

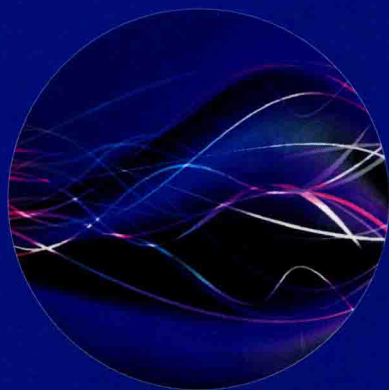


普通高等院校电子信息与电气工程类专业教材

周 斌 主 编

数字电路与逻辑设计

Shuzi Dianlu yu Luoji Sheji



华中科技大学出版社

<http://www.hustp.com>

普通高等院校电子信息与电气工程类专业教材

数字电路与逻辑设计

主 编 周 斌

副主编 蔡 苗 蔡红娟 陈 艳

华中科技大学出版社

中国·武汉

内 容 简 介

本书系统介绍了数字电路设计的基本理论、知识和器件,详细介绍了数字电路的分析和设计方法,同时,为了符合数字系统设计的发展趋势,引入了可编程逻辑器件和硬件描述语言。

本书共有 9 章,主要内容包括数字逻辑基础、组合逻辑电路、触发器、时序逻辑电路、数/模和模/数转换、脉冲波形的产生与变换、综合案例应用设计、可编程逻辑器件、Verilog HDL 硬件描述语言设计基础。

本书可作为高等院校电气信息类相关专业和部分非电类专业的基础教材和教学参考书,也可作为相关专业工程技术人员的参考书。

图书在版编目(CIP)数据

数字电路与逻辑设计/周斌主编. —武汉:华中科技大学出版社,2018.1
ISBN 978-7-5680-3147-9

I. ①数… II. ①周… III. ①数字电路-逻辑设计-教材 IV. ①TN79

中国版本图书馆 CIP 数据核字(2017)第 171060 号

数字电路与逻辑设计

Shuzi Dianlu yu Luoji Sheji

周 斌 主编

策划编辑:谢燕群

责任编辑:汪 黎

封面设计:原色设计

责任校对:何 欢

责任监印:周治超

出版发行:华中科技大学出版社(中国·武汉)

电话:(027)81321913

武汉市东湖新技术开发区华工科技园

邮编:430223

录 排:武汉市洪山区佳年华文印部

印 刷:武汉华工鑫宏印务有限公司

开 本:787mm×1092mm 1/16

印 张:22.25

字 数:537 千字

版 次:2018 年 1 月第 1 版第 1 次印刷

定 价:48.80 元



华中出版

本书若有印装质量问题,请向出版社营销中心调换
全国免费服务热线:400-6679-118 竭诚为您服务
版权所有 侵权必究

前 言

“数字电路与逻辑设计”是高等院校电气信息类相关专业学生必修的一门重要的学科专业基础课,其主要任务是使学生获得数字电路方面的基础理论、基本知识和基本技能,熟悉各种不同规模的逻辑器件,掌握各类逻辑电路分析与设计的基本方法,为今后学习相关专业课程以及专业应用奠定数字电路方面的理论基础。本书是按照“数字电路与逻辑设计”课程教学要求,以“夯实理论基础,强化应用能力”为原则,从传授知识和能力培养的目标出发,结合“数字电路与逻辑设计”课程的学习特点,根据教学改革与实践的需要编写的。

本书在编写过程中,力求做到概念清楚、内容完整实用、论述深入浅出、便于自学。本书对传统的内容进行了精选和更新,简明扼要地介绍了学生在学习该课程时,需要掌握的基础知识和技术。在保证基本理论完整性的原则下,精简分立元件电路内容,增强集成电路的应用,对集成电路的讨论强化“外部”,淡化“内部”,更加注重实用性和创新意识的培养。此外,鉴于可编程逻辑器件在数字系统中的应用越来越广泛,在本书中对大规模可编程逻辑器件以及 Verilog HDL 硬件描述语言进行了介绍。本书还通过例题、习题以及综合案例来说明理论的实际应用,以加深学生对本书内容的掌握和理解。

全书共有 9 章。第 1 章介绍数字电路的基本知识和基本理论,这些内容是分析和设计数字电路的基础。第 2 章、第 3 章和第 4 章以集成逻辑电路的应用为核心,介绍组合逻辑电路以及时序逻辑电路的设计和分析方法。第 5 章主要介绍数字电路和模拟电路之间的接口电路。第 6 章主要介绍各种脉冲信号的产生与变换电路。第 7 章综合运用课程知识,结合实际应用进行实际问题设计举例,旨在进一步理论联系实际,强化应用能力。第 8 章主要介绍大规模可编程逻辑器件及其在逻辑设计中的应用,使读者在学习数字电路的基本内容后,能够了解数字系统的概念,掌握数字系统设计的基本方法,进而能够从系统的高度来分析和解决设计问题。第 9 章主要介绍 Verilog HDL 硬件描述语言及其案例应用,使读者能够将本课程知识与当前流行的数字电路设计方法相融合,顺应数字电子技术的发展趋势。

本书由周斌、蔡苗、蔡红娟、陈艳共同编写。具体分工如下:周斌编写第 4.4 节、第 4 章习题、第 7 章、第 8 章、第 9 章;蔡苗编写第 2 章、第 5 章;蔡红娟编写第 3 章、第 6 章、第 4.3 节;陈艳编写第 1 章、第 4.1 节、第 4.2 节。周斌任主编,并负责全书的统稿、校订,蔡红娟、蔡苗和陈艳任副主编。在本书的编写过程中,翟晟、吕建才、黎贝贝等老师也做了大量的工作,在此对他们表示深深的谢意。此外,在本书的编写过程中,殷小贡、李海、徐安静老师对本书也进行了审阅,并提出了许多宝贵的意见,在此对他们表示衷心的感谢。

由于编者能力和水平有限,书中难免存在疏漏和不足之处,敬请广大读者批评指正。

编 者

2017 年 6 月于武汉

目 录

第 1 章 数字逻辑基础	(1)
1.1 概述	(1)
1.2 数制和代码	(1)
1.2.1 十进制数和二进制数	(1)
1.2.2 十六进制和八进制	(2)
1.2.3 不同进制数之间的转换	(3)
1.2.4 二进制符号数的表示法	(6)
1.2.5 二进制代码	(7)
1.3 逻辑运算	(9)
1.3.1 基本逻辑运算	(10)
1.3.2 复合逻辑运算	(11)
1.3.3 正负逻辑问题	(12)
1.4 逻辑门电路	(14)
1.4.1 半导体的开关特性	(14)
1.4.2 简单门电路	(17)
1.4.3 TTL 集成门电路	(18)
1.4.4 CMOS 集成门电路	(26)
1.5 逻辑函数的化简法	(27)
1.5.1 基本公式和定律	(28)
1.5.2 基本运算规则	(31)
1.5.3 逻辑函数代数法化简	(32)
1.6 逻辑函数的卡诺图化简法	(34)
1.6.1 最小项的定义及其性质	(34)
1.6.2 卡诺图	(35)
1.6.3 逻辑函数的卡诺图表示	(37)
1.6.4 逻辑函数卡诺图化简	(37)
1.6.5 具有约束的逻辑函数化简	(39)
1.7 逻辑函数的描述方法及转换	(41)
1.7.1 逻辑函数的描述方法	(41)
1.7.2 几种描述方法之间的转换	(43)
本章小结	(45)

习题 1	(45)
第 2 章 组合逻辑电路	(50)
2.1 组合逻辑电路概述	(50)
2.2 组合逻辑电路的分析与设计	(50)
2.2.1 组合逻辑电路的分析	(50)
2.2.2 组合逻辑电路的设计	(52)
2.3 组合逻辑电路中的竞争冒险	(56)
2.3.1 产生竞争冒险的原因	(57)
2.3.2 竞争冒险的判断	(57)
2.3.3 消除竞争冒险的方法	(58)
2.4 加法器与算术逻辑单元	(60)
2.4.1 半加器和全加器	(60)
2.4.2 集成加法器	(62)
2.4.3 算术逻辑单元	(64)
2.5 数值比较器	(66)
2.5.1 数值比较器的设计	(66)
2.5.2 集成数值比较器	(68)
2.6 编码器	(69)
2.6.1 编码器的工作原理	(70)
2.6.2 集成优先编码器	(73)
2.7 译码器与数据分配器	(76)
2.7.1 译码器的分析及设计	(76)
2.7.2 集成译码器	(78)
2.7.3 数据分配器	(86)
2.8 数据选择器	(87)
2.8.1 数据选择器的类型及功能	(87)
2.8.2 集成数据选择器	(89)
本章小结	(94)
习题 2	(95)
第 3 章 触发器	(99)
3.1 基本 RS 触发器	(99)
3.1.1 工作原理和逻辑功能	(99)
3.1.2 基本 RS 触发器的特点	(101)
3.1.3 集成 RS 触发器	(102)
3.2 同步触发器	(103)
3.2.1 同步 RS 触发器	(103)

3.2.2	同步 D 触发器	(104)
3.2.3	同步 JK 触发器	(106)
3.2.4	同步 T 触发器	(108)
3.2.5	同步触发器的特点	(110)
3.3	边沿触发器	(110)
3.3.1	边沿 D 触发器	(110)
3.3.2	边沿 JK 触发器	(111)
3.3.3	集成边沿触发器	(112)
3.4	不同类型触发器之间的相互转换	(114)
3.4.1	JK 触发器转换成 RS、D 和 T 触发器	(115)
3.4.2	D 触发器转换成 RS、JK 和 T 触发器	(116)
3.5	触发器的电气特性	(117)
3.6	触发器的应用举例	(118)
	本章小结	(120)
	习题 3	(121)
第 4 章	时序逻辑电路	(127)
4.1	时序逻辑电路概述	(127)
4.1.1	时序逻辑电路的特点及分类	(127)
4.1.2	时序逻辑电路的功能描述方法	(128)
4.2	时序逻辑电路的分析	(129)
4.2.1	时序逻辑电路的分析步骤	(129)
4.2.2	同步时序逻辑电路分析举例	(130)
4.2.3	异步时序逻辑电路分析举例	(136)
4.3	时序逻辑电路的设计	(138)
4.3.1	同步时序逻辑电路的设计	(138)
4.3.2	异步时序逻辑电路的设计	(146)
4.4	常用中规模集成时序逻辑电路	(148)
4.4.1	寄存器与移位寄存器	(148)
4.4.2	计数器	(153)
4.4.3	脉冲序列信号发生器	(167)
4.4.4	脉冲分配器	(171)
	本章小结	(172)
	习题 4	(173)
第 5 章	数/模转换与模/数转换	(179)
5.1	概述	(179)
5.2	DAC	(180)

5.2.1	D/A 转换的基本知识	(180)
5.2.2	常用的数模转换技术	(181)
5.2.3	数模转换器的性能指标	(184)
5.2.4	集成 DAC	(185)
5.3	ADC	(186)
5.3.1	A/D 转换的基本知识	(186)
5.3.2	常用的 A/D 转换技术	(188)
5.3.3	ADC 的性能指标	(194)
5.3.4	集成 ADC	(195)
	本章小结	(196)
	习题 5	(197)
第 6 章	脉冲波形的产生与变换	(201)
6.1	集成定时器 555	(201)
6.2	多谐振荡器	(202)
6.2.1	555 定时器构成的多谐振荡器	(203)
6.2.2	门电路构成的多谐振荡器	(205)
6.2.3	石英晶体多谐振荡器	(206)
6.2.4	多谐振荡器的应用	(207)
6.3	单稳态触发器	(209)
6.3.1	555 定时器构成的单稳态触发器	(209)
6.3.2	门电路构成的单稳态触发器	(211)
6.3.3	集成单稳态触发器	(214)
6.3.4	单稳态触发器的应用	(216)
6.4	施密特触发器	(217)
6.4.1	555 定时器构成的施密特触发器	(218)
6.4.2	门电路构成的施密特触发器	(219)
6.4.3	集成施密特触发器	(220)
6.4.4	施密特触发器的应用	(220)
	本章小结	(222)
	习题 6	(223)
第 7 章	综合案例应用设计	(226)
7.1	彩灯控制器设计	(226)
7.1.1	设计要求	(226)
7.1.2	基本结构	(227)
7.1.3	设计实现	(228)
7.2	温度监控报警电路设计	(231)

7.2.1 设计要求	(231)
7.2.2 基本结构	(231)
7.2.3 设计实现	(232)
7.3 交通灯信号控制器设计	(235)
7.3.1 设计要求	(235)
7.3.2 基本结构	(235)
7.3.3 设计实现	(236)
本章小结	(239)
习题 7	(239)
第 8 章 可编程逻辑器件	(240)
8.1 概述	(240)
8.2 基本结构和表示方法	(241)
8.2.1 基本结构	(241)
8.2.2 PLD 电路的表示方法	(241)
8.2.3 PLD 的分类	(244)
8.3 低密度可编程逻辑器件	(244)
8.3.1 可编程只读存储器	(244)
8.3.2 可编程逻辑阵列	(246)
8.3.3 可编程阵列逻辑	(246)
8.3.4 通用阵列逻辑(GAL)	(247)
8.4 复杂可编程逻辑器件	(249)
8.4.1 CPLD 的基本结构	(250)
8.4.2 典型 CPLD 器件的结构	(250)
8.5 现场可编程门阵列	(254)
8.5.1 FPGA 的基本结构	(255)
8.5.2 典型 FPGA 器件的结构	(257)
8.6 CPLD/FPGA 的设计流程和编程	(263)
8.6.1 CPLD/FPGA 的设计流程	(264)
8.6.2 CPLD 器件的编程	(265)
8.6.3 FPGA 器件的配置	(265)
本章小结	(268)
习题 8	(269)
第 9 章 Verilog HDL 硬件描述语言设计基础	(270)
9.1 Verilog 程序的基本结构	(270)
9.2 Verilog 语言要素	(272)
9.3 Verilog 常量	(272)

9.3.1	整数	(273)
9.3.2	实数	(274)
9.3.3	字符串	(274)
9.3.4	符号常量	(275)
9.4	数据类型	(276)
9.4.1	线网(net)类型	(276)
9.4.2	寄存器类型	(277)
9.4.3	向量	(279)
9.5	Verilog 的运算符	(281)
9.6	Verilog 的行为级建模	(285)
9.6.1	过程语句	(285)
9.6.2	语句块	(289)
9.6.3	赋值语句	(291)
9.6.4	程序控制语句	(293)
9.6.5	Verilog 的编译指示语句	(299)
9.6.6	任务和函数	(302)
9.7	Verilog 的结构级建模	(306)
9.7.1	门级建模	(306)
9.7.2	用户自定义元件	(309)
9.7.3	模块级建模	(314)
9.8	数字电路的 Verilog 描述实例	(321)
9.8.1	常用组合逻辑电路的 Verilog 描述	(322)
9.8.2	常用时序逻辑电路的 Verilog 描述	(328)
9.8.3	有限状态机的 Verilog 描述	(335)
	本章小结	(340)
	习题 9	(341)
	附录 A Verilog HDL(IEEE Std 1364—2001) 支持的关键字	(343)
	参考文献	(344)

第 1 章 数字逻辑基础

本章是学习数字逻辑电路的基础,首先介绍数制与代码,然后介绍逻辑代数的定义,以及逻辑运算、逻辑代数的基本公式、定律和法则,并在此基础上详细介绍逻辑函数的化简方法,最后简要介绍逻辑函数的表示方法及各表示方法之间的相互转换。

1.1 概 述

在实际生活中,存在着两类物理量:一类称为数字量,它具有时间上离散变化、值域内只能取某些特定值的特点;另一类称为模拟量,它具有时间上连续变化、在一定动态范围内任意取值的特点。在电子设备中,数字量和模拟量都是以电信号形式出现的。人们常常将表示模拟量的电信号称为模拟信号,将表示数字量的电信号称为数字信号。数字信号是一种脉冲信号,脉冲信号具有边沿陡峭、持续时间短的特点。广义地讲,凡是非正弦信号都称为脉冲信号。

在电子电路中,具有对数字信号进行产生、存储、变换、处理、传送的电子电路称为数字电路。数字电路不仅能够完成算术运算,而且能够完成逻辑运算。它具有逻辑推理和逻辑判断的能力,因此也称数字逻辑电路或逻辑电路。在数字逻辑电路中,输入量和输出量的稳定状态通常都用电位的高和低、脉冲的有和无来表示,因此数字逻辑电路的输入和输出可以抽象为逻辑命题的真和假。在二值逻辑中,变量的取值不是“1”就是“0”,没有第三种可能。而且这里的“1”和“0”并不是表示数值的大小,它们代表的只是两种不同的逻辑状态。

数字电路中的电子器件都工作在开关状态,电路的输出只有高、低两个电平,因而很容易实现二值逻辑。在分析实际电路时,逻辑高电平和逻辑低电平都对应一定的电压范围,不同系列的数字集成电路,其输入、输出为高电平或低电平时所对应的电压范围是不同的,一般用逻辑高电平表示逻辑 1 和二进制数的 1,用逻辑低电平表示逻辑 0 和二进制数的 0。

数字电路只能处理用二进制数表示的数字信号,而人们习惯使用的十进制数是不能被数字电路直接识别并处理的,因此,为了便于人与数字电路的信号交换和传输,需要研究各种进制之间的相互转换以及不同的编码方式。

1.2 数制和代码

数制是人们对数量计算的一种统计规律,是计数进位制的简称。在日常生活中,最常用的数制是十进制。而在数字系统中,多采用二进制数,有时也采用八进制数或十六进制数。

1.2.1 十进制数和二进制数

1. 十进制数

十进制是广泛使用的计数进位制。这种计数进位制的每一位数都用 0~9 十个数码中

的一个数码来表示,其计数基数是十。计数规则为“逢十进一、借一当十”,故称为十进制。一个数的大小取决于数码的位置,即数位。数码相同,所在的位置不同,则数的大小不同。例如:十进制数 2016.86 的展开式为

$$2016.86 = 2 \times 10^3 + 0 \times 10^2 + 1 \times 10^1 + 6 \times 10^0 + 8 \times 10^{-1} + 6 \times 10^{-2}$$

式中,10 称为基数, 10^0 、 10^1 、 10^2 、 10^3 称为各位数的“权”。十进制数的个位的权为 1,十位的权为 10,百位的权为 100……任何一个十进制数 N_D 可表示为

$$\begin{aligned} N_D &= d_{n-1} \times 10^{n-1} + d_{n-2} \times 10^{n-2} + \cdots + d_1 \times 10^1 + d_0 \times 10^0 + \cdots + d_{-m} \times 10^{-m} \\ &= \sum_{i=-m}^{n-1} d_i \times 10^i \end{aligned}$$

式中: d_i 为各位数的数码; 10^i 为各位数的权;所对应的数值为 $d_i \times 10^i$ 。

2. 二进制数

在数字系统中,应用最广泛的数是二进制数。在二进制数中,每一位仅有 0 和 1 两个可能的数码,所以计数基数是 2。低位和相邻的高位之间的进位关系是“逢二进一、借一当二”,故称之为二进制。任意一个二进制数 N_B 的展开式为

$$\begin{aligned} N_B &= b_{n-1} \times 2^{n-1} + b_{n-2} \times 2^{n-2} + \cdots + b_1 \times 2^1 + b_0 \times 2^0 + \cdots + b_{-m} \times 2^{-m} \\ &= \sum_{i=-m}^{n-1} b_i \times 2^i \end{aligned}$$

式中:2 为基数; 2^i 为各位数的权; b^i 为各位数的数码。

例如:一个二进制数 11010.101 可展开为

$$11010.101 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

从二进制数的特点中可以看到它具有的优点。第一,只有两个数码,只需反映两种不同稳定状态的元件就可表示一位数。因此,构成二进制数电路的基本单元结构简单。第二,储存和传递可靠。第三,运算规则简单,操作方便。所以在数字系统中基本都采用二进制数。

1.2.2 十六进制和八进制

1. 十六进制

十六进制的每一位数都有十六种可能出现的数字,分别用 0~9、A~F 表示数,其中 A~F 分别对应十进制数的 10~15。低位数和高一位数之间的计数规律是“逢十六进一、借一当十六”。任意一个十六进制数均可展开为

$$\begin{aligned} N_H &= h_{n-1} \times 16^{n-1} + h_{n-2} \times 16^{n-2} + \cdots + h_1 \times 16^1 + h_0 \times 16^0 + h_{-1} \times 16^{-1} + \cdots + h_{-m} \times 16^{-m} \\ &= \sum_{i=-m}^{n-1} h_i \times 16^i \end{aligned}$$

式中:16 为基数; 16^i 为各位数的权; h_i 为 i 位数的数码。

例如:一个十六进制数 69AB.7F 可展开为

$$\begin{aligned} 69AB.7F &= 6 \times 16^3 + 9 \times 16^2 + A \times 16^1 + B \times 16^0 + 7 \times 16^{-1} + F \times 16^{-2} \\ &= 6 \times 16^3 + 9 \times 16^2 + 10 \times 16^1 + 11 \times 16^0 + 7 \times 16^{-1} + 15 \times 16^{-2} \end{aligned}$$

2. 八进制

在八进制数中,每一位用 0~7 八个数码表示,所以计数基数为 8。低位数和高一位数之

间的关系是“逢八进一”。任何一个八进制数都可以展开为

$$N_O = o_{n-1} \times 8^{n-1} + o_{n-2} \times 8^{n-2} + \cdots + o_1 \times 8^1 + o_0 \times 8^0 + o_{-1} \times 8^{-1} + \cdots + o_{-m} \times 8^{-m}$$

$$= \sum_{i=-m}^{n-1} o_i \times 8^i$$

式中:8 为基数; 8^i 为各位数的权; o_i 为 i 位数的数码。

1.2.3 不同进制数之间的转换

目前由于在微型计算机中普遍采用 8 位或 16 位二进制数并行运算,而 8 位和 16 位二进制数可以用 2 位和 4 位十六进制数来表示,因此用十六进制符号书写程序十分简便。同时,十六进制数和十进制数之间的转换又非常简单,这就使得十六进制的应用比八进制的应用还要广泛。为此,要熟练地掌握十进制数、二进制数和十六进制数间的相互转换。它们之间的关系如表 1-1 所示。

表 1-1 十进制数、二进制数及十六进制数对照表

十进制数	二进制数	十六进制数	十进制数	二进制数	十六进制数
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

为了区别十进制、二进制及十六进制这三种数制,可用括号把数括起来,在括号的右下角注明相应数制的进位计数,或者在数字后面加上相应的大写字母,标示出数制。如 B(binary)表示二进制数制,D(decimal)或不带字母表示十进制数制,H(hexadecimal)表示十六进制数制。

1. 二进制数转换为十六进制数

由于 4 位二进制数恰好有十六个状态,而且当把这 4 位二进制数看成一个数位时,它向高位的进位又正好是逢十六进一,因此可以用 4 位二进制数代表 1 位十六进制数。根据表 1-1 所示的对应关系即可实现二进制数和十六进制数之间的转换。

将二进制整数转换为十六进制数,其方法是从小数点向左将二进制整数部分分组,每 4 位为一组,最后一组若不足 4 位则在其左边添加 0 以凑成 4 位一组,每组用 1 位等值的十六进制数表示。对于小数部分的转换,则应从小数点向右将二进制小数部分分组,每 4 位为一组,最后一组若不足 4 位则在其右边添加 0 以凑成 4 位一组,然后每组用 1 位等值的十六进制数表示。例如:

$$1111101011.01111010101\text{ B}= 0011\ 1110\ 1011.0111\ 1010\ 1010\text{ B}= 3\text{EB}.7\text{AA H}$$

2. 十六进制数转换为二进制数

十六进制数转换为二进制数,只需将十六进制的每位转换为 4 位二进制数,并按原来的顺序排列起来即可。例如:

$$C5EA.49H = 1100\ 0101\ 1110\ 1010.0100\ 1001\ B$$

3. 十六进制数转换为十进制数

十六进制数转换为十进制数十分简单,只需将十六进制数按位权展开,然后按照十进制数的运算法则相加即可。例如:

$$\begin{aligned} 3FA.5\ H &= 3 \times 16^2 + 15 \times 16^1 + 10 \times 16^0 + 16^{-1} \times 5 \\ &= 3 \times 256 + 15 \times 16 + 10 \times 1 + 0.0625 \times 5 \\ &= 768 + 240 + 10 + 0.3125 = 1018.3125 \end{aligned}$$

4. 十进制数转换为十六进制数

1) 整数部分的转换

十进制整数转换为非十进制数时,一般采用“除基取余”法。将十进制数不断除以将转换进制的基数,直至商为 0。每除一次取余数,先得到的余数为低位,后得到的余数为高位,最后由余数依次从高位向低位排列的数就是转换的结果。十进制整数转换为十六进制整数可用除 16 取余法,即用 16 不断地去除待转换的十进制数,直至商等于 0 为止。将所得的各次余数依倒序排列,即可得到所转换的十六进制数。如将 38947 转换为十六进制数,其方法及算式如下:

		余数
16	38947	3 最低位
16	2434	2
16	152	8
	9	9 最高位
	0	

即 $38947 = 9823\ H$ 。

2) 小数部分的转换

十进制小数转换为非十进制数时,可用“乘基取整”法。用十进制数的小数部分反复乘以基数,直到小数部分为 0 或达到转换精度要求的位数(小数部分永不为 0,可根据精度要求的位数决定转换后的小数位数),依次取积的整数(为十进制数),第一个整数为最高位,最后一个整数为最低位,从最高小数位依次排到最低小数位即可。如将 0.3584 转换为十六进制数,结果保留 3 位小数,其方法及算式如下:

$0.3584 \times 16 = 5.7344$	取整 5	最高位
$0.7344 \times 16 = 11.7504$	取整 11	
$0.7504 \times 16 = 12.0064$	取整 12	最低位

即 $0.3584 = 0.5BC\ H$ 。

例如:将 39.625 转换为十六进制数,算式如下:

16

39

16

2

0

余数

.....7

.....2

0.625

×

16

3750

+

625

10.000

.....A

即 39.625=27. A H。

5. 二进制数与十进制数间的相互转换

1) 利用十六进制数为桥梁实现二进制数与十进制数间的转换

把一个十进制数转换为二进制数,可以先把该数转换为十六进制数,然后再转换为二进制数,这样可以减少计算次数;反之,要把一个二进制数转换为十进制数,也可以采用同样的办法。若使用 2ⁿ(2ⁿ 的二进制数等于 1 后跟 n 个 0)和十六进制数、十进制数的对应关系(见表 1-2),以及个别十进制整数和十六进制数的对应关系(如 50=32 H,80=50 H,100=64 H 等),则转换起来更为方便。例如:

38947 = 32768+4096+2048+32+3=8000H+1000H+800H+20H+3H

=9823H=1001 1000 0010 0011B

0001 1111 0011 1101B=1F3DH=2000H-(80H+40H+3H)

=8192-(128+64+3)=7997

表 1-2 部分二进制数与十进制数的对应关系

2 ⁿ	十六进制数	十进制数	常用缩写	2 ⁿ	十六进制数	十进制数	常用缩写
2 ⁵	20	32		2 ¹³	2000	8192	8K
2 ⁶	40	64		2 ¹⁴	4000	16384	16K
2 ⁷	80	128		2 ¹⁵	8000	32768	32K
2 ⁸	100	256		2 ¹⁶	10000	65536	64K
2 ⁹	200	512		2 ²⁰			1M
2 ¹⁰	400	1024	1K	2 ³⁰			1G
2 ¹¹	800	2048	2K	2 ⁴⁰			1T
2 ¹²	1000	4096	4K	2 ⁵⁰			1P

2) 采用将二进制数按位权展开相加的方法将二进制数转换为十进制数

例如,将二进制数 1011.01 转换为十进制数,转换过程如下:

1011.011 B=1×2³+0×2²+1×2¹+1×2⁰+1×2⁻¹+0×2⁻²+1×2⁻³+1×2⁻⁴

=8+0+2+1+0.5+0+0.125+0.0625

=11.6875

3) 采用“除基取余”法和“乘基取整”法将十进制数转换为二进制数

例如,将十进制数 39.625 转换成二进制数,转换过程如下:

2	39	1		0.625	
2	19	1	$\times$	2	
2	9	1		1.250	1
2	4	0		2	
2	2	0		0.500	0
2	1	1		2	
0				1.000	1

即 $39.625 = 10\ 0111.101\ \text{B}$ 。

1.2.4 二进制符号数的表示法

1. 机器数与真值

在计算机中,常用数的符号和数值部分一起编码的方法表示符号数。二进制数与十进制数一样有正负之分。通常正号用“0”表示,负号用“1”表示。为了区分,一般书写时通常将用“+”“-”表示的正、负二进制数称为数的真值,而把数值连同符号数码“0”或“1”一起编码表示的二进制数称为机器数。把机器数的符号位也当作数值的数,就是无符号数。常用的机器数表示法有原码、反码和补码表示法。这几种表示法都将数的符号数码化。

2. 原码

原码表示法是机器数的一种简单的表示法。数值用其绝对值,正数的符号位用 0 表示,负数的符号位用 1 表示,即“符号+绝对值”。例如:

$$X_1 = +24, \quad [X_1]_{\text{原}} = 011000$$

$$X_2 = -1101, \quad [X_2]_{\text{原}} = 11101$$

$$X_3 = 0000, \quad [X_3]_{\text{原}} = 00000 \text{ 或 } [X_3]_{\text{原}} = 10000$$

原码表示法简单易懂,而且原码与真值转换方便。若是两个异号数相加,或两个同号数相减,就要做减法。

3. 反码

用反码表示带符号的二进制数时,符号位与原码相同,即用 0 表示正号,用 1 表示负号。不同的是正数反码的数值部分与原码数值部分相同,而负数反码的数值部分是原码数值部分按位取反。例如:

$$X_1 = 5, \quad [X_1]_{\text{原}} = 0101, \quad [X_1]_{\text{反}} = 0101$$

$$X_2 = -5, \quad [X_2]_{\text{原}} = 1101, \quad [X_2]_{\text{反}} = 1010$$

4. 补码

用补码表示带符号的二进制数时,符号位与原码、反码相同,即用 0 表示正号,用 1 表示负号。不同的是正数补码的数值部分与原码、反码数值部分相同,而负数补码的数值部分是原码数值部分按位取反,并在最低位加 1,即先求负数的反码,再将反码加 1。例如:

$$X_1 = 1010110, \quad [X_1]_{\text{原}} = 01010110, \quad [X_1]_{\text{反}} = 01010110, \quad [X_1]_{\text{补}} = 01010110$$

$$X_2 = -1001010, \quad [X_2]_{\text{原}} = 11001010, \quad [X_2]_{\text{反}} = 10110101, \quad [X_2]_{\text{补}} = 10110110$$

求补数还可以直接求,方法是从最低位向最高位扫描,保留直至第一个“1”的所有位,以后各位按位取反。负数的补码可以由与其绝对值相等的正数求补得到。根据两数互为补数的原理,对补码表示的负数求补就可以得到该负数的绝对值。例如:

$$[-105]_{补} = 10010111 \text{ B} = 97\text{H}$$

对其求补,从右向左扫描,第一位就是1,故只保留该位,对其左面的7位均求反得01101001,即补码表示的机器数97H的真值是-69H(=-105)。

一个用补码表示的机器数,若最高位为0,则其余几位即为此数的绝对值。若最高位为1,则其余几位不是此数的绝对值,对该数(连同符号位)求补,才得到它的绝对值。

当数采用补码进行加减运算时,可以将加减运算均通过加法实现。例如:

$$\begin{aligned} 64 - 10 &= 64 + (-10) \\ [64]_{补} &= 40\text{H} = 0100\ 0000\text{B} \\ [10]_{补} &= 0\text{AH} = 0000\ 1010\text{B} \\ [-10]_{补} &= 1111\ 0110\text{B} \end{aligned}$$

做减法运算过程如下:

$$\begin{array}{r} 0100\ 0000 \\ -0000\ 1010 \\ \hline 0011\ 0110 \end{array}$$

用补码相加过程如下:

$$\begin{array}{r} 0100\ 0000 \\ +1111\ 0110 \\ \hline 1\ 0011\ 0110 \end{array}$$

↑ 进位自然丢失

结果相同,其真值为54(36H=48+6)。

最高位的进位是自然丢失的,故做减法与用补码相加的结果是相同的。

1.2.5 二进制代码

数字系统不仅用到数字,还要用到各种字母、符号和控制信号等。为了表示这些信息,将字母、数字、符号和信息等用一组特定的二进制数来表示的过程称为编码。而这种有特定含义的二进制数码称为二进制代码。常用的二进制代码有二-十进制代码和ASCII码。

1. 二-十进制代码

二-十进制代码是用4位二进制代码来表示1位十进制数的编码,称为二-十进制代码,简称BCD(binary coded decimal)码。因为1位十进制数有0~9十个数码,至少需要4位二进制编码才能表示1位十进制数。4位二进制数可以表示十六种不同的状态,用它来表示1位十进制数时就要丢掉六种状态。根据所用十种状态与1位十进制数码对应关系的不同,产生了各种BCD码,最常用的是8421BCD码、2421BCD码、4221BCD码、5421BCD码、余3码等。2421BCD码、4221BCD码、5421BCD码的变换和8421BCD码相似,只是权不同。余3码可以由8421BCD码得到,即某十进制数的8421BCD码加3所对应的二进制码便是该十