第1章　数字设备中数和字符的表示方法

本章要点:

- 掌握微型计算机的特点和用途;
- 掌握数字设备中数的表示方法和各种进制间数的转换方法;
- 掌握原、补、反码的概念;
- 初步掌握数字设备中常用的编码。

在当今的工作、学习和生活中，随处可见数字设备的存在。由于计算机，特别是微型计算机已经融入人们中间，往往会使初学者认为数字设备就是计算机。鉴于此，在学习本课程前再重申两点：

- 计算机是数字设备的一种，它有数字设备的基本属性，但又有其鲜明的特点;
- 在数字设备中，0 和 1 是基本元素，与、或、非是基本运算，组合（逻辑）电路和时序（逻辑）电路是基本表现形式。

1.1　微型计算机概述

微型计算机是 20 世纪 70 年代初期发展起来的。它的产生、发展和壮大及对国民经济的巨大作用引起人们的高度重视，它使人类社会发生翻天覆地的变化。

随着微型计算机的高速发展，单片微型计算机、单板微型计算机、微型计算机系统、微型计算机开发系统和计算机网络工作站等新机种不断涌现。为了掌握好微型计算机，从概念上弄清微型计算机和这些新机种之间的异同是十分重要的，读者应予重视。

1.1.1　微型计算机的特点和发展

电子计算机通常按体积、性能和价格分为巨型机、大型机、中型机、小型机和微型计算机 5 类。从系统结构和基本工作原理上说，微型计算机和其他几类计算机并没有本质的区别，所不同的是微型计算机广泛采用集成度相当高的器件和部件，因此带来以下所述的一系列特点。

1．体积小，质量轻

由于采用大规模集成电路（LSI）和超大规模集成电路（VLSI），使微型计算机所含的器件数目大为减少，体积大为缩小。对于 20 世纪 50 年代要由占地上百平方米，耗电上百千瓦的电子计算机实现的功能，当今，内部只含几十片集成电路的微型计算机即已具备。

2．价格低廉

如今的微型计算机价格不断下降，而性能却迅速提高。在我国，许多家庭已经或正在筹划购置微型计算机。微型计算机进入家庭的时代已经到来。

3．可靠性高，结构灵活

由于内部元器件数目少，所以连线比较少，这样，使微型计算机的可靠性较高，结构灵活方便。

4．应用面广

现在，微型计算机不仅占领了原来使用小型机的各个领域，而且广泛应用于过程控制等新的场合。此外，微型计算机还进入了过去电子计算机无法进入的领域，如测量仪器、仪表、教学部门、医疗设备及家用电器等。

由于微型计算机具有上面这些特点，所以它的发展速度大大超过了前几代计算机。自从 20 世纪 70 年代初第一个微处理器诞生以来，微处理器的性能和集成度几乎每两年提高一倍，而价格却降低一半。

第一个微处理器是 1971 年美国 Intel 公司生产的 Intel 4004。

以后，出现了许多生产微处理器的厂家。1973—1977 年，这些厂家生产了多种型号的微处理器，其中设计最成功、应用最广泛的是 Intel 公司的 8080/8085，Zilog 公司的 Z80，Motorola 公司的 6800/6802 及 Rockwell 公司的 6502。通常，人们把它们称为第二代微处理器。在这个时期，微处理器的设计和生产技术已相当成熟，配套的各类器件也很齐全。后来，微处理器在以下几个方面得到很大发展：提高集成度，提高功能和速度，以及增加外围电路的功能和种类。

1978—1979 年，一些厂家推出了性能可与过去中档小型计算机相比的 16 位微处理器。这中间，有代表性的 3 种芯片是 Intel 的 8086/8088，Zilog 的 Z8000，以及 Motoro1a 的 MC 68000。人们将这些微处理器称为第一代超大规模集成电路的微处理器。

1980 年以后，半导体生产厂家继续在提高电路的集成度、速度和功能方面取得了很大进展，相继出现 Intel 80286、Motoro1a 68010 等 16 位高性能微处理器。1983 年以后，又生产出 Intel 80386 和 Motorola 68020。这两者都是 32 位的微处理器。

1993 年，Intel 推出 Pentium 微处理器，接着是 Pentium MMX（带多媒体指令的 Pentium 微处理器）、Pentium Pro、Pentium II 和 Pentium III 相继问世。Pentium III 的主频可达 450 MHz 以上，增加了浮点运算、并行处理、图像处理和接入互联网的功能。最新推出的 Pentium IV 微处理器的主频已达 1.7 GHz 以上。

1.1.2 微型计算机的分类

人们可以从不同的角度对微型计算机进行分类。有人按机器组成来分，将微型计算机分为位片式、单片式和多片式；有人按制造工艺来分，将微型计算机分为 MOS 型和双极型。由于微型计算机性能的高低在很大程度上取决于核心部件微处理器，所以，最通常的做法是把微处理器的字长作为微型计算机的分类标准。

当前，可以见到由以下几类微处理器构成的微型计算机。

1．4 位微处理器

最初的 4 位微处理器就是 Intel 4004，后来改进为 4040。目前常见的是 4 位单片微型计算机，即在一个芯片内集中了 4 位的 CPU、 RAM、 ROM、 I/O 接口和时钟发生器。这种单片机价格低廉，但运算能力弱，存储容量小，存储器中只存放固定程序。这些特点使它广泛用于各类袖珍计算器中进行简单的运算，或者用于家用电器和娱乐器件中进行简单的过程控制。

2．8 位微处理器

8 位微处理器推出时，微型计算机技术已经比较成熟。因此，在 8 位微处理器基础上构成的微型计算机系统，通用性较强，它们的寻址能力可以达到 64 KB，有功能灵活的指令系统和较强的中断能力。另外，8 位微处理器有比较齐备的配套电路。这些因素使 8 位微型计算机的应用范围很宽，可广泛用于事务管理、工业控制、教育及通信等行业。8 位微处理器也常用来构成智能终端，而 8 位微型计算机则被许多家庭用做学习机和个人计算机。

常见的 8 位微处理器有 Zilog 的 Z80，Intel 的 8080/8085，Motorola 6800/6802 和 Rockwell 6502。由这些微处理器构成的微型计算机在国内占有相当数量。

3．16 位微处理器

16 位微处理器不仅在集成度、处理速度和数据总线宽度等方面优于前几类微处理器，而且在功能和处理方法上也做了改进。在此基础上构成的微型计算机系统，在性能方面已经和 20 世纪 70 年代的中档小型计算机相当。

16 位微处理器中最有代表性的是 Intel 8086/8088 和 Motorola 68000。以 Intel 8086/8088 为 CPU 的 16 位微型计算机 IBM PC/XT 是目前的主流机型，它拥有的用户在计算机世界首屈一指，以至于在设计更高档的微型计算机时，都尽量保持与它兼容。

4．32 位微处理器

32 位微处理器是当前性能最优的微处理器，典型产品为 Intel 80386，Motorola MC68020。它们的主频率高达 20～40 MHz，平均指令执行时间为 0.05 μs。近两年，32 位微型计算机已投入使用，但由于价格比较高，因此，一般作为工作站进行计算机辅助设计、工程设计，或者作为微型计算机局域网中的资源站点。

除此以外，还有一类叫位片式处理器，利用这类微处理器，可以构成不同字长的微型计算机。位片机结构灵活，主要用于对中、小型计算机的仿真，或者用于高速实时控制专用系统中，也常用在分布式系统和高速智能外设中。

1.1.3　微型计算机的应用

1．科学计算

科学计算一直是电子计算机的重要应用领域之一。例如，在天文学、量子化学、空气动力学及核物理学等领域中，都需要依靠计算机进行复杂的运算。在军事上，导弹的发射和飞行轨道的计算控制，以及先进防空系统等现代化军事设施，通常都是由计算机控制的大系统，其中包括雷达、地面设施和海上装备等。现代的航空航天技术的发展，例如超音速飞行器的设计和人造卫星与运载火箭轨道的计算等，更离不开计算机。

除了国防及尖端科学技术以外，计算机在其他学科和工程设计方面，诸如数学、力学、晶体结构分析、石油勘探、桥梁设计、建筑及土木工程设计等领域，也得到了广泛的应用，促进了各门科学技术的发展。

有些系统，要求计算机处理所得的结果立即反过来作用或影响正在被处理的事物本身。例如，在控制导弹飞行的系统中，通过不断测量导弹飞行的参数（包括飞行环境），并及时做出反应，以不断地修正导弹飞行的姿态与轨道，这样的系统称为实时处理系统。

科学计算的特点是计算量大和数值变化范围大。

2．信息处理和事务管理

在短时间内完成对大量信息的处理是进入信息时代的必然要求。微型计算机配上数据库管理软件以后，可以很灵活地对各种信息按不同的要求进行分类、检索、转换、存储和打印，加上一些专用部件（如传感器）以后，还可以处理光、热、力和声音等物理信号。

3．过程控制

过程控制是微型计算机应用最多，也是最有效的方面之一。现在，制造工业和日用品生产厂家中都可见到微型计算机控制的自动化生产线。微型计算机在这些部门的应用为生产能力和产品质量的迅速提高开辟了广阔的前景。

4．仪器、仪表控制

在许多仪器、仪表中，已经用微处理器代替传统的机械部件或分立的电子部件，这使产品降低了价格，而可靠性和功能却得到了提高。此外，微处理器的应用还导致了一些原来没有的新仪器的诞生。在实验室里，出现了用微处理器控制的示波器——逻辑分析仪，它使电子工程技术人员能够用以前不可能采用的办法同时观察许多信号的波形和相互之间的时序关系。在医学领域中，出现了用微处理器作为核心控制部件的 CT 扫描仪和超声波扫描仪，增加了疾病的诊断手段。

5．家用电器和民用产品控制

由微处理器控制的洗衣机、电冰箱，现在已经是很普通的民用电器了。此外，微处理器控制的自动报时、自动调温及自动报警系统也已经进入发达国家的家庭。还有，装有微处理器的娱乐产品往往将智能功能融于娱乐中；以微处理器为核心的盲人阅读器则能自动扫描文本，并读出文本的内容，从而为盲人带来福音。确切地讲，微处理器在人们日常生活中的应用所受到的主要限制不是技术问题，而是创造力和技巧上的问题。

当前，微型计算机技术正往两个方向发展：一个是高性能、高价格的方向，从这方面不断取得的成就可能使微型计算机代替价格昂贵、功能优越的巨型机；另一个是价格低廉、功能专一的方向，这方面的发展使微型计算机在生产领域、服务部门和日常生活中得到越来越广泛的应用。

6．计算机辅助设计/计算机辅助制造（CAD/CAM）

由于计算机有快速的数值计算、较强的数据处理及模拟的能力，因而目前在飞机、船舶、光学仪器及超大规模集成电路（VLSI）等的设计制造过程中，CAD/CAM 占据着越来越重要的地位。

在超大规模集成电路的设计和生产过程中，要经过设计制图、照相制版、光刻、扩散和内部连接等多道复杂工序，这是人工难以解决的。

使用已有的计算机，辅助设计新的计算机，达到设计自动化或半自动化程度，从而减轻人的劳动强度并提高设计质量，这也是计算机辅助设计的一项重要内容。

由于设计工作与图形制作密不可分，一般供辅助设计用的计算机都配备图形显示、绘图仪等设备，以及图形库、图形制作软件等。设计人员可借助这些专用软件和输入/输出设备把设计要求或方案输入计算机，并通过相应的应用程序进行计算处理后把结果显示出来。设计人员可用光笔或鼠标器进行修改或选择，直到满意为止。

7．人工智能

人类的许多脑力劳动，诸如证明数学定理，进行常识性推理，理解自然语言，诊断疾病，下棋游戏及破译密码等都需要“智能”。

人工智能学科研究的主要内容包括：知识表示，自动推理和搜索方法，机器学习和知识获取，知识处理系统，自然语言理解，计算机视觉及智能机器人等。

1.2　数和数制

迄今为止，所有计算机都是以二进制形式进行算术运算和逻辑操作的，微型计算机和其他数字设备也不例外。因此，用户在键盘上输入的十进制数字和符号命令，微型计算机都必须先把它们转换成二进制形式进行识别、运算和处理，然后再把运算结果还原成十进制数字和符号在输出设备（如 CRT）上显示出来。

虽然上述过程十分烦琐，但都是由微型计算机自动完成的。为了使读者最终弄清机器的这一工作机理，这里先对微型计算机中常用的数制和数制间数的转换进行讨论。

1.2.1　各种数制及其表示法

所谓数制指的是数的制式，是人们利用符号计数的一种科学方法。数制是人类在长期的生存斗争和社会实践中逐步形成的。数制有很多种，微型计算机中常用的数制有十进制、二进制和十六进制 3 种。

1．十进制（Decimal）

十进制是大家很熟悉的进位计数制，它共有 0,1,2,3,4,5,6,7,8 和 9 这 10 个数字符号。这 10 个数字符号又称为数码，每个数码在数中最多可表示两种含义的值。例如，十进制数 45 中的数码 4，其本身的值为 4，但它实际代表的值为 40。在数学上，数制中数码的个数定义为基数，故十进制的基数为 10。

十进制是一种科学的计数方法，它所能表示的数的范围很大。十进制数通常具有如下两个主要特点。

- 它有 0～9 十个不同的数码，这是构成所有十进制数的基本符号。
- 它是逢 10 进位的。十进制数在计数过程中，当它的某位计满 10 时就要向它邻近的高位进 1。

因此，任何一个十进制数不仅和构成它的每个数码本身的值有关，而且还和这些数码在

数中的位置有关。这就是说，任何一个十进制数都可以展开为幂级数形式。例如：

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

式中，指数 $10^{2},10^{1},10^{0},10^{-1}$ 和 10^{-2} 在数学中称为权，10 为它的基数；整数部分中每位的幂是该位位数减 1；小数部分中每位的幂是该位小数的位数。

一般地说，任意一个十进制数 N 均可表示为：

$$\begin{aligned}N=\pm&(a_{n-1}\times10^{n-1}+a_{n-2}\times10^{n-2}+\cdots+a_0\times10^{0}+\\&a_{-1}\times10^{-1}+a_{-2}\times10^{-2}+\cdots+a_{-m}\times10^{-m})\\=\pm&\sum_{i=n-1}^{-m}a_i\times10^{i}\end{aligned}$$

式中，i 为数中任一位，是一个变量；a_i 为第 i 位的数码；n 为该十进制数整数部分的位数；m 为小数部分的位数。

2．二进制（Binary）

二进制比十进制更为简单，它是随着计算机的发展而兴旺起来的。二进制数也有如下两个主要特点。

- 它共有 0 和 1 两个数码，任何二进制数都是由这两个数码组成的。
- 二进制数的基数为 2，它奉行逢 2 进 1 的进位计数原则。

因此，二进制数同样也可以展开为幂级数形式，不过内容有所不同罢了。例如：

$$\begin{aligned}10110.11&=1\times2^{4}+0\times2^{3}+1\times2^{2}+1\times2^{1}+0\times2^{0}+1\times2^{-1}+1\times2^{-2}\\&=1\times2^{4}+1\times2^{2}+1\times2^{1}+1\times2^{-1}+1\times2^{-2}\\&=[22.75]_{10}\end{aligned}$$

式中，指数 $2^{4},2^{3},2^{2},2^{1},2^{0},2^{-1}$ 和 2^{-2} 为权，2 为基数，其余和十进制数相同。

为此，任何二进制数 N 的公式为：

$$\begin{aligned}N=\pm&(a_{n-1}\times2^{n-1}+a_{n-2}\times2^{n-2}+\cdots+a_0\times2^{0}+\\&a_{-1}\times2^{-1}+a_{-2}\times2^{-2}+\cdots+a_{-m}\times2^{-m})\\=\pm&\sum_{i=n-1}^{-m}a_i\times2^{i}\quad（a_i\text{为 0 或 1}）\end{aligned}$$

式中，a_i 为第 i 位数码，可取 0 或 1；n 为该二进制数整数部分的位数；m 为小数部分的位数。

3．十六进制（Hexadecimal）

十六进制是人们学习和研究计算机中二进制数的一种工具，它是随着计算机的发展而广泛应用的。十六进制数也有两个主要特点。

- 它有 0,1,2,…,9,A,B,C,D,E,F 等 16 个数码，任何一个十六进制数都是由其中的一些或全部数码构成的。
- 十六进制数的基数为 16，进位计数为逢 16 进 1。

十六进制数也可展开为幂级数形式。例如：

$$70F.B1=7\times16^{2}+0\times16^{1}+F\times16^{0}+B\times16^{-1}+1\times16^{-2}=[1807.6914]_{10}$$

其通式为：

$$\begin{aligned}N=\pm&(a_{n-1}\times16^{n-1}+a_{n-2}\times16^{n-2}+\cdots+a_0\times16^{0}+\\&a_{-1}\times16^{-1}+a_{-2}\times16^{-2}+\cdots+a_{-m}\times16^{-m})\end{aligned}$$

$$=\pm\sum_{i=n-1}^{-m}a_i\times 2^i \qquad (a_i \text{为 } 0\sim\text{F})$$

式中，a_i 为第 i 位数码，取值为 0～F 中的一个；n 为该十六进制数整数部分的位数；m 为小数部分的位数。

为方便起见，现将部分十进制、二进制和十六进制数的对照表列于表 1.1 中。

表 1.1　部分十进制、二进制和十六进制数的对照表

整　数			小　数		
十 进 制	二 进 制	十六进制	十 进 制	二 进 制	十六进制
0	0000	0	0	0	0
1	0001	1	0.5	0.1	0.8
2	0010	2	0.25	0.01	0.4
3	0011	3	0.125	0.001	0.2
4	0100	4	0.0625	0.0001	0.1
5	0101	5	0.03125	0.00001	0.08
6	0110	6	0.015625	0.000001	0.04
7	0111	7			
8	1000	8			
9	1001	9			
10	1010	A			
11	1011	B			
12	1100	C			
13	1101	D			
14	1110	E			
15	1111	F			
16	10000	10			

在阅读和书写不同数制的数时，如果不在每个数上外加一些辨认标记，就会混淆而无法分清。通常，标记方法有两种：一种是把数加上方括号，并在方括号右下角标注数制代号，如$[101]_{16}$、$[101]_2$ 和$[101]_{10}$ 分别表示十六进制、二进制和十进制数；另一种是用英文字母标记，加在被标记数的后面，分别用大写字母 B、D 和 H 表示二进制、十进制和十六进制数，如 89H 为 16 进制数，101B 为二进制数等。其中，十进制数中的 D 标记也可以省略。另外，在书写十六进制数时，若最高位是字母时，则必须在其前面加 0，以免与英文单词混淆。例如，F9H 应写成 0F9H。

在微型计算机内部，数的表示形式是二进制的。这是因为二进制数只有 0 和 1 两个数码，人们采用晶体管的导通和截止，脉冲的高电平和低电平等，就能很容易地表示它。此外，二进制数运算简单，便于用电子线路实现。

人们采用十六进制数可以大大减轻阅读和书写二进制数时的负担。例如：

11011011B＝0DBH

1001001111110010B＝93F2H

显然，采用十六进制数描述一个二进制数特别简短，尤其在被描述二进制数的位数较长时，更令计算机工作者感到方便。

1.2.2　各种数制间数的相互转换

微型计算机是采用二进制数操作的，但人们习惯于使用十进制数，这就要求机器能自动

对不同数制的数进行转换。这里暂且不讨论微型计算机是如何进行这种转换的，先来看看数学上是如何进行上述 3 种数制间数的转换的。3 种数制间数的转换方法如图 1.1 所示。

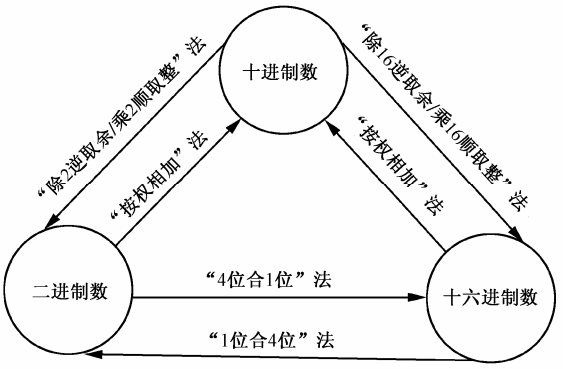

图 1.1　3 种数制间数的转换方法

1．二进制数和十进制数之间的转换

（1）二进制数转换成十进制数

只要把欲转换数按权展开后相加即可。也可以从小数点开始每 4 位 1 组，然后按 16 进制的权展开并相加。例如：

$$11010.01B=1\times2^4+1\times2^3+1\times2^1+1\times2^{-2}=26.25$$

或者　$11010.01B=1A.4H=1\times16^1+10\times16^0+4\times16^{-1}=26.25$

（2）十进制数转换成二进制数

本转换过程是上述转换过程的逆过程。但十进制整数和小数转换成二进制的整数和小数的方法是不相同的，下面分别进行介绍。

① 十进制整数转换成二进制整数的方法有很多种，但最常用的是“除 2 逆取余法”。

“除 2 逆取余法”的法则是：用 2 连续去除要转换的十进制数，直到商小于 2 为止，然后把各次余数按最后得到的为最高位，最先得到的为最低位依次排列起来，所得到的数便是所求的二进制数。现举例加以说明。

【例 1.1】　试求出十进制数 215 的二进制数。

解：把 215 连续除以 2，直到商数小于 2 为止。相应竖式为：

2	215	余1	最低位
2	107	余1	
2	53	余1	
2	26	余0	
2	13	余1	
2	6	余0	
2	3	余1	
	1	余1	最高位

把所得余数按箭头方向从高到低排列起来便可得到：

$$215=11010111B$$

② 十进制小数转换成二进制小数通常采用“乘 2 顺取整法”。

“乘 2 顺取整法”的法则是：用 2 连续去乘要转换的十进制小数，直到所得乘积的小数

部分为 0 或满足所需精度为止，然后把各次整数按最先得到的为最高位，最后得到的为最低位，依次排列起来，所对应的数便是所求的二进制小数。现结合实例加以介绍。

【例 1.2】　试把十进制小数 0.6879 转换为二进制小数。

解：把 0.6879 不断地乘以 2，取每次所得乘积的整数部分，直到乘积的小数部分满足所需精度为止。其相应的竖式是：

```
    0.6879
×       2
------------
    1.3758 ····· 取得整数1      最高位
    0.3758                        ↓
×       2                         ↓
------------                      ↓
    0.7516 ····· 取得整数0        ↓
×       2                         ↓
------------                      ↓
    1.5032 ····· 取得整数1        ↓
    0.5032                        ↓
×       2                         ↓
------------                      ↓
    1.0064 ····· 取得整数1      最低位
```

把所得整数按箭头方向从高到低排列后得到：

0.6879D≈0.1011B

对同时有整数和小数两部分的十进制数，在将其转换成二进制数时，可以把它的整数和小数部分分别转换后，再合并起来。例如，把例 1.1 和例 1.2 合并起来便可得到：

215.6879≈11010111.1011B

应当指出，任何十进制整数都可以精确地转换成一个二进制整数，但任何十进制小数却不一定可以精确地转换成一个二进制小数。例 1.2 中的情况就是一例。

2．十六进制数和十进制数之间的转换

（1）十六进制数转换成十进制数

方法和二进制数转换成十进制数的类似，即可把十六进制数按权展开后相加。例如：

$$3FEAH=3\times16^3+15\times16^2+14\times16^1+10\times16^0=16362$$

（2）十进制数转换成十六进制数

① 十进制整数转换成十六进制整数的方法和十进制整数转换成二进制整数的方法类似。十进制整数转换成十六进制整数可以采用“除 16 逆取余法”。

“除 16 逆取余法”的法则是：用 16 连续去除要转换的十进制整数，直到商数小于 16 为止，然后把各次余数按逆次序排列起来，所得的数便是所求的十六进制数。

【例 1.3】　求 3901 所对应的十六进制数。

解：把 3901 连续除以 16，直到商数为小于 16 为止。相应竖式为：

```
16 | 3901 ····· 余13  写做D      最低位
16 |  243 ····· 余3   写做3        ↑
   |   15 ····· 余15  写做F      最高位
```

∴　3901＝0F3DH

② 十进制小数转换成十六进制小数的方法类似于十进制小数转换成二进制小数，常采用“乘 16 顺取整法”。

“乘 16 顺取整法”的法则是：把欲转换的十进制小数连续乘以 16，直到所得乘积的小数部分为 0 或达到所需精度为止，然后把各次乘积的整数按相同次序排列起来，所得的数便是所求的十六进制小数。

【例 1.4】 求 0.76171875 的十六进制数。

解：把 0.76171875 连续乘以 16，直到所得乘积的小数部分为 0。相应竖式为：

```
        0.76171875
×               16
-------------------
       12.18750000 ·····  取整数12  写做C   最高位
        0.18750000                          ↓
×               16
-------------------
        3.00000000 ·····  取整数3   写做3   最低位
```

∴ 0.76171875＝0.C3H

3．二进制数和十六进制数之间的转换

二进制数和十六进制数之间的转换十分方便，这就是为什么人们要采用十六进制数形式来表达二进制数的内在原因。

① 二进制数转换成十六进制数：可采用“4 位合 1 位法”。

“4 位合 1 位法”的法则是：从二进制数的小数点开始，分别向左向右每 4 位 1 组，不足 4 位的以 0 补足，然后分别把每组数用十六进制数码表示，并按序相连。

【例 1.5】 把 1101111100011.10010100B 转换为十六进制数。

解：

```
0001  1011  1110  0011 . 1001  0100
 |     |     |     |     |     |
 1     B     E     3     9     4
```

∴ 1101111100011.10010100B＝1BE3.94H

② 十六进制数转换成二进制数：可采用“1 位分 4 位法”。

“1 位分 4 位法”的法则是：把十六进制数的每位分别用 4 位二进制数码表示，然后把它们连成一体。

【例 1.6】 把十六进制数 3AB.7A5H 转换为二进制数。

解：

```
 3     A     B  .  7     A     5
 |     |     |     |     |     |
0011  1010  1011 . 0111  1010  0101
```

∴ 3AB.7A5H＝1110101011.011110 100101B

1.2.3 二进制数的运算

在微型计算机中，经常碰到的运算分为两类：一类是算术运算，另一类是逻辑运算。在

算术运算中，可以对无符号数或有符号数进行加、减、乘、除运算；逻辑运算有逻辑乘、逻辑加、逻辑非和逻辑异或等。现分别加以介绍。

算术运算

（1）加法运算

二进制数加法法则为：

0＋0＝0

1＋0＝0＋1＝1

1＋1＝10　　（向邻近高位有进位）

1＋1＋1＝11　　（向邻近高位有进位）

两个二进制数的加法过程和十进制数的加法过程类似，现举例加以说明。

【例 1.7】　设两个 8 位二进制数 X＝10110110B，Y＝11011001B，试求出 $X+Y$ 的值。

解：$X+Y$ 可写成如下竖式：

被加数 X	10110110B
加数 Y	11011001B
和　$X+Y$	110001111B

∴ X ＋ Y＝10110110B ＋ 11011001B＝110001111B

两个二进制数相加时要注意低位的进位，且两个 8 位二进制数的和最大不会超过 9 位。

（2）减法运算

二进制数减法法则为：

0－0＝0

1－1＝0

1－0＝1

0－1＝1　　（向邻近高位借 1 当作 2）

两个二进制数的减法运算过程和十进制数的减法类似，现举例进行说明。

【例 1.8】　设两个 8 位二进制数 X＝10010111B，Y＝11011001B，试求 $X-Y$ 之值。

解：由于 $Y>X$，故有 $X-Y=-(Y-X)$，相应竖式为：

被减数 Y	11011001B
减数 X	10010111B
差数　$Y-X$	01000010B

∴　$X-Y$＝10010111B－11011001B＝－01000010B

两个二进制数相减时先要判断它们的大小，把大数作为被减数，小数作为减数，差的符号由两数关系决定。此外，在减法运算过程中还要注意低位向高位借 1 应看作 2。

（3）乘法运算

二进制数乘法法则为：

0×0＝0

1×0＝0×1＝0

1×1＝1

两个二进制数相乘与两个十进制数相乘类似。可以用乘数的每一位分别去乘被乘数，所得结果的最低位与相应乘数位对齐，最后把所有结果加起来，便得到积，这些中间结果又称为部分积。

【例 1.9】 设两个 4 位二进制数 X=1101B 和 Y=1011B，试用手工算法求出 $X\times Y$ 之值。

解：二进制数乘法运算竖式为：

被乘数 X　　1 1 0 1 B
乘数 Y ×　　1 0 1 1 B
1 1 0 1
1 1 0 1
0 0 0 0
1 1 0 1
乘积　1 0 0 0 1 1 1 1

∴ $X\times Y$=1101B×1011B=10001111B

（4）除法运算

除法是乘法的逆运算。与十进制数类似，二进制数除法也是从被除数最高位开始的。其过程是先查找出够减除数的位数，在其最高位处上商 1 并完成它对除数的减法运算，然后把被除数的下一位移到余数的位置上。若余数不够减除数，则上商 0，并把被除数的再下一位移到余数的位置上；若余数够减除数，则上商 1，余数减除数。这样反复进行，直到全部被除数的各位都下移到余数位置上为止。

【例 1.10】 设 X=10101011B，Y=110B，试求 $X\div Y$ 之值。

解：$X\div Y$ 的竖式是：

```
                1  1  1  0  0
          ┌──────────────────
110       │ 1  0  1  0  1  0  1  1
            1  1  0
          ──────────
            1  0  0  1
               1  1  0
          ──────────────
                  1  1  0
                  1  1  0
          ──────────────────
                        1  1
```

∴ $X\div Y$=10101011B÷110B=11100B …余 11B

1.2.4 逻辑运算

计算机处理数据时常常要用到逻辑运算，逻辑运算是由专门的逻辑电路完成的。下面介绍几种常用的逻辑运算。

1. 逻辑乘运算

逻辑乘又称逻辑与，常用运算符“∧”表示。逻辑乘的运算规则为：

0∧0=0

1∧0=0∧1=0

1∧1=1

两个二进制数进行逻辑乘运算，其方法类似二进制数的算术运算。

【例 1.11】　已知 X=01100110B，Y=11110000B，试求 $X \wedge Y$ 之值。

解：$X \wedge Y$ 的运算竖式为：

$$\begin{array}{rr} & 01100110\text{B} \\ \wedge & 11110000\text{B} \\ \hline & 01100000\text{B} \end{array}$$

∴　$X \wedge Y$=01100110B∧11110000B=01100000B

2．逻辑加运算

逻辑加又称逻辑或，常用运算符“∨”表示。逻辑加的运算规则为：

0∨0=0

1∨0=0∨1=1

1∨1=1

【例 1.12】　已知 X=00110101B，Y=00001111B，试求 $X \vee Y$ 之值。

解：$X \vee Y$ 的运算竖式为：

$$\begin{array}{rr} & 00110101\text{B} \\ \vee & 00001111\text{B} \\ \hline & 00111111\text{B} \end{array}$$

∴　$X \vee Y$=00110101B∨00001111B=00111111B

3．逻辑非运算

逻辑非运算又称逻辑取反，常采用运算符“ $\bar{}$ ”表示。逻辑非的运算规则为：

$\overline{0}=1$

$\overline{1}=0$

【例 1.13】　已知 X=11000011B，试求 $\overline{X}$ 之值。

∵　X =11000011B

∴　$\overline{X}$ =00111100B

4．逻辑异或运算

逻辑异或运算又称为半加，是不考虑进位的加法，常采用运算符“⊕”表示。逻辑异或的运算规则为：

0⊕0=1⊕1=0

1⊕0=0⊕1=1

【例 1.14】　已知 X=10110110B，Y=11110000B，试求 $X \oplus Y$ 的值。

解：$X \oplus Y$ 的运算竖式为：

$$\begin{array}{rr} & 10110110\text{B} \\ \oplus & 11110000\text{B} \\ \hline & 01000110\text{B} \end{array}$$

∴　$X \oplus Y$=10110110B⊕11110000B=01000110B

1.3 有符号二进制数的表示方法及溢出问题

1.3.1 有符号二进制数的表示方法

计算机内的数分为无符号数和有符号数。无符号数可以理解为正整数，有符号数可表示正数和负数。计算机中，为便于识别，需将有符号数的正、负号数字化。通常的做法是用一位二进制数表示符号，称为“符号位”，放在有效数字的前面，用“0”表示正，用“1”表示负。有符号的二进制数在计算机中有 3 种表示形式：原码、反码和补码。下面的讨论中假定字长为 8 位。

1．原码、反码和补码的表示方法

（1）原码

在数值的前面直接加一符号位的表示法称为原码表示法。例如，数＋7 和－7 的原码分别为：

符号位　数值位

$[+7]_{原}$＝　0　　0000111

$[-7]_{原}$＝　1　　0000111

在这种表示法中，数 0 的原码有两种形式，即：

$[+0]_{原}$＝00000000　　$[-0]_{原}$＝10000000

若字长为 8 位，则原码的表示范围为－127～＋127；若字长为 16 位，则原码的表示范围为－32767～＋32767。

（2）反码

正数的反码与原码相同；负数的反码，符号位仍为“1”，数值部分“按位取反”。例如，＋7 和－7 的反码分别为：

$[+7]_{反}$＝00000111B＝07H

$[-7]_{反}$＝11111000B＝F8H

在这种表示法中，数 0 的反码也有两种形式，即：

$[+0]_{反}$＝00000000＝00H　　$[-0]_{反}$＝11111111＝FFH

字长为 8 位和 16 位时，反码的表示范围分别为－127～＋127 和－32767～＋32767。

（3）补码

① 模的概念：把一个计量单位称之为模或模数，用 M 表示。例如，时钟以 12 为计数循环，即以 12 为模。在时钟上，时针加上（正拨）12 的整数倍或减去（反拨）12 的整数倍，时针的位置不变。14 点钟在舍去模 12 后，成为（下午）2 点钟。从 0 点出发逆时针拨 10 格即减去 10h，也可看成从 0 点出发顺时针拨 2 格（加上 2h），即 2 点。因此，在模 12 的前提下，－10 可映射为＋2。再如在讨论三角函数时以 360° 为计数循环，即 410° －360° ＝50° 。现实中还有许多以模为计数单位的例子。

计算机的运算部件与寄存器都有一定字长的限制，因此它的运算也是一种模运算。例如，一定位数的计数器，在计满后会产生溢出，又从头开始计数。产生溢出的量就是计数器的模。如 8 位二进制数，它的模数为 2^8＝256。

由此可以看出，对于一个模数为 12 的循环计数系统来说，加 2 和减 10 的效果是一样的。

因此，在以 12 为模的系统中，凡是减 10 的运算都可以用加 2 来代替，这就把减法问题转化成加法问题了。10 和 2 对模 12 而言互为补数，在计算机中称为“补码”。

② 补码的表示：在补码表示法中，正数的补码与原码相同；负数的补码则是符号位为“1”，数值部分按位取反后再在末位（最低位）加 1。

例如，＋7 和－7 的补码分别为：

$[+7]_{补}$＝00000111B＝07H

$[-7]_{补}$＝11111001B＝F9H

补码在微型计算机中是一种重要的编码形式，请注意如下事项。

注意

a. 采用补码后，可以方便地将减法运算转化为加法运算，运算过程得到简化。因此，计算机中有符号数一般采用补码表示。

b. 正数的补码即它所表示的数的真值，而负数补码的数值部分却不是它所表示的数的真值。

c. 采用补码进行运算，所得结果仍为补码。为了得到结果的真值，还得进行转换（还原）。转换前应先判断符号位，若符号位为 0，则所得结果为正数，其值与真值相同；若符号位为 1，则应将它转换成原码，然后得到它的真值。

d. 与原码、反码不同，数值 0 的补码只有一个，即 $[0]_{补}$ = 00000000B = 00H。

e. 若字长为 8 位，则补码所表示的范围为－128～＋127；若字长为 16 位，则补码所表示的范围为－32768～＋32767。

f. 进行补码运算时，应注意所得结果不应超过上述补码所能表示数的范围，否则会产生溢出而导致错误。采用其他码制运算时，同样应注意这一问题。

2．原码、反码和补码之间的转换

由于正数的原码、反码和补码表示方法相同，因此不存在转换问题。下面仅分析负数的原码、反码和补码之间的转换。

（1）已知原码，求补码

【例 1.15】　已知某数 X 的原码为 10110100B，试求 X 的补码。

解：由 $[X]_{原}$＝10110100B 知，X 为负数。求其补码表示时，符号位不变，数值部分按位求反，再在末位加 1。

```
  1  0  1  1  0  1  0  0     原码
  ↓  ↓  ↓  ↓  ↓  ↓  ↓  ↓
  1  1  0  0  1  0  1  1     符号位不变，数值位取反
                    1  +1
-------------------------
  1  1  0  0  1  1  0  0     补码
```

∴ $[X]_{补}$＝11001100B

（2）已知补码，求原码

【例 1.16】　已知某数 X 的补码为 11101110B，试求其原码。

解：由［X］$_{补}$=11101110B 知，X 为负数。求其原码表示时，符号位不变，数值部分按位求反后，再在末位加 1。

1	1	1	0	1	1	1	0	补码
↓	↓	↓	↓	↓	↓	↓	↓	
1	0	0	1	0	0	0	1	符号位不变，数值位取反
							1	+1
1	0	0	1	0	0	1	0	原码

∴［X］$_{原}$=10010010B

说明：按照求负数补码的逆过程，数值部分应是最低位减 1，然后取反。但是对二进制数来说，先减 1 后取反和先取反后加 1 得到的结果是一样的，故仍采用取反加 1 的方法。

（3）求补（已知［X］$_{补}$，求［$-X$］$_{补}$的过程称为求补）

所谓求补，就是将［X］$_{补}$的所有位（包括符号位）一起逐位取反，然后在末位加 1，即可得到$-X$的补码，即［$-X$］$_{补}$。不管 X 是正数还是负数，都应按该方法操作。

【例 1.17】 试求+97、−97 的补码。

解：+97=01100001，于是［+97］$_{补}$=01100001

求［−97］$_{补}$的方法是：

［+97］$_{补}$=	0	1	1	0	0	0	0	1	［+97］$_{补}$
	↓	↓	↓	↓	↓	↓	↓	↓	
	1	0	0	1	1	1	1	0	逐位取反
								1	+1
	1	0	0	1	1	1	1	1	［−97］$_{补}$

（4）已知补码，求对应的十进制数

【例 1.18】 已知某数 X 的补码为 10101011B，试求其所对应的十进制数。

解：该补码最高位（符号位）为 1，因此它表示的是负数。其数值部分（D_0～D_6）不等于真值，应予转换。转换时可采用以下两种方法。

方法 1：“求反加 1”法。

采用这种方法时，将补码的符号位和数值部分视为一个整体，按位取反，再在最低位上加 1，得到真实结果的二进制数的绝对值。在此结果前面加一负号即得正确答案。将上面的补码按位求反，并加 1，可得：01010100+1=01010101B=85，所求十进制数为−85。

方法 2：“零减补码”法。

该方法仍将补码的符号位和数值部分视为一个整体，用数零去减补码，做减法时不理会最高位产生的借位，所得结果即为该二进制数的绝对值。此例的计算过程如下。

	0	0	0	0	0	0	0	0	
−	1	0	1	0	1	0	1	1	
	0	1	0	1	0	1	0	1	55H=85D

所求十进制数为−85，与方法 1 所得结果相同。

表 1.2 列出了部分 8 位二进制代码分别代表无符号十进制数、原码、反码、补码时所表示的值。

表 1.2　8 位原码、反码和补码的对照表

二进制代码表示	无符号十进制数	原　码	反　码	补　码
0000 0000	0	+0	+0	+0
0000 0001	1	+1	+1	+1
0000 0010	2	+2	+2	+2
⋮	⋮	⋮	⋮	⋮
0111 1100	124	+124	+124	+124
0111 1101	125	+125	+125	+125
0111 1110	126	+126	+126	+126
0111 1111	127	+127	+127	+127
1000 0000	128	−0	−127	−128
1000 0001	129	−1	−126	−127
1000 0010	130	−2	−125	−126
⋮	⋮	⋮	⋮	⋮
1111 1100	252	−124	−3	−4
1111 1101	253	−125	−2	−3
1111 1110	254	−126	−1	−2
1111 1111	255	−127	−0	−1

1.3.2　有符号数运算时的溢出问题

在讨论溢出问题之前应明确：在计算机中所有信息均用 0 和 1 表示。具体地讲，有符号数和无符号数在表现形式上是无法分辨的，而是由程序设计者人为规定的，当然在处理上也就不同了。由于无符号数的各位均为数值，判断运算结果是否溢出，只要测试进位位即可。若为“1”则表示溢出，反之结果正确。

溢出是在一定字长下发生的。从理论上讲，溢出是不可能发生的，因为可以增加位数，实际很难做到，尤其是在单片机开发中更要引起注意。

如果计算机的字长为 n 位，n 位二进制数的最高位为符号位，其余 $n-1$ 位为数值位，则采用补码表示法时，可表示的数 X 的范围是：

$$-2^{n-1} \leqslant X \leqslant 2^{n-1}-1$$

当 $n=8$ 时，可表示的有符号数的范围为−128～+127；当 $n=16$ 时，可表示的有符号数的范围为−32768～+32767。两个有符号数进行加法运算时，如果运算结果超出可表示的有符号数的范围，就会发生溢出，使计算结果出错。很显然，溢出只能出现在两个同符号数相加或两个异符号数相减的情况下。

具体地讲，对于加运算，如果次高位（数值部分最高位）形成进位加入最高位，而最高位（符号位）相加（包括次高位的进位）却没有进位输出时；或者反过来，次高位没有进位加入最高位，但最高位却有进位输出时，都将发生溢出。因为这两种情况分别是：两个正数相加，结果超出了范围，形式上变成了负数；两负数相加，结果超出了范围，形式上变成了正数。

【例 1.19】

(+72)+(+98)

```
    0 1 0 0 1 0 0 0 B     +72
+   0 1 1 0 0 0 1 0 B     +98
-------------------------------
    1 0 1 0 1 0 1 0 B     −42
    ↑↑
    ├── 有进位
    └── 无进位            溢出，结果出错
```

【例 1.20】

(−83)+(−80)

```
  1 0 1 0 1 1 0 1 B     −83
+ 1 0 1 1 0 0 0 0 B     −80
-----------------------------
  0 1 0 1 1 1 0 1 B     +93
```

无进位

有进位　　溢出，结果出错

对于减法运算，当次高位不需从最高位借位，但最高位却需借位（正数减负数，差超出范围），或者反过来，次高位需从最高位借位，但最高位不需借位（负数减正数，差超出范围），也会出现溢出。

【例 1.21】

(+72)−(−98)

```
  0 1 0 0 1 0 0 0 B     +72
− 1 0 0 1 1 1 1 0 B     −98
-----------------------------
  1 0 1 0 1 0 1 0 B     −86
```

无借位

有借位　　溢出，结果出错

【例 1.22】

(−83)−(+80)

```
  1 0 1 0 1 1 0 1 B     −83
− 0 1 0 1 0 0 0 0 B     +80
-----------------------------
  0 1 0 1 1 1 0 1 B     +93
```

有借位

无借位　　溢出，结果出错

在后续课程中将说明，当在加减运算过程中出现结果超出有符号数所能表示的数值范围时，溢出标志位 OF 被置 1。

1.4 定点数和浮点数

在计算机中，用二进制数表示实数的方法有两种，即定点法和浮点法。

1.4.1 定点法

所谓定点数是指小数点的位置固定不变，以定点表示的数。

通常，定点表示有以下两种方法。

方法 1：规定小数点固定在最高数值位之前，机器中能表示的所有数都是小数。n 位数值部分所能表示的数 N 的范围（原码表示，下同）是：

$$-1<N<1$$

它能表示的数的绝对值为$|x|<1$。

方法 2：规定小数点固定在最低数值位之后，机器中能表示的所有数都是整数。n 位数值部分所能表示的数 N 的范围是：