

普通高等教育“十二五”规划教材

叶青 主编 / 刘铮 张静 雷辉 副主编

微机原理与 接口技术

清华大学出版社

普通高等教育“十二五”规划教材

微机原理与 接口技术

叶青 主编 / 刘铮 张静 雷辉 副主编

清华大学出版社
北京

内 容 简 介

本书讲解微型计算机的工作原理和接口应用技术。书中以 Intel 8086CPU 为主线,系统讲述微型计算机系统的基本组成、工作原理、指令系统及汇编语言程序设计、半导体存储器技术、硬件接口技术、总线技术、PC 的软件体系、PC 应用系统设计举例,使学生牢固掌握微型计算机的原理和硬件接口技术,建立微型计算机系统的整体概念,并从 PC 应用系统的角度了解其软件体系和相关接口。

本书适合作为高等学校非计算机专业微型计算机原理与接口技术、微型计算机原理及应用课程的教材,也可供从事微型计算机硬件和软件设计的工程技术人员参考。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

微机原理与接口技术/叶青主编.--北京:清华大学出版社,2011.9

ISBN 978-7-302-26624-2

I. ①微… II. ①叶… III. ①微型计算机—理论 ②微型计算机—接口 IV. ①TP36

中国版本图书馆 CIP 数据核字(2011)第 172226 号

责任编辑:张占奎

责任校对:刘玉霞

责任印制:李红英

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京密云胶印厂

装 订 者:三河市新茂装订有限公司

经 销:全国新华书店

开 本:185×260 印 张:16.25 字 数:392 千字

版 次:2011 年 9 月第 1 版 印 次:2011 年 9 月第 1 次印刷

印 数:1~4000

定 价:29.80 元

产品编号:038423-01

“微机原理与接口技术”是大学本科信息类、电气类、机电类等非计算机专业的一门重要基础课程,学生通过对这门课程的学习,掌握微型计算机硬件系统的组成和工作原理,提高对微型计算机系统的理解和应用能力,为将来学习和应用层出不穷的微型计算机新技术打下良好的基础。

目前开设的“微机原理与接口技术”课程主要有基于 51 系列单片机和基于 80x86CPU 的两种模式,两种模式各有所长。考虑到以 Intel CPU 为核心的 PC 系列微型计算机结构的完整性和应用普及性,同时考虑到原理性课程的讲解平台不宜太复杂,本书选用以 8086/8088CPU 为主来介绍微型计算机工作原理、汇编语言及接口技术。除了该课程的基本内容外,本书还增加了现代 PC 常用的总线技术、存储器管理、80x86/Pentium 微处理器的发展等内容。特别是考虑到 PC 作为上位机广泛应用于各种监测、控制、网络通信等场合,其在接口的应用方式上与芯片级接口有很大不同,本书从 PC 系统应用的需求出发,增加了 PC 的软件体系与软件接口基础知识、基于 PC 的应用系统设计举例。这样,读者在学习传统的微机硬件系统工作原理的基础上,也学习到更实用的总线技术,了解 PC 系统软件体系及接口,为进一步理解和应用纷繁复杂的微型计算机新技术打下基础。

全书分为 9 章,第 1 章至第 5 章为基本原理部分,主要讲解 8088/8086 微型计算机的基本原理;第 6 章至第 9 章为应用技术部分,主要介绍微型计算机常用接口技术和 PC 应用技术。其中,第 1 章简要介绍微型计算机基本概念、发展历程和 PC 系统的典型配置;第 2 章阐述 8088/8086 16 位微处理器的编程结构及构成硬件系统的结构、时序,介绍 80x86/Pentium 微处理器技术的发展;第 3 章讲解 8086/8088 16 位微处理器的指令系统和汇编语言程序设计;第 4 章主要阐述微型计算机存储器技术基本原理,也介绍了现代微处理器的存储器管理技术;第 5 章阐述输入/输出的概念以及 8086/8088CPU 的中断系统;第 6 章介绍微机常用接口芯片及应用;第 7 章介绍 PCI 等微机系统总线、IEEE 1394、USB 等外部总线,阐述了总线基本概念;第 8 章介绍操作系统概念、Windows 2000/XP 的体系结构、Windows API 等软件接口;第 9 章通过举例介绍了 PC 系统基于 RS-232C、RS-485 串行通信以及基于 PCI 总线的应用方法。

本书在编写上力求循序渐进,简洁明了,突出基本原理和基本概念,同时注重系统的应用。例如将 IBM PC/XT 系统的背景穿插其中,以使读者了解原理在一个系统中的具体应用方式。书中还采用了较多的表格对指令功能、接口功能等进行归纳,便于教师在给出整体面貌的基础上筛选重点。由于该课程内容繁多,教师可根据具体教学时数,安排课程讲解和自学的內容。

本书第 2、4、5 章及第 1.2 节由叶青编写,第 3 章由张静编写,第 6 章及第 1.1 节由雷辉编写,第 7、8、9 章由刘铮编写,何青编写了第 1.3 节,全书由叶青负责筹划和统稿。参与本书编写工作的还有彭赋、付成宏、马洪江,在此一并感谢。

对为本书付出辛勤劳动的清华大学出版社的编辑及工作人员表示衷心感谢。

由于编者的水平和经验有限,书中难免有疏漏和错误之处,恳请专家和读者批评指正。

编 者

2011 年 6 月

第 1 章 微型计算机基础知识	1
1.1 计算机中数的表示与编码	1
1.1.1 数制及其转换.....	1
1.1.2 带符号数的表示.....	5
1.1.3 定点数与浮点数.....	7
1.1.4 计算机中的编码.....	9
1.2 微型计算机系统的构成.....	11
1.2.1 微型计算机硬件系统的组成和结构	11
1.2.2 微型计算机基本工作原理和工作过程	13
1.2.3 微型计算机系统及其主要技术指标	16
1.3 微型计算机的发展及典型配置.....	18
1.3.1 微型计算机的发展	18
1.3.2 典型微型计算机系统的组成	24
习题 1	28
第 2 章 微处理器	29
2.1 8086/8088 微处理器的编程结构	29
2.1.1 8086CPU 的内部功能结构	29
2.1.2 8086CPU 的寄存器结构	31
2.2 存储器及 I/O 端口的组织与管理	33
2.3 8086CPU 的引脚信号及工作模式	35
2.3.1 8086CPU 的引脚及其功能	35
2.3.2 最小工作模式	39
2.3.3 最大工作模式	40
2.4 8086CPU 总线操作时序	41
2.4.1 时序的基本概念	41
2.4.2 典型时序分析	42
2.5 80x86 微处理器简介	44
习题 2	48
第 3 章 指令系统及汇编语言程序设计	49
3.1 8086 指令系统概述	49

3.1.1	8086 指令的基本格式	49
3.1.2	8086 寻址方式	50
3.2	8086CPU 基本指令	54
3.2.1	传送类指令	54
3.2.2	数据操作类指令	59
3.2.3	串操作类指令	66
3.2.4	控制转移类指令	68
3.2.5	处理器控制指令	71
3.3	8086 汇编语言程序的编程格式	73
3.3.1	汇编语言及其源程序结构	73
3.3.2	汇编语言语句格式	74
3.3.3	伪指令	77
3.4	汇编语言程序的上机过程	81
3.4.1	上机运行的软件环境	81
3.4.2	上机过程	81
3.5	汇编语言基本结构程序设计	82
3.5.1	程序设计步骤	82
3.5.2	顺序程序	83
3.5.3	分支程序	84
3.5.4	循环程序	86
3.5.5	子程序	88
3.5.6	DOS 和 BIOS 的中断功能调用	90
习题 3	93
第 4 章	半导体存储器	97
4.1	半导体存储器概述	97
4.1.1	存储器的分类	97
4.1.2	半导体存储器的性能指标	98
4.1.3	存储系统的层次结构	99
4.1.4	半导体存储器的内部功能结构	99
4.2	随机存取存储器	101
4.2.1	静态随机存取存储器 SRAM	101
4.2.2	动态随机存取存储器 DRAM	103
4.3	只读存储器	104
4.3.1	掩膜 ROM	104
4.3.2	可编程 ROM	105
4.3.3	可擦除可编程 ROM	105
4.3.4	电可擦除可编程 ROM	106
4.3.5	闪速存储器	106

4.4	存储器系统设计	107
4.4.1	一般要求	107
4.4.2	存储器与 CPU 的连接	107
4.5	存储器的管理	112
4.5.1	PC 的内存	112
4.5.2	存储器的管理模式	113
4.5.3	虚拟存储技术	114
4.6	高速缓存技术	117
	习题 4	118
第 5 章	输入/输出接口及中断	119
5.1	输入/输出接口概述	119
5.1.1	I/O 接口的组成及功能	119
5.1.2	I/O 端口的寻址	121
5.2	CPU 与外设间的数据传送方式	122
5.2.1	直接传送方式	122
5.2.2	查询传送方式	123
5.2.3	中断传送方式	125
5.2.4	直接存储器存取方式	125
5.3	中断	127
5.3.1	中断系统的基本概念	127
5.3.2	8086 中断系统	130
5.3.3	8086 中断向量表与中断响应过程	132
	习题 5	135
第 6 章	微型计算机常用接口技术	136
6.1	IBM PC/XT 系统的接口	136
6.1.1	IBM PC/XT 系统接口的组成	136
6.1.2	接口地址分配	137
6.2	可编程中断控制器	139
6.2.1	8259A 的内部结构和引脚	139
6.2.2	8259A 的工作方式	142
6.2.3	8259A 的编程	146
6.2.4	中断应用程序举例	151
6.3	定时/计数器	153
6.3.1	概述	153
6.3.2	可编程定时/计数芯片 8253 及其应用	153
6.4	并行通信接口	162
6.4.1	概述	162

6.4.2	可编程的并行接口 8255A 及其应用	162
6.5	串行通信接口	172
6.5.1	串行通信的基本概念	172
6.5.2	串行通信的接口标准	174
6.5.3	可编程串行接口芯片 16550 及其应用	176
6.6	模/数和数/模转换器	180
6.6.1	D/A 转换器	181
6.6.2	A/D 转换器	184
	习题 6	188
第 7 章	总线技术	190
7.1	总线的基本概念	190
7.1.1	总线及总线标准的定义	190
7.1.2	总线的分类	191
7.1.3	总线的性能指标	193
7.1.4	总线的层次化结构	195
7.1.5	总线的控制与总线传输	197
7.2	常用系统总线	197
7.2.1	STD 总线	198
7.2.2	PC 系列总线	198
7.2.3	PCI 总线	200
7.3	常用通信总线	204
7.3.1	IEEE 488 总线	204
7.3.2	IEEE 1394 总线	206
7.3.3	USB 总线	207
7.3.4	现场总线简介	210
	习题 7	212
第 8 章	PC 的软件体系与软件接口	213
8.1	Windows 2000/XP 的体系结构	213
8.1.1	操作系统简介	213
8.1.2	Windows 2000/XP 的系统构成	215
8.1.3	对硬件的支持——驱动程序	217
8.2	软件接口	219
8.2.1	软件接口概述	219
8.2.2	Windows API 接口	220
8.2.3	动态链接库	222
8.2.4	Socket 接口	223
8.3	Windows 应用程序设计	225

8.3.1 PC 中常用的软件体系结构	225
8.3.2 Windows 应用程序设计模式	226
习题 8	229
第 9 章 基于 PC 的应用系统设计举例	230
9.1 串行数据采集系统设计	230
9.1.1 基于 RS-232C 的数据采集系统	230
9.1.2 基于 RS-485 的数据采集系统	236
9.2 基于 PCI 总线的系统设计	239
9.2.1 PCI 总线 I/O 接口电路设计	240
9.2.2 PCI 总线接口卡的驱动程序设计	243
习题 9	248
参考文献	249

微型计算机基础知识

电子计算机的诞生和发展是 20 世纪最重要的科技成果之一。微型计算机(简称“微机”)是计算机的一个重要分支。本章介绍微型计算机系统的基础知识,概述其发展变化,内容包括计算机中的数制和编码,微型计算机的结构、工作原理、分类及主要性能指标,典型微型计算机系统。

1.1 计算机中数的表示与编码

1.1.1 数制及其转换

数制是人们利用符号来计数的科学方法。数制有很多种,在日常生活中人们常用十进制计数,在计算机内部,一切信息的存储、处理与传送均采用二进制的形式。一个具有两种不同稳定状态且能相互转换的器件即可以用来表示一位二进制数,因而一个二进制数在计算机内部是以电子器件的物理状态来表示的,二进制的表示是最简单且最可靠的。由于八进制、十六进制与二进制之间有非常简单的对应关系,而且位数相对减少,在阅读与书写时常常采用八进制或十六进制。因而在计算机的设计及使用中,通常使用的计数方法是二进制、八进制、十进制和十六进制。

1. 进位计数制

进位计数制是采用位置表示法,即处于不同位置的同一数字符号所表示的数字不同。一般说来,如果数制只采用 R 个基本符号,则称为基 R 数制, R 称为数制的“基数”或简称“基”;而数制中每一个固定位置对应的单位值称为“权”。

对 R 进制数来说,有以下特点:

- ① 能选用的数码的个数等于基数 R ,即各数位只允许是 $0, 1, \dots, R-1$;
- ② 各位的权是以 R 为底的幂;
- ③ 计数规则是“逢 R 进一”。

常用四种进制数的表示法如表 1.1 所示。

对任意一个进制数,都可以按权展开成多项式,其中每一项表示相应数位代表的数值。例如“逢十进一”的十进制数 258.5 可写为

$$258.5 = 2 \times 10^2 + 5 \times 10^1 + 8 \times 10^0 + 5 \times 10^{-1}$$

对 R 进制数 N , 若用 $n+m$ 个代码 D_i ($-m \leq i \leq n-1$) 表示, 从 D_{n-1} 到 D_{-m} 自左至右排列, 则其按权展开多项式为

$$N = D_{n-1}R^{n-1} + D_{n-2}R^{n-2} + \cdots + D_0R^0 + D_{-1}R^{-1} + \cdots + D_{-m}R^{-m} \quad (1.1)$$

式中, D_i 为第 i 位代码, 它可取 $0 \sim (R-1)$ 之间的任何数字符号; m 和 n 均为正整数, n 表示整数部分的位数, m 表示小数部分的位数。表 1.2 列出了四种进制数表示数的对应关系。

表 1.1 常用四种数制的表示法

数制	二进制	八进制	十进制	十六进制
进位规则	逢二进一	逢八进一	逢十进一	逢十六进一
基数	2	8	10	16
所用符号	0, 1	0, 1, 2, ..., 7	0, 1, 2, ..., 9	0, 1, 2, ..., 9, A, B, ..., F
权	2^i	8^i	10^i	16^i
数制标识	B	Q	D	H

表 1.2 四种数制表示数的对应关系

十进制	二进制	八进制	十六进制	十进制	二进制	八进制	十六进制
0	0	0	0	8	1000	10	8
1	1	1	1	9	1001	11	9
2	10	2	2	10	1010	12	A
3	11	3	3	11	1011	13	B
4	100	4	4	12	1100	14	C
5	101	5	5	13	1101	15	D
6	110	6	6	14	1110	16	E
7	111	7	7	15	1111	17	F

2. 数制之间的转换

1) 二、八、十六进制数转换成十进制数

转换规则为“按权相加”, 即将二、八、十六进制数按权展开, 求出按权展开式的值就是该数转换为十进制的等价值。

例 1.1 $(1011.01)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = (11.25)_{10}$

$$(54)_8 = 5 \times 8^1 + 4 \times 8^0 = (44)_{10}$$

$$(7A)_{16} = 7 \times 16^1 + 10 \times 16^0 = (122)_{10}$$

2) 十进制转换成二、八、十六进制数

十进制数转换成二、八、十六进制数时, 需要把整数部分与小数部分分别转换, 然后拼接起来。

(1) 整数部分的转换

十进制整数转换为二进制整数的方法: 把被转换的十进制整数反复地除以 2, 直到商为 0, 所得的余数(从末位读起)就是这个数的二进制表示。简单地说, 就是“除 2 取余法”。

例 1.2 将十进制整数 105 转换成二进制整数。

解：按“除 2 取余”方法进行，转换的过程如下：

		余数	
2	105		低位
2	52	1	↑
2	26	0	↑
2	13	0	↑
2	6	1	↑
2	3	0	↑
2	1	1	↑
	0	1	↑
			高位

转换结果为： $(105)_{10} = (1101001)_2$ 。

同理，将十进制整数转换为 R 进制数，按照“除 R 取余”规则即可。

例 1.3 将十进制数 545 转换为十六进制数。

解：按“除 16 取余”方法进行，转换过程如下：

		余数	
16	545		低位
16	34	1	↑
16	2	2	↑
	0	2	↑
			高位

转换结果为： $(545)_{10} = (221)_{16}$ 。

(2) 小数部分的转换

十进制小数转换成二进制小数的方法：将十进制小数连续乘以 2，选取进位整数，直到满足精度要求为止，简称“乘 2 取整法”。

例 1.4 将十进制小数 0.625 转换成二进制小数。

解：转换过程如下：

0.625		
× 2	1.25	整数部分为 1
	0.25	
× 2	0.5	整数部分为 0
	0.5	
× 2	1.0	整数部分为 1

高位
↓
低位

将十进制小数 0.625 连续乘以 2，把每次所进位的整数，按从上往下的顺序写出。于是，转换结果为： $(0.625)_{10} = (0.101)_2$ 。

同理,十进制小数转换成 R 进制小数的方法是“乘 R 取整法”。

例 1.5 将十进制小数 0.145 转换成十六进制小数。

解: 转换过程如下:

0.145		
× 16		
2.32	整数部分为 2	高位
0.32		
× 16		
5.12	整数部分为 5	
0.12		
× 16		
1.92	整数部分为 1	
0.92		
× 16		
14.72	整数部分为 14(E)	低位

将十进制小数 0.145 连续乘以 16,取每次进位的整数,直到满足精度要求为止,因而转换结果为: $(0.145)_{10} = (0.251E)_{16}$ 。

由上可知,十进制整数部分的转换采用基数不断去除要转换的十进制数,直到商为 0 为止,将各次计算所得的余数,按最后的余数为最高位,第一位为最低位,依次排列,即得转换结果。十进制小数部分的转换采用基数不断去乘需要转换的十进制小数,直到满足要求的精度或小数部分等于 0 为止,然后取每次乘积结果的整数部分,以第一次取整位为最高位,依次排列,即可得到转换结果。

如果一个数既有小数又有整数,则应将整数部分与小数部分分别进行转换,然后用小数点将两部分连起来,即为转换结果。

例如: $(42.125)_{10} = (42)_{10} + (0.125)_{10}$

$$\begin{array}{ccc} \downarrow & & \downarrow \\ (101010)_2 & & (0.001)_2 \end{array}$$

所以 $(42.125)_{10} = (101010.001)_2$

3) 二进制与八进制、十六进制间的相互转换

由于 $2^3=8, 2^4=16$,因此二进制与八进制、十六进制之间的转换很简单。将二进制数从小数点位开始,向左每 3 位产生一个八进制数字,不足 3 位的左边补零,这样就得到整数部分的八进制数;向右每 3 位产生一个八进制数字,不足 3 位右边补 0,就得到小数部分的八进制数。同理,向左每 4 位产生一个十六进制数字,不足 4 位的左边补零,就得到整数部分的十六进制数;向右每 4 位产生一个十六进制数字,不足 4 位右边补 0,就得到小数部分的十六进制数。

例如: $(01111100.1001001)_2 = (174.444)_8 = (7C.92)_{16}$

八进制要转换成二进制,只需将八进制数分别用对应的三位二进制数表示即可;十六进制要转换成二进制,只需将十六进制数分别用对应的四位二进制数表示即可。

为了便于区别不同数制表示的数,规定在数字后面用一个 H 表示十六进制数,用 B 表示二进制数,用 D(或不加标志)表示十进制数,如 64H、1101B、369D 分别表示十六进制、二

进制和十进制数。另外,规定当十六进制数以字母开头时,为了避免与其他字符相混,在书写时前面加一个数0,如十六进制数B9H,应写成0B9H。

1.1.2 带符号数的表示

1. 机器数与真值

日常生活中遇到的数,除了上述无符号数外,还有带符号数。对于带符号的二进制数,其正负符号如何表示呢?在计算机中,为了区别正数和负数,通常用二进制数的最高位表示数的符号,对于一个字节型二进制数来说, D_7 位为符号位, $D_6 \sim D_0$ 位为数值位。在符号位中,规定用“0”表示正,“1”表示负,而数值位表示该数的数值大小。把一个数及其符号位在机器中的一组二进制数表示形式,称为“机器数”,机器数所表示的值称为该机器数的“真值”。

2. 机器数的表示方法

机器数可以用不同的表示方法,常用的有原码表示法、反码表示法和补码表示法。

1) 原码表示法

最高位为符号位(正数为0,负数为1),其余数字位表示数的绝对值。

例如,当机器字长为8时:

$$\begin{aligned} [+0]_{\text{原}} &= 00000000\text{B} & [-0]_{\text{原}} &= 10000000\text{B} \\ [+4]_{\text{原}} &= 00000100\text{B} & [-4]_{\text{原}} &= 10000100\text{B} \\ [+127]_{\text{原}} &= 01111111\text{B} & [-127]_{\text{原}} &= 11111111\text{B} \end{aligned}$$

注意:

① “0”的原码有两种表示法:00000000表示+0,10000000表示-0。

② 若微机字长为8位,则原码的表示范围为-127~+127;若字长为16位,则原码的表示范围为-32767~+32767。

原码表示法简单直观,且与真值的转换很方便,但不便于在计算机中进行加减运算。如进行两数相加,必须先判断两个数的符号是否相同。如果相同,则进行加法运算,否则进行减法运算。如进行两数相减,必须比较两数的绝对值大小,再由大数减小数,结果的符号要和绝对值大的数的符号一致。按上述运算方法设计的算术运算电路很复杂。因此,计算机中通常使用补码进行加减运算,这样就引入了反码表示法和补码表示法。

2) 反码表示法

正数的反码与其原码相同,负数的反码是在原码基础上,符号位不变(仍为1),数值位则按位取反。例如,当机器字长为8时:

$$\begin{aligned} [+0]_{\text{反}} &= [+0]_{\text{原}} = 00000000\text{B} & [-0]_{\text{反}} &= 11111111\text{B} \\ [+4]_{\text{反}} &= [+4]_{\text{原}} = 00000100\text{B} & [-4]_{\text{反}} &= 11111011\text{B} \\ [+127]_{\text{反}} &= [+127]_{\text{原}} = 01111111\text{B} & [-127]_{\text{反}} &= 10000000\text{B} \end{aligned}$$

注意:

① “0”的反码有两种表示法:00000000表示+0,11111111表示-0。

② 若微机字长为8位,则反码的表示范围为-127~+127;若字长为16位,则反码的表示范围为-32767~+32767。

3) 补码表示法

正数的补码与其原码相同,负数的补码是在原码基础上,符号位不变(仍为1),数值位

则按位取反再加一。例如,当机器字长为 8 时:

$$\begin{aligned} [+0]_{\text{补}} &= [+0]_{\text{原}} = 00000000\text{B} & [-0]_{\text{补}} &= [-0]_{\text{反}} + 1 = 00000000\text{B} \\ [+4]_{\text{补}} &= [+4]_{\text{原}} = 00000100\text{B} & [-4]_{\text{补}} &= [-4]_{\text{反}} + 1 = 11111100\text{B} \\ [+127]_{\text{补}} &= [+127]_{\text{原}} = 01111111\text{B} & [-127]_{\text{补}} &= [-127]_{\text{反}} + 1 = 10000001\text{B} \end{aligned}$$

注意:

① $[+0]_{\text{补}} = [-0]_{\text{补}} = 00000000$, 无 +0 和 -0 之分。

② 正因为补码中没有 +0 和 -0 之分, 所以 8 位二进制补码所能表示的数值范围为 +127~-128; 16 位二进制补码表示数的范围为 -32768~+32767。

3. 机器数与真值之间的转换

1) 原码转换为真值

根据原码定义, 将原码数值位各位按权展开求和, 由符号位决定数的正负, 即可由原码求出真值。

例 1.6 已知 $[X]_{\text{原}} = 00011111\text{B}$, $[Y]_{\text{原}} = 10011101\text{B}$, 求 X 和 Y。

解:

$$X = 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 31$$

$$Y = -(0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0) = -29$$

2) 补码转换为真值

求补码的真值, 先求出补码对应的原码, 再按原码转换为真值的方法即可求出其真值。

正数的原码与补码相同。负数的原码可在补码基础上再次求补。

例 1.7 已知 $[X]_{\text{补}} = 00001111\text{B}$, $[Y]_{\text{补}} = 11100101\text{B}$, 求 X 和 Y。

解:

$$[X]_{\text{原}} = [X]_{\text{补}} = 00001111\text{B}, \quad X = 15$$

$$[Y]_{\text{原}} = [[Y]_{\text{补}}]_{\text{补}} = 10011011\text{B}, \quad Y = -27$$

4. 补码的加减运算

在计算机中, 凡是带符号的数一律用补码表示, 运算结果自然也是补码。其运算特点是: 符号位和数值位一起参加运算, 并且自动获得结果(包括符号位与数值位)。

补码加法的运算规则为: $[X+Y]_{\text{补}} = [X]_{\text{补}} + [Y]_{\text{补}}$

补码减法的运算规则为: $[X-Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$

当运算结果不超出补码所表示的范围时, 运算结果是正确的补码形式。但当运算结果超出补码表示范围时, 结果就不正确了, 这种情况称为溢出。当最高位向更高位的进位由于机器字长的限制而自动丢失时, 并不会影响运算结果的正确性。

计算机中带符号数用补码表示时有如下优点:

① 可以将减法运算变为加法运算, 因此可使用同一个运算器实现加法和减法运算, 简化了电路;

② 无符号数和带符号数的加法运算可以用同一个加法器实现, 简化了微机内部的电路结构。

5. 溢出及其判断方法

1) 进位与溢出

所谓进位是指运算结果的最高位向更高位的进位, 用来判断无符号数运算结果是否超出

了计算机所能表示的最大无符号数的范围。

溢出是指带符号数的补码运算溢出,用来判断带符号数补码运算结果是否超出了补码所能表示的范围。例如,字长为 n 位的带符号数,它能表示的补码范围为 $-2^{n-1} \sim +2^{n-1} - 1$,如果运算结果超出了此范围,就称为补码溢出,简称溢出。

2) 溢出的判断方法

判断溢出常见的方法有:①直接由参加运算的两个数的符号及运算结果的符号进行判断;②通过符号位和数值部分最高位的进位状态来判断结果是否溢出。第一种方法适用于手工运算时对结果是否溢出的判断,第二种方法通常在计算机中使用。

设符号位进位状态用 CF 来表示。当符号位向前有进位时,CF=1,否则 CF=0。数值部分最高位的进位状态用 DF 来表示。当该位向前有进位时,DF=1,否则 DF=0。于是溢出 $OF = CF \oplus DF$,若 $OF=1$,则说明结果溢出, $OF=0$,说明结果未溢出。也就是说,当符号位和数值部分最高位同时有进位或同时没有进位时,运算结果不溢出;否则结果溢出。

例 1.8 设有两个操作数 $x=01000100B$, $y=01001000B$,将这两个操作数送运算器做加法运算:①若为无符号数,计算结果是否正确?②若为带符号补码数,计算结果是否溢出?

解:①若为无符号数,由于 CF=0,说明结果未超出 8 位无符号数所能表达的数值范围(0~255),计算结果 10001100B 为无符号数,其真值为 140,计算结果正确。

②若为带符号数补码,由于 $OF=1$,表明结果溢出;也可通过参加运算的两个数的符号及运算结果的符号进行判断。由于两操作数均为正数,而结果却为负数,所以结果溢出;+68 和 +72 两数补码之和应为 +140 的补码,而 8 位带符号数补码所能表达的数值范围为 $-128 \sim +127$,结果超出该范围,因此结果是错误的。

1.1.3 定点数与浮点数

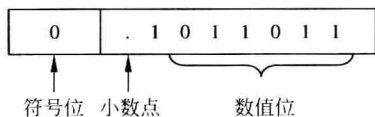
在一般书写中,小数点是用记号“.”来表示的,但在计算机中表示任何信息只能用 0 或 1 两种数码。如果计算机中的小数点用数码表示的话,则不易与二进制数位区分开,所以在计算机中小数点不能用记号“.”表示,那么在计算机中小数点又如何确定呢?

为了确定小数点的位置,在计算机中,数的表示有两种方法,即定点表示法和浮点表示法。

1. 定点表示法

所谓定点表示法,是指在计算机中所有数的小数点的位置人为约定固定不变。一般来说,小数点可约定固定在任何数位之后,但常用下列两种形式:

① 定点纯小数,约定小数点位置固定在符号之后,如:



假设字长为 n 时,定点小数表示范围为 $1-2^{n-1} \sim -(1-2^{n-1})$ 。

② 定点纯整数,约定小数点位置固定在最低数值位之后,如:

