2022年春季学期

EE202-17 - 数字电路

理论课教师：王恺

助教：王珊珊

教材：《数字电子技术基础（第五版）》（或第六版）

时间：周一7-8节，周三（单周）1-2节

教师介绍：https://faculty.sustech.edu.cn/wangk/

第一章 数码和码制

内容提要

1. 数字电路简介
2. 数制和码制的一些基本概念和术语
3. 数字电路中常用的数制和编码
4. 不同数制之间的转化方法
5. 二进制数算术运算的原理和方法

1.1 概述

数字量： 物理量的变化在时间上和数量上都是离散的，且其数值的大小和每次的增减变化为某一个最小数量的整数倍，小于这个最小数量单位则无意义。

模拟量： 物理量的变化在时间上或数值上是连续的。连续变化过程中任何取值都有物理意义。

数字信号/模拟信号：用以表示数字/模拟量的信号

数字电路/模拟电路：工作在数字/模拟信号下的电子电路

图1-1所示的为各种模拟信号

（a）正弦波　（b）矩形波（方波）　（c）尖脉冲　（d）锯齿冲

图1-1 几种模拟信号的波形

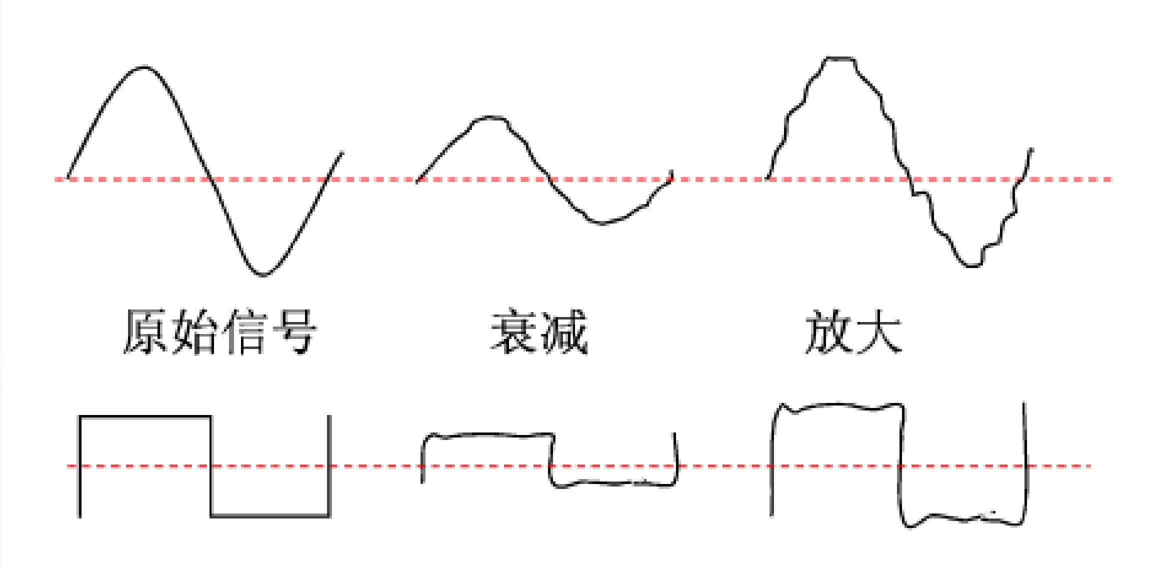
数字信号是表示数字量的信号，数字量在时间和数值上都是离散的。实现数字信号的产生、传输和处理的电路称为数字电路。数字信号包括脉冲型和电平型。如图1-2所示

1，0代表两种状态

(a)电平型数字信号　(b)脉冲型数字信号

图1-2 两种数字信号波形

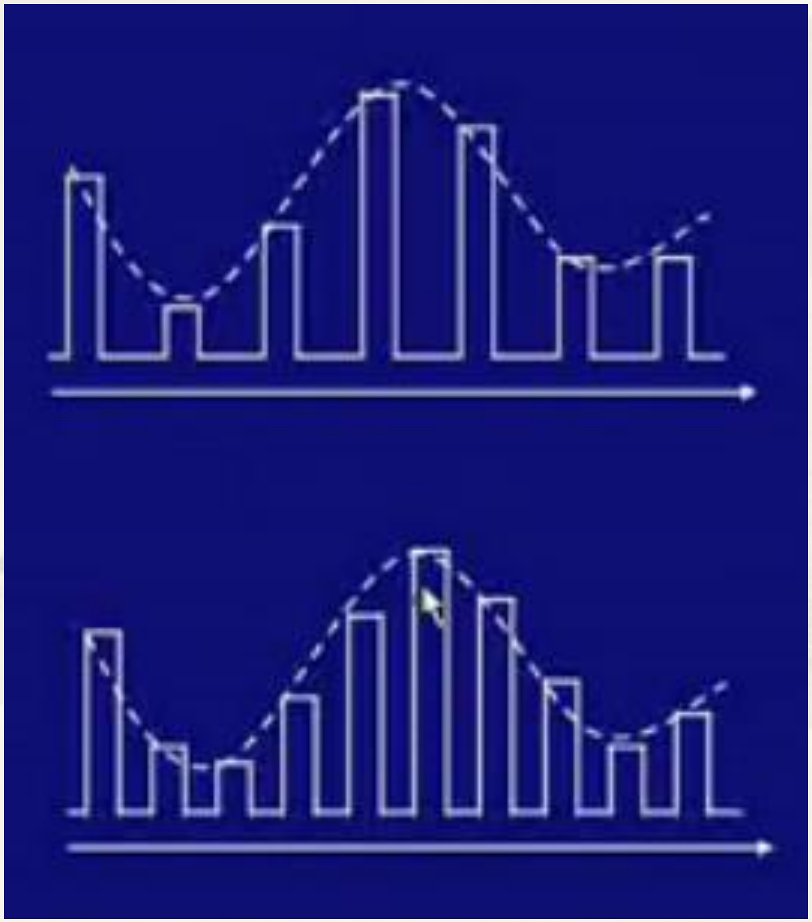
模拟 vs. 数字



模拟电路： 传输过程中各类因素干扰，导致失真
数字电路： 抗干扰能力强

结果可再现性-稳定可靠，精度更高
易于设计，灵活性和功能性
可编程性（HDL硬件描述语言）

模拟 vs 数字信号



足够多的数字量可以很准确地表达出模拟量

数字时代：
数码音乐
数码图片与视频

储存容量和质量的平衡

音乐”*when you believe*”片段:

1.1.1 数字技术的发展过程

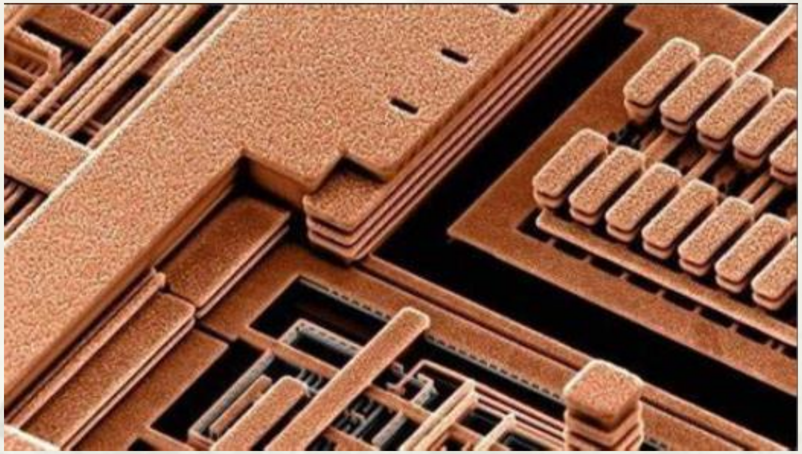
数字技术是一门应用学科，它的发展可分为5个阶段

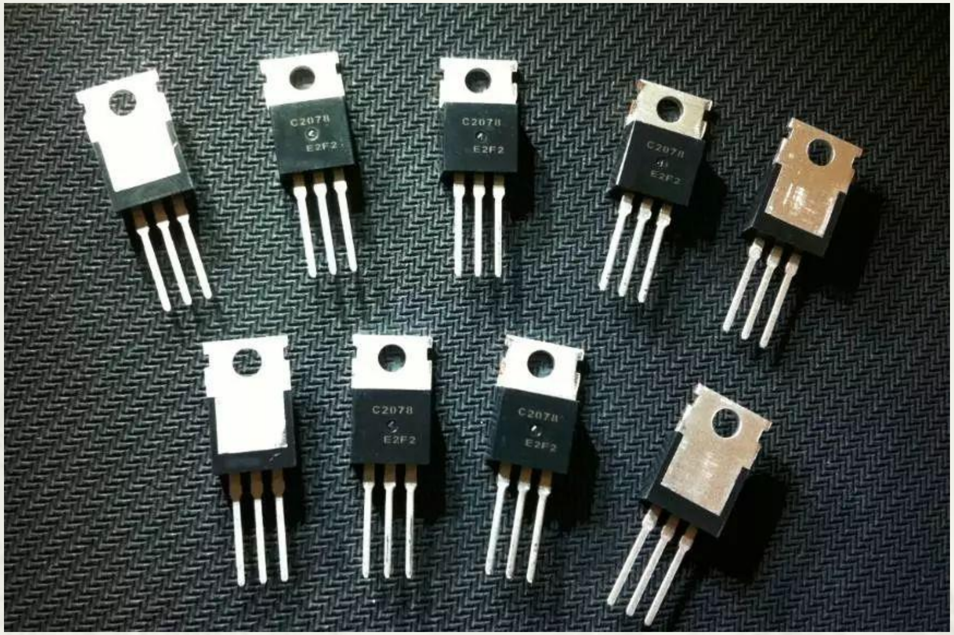
产生：20世纪30年代在通讯技术（电报、电话）首先引入二进制的信息存储技术。在1847年由英国科学家乔治.布尔(George Boole)创立布尔代数，并在电子电路中得到应用，形成开关代数，并有一套完整的数字逻辑电路的分析和设计方法。

① **初级阶段**：**20世纪40年代电子计算机中的应用，此时以电子管（真空管）作为基本器件。另外在电话交换和数字通讯方面也有应用。**

这部机器使用了18800个真空管，长50英尺，宽30英尺， 占地1500平方英尺，重达30吨（大约是一间半的教室大，六只大象重）。它的计算速度快，每秒可从事5000次的加法运算，运作了九年之久。耗电量150KW

② **第二阶段：** 20世纪60年代晶体管的出现，使得数字技术有一个飞跃发展，除了计算机、通讯领域应用外，在其它如测量领域得到应用。

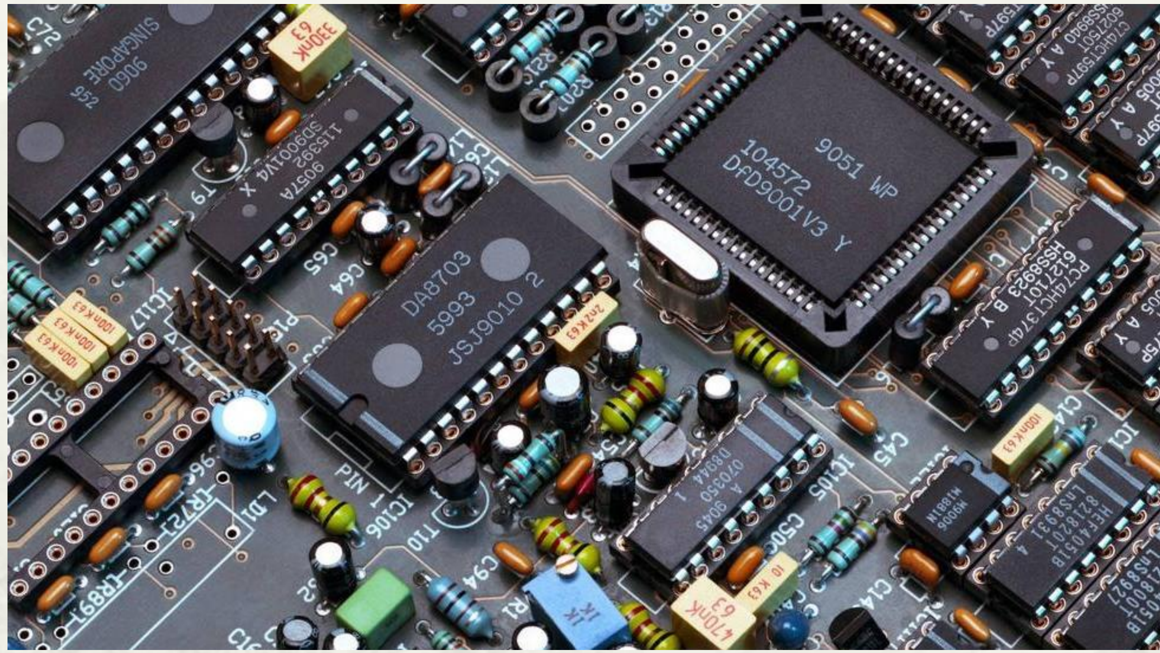




晶体管 vs. 电子管

寿命长，功耗小，响应速度快，稳定可靠，体积小

③ **第三阶段：**20世纪70年代中期集成电路的出现，使得数字技术有了更广泛应用，在各行各业医疗、雷达、卫星等领域都得到应用。



④ **第四阶段：**20世纪70年代中期到80年代中期，微电子技术的发展，使得数字技术得到迅猛的发展，产生了大规模和超大规模的集成数字芯片，应用在各行各业和我们的日常生活。

⑤ **20世纪80年代中期以后，产生一些专用和通用的集成芯片，以及一些可编程的数字芯片，并且制作技术日益成熟，使得数字电路的设计模块化和可编程的特点，提高了设备的性能、适用性，并降低成本。**

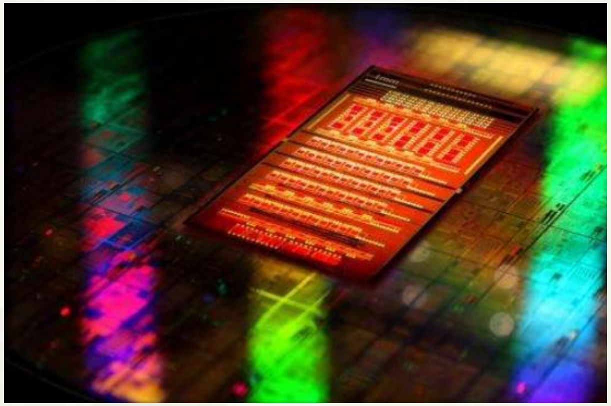
未来：

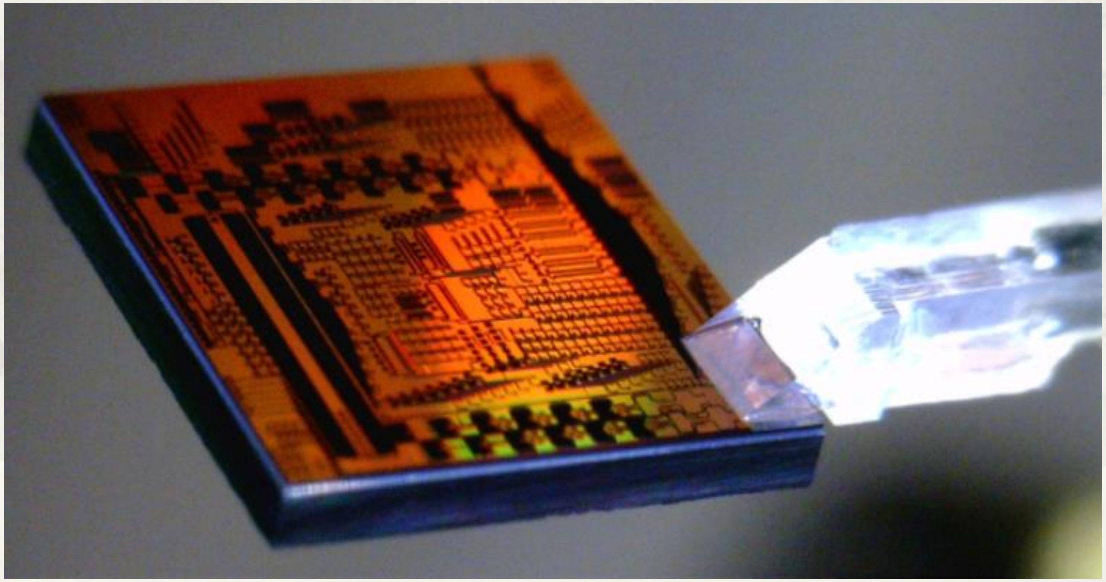
新的电子元器件制备技术

新材料，如纳米碳管、石墨烯等

新的电路设计

新概念，新产品的设计，如光计算/量子计算





1.1.2. 模拟信号与数字信号

模拟信号是表示模拟量的信号，模拟量是在时间和数值上都是连续的的物理量。模拟信号包括正弦波信号和脉冲信号，脉冲信号如方波、矩形波、尖脉冲锯齿波、梯形波等。

数字信号是用数码表示的，其数码中只有“1”和“0”两个数字，而“1”和“0”没有数量的意义，表示事物的两个对立面。数码可以表示数字信号的大小和状态，如1010可表示数量“10”，也可以表示某个事物的代号，如运动员的编号，这时将这些数码称为代码。数码的编写形式是多样的，其遵循的原则称为码制。码制的编写不受限制，但有一些通用的码制，如十进制、二进制、八进制和十六进制等等。

数字信号的魅力

例：设计一个自动售货饮料机的逻辑电路。它的投币口每次只能投入一枚五角或一元的硬币。投入一元五角钱硬币后机器自动给出一杯饮料；投入两元（两枚一元）硬币后，在给出饮料的同时找回一枚五角的硬币。

设投币信号为输入变量A和B，投入一枚一元硬币时为A＝1，否则为A＝0；投入一枚五角硬币时为B＝1，否则为B＝0。输出为Y和Z，给出饮料为Y＝1，否则为Y＝0；找回一枚五角硬币时为Z＝1，否则为Z＝0。

$$\boxed{\begin{cases} Y = Q_1 B + Q_1 A + Q_0 A \\ Z = Q_1 A \end{cases}}$$

Q：D触发器

1.2 几种常用的数制

数制：就是数的表示方法，把多位数码中每一位的构成方法以及按从低位到高位的进位规则进行计数称为进位计数制，简称数制。

最常用的是十进制，除此之外在数字电路和计算机中常用的是二进制、八进制和十六进制。

1.2.1 十进制

进位规则是 ***"逢十进一"***。任意一个n位整数、m位小数的十进制可表示为

$$
\begin{aligned}
(D)_{10} &= k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m} \\
&= k_{n-1}\times 10^{n-1}+\cdots+k_o\times 10^0+k_{-1}\times 10^{-1}+\cdots+k_{-m}\times 10^{-m}=\sum_{i=-m}^{n-1}k_i\times 10^i
\end{aligned}
$$

$$(D)_{10} = k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m}$$
$$= k_{n-1}\times 10^{n-1} + \cdots + k_o \times 10^0 + k_{-1}\times 10^{-1} + \cdots + k_{-m}\times 10^{-m} = \sum_{i=-m}^{n-1} k_i \times 10^i$$

其中：

k_i - 称为数制的系数，表示第i位的系数，十进制k_i的取值为0 ~ 9十个数， i 取值从 (n - 1) ~ 0的所有正整数到 - 1 ~ - m的所有负整数

10^i - 表示第i位的权值，10为基数，即采用数码的个数

n、m - 为正整数， n为整数部分的位数， m为小数部分的位数

例如:　　$(249.56)_{10} = 2\times10^2 + 4\times10^1 + 9\times10^0 + 5\times10^{-1} + 6\times10^{-2}$

其中$n = 3$，$m = 2$

若用N表示任意进制（称为N进制）的基数，则展成十进制数的通式为

$$
\boxed{\begin{aligned}(D)_N &= k_{n-1}k_{n-2}\cdots k_0k_{-1}\cdots k_{-m} \\ &= k_{n-1}\times N^{n-1}+\cdots+k_o\times N^0+k_{-1}\times N^{-1}+\cdots+k_{-m}\times N^{-m}=\sum_{i=-m}^{n-1}k_i\times N^i\end{aligned}}
$$

如N = 10为十进制，N = 2为二进制，N = 8为八进制， N = 16为十六进制。其中N为基数，k_i为第i位的系数， N^i表示第i位的权值。

一个数码的进制表示，可用下标，如$(N)_2$表示二进制；$(N)_{10}$表示十进制；$(N)_8$表示八进制，$(N)_{16}$表示十六进制。

也可用字母做下标，如$(N)_B$表示二进制，Binary；$(N)_D$表示十进制，Decimal；$(N)_O$表示八进制，Octal；$(N)_H$ 表示十六进制，Hexadecimal。

1.2.2 二进制：

进位规则是 **"逢二进一"**，任意一个n位整数、m位小数的二进制可表示为

$$
\boxed{\begin{aligned}&(D)_2 = k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m}\\&= k_{n-1}\times 2^{n-1}+\cdots+k_o\times 2^0+k_{-1}\times 2^{-1}+\cdots+k_{-m}\times 2^{-m}=\sum_{i=-m}^{n-1}k_i\times 2^i\end{aligned}}
$$

其中 k_i - 取值只有两个数码：0和1

2^i - 为二进制的权，基数为2

n、m - 为正整数

请写出（11011.101)$_2$的10进制表达式

$(11011.101)_2=1\times2^4+1\times2^3+0\times2^2+1\times2^1+1\times2^0$

$+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=(27.625)_{10}$

1.2.3 八进制

进位规则是 **“逢八进一”**，其基数为8。任意一个n位整数、m位小数的八进制可表示为

$$
\begin{aligned}
(N)_8 &= k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m} \\
&= k_{n-1}\times 8^{n-1}+\cdots+k_o\times 8^0+k_{-1}\times 8^{-1}+\cdots+k_{-m}\times 8^{-m}=\sum_{i=-m}^{n-1} k_i\times 8^i
\end{aligned}
$$

其中　k_i - 取值有8个数码：0 ~ 7

8^i - 为八进制的权，基数为8

n、m - 为正整数

如（$13.74)_8=1\times 8^1+3\times 8^0+7\times 8^{-1}+4\times 8^{-2}=(11.9375)_{10}$

1.2.4 十六进制

进位规则是 **"逢十六进一"**，其基数为十六。任意一个n位整数、m位小数的十六进制可表示为

$$
\begin{aligned}
(N)_{16} &= k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m} \\
&= k_{n-1}\times 16^{n-1}+\cdots+k_o\times 16^0+k_{-1}\times 16^{-1}+\cdots+k_{-m}\times 16^{-m}=\sum_{i=-m}^{n-1}k_i\times 16^i
\end{aligned}
$$

其中 k_i - 取值有16个数码：0~9、A(10)、B(11)、 C(12)、D(13)、E(14)、F(15)

16^i - 为十六进制的权，基数为16

n、m - 为正整数

如$(F9.1A)_{16}=15\times16^1+9\times16^0+1\times16^{-1}+10\times16^{-2}$

$=(249.1015625)_{10}$

目前在计算机上常用的是8位、16位和32位二进制数表示和计算 （64位计算机也已经商业化），由于8位、16位和32位二进制数都可以用2位、4位和8位十六进制数表示，故在编程时用十六进制书写非常方便。

表1.2.1为0～15个数码的不同进制表示。

表1.2.1

D	B	O	H	D	B	O	H
0	0000	00	0	8	1000	10	8
1	0001	01	1	9	1001	11	9
2	0010	02	2	10	1010	12	A
3	0011	03	3	11	1011	13	B
4	0100	04	4	12	1100	14	C
5	0101	05	5	13	1101	15	D
6	0110	06	6	14	1110	16	E
7	0111	07	7	15	1111	17	F

1.3 不同数制间的转换

数制转换：不同进制的数码之间的转换叫做数制转换

1.3.1 二进制数、八进制数和十六进制数转换成十进制数

即将二进制数、八进制数和十六进制数转换成十进制数，方法是将二进制数、八进制数和十六进制数按下列公式进行展开即可

$$
\boxed{
\begin{aligned}
&(D)_N = k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m} \\
&= k_{n-1}\times N^{n-1} + \cdots + k_o \times N^0 + k_{-1}\times N^{-1} + \cdots + k_{-m}\times N^{-m} = \sum_{i=-m}^{n-1} k_i \times N^i
\end{aligned}
}
$$

例如：

$$(11011.11)_B = 1\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 + 1\times 2^{-1} + 1\times 2^{-2}$$
$$= 16 + 8 + 2 + 1 + 0.5 + 0.25 = (27.75)_D$$

$$(176.51)_O = 1\times 8^2 + 7\times 8^1 + 6\times 8^0 + 5\times 8^{-1} + 1\times 8^{-2}$$
$$= 64 + 56 + 6 + 0.625 + 0.015625 = (126.64)_D$$

请试试将$(2AF.CE)_H$转换为10进制数

$$(2AF.CE)_H = 2\times 16^2 + 10\times 16^1 + 15\times 16^0 + 12\times 16^{-1} + 14\times 16^{-2}$$
$$= 512 + 160 + 15 + 0.75 + 0.0546875 = (687.80)_D$$

1.3.2 十进制数转换成二进制数：

将十进制数转换成二进制数，原则是“整数除2，小数乘2”

a. 十进制的整数转换

将十进制的整数部分用基数2去除，保留余数，再用商除2，依次下去，直到商为0为止，其余数即为对应的二进制数的整数部分（从低位到高位排列）。(课本p5例子)

$$A=a_{n-1}\times 2^{n-1}+a_{n-2}\times 2^{n-2}+\ldots\ldots+a_2\times 2^2+a_1\times 2^1+a_0\times 2^0$$

$$=2\times(a_{n-1}\times 2^{n-2}+a_{n-2}\times 2^{n-3}+\ldots\ldots+a_2\times 2^1+a_1\times 2^0)+a_0$$

$$=2\times(2\times(a_{n-1}\times 2^{n-3}+a_{n-2}\times 2^{n-4}+\ldots\ldots+a_2\times 2^0)+a_1)+a_0$$

$$=\ldots\ldots$$

1.3.2 十进制数转换成二进制数：

将十进制数转换成二进制数，原则是“整数除2，小数乘2”

b. 十进制的小数转换

将小数用基数2去乘，保留积的整数，再用积的小数继续乘2，依次下去，直到乘积是0为或达到要求的精度，其积的整数部分即为对应的二进制数的小数部分。

$$B=b_1\times 2^{-1}+ b_2\times 2^{-2}+ b_3\times 2^{-3}+\ldots\ldots+ b_{m-1}\times 2^{-(m-1)}+ b_m\times 2^{-m}$$

$$2\times B= b_1+(b_2\times 2^{-1}+ b_3\times 2^{-2}+\ldots\ldots+ b_{m-1}\times 2^{-(m-2)}+ b_m\times 2^{-(m-1)})$$

$$2\times(b_2\times 2^{-1}+ b_3\times 2^{-2}+\ldots\ldots+ b_{m-1}\times 2^{-(m-2)}+ b_m\times 2^{-(m-1)})$$
$$= b_2+ (b_3\times 2^{-1}+\ldots\ldots+ b_{m-1}\times 2^{-(m-3)}+ b_m\times 2^{-(m-2)})$$

……

例1.3.1 将（173.39)$_D$转化成二进制数,要求精度为1%。

解：其过程如下

a. 整数部分

即(173)$_D$=(10101101) $_B$

$$
\begin{array}{l}
2\underline{|173}\ldots\ldots\ldots 1(k_0)\\
2\underline{|86}\ldots\ldots\ldots\ldots 0(k_1)\\
2\underline{|43}\ldots\ldots\ldots\ldots 1(k_2)\\
2\underline{|21}\ldots\ldots\ldots\ldots 1(k_3)\\
2\underline{|10}\ldots\ldots\ldots\ldots 0(k_4)\\
2\underline{|5}\ldots\ldots\ldots\ldots\ldots 1(k_5)\\
2\underline{|2}\ldots\ldots\ldots\ldots\ldots 0(k_6)\\
2\underline{|1}\ldots\ldots\ldots\ldots\ldots 1(k_7)\\
\quad 0
\end{array}
$$

a. 小数部分

由于精度要求为1%，故应该令 $2^{-m} \le 1\%$

取对数，可得 $-m\log 2 \le -2$ $m \ge 6.6$

取m = 7 满足精度要求，过程如下

$0.39 \times 2 = 0.78 \cdots\cdots 0\ (k_{-1})$
$0.78 \times 2 = 1.56 \cdots\cdots 1\ (k_{-2})$
$0.56 \times 2 = 1.12 \cdots\cdots 1\ (k_{-3})$
$0.12 \times 2 = 0.24 \cdots\cdots 0\ (k_{-4})$
$0.24 \times 2 = 0.48 \cdots\cdots 0\ (k_{-5})$
$0.48 \times 2 = 0.96 \cdots\cdots 0\ (k_{-6})$
$0.96 \times 2 = 1.92 \cdots\cdots 1\ (k_{-7})$

即 $(0.39)_D = (0.0110001)_B$

故 $(173.39)_D$

$= (10101101.0110001)_B$

依此类推，对于十进制转换成其它进制，只要把基数2换成其它进制的基数即可。

1.3.3 二进制转换成八进制和十六进制

方法：由于3位二进制数可以有8个状态，000~111，正好是8进制，而4位二进制数可以有16个状态，0000 ~ 1111，正好是16进制，故可以把二进制数进行分组。八进制三位分为一组，不够补零，十六进制四位分为一组。

注：若将八进制或十六进制转换成二进制，即按三位或四位转成二进制数展开即可。

例1.3.2 将$(1011110.1011001)_2$转换成八进制和十六进制。

解：

$$(1011110.1011001)_B = (001\ \ 011\ \ 110.101\ \ 100\ \ 100)_2$$
$$= (136.544)_O$$

$$(1011110.1011001)_B = (0101\ \ 1110.1011\ \ 0010)_2$$
$$= (5E.B2)_H$$

例1.3.3 将$(703.65)_O$和$(9F12.04A)_H$转换成二进制数

解： $(703.65)_O = (111\ 000\ 011.110\ 101)_B$

$(9F12.04A)_H = (1001\ 1111\ 0001\ 0010.0000\ 0100\ 101)_B$

☺*提醒：*若要将十进制转换成八进制或十六进制，可先转换成二进制，再分组，转换成八进制或十六进制。

例1.3.4　将$(87)_D$ 转换成八进制数和十六进制数

解：先将87转化成二进制，过程如图,则

$$(87)_D = (1010111)_B=(001\ 010\ 111)_B= (127)_O$$

$$=(1010111)_B = (0101\ 0111)_B = (57)_H$$

2	87	……	1 (k_0)
2	43	……	1 (k_1)
2	21	……	1 (k_2)
2	10	……	0 (k_3)
2	5	……	1 (k_4)
2	2	……	0 (k_5)
2	1	……	1 (k_6)
	0		

1.4 二进制的算术运算

1.4.1 二进制算术运算的特点

看

当两个二进制数码表示两个数量的大小，并且这两个数进行数值运算，这种运算称为*算术运算*。其规则是“逢二进一”、“借一当二”。算术运算包括“加减乘除”，**但减、乘、除最终都可以化为带符号的加法运算。**

如两个数1001和0101的算术运算如下

$$\begin{array}{r} 1001 \\ +\,0101 \\ \hline 1110 \end{array}$$

$$\begin{array}{r} 1001 \\ -\,0101 \\ \hline 0100 \end{array}$$

$$\begin{array}{r} 1001 \\ \times\ 0101 \\ \hline 1001 \\ 0000 \\ 1001 \\ 0000 \\ \hline 0101101 \end{array}$$

$$\begin{array}{r} 1.11\cdots \\ 0101\,\overline{)\,1001} \\ 0101 \\ \hline 1000 \\ 0101 \\ \hline 0110 \\ 0101 \\ \hline 0010 \end{array}$$

1.4.2 反码、补码和补码运算

一、原码

在用二进制数码表示一个数值时，其正负是怎么区别的呢？二进制数的正负数值的表述是在二进制数码前加一位**符号位**，用“0”表示正数，用“1”表示负数，**这种带符号位的二进制数码称为原码。**

例如：＋17的原码为010001，－17的原码为110001

二、反码

反码是为了在求补码时不做减法运算。二进制的反码求法是：**正数的反码与原码相同，负数的反码为将原码除了符号位外的数值部分按位取反，即“1”改为“0”，“0”改为“1”。**

例如＋7和－7的原码和反码为：

＋7的原码为0 111，反码为0 111

－7的原码为1 111，反码为1 000

三、补码

当做二进制减法时，可利用补码将减法运算转换成加法运算。

1.模（模数）的概念：

把一个事物的循环周期的长度，叫做这个事件的模或模数。

如一年365天，其模数为365；钟表是以12为一循环计数的，故模数为12。十进制计数就是10个数码0～9的循环，故模为10。

以表为例来介绍补码运算的原理：
对于图1.4.1所示的钟表

当在5点时发现表停在10点，若想拨回有两种方法：

a.逆时针拨5个格，即 10 - 5 = 5，这是做减法。

b.顺时针拨七个格，即 10 + 7 = 17，由于模是12，故1相当于进位12，1溢出，故为7格，也是17 - 12 = 5，这是做加法。

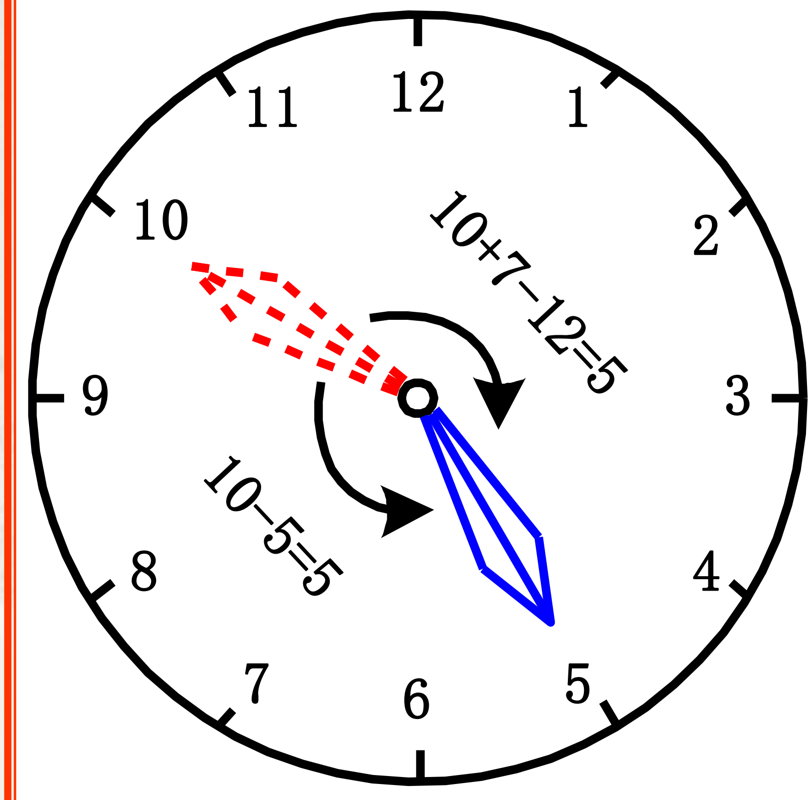


图1.4.1 补码的原理

由此可见10 + 7和10 - 5的效果是一样的，而5 + 7 = 12，故将7称为 - 5的补数（在模为12时），即补码，也可以说减法可以由补码的加法来代替

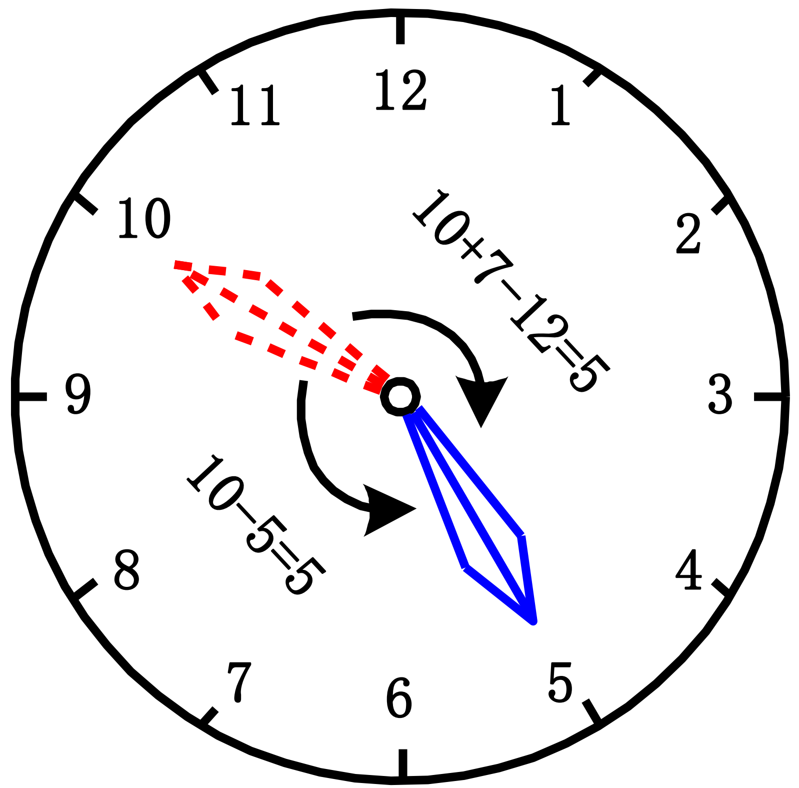


图1.4.1 补码的原理

2.补码的表示

正数的补码和原码相同，负数的补码是符号位为“1”，数值位按位取反加“1”，即“反码加1”

例如：	原码	反码	补码
[+7]	0 111	0 111	0 111
[-7]	1 111	1 000	1 001

注意:

1.采用补码后，可以方便地将减法运算转换成加法运算，而乘法和除法通过移位和相加也可实现，这样可以使运算电路结构得到简化。

2.正数的补码既是它所表示的数的真值，负数的补码数值部分不是它所示的数的真值。

3.已知原码，求补码和反码：正数的原码和补码、反码相同；负数的反码是符号位不变，数值位取反，而补码是符号位不变，数值位取反加“ 1”。

如：原码为10110100，其反码为11001011，补码为11001100。

5.已知补码，求原码：正数的补码和原码相同；负数的补码应该是数值位减“1”再取反。

如已知某数的补码为$(11101110)_B$，其原码为$(10010010)_B$

6.如果二进制的位数为n，则可表示的有符号位数的范围为（-2^{n-1} ~ $2^{n-1}-1$），如n＝8，则可表示(-128~127)，故在做加法时，注意两个数的绝对值不要超出它所表示数的范围。

例1.4.1 用二进制补码计算：75 + 28 、75 - 28、 - 75 + 28、 - 75 - 28

解：先求两个数的二进制原码和补码（用8位代码）

原码

$$\begin{cases} (+75)_D = (01001011)_B \\ (+28)_D = (00011100)_B \\ (-75)_D = (11001011)_B \\ (-28)_D = (10011100)_B \end{cases}$$

11 00011

1100100

补码

$$\begin{cases} (-75)_D = (10110101)_B \\ (-28)_D = (11100100)_B \end{cases}$$

$$\begin{array}{r} 75 \\ +\ 28 \\ \hline 103 \end{array} \quad \Rightarrow \quad \begin{array}{r} 0\ 1001011 \\ +\ \ 0\ 0011100 \\ \hline 0\ 1100111 \end{array}$$

$$\begin{array}{r} 75 \\ -\ 28 \\ \hline 47 \end{array} \quad \Rightarrow \quad \begin{array}{rrr} & 0 & 1001011 \\ + & 1 & 1100100 \\ \hline 1 & 0 & 0101111 \end{array}$$

溢出
(减去模)

$$\begin{array}{r} -\ 75 \\ +\ 28 \\ \hline -\ 47 \end{array} \quad \Rightarrow \quad \begin{array}{rrr} & 1 & 0110101 \\ + & 0 & 0011100 \\ \hline & 1 & 1010001 \end{array}$$

补码

$$\begin{array}{r} -\ 75 \\ -\ 28 \\ \hline -\ 103 \end{array} \quad \Rightarrow \quad \begin{array}{rrr} & 1 & 0110101 \\ + & 1 & 1100100 \\ \hline 1 & 1 & 0011001 \end{array}$$

溢出
(减去模)

表4-1. 4位带符号位二进制代码的原码、反码和补码对照表

十进制数	原码	反码	补码	十进制数	原码	反码	补码
＋7	0111	0111	0111	－1	1001	1110	1111
＋6	0110	0110	0110	－2	1010	1101	1110
＋5	0101	0101	0101	－3	1011	1100	1101
＋4	0100	0100	0100	－4	1100	1011	1100
＋3	0011	0011	0011	－5	1101	1010	1011
＋2	0010	0010	0010	－6	1110	1001	1010
＋1	0001	0001	0001	－7	1111	1000	1001
0	0000	0000	0000	－8	1000	1111	1000

1.5 二进制编码

1.5.1 三个术语

数码：代表一个确切的数字，如二进制数，八进制数等。

代码：特定的二进制数码组，是不同信号的代号，不一定有数的意义。

编码：n 位二进制数可以组合成2^n个不同的信息，给每个信息规定一个具体码组，这种过程叫编码。 数字系统中常用的编码有两类，一类是二进制编码；另一类是 二 - 十进制编码。另外无论二进制编码还是二 - 十进制编码，都可分成有权码（每位数码代表的权值固定）和无权码

1.5.2 十进制代码

用4位二进制代码表示1位十进制的0～9个数码，即二－十进制的编码。4位二进制代码可以有0000～1111十六个状态，则表示0～9十个状态可以有多种编码形式，其中常用的有8421码、余3码、2421码、5211码、余3循环码等，其中8421码、2421码、5211码为有权码，即每一位的1都代表固定的值。

什么是有权码

表1.5.1 几种编码形式

编码种类 十进制数	8421码 （BCD代码）	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
权	8421		2421	5211	

说明：

1. 8421码:又称BCD码，是最常用的十进制编码。其每位的权为8、4、2、1，按公式 $D=\sum k_i 2^i$ 展开，即可得对应的十进制数，如（0101）$_2 = 1\times 2^4 + 1\times 2^0 = 5$

2. 余3码不是有权码，由于它按二进制展开后十进制数比所表示的对应的十进制数大3。如0101表示的是2，其展开十进制数为5，故称为余3码。采用余3码的好处是：利用余3码做加法时，如果所得之和为10，恰好对应二进制16，可以自动产生进位信号。如0110 (3) + 1010 (7) = 1 0000 (10)；另外0和9、1和8、2和7…是互为反码，这对于求补很方便。

3\. 2421码是有权码，其每位的权为2、4、2、1，如$(1100)_2=1\times2+1\times4=6$，与余3码相同0和9、1和8、2和7…是互为反码。另外当任何两个这样的编码值相加等于9时，结果的4个二进制码一定都是1111。

4\. 5211码也是有权码，其每位的权为5、2、1、1，如$(0111)_2=1\times2+1\times1+1\times1=4$，主要用在分频器上。

5\. 余3循环码是无权码，它的特点是相邻的两个代码之间只有一位状态不同。这在译码时不会出错（竞争－冒险）。

自然码：有权码，每位代码都有固定权值，结构形式与二进制数完全相同，最大计数为$2^n - 1$，n为二进制数的位数，也叫二进制代码

循环码：也叫格雷码，它是无权码，每位代码无固定权值，其组成是格雷码的最低位是0110循环；第二位是00111100循环；第三位是0000111111110000循环，以此类推可以得到多位数的格雷码。格雷码的特点是任何相邻的两个码组中，仅有一位代码不同，抗干扰能力强，主要用在计数器中。

试试写出$(34.7)_{10}$的8421码和二进制码

$(34.7)_{10} = (0011\ 0100.0111)_{8421}$

$(34.7)_{10} = (100010.1011)_2$

1.5.4 美国信息交换标准代码（ASCII）

第一章 数码和码制

内容提要

1. 数字电路简介
2. 数制和码制的一些基本概念和术语
3. 数字电路中常用的数制和编码
4. 不同数制之间的转化方法
5. 二进制数算术运算的原理和方法

作 业 1:

请见Blackboard