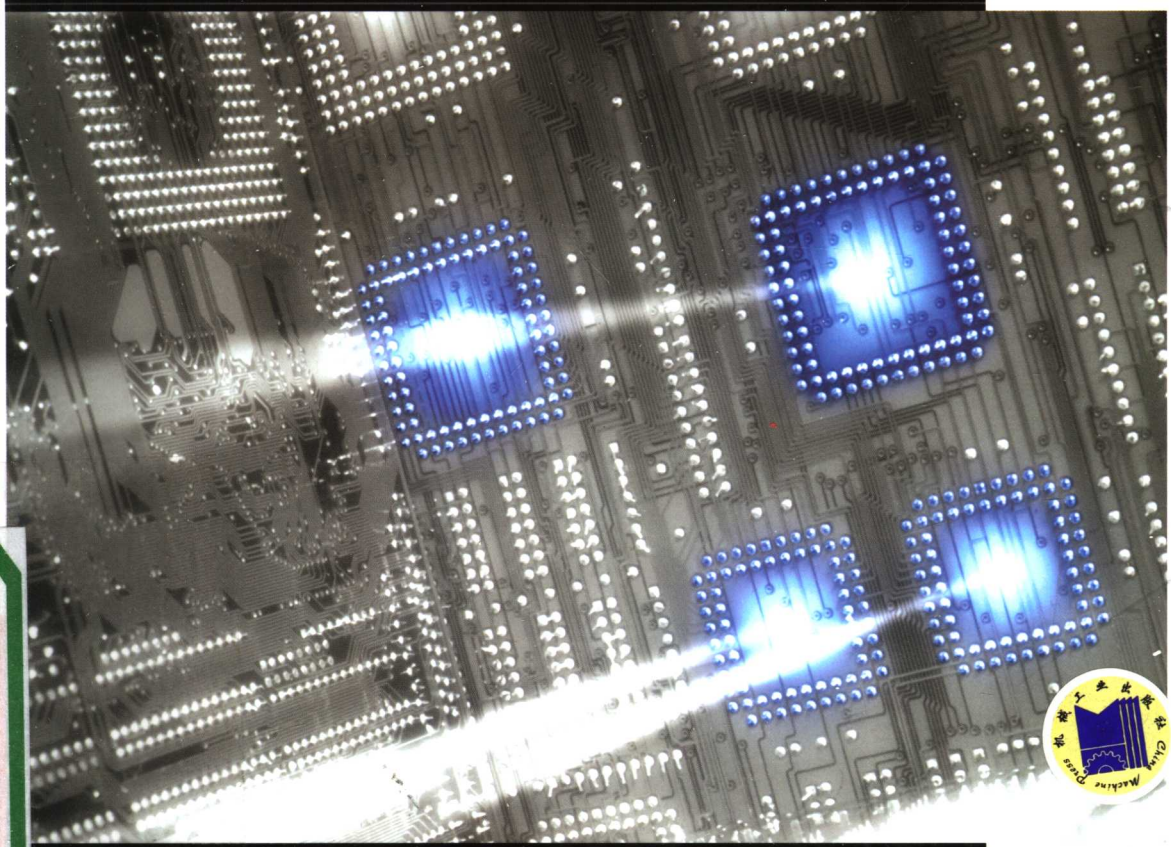
21世纪重点大学规划教材

武庆生　邓建　编著

数字逻辑



机械工业出版社
CHINA MACHINE PRESS

配电子教案

21世纪重点大学规划教材

数字逻辑

武庆生　邓　建　编著

机械工业出版社

“数字逻辑”是计算机专业本、专科学生的一门必修课程。它是“计算机组成原理”、“计算机接口技术”、“微机原理”、“单片机及接口技术”等课程的先修课程之一。本课程的主要目的是使学生从了解数字系统开始，直到能使用数字集成电路实现工程所需逻辑设计的完整过程。

本书根据《计算机学科教学计划》大纲编写，全书共9章，主要内容包括数制和码制、逻辑代数基础、集成门电路、组合逻辑电路、触发器、同步时序逻辑电路、异步时序逻辑电路、硬件描述语言与可编程逻辑器件(PLD)的应用、脉冲波形的产生与整形等。

本书不仅介绍了经典的数字逻辑分析设计方法，而且还介绍了数字电路与逻辑设计的一些最新内容。全书体系新颖、取材科学、内容精练、文字流畅、题例丰富。

本书可作为高等学校计算机、信息、电子工程、自动控制、通信等专业的教材，也可作为成人教育相关课程的教材，并可作为相关专业科技人员的参考书。

图书在版编目(CIP)数据

数字逻辑/武庆生，邓建编著. —北京：机械工业出版社，2007.9
(21世纪重点大学规划教材)
ISBN 978-7-111-22303-0

Ⅰ.数… Ⅱ.①武…②邓… Ⅲ.数字逻辑－高等学校－教材
Ⅳ.TP302.2

中国版本图书馆CIP数据核字(2007)第139250号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）

责任编辑：张宝珠

责任印制：杨 曦

三河市国英印务有限公司印刷

2008年1月第1版·第1次印刷

184mm×260mm·17印张·417千字
0001—5000册
标准书号：ISBN 978-7-111-22303-0
定价：27.00元

凡购本书，如有缺页，倒页，脱页，由本社发行部调换
销售服务热线电话：(010) 68326294
购书热线电话：(010) 88379639 88379641 88379643
编辑热线电话：(010) 88379739
封面无防伪标均为盗版

出 版 说 明

“211 工程”是“重点大学和重点学科建设项目”的简称，是国家“九五”期间惟一的教育重点项目。

进入“211 工程”的 100 所学校拥有全国 32%的在校本科生、69%的硕士、84%的博士生，以及 87%的有博士学位的教师；覆盖了全国 96%的国家重点实验室和 85%的国家重点学科。相对而言，这批学校中的教授、教师有着深厚的专业知识和丰富的教学经验，其中不少教师对我国高等院校的教材建设做过很多重要的工作。为了有效地利用“211 工程”这一丰富资源，实现以重点建设推动整体发展的战略构想，机械工业出版社推出了“21 世纪重点大学规划教材”。

本套教材以重点大学、重点学科的精品教材建设为主要任务，组织知名教授、教师进行编写。教材适用于高等院校计算机及其相关专业，选题涉及公共基础课、硬件、软件、网络技术等，内容紧密贴合高等院校相关学科的课程设置和培养目标，注重教材的科学性、实用性、通用性，在同类教材中具有一定的先进性和权威性。

为了体现建设“立体化”精品教材的宗旨，本套教材为主干课程配备了电子教案、学习指导、习题解答、课程设计、毕业设计指导等内容。

机械工业出版社

前　言

根据高等学校工科计算机专业“数字逻辑”课程教学大纲的要求，并考虑自控、信息、电子工程、通信等专业学习“数字逻辑”课程的需要，编者结合多年的教学经验编写了本书。

“数字逻辑”是计算机科学与技术(类)本、专科学生必修的一门重要专业基础课。本课程的目的是使学生从对数字系统的了解开始，直到能使用数字集成电路实现工程所需的逻辑设计。这为数字计算机和其他数字系统的分析和设计奠定了良好的基础。熟练掌握数字系统逻辑分析和设计的方法，对从事计算机软硬件研制、开发和应用的工程技术人员是非常重要的。

数字集成电路是数字系统与计算机功能实现的基础。所以将数字逻辑设计和数字集成电路结合起来讲授，可使学生既掌握数字逻辑部件的分析与设计方法，又了解标准数字集成电路的原理和使用方法。

全书共分 9 章，第 1 章为数制与码制，介绍了数字系统中常用的数制及转换、码制和编码。第 2 章为逻辑代数，介绍了逻辑代数、逻辑函数及函数化简。第 3 章为集成门电路，介绍了典型 TTL 门，CMOS 门的结构和原理。第 4 章为组合逻辑电路，介绍了组合逻辑电路的分析和设计方法以及典型中规模器件的原理和应用。第 5 章为触发器，介绍了各种触发器的组成、原理和应用。第 6 章为同步时序逻辑电路，介绍同步时序逻辑电路的分析和设计方法以及中规模计数器的组成原理及应用。第 7 章为异步时序逻辑电路，介绍了脉冲异步时序逻辑电路和电平异步时序逻辑电路的分析和设计方法以及中规模异步计数器的原理和应用。第 8 章为可编程逻辑电路，介绍了可编程逻辑器件(PLD)的几种典型结构及应用开发，简要介绍了 ABEL 语言、Verilog HDL 语言和 VHDL 语言等三种最常用的硬件描述语言，并介绍了使用硬件描述语言实施编程应用的方法和例子。第 9 章为脉冲波形的产生与整形，介绍了 555 时基电路、多谐振荡器、单稳态触发器以及施密特触发器的构成与工作原理。

本书在内容和结构上力求主次分明、突出重点、叙述清楚、由浅入深。

本课程的先修课程是“电路与电子技术”。本课程的参考课时为 64 学时，使用者可根据需要和具体情况对内容进行取舍。

本书第 3、4、5、6、9 章由武庆生编写；第 1、2、7、8 章由邓建编写，全书由武庆生统稿。在编写过程中得到了很多同行的大力支持和关怀，电子科技大学计算机学院傅彦副院长十分关心本书的编写工作，并提出了许多宝贵意见，在此表示衷心的感谢。

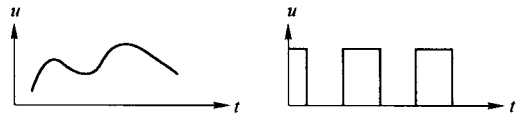
由于编者水平有限，书中难免有不妥之处，敬请读者批评指正。

编　者

目　录

第 1 章　数制与码制

21 世纪人类已经进入信息时代，信息技术与国民经济、人们的日常生活息息相关，信息的采集、存储、传输、计算、利用等技术已成为当前科技研究的重要内容。信息技术处理的信号分为模拟信号和数字信号两大类，如图 1-1 所示。在时间和数值上是连续的信号称为模拟信号。而数字信号是指在低电平和高电平两种状态（即 0 和 1 两个逻辑值）之间作阶跃式变化的信号，这种在时间上和数值上不连续的信号又称为离散信号。



a)　　b)

图 1-1　模拟信号与数字信号

a）模拟信号示例　b）数字信号示例

对数字信号进行传递、处理的电路称为数字电路。由于数字电路不仅能对信号进行数值运算，而且还能进行逻辑运算和逻辑判断，数字电路的输入量和输出量之间的关系是一种因果关系，它可以用逻辑函数来描述，所以又称为数字逻辑电路或逻辑电路。数字逻辑电路主要研究电路输出信号状态与输入信号状态之间的逻辑关系。

在数字系统中，目前广泛采用二进制系统。本章讨论数字和符号在二进制系统中的各种表示方法，为数字系统的设计奠定基础。

1.1　数制

数制即计数体制，它是按照一定规律表示数值大小的计数方法。日常生活中最常用的计数体制是十进制，数字电路中最常用的计数体制是二进制。

在数字电路中，常用一定位数的二进制数码表示不同的事物或信息，这些数码称为代码。编制代码时要遵循一定的规则，这些规则叫码制。

1.1.1　进位计数制

人类在长期的生产实践中学会了用 10 个指头（0～9）计数，很显然仅用一位数码往往不够，必须采用多位数码按先后次序把它们排成数位，由低到高进行计数，计满后进位，这就产生了人们最熟悉的十进制。数位计数制（positional number systems）是人们对数量计数的规律的总结。任何一种进位计数制都包含着基数（base/radix）和位权（weight）两个特征。

1. **基数**

基数是指数制中所采用的数字符号个数，基数为 R 的数制称为 R 进制。R 进制中能表示 0～R－1 的共 R 个数字符号。

2. 位权

位权(简称为权)是指进位计数制中不同数位上的数值,即平常所讲的整数中的个位、十位、百位或小数中的十分位、百分位等。

进位规律是"逢 R 进一"。一个 R 进制数 N 可表示为

$(N)_R=(K_{n-1}\,K_{n-2}\cdots\,K_1K_0.\,K_{-1}\cdots\,K_{-m})_R$　　并列表示法(位置记数法)

或 $(N)_R=\sum_{i=-m}^{n-1}K_iR^i$　　多项式表示法(按权展开式)

式中,R 为基数,Ki 为 0～R－1 中的任何一个字符,n 为整数部分位数,m 为小数部分位数,R^i 为第 i 位的位权。

如 $(6789)_{10}=6\times10^3+7\times10^2+8\times10^1+9\times10^0$

$(4032.8)_{10}=4\times10^3+0\times10^2+3\times10^1+2\times10^0+8\times10^{-1}$

1.1.2 二进制

日常生活中常采用十进制,所以在早期的计算工具的设计中也采用十进制,但随着数字系统的发展,为了便于工程实现,广泛采用二进制。这是因为二进制表示的数的每一位只取数码 0 或 1,因而可以用具有两个不同状态的电子元件来表示。它的运算规则简单,0、1 与逻辑命题中的真假相对应,为计算机中实现逻辑运算和逻辑判断提供了有利条件。

二进制的基数是 2,其计数规律是"逢二进一"。一个二进制数可以表示成

$(N)_2=(K_{n-1}\,K_{n-2}\cdots\,K_1K_0.\,K_{-1}\cdots\,K_{-m})_2$　　并列表示法(位置记数法)

或 $(N)_2=\sum_{i=-m}^{n-1}K_i2^i$　　多项式表示法(按权展开式)

如 $(101.001)_2=1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+0\times2^{-2}+1\times2^{-3}$

$=1\times4+0\times2+1\times1+0\times0.5+0\times0.25+1\times0.125$

$=(5.125)_{10}$

采用二进制计数制,对数字的运算、存储和传输极为方便,然而二进制数书写起来很不方便。为此,人们经常采用八进制数和十六进制数来进行书写或打印。

八进制数的基数是 8,它有 0～7 共 8 个符号。十六进制数的基数是 16,它有 16 个符号,除了0～9以外,还需补充 6 个符号,它们是 A(代表 10),B (代表 11),C (代表 12),D (代表 13),E (代表 14),F (代表 15)。

由于 $2^3=8$,所以 1 位八进制数所能表示的数值,正好和 3 位二进制数对应 。同理,由于 $2^4=16$,所以 1 位十六进制数所能表示的数值,正好和 4 位二进制数对应。

根据上述特点,二进制、八进制、十六进制之间可以很方便地相互转换。二进制数转换为八进制数(十六进制)时以小数点为界,分别往左、右每 3 位(4 位)为一组,最后不足 3 位(4 位)时用 0 补充,然后写出每组对应的八进制(十六进制)字符,即为对应八进制(十六进制)数。

【例 1-1】将 $(374.27)_8$ 和 $(AF4.74)_{16}$ 转为二进制。

解:$(374.27)_8=(011\quad 111\quad 100.\quad 010\quad 111)_2$

$(AF4.74)_{16}=(1010\quad 1111\quad 0100.\quad 0111\quad 0100)_2$

【例 1-2】将 $(10110.1011)_2$ 转为八进制、十六进制数。

解:$(10110.1011)_2=(010\quad 110.\quad 101\quad 100)_2$

$$=(2\quad 6.\quad 5\quad 4)_8$$

$$(10110.1011)_2=(0001\quad 0110.\quad 1011)_2$$

$$=(1\quad 6.\quad B)_{16}$$

即$(10110.1011)_2=(26.54)_8=(16.B)_{16}$

1.1.3 任意进制数转换为十进制数

将一个任意进制数转换成十进制数时采用多项式替代法，即将 R 进制数按权展开，求出各位数值之和，即可得到相应的十进制数。对于八、十六进制这类容易转换为二进制的数，也可以先将其转换成二进制数，然后再转换成十进制数。

【例 1-3】将$(10110.101)_2$、$(436.5)_8$、$(1C4)_{16}$、$(D8.A)_{16}$转换成十进制数。

解：$(10110.101)_2=1\times2^4+0\times2^3+1\times2^2+1\times2^1+0\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$

$$=16+0+4+2+0+0.5+0+0.125$$

$$=(22.625)_{10}$$

$(436.5)_8=4\times8^2+3\times8^1+6\times8^0+5\times8^{-1}=(286.625)_{10}$

$(1C4)_{16}=1\times16^2+12\times16^1+4\times16^0=256+192+4=(452)_{10}$

$(D8.A)_{16}=13\times16^1+8\times16^0+10\times16^{-1}=(216.625)_{10}$

1.1.4 十进制数转换为任意进制数

十进制数转换为其他进制数时，其整数部分和小数部分要分别转换。整数部分采用除基取余法，小数部分采用乘基取整法。

除基取余法是将十进制整数 N 除以基 R，取余数为 K_0，再将所得商除以 R，取余数为 K_1，依此类推，直至商为 0，取余数为 K_{n-1}为止，即得到与 N 对应的 R 进制数$(K_{n-1}\cdots K_1\ K_0)_R$。

【例 1-4】将$(217)_{10}$转换成二进制数。

解：			余数	
2	|	2 1 7		
2	|	1 0 8	1	最低位 K_0
2	|	5 4	0	
2	|	2 7	0	
2	|	1 3	1	
2	|	6	1	
2	|	3	0	
2	|	1	1	
2	|	0	1	最高位 K_7

所以，$(217)_{10}=(11011001)_2$。

乘基取整法是将十进制小数 N 乘以基数 R，取整数部分为 K_{-1}，再将其小数部分乘以 R，取整数部分为 K_{-2}，依此类推，直至其小数部分为 0 或达到规定精度要求，取整数部分记为 K_{-m}为止，即可得到与 N 对应的 m 位 R 进制数$(0.\ K_{-1}\ K_{-2}\cdots\ K_{-m})_R$。

在进制转换时，请不要将结果的次序写错，注意除基取余法、乘基取整法中得到的第一个数最接近小数点就不会混淆了。

【例 1-5】将小数 0.6875 转换成二进制小数。

解：

0.6875			
× 2	整数部分		
1 .3750	1	最高位	K_{-1}
× 2			
0 .7500	0		K_{-2}
× 2			
1 .5000	1		K_{-3}
× 2			
1 .0000	1	最低位	K_{-4}

所以，$(0.6875)_{10}=(0.1011)_2$。

若十进制小数不能用有限位二进制小数精确表示，则应根据精度要求，当要求二进制数取 m 位小数时可求出 m+1 位，然后对最低位作 0 舍 1 入处理。

【例 1-6】将小数 0.324 转换成二进制小数，保留 3 位小数。

解：

0.3245			
× 2	整数部分		
0 .648	0	最高位	K_{-1}
× 2			
1 .296	1		K_{-2}
× 2			
0 .592	0		K_{-3}
× 2			
1 .184	1	最低位	K_{-4}

所以，$(0.324)_{10}\approx(0.0101)_2\approx(0.011)_2$。

如果一个十进制数既有整数部分又有小数部分，可将整数部分和小数部分分别进行转换，然后合并就可得到结果。

【例 1-7】将$(24.625)_{10}$转换成二进制。

解：$(24.625)_{10}=(24)_{10}+(0.625)_{10}$

$=(11000)_2+(0.101)_2$

$=(11000.101)_2$

1.2 带符号数的表示

1.2.1 原码及其运算

日常书写时在数值前面用“+”号表示正数，“-”号表示负数，这种带符号二进制数称为真值。在计算机处理时，必须将“+”和“-”转换为数码，符号数码化的数称为机器数。一般将符号位放在最高位，用“0”表示符号为“+”，用“1”表示符号为“-”。根据数值位的表示方法不

同,有三种类型:原码、反码和补码。

原码是指符号位用 0 表示正,1 表示负;数值位与真值一样,保持不变。

如已知 $X_1=+1101$,$X_2=-1101$,则$[X_1]_{原}=01101$,$[X_2]_{原}=11101$。

已知 $X_3=+0.1011$,$X_4=-0.1011$,则$[X_3]_{原}=0.1011$,$[X_4]_{原}=1.1011$。

可以用下面的公式来将真值转换为原码(含一位符号位,共 n 位)。

整数:

$$[N]_{原}=\begin{cases}N & 0\leqslant N<2^{n-1}\\ 2^{n-1}-N & -2^{n-1}<N\leqslant 0\end{cases}$$

表示范围为:$-127\sim+127$(8 位整数时)。

注意整数"0"的原码有两种形式,即$[+0]_{原}=00\cdots0$ 和$[-0]_{原}=10\cdots0$。

纯小数:

$$[N]_{原}=\begin{cases}N & 0\leqslant N<1\\ 1-N & -1<N\leqslant 0\end{cases}$$

纯小数"0.0"的原码也有两种形式。

原码的优点是容易理解,它和代数中的正负数的表示方法很接近。但原码的缺点是"0"的表示有两种;原码的运算规则复杂,$X=Y+Z$,但$(X)_{原}\neq(Y)_{原}+(Z)_{原}$。

例如,$(+1000)_2=(+1011)_2+(-0011)_2$,但是$(+1011)_{原}=01011$,$(-0011)_{原}=10011$,它们直接相加不等于$(+1000)_2$ 的原码(01000),所以为了使结果正确,原码的加法规则为:

1) 判断被加数和加数的符号是同号还是异号。

2) 同号时,做加法,结果的符号就是被加数的符号。

3) 异号时,先比较被加数和加数的数值(绝对值)的大小,然后由大值减去小值,结果的符号取大值的符号。

由于原码的运算规则复杂,为了简化机器数的运算,因此需要寻找其他表示负数的方法。

1.2.2 反码及其运算

反码的符号位用 0 表示正,1 表示负;正数反码的数值位和真值的数值位相同,而负数反码的数值位是真值的按位变反。

可以用下面的公式来将真值转换为反码(含一位符号位,共 n 位)。

整数:

$$[N]_{反}=\begin{cases}N & 0\leqslant N<2^{n-1}\\ (2^n-1)+N & -2^{n-1}<N\leqslant 0\end{cases}$$

表示范围为:$-127\sim+127$(8 位整数时)。

注意整数"0"的反码有两种形式,即$[+0]_{反}=00\cdots0$ 和$[-0]_{反}=11\cdots1$。

纯小数:

$$[N]_{反}=\begin{cases}N & 0\leqslant N<1\\ (2-2^{-m})+N & -1<N\leqslant 0,m=n-1\end{cases}$$

同理,纯小数"0.0"的反码也有两种形式,即$[+0.0]_{反}=[+.0]_{反}=00\cdots0$ 和$[-0.0]_{反}=[-.0]_{反}=11\cdots1$。纯小数中的小数点约定在符号位后面,其中的小数点和小数点前面的 0 省略不写。

采用反码进行加、减运算时，无论进行两数相加还是两数相减，均可通过加法实现。

$$[X_1+X_2]_{反}=[X_1]_{反}+[X_2]_{反}$$

$$[X_1-X_2]_{反}=[X_1]_{反}+[-X_2]_{反}$$

运算时符号位和数值位一起参加运算。当符号位有进位产生时，应将进位加到运算结果的最低位，才能得到最后结果，称之为“循环相加”（或“循环进位”）。

【例 1-8】已知 $X_1=+0.1110$，$X_2=+0.0101$，求 X_1-X_2。

解：X_1-X_2 可通过反码相加实现。运算如下：

$$\begin{aligned}[X_1-X_2]_{反}=[X_1]_{反}+[-X_2]_{反}&=0.1110+1.1010\\&=10.1000\quad\text{（符号位上有进位）}\\&=0.1001\end{aligned}$$

在反码中，$[X]_{反}\rightarrow[-X]_{反}$ 的方法是符号位连同数值位一起 0 变 1，1 变 0。

如 $[X]_{反}$ 为 0.0101，则 $[-X]_{反}$ 为 1.1010。

1.2.3 补码及其运算

补码是符号位用 0 表示正，1 表示负，正数补码的数值位和真值相同，而负数补码的数值位是真值的按位变反，再最低位加 1。

如 $X_1=+1101$，$X_2=-1101$，则 $[X_1]_{补}=01101$，$[X_2]_{补}=10011$。

$X_3=+0.1011$，$X_4=-0.1011$，则 $[X_3]_{补}=0.1011$，$[X_4]_{补}=1.0101$。

可以用下面的公式来将真值转换为补码（含一位符号位，共 n 位）。

整数：

$$[N]_{补}=\begin{cases}N & 0\leqslant N<2^{n-1}\\ 2^n+N & -2^{n-1}\leqslant N<0\end{cases}$$

表示范围为：$-128\sim+127$（8 位整数时），补码 10000000 表示 $-128(-2^7)$。

整数“0”的补码只有一种形式，即 00…0。

纯小数：

$$[N]_{补}=\begin{cases}N & 0\leqslant N<1\\ 2+N & -1\leqslant N<0\end{cases}$$

采用补码进行加、减运算时，无论进行两数相加还是两数相减，均可通过加法实现。

$$[X_1+X_2]_{补}=[X_1]_{补}+[X_2]_{补}$$

$$[X_1-X_2]_{补}=[X_1]_{补}+[-X_2]_{补}$$

运算时符号位和数值位一起参加运算，不必处理符号位上的进位（即丢弃符号位上的进位）。

【例 1-9】已知 $X_1=+0.1110$，$X_2=+0.0101$，求 X_1-X_2。

解：X_1-X_2 可通过补码相加实现。运算如下：

$$\begin{aligned}[X_1-X_2]_{补}=[X_1]_{补}+[-X_2]_{补}&=0.1110+1.1011\\&=10.1001\quad\text{（丢弃符号位上的进位）}\\&=0.1001\end{aligned}$$

当真值用补码表示时，补码加法的规律和无符号数的加法规律完全一样，因此简化了加法器的设计。

从$[X]_{补}$求$[-X]_{补}$的方法是符号位连同数值位一起变反，尾数再加1。

【例1-10】已知X=+100 1001，求$[X]_{补}$和$[-X]_{补}$。

解：$[X]_{补}$= 0100 1001

$[-X]_{补}$=1011 0110+1=1011 0111

1.2.4 符号位扩展

在实际工作中，有时会遇到两个不同位长的数的运算，如将8位与16位机器数相加，为确保运算正确，两者在运算之间应将符号位对齐。例如，将8位与16位机器数相加，则应将8位数扩展为16位数，一种容易理解的方法是先根据机器数得到真值，在真值前面添8个0，然后转变为机器数。

1.3 数的浮点表示

当一个数既有整数部分，又有小数部分时，如何在机器里表达呢？我们知道，可以将$(353.75)_{10}$表示为0.35375×10^3的形式，所以在机器里也可以将二进制数表示为这种浮点数的形式。其一般形式为$N=2^J\times S$，其中2^J称为N的指数部分，J称为阶码，表示小数点的位置，S为N的尾数部分，表示数的符号和有效数字。阶码的符号位称为阶符J_f，尾数的符号位称为尾符S_f。

如$N=(-0.00001101)_2=2^{(-4)_{10}}\times(-0.1101)_2=2^{(-100)_2}\times(-0.1101)_2$

规格化数是使尾数最高数值位非0，可以提高运算精度。例如

$(1011)_2=(10000)_2\times(0.1011)_2=2^{(4)_{10}}\times(0.1011)_2=2^{(100)_2}\times(0.1011)_2$，如果表示成$2^{101}\times0.01011$形式，则尾数需要更多的符号位才能保持精度不变。如果尾数的数值部分只有4位，则会产生误差。

在规格化数中，用原码表示尾数时，使小数点后的最高数据位为1；用补码表示尾数时，使小数点后的数值最高位与数的符号位相反。

两个浮点数作加减运算时要先对阶。

【例1-11】已知$N_1=2^{011}\times0.1001$，$N_2=2^{001}\times0.1100$，求N_1+N_2。

解：先对阶：$N_2=2^{001}\times0.1100=2^{011}\times0.0011$（小数点左移2位，阶码加2）

$$\begin{aligned}N_1+N_2&=2^{011}\times0.1001+2^{011}\times0.0011\\&=2^{011}(0.1001+0.0011)\\&=2^{011}\times0.1100\end{aligned}$$

两个浮点数作乘除法，其规则如下：

若$N_1=2^{j_1}\times S_1$，$N_2=2^{j_2}\times S_2$，则

$$\begin{aligned}N_1\times N_2&=(2^{j_1}\times S_1)\times(2^{j_2}\times S_2)\\&=2^{(j_1+j_2)}\times(S_1\times S_2)\\N_1\div N_2&=2^{(j_1-j_2)}\times(S_1\div S_2)\end{aligned}$$

1.4 数和字符的编码

数字系统中的信息有两类：一类是数码信息，另一类是代码信息。数码信息的表示方法如