普通高等教育“十一五”国家级规划教材

数字逻辑电路

第二版

刘常澍　主编

张　涛　王　炜　副主编

刘常澍
赵雅兴
张　涛
王　炜
雷淑英　编著

高等教育出版社
HIGHER EDUCATION PRESS

普通高等教育"十一五"国家级规划教材

数字逻辑电路

Shuzi Luoji Dianlu

第二版

刘常澍　主编
张　涛　王　炜　副主编
刘常澍　赵雅兴　张　涛　王　炜　雷淑英　编著

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容简介

本书是普通高等教育“十一五”国家级规划教材，是在本书第一版试用的基础上修订而成的。

全书共有11章内容：数字逻辑的基础知识，晶体管开关及门电路，组合逻辑电路，集成触发器，时序逻辑电路，中规模集成时序逻辑电路及其应用，存储器与可编程逻辑器件，硬件描述语言 VHDL，可测性设计及边界扫描技术，波形变换与产生电路，数模与模数转换。

本课程是电子信息类专业的主要技术基础课。书中内容的基础理论部分深入浅出，注重实践性，备有大量例题和习题。本书采用国家标准图形符号，在出现符号的地方对其所表示的意义进行简要地解释，读者在学习本书过程的同时逐渐学会识读常用的逻辑符号。

本书适合高等工科院校电子信息、通信、自动化等专业作为技术基础课教材，也可供其他相关专业选用和社会读者阅读。

图书在版编目（CIP）数据

数字逻辑电路/刘常澍主编．—2版．—北京：高等教育出版社，2010．12

ISBN 978-7-04-030696-5

Ⅰ．①数… Ⅱ．①刘… Ⅲ．①数字电路：逻辑电路-高等学校-教材 Ⅳ．①TN79

中国版本图书馆 CIP 数据核字（2010）第198281号

策划编辑 吴陈滨 **责任编辑** 王莉莉 **封面设计** 李卫青 **责任绘图** 郝 林
版式设计 马敬茹 **责任校对** 金 辉 **责任印制** 韩 刚

出版发行	高等教育出版社	**购书热线**	010-58581118
社　　址	北京市西城区德外大街4号	**咨询电话**	400-810-0598
邮政编码	100120	**网　　址**	http://www.hep.edu.cn http://www.hep.com.cn
经　　销	蓝色畅想图书发行有限公司	**网上订购**	http://www.landraco.com http://www.landraco.com.cn
印　　刷	天津新华二印刷有限公司	**畅想教育**	http://www.widedu.com
		版　　次	2008年2月第1版 2010年12月第2版
开　　本	787×1092　1/16		
印　　张	26.75	**印　　次**	2010年12月第1次印刷
字　　数	660 000	**定　　价**	38.80元

物料号　30696-00

第二版前言

根据电子信息技术领域对数字电路EDA（电子设计自动化）技术应用越来越普及的发展趋势，考虑到数字器件的发展和数字电路设计理念的变更，作为普通高等教育“十一五”国家级规划教材，本书在第一版的基础上作了如下调整：

1. 由于大规模集成电路的发展以及可编程器件的广泛应用，使得施密特电路、单稳态电路和多谐振荡器电路的形式已经发生了变化，但是集成触发器是必须让学生掌握的基本知识。为此，将集成触发器的有关内容单独设一章，采用传统分析方法的波形变换与产生电路的内容另设一章，供选用。

2. 时序逻辑电路部分着重基本概念和时序逻辑电路特点的介绍。因中规模集成器件的使用将会逐渐减少，代之以可编程逻辑器件等，时序逻辑电路的设计理念和设计方法也随之在改变。尽管如此，但在利用可编程逻辑器件设计数字电路系统的编程过程中，由于设计中常常会调用以库文件形式存在的中规模数字器件，所以中规模数字器件时序逻辑电路的内容又是不可或缺的，故将中规模数字器件时序逻辑电路单独设一章，并加进了少量的内容和例题，供读者选学。

3. 在可编程逻辑器件章节加强了有关开发平台的内容介绍，以跟随不断发展的EDA新技术。

4. 对第一版书中个别欠妥的内容作了修订，有些图形格式作了调整，使之更加合理以及在形式上保持一致。

本书内容共11章：数字逻辑的基础知识，晶体管开关及门电路，组合逻辑电路，集成触发器，时序逻辑电路，中规模集成时序逻辑电路及其应用，存储器与可编程逻辑器件，硬件描述语言VHDL，可测性设计及边界扫描技术，波形变换与产生电路，数模与模数转换。

考虑到各学校课程设置和要求不尽一致，本书没有将可选讲的部分标出，在选讲时可根据各校的教学情况进行取舍。

本书的编写情况如下：第7章由张涛执笔，第8章由赵雅兴执笔，第11章由王炜执笔，第2章由雷淑英执笔，其余章节由刘常澍执笔并完成全书的统稿工作。在全书的编写过程中一直得到赵雅兴教授的关心与指导，在此表示感谢。

鉴于作者的水平有限，不足甚至错误在所难免，恳请读者指正。

作者

2010年7月于天津

第一版前言

近年来，电子信息领域的技术发展日益加快，数字集成电路的复杂程度、集成度越来越高，因而对EDA（电子设计自动化）的需求越来越高，对于相关基本技能的提高要求也越来越迫切。

教育部高等学校电子电气基础课程教学指导分委员会对于“数字电路与逻辑设计”课程拟定了教学基本要求，要求中指出，本课程是电子信息类专业的主要技术基础课。其作用与任务是：使学生掌握数字电路的基本分析方法和逻辑设计方法。

本书在多年实际教学的基础上编写而成，考虑到不同院校的课程设置不同，本书没有将可选讲部分标出，在选讲时可根据各校教学要求进行取舍，并建议如下：

1. 第1、2、4章内容分别为数字逻辑的基础知识，晶体管开关及门电路，触发器与波形变换、产生电路，这三章以及第6章存储器与可编程逻辑器件是数字电路的基础内容，教学中选取的内容应以够用为度。

2. 第3章组合逻辑电路与第5章时序逻辑电路两章的内容较多，可在课内讲授基本概念和原理部分，基本内容以外的扩展和例题可以让学生自学。

3. 将硬件描述语言（VHDL）写成独立的第7章，并比较详细地讲述具体内容与列举大量实例，使读者通过本章能够较为全面地学习VHDL。若这部分内容单独设立一门课，则本章可作为参考资料。

4. 第8章可测性设计及边界扫描技术是综述性内容，可使读者了解近年出现的应用最为普遍的边界扫描技术。

5. 第9章数模与模数转换是模拟、数字的结合电路，可将其安排在诸如智能仪器、测控电路等课程内讲授。

书中采用国家标准图形符号，在出现符号的地方对其所表示的意义进行简要解释，使读者在学习本书的过程中逐渐学会识读常用的逻辑符号。

书中备有大量例题和习题，为读者提供了较多的参考和练习。

本书第2章由雷淑英编写，第7章由赵雅兴编写，第9章由王炜编写，其余章节的编写和统稿工作全部由刘常澍完成。赵雅兴教授在本书的编写过程中给予了指导。

北京联合大学李哲英教授审阅了本书的全部书稿，提出了很多改进意见，编者在此表示由衷的感谢。还要感谢李学花对本书第6章编写所做的工作。

限于编者的水平，本书难免存在缺点或错误，恳请读者批评指正并及时与我们联系。

编者

2007年9月于天津

目　　录

第1章　数字逻辑的基础知识

引言

21世纪是信息化时代，数字化是进入信息化时代的必要条件。数字化的内涵即将信息用数字0、1编码来表述，对于任意的信息传输、处理、存储均用0、1码的形式进行。

0和1两个数码用电路中的两种对立状态表示，如电压的高低、电流的大小、脉冲的有无等，基于这种原理的电路即是数字电路，它通过对电路中状态的变换，来对其代表的信息进行运算、处理、控制、变换等操作。这种电路表现出许多优点：

① 电路仅稳定工作在不连续的、特征差别大的两个对立状态下，则电路的可靠性和稳定性非常高。

② 从理论上讲，信号在处理的过程中不会产生失真。

③ 信息的传输、运算、处理和保存变得更为方便。

④ 对电路的实时控制准确有效。

⑤ 与计算机及外围电路兼容，便于利用计算机进行运算、处理和控制。

⑥ 数字电路可以很方便地级联和扩展，中、大规模数字集成电路的生产和应用都呈现出广阔的空间。

正是基于上述原因，数字电路在近年来得到长足的进步，使得电子设备的可靠性和准确性得到极大提高。

数字电路越来越广泛地应用于工业、交通、军事、广播电视等领域，近些年消费电子的发展方兴未艾，数字电路在此领域大放异彩。

本课程是许多学科的技术基础，也是相关课程的理论基础，如计算机原理、计算机网络、数字通信、数字信号处理、智能仪器及测量、现代自动控制等。

1.1　数字电路的信号

1.1.1　模拟量与数字量

自然界中存在的物理量，如海拔高度、温度、气压、质量、距离、物种数量等，各种各样，千变万化，根据各种量的变化特性，可以概括为模拟量和数字量两大类。模拟量是在时间和数值上连续变化的物理量，如海拔、温度、气压等。数字量在时间和数值上都是不连续的，或者说是离散的物理量，例如对生产线上产品的计件、物种数量的统计等。

模拟量的数字化是对模拟量分离取值的过程。例如气温，一般只按整摄氏度数记录，最小

的统计单位是“摄氏度”，而实际气温变化是连续的，所以记录气温的过程实际是对模拟量数字化的过程。

数字量有一个最小数量单位，数值的不同就是这个最小单位的整数倍不同，而小于这个最小数量单位的数值是没有意义的。例如人口统计中的最小数量单位是1人。

1.1.2 数字电路及其信号

表示数字量的信号称为数字信号，工作在数字信号下的电子电路称为数字电路。数字信号只有**0**和**1**两个数码，用电路的两个对立的状态表示这两个数码（如电压的高与低、晶体管的饱和与截止、开关的通与断等）。由于数字电路中只有**0**和**1**两个数字，所以在数字运算时采用二进制数制，与人们习惯的十进制有所不同。相对于模拟电路而言，数字电路具有误差小、抗干扰性强、精度高、容易保存等优点。

1.2 数字电路所用的数制

1.2.1 二进制数

用数字表示数量即计数，用一组数码采用一定的计数规则表示数量，即是数制。例如，日常生活中通常采用的数制是十进制，即科学计数制，基数是10，它有0、1、…、9共10个数字符号，计数的规则是“逢十进一”。即在计数过程中，一旦计数满十，就向高位进一。可以用以10为底的幂级数表达式来表示一个n位整数、m位小数的十进制数

$$\begin{aligned}(D)_{10} &= k_{n-1}\times 10^{n-1}+k_{n-2}\times 10^{n-2}+\cdots+k_1\times 10^1+k_0\times 10^0+\\ &\quad k_{-1}\times 10^{-1}+k_{-2}\times 10^{-2}+\cdots+k_{-m}\times 10^{-m}\\ &= \sum_{i=-m}^{n-1} k_i\times 10^i \end{aligned} \tag{1-1}$$

式中，k_i为第i位的系数；10为基数；10^i为第i位的权。处在不同位的数，主要体现在权的大小不同，如十位、百位、千位等；其次是系数不同。任意一个十进制数都可按位展开，即把每一位的位权值与该位的系数相乘，然后对所有位求和。例如

$$(1234.056)_{10}=1\times 10^3+2\times 10^2+3\times 10^1+4\times 10^0+0\times 10^{-1}+5\times 10^{-2}+6\times 10^{-3}$$

人类文明历史上曾用过不同的计数制，如十二进制、十六进制、六十进制等，按照上述方法，可以写出任意进制（N进制）数的表达式

$$\begin{aligned}(D)_N &= k_{n-1}\times N^{n-1}+k_{n-2}\times N^{n-2}+\cdots+k_1\times N^1+k_0\times N^0+\\ &\quad k_{-1}\times N^{-1}+k_{-2}\times N^{-2}+\cdots+k_{-m}\times N^{-m}\\ &= \sum_{i=-m}^{n-1} k_i\times N^i \end{aligned} \tag{1-2}$$

一个N进制数，基数是N，它有0、1、…、$N-1$共N个数字符号，以N为基数，计数的规则是“逢N进一”，可以用以N为底的幂级数表达式来表示一个N进制数。

在数字电路和计算机中，都采用二进制计数，即采用“逢二进一”的计数方法。同样，

可以用下式表示一个二进制数

$$(D)_2 = \sum_{i=-m}^{n-1} k_i \times 2^i \tag{1-3}$$

二进制的基数是 2，只有 **0** 和 **1** 两个数字符号，在数字电路和计算机中可以很方便地进行处理运算。但二进制数通常位数很多，而且与一般人们计数习惯和形式不一致，因而需将二进制数转换为十进制数，或反之，以利认识和应用，即进行数制转换。

1.2.2　十进制数和二进制数的互相转换

1. 十进制数 - 二进制数转换

十进制数转换到二进制数的过程分两步，对整数部分和小数部分分别进行转换，然后再合成得到结果。

整数部分的转换：一个十进制的整数 $(D)_{10}$ 总可以用二进制数展开，即

$$(D)_{10} = k_n 2^n + k_{n-1} 2^{n-1} + \cdots + k_1 2^1 + k_0 2^0$$

二进制数写为 $(k_n k_{n-1} \cdots k_0)_2$。若将 $(D)_{10}$ 除以 2，则得到的商为

$$k_n 2^{n-1} + k_{n-1} 2^{n-2} + \cdots + k_1$$

而余数即 k_0；然后，再将商数除以 2，则所得余数即 k_1；……将每次得到的商依次除以 2，取它们的余数 $k_0 \cdots k_n$，就可求出二进制数的每一位了。所以，十进制数整数部分转换为二进制数的过程是采用逐次除以基数 2、再取余数的方法。

例 1-1　将十进制数 $(27)_{10}$ 转换成二进制数。

解：按上述连续除以 2 的步骤进行

$$\begin{array}{r|l l}
2 & 27 \quad 商 & \quad 余数 \\
2 & 13 & \cdots\cdots \mathbf{1} = k_0 \ 最低位 \\
2 & 6 & \cdots\cdots \mathbf{1} = k_1 \\
2 & 3 & \cdots\cdots \mathbf{0} = k_2 \\
2 & 1 & \cdots\cdots \mathbf{1} = k_3 \\
 & 0 & \cdots\cdots \mathbf{1} = k_4 \ 最高位
\end{array}$$

故 $(27)_{10} = (\mathbf{11011})_2$。

小数部分的转换与整数部分的转换不同，是通过乘以基数 2 实现的。

若 $(D)_{10}$ 是一个十进制数的小数，对应的二进制小数为 $(0.k_{-1} k_{-2} \cdots k_{-m})_2$，应有

$$(D)_{10} = k_{-1} 2^{-1} + k_{-2} 2^{-2} + \cdots + k_{-m} 2^{-m}$$

如果将两边同乘以 2 得到

$$2(D)_{10} = k_{-1} + (k_{-2} 2^{-1} + k_{-3} 2^{-2} + \cdots + k_{-m} 2^{-m+1})$$

说明将小数 $(D)_{10}$ 乘以 2 后，等式右边所得乘积的整数部分即 k_{-1}，其余为小数部分。

取出 k_{-1}，将小数部分再乘以 2 又可以得到

$$k_{-2} + (k_{-3} 2^{-1} + \cdots + k_{-m} 2^{-m+2})$$

同样取出 k_{-2}，将小数部分再乘以 2，……当乘到取出 k_{-m} 后，就依次得到 $k_{-1} k_{-2} \cdots k_{-m}$。所

以，十进制数（小数部分）转换为二进制数的过程是采用逐次乘以基数 2、再取整数的方法。

例 1-2 将十进制小数 $(0.625)_{10}$转换成二进制小数。

解：按上述连续乘以 2 的步骤进行

$$
\begin{array}{lr}
 & 0.625 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{1}\cdots\cdots & \mathbf{1}.250 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{0}\cdots\cdots & \mathbf{0}.500 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{1}\cdots\cdots & \mathbf{1}.000
\end{array}
$$

故 $(0.625)_{10}=(0.\mathbf{101})_2$

不是所有的十进制小数连乘 2 后小数部分都能达到 0，有的会出现无限循环，有的可能二进制小数的位数很多，在这种情况下，转换达到一定精度的位数时舍去后面的小数即可。转换精度的确定，一般以十进制所要求的精度进行换算：二进制每 4 位小数相当于十进制 1 位小数的精度（$1/16\approx1/10$），二进制每 7 位小数相当于十进制 2 位小数的精度（$1/128\approx1/100$），二进制每 10 位小数相当于十进制 3 位小数的精度（$1/1024\approx1/1000$），等等，以此类推。

例 1-3 将 $(0.4)_{10}$转换成二进制小数。

解：

$$
\begin{array}{lr}
 & 0.4 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{0}\cdots\cdots & \mathbf{0}.8 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{1}\cdots\cdots & \mathbf{1}.6 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{1}\cdots\cdots & \mathbf{1}.2 \\
 & \times \quad 2 \\ \cline{2-2}
\text{取 } \mathbf{0}\cdots\cdots & \mathbf{0}.4 \\
\vdots & \vdots
\end{array}
$$

以下循环，故 $(0.4)_{10}=0.\mathbf{0110}\cdots$，取所需要的位数即可。

2. 二进制数-十进制数转换

将一个二进制数转换成十进制数的过程很简单，只需将二进制数按位权展开相加即可。

例 1-4 将二进制数 $(\mathbf{11010.011})_2$ 转换成十进制数。

解：

$$
\begin{aligned}
(\mathbf{11010.011})_2 &= \mathbf{1}\times2^4+\mathbf{1}\times2^3+\mathbf{0}\times2^2+\mathbf{1}\times2^1+\mathbf{0}\times2^0+\mathbf{0}\times2^{-1}+\mathbf{1}\times2^{-2}+\mathbf{1}\times2^{-3} \\
&= \mathbf{1}\times16+\mathbf{1}\times8+\mathbf{0}\times4+\mathbf{1}\times2+\mathbf{0}\times1+\mathbf{0}\times0.5+\mathbf{1}\times0.25+\mathbf{1}\times0.125 \\
&= (26.375)_{10}
\end{aligned}
$$

1.2.3 八进制数和十六进制数

用二进制数表示一个数的位数会很多，书写和识别都很不方便，因而将二进制数用八进制数或十六进制数表示被广泛应用。

八进制数有0、1、…、7共8个数字符号，根据数制规律可知，3位二进制数可表示成1位八进制数，因而转换时只需将二进制数的每3位对应转换成1位八进制数。带小数的二进制数转换为八进制数时，整数与小数应分别进行转换。即以小数点为基准，向左、向右每3位为一组（首尾不足3位的以0补足3位），每组对应转换成1位八进制数。

反过来，1位八进制数则对应3位二进制数。

例1-5 将$(10110.01011)_2$转换成八进制数。

解：

$$\begin{array}{cccc} (010 & 110. & 010 & 110)_2 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ (\ 2 & 6. & 2 & 6)_8 \end{array}$$

所以$(10110.01011)_2=(26.26)_8$

十六进制数由0、1、…、9和A、B、C、D、E、F共16个数字组成，其中A、…、F分别等值于十进制数中的10、…、15。4位二进制数可表示为1位十六进制数，转换方法可参照八进制数的转换方法进行，但需将二进制数的每4位对应转换成1位十六进制数。

例1-6 将二进制数$(1011010.01111)_2$转换成十六进制数。

解：$(1011010.01111)_2=(5A.78)_{16}$

欲将八进制数和十六进制数转换成十进制数，只需像二进制数转换时那样，按位权展开求和即可。

例1-7 将十六进制数$(5A.78)_{16}$转换成十进制数。

解：$(5A.78)_{16}=5\times16^1+10\times16^0+7\times16^{-1}+8\times16^{-2}=(90.46875)_{10}$

十进制数直接转换成八进制数和十六进制数比较繁琐，有时计算容易出错。可以将十进制数先转换成二进制数，然后再转换为八进制数或十六进制数。

例1-8 将$(43.3125)_{10}$转换成八进制数和十六进制数。

解：$(43.3125)_{10}=(101011.0101)_2=(53.24)_8=(2B.5)_{16}$

1.3 数字电路常用的码制与编码

计算机可以存储和处理很多不同的信息，而实际上机器只能识别二进制码，因而要用数码来表示各种各样的信息。

用数码来表示信息的方法很多，建立这种对应关系的过程称为编码，编码所依据的不同的编码规则就称为码制。所以，数码不仅可以表示数量的不同大小，而且还能表示不同的事物。表示事物的这些数码只是事物的代号，而没有数量的含义。例如，不同的电话号码，只是区别不同的电话终端，并不表示数值大小。代替不同事物的数码称为代码。当然，数量也可以用数码来表示，但可以有不同的表示规则，甚至同一个数可以有不同的代码表示形式，因为依据的码制不同。

1.3.1 原码、反码和补码

书写一个带符号数时，可以在数的前面加上一个符号，如+3、-0.5等。而在计算机中

的正、负号是用数码来表示的。通常带符号数的最高位为符号位：该位为 0 表示是正数，为 1 则表示是负数。符号位后面的数码表示数值。在计算机中，带符号的二进制数通常有原码、反码和补码三种表示方法。

1. 原码

用原码表示带符号数时，只需将符号位用 0 或 1 表示即可，后面的数字不变。如有两个带符号数 $N_1 = +1100101$ 和 $N_2 = -1100101$，用原码表示为

$$N_1 = (01100101)_{原} \qquad N_2 = (11100101)_{原}$$

原码表示方法简单，但在运算时比较麻烦（尤其是减法运算），运算过程加长，造成电路成本增加和运算速度降低。为此，可采用补码表示方法来加以改善。

2. 反码

用反码表示一个数时，正数的表示方法与原码表示方法相同；如果是负数，最高位仍为符号位，记为 1，其余各位将原数码按位取反，即 0 变成 1，1 变成 0。例如

$$N_1 = +1100101 = (01100101)_{反}$$

$$N_2 = -1100101 = (10011010)_{反}$$

3. 补码

在补码表示中，正数的表示方法也与原码表示方法相同；如果数 N 是负数，符号位仍为 1，其余各位取其对应于 2^m 的补数（数 N 的补数为 $2^m - N$，这里 $2^{m-1} < N \leqslant 2^m$）。由于是二进制数，求负数的补码可先将其变成反码，然后在最低位加 1 即可。例如

$$N_1 = +1100101 = (01100101)_{补}$$

$$N_2 = -1100101 = (10011011)_{补}$$

三种表示方法如表 1－1 所示。

表 1－1 原码、反码和补码的对应关系

十进制	二进制		
	原码	反码	补码
7	0111	0111	0111
6	0110	0110	0110
5	0101	0101	0101
4	0100	0100	0100
3	0011	0011	0011
2	0010	0010	0010
1	0001	0001	0001
0	0000	0000	0000
-0	1000	1111	—
-1	1001	1110	1111
-2	1010	1101	1110

续表

十进制	二　进　制		
	原码	反码	补码
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	11000	10111	1000

注：各编码的第1位是符号位；补码列中的空格表示 -0 无编码； -8 的原码和反码应扩展为5位。

原码表示方法简单，符合人们通常的计数习惯，但**0**的表示不是唯一的。补码的转换虽然稍微麻烦一些，但是**0**的表示方法是唯一的，主要是用于将减法运算变为用补码的加法来实现。

4. 小数的表示与字长

不论原码、反码或补码，都是按进位关系和整数连续排列起来的，但是当两个数进行运算（指的是定点运算）时，必须整数和小数字长分别相等，即两个数的整数位数应相同，小数的位数也应相同。如果两个数的位数不同或在运算时产生进位超出了原来的位数（称为溢出），就应增加位数（扩展位数），即增加数的字长。

扩展位数应按整数、小数分别扩展，扩展的方法如下。

（1） 正数

无论带符号和不带符号的原码、反码和补码表示的正整数，一律在高位填**0**补足所少的位数；正小数，一律在低位填**0**补足所少的位数。

（2） 原码带符号的负数

整数在符号位后的高位填**0**补足所少的位数，小数在低位填**0**补足所少的位数。

（3） 反码带符号的负数

整数在符号位后的高位填**1**补足所少的位数，小数在低位填**1**补足所少的位数。

（4） 补码带符号的负数

整数在符号位后的高位填**1**补足所少的位数，小数在低位填**0**补足所少的位数。

例1-9　写出正数7.5、4.6875的整数、小数各4位字长以及各8位字长的原码、反码和补码。

解：

4位 $(7.5)_{10}$原码、反码、补码全为：$(\mathbf{0111.1000})_2$

4位 $(4.6875)_{10}$原码、反码、补码全为：$(\mathbf{0100.1011})_2$

8位 $(7.5)_{10}$原码、反码、补码全为：$(\mathbf{00000111.10000000})_2$

8位 $(4.6875)_{10}$原码、反码、补码全为：$(\mathbf{00000100.10110000})_2$

例1-10　写出负数 -7.5、 -4.6875 的整数、小数各4位字长以及各8位字长的原码、

反码和补码。

解：

4位原码：$(-7.5)_{10}=(1111.1000)_{原}$　　$(-4.6875)_{10}=(1100.1011)_{原}$

8位原码：$(-7.5)_{10}=(10000111.10000000)_{原}$　　$(-4.6875)_{10}=(10000100.10110000)_{原}$

4位反码：$(-7.5)_{10}=(1000.0111)_{反}$　　$(-4.6875)_{10}=(1011.0100)_{反}$

8位反码：$(-7.5)_{10}=(11111000.01111111)_{反}$　　$(-4.6875)_{10}=(11111011.01001111)_{反}$

4位补码：$(-7.5)_{10}=(1000.1000)_{补}$　　$(-4.6875)_{10}=(1011.0101)_{补}$

8位补码：$(-7.5)_{10}=(11111000.10000000)_{补}$　　$(-4.6875)_{10}=(11111011.01010000)_{补}$

例1-11　有两个数：$N_1=+1011000$，$N_2=+0100110$，计算 N_1-N_2 的值。

解：

$$N_1-N_2=N_1+(-N_2)=(+N_1)_{补}+(-N_2)_{补}$$
$$=01011000+11011010$$

用竖式表示

$$\begin{array}{r} 01011000 \\ +\quad 11011010 \\ \hline \boxed{1}\,00110010 \end{array}$$

竖式结果中框内的1是溢出，已超出计算所用的位数，舍去溢出，所得的结果是两个数差的补码00110010，将结果还原为带符号数，则

$$N_1-N_2=+0110010$$

例1-12　有两个数：$N_1=+0100110$，$N_2=+1010010$，计算 N_1-N_2 的值。

解：

$$N_1-N_2=N_1+(-N_2)=(+N_1)_{补}+(-N_2)_{补}$$
$$=00100110+10101110$$

用竖式表示

$$\begin{array}{r} 00100110 \\ +\quad 10101110 \\ \hline 11010100 \end{array}$$

竖式所得的结果是两个数差的补码11010100，还原为带符号数，则

$$N_1-N_2=-0101100$$

从以上两个例子看，用补码的加法做减法运算，则将两个加数都变成补码连同符号一起相加，得到的和还是补码，运算的结果还原成带符号的数（即减法的差）。

1.3.2 BCD码（二-十进制编码）

人们通常熟悉的是十进制数，而数字电路中用的是二进制的计数方法，为此用二进制数表示十进制数，即采用二-十进制（Binary Coded Decimal，BCD）码。1位十进制数最少要4位二进制数来表示，即可以从0000、…、1111中取10个组合数来表示0、1、…、9。常用的一种编码方法是取自然二进制数的前10个数，称为8421BCD码。它是一种有权码（也称恒权码、定权码、含权码），即每一位的1代表一个固定的权，将二进制数中所有1代表的数值相加就可得到相应的十进制数。这种码表示的意义明显，转换也很方便，因而应用最多。除了8421码外，还有几种常用BCD码，各有其编码特点和用途，见表1-2。