

21世纪高等院校信息与通信工程规划教材
21st Century University Planned Textbooks of Information and Communication Engineering

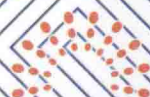
数字电路 与逻辑设计

杨永健 主编
玄玉波 张伟 副主编

Digital Circuit
and Logic Design



人民邮电出版社
POSTS & TELECOM PRESS



高校系列

高等院校信息与通信工程规划教材
University Planned Textbooks of Information and Communication Engineering

数字电路 与逻辑设计

杨永健 主编
玄玉波 张伟 副主编

Digital Circuit
and Logic Design

人民邮电出版社
北京



图书在版编目 (C I P) 数据

数字电路与逻辑设计 / 杨永健主编. — 北京 : 人民邮电出版社, 2015. 2
21世纪高等院校信息与通信工程规划教材
ISBN 978-7-115-38153-8

I. ①数… II. ①杨… III. ①数字电路—逻辑设计—高等学校—教材 IV. ①TN79

中国版本图书馆CIP数据核字 (2015) 第014727号

内 容 提 要

本书主要介绍数字电路的基本分析方法和设计方法, 以及用可编程逻辑器件设计电路的软件平台和硬件描述语言设计方法。主要内容包括数制和码制、逻辑代数基础、逻辑门电路、组合逻辑电路分析与设计、中规模组合逻辑器件应用、触发器、时序逻辑电路的分析与设计、常用时序集成器件、555 定时器及其应用、半导体存储器件和可编程器件、硬件描述语言、数模和模数转换。

本书编写语言简练, 由浅入深, 章节顺序安排合理, 有较多的例题和习题可供参考和练习, 利于教学和自学。本书适合作为高等本科院校通信、电子、电气、信息技术和计算机等专业“数字电路与逻辑设计”课程教材, 也适合作为高职高专相关专业的教材以及工程技术人员的技术参考书。

-
- ◆ 主 编 杨永健
 - 副 主 编 玄玉波 张 伟
 - 责任编辑 张孟玮
 - 执行编辑 税梦玲
 - 责任印制 沈 蓉 彭志环
 - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
 - 邮编 100164 电子邮件 315@ptpress.com.cn
 - 网址 <http://www.ptpress.com.cn>
 - 北京圣夫亚美印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
 - 印张: 19.75 2015 年 2 月第 1 版
 - 字数: 494 千字 2015 年 2 月北京第 1 次印刷
-

定价: 46.00 元

读者服务热线: (010)81055256 印装质量热线: (010)81055316
反盗版热线: (010)81055315

“数字电路与逻辑设计”是通信、电子、信息技术、自动化、计算机、电力系统以及机电一体化等专业的一门重要的专业基础课程。当今，电子技术飞速发展，许多新的数字电子器件和电路分析方法不断出现，尤其是中规模电子器件的发展更为迅速，因此，数字电路与逻辑设计课程的内容也要不断更新，以适应飞速发展的数字电路与逻辑设计的需要。

全书共分为 10 章。第 1 章、第 2 章介绍数字电路与逻辑设计的基础知识，主要讲解数字电子技术中数制、码制及逻辑代数基础。第 3 章介绍分立元件、TTL 和 CMOS 集成门电路的构成。第 4 章重点介绍组合逻辑电路的分析与设计方法，并详细讲述常用中规模集成组合逻辑器件及其应用。第 5 章介绍时序电路的常用器件——触发器。第 6 章重点讨论时序逻辑电路分析、设计的常用方法以及常用中规模时序逻辑器件与应用。第 7 章介绍半导体存储器和可编程逻辑器件。第 8 章对硬件描述语言做了简要介绍。第 9 章讲述数模转换和模数转换。第 10 章从 555 定时器入手，详细讨论脉冲产生、整形及变换电路的构成方法及应用。每一章都有例题及习题，以帮助读者消化、巩固所学的理论及方法。

本书是作者近年来的教学实践经验的总结，并迎合近年来压缩“数字电路与逻辑设计”课时的课程改革，将重点的内容放在每章的前面，删去进化过程中的器件讲解（如第 5 章中删去了主从 JK 触发器）。本书强调理论联系实际，书中以不同的方式，安排了一定数量的电路实例并和习题紧密配合，以期提高教学效果。本书仍沿用从模拟到数字的体系，如需从数字到模拟的体系讲授，只需将书中第 3 章“门电路”放到前面讲授即可。

本书由杨永健任主编，负责编写提纲的制订、各章初稿的修订和统稿。第 1 章、第 2 章、第 3 章、第 4 章及第 9 章由玄玉波编写；第 5 章、第 6 章、第 10 章由张伟编写；第 7 章、第 8 章由杨永健编写。在编写过程中王勇和吴宁参加了许多工作，给与了很大的支持，在此表示衷心的感谢。

由于时间仓促，作者水平有限，书中难免有欠妥之处，恳请读者批评指正。

作 者

2014 年 10 月

第 1 章 数制与码制 1	2.3 逻辑代数中的基本规则 22
1.1 概述..... 1	2.3.1 代入规则..... 22
1.2 几种常用的数制..... 2	2.3.2 反演规则..... 23
1.2.1 十进制..... 2	2.3.3 对偶规则..... 23
1.2.2 二进制..... 3	2.4 逻辑函数的表示方法 24
1.2.3 八进制..... 3	2.5 逻辑函数的公式化简法 27
1.2.4 十六进制..... 3	2.6 逻辑函数的卡诺图化简法 29
1.3 不同进制间的转换..... 4	2.6.1 最小项与最小项表达式..... 29
1.3.1 二进制数转换十进制数..... 4	2.6.2 用卡诺图表示逻辑函数..... 31
1.3.2 十进制数转换二进制数..... 4	2.6.3 卡诺图化简逻辑函数..... 34
1.3.3 八进制、十六进制与 二进制相互转换..... 6	2.6.4 具有无关项的逻辑函数式 及其化简方法..... 35
1.4 二进制算数运算..... 6	2.7 逻辑函数的表达式形式及 其转化 37
1.4.1 二进制算术运算的特点..... 6	本章小结 39
1.4.2 原码、反码、补码和 补码运算..... 7	习题 39
1.5 几种常用的编码..... 9	第 3 章 门电路 44
1.5.1 二十进制编码..... 9	3.1 概述..... 44
1.5.2 可靠性编码..... 10	3.2 半导体管的开关特性..... 45
1.5.3 字符代码..... 12	3.2.1 晶体二极管开关特性..... 45
本章小结 14	3.2.2 晶体三极管开关特性..... 46
习题 14	3.2.3 MOS 管开关特性..... 47
第 2 章 逻辑代数基础 16	3.3 分立元件逻辑门电路..... 48
2.1 逻辑代数中的三种基本运算..... 16	3.3.1 与门电路..... 49
2.1.1 逻辑代数中的变量和常量..... 16	3.3.2 或门电路..... 49
2.1.2 基本逻辑运算..... 16	3.3.3 非门电路..... 50
2.1.3 几种常用的逻辑运算..... 18	3.3.4 与非门电路..... 50
2.2 逻辑代数基本定律和常用公式..... 21	3.3.5 或非门电路..... 51
2.2.1 基本定律..... 21	3.4 TTL 门电路 52
2.2.2 常用公式..... 22	3.4.1 TTL 反相器电路结构及

工作原理.....	52	第 5 章 触发器.....	122
3.4.2 TTL 的反相器电气特性.....	53	5.1 基本触发器.....	122
3.4.3 其他类型的 TTL 门电路.....	58	5.1.1 与非门组成的基本 SR	
3.4.4 TTL 电路的改进系列.....	64	触发器.....	122
3.4.5 ECL 和 I ² L.....	65	5.1.2 或非门组成的基本 SR	
3.5 CMOS 门电路.....	66	触发器.....	125
3.5.1 CMOS 反相器电路结构及		5.2 电平触发器.....	125
工作原理.....	66	5.2.1 电平触发 SR 触发器.....	126
3.5.2 CMOS 的反相器电气特性.....	67	5.2.2 电平触发 D 触发器.....	127
3.5.3 其他类型的 CMOS 门电路.....	70	5.2.3 电平触发器的空翻现象.....	128
3.5.4 BiCMOS 电路.....	75	5.3 边沿触发器.....	128
3.5.5 CMOS 逻辑门电路		5.3.1 边沿 D 触发器.....	128
技术参数.....	75	5.3.2 边沿 JK 触发器.....	130
3.6 数字集成电路的正确使用.....	76	5.3.3 维持-阻塞型 D 触发器.....	132
3.6.1 TTL 电路的正确使用.....	76	5.4 触发器的逻辑功能和功能转换.....	133
3.6.2 CMOS 电路的正确使用.....	76	5.5 触发器的动态特性.....	134
3.7 TTL 电路与 CMOS 电路的接口.....	77	5.5.1 基本 SR 触发器.....	135
3.8 门电路带负载时的接口电路.....	78	5.5.2 同步电平触发 SR 触发器的	
3.8.1 用门电路直接驱动显示器件.....	78	动态特性.....	136
3.8.2 机电性负载接口.....	79	5.5.3 维持阻塞触发器的动态	
本章小结.....	79	特性.....	137
习题.....	80	本章小结.....	138
第 4 章 组合逻辑电路.....	86	习题.....	139
4.1 概述.....	86	第 6 章 时序逻辑电路.....	143
4.2 组合逻辑电路的分析.....	86	6.1 概述.....	143
4.3 常用的组合逻辑器件.....	88	6.1.1 时序逻辑电路的结构.....	143
4.3.1 编码器.....	88	6.1.2 描述时序电路逻辑	
4.3.2 译码器.....	91	功能的函数.....	143
4.3.3 数据选择器.....	97	6.1.3 时序电路的分类.....	144
4.3.4 加法器.....	99	6.2 时序电路的分析方法.....	144
4.3.5 数值比较器.....	102	6.2.1 同步时序逻辑电路的	
4.4 组合逻辑电路的设计方法.....	105	分析方法.....	144
4.4.1 组合逻辑电路的设计方法.....	105	6.2.2 异步时序逻辑电路的	
4.4.2 用 SSI 设计组合逻辑电路.....	106	分析举例.....	149
4.4.3 用中规模集成组件设计		6.3 常用时序逻辑电路及应用.....	150
组合逻辑电路.....	110	6.3.1 寄存器与移位寄存器.....	150
4.5 组合逻辑电路中的竞争冒险.....	113	6.3.2 计数器.....	155
本章小结.....	115	6.3.3 常用时序逻辑电路的应用.....	168
习题.....	116	6.4 时序逻辑电路的设计方法.....	178

6.4.1 同步时序逻辑电路的设计方法	178	8.2 Verilog HDL 硬件描述语言程序基本结构	233
6.4.2 时序逻辑电路的启动设计	186	8.2.1 Verilog 语言程序的模块	234
6.5 异步时序逻辑电路的设计方法	188	8.2.2 逻辑功能的几种基本描述方法	235
本章小结	190	8.3 Verilog HDL 语言要素	237
习题	191	8.3.1 标识符	237
第 7 章 大规模集成电路	198	8.3.2 关键字	237
7.1 概述	198	8.3.3 格式	238
7.2 只读存储器 ROM	199	8.3.4 注释	238
7.2.1 ROM 的结构和工作原理	199	8.3.5 数字与字符串	238
7.2.2 ROM 的分类	200	8.3.6 数据类型	239
7.2.3 ROM 的应用	202	8.3.7 参数	240
7.3 随机存储器 RAM	203	8.3.8 运算符及表达式	240
7.3.1 RAM 的结构和原理	203	8.4 Verilog HDL 语句	243
7.3.2 RAM 的存储单元	203	8.4.1 赋值语句	243
7.3.3 存储容量的扩展	204	8.4.2 条件语句	244
7.4 可编程逻辑器件概述	205	8.4.3 循环语句	244
7.5 简单的可编程逻辑器件	206	8.4.4 过程语句	245
7.5.1 简单可编程逻辑器件的阵列结构特点	206	8.5 系统任务和系统函数	246
7.5.2 与阵列和或阵列编程方法	209	8.5.1 任务	246
7.5.3 通用阵列逻辑器件 GAL 结构	212	8.5.2 函数	246
7.6 复杂的可编程逻辑器件 CPLD	214	8.6 用 Verilog HDL 描述逻辑电路的实例	246
7.7 现场可编程门阵列 FPGA	218	本章小结	251
7.7.1 FPGA 性能及基本结构	218	习题	251
7.7.2 嵌入式阵列和逻辑阵列块	220	第 9 章 数模与模数转换器	252
7.7.3 逻辑单元、快速通道互连及 I/O 单元	222	9.1 概述	252
7.8 CPLD 和 FPGA 的编程与配置	225	9.2 D/A 转换器	252
7.9 数字小系统的设计及实现	227	9.2.1 权电阻网络 D/A 转换器	253
7.9.1 数字系统的 6 个设计层次	228	9.2.2 倒 T 型电阻网络 D/A 转换器	254
7.9.2 应用 FPGA/CPLD 的 EDA 开发流程	228	9.2.3 权电流型 D/A 转换器	256
本章小结	230	9.2.4 D/A 转换器的转换精度与转换速度	259
习题	231	9.3 A/D 转换器	260
第 8 章 硬件描述语言简介	233	9.3.1 A/D 转换的基本概念	260
8.1 概述	233	9.3.2 并行比较型 A/D 转换器	263
		9.3.3 逐次逼近型 A/D 转换器	265

9.3.4 双积分型 A/D 转换器	268	施密特触发器	279
9.3.5 A/D 转换器的转换精度 与转换时间	272	10.3.2 用 555 定时器构成 单稳态触发器	282
本章小结	272	10.3.3 用 555 定时器构成 自激多谐振荡器	285
习题	273	10.4 常用脉冲发生和整形电路	288
第 10 章 脉冲波形的产生与变换	277	10.4.1 施密特触发器	288
10.1 概述	277	10.4.2 单稳态触发器	292
10.2 555 定时器的电路结构及 工作原理	277	10.4.3 多谐振荡器	296
10.2.1 电路结构	278	本章小结	300
10.2.2 555 定时器工作原理	279	习题	301
10.3 用 555 定时器构成脉冲电路	279	参考文献	308
10.3.1 用 555 定时器构成			

现代信息的采集、存储、处理和传输越来越趋于数字化。常用的电子设备,如计算机、电话、照相机等,都是采用数字电路。本章首先介绍模拟信号和数字信号的定义,然后讨论数制、二进制数的算术运算和几种常见的编码。

1.1 概述

电子信号可用于表示任何信息,如符号、文字、语音、图像等,从表现形式上可归为两类:模拟信号和数字信号。模拟信号的特点是时间和幅度上都连续变化(连续的含义是在某一取值范围内可以取无限多个数值)。交流放大电路的电信号就是模拟信号,如图 1-1 所示。我们把处理模拟信号的电子电路称为模拟电路。

数字信号是时间和幅度上都不连续变化的离散的脉冲信号,例如图 1-2 所示。用数字信号对数字量进行算术运算和逻辑运算的电路称为数字电路,或数字系统。由于它具有逻辑运算和逻辑处理功能,所以又称为数字逻辑电路。

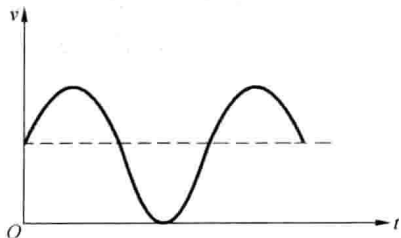


图 1-1

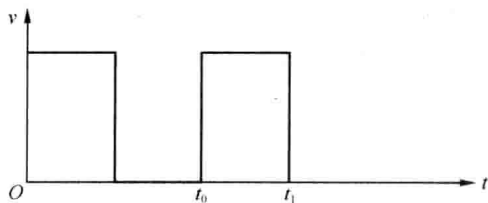


图 1-2

数字信号的高、低电平可以用“1”和“0”来表示。如果高电平用“1”表示,低电平用“0”表示,则称这种表示方法为正逻辑。如果高电平用“0”表示,低电平用“1”表示,则称这种表示方法为负逻辑。本书均采用正逻辑。

对于图 1-2 所示的脉冲波形,它从高电平变化到低电平,再从低电平变化到高电平。在 t_0 时刻出现的波形称为上升沿,在 t_1 时刻出现的波形称为下降沿。对于非理想脉冲波形及脉冲波形的产生和变换将在第 10 章详细介绍。

数字电路通常是根据脉冲信号的有无进行工作的,而与脉冲幅度无关,所以抗干扰能力强、准确度高。虽然数字信号的处理电路比较复杂,但因信号本身的波形十分简单,只有两

种状态——有或无，在电路中具体表现为高电平和低电平，所以用于数字电路的半导体管不是工作在放大状态而是工作在开关状态，或饱和导通，或截止。因此制作时工艺要求相对低，易于集成化。随着数字集成电路制作技术的发展，数字电路在通信、计算机、自动控制、航天等各个领域获得了广泛的应用。

数字信号通常都是用数码表示的。数码不仅可以用来表示数量的大小，还可以用来表示事物或事物的不同状态。用数码表示数量大小时，需要用多位数码表示。通常把多位数码中每一位的构成方法及从低位到高位进位的规则称为数制。而在用于表示不同事物时，这些数码已经不再具有表示数量大小的含义，它们只是不同事物的代号，比如，我们每个人的身份证号码。这些号码仅仅表示不同对象，没有数量大小的含义。为了便于记忆和查找，在编制代码时总要遵循一定的规则，这些规则就称为码制。考虑到信息交换的需要，通常会制定一些大家共同使用的通用代码，例如：目前国际上通用的美国信息交换标准代码（ASCII 码，见本章第 1.5 节）就属于这一种。

1.2 几种常用的数制

任何一个数都可以用不同的进位体制来表示，但不同进位计数体制的运算方法和难易程度各不相同，这对数字系统的性能有很大影响。常用的进位计数制有十进制、二进制、八进制、十六进制。

1.2.1 十进制

十进制数的特点：

- (1) 每一位有 10 个数码，即 0~9；
- (2) 由低位向高位的进位原则是“逢十进一”。

这里先说明两个概念，用来表示某种进位体制的数码的个数叫做基数。不同位置上数字代表的数值大小叫做位权，简称权。因此，十进制的基数为 10，权为 10^i 。例如 125.37，从左至右，第一个为百位，该位置上的 1 代表 100，权为 10^2 ，把 125.37 按权的形式展开为

$$125.37 = 1 \times 10^2 + 2 \times 10^1 + 5 \times 10^0 + 3 \times 10^{-1} + 7 \times 10^{-2}$$

所以任何一个十进制数 D 均可展开为

$$D = \sum_{i=-m}^{n-1} k_i 10^i \quad (1-1)$$

式中 k_i 是第 i 位的系数，它可以是 0~9 这 10 个数码中的任何一个。若整数部分的位数是 n ，小数部分的位数是 m ，则 i 是包含从 $n-1 \sim 0$ 的所有正整数和从 $-1 \sim -m$ 的所有负整数。

若以 N 取代式中的 10，即可得到任意进制（ N 进制）数按十进制展开式的普遍形式

$$D = \sum_{i=-m}^{n-1} k_i N^i \quad (1-2)$$

式中 i 的取值与式 (1-1) 的规定相同。 N 是计数的基数， k_i 是第 i 位的系数， N^i 称为第 i 位的权。

1.2.2 二进制

二进制数的特点:

- (1) 每一位有 2 个数码, 即 0 和 1;
- (2) 由低位向高位的进位原则是“逢二进一”。

所以二进制数各位的权是基数 2 的幂。任意一个二进制数按权展开式为

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

由式 1-3 可计算出它所表示的十进制数的大小。例如

$$(110.01)_2 = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = (6.75)_{10}$$

上式中分别使用下脚标 2 和 10 表示括号里的数是二进制数和十进制数。有时也用 B (Binary) 和 D (Decimal) 代替 2 和 10 这两个脚注。采用二进制计数制, 对计算机等数字系统来说, 运算、存储和传输极为方便可靠, 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计数制来进行书写或打印。

1.2.3 八进制

八进制数的特点:

- (1) 每一位有 8 个数码, 即 0~7;
- (2) 由低位向高位的进位原则是“逢八进一”。

所以八进制数各位的权是基数 8 的幂。任意一个八进制数按权展开式为

$$D = \sum_{i=-m}^{n-1} k_i 8^i \quad (1-4)$$

由式 1-4 可计算出它所表示的十进制数的大小。例如

$$(23.8)_8 = 2 \times 8^1 + 3 \times 8^0 + 8 \times 8^{-1} = (20)_{10}$$

上式中使用下脚标 8 表示括号里的数是八进制数。有时也用 O (Octal) 表示八进制数。

1.2.4 十六进制

十六进制的特点:

- (1) 每一位有 16 个数码, 即它由 0~9, A~F 组成, 与 10 进制的对应关系是 0~9 对应 0~9, A~F 对应 10~15;

- (2) 由低位向高位的进位原则是“逢十六进一”。

所以十六进制数各位的权是基数 16 的幂。任意一个十六进制数按权展开式为

$$D = \sum_{i=-m}^{n-1} k_i 16^i \quad (1-5)$$

由式 1-5 可计算出它所表示的十进制数的大小。例如

$$(1B.8)_{16} = 1 \times 16^1 + 11 \times 16^0 + 8 \times 16^{-1} = (27.5)_{10}$$

上式中使用下脚标 16 表示括号里的数是十六进制数。

几种常用数制之间的变换关系如表 1-1 所示。

表 1-1 几种常用数制之间的变换关系

十 进 制 数	二 进 制 数	八 进 制 数	十六进制数
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

1.3 不同进制间的转换

1.3.1 二进制数转换十进制数

把二进制数转换为等值的十进制数，通常采用“加权法”，即把二进制数首先写成加权系数展开式，然后按十进制加法规则求和。例如

$$(101.01) = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = (5.25)_{10}$$

1.3.2 十进制数转换二进制数

十进制数转换成等值的二进制数，需要将十进制数的整数部分和小数部分分别进行转换，因为二者的转换方法是不相同的。

1. 整数部分的转换

假定十进制整数为 $(D)_{10}$ 等值的二进制数为 $k_n k_{n-1} \cdots k_0$ ，则有

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

上式表明，若将 $(D)_{10}$ 除以 2 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 ，因此不难看出，若将 $(D)_{10}$ 除以 2 得到的商再除以 2，则所得余数即 k_1 。

具体做法是：用 2 去除十进制整数，可以得到一个商和余数；再用 2 去除商，又会得到一个商和余数，如此进行，直到商为 0 时为止。然后把所有余数按逆序排列，也就是把先得到的余数作为二进制数的低位有效位，后得到的余数作为二进制数的高位有效位，依次排列起来，这就是所谓“除 2 取余，逆序排列”。

例如，将十进制数 98 转化为二进制数，可如下进行

		余数	
2	98	0	低位
2	49	1	
2	24	0	
2	12	0	
2	6	0	
2	3	1	
2	1	1	高位
	0		

故 $(98)_{10} = (1100010)_2$

2. 小数部分的转换

假定十进制小数为 $(D)_{10}$ ，等值的二进制数为 $0.k_n k_{n-1} \dots k_0$ ，则有

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

将上式两边同乘以 2 得到

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

上式说明，将小数 $(D)_{10}$ 乘以 2 所得乘积的整数部分即 k_{-1} 。同理，将乘积的小数部分再乘以 2 又可得到

$$2(k_{-2} 2^{-1} + k_{-3} 2^{-2} \dots + k_{-m} 2^{-m+1}) = k_{-2} + (k_{-3} 2^{-1} + k_{-4} 2^{-2} \dots + k_{-m} 2^{-m+2})$$

以此类推，将每次乘 2 后得到乘积的小数部分再乘以 2，便可求出二进制小数的每一位了。

具体做法是：用 2 乘十进制小数，可以得到积，将积的整数部分取出，再用 2 乘余下的小数部分，又得到一个积，再将积的整数部分取出，如此进行，直到积中的小数部分为 0，或者达到所要求的精度为止。然后把取出的整数部分按顺序排列起来，先取的整数作为二进制小数的高位有效位，后取的整数作为低位有效位。总结来说，十进制小数转换成二进制小数采用“乘 2 取整，顺序排列”法。

例如，将 $(0.625)_{10}$ 转化为二进制小数时，可如下进行

$0.625 \times 2 = 1.25$	整数部分=1	高位
$0.25 \times 2 = 0.5$	整数部分=0	
$0.5 \times 2 = 1.0$	整数部分=1	
故	$(0.625)_{10} = (0.101)_2$	低位

1.3.3 八进制、十六进制与二进制相互转换

由于八进制数的基数 $8=2^3$ ，而十六进制的基数 $16=2^4$ ，所以 3 位二进制数恰好能表示 1 位八进制数，4 位二进制数恰好能表示 1 位十六进制数，因此，二进制转换成等值的八进制（或十六进制）的规则是：从二进制的小数点处开始，向左右两边按每 3 位（或 4 位）二进制数化为一组，不是 3 位（或 4 位）的，整数部分可在最高位的左边添 0，小数部分可在最低位的右边添 0，每组用 1 位等值的八进制（或十六进制）数代替，即可得到相应的八进制（或十六进制）数。举例说明如下。

将二进制数 110 100 011 . 101 000 011 转换成等值的八进制数和十六进制数。

$$(10\ 100\ 011\ .\ 101\ 000\ 011)_2 = (010\ /000\ /011\ .\ 101\ /000\ /100)_2 = (243.504)_8$$

$$(10\ 100\ 011\ .\ 101\ 000\ 011)_2 = (1010\ /0011\ .\ 1010\ /0010)_2 = (A3.A2)_6$$

八进制（或十六进制）数转换成等值的二进制数时，只要按照上述规则进行逆变换即可。例如， $(1C9.2F)_{16} = (0001\ 1100\ 1001.0010\ 1111)_2$

在将十六进制数转换为十进制数时，可将各位按权展开后相加求和得到。在将十进制数转换为十六进制数时，可以先转换为二进制数，然后再将得到的二进制数转换为等值的十六进制数。

1.4 二进制算数运算

1.4.1 二进制算术运算的特点

二进制数的加、减、乘、除四则运算，在数字系统中是经常遇到的，它们的运算规则与十进制数很相似。加法运算是最基本的一种运算。在计算机中，引入补码表示后，加上一些控制逻辑，利用加法就可以实现二进制的减法、乘法和除法运算。

1. 二进制的加法运算

二进制加法运算法则： $0+0=0$ ； $0+1=1+0=1$ ； $1+1=10$ （逢 2 进 1）。

【例 1-1】 求 $(1011011)_2 + (1010.11)_2$

$$\begin{array}{r} 1011011 \\ +\quad 1010.11 \\ \hline 1100101.11 \end{array}$$

则 $(1011011)_2 + (1010.11)_2 = (1100101.11)_2$

2. 二进制数的减法运算

二进制减法运算法则为： $0-0=1-1=0$ ； $0-1=1$ （借 1 当 2）； $1-0=1$ 。

【例 1-2】 求 $(1010110)_2 - (1101.11)_2$

$$\begin{array}{r} 1010110 \\ -\quad 1101.11 \\ \hline 1001000.01 \end{array}$$

则 $(1010110)_2 - (1101.11)_2 = (1001000.01)_2$

3. 二进制数的乘法运算

二进制乘法运算法则为： $0 \times 0 = 0$ ； $0 \times 1 = 1 \times 0 = 0$ ； $1 \times 1 = 1$ 。

【例 1-3】 求 $(1011.01)_2 \times (101)_2$

$$\begin{array}{r}
 1011.01 \\
 \times \quad 101 \\
 \hline
 1011\ 01 \\
 00000\ 0 \\
 +\ 101101 \\
 \hline
 111000\ 01
 \end{array}$$

则 $(1011.01)_2 \times (101)_2 = (111000.01)_2$

可见，二进制乘法运算可归结为“移位与加法”。

4. 二进制数的除法运算

二进制除法运算法则为： $0 \div 0 = 0$ ； $0 \div 1 = 0$ ； $1 \div 1 = 1$ 。

【例 1-4】 求 $(100100.01)_2 \div (101)_2 = ?$

$$\begin{array}{r}
 111.01 \\
 101 \overline{) 100100.01} \\
 \underline{101} \\
 1000 \\
 \underline{101} \\
 110 \\
 \underline{101} \\
 101 \\
 \underline{101} \\
 0
 \end{array}$$

则 $(100100.01)_2 \div (101)_2 = (111.01)_2$

可见，二进制除法运算可归结为“减法与移位”。

1.4.2 原码、反码、补码和补码运算

1. 原码

正数的符号位为 0，负数的符号位为 1，其他位按照一般的方法数的绝对值来表示就构成了原码。

【例 1-5】 求当机器字长为 8 位二进制数的原码。

$$X = +1011011 \quad Y = -1011011$$

解：

$$[X]_{\text{原码}} = 01011011 \quad [Y]_{\text{原码}} = 11011011$$

2. 反码

对于有效数字（不包括符号位）为 n 位的二进制数 N 的反码 $(N)_{\text{INV}}$ 是这样定义的

$$(N)_{\text{INV}} = \begin{cases} N & (\text{当 } N \text{ 为正数}) \\ 2^n - 1 - |N| & (\text{当 } N \text{ 为负数}) \end{cases} \quad (1-6)$$

具体来说，对于一个带符号的数来说，正数的反码与其原码相同，负数的反码为其原码除符号位以外的各位按位取反。

【例 1-6】 求机器字长为 8 位二进制数的反码。

$$X = +1011011 \quad Y = -1011011$$

解：

$$[X]_{\text{原码}} = 01011011 \quad [X]_{\text{反码}} = 01011011$$

$$[Y]_{\text{原码}} = 11011011 \quad [Y]_{\text{反码}} = 10100100$$

负数的反码与负数的原码有很大的区别，反码通常用作求补码过程中的中间形式。

3. 补码

引入补码以后，计算机中的加减运算都可以统一化为补码的加法运算，其符号位也参与运算。为了说明补码运算的原理，我们先来讨论一个生活中常见的事例。例如，你在 6 点钟的时候发现自己的手表停在 9 点上了，因而必须把表针拨回到 6 点。由图 1-3 可以看出，这时有两种拨法：第一种拨法是往回拨 3 格， $9-3=6$ ，拨回到了 6 点；另一种拨法是往前拨 9 格， $9+9=18$ 。由于表盘的最大数只有 12，超过 12 以后的“进位”将自动消失，于是就只剩下减去 12 以后的余数了，即 $18-12=6$ ，也将表针拨回到了 6 点。这个例子说明， $9-3$ 的减法运算可以用 $9+9$ 的加法运算代替。因为 3 和 9 相加正好等于产生进位的模数 12，所以我们称 9 为 -3 对模 12 的补数，也称为补码 (Complement)。

从这个例子中可以得出一个结论，就是在舍弃进位的条件下，减去某个数可以用加上它的补码来代替。这个结论同样适用于二进制数的运算。一个 4 位二进制数的模为 16。例如：0111(7)是 -1001(9)对模 16 的补码。

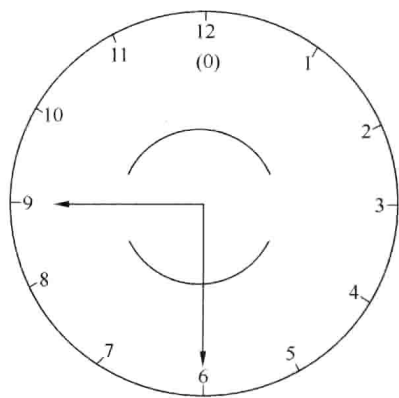


图 1-3 说明补码运算原理的例子

基于上述原理，对于有效数字（不包括符号位）为 n 位的二进制数 N ，它的补码 $(N)_{\text{COMP}}$ 表示方法为

$$(N)_{\text{COMP}} = \begin{cases} N & (\text{当 } N \text{ 为正数}) \\ 2^n - |N| & (\text{当 } N \text{ 为负数}) \end{cases} \quad (1-7)$$

由式 1-7 可知，正数（当符号位为 0 时）的补码与原码相同，负数（当符号位为 1 时）的补码等于 $(N)_{\text{INV}} + 1 = 2^n - N$ ，即负数的补码为其反码加 1。

【例 1-7】 写出二进制数的补码：(1) $X = +1011011$ (2) $Y = -1011011$ 。

解：(1) 根据定义有： $[X]_{\text{原码}} = 01011011$ $[X]_{\text{补码}} = 01011011$

(2) 根据定义有： $[Y]_{\text{原码}} = 11011011$ $[Y]_{\text{反码}} = 10100100$

$$[Y]_{\text{补码}} = 10100101$$

补码表示的整数范围是 $-2^{n-1} \sim 2^{n-1} - 1$ ，其中 n 为机器字长。8 位二进制补码表示的整数范围是 $-128 \sim +127$ （ -128 表示为 10000000，无对应的原码和反码）。16 位二进制补码表示的整数范围是 $-32768 \sim +32767$ 。当运算结果超出这个范围时，就不能正确表示数了，此时称为溢出。

4. 补码加、减运算规则

(1) 运算规则

$$[X+Y]_{补}=[X]_{补}+[Y]_{补}$$
$$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$$

(2) 溢出判断，一般用双符号位进行判断

符号位 00 表示正数；11 表示负数；结果的符号位为 01 时，称为上溢；为 10 时，称为下溢。

【例 1-8】 设 $x=+1101$ ， $y=-0111$ ，符号位为双符号位用补码求 $x+y$ ， $x-y$ 。

解：

$$[x]_{补}+[y]_{补}=00\ 1101+11\ 1001=00\ 0110$$
$$[x-y]_{补}=[x]_{补}+[-y]_{补}=00\ 1101+00\ 0111=01\ 0100。$$

上溢出，结果错误。

1.5 几种常用的编码

在数字系统中，用预先规定的方法将文字、数字或其他对象编成二进制的数码，这种给信息分配的二进制代码称为对信息的编码。

1.5.1 二-十进制编码

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 4 位二进制数表示 1 位十进制数的方法。这种二进制代码称为二-十进制代码（Binary Coded Decimal），简称为 BCD 码。

十进制数有 0~9 共 10 个数码，所以表示 1 位十进制数，至少需要 4 位二进制数，但 4 位二进制数可以产生 $2^4=16$ 种组合，因此用 4 位二进制数表示 1 位十进制数，有 6 种组合是多余的。因而，十进制数的二进制编码可以有许许多多种编码方案，每种编码都有它的特点。表 1-2 列举了目前常用的几种编码。

表 1-2 常用的十进制代码

十 进 制 数	8421 码	余 3 码	2421 码	5211 码	余 3 循环码
0	0000	0011	0000	0000	0010
1	0001	0100	0001	0001	0110
2	0010	0101	0010	0100	0111
3	0011	0110	0011	0101	0101
4	0100	0111	0100	0111	0100
5	0101	1000	1011	1000	1100
6	0110	1001	1100	1001	1101
7	0111	1010	1101	1100	1111
8	1000	1011	1110	1101	1110
9	1001	1100	1111	1111	1010