第3章　计算机运算基础

在计算机中，由于使用电子器件的不同状态来表示信息，而电信号一般只有两种状态，如高电平和低电平、通路和断路，因此，计算机内部是一个二进制数字世界。计算机之所以具有逻辑处理能力，是由于计算机内部具有能够实现各种逻辑功能的逻辑电路，逻辑代数是进行逻辑电路设计的数学基础。本章介绍逻辑代数及其运算、二进制以及与十进制的转换、数值数据在计算机中的表示方法、逻辑电路的基本原理，它们构成了计算机的运算基础。本章的内容将在“离散数学”、“计算机原理”、“数字电路与逻辑设计”等课程中进一步介绍。

3.1　数理逻辑基础

数理逻辑与计算机学科关系密切，在计算机科学与技术的许多领域都有广泛的应用，例如逻辑电路的设计以逻辑代数为基础，人工智能的许多领域（如知识的表示和推理、自然语言理解等）与数理逻辑密切相关，程序正确性证明以数理逻辑为基础，程序语言学的研究从模型论衍生而来。

数理逻辑的主要分支有公理化集合论、证明论、递归函数论、模型论等。逻辑代数是数理逻辑的基础部分，而逻辑代数源于对命题逻辑的研究。

3.1.1　数理逻辑的起源和发展

逻辑(logic)一词源于希腊文 logoc，有“思维”和“表达思考的言辞”之意。一直以来，人们希望使用数学方法来研究思维，具体地说，使用一种符号语言来代替自然语言对思维过程进行描述，把人类的思维过程转换为数学的计算。

最早提出用数学方法来描述和处理思维的是德国数学家莱布尼茨(G. W. Leibnitz)，但直到 1847 年英国数学家乔治·布尔(George Boole)发表著作《逻辑的数学分析》后才有所发展。1879 年德国数学家弗雷格(G. Frege)在《概念语言——一种按算术的公式语言构成的纯思维公式语言》一书中建立了第一个比较严格的逻辑演算系统，英国逻辑学家怀特海(A. N. Witehead)和罗素(B. Russell)合著的《数学原理》一书，对当时数理逻辑的成果进行了总结，使得数理逻辑形成了专门的学科。

> 英国数学家乔治·布尔(George Boole)16 岁就开始任教以维持生活，20 岁时对数学产生了浓厚的兴趣，开始广泛涉猎著名数学家牛顿、拉普拉斯、拉格朗日等人的数学名著，并写下了大量笔记。1847 年，发表了著作 *The Mathematical Analysis of Logic*，在这本书中，阐述了逻辑学公理，建立了逻辑代数，因此，逻辑代数也称为布尔代数。逻辑代数是建立在两个逻辑值 0、1 和三个运算符与、或、非的基础上，这种简化的二值逻辑为计算机的二进制数、开关逻辑元件和逻辑电路的设计铺平了道路，并最终为计算机的发明奠定了数学基础。

数理逻辑是用数学的方法来研究推理规律的科学，它采用符号的方法来描述和处理思维形式、思维过程和思维规律，进一步地，数理逻辑就是研究推理中前提和结论之间的形式关系，这种形式关系是由作为前提和结论的命题的逻辑形式决定的，因此，数理逻辑又称为形式逻辑或符号逻辑。

数理逻辑最显著的特征就是**符号化**和**形式化**，即把逻辑思维所涉及的概念、判断、推理用符号来表示，用公理化体系来刻画，并基于符号串形式的演算来描述推理过程的一般规律，从而实现人类思维过程的演算化、机械化，最终计算机化（即在计算机上实现）。1930 年以前，数理逻辑的发展主要是针对纯数学的需要，其后，数理逻辑开始应用于所有开关线路的理论中，并在计算机科学等方面获得应用，成为计算机科学的基础理论之一。

3.1.2 命题逻辑与命题代数

命题是推理的基本要素，**命题逻辑**研究以命题为基本单位构成的前提和结论之间的推理关系，因此又称命题演算。所谓**命题**是一个有具体意义且能够判断真假的陈述句。判断是对事物表示肯定或否定的一种思维形式，所以表达判断的命题总是具有“真”(true，T)或“假”(false，F)两种取值，命题所具有的值称为命题的真值。

命题分为**原子命题**和**复合命题**两种类型。原子命题是不能分解为更为简单的陈述句的命题，而复合命题则是将原子命题用连接词复合而成的命题。

例 3.1 以下是几个命题实例。

(1) 长春是吉林省的省会城市。

(2) 3 乘以 8 等于 16。

(3) 姚大龙既擅长书法又擅长绘画。

(4) 如果今天不下雨，我就去逛街。

其中，第一个命题是原子命题，该命题为真，即它的真值为 T；第二个命题也是原子命题，该命题为假，即它的真值为 F；第三个命题和第四个命题是复合命题，其真值需要根据实际情况确定。

命题代数和普通代数一样，用字母 $A,B,C,\cdots$ 表示变量，称为命题变量（或命题变元），但是命题变元的取值只有两种：T 或 F。连接词相当于普通代数中的运算符，在命题代数中主要的连接词有与、或、非、异或、条件和双条件等。

两个命题 A 和 B 的与（又称 A 和 B 的合取）记为 $A \wedge B$，表示当且仅当 A 和 B 同时为真时 $A \wedge B$ 为真，其他情况下 $A \wedge B$ 为假，其真值表如图 3.1(a)所示。

A	B	$A \wedge B$
T	T	T
T	F	F
F	T	F
F	F	F

(a) 与的真值表

A：姚大龙擅长书法。
B：姚大龙擅长绘画。
$A \wedge B$：姚大龙既擅长书法又擅长绘画。
只有当命题 A 和 B 均为真时 $A \wedge B$ 才为真。

(b) 与运算的一个实例

图 3.1 与运算

两个命题 A 和 B 的或（又称 A 和 B 的析取）记为 $A \vee B$，表示当且仅当 A 和 B 同时为

假时 $A \vee B$ 为假，其他情况下 $A \vee B$ 为真，其真值表如图 3.2(a)所示。

A	B	$A \vee B$
T	T	T
T	F	T
F	T	T
F	F	F

(a) 或的真值表

A：姚大龙擅长书法。
B：姚大龙擅长绘画。
$A \vee B$：姚大龙擅长书法或绘画。
只有当命题 A 和 B 均为假时 $A \vee B$ 才为假。

(b) 与运算的一个实例

图 3.2　或运算

命题 A 的非(又称 A 的否)记为 $\neg A$，表示当 A 为真时 $\neg A$ 为假，当 A 为假时 $\neg A$ 为真，其真值表如图 3.3(a)所示。

A	$\neg A$
T	F
F	T

(a) 非的真值表

A：姚大龙擅长书法。
$\neg A$：姚大龙不擅长书法。
当命题 A 为真时 $\neg A$ 为假。

(b) 非运算的一个实例

图 3.3　非运算

两个命题 A 和 B 的异或(又称 A 和 B 的不可兼或)记为 $A \oplus B$，表示当且仅当 A 和 B 同时为真或同时为假时 $A \oplus B$ 为假，其他情况下 $A \oplus B$ 为真，其真值表如图 3.4(a)所示。

A	B	$A \oplus B$
T	T	F
T	F	T
F	T	T
F	F	F

(a) 异或的真值表

A：姚大龙擅长书法。
B：姚大龙擅长绘画。
$A \oplus B$：姚大龙书法或绘画只擅长其一。
当命题 A 和 B 均为真或均为假时 $A \oplus B$ 为假。

(b) 异或运算的一个实例

图 3.4　异或运算

两个命题 A 和 B 的条件(又称如果 A 则 B)记为 $A \rightarrow B$，表示当且仅当 A 为真且 B 为假时，$A \rightarrow B$ 为假，在其他情况下 $A \rightarrow B$ 为真，其真值表如图 3.5(a)所示。需要强调的是，条件运算的命题 A 和 B 之间没有要求一定具有因果关系。

A	B	$A \rightarrow B$
T	T	T
T	F	F
F	T	T
F	F	T

(a) 条件的真值表

A：雪是白的。
B：太阳从西边出来。
$A \rightarrow B$：如果雪是白的，则太阳从西边出来。
当命题 A 为真而 B 为假时 $A \rightarrow B$ 为假。

(b) 条件运算的一个实例

图 3.5　条件运算

两个命题 A 和 B 的双条件(又称 A 当且仅当 B)记为 $A \leftrightarrow B$，表示当且仅当 A 的真值与 B 的真值相同时，$A \leftrightarrow B$ 为真，在其他情况下 $A \leftrightarrow B$ 为假。其真值表如图 3.6(a)所示。需要强调的是，双条件运算的命题 A 和 B 之间也没有要求一定具有因果关系，但要求是等价

关系。

A	B	$A\leftrightarrow B$
T	T	T
T	F	F
F	T	F
F	F	T

(a) 双条件的真值表

A：雪是白的。
B：太阳从西边出来。
$A\leftrightarrow B$：雪是白的当且仅当太阳从西边出来。
当命题 A 和 B 均为真或均为假时 $A\leftrightarrow B$ 为真。

(b) 双条件运算的一个实例

图 3.6　双条件运算

由命题变元、连接词和括号组成的式子称为命题公式，连接词的优先级从强到弱依次为 $\neg$、$\wedge$、$\vee$、$\oplus$、$\rightarrow$、$\leftrightarrow$。

例 3.2　将下列语句形式化，并表示为命题公式。

(1) 我和他既是同学又是兄弟。

可表示为 $A\wedge B$，其中 A：我和他是兄弟，B：我和他是同学。

(2) 我和他之间至少有一个去西部。

可表示为 $A\vee B$，其中 A：我去西部，B：他去西部。

(3) 狗急了要跳墙。

可表示为 $A\rightarrow B$，其中 A：狗急了，B：狗跳墙。

(4) 除非他努力学习，否则他的学习成绩不会提高。

可表示为 $A\leftrightarrow B$，其中 A：他努力学习，B：他的学习成绩会提高。

(5) 如果他没来，那么或者他生病了，或者他不在本市。

可表示为 $\neg A\rightarrow(B\vee\neg C)$，其中 A：他来了，B：他生病了，C：他在本市。

对于两个不同的命题公式 P 和 Q，如果无论其命题变元取什么值，P 和 Q 的真值都相同，则称这两个命题公式等价，记为 $P=Q$。利用命题公式的等价关系可以将命题公式化简或进行变换。下面给出了一些重要的等价律。

- 零律：$A\vee \mathrm{F}=A$　　$A\wedge \mathrm{F}=\mathrm{F}$
- 幺律：$A\vee \mathrm{T}=\mathrm{T}$　　$A\wedge \mathrm{T}=A$
- 幂等律：$A\vee A=A$　　$A\wedge A=A$
- 互补律：$A\vee\neg A=\mathrm{T}$　　$A\wedge\neg A=\mathrm{F}$
- 交换律：$A\vee B=B\vee A$　　$A\wedge B=B\wedge A$
- 结合律：$A\vee(B\vee C)=(A\vee B)\vee C$　　$A\wedge(B\wedge C)=(A\wedge B)\wedge C$
- 分配律：$A\wedge(B\vee C)=A\wedge B\vee A\wedge C$　　$(A\vee B)\wedge C=A\wedge C\vee B\wedge C$
- 吸收律：$A\wedge B\vee A\wedge\neg B=A$　　$(A\vee B)\wedge(A\vee\neg B)=A$
- 摩根定律：$\neg(A\vee B)=\neg A\wedge\neg B$　　$\neg(A\wedge B)=\neg A\vee\neg B$
- 双重否定律：$\neg\neg A=A$

3.1.3　逻辑代数

可以将命题代数直接推广到逻辑代数。逻辑代数又称布尔代数，是 19 世纪英国数学家乔治·布尔创立的。1938 年，香农(C. E. Shannon)将逻辑代数直接应用于开关电路，因此，

逻辑代数也称为开关代数。

香农(C. E. Shannon)1916 年出生于美国密歇根州,1936 年毕业于密歇根大学获得数学和电子工程学士学位,1940 年获得麻省理工学院数学博士学位和电子工程硕士学位。1938 年,香农发明了以脉冲方式处理信息的继电器开关,彻底改变了数字电路的设计。香农在 1948 年发表的论文《通讯的数学原理》,1949 年发表的论文《噪声下的通信》中,解决了过去许多悬而未决的问题,被尊称为“信息论之父”。

逻辑代数是以代数的方式对逻辑变量进行描述和运算的数学工具。描述逻辑变量关系的函数称为**逻辑函数**(或称逻辑表达式),实现逻辑函数的电路称为**逻辑电路**,用逻辑电路可以做成计算机系统中常用的部件,称为**逻辑部件**。逻辑代数是进行逻辑电路设计的数学基础,广泛用于数字电路的分析与设计。

在逻辑代数中,逻辑变量的取值只有“真”和“假”,通常以 1 和 0 来表示;通常用符号“·”表示与运算(在不至于混淆的情况下,符号·也可以省略),用符号+表示或运算,用上划线“¯”表示非运算。逻辑运算的基本规则如表 3.1 所示。需要强调的是,参与逻辑运算的数值没有符号位。

表 3.1　逻辑运算的基本规则

与	或	非
$0\cdot0=0$	$0+0=0$	
$0\cdot1=0$	$0+1=1$	非 0=1
$1\cdot0=0$	$1+0=1$	非 1=0
$1\cdot1=1$	$1+1=1$	

命题代数中的等价律在逻辑代数中仍然成立,只需要将 T 替换为 1,将 F 替换成 0 即可。这样,逻辑代数中的等价律表示为:

- 零律:$A+0=A$　　$A0=0$
- 幺律:$A+1=1$　　$A1=A$
- 幂等律:$A+A=A$　　$AA=A$
- 求补律:$A+\overline{A}=1$　　$A\overline{A}=0$
- 交换律:$A+B=B+A$　　$AB=BA$
- 结合律:$A+(B+C)=(A+B)+C$　　$A(BC)=(AB)C$
- 分配律:$A(B+C)=AB+AC$　　$(A+B)C=AC+BC$
- 吸收律:$AB+A\overline{B}=A$　　$(A+B)(A+\overline{B})=A$
- 摩根定律:$\overline{A+B}=\overline{A}\,\overline{B}$　　$\overline{A}\,\overline{B}=\overline{A+B}$
- 双重否定律:$\overline{\overline{A}}=A$

3.2　二　进　制

逻辑代数只使用 1(真)和 0(假)两个数,这样,当二进制的加法、乘法等算术运算与逻辑代数的逻辑运算建立了对应关系后,就可以用逻辑部件来实现二进制数据的各种运算。计

算机内部采用二进制至少具有如下优点。

(1) 易于实现。二进制中只有 0 和 1 两个状态，容易用物理器件实现，如门电路的导通与截止、电压的高与低，它们恰好对应表示“1”和“0”两个符号。

(2) 运算简单。二进制的加法和乘法的运算规则仅有 4 种，因而简化了运算器等物理器件的设计。

(3) 可靠性高。由于电压的高低、电流的有无等都是一种质的变化，两种状态区别明显，所以，二进制数据的传递抗干扰能力强。

(4) 通用性强。二进制不仅适用于数值信息的编码，而且各种非数值信息都可以方便地转换为二进制编码。

📖 二进制是德国数学家莱布尼茨在 18 世纪发明的，他的发明是受中国八卦的启迪。莱布尼茨曾写信给当时在康熙皇帝身边工作的法国传教士白晋，询问有关八卦的问题并仔细研究过八卦，莱布尼茨还把自己制造的一台手摇计算机托人送给康熙皇帝。

八卦由爻组成，爻分阴爻(用“--”表示)和阳爻(用“—”表示)，三个爻的八种排列组成了八卦：☰乾、☷坤、☳震、☴巽、☵坎、☲离、☶艮、☱兑，分别代表天、地、雷、风、水、火、山和泽 8 种自然现象。用 1 表示阳爻，用 0 表示阴爻，则八卦可以用二进制的 0 和 1 的组合来表示。

3.2.1 进位计数制

按进位(当某一位的值达到某个固定量时，就要向高位产生进位)的原则进行计数的方法称为**进位计数制**，简称**进制**。在日常生活中，人们使用最多的是十进制。此外，也使用许多非十进制的计数方法，例如，时间采用的是六十进制，即 60 秒为 1 分钟，60 分钟为 1 小时；月份采用的是十二进制，即 1 年有 12 个月，等等。

📖 中国人在计数时常常用“正”字代表 5，两个“正”字就是 10，以此类推。这个计数方法简便易懂，很受中国人欢迎。是谁最早使用这个方法的呢？清末民初，戏园(俗称茶园)是人们日常生活中重要的娱乐场所。那时候还没有门票，戏园就安排“案目”(就是现在的服务员)在戏园门口招徕看客，领满五位入座，司事(记账先生)便在大水牌(类似黑板)上写出一个“正”字，并标明某案目的名字。这个方法随着门票制而被废弃了，但是作为一种简明的计数方法，一直流行于民间。

不同的进制以基数来区分，若以 r 代表基数，则

$r=10$ 为十进制，可使用 0，1，2，…，9 共 10 个数码；

$r=2$ 为二进制，可使用 0，1 共 2 个数码；

$r=8$ 为八进制，可使用 0，1，2，…，7 共 8 个数码；

$r=16$ 为十六进制，可使用 0，1，2，…，9，A，B，C，D，E，F 共 16 个数码。

r 进制数通常写作 $(a_n\cdots a_1a_0.a_{-1}\cdots a_{-m})_r$，其中 $a_i\in\{0,1,\cdots,r-1\}(-m\leqslant i\leqslant n)$。例如，二进制数 1101 写作 $(1101)_2$，十进制数 689.12 写作 $(689.12)_{10}$。

进位计数制采用位置记数法(数码按顺序排列)表示数,位置记数法有两个要点:**按基数进位或借位,用位权值计数。**

所谓按基数进位和借位,就是在执行算术运算时,遵守"逢 r 进 1,借 1 当 r"的规则。如十进制的规则为"逢 10 进 1,借 1 当 10",二进制的规则为"逢 2 进 1,借 1 当 2"。二进制的算术运算规则非常简单,如表 3.2 所示。

表 3.2　二进制的算术运算规则

加	减	乘	除
0+0=0	0−0=0	0×0=0	0÷0(没有意义)
0+1=1	0−1=1(向高位借 1)	0×1=0	0÷1=0
1+0=1	1−0=1	1×0=0	1÷0(没有意义)
1+1=0(向高位进 1)	1−1=0	1×1=1	1÷1=1

例 3.3　计算 1010+10 和 1010−100 的值。

解:

$$\begin{array}{r} 1010 \\ +\quad 10 \\ \hline 1100 \end{array} \qquad \begin{array}{r} 1010 \\ -\quad 100 \\ \hline 110 \end{array}$$

则:1010+10=1100,1010−100=110

例 3.4　计算 1010×101 和 10101÷100 的值。

解:

$$\begin{array}{r} 1010 \\ \times\quad 101 \\ \hline 1010 \\ 0000 \\ 1010 \\ \hline 110010 \end{array} \qquad \begin{array}{r|l} & 101 \\ \hline 100 & 10101 \\ & 100 \\ \hline & 101 \\ & 100 \\ \hline & 1 \end{array}$$

则:1010×101=110010,10101÷100=101 余 1

在任何一种进制中,处于不同位置上的数码代表不同的值,例如,在十进制中,数码 8 在个位上表示 8,在十位上表示 80,在百位上表示 800,而在小数点后 1 位表示 0.8,所以,每个位置都对应一个位权值。对于 r 进制数 $(a_n\cdots a_1a_0.a_{-1}\cdots a_{-m})_r$,小数点左面的位权值依次为 r^0, r^1, …, r^n,小数点右面的位权值依次为 r^{-1}, …, r^{-m}。每个位置上的数码所表示的数值等于该数码乘以该位置的位权值。例如,十进制数 198.63 可以表示成:

$$198.63=1\times10^2+9\times10^1+8\times10^0+6\times10^{-1}+3\times10^{-2}$$

二进制数 1101.11 可以表示成:

$$1101.11=1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}$$

3.2.2　二进制数和十进制数之间的转换

由于人们习惯使用十进制,而计算机内部使用的是二进制,所以,计算机系统需要进行十进制数和二进制数之间的转换。

将十进制数转换为二进制数需要将十进制数分解为整数部分和小数部分，分别进行转换，然后相加得到转换的最终结果。

将十进制整数转换为二进制整数的规则是：**除基取余，逆序排列**，即将十进制整数逐次除以二进制的基数 2，直到商为 0，然后将得到的余数逆序排列，先得到的余数为低位，后得到的余数为高位。

例 3.5 将十进制整数 46 转换为二进制整数。

解：

除数	商	余数	
46	23	0	↑
23	11	1	逆序排列
11	5	1	
5	2	1	
2	1	0	
1	0	1	

则：$(46)_{10}=(101110)_2$

📖 **十进制整数转换为 r 进制整数的数学原理** 已知十进制整数 x，设 x 对应的 r 进制整数 $y=(a_n\cdots a_1a_0)r$，则：

$$y=(a_n\cdots a_1a_0)_r=a_nr^n+\cdots a_1r^1+a_0r^0=((\cdots(a_nr+a_{n-1})r+\cdots+a_2)r+a_1)r+a_0$$

将 y 除以 r，商为 $(\cdots(a_nr+a_{n-1})r+\cdots+a_2)r+a_1$，余数为 a_0；

所得商再除以 r，商为 $(\cdots(a_nr+a_{n-1})r+\cdots)r+a_2$，余数为 a_1；

依此类推，直至商为 0，余数为 a_n。

将十进制小数转换为二进制小数的规则是：**乘基取整，正序排列**，即将十进制小数逐次乘以二进制的基数 2，直到积的小数部分为 0，然后将得到的整数正序排列，先得到的整数为高位，后得到的整数为低位。

例 3.6 将十进制小数 0.375 转换为二进制小数。

解：

乘数	积	整数	
0.375	0.75	0	↓
0.75	1.5	1	正序排列
0.5	1.0	1	

则：$(0.325)_{10}=(0.011)_2$

并不是所有的十进制小数都可以精确地转换为二进制小数，如果乘基取整后的小数部分始终不为 0，则可以根据精度要求转换到一定的位数为止。由于十进制小数转换为二进制小数有可能存在精度上的误差，所以，在程序设计中要尽量避免处理小数。如果必须处理小数，有时可以先将小数变为整数，运算后再将整数变为小数。

例 3.7 将十进制小数 0.325 转换为二进制小数。

解：

乘数	积	整数	
0.325	0.65	0	↓ 正序排列
0.65	1.3	1	
0.3	0.6	0	
0.6	1.2	1	
0.2	0.4	0	
0.4	0.8	0	
0.8	1.6	1	
0.6	1.2	1	

此后处于无限循环状态，假设精度为小数点后 8 位，则：$(0.325)_{10}=(0.01010011)_2$

📖 1991 年 2 月 25 日，海湾战争期间，一枚飞毛腿导弹击中了美国陆军的军营，造成 28 名士兵死亡、100 多人受伤。由于软件错误，位于沙特阿拉伯 Dhahran 的美国爱国者导弹发射器没有成功跟踪并阻截伊拉克的飞毛腿导弹。错误发生在软件中有一个运算涉及 0.1 的乘法，这个数不能精确地转换为二进制数。在 100 个小时的跟踪操作中，这个运算的积累误差达到 0.34 秒，足够使拦截导弹偏离它的目标。

将二进制数转换为十进制数只需将二进制数按位权值展开然后求和，所得结果即为对应的十进制数。

例 3.8 将二进制数 1101.11 转换为十进制数。

解： $1101.11=1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}=13.75$

则：$(1101.11)_2=(13.75)_{10}$

一般地，十进制数与二进制数之间的转换方法可以推广到任意进制。

3.2.3 二进制数与八进制数和十六进制数之间的转换

二进制数的位数太长，不便于书写和表示，而八进制和十六进制与二进制具有较为直观的对应关系（$2^3=8$，$2^4=16$），因而在计算机程序中通常采用八进制数或十六进制数。表 3.3 给出了二进制与八进制和十六进制之间的对应关系。

表 3.3 二进制与八进制和十六进制的对应关系

二进制	八进制	十六进制	二进制	八进制	十六进制
000	0	0	1000	10	8
001	1	1	1001	11	9
010	2	2	1010	12	A
011	3	3	1011	13	B
100	4	4	1100	14	C
101	5	5	1101	15	D
110	6	6	1110	16	E
111	7	7	1111	17	F

由于3位二进制数恰好是1位八进制数，所以，将二进制数转换为八进制数只需以小数点为界，将整数部分自右向左、小数部分自左向右每3位一组(不足3位用0补足)转换为对应的1位八进制数，反之，将八进制数转换为二进制数只需把每1位八进制数转换为对应的3位二进制数。

例 3.9 将八进制数345.67转换为二进制数。

$$(345.67)_8=(\underline{011}\ \underline{100}\ \underline{101}.\underline{110}\ \underline{111})_2=(11100101.110111)_2$$

例 3.10 将二进制数11100101.110111转换为八进制数。

$$(11100101.110111)_2=(\underline{011}\ \underline{100}\ \underline{101}.\underline{110}\ \underline{111})_2=(345.67)_8$$

同理，由于4位二进制数恰好是1位十六进制数，所以，将二进制数转换为十六进制数只需以小数点为界，将整数部分自右向左、小数部分自左向右每4位一组(不足4位用0补足)转换为对应的1位十六进制数，反之，将十六进制数转换为二进制数只需把每1位十六进制数转换为对应的4位二进制数。

例 3.11 将十六进制数6AC.57转换为二进制数。

$$(6AC.57)_{16}=(\underline{0110}\ \underline{1010}\ \underline{1100}.\underline{0101}\ \underline{0111})_2=(11010101100.01010111)_2$$

例 3.12 将二进制数11010101100.01010111转换为十六进制数。

$$(11010101100.01010111)_2=(\underline{0110}\ \underline{1010}\ \underline{1100}.\underline{0101}\ \underline{0111})_2=(6AC.57)_{16}$$

3.3 数值数据的表示方法

在计算机中表示一个数值数据，需要考虑三个问题：数的长度、符号的表示方法和小数点的表示方法。

3.3.1 数的长度

在数学中，数的长度是指该数所占的实际位数，例如在十进制中，整数12345的长度为5。在计算机中，数的长度是指该数所占的二进制位数，由于存储单元通常以字节为单位，因此，数的长度也指该数所占的字节数。

在数学中，数的长度不是固定的，例如135的长度是3，5的长度是1，实际应用时有几位就写几位，但在计算机中，同类型的数据长度一般是固定的，由机器的字长确定，不足部分用0补足。换言之，计算机中同一类型的数据具有相同长度，与数据的实际长度无关。例如，某16位计算机，其整数占两个字节(即16位二进制)，所有整数的长度都是16位，则$(68)_{10}=(1000100)_2=(00000000\ 01000100)_2$。

当一个数的二进制位数确定后，其表示范围也就确定了，如果一个数超出了这个范围，这种现象称为**溢出**。例如，16位二进制数表示的正整数范围是0～65536，超出表示范围的上界称为上溢，上溢时计算机将不能进行运算。超出表示范围的下界称为下溢，下溢时计算机将该数作为机器零来处理。

在以下讨论中，为简单起见，不失一般性，假设用八位二进制表示一个整数。

3.3.2 数的原码、反码和补码

数值数据有正数和负数之分，由于计算机中使用二进制0和1，因此，可以采用一位二