或者得到要求的位数为止，即得到 b_{-m}。将各次获得的整数依次“前高后低”的顺序写出来即为转换后的二进制纯小数结果。

【例 1.3】 0.1875＝(?)B

转换过程见表 1.4。

表 1.4　　乘 2 取整法

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

结果得 0.1875＝0.0011B。

(4) 将十进制数小数部分转换为十六进制小数，采用乘 16 取整法，即将十进制纯小数部分乘以 16，摘除乘积中的整数后保留小数部分再乘以 16，第一次提取的整数就是 h_{-1} 如此循环下去直至乘积小数部分为零或者得到要求的位数为止，即得到 h_{-m}。将各次摘取的整数依“前高后低”的顺序写出来即为转换后的十六进制纯小数结果。

【例 1.4】 0.78125＝(?)H

转换过程见表 1.5 所示。

表 1.5　　乘 16 取整法

	积的整数部分	被乘数/积的小数部分	乘　数
最高位 ↓		0.78125	16
	12	0.5	16
最低位	8	0.0	

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表示的是 4 位二进制数，所以两者之间的转换是比较容易

的。将二进制数转换成十六进制数时，以小数点为界，整数部分从右至左每 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　0011.0110　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”误认为是“3”，它的后面还有两个 0。

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，克服容易出错的弊病，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，只需要将十六进制数的每一位用对应的 4 位二进制数表示，并依照原顺序排列即可。例如：

5　　F　　8　　4　.　E　　4　H

=0101　1111　1000 0100.　1110　0100　B

1.1.3　带符号数

1. 有符号数的原码表示法

如果用“+”、“-”符号表示正数与负数，那么当数据带符号时，一般规定计算机用数据的最高位作为符号位，且规定该位为“0”表示正数，该位为“1”表示负数。这样，在一个字节数据的 8 位二进制中就只有 7 位表示数值了。同理，在 16 位二进制的字节数据中就只有 15 位表示数值；32 位的双字节数据中则只有 31 个数据位。由于这里将符号位数字化，因此带符号数与无符号数两者在表达形式上看不出任何差别，但它们所表示的数值范围是完全不同的。

例如　00000100　表示+4，　10000100　表示-4。

8 位有符号二进制数表达的整数范围是：+127～-127，有+0，-0 两种。

16 位有符号二进制数表达的范围是：+32767～-32767，有+0，-0 两种。

2. 有符号数的反码表示法

正数的反码与原码相同，负数的反码就是它的正数（连符号位）按位取反后得到：

例如　00000100　表示+4，　　11111100　表示-3　（由 00000011 各位取反后得到）

(+127) =01111111　(-127) =10000000

(-31) =11100000

(-0) =11111111　　(+0) =00000000

8 位带符号的反码所能表示数的范围：+127～-127。

3. 负数的补码表示法

引入符号位之后，由于正数的符号位为 0，同不带符号的数相比并没有发生“质”的变化，只是能表示的数值范围变小了。

例如不带符号时，8 位二进制数 01111111B 表示 127，11111111B 表示 255，它是可以表示的最大数。考虑符号位后，+127 依然用 01111111B 来表示，然而它却成了用 8 位二进制可以表示的最大正数，因为再加 1 即为 10000000B，数值位向符号位进位，结果是

负数了，运算出错，此种现象称为溢出。一定码长的数所能表达有符号数的范围一定，超出所能表达数的范围就称为溢出。

那么负数又如何表示呢？下面以－72为例来说明它的表示方法与步骤：

(1) 写出该负数对应的正数的二进制数，例如72＝01001000B。

(2) 将该二进制数按位取反，即1改写为0，0改写为1，这个过程简称为“取反”(Not)，相当于逻辑非，例如01001000B取反得10110111B。

(3) 再在最低位加上1，例如10110111B加1后得10111000B。

整个过程称为“求补” (Complement)，其结果就是负数的补码格式，因此－72＝B8H。

由于构成计算机的数字逻辑电路对于“取反”、“加1”、“进位”等操作轻而易举，所以引入补码表示负数为计算机的运算操作提供了极大的方便，提高了机器的运算速度，例如采用这种格式表示负数后，减法就可以当作加法来操作。

8位带符号的补码所能表示数的范围：＋127～－128。

【例1.5】 127－16＝?

因为127－16＝127＋(－16)，而－16的补码格式表示为：

```
    0  0  0  1  0  0  0  0
    1  1  1  0  1  1  1  1   ;  取反
+                        1   ;  加1
=   1  1  1  1  0  0  0  0
```

所以计算机只需做以下加法：

```
       0  1  1  1  1  1  1  1
  +    1  1  1  1  0  0  0  0
  =  1  0  1  1  0  1  1  1  1
```

忽略进位即为运算结果，用十六进制表示是6FH。将其转换为十进制来验证：

$$01101111B=6FH=6\times16+15=111$$

补码加法的实质是引入了参考数的概念：

求(－16)的补码可以写成：$2^8-16=100000000-10000=11110000$

计算机补码加法运算：

```
          0  1  1  1  1  1  1  1
      +   1  1  1  1  0  0  0  0
      =   1  0  1  1  0  1  1  1  1
```

为了得到正确的结果必须舍去最高位，即$2^8=100000000$才能得到正确的结果。

将上面过程归纳：$127-16=127+2^8-2^8-16=127+2^8-16-2^8=127+16$的补码

-2^8

$=$ 01111111＋11110000－100000000＝01101111B

注意：正数的原码、反码和补码是一样。只有负数的原码、反码和补码是不一样，且只有负数才有补码。

1.1.4　符号数的运算

因为任何正数加上它对应的负数必为 0，所以它们互补。对正数求补可得到它对应的负数，对负数求补又可以得到对应的正数，也就是得到了绝对值。因此带符号数的换算并不复杂。

已知二进制的符号数，将它转换为十进制的方法如下：

(1) 判断最高位，即符号位是 0，还是 1。如果符号位是 0，说明是正数，此时直接转换即可。

(2) 如果符号位是 1，说明是负数，对该数求补。

(3) 将所得结果转换成十进制数，该数对应的负数即为所求结果。

【例 1.6】 将符号数 88H 转换成十进制数。

88H＝10001000B，符号位为 1 是负数，因此要对 88H 求补：

```
    1  0  0  0  1  0  0  0
    0  1  1  1  0  1  1  1   ；取反
+                       1   ；加 1
=   0  1  1  1  1  0  0  0   ＝ 78H
```

因为 78H＝120，故结果得出符号数 88H＝－120。

表 1.6　　8 位/16 位/32 位二进制无符号/带符号数的表示法

8　位			16　位			32　位		
十六进制	无符号十进制	带符号十进制	十六进制	无符号十进制	带符号十进制	十六进制	无符号十进制	带符号十进制
00H	0	0	0000H	0	0	00000000H	0	0
01H	1	＋1	0001H	1	＋1	00000001H	1	＋1
…	…	…	…	…	…	…	…	…
7FH	127	＋127	7FFFH	32767	＋32767	7FFFFFFFH	2147483647	＋2147483647
80H	128	－128	8000H	32768	－32768	80000000H	2147483648	－2147483648
81H	129	－127	8001H	32769	－32767	80000001H	2147483649	－2147483647
…	…	…	…	…	…	…	…	…
0FEH	254	－2	0FFFEH	65534	－2	0FFFFFFFEH	4294967294	－2
0FFH	255	－1	0FFFFH	65535	－1	0FFFFFFFFH	4294967295	－1

表 1.6 列出了部分带符号数与不带符号数的对应关系，从中可以发现它们拥有以下一些特点：

1) 8 位带符号数的数值范围为$-2^7 \sim +2^7-1$(－128～＋127)。

2) 16 位带符号数的数值范围为$-2^{15} \sim +2^{15}-1$(－32768～＋32767)。

3) 32 位带符号数的数值范围为$-2^{31} \sim +2^{31}-1$(－2147483648～＋2147483647)。

4) 当要将 8 位带符号数扩展成 16 位带符号数，或者将 16 位带符号数扩展成 32 位带符号数时，只需将最左边的符号再往左延伸至 16 位或者 32 位即可。

【例 1.7】 将 8 位带符号数 64H(100) 扩展为 16 位带符号的二进制数。

64H＝01100100B，根据符号位向左扩展法则，用 16 位带符号的二进制表示则为：