第 1 章 数字逻辑基础

进入 21 世纪，集成电路已经广泛应用于人类社会的生活和生产之中，涉及信息、 生物、新材料、能源、激光、自动化、航天、海洋等几乎所有科学技术领域。小到移动电话，电视、个人电脑等，大到雷达、航天飞机、人造卫星等，几乎所有电器和包含电子部件的装备中都包含集成电路。所谓集成电路，也称为微电路、微芯片、芯片，在电子学中是一种把电路小型化的方式，通常制造于半导体晶圆表面上。从处理信号的形式看，集成电路可以分为处理模拟信号的模拟电路和处理数字信号的数字电路。由于在结构与功能方面所特有的优点，数字电路伴随着计算机和数字通信等技术的发展和广泛应用，正在被越来越多的人了解和掌握。数字逻辑课程的主要目的是使学生了解和掌握从对数字电路提出要求开始，一直到用电路实现所需逻辑功能为止的整个过程的完整知识。作为该课程的开始，本章将介绍有关数字逻辑学习的一些基本的预备概念，内容包含数字逻辑概述和数码表示等。

1.1 概述

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。

1.1.1 数字逻辑研究的对象及方法

1．数字电路与数字系统

在自然界中，所有物理量都可以分为模拟量和数字量两种。模拟量是指取值连续的物理量，如温度、速度和压强等。数字量是指取值分立的物理量，如人口数量、书本页数以及羊群数目等。在电路中，电信号同样可以分为模拟信号和数字信号。

(1) 模拟信号和数字信号

当用电路表达物理量时，必须先将物理量变换为电路易于处理的信号形式，一般用变化的电压（或电流）表示。模拟信号是一种连续信号，任一时间段都包含了信号的信息分量。如图 1-1 所示的模拟信号为正弦电压信号。

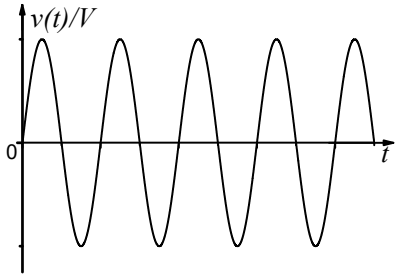

图 1-1 正弦电压信号的波形图

而数字信号是离散的，一方面，其变化在时间上是不连续的，总是发生在一系列离散的瞬间；另一方面，数字信号的取值也是分立的，只包含有限个数值，属于一种脉冲信号。应用最广泛的数字信号是二值信号，只有“0”和“1”两种取值。除了二值信号外，还存在多进制电压信号。图 1-2 (a) 所示的是一个二值电压信号的波形图，该信号只有 0V 和＋5V 两种电压取值，其中低高电平可分别用以表示“0”和“1”两种逻辑值。如果用“0”表示低电平，用“1”表示高电平，称为正逻辑表示；若用“0”表示高电平，用“1”表示低电平，则称为负逻辑表示。图 1-2 (b) 也展示了多进制电压信号的波形图。

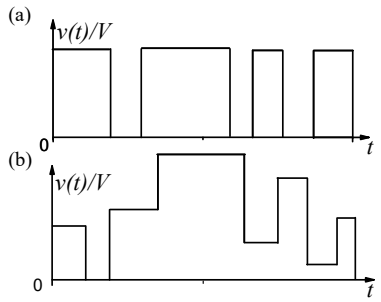

图 1-2　数字信号的波形图 (a) 二进值电压信号；(b) 多进值电压信号

(2) 模拟电路和数字电路

与之相对应的，处理模拟信号的电路是模拟电路，如集成运算放大器；处理数字信号的电路是数字电路，例如编码器、译码器和计数器。从概念上讲，凡是利用数字技术对信息进行处理、传输的电子系统均可称为数字系统。

相比于模拟电路，数字电路具有以下优点：

- 稳定性好。数字电路不像模拟电路那样易受噪声的干扰。
- 可靠性高。数字电路中只需分辨出信号的有无，因此电路的元件参数允许存在较大的变化（漂移）范围。
- 易于长期存储。数字信息可以利用某种媒介，如磁带、磁盘、光盘等进行长时期的存储。
- 便于计算机处理。数字信号的输出除了具有直观、准确的优点外，最主要的还是便于利用电子计算机来对于信息进行处理。
- 便于高度集成化。由于数字电路中基本单元电路的结构比较简单，并允许元件有较大的分散性，这不仅可把众多的基本逻辑单元集成在同一块硅片上，而且又能达到大批量生产所需要的良率。

2．数字电路的分析和设计

数字电路是以二值数字逻辑为基础的，输入和输出信号为离散数字信号，电子元器件工作在开关状态。数字电路响应输入的方式叫做数字逻辑，服从布尔代数的逻辑规律。因此，数字电路又叫做逻辑电路。

在数字电路中，人们关注的是输入、输出信号之间的逻辑关系。输入信号和输出信号分别被称为输入和输出逻辑变量，它们之间的因果关系可由逻辑函数来描述，其数学基础为逻辑代数或布尔代数。所谓数字分析，就是针对已知的数字系统，分析其工作原理、确定输入与输出信号之间的关系、明确整个系统及其各组成部件的逻辑功能。描述数字电路逻辑功能的常用方法有真值表、逻辑表达式、波形图、逻辑电路图等。

数字设计是与数字分析互逆的过程，即针对特定的功能需求，采用一定的设计手段和步骤，实现一个符合功能需求的数字系统。通常地，数字设计的层次由高到低可以分为系统级、

模块级、门级、晶体管级和物理级。最终数字系统的逻辑功能被表示为一组逻辑函数，进而可以利用逻辑门单元实现。逻辑门是实现基本逻辑运算的最小逻辑单元，用逻辑门实现逻辑功能是数字电路设计的基本内容之一。随着可编程逻辑器件 (Programmable Logic Device, PLD) 的广泛应用，硬件描述语言（Hardware Description Language, HDL）已成为数字系统设计的主要描述方式，目前较为流行的硬件语言有 VHDL 和 Verilog HDL 等。

1.1.2 数字电路的发展

数字技术的应用已经渗透到了人类生活和生产的各个方面。从计算机到家用电器，从手机到数字电话，以及绝大多数医用设备、军用设备、导航系统等，无不尽可能地采用数字技术。从概念上讲，凡是利用数字技术对信息进行处理、传输的电子系统均可称为数字系统。

1. 数字集成电路的发展

一方面，数字系统的发展很大程度上得益于器件和集成技术的发展。几十年来，半导体集成电路的发展印证了著名的摩尔定律，即每 18 个月，芯片的集成度提高一倍，而功耗下降一半。数字电路的发展经历了从电子管、半导体分立器件到集成电路等几个阶段，由于自身的独特优势其发展比模拟电路越来越快。从 60 年代开始，以双极型工艺制成了小、中规模逻辑器件。70 年代末，随着微处理器的出现，使数字集成电路的性能产生质的飞跃。数字集成器件所用的材料以硅材料为主，在高速电路中也使用化合物半导体材料，例如砷化镓等。逻辑门是数字电路中一种重要的逻辑单元电路。晶体管-晶体管逻辑门(Transistor-Transistor Logic, TTL)问世较早，其制作工艺经过不断完善，目前为主要的基本逻辑器件之一。随着互补金属氧化物半导体(Complementary Metal-Oxide-Semiconductor Transistor, CMOS) 制作工艺的发展，CMOS 器件广泛应用于各种数字电路，大有取代 TTL 器件的趋势。

另一方面，PLD 器件和电子设计自动化 (Electronic Design Automation, EDA) 技术的出现使数字系统的设计思想和设计方式发生了根本的变化。PLD 是作为一种通用集成电路生产的，其逻辑功能通过用户对器件编程来实现。一般地，PLD 的集成度很高，足以满足一般数字系统的功能需求。随着 PLD 器件的快速发展，集成度越来越高，速度也越来越快，并可以将微处理器、数字信号处理器 (Digital Signal Processor, DSP)、存储器和标准接口等功能部件全部集成其中，真正实现“系统芯片 (System On a Chip)”。EDA 技术以计算机为工具，设计者在 EDA 软件平台上用硬件描述语言 VHDL 完成设计文件，然后由计算机自动地完成逻辑编译、化简、分割、综合、优化、布局、布线和仿真，直至对于特定目标芯片的适配编译、逻辑映射和编程下载等工作。

总而言之，数字电路集成规模越来越大，并将硬件与软件相结合，使器件的逻辑功能更加完善，使用更加灵活，功能也更加强大。

2. 数字集成电路的发展趋势

目前，数字集成电路正朝着大规模、低功耗、高速度、可编程、可测试和多值化方向发展。

(1) **大规模** 随着集成电路技术的飞速发展，一块半导体硅片上能够集成百万个以上的逻辑门，新兴的纳米技术进一步扩大了数字电路的集成规模。集成规模的提高不仅缩小了数字系统的体积，降低了功耗与成本，而且显著提升了可靠性。

(2) **低功耗** 功耗是制约电子设备研制、生产、推广以及使用的一个重要因素，在很大程度上取决于所使用的芯片或模块，功耗的降低大大扩展了数字集成电路的应用领域。

(3) **高速度** 当今社会处于信息大爆炸的时代，人们对信息处理速度的要求越来越高。

以电子计算机为例，我们亟需越来越快的运行速度。虽然这种高速度在很大程度上依赖于并行处理技术，但集成芯片本身的速度在不断提高也是毋容置疑。

(4) **可编程** 传统的标准中、大规模集成电路是一种通用性集成电路。当使用这种集成电路设计复杂数字系统时，所需要的逻辑模块数量和种类往往比较多，这不仅增加了系统的体积和功耗，也降低了系统的可靠性，而且给器件的保存、电路和设备的调试、知识产权的保护带来了难题。可编程的数字集成电路可以很好地解决上述问题。

(5) **可测试** 数字集成电路的规模越来越大，功能也越来越复杂。为了便于数字系统的使用与维护，要求可以方便地对逻辑模块进行功能测试和故障诊断，即具有“可测试性”。

(6) **多值化** 传统的数字集成电路是一种二值电路，在信号的产生、存储、传输、识别、处理等方面具有明显优势。为了进一步提升集成电路的信息处理能力，除了在速度上下功夫外，还可采用多值逻辑电路。

1.1.3 数字电路的分类

根据不同的区分角度，数字电路可以分成不同类型。例如根据电路结构的不同，数字电路可分为分立元件电路和集成电路两大类；根据所用器件制作工艺不同，数字电路可分为双极型 (TTL 型) 和单极型 (MOS 型) 两类；根据电路的结构和工作原理不同，数字电路可分为组合逻辑电路和时序逻辑电路。

1. 电路结构的划分

根据电路结构不同，数字电路可分为分立元件电路和集成电路两大类。分立元件电路是由二极管、三极管、电阻、电容等元件组成的电路；集成电路是将上述元件通过半导体制造工艺集成于一块芯片上。根据集成度不同，可分为小规模、中规模、大规模、超大规模集成电路。

小规模集成电路 (Small-Scale Integration, SSI)：集成度在 100 个元件以内或 10 个门电路以内。例如常见的与门、或非门等逻辑实验电路。

中规模集成电路 (Medium-Scale Integration, MSI)：集成度在 100~1000 个元件之间，或在 10~100 个门电路之间。例如译码器、编码器以及数据选择器等。

大规模集成电路 (Large-Scale Integration, LSI)：集成度在 1000 个元件以上，或 100 个门电路以上。例如微处理器和小型控制器等。

超大规模集成电路 (Very Large-Scale Integration, VLSI)：集成度达十万个元件以上，或等效于一万个门电路。例如中央处理机 (Central Processing Unit)、数字化视频光盘 (Digital Video Disk, DVD) 解码器、大容量内存芯片等。

2. 制作工艺的划分

根据所用器件制作工艺不同，数字电路可分为双极型 (TTL 型) 和单极型 (MOS 型) 两类。双极型和单极型是针对组成集成电路的晶体管的极性而言的。

⑴ 双极型集成电路是由 NPN 或 PNP 型晶体管组成。由于电路中载流子有电子和空穴两种极性，故称为为双极型集成电路，即通常所说的 TTL 集成电路。

⑵ 单极型集成电路是由 MOS 场效应晶体管组成的。因场效应晶体管只有多数载流子参加导电，故称为单极晶体管，即平时说的 MOS 集成电路。

此外，还常常将单极型电路作输入电路，双极型晶体管作输出电路，构成 BIMOS 集成电路。

3. 工作原理的划分

根据电路的结构和工作原理不同，是否具有记忆，数字电路可分为组合逻辑电路和时序逻辑电路。

⑴ 组合逻辑电路：任意时刻的输出仅仅取决于该时刻的输入组合，而与输入信号作用前电路的原状态无关(与过去的输入无关)。常用的电路有编码器、译码器、数据选择器、加法器、数值比较器等。

(2)时序逻辑电路：任意时刻的输出不仅仅与该时刻的输入有关，而且还与电路的原状态有关(与过去的输入有关)。如图 1-3，它是由最基本的逻辑门电路加上反馈逻辑回路（输出到输入）或器件组合而成的电路，类似于含储能元件的电感或电容的电路，如触发器、锁存器、计数器、移位寄存器、储存器等电路都是时序电路的典型器件。

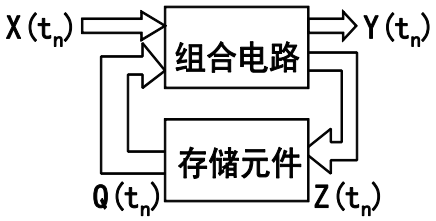

图 1-3　时序逻辑电路的示意图

1.2 数制及其转换

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。

1.2.1 进位计数制

一、十进制数的表示

人有 10 个手指和 10 个脚趾，所以在日常生活中人们通常采用十进制数来计数，每位数可用下列十个数码之一来表示，即 0、1、2、3、4、5、6、7、8、9。十进制的基数为 10，基数表示进位制所具有的数字符号个数，十进制具有的数字符号个数为 10。

十进制数的计算规律是由低位向高位进位“逢十进一”，也就是说，每位累计不能超过 9，计满 10 就应向高位进 1。

当人们看到一个十进制数，如 123.45 时，就会立刻想到：这个数的最左位为百位（1 代表 100），第二位为十位（2 代表 20），第三位为个位（3 代表 3），小数点右边第一位为十分位（4 代表 4/10），第二位为百分位（5 代表 5/100）。这里百、十、个、十分之一和百分之一都是 10 的次幂，它取决于系数所在的位置，称之为“权”。十进制数 123.45 从左至右各位的权分别是 10^2、10^1、10^0、10^{-1}、10^{-2}。这样，123.45 按权展开的形式如下：

$123.45 = 1\times10^2 + 2\times10^1 + 3\times10^0 + 4\times10^{-1} + 5\times10^{-2}$

等式左边的表示方法称之为位置计数法，等式右边则是其按权展开式。

一般说来，对于任意一个十进制数 N，可用位置计数法表示如下：

$(N)_{10} = (a_{n-1}\ a_{n-2}\cdots\ a_1 a_0.\ a_{-1}\cdots\ a_{-m})_{10}$

也可用按权展开式表示如下：

$$(N)_{10} = a_{n-1}\times 10^{n-1} + a_{n-2}\times 10^{n-2} + \cdots + a_1\times 10^1 + a_0\times 10^0 + a_{-1}\times 10^{-1} + a_{-2}\times 10^{-2} + \cdots + a_{-m}\times 10^{-m}$$

$$= \sum_{i=-m}^{n-1} a_i\times 10^i$$

式中 a_i 表示各个数字符号为 0～9 这 10 个数码中的任意一个；n 为整数部分的位数；m 为小数部分的位数。

通常，对于十进制数的表示，可以在数字的右下角标注 10 或 D。

二、二进制数的表示

数字系统使用电平的高低或者脉冲的有无来表示信息，因此数字系统采用二进制来计数。在二进制中，只有 0 和 1 两个数码，计数规则是由低位向高位“逢二进一”，即每位计满 2 就向高位进 1，例如$(1101)_2$就是一个二进制数，不同数位的数码表示的值不同，各位的权值是以 2 为底的连续整数幂，从右向左递增。

对于任意一个二进制数 N，用位置计数法表示为

$(N)_2 = (a_{n-1}\ a_{n-2}\cdots\ a_1 a_0.\ a_{-1}\cdots\ a_{-m})_2$

用按权展开式表示为

$$(N)_2 = a_{n-1}\times 2^{n-1} + a_{n-2}\times 2^{n-2} + \cdots + a_1\times 2^1 + a_0\times 2^0 + a_{-1}\times 2^{-1} + a_{-2}\times 2^{-2} + \cdots + a_{-m}\times 2^{-m}$$

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

式中 a_i 表示各个数字符号为数码 0 或 1；n 为整数部分的位数；m 为小数部分的位数。

通常，对二进制数的表示，可以在数字右下角标注 2 或 B。

三、任意进制数的表示

二进制数运算规则简单，便于电路实现，它是数字系统中广泛采用的一种数制。但用二进制表示一个数时，所用的位数比用十进制数表示的位数多，人们读写很不方便，容易出错。因此，常采用八进制或十六进制。

八进制数的基数是 8，采用的数码是 0、1、2、3、4、5、6、7。计数规则是从低位向高位“逢八进一”，相邻两位高位的权值是低位权值的 8 倍。例如数$(47.6)_8$就表示一个八进制数。由于八进制的数码和十进制前 8 个数码相同，所以为了便于区分，通常在数字的右下角标注 8 或 O。

十六进制的基数是 16，采用的数码是 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F。其中，A、B、C、D、E、F 分别代表十进制数字 10、11、12、13、14、15。十六进制的计数规则是从低位向高位“逢十六进一”，相邻两位高位的权值是低位权值的 16 倍。例如$(54AF.8B)_{16}$就是一个十六进制数。通常，在数字右下角标注 16 或 H。

与二进制数一样，任意一个八进制数和十六进制数均可用位置计数法的形式和按权展开的形式表示。一般来说，对于任意的数 N，都能表示成 r 为基数的 r 进制数，数 N 的表示方

法也有两种形式，即

位置计数法：$(N)_r = (a_{n-1}\ a_{n-2}\cdots\ a_1 a_0.\ a_{-1}\cdots\ a_{-m})_r$

按权展开式：$(N)_2 = a_{n-1}\times r^{n-1} + a_{n-2}\times r^{n-2} + \cdots + a_1\times r^{1} + a_0\times r^{0}$

$$+ a_{-1}\times r^{-1} + a_{-2}\times r^{-2} + \cdots + a_{-m}\times r^{-m}$$

$$= \sum_{i=-m}^{n-1} a_i r^i$$

式中 a_i 表示各个数字符号为 0～r-1 数码中任意一个；r 为进位制的基数；n 为整数部分的位数；m 为小数部分的位数。

r 进制的计数规则是从低位向高位“逢 r 进一”。

1.2.2 数制转换

将数由一种数制转换成另一种数制称为数制转换。由于计算机采用二进制表示，而解决实际问题时计算机的数值输入与输出通常采用十进制。在进行数据处理时首先必须把输入的十进制数转换成计算机所能接受的二进制数，并在计算机运行结束后把二进制数转换为人们所习惯的十进制数输出。

1. 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 8、16 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 1-1】 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）$_{16}=1\times16^2$ 十 12×16^1 十 $4\times16^0=256+192+4=$（452）$_{10}$

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

2. 十进制数转换为任意进制数

十进制数转换为其它进制数时，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（$K_{n-1}\cdots\cdots\ K_1\ K_0$）$_R$。

【例 1-2】 将$(217)_{10}$ 转换成二进制数。

解：

2	｜	2 1 7	余数	
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}，依次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（$0.K_{-1}K_{-2}\cdots\cdots K_{-m}$）$_R$。

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则不会混淆了。

【例 1-3】 将小数 0.6875 转换成二进制小数。

解： 0.6875

	整数部分		
× 2			
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 1-4】 将小数 0.324 转换成二进制小数，保留 3 位小数。

解： 0.324

	整数部分		
× 2			
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又有小数，可将整数和小数分别进行转换，然后合并就可得到结果。

【例 1-5】 将$(24.625)_{10}$转换成二进制。

解：$(24.625)_{10}=24_{10}+0.625_{10}$

$=11000_2+0.101_2$

$=11000.101_2$

1.3 带符号数的代码表示

1.3.1 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 X1=+1101，X2= -1101，则$[X1]_{原}$=01101，$[X2]_{原}$=11101。

已知 X3=+0.1011，X4= -0.1011，则$[X3]_{原}$=0.1011，$[X4]_{原}$=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：

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

表示范围：-127～+127（8 位整数时）

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

纯小数：

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

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，若 X=Y+Z，但（X）$_{原}$≠（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，因此需要寻找其它表示负数的方法。

1.3.2 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：

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

表示范围：-127～+127（8 位整数时）

注意整数“0”的反码有两种形式，即$[+0]_{反}$=00…0 和$[-0]_{反}$=11…1。

纯小数：

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

同理，纯小数“0.0”的反码也有两种形式。

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。

$[X1+X2]_{反}=[X1]_{反}+[X2]_{反}$

$[X1-X2]_{反}=[X1]_{反}+[-X2]_{反}$

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，我们称之为“循环相加”（或“循环进位”）。

【例 1-6】 已知 X1 = +0.1110, X2 =+0.0101, 求 X1-X2。

解：X1-X2 可通过反码相加实现。运算如下：

$$[X1-X2]_{反}=[X1]_{反}+[-X2]_{反}=\begin{array}{r}0.1110+\\ 1.1010\end{array}$$

$$=10.1000\quad（符号位上有进位）$$

$$=0.1001$$

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。

1.3.3 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： X1=+1101，X2= -1101，则$[X1]_{补}$=01101，$[X2]_{补}$=10011。

X3=+0.1011，X4= -0.1011，则$[X3]_{补}$=0.1011，$[X4]_{补}$=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：

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

表示范围：-128～+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：

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

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。

$[X1+X2]_{补}=[X1]_{补}+[X2]_{补}$

$[X1-X2]_{补}=[X1]_{补}+[-X2]_{补}$

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 1-7】 已知 X1 = +0.1110, X2 =+0.0101, 求 X1-X2。

解：X1-X2 可通过补码相加实现。运算如下：

$$[X1-X2]_{补}=[X1]_{补}+[-X2]_{补}=\begin{array}{r}0.1110+\\ 1.1011\end{array}$$

$$=10.1001\quad（丢弃符号位上的进位）$$

$$=0.1001$$

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 1-8】 已知 X = +100 1001，求$[X]_{补}$和$[-X]_{补}$。

解：$[X]_{补}$= 0100 1001 ， $[-X]_{补}$= 1011 0110 + 1 = 1011 0111

1.3.4 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。更快的方法是，对于反码、补码，扩展的数据位的值和原来符号位的值是一样的。

1.4 数的定点与浮点表示

在计算机中，小数点不用专门的器件表示，而是按约定的方式标出。共有两种方法来表示小数点的存在，即定点表示和浮点表示。定点表示的数称为定点数，浮点表示的数称为浮点数。

1. 定点表示

小数点固定在某一位置的数为定点数，包含两种格式：当小数点位于数符和第一数值位之间时，机器内的数为纯小数；当小数点位于数值位之后时，机器内的数为纯整数。采用定点数的机器叫做定点机。数值部分的位数 n 决定了定点机中数的表示范围。若机器数采用原码，小数定点机中数的表示范围是$-(1-2^{-n})\sim(1-2^{-n})$，整数定点机中数的表示范围是$-(2^{n}-1)\sim(2^{n}-1)$。在定点机中，由于小数点的位置固定不变，故当机器处理的数不是纯小数或纯整数时，必须乘上一个比例因子，否则会产生“溢出”。

2. 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要先对阶。

【例 1-9】 已知 $N1=2^{011}\times0.1001$ ，$N2=2^{001}\times0.1100$ ， 求 N1+N2

解：先对阶：$N2=2^{001}\times0.1100=2^{011}\times0.0011$ (小数点左移 2 位，阶码加 2)

$$N1+N2=2^{011}\times0.1001+2^{011}\times0.0011$$

$$=2^{011}(0.1001+0.0011)$$

$$= 2^{011} \times 0.1100$$

两个浮点数作乘除法，其规则为：

若 $N1=2^{j1} \times S1$ ，$N2=2^{j2} \times S2$ 则

$$N1 \times N2 = (2^{j1} \times S1) \times (2^{j2} \times S2)$$
$$= 2^{(j1+j2)} \times (S1 \times S2)$$
$$N1 \div N2 = 2^{(j1-j2)} \times (S1 \div S2)$$

1.5 数码和字符的编码

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前所述，以便在数字系统中进行运算、存储和传输。为了表示字符等一类被处理的信息，也需要用一定位数的二进制数码表示，这个特定的二进制码称为代码。注意，“代码”和“数码”的含义不尽相同，代码是不同信息的代号。

每个信息制定一个具体的码字去代表它，这一指定过程称为编码。由于指定的方法不是惟一的，故对一组信息存在着多种编码方案。

1.5.1 BCD 编码

在数字系统中，各种数据要转换为二进制代码才能进行处理，而人们习惯于使用十进制数，所以在数字系统的输入输出中仍采用十进制数，这样就产生了用四位二进制数表示一位十进制数的方法，这种用于表示十进制数的二进制代码称为二－十进制代码（Binary Coded Decimal），简称为 BCD 码。它具有二进制数的形式以满足数字系统的要求，又具有十进制的特点（只有十种有效状态）。在某些情况下，计算机也可以对这种形式的数直接进行运算。常见的 BCD 码表示有 8421 码、2421 码和余 3 码。

余 3 码是每个 8421 码加 3。三种 BCD 码都是用四位二进制代码表示一位十进制数字，与十进制数之间的转换是以四位二进制对应一位十进制直接进行变换。一个 n 位十进制数对应的 BCD 码一定为 4n 位。

在 BCD 编码中，若四位二进制一组中的每位都有固定权值，则称为有权码 (weighted code)，如 8421 码、2421 码是有权码。余 3 码是无权码。

当两个十进制数为互反，它们对应的二进制码互反，则称为自补码(self-complementing code)。2421 码、余 3 码是自补码。

常见 BCD 编码如表 1.1 所示。

表 1.1　常用 BCD 码

十进制数	8421 码	2421 码	余 3 码
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	0100	0111
5	0101	1011	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

8421 码是一种使用最广的 BCD 码，是一种有权码，其各位的权分别是（从最有效高位开始到最低有效位）8,4,2,1。

【例 1-10】 写出十进制数（1592）$_{10}$ 对应的 8421 码和余 3 码。

解：（1592）$_{10}$=（0001 0101 1001 0010）$_{8421}$

=（0100 1000 1100 0101）$_{余3码}$

【例 1-11】 写出 8421 码（1101001.01011）$_{8421}$ 对应的十进制数。

解：（1101001.01011）$_{8421}$＝（0110 1001 . 0101 1000）$_{8421}$=（69.58）$_{10}$

在使用 8421BCD 码时一定要注意其有效的编码仅十个，即：0000～1001。四位二进制数的其余六个编码 1010,1011,1100,1101,1110,1111 不是有效编码。在余 3 码中 0000～0010，1101～1111 这六个编码不是有效编码。在 2421 码中，0101～1010 这 6 个编码仍表示有效的十进制数，只不过由于它们和已有的十进制数重复，所以不使用。

1.5.2 可靠性编码

一、格雷码

格雷码（Gray）的编码规则是任何相邻的两个码字中，仅有一位不同，其它位则相同，如表 1.2 所示。格雷码可以用在计数器中，当从某一编码变到下一个相邻编码时，只有一位的状态发生变化，这有利于提高系统的工作速度和可靠性。很显然，格雷码中的每一位都没有固定的权值，是无权码。

表 1.2 四位二进制数与格雷码

十进制数	二进制数	格雷码		十进制数	二进制数	格雷码
0	0000	0000		8	1000	1100
1	0001	0001		9	1001	1101
2	0010	0011		10	1010	1111
3	0011	0010		11	1011	1110
4	0100	0110		12	1100	1010
5	0101	0111		13	1101	1011
6	0110	0101		14	1110	1001
7	0111	0100		15	1111	1000

将二进制转换到格雷码的方法为：保持最高位不变，其它位与前面一位异或。假设二进制数为 $B_{n-1}B_{n-2}\ldots B_0$，格雷码为 $G_{n-1}G_{n-2}\ldots G_0$，其公式是

$G_{n-1}=B_{n-1}$

$G_i=B_{i+1}\oplus B_i$ ， i = n-2….0

⊕称为异或运算，两个数不同则结果为 1。具体运算规则为 0⊕0=0，0⊕1=1，1⊕0=1，1⊕1=0。

如：二进制数为 1 0 1 1 0 1 0 0

⊕ ⊕ ⊕ ⊕ ⊕ ⊕ ⊕

Gray 码 1 1 1 0 1 1 1 0

二、奇偶校验码

奇偶校验码是为检查数据传输是否出错设置的，在每组数据信息上附加一位奇偶校验位，若采用奇校验方式，则使包括校验码在内的数据含有奇数个“1”；而偶校验方式则使包括校验码在内的数据含有偶数个“1”。

例如字母“B”的 7 位 ASCII 码为 1000010，在最高位增加一位奇偶校验位。若采用其奇校验，则为 1 1 0 0 0 0 1 0；若采用偶校验，则为 0 1 0 0 0 0 1 0。

奇偶校验码可发现奇数个错误，但不能发现偶数个错误。当发现奇数个错误时，由于不知道是哪些位出错，所以奇偶校验码没有纠错能力。

奇偶校验码如表 1.3 所示。

表 1.3 奇偶校验码

十进制	奇校验		偶校验	
	信息位	校验位	信息位	校验位
0	0000	1	0000	0
1	0001	0	0001	1
2	0010	0	0010	1
3	0011	1	0011	0
4	0100	0	0100	1
5	0101	1	0101	0
6	0110	1	0110	0
7	0111	0	0111	1
8	1000	0	1000	1
9	1001	1	1001	0

1.5.3 字符编码

最常用的字符代码是 ASCII 码，每个字符是用 7 位二进制码表示，见表 1.4。它是由 128 个字符组成的字符集，其中 32 个控制字符，然后是空格，数字，大写字母，小写字母。数字 0~9 的高 3 位是 011,低 4 位是 0000~1001 所以和二进制间进行转换很容易。大小写字母之间进行转换也很容易，因为只是 a5 位的不同，例字符 B 的编码为 1000010，而字符 b 的编码为 1100010。

表 1.4 七位 ASCII 码表

低 4 位 $a_3a_2a_1a_0$	高 3 位 $a_6a_5a_4$							
	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	`	p

0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	″	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	′	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	−	=	M	]	m	}
1110	SO	RS	.	>	N	↑	n	~
1111	SI	US	/	?	O	←	o	DEL

1.6 本章小结

本章首先介绍了数字电路的一些基础知识，然后学习了自然界中以十进制形式表示的数在计算机中如何表示，给出了二进制、八进制和十六进制数的表示方法，以及各种进制数之间的相互转换方法；其次，学习了带符号数的代表表示，给出了原码、反码和补码的表示方法和运算规则；最后学习了数的定点与浮点表示和数码与字符的编码。

具体关键知识点梳理如下：

1、 计算机用电平的高低和脉冲的有无来表示信息，因此计算机采用二进制计数法，为了方便记忆和书写，将二进制数分三位一组表示引出了八进制，将二进制数分四位一组表示引出了十六进制数。

2、 在计算机中，带符号数的表示方法有三种：原码、反码和补码，需要掌握它们各自的表示方法和运算规则。

3、 在计算机中，小数的表示有两种方法：定点表示和浮点表示。

4、 人们习惯于用 4 位二进制数来表示 1 位十进制数，这就是 BCD 码，常见的 BCD 码有 8421BCD 码、余 3 码、5421BCD 码和 2421BCD 码。

5、 格雷码和奇偶校验码都属于可靠性编码，格雷码可以避免多位数据在同时变化时产生的错误，奇偶校验码是一种最基本的检错码，可以发现数据单个或奇数个错误。

1.7 习题

1．把下列各数写成按权展开的形式。

365.9_{10}，EB_{16}，011101101_2

2．将下列十进制数转换成二进制。

34，67，126，215

3．将下列十进制数转换成二进制（准确到小数点后四位）。

0.5，0.8125，27.75，61.452

4．下列二进制数转换成十进制、八进制和十六进制数。

$(1100010110)_2$，$(0.11001)_2$

5．把十六进制$(2A3B.D)_{16}$转换为二进制、八进制数。

6．把十进制数$(62.25)_{10}$转换成二进制、八进制和十六进制。

7．用二进制补码运算求（-54-30）$_{10}$

8．用十六进制运算求（08D+03F）$_{16}$

9．用 8421 码和余 3 码表示下列各数。

$(53.69)_{10}$ ，$(214.78)_{10}$

10．请写出下列 BCD 码对应的十进制数。

（10000110）$_{8421}$，（10100100）$_{余3码}$

11．将二进制码 11010001 转换为格雷码。

12．请设计将格雷码转换为二进制码的转换规则，并举例加以验证。

13．已知$[X]_{反}$=11001,求$[-X]_{补}$，$[X/2]_{补}$和$[2X]_{原}$。

14．分别用原码、反码、补码表示下列各数。

（+1011），（-1011），（+0.0011），（-0.0011）

15．请将十进制数-0.07525 表示为规格化浮点数，阶码（包括阶符）为 4 位二进制位，尾数（包括尾符）为 8 个二进制位，均采用补码形式。

16．如何判断二进制数、8421 码、余 3 码所表示的整数是奇数还是偶数？

17．十进制数 0.7563，若要求舍入误差小于 1%，则用于表示此值的二进制数小数点后需要多少个数据位？

18．计算下列两个编码的奇校验位。

0110010，1010011

19．下列编码为两个采用偶校验传输的结果，最高位为校验位，请判断每个传输中是否有错误。

（1） 10111001

（2） 01101010

20．写出字符串 356 对应的 ASCII 编码。