数字电子技术基础

授课教师：梁亮，喻锦程

L301（4-5月搬至G311）

E-mail: yujincheng@hit.edu.cn

QQ: 935853739

成绩计算方式：

平时成绩20%+期末考试80%

平时成绩包括作业、考勤

作业：每章1次，布置后一周内完成（有假期可能延后）
并交给各班学委，学委交各班负责助教

考勤：每学期进行4-5次小测，作为考勤参照

上课时间地点：

1-15周 周一3-4节周三1-2节 T3402 (未包括特殊调休安排)

数字电子技术基础

教材和参考书

数字电子技术基础（第六版）清华高教版

数字电子技术基础（第二版）哈工大高教版

任意版本的Verilog语言 参考书

XILINX FPGA/VIVADO软件

相关课程材料

B站 数字电子技术基础/清华大学 王红

MOOC 国防科技大学 丁文霞

数字电子技术基础

电路原理

模拟电子技术基础
数字电子技术基础

模拟电子技术实验
数字电子技术实验

微机原理、嵌入式系统

电力电子

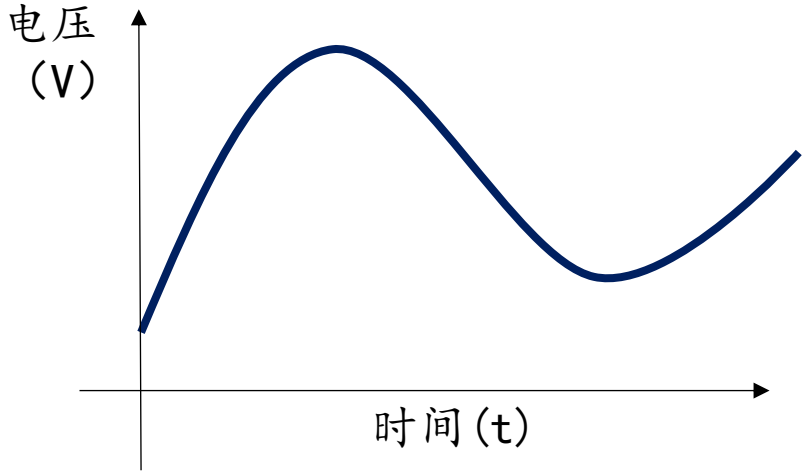
课程目的：掌握基本概念，基本分析方法，
基本设计方法和基本的实验技能

容易之处：涉及电路分析较少，题型变化相对小

困难之处：内容丰富度较高，同志仍需努力

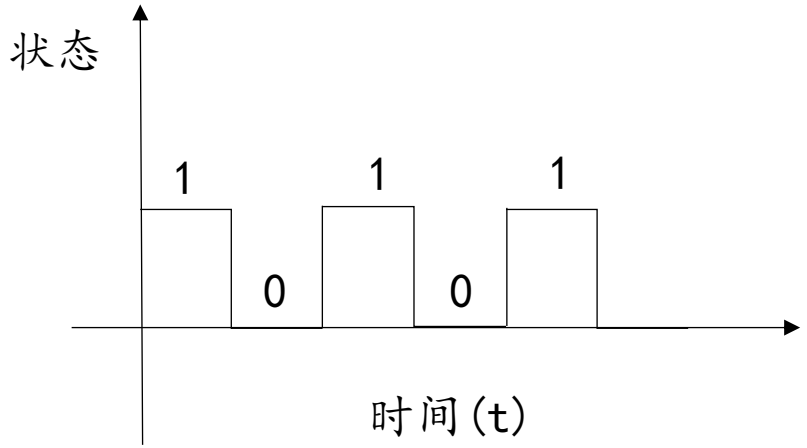
1. 模拟信号与数字信号

在时间上和数量上都是连续的物理量称为模拟量。表示模拟量的信号叫做模拟信号。把工作在模拟信号下的电子电路叫模拟电路。

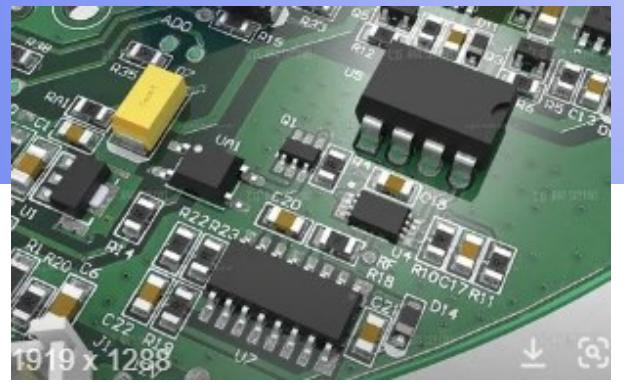


1. 模拟信号与数字信号

在时间上和数量上都是离散的物理量称为数字量（存在一个最小数量单位Δ）。把表示数字量的信号叫数字信号。把工作在数字信号下的电子电路叫数字电路。



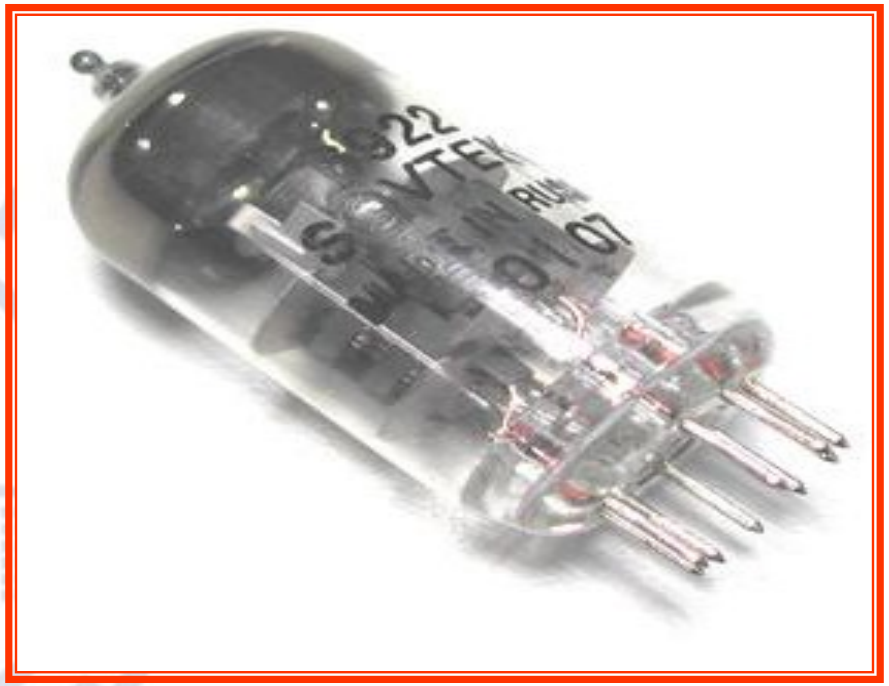
绪论



2. 电子技术的发展过程：

与电子器件的更新紧密相连

初级阶段：20世纪40年代电子计算机中的应用。此外在电话交换和数字通讯方面也有应用。此时基本器件是电子管（真空管）。



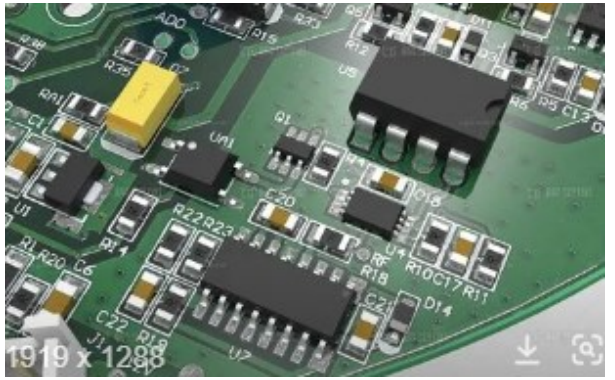
真空管的电极封装在真空的玻璃或金属外壳中，工作时必须将阴极加热至很高的温度才能发射出电子流，所以不仅功耗大，而且寿命短，同时体积和重量也比较大。

绪论

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

与电子管相比，晶体管功耗小，寿命长，体积重量也大大减小。

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的应用，在各行各业医疗、雷达、卫星等领域都得到应用。



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。

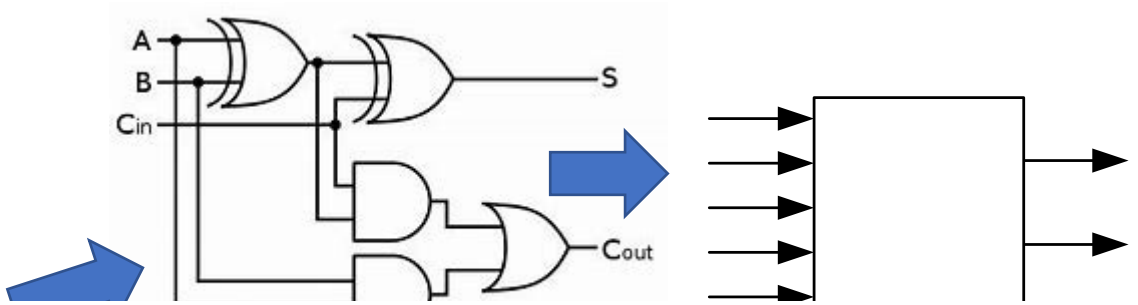
绪论

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，这是数字电路今后发展的趋势。

“数电”是进行芯片研发与设计的基础。

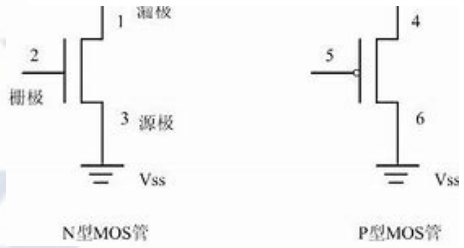
绪论

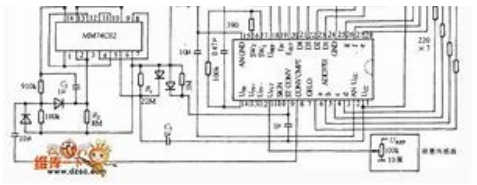
学习对象：



数电学习注意2：

数电是应用性极强的学科，学习过程中应该相对重电子器件的应用，轻内部结构。学习电路结构不是数电的目的，掌握电路功能、分析方法和思路才是目的。





复杂系统

绪论

学习目的：

- 理解底层系统如何工作　Bottom-up
- 学习数字电路的理论基础
- 学习数字电路设计和分析的原则（抽象方法，接口等）
- 学习各个层级上设计电路的方法

- 学科历史
- 系统方法
- 算法
- 电路调试方法

本章内容

1.1 概述
1.2 几种常用的数制
1.3 不同数制间的转换
1.4 二进制算数运算
1.5 几种常用的编码

数字信号是以**数码**表示的。**数码的编写规则**称为**数制**。

数码的数制的编写不受限制，但有一些**通用的数制**，如十进制、二进制、八进制和十六进制等等。

数字电路中使用最多的是**二进制**，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。

本章内容

1.1 概述
1.2 几种常用的数制
1.3 不同数制间的转换
1.4 二进制算数运算
1.5 几种常用的编码

数码可以表示**数字信号的大小和状态**，如1010可表示**数量**“10”。当两个数码分别表示两个数量大小时，可以进行数量间的加减乘除等**算数运算**。

本章内容

1.1 概述
1.2 几种常用的数制
1.3 不同数制间的转换
1.4 二进制算数运算
1.5 几种常用的编码

数码可以表示数字信号的大小和状态，如1010可表示**数量**“10”。当两个数码分别表示两个数量大小时，可以进行数量间的加减乘除等**算数运算**。

同时，数码亦可表示**不同事物的代号，不再有数量大小的含义**，如运动员的编号等。这时，数码又可称为**代码**。为便于记忆和查找，在编制代码时需要遵循一定的规则，这些规则成为**码制**。

本章内容

1.1 概述
1.2 几种常用的数制
1.3 不同数制间的转换
1.4 二进制算数运算
1.5 几种常用的编码

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

常用数制：

十进制 (Decimal, D)

二进制 (Binary, B)

八进制 (Octal, O)

十六进制 (Hexadecimal, H)

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

有时也用字母做下标，如 $(N)_B$表示二进制，B－Binary；$(N)_D$表示十进制，D－Decimal；$(N)_O$表示八进制，O－Octal；$(N)_H$ 表示十六进制，H－Hexadecimal；

1.2 几种常用的数制

数制	基数	数码	计数规则	一般表达式	计算机中英文表示
十进制	10	0~9	逢十进一	$\sum_{i=-m}^{n-1} k_i \times 10^i$	D
二进制	2	0、1	逢二进一	$\sum_{i=-m}^{n-1} k_i \times 2^i$	B
八进制	8	0~7	逢八进一	$\sum_{i=-m}^{n-1} k_i \times 8^i$	O
十六进制	16	0~9、ABCDEF	逢十六进一	$\sum_{i=-m}^{n-1} k_i \times 16^i$	H
N进制	N	0~ (N-1)	逢N进一	$\sum_{i=-m}^{n-1} k_i \times N^i$	-

例如：

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

1.2 几种常用的数制

数制	基数	数码	计数规则	一般表达式	计算机中英文表示
十进制	10	0~9	逢十进一	$\sum_{i=-m}^{n-1} k_i \times 10^i$	D
二进制	2	0、1	逢二进一	$\sum_{i=-m}^{n-1} k_i \times 2^i$	B
八进制	8	0~7	逢八进一	$\sum_{i=-m}^{n-1} k_i \times 8^i$	O
十六进制	16	0~9、ABCDEF	逢十六进一	$\sum_{i=-m}^{n-1} k_i \times 16^i$	H
N进制	N	0~ (N-1)	逢N进一	$\sum_{i=-m}^{n-1} k_i \times N^i$	-

例如：

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

$$(1011.01)_{2} = 1\times2^{3} + 0\times2^{2} + 1\times2^{1} + 1\times2^{0} + 0\times2^{-1} + 1\times2^{-2} = (11.25)_{10}$$

1.2 几种常用的数制

数制	基数	数码	计数规则	一般表达式	计算机中英文表示
十进制	10	0~9	逢十进一	$\sum_{i=-m}^{n-1} k_i \times 10^i$	D
二进制	2	0、1	逢二进一	$\sum_{i=-m}^{n-1} k_i \times 2^i$	B
八进制	8	0~7	逢八进一	$\sum_{i=-m}^{n-1} k_i \times 8^i$	O
十六进制	16	0~9、ABCDEF	逢十六进一	$\sum_{i=-m}^{n-1} k_i \times 16^i$	H
N进制	N	0~ (N-1)	逢N进一	$\sum_{i=-m}^{n-1} k_i \times N^i$	-

例如：

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

$$(1011.01)_2 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 + 0\times2^{-1} + 1\times2^{-2} = (11.25)_{10}$$

$$(13.74)_8 = 1\times8^1 + 3\times8^0 + 7\times8^{-1} + 4\times8^{-2} = (11.9375)_{10}$$

1.2 几种常用的数制

数制	基数	数码	计数规则	一般表达式	计算机中英文表示
十进制	10	0~9	逢十进一	$\sum_{i=-m}^{n-1} k_i \times 10^i$	D
二进制	2	0、1	逢二进一	$\sum_{i=-m}^{n-1} k_i \times 2^i$	B
八进制	8	0~7	逢八进一	$\sum_{i=-m}^{n-1} k_i \times 8^i$	O
十六进制	16	0~9、ABCDEF	逢十六进一	$\sum_{i=-m}^{n-1} k_i \times 16^i$	H
N进制	N	0~ (N-1)	逢N进一	$\sum_{i=-m}^{n-1} k_i \times N^i$	-

例如：

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

$(1011.01)_2 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 + 0\times2^{-1} + 1\times2^{-2} = (11.25)_{10}$

$(13.74)_8 = 1\times8^1 + 3\times8^0 + 7\times8^{-1} + 4\times8^{-2} = (11.9375)_{10}$

$(F9.1A)_{16} = 15\times16^1 + 9\times16^0 + 1\times16^{-1} + 10\times16^{-2} = (249.1015625)_{10}$

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，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

1.2 几种常用的数制

数制	基数	数码	计数规则	一般表达式	计算机中英文表示
十进制	10	0~9	逢十进一	$\sum_{i=-m}^{n-1} k_i \times 10^i$	D
二进制	2	0、1	逢二进一	$\sum_{i=-m}^{n-1} k_i \times 2^i$	B
八进制	8	0~7	逢八进一	$\sum_{i=-m}^{n-1} k_i \times 8^i$	O
十六进制	16	0~9、ABCDEF	逢十六进一	$\sum_{i=-m}^{n-1} k_i \times 16^i$	H
N进制	N	0~ (N-1)	逢N进一	$\sum_{i=-m}^{n-1} k_i \times N^i$	-

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

（一）二/八/十六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十进制数一般表达式进行展开即可：“按位加权求和”

$$(D)_N = k_{n-1}k_{n-2}\cdots k_0 k_{-1}\cdots k_{-m}$$

$$= k_{n-1}\times N^{n-1} + \cdots + k_o \times N^0 + k_{-1}\times N^{-1} + \cdots + k_{-m}\times N^{-m} = \sum_{i=-m}^{n-1} k_i \times N^i$$

例如：

$$(11011.11)_B = 1\times2^4+1\times2^3+0\times2^2+1\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}$$
$$= 16+8+2+1+0.5+0.25 = (27.75)_D$$

$$(176.51)_O = 1\times8^2+7\times8^1+6\times8^0+5\times8^{-1}+1\times8^{-2}$$
$$= 64+56+6+0.625+0.015625 = (126.64)_D$$

$$(2AF.CE)_H = 2\times16^2+10\times16^1+15\times16^0+12\times16^{-1}+14\times16^{-2}$$
$$= 512+160+15+0.75+0.0546875 = (687.80)_D$$

（二）十进制数转换成二进制数

原则是“整数：除2取余，逆序排列；
小数：乘2取整，顺序排列”

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，将其余数逆序排列即为对应的二进制数的整数部分。

（二）十进制数转换成二进制数

a. 十进制的整数转换

整数转换规则证明

假定十进制整数为$(S)_{10}$，其等值的二进制数为$(k_nk_{n-1}\ldots k_0)_2$，

则可知：

$$(S)_{10}=k_n2^n+k_{n-1}2^{n-1}+\ldots k_n2^1+k_02^0=2\times(k_n2^{n-1}+k_{n-1}2^{n-2}+\ldots+k_1)+k_0 \quad (1)$$

上式表明，若将$(S)_{10}$÷2，则得到的商为$k_n2^{n-1}+k_{n-1}2^{n-2}+\ldots+k_1$，

而余数即k_0若将$(S)_{10}$÷2的商进行下列变换：

$$k_n2^{n-1}+k_{n-1}2^{n-2}+\ldots k_1=2\times(k_n2^{n-2}+k_{n-1}2^{n-3}+\ldots+k_2)+k_1 \quad (2)$$

将(2)再÷2，则可得余数k_1。

依次类推，可得二进制整数的每一位，且最后得到的是最高位。

（二）十进制数转换成二进制数

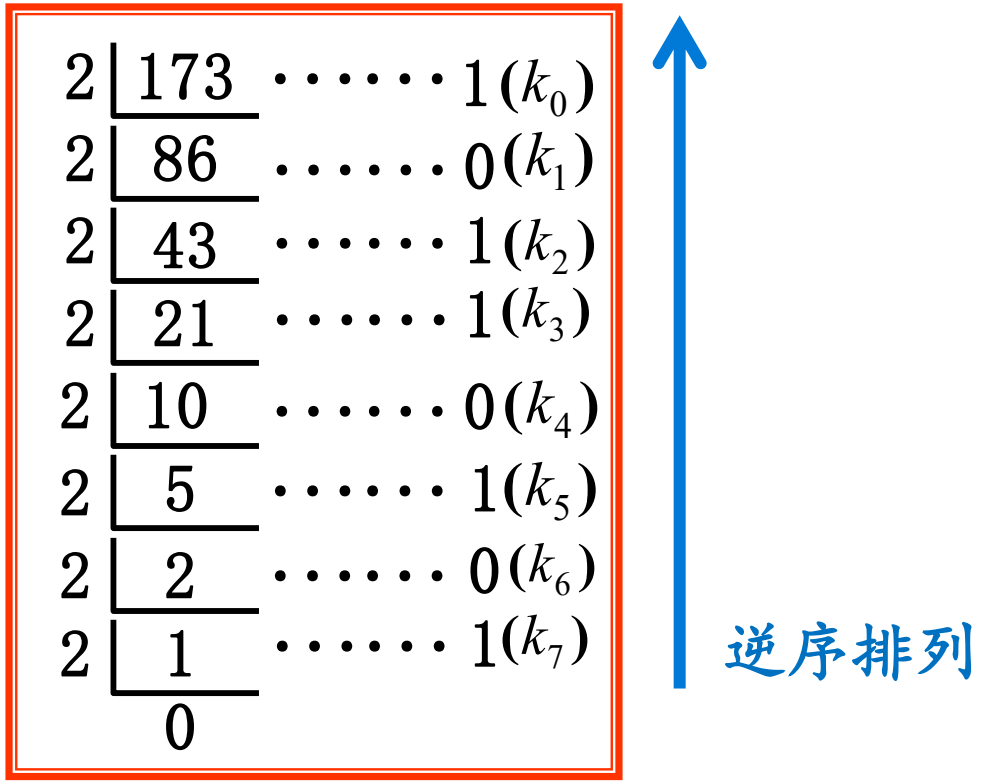
a. 十进制的整数转换

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

整数部分:

即 $(173)_D=(10101101)_B$



（二）十进制数转换成二进制数

b.十进制的小数转换

小数转换规则证明

假定十进制小数为$(S)_{10}$，其等值的二进制数为$(0.k_{-1}k_{-2}\ldots k_{-m})_2$，则可知：

$$(S)_{10}=k_{-1}2^{-1}+k_{-2}2^{-2}+\ldots k_{-m+1}2^{-m+1}+k_{-m}2^{-m} \quad (1)$$

$$(S)_{10}=2^{-1}[k_{-1}+(k_{-2}2^{-1}+\ldots k_{-m+1}2^{-m+2}+k_{-m}2^{-m+1})] \quad (2)$$

由(2)知，将$(S)_{10}$×2所得乘积的整数部分即为k_{-1}，将(2)的小数部分再×2：

$$k_{-2}2^{-1}+\ldots k_{-m+1}2^{-m+2}+k_{-m}2^{-m+1}=2^{-1}[k_{-2}+(k_{-3}2^{-1}+\ldots+k_{-m}2^{-m+2})] \quad (3)$$

亦即乘积的整数部分就是k_{-2}。

依次类推，可得二进制整数的每一位，且最后得到的是最高位。

（二）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，将其余数**逆序排列**即为对应的二进制数的整数部分。

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**顺序排列**即为对应的二进制数的小数部分

例1.3.1 将 $(173.39)_D$ 转化成二进制数,要求精度为1%。

解：小数部分

由于精度要求，令 $2^{-m} \le 1\%$ $2^{-m} \le 1\% = 10^{-2}$ $2^{m} \ge 100$

取对数，可得 $m \lg_{10} 2 \ge \lg_{10} 100 = 2$ $m \ge 6.6$

取m＝7 满足精度要求，过程如下

即$(0.39)_D$=$(0.0110001)_B$

故 $(173.39)_D$

=$(10101101.0110001)_B$

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

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。

三、 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：

$$(703.65)_O = (111000011.110101)_B$$

$$(9F12.04A)_H = (100111110010010.00000100101)_B$$

☺提醒：若要将十进制转换成八进制或16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

$$(87)_D = (1010111)_B = (001\ 010\ 111)_B$$
$$= (0101\ 0111)_B = (127)_O$$
$$= (57)_H$$

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

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

```
  1001
+ 0101
------
  1110
```

```
  1001
- 0101
------
  0100
```

```
    1001
  × 0101
--------
    1001
   0000
  1001
 0000
-------
0101101
```

```
          1.11···
      ------------
0101 ⟌ 1001
       0101
       ----
       1000
        0101
        ----
        0110
         0101
         ----
          0010
```

若能再设法将减法操作转为某种形式的加法操作，那么加减乘除都可以通过“**移位**”与“**相加**”实现。

介绍模（或模数）的概念

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

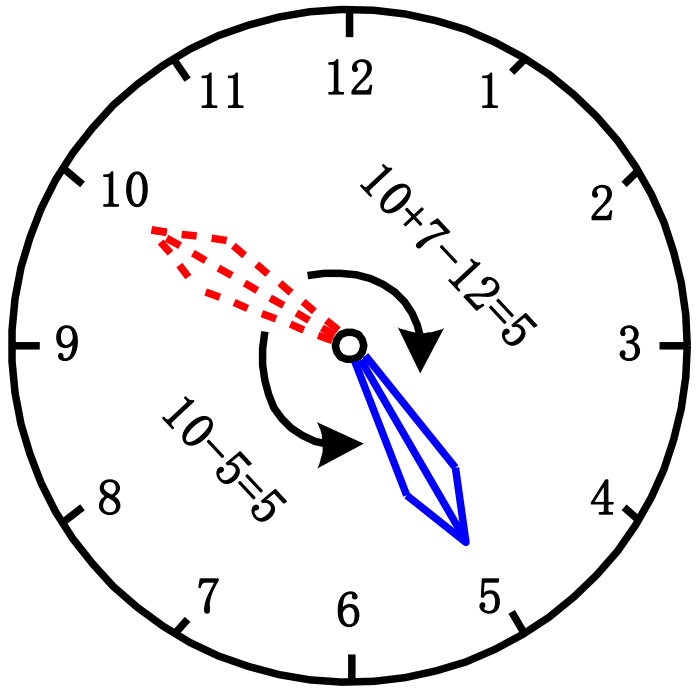


图1.4.1 补码的原理

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7个格，即10＋7＝17，由于**模是12**，故1相当于进位12，1溢出、**舍弃进位，**故为7格，也是17－12＝5，这是做加法。

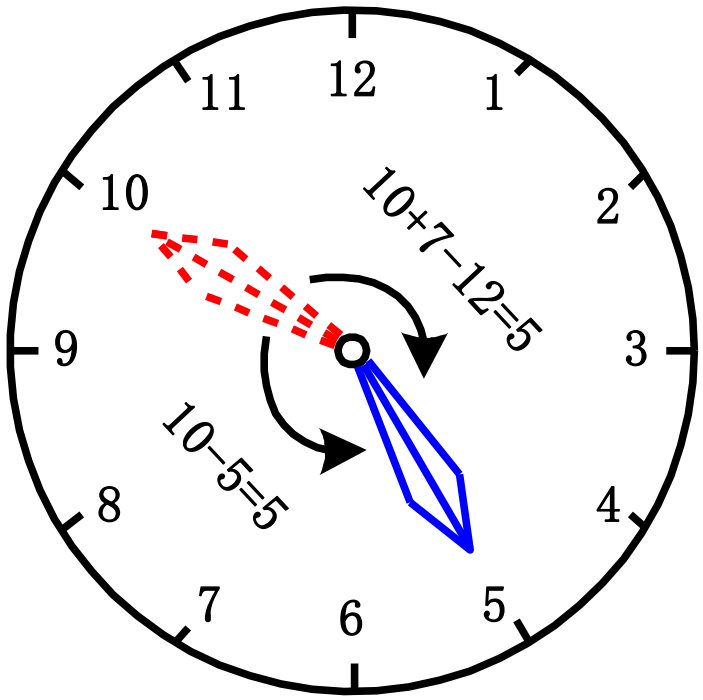


图1.4.1 补码的原理

由此可见10＋7和10－5的效果是一样的，而5＋7＝12即为模数，故将7称为5的补数，即补码 (Complement)，也可以说在舍弃进位条件下，减法可以由补码的加法来代替，补码的概念可将所有减法都转换为加法！

1.4.2 原码、反码和补码运算

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**

使用原码进行减法运算步骤较为复杂。为简化运算，采用**补码相加**代替**原码相减**。

二、反码

反码是为了在求补码时不做减法运算而设定的。二进制的反码求法是：正数的反码与原码相同，负数的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

注：0的反码有两种表示，＋0的反码为0 000，－0的反码为1 111

三、补码：

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

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***，**模数为2^n**，它的**补码 $(N)_{COMP}$** 表示方法为**模数-*N***

$$(N)_{COMP}=\begin{cases}N，\ N为正数\\ 2^n-N，\ N为负数\end{cases}$$

	原码	反码	补码
[-7]	1 111	1 000	1 001

若正常计算：

[-7的补码] =[-8] 1000 – [-7] 111= 001 较为复杂，若用反码概念计算则简单许多！

为了**避免在求补码的过程中做减法运算**，通常先求出N的反码$(N)_{INV}$，再在负数的反码上加1而得到补码。因此反码定义：

$$(N)_{INV}=\begin{cases}N，\ N为正数\\ (2^n-1)-N，\ N为负数\end{cases}$$

2^n-1为n位全为1的二进制数，所以只要将N中每一位的1改为0、0改为1，就得到了$(N)_{INV}$。因此常常**先求得反码，再求补码**。

二进制数补码运算：

在数字电路中，用原码运算求两个正数M和N的差值M-N时，首先要对减数和被减数进行比较，然后由大数减去小数，最后决定差值的符号，完成这个运算，电路复杂，速度慢，所以常引入**补码来实现减法运算**。

设A和B依次为被加数（或被减数）和加数（或减数），用补码实现加减运算的步骤如下：

Step1：把A与B（减法时为-B）均表示成补码形式；
Step2：把两个补码相加，且把符号位也看成二进制的最高位参与运算；
Step3：若和数的最高位有进位，将该进位舍弃。

“$[A]_{补}+[B]_{补}=[A+B]_{补}$”

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）

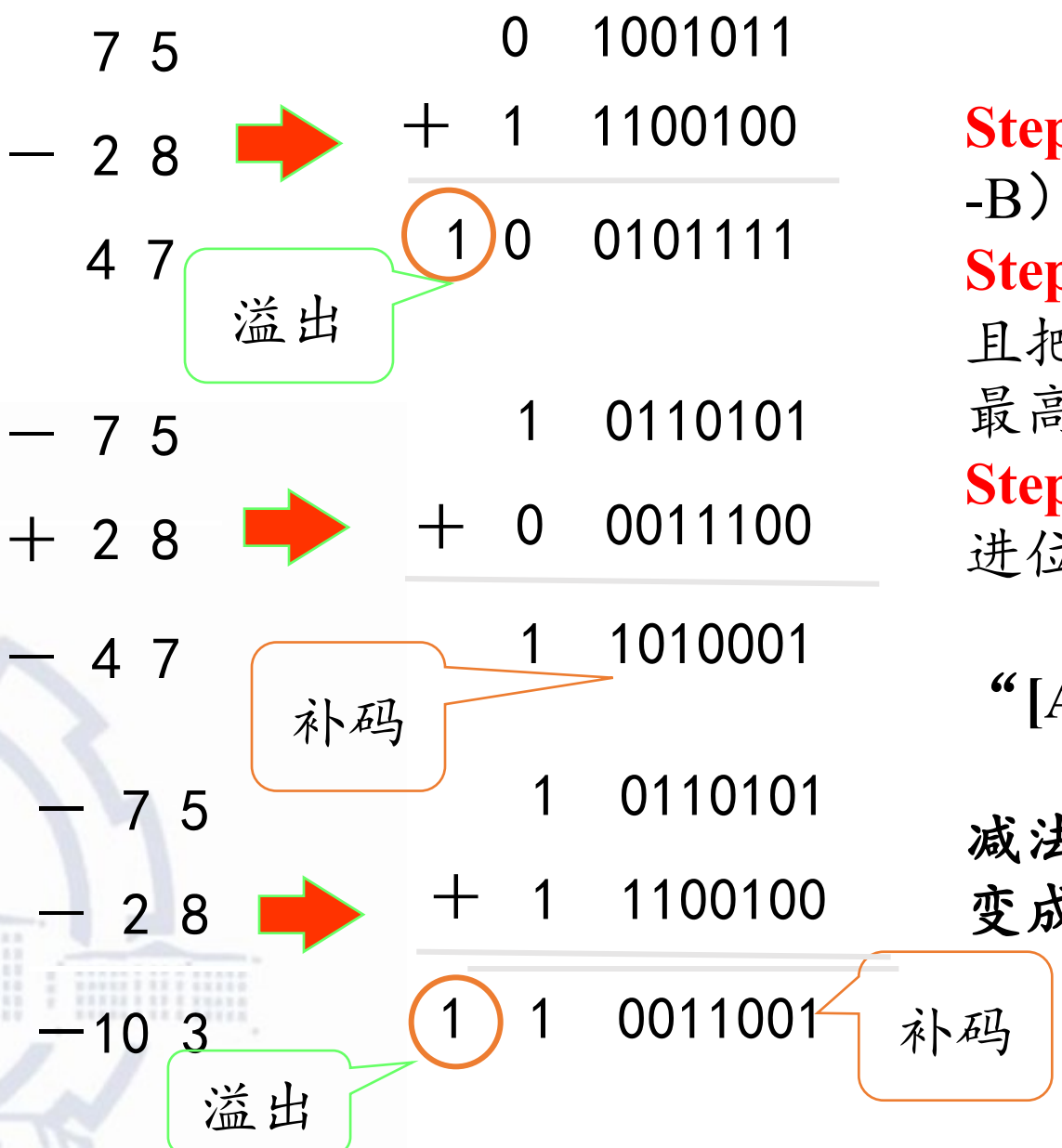
原码 →

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

补码 →

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

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



Step1：把A与B（减法时为-B）均表示成补码形式；

Step2：把两个补码相加，且把符号位也看成二进制的最高位参与运算；

Step3：若和数的最高位有进位，将该进位舍弃。

“$[A]_{补}+[B]_{补}=[A+B]_{补}$”

减法变加法的实质是把 A-B 变成了 A+(-B)

表4—1为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				

1.5 几种常用的编码

一、十进制代码：

用二进制代码表示十进制数的0-9这十个状态，即每位十进制数字都可用这种十进制代码来表示。

$(234)_{10}=(0010\ 0011\ 0100)_{8421\text{-}BCD}$

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

1. 8421码: 是最常用的十进制编码。其是一种恒权代码，每位的权为8、4、2、1，按如下公式展开：

$$D=\sum k_i 2^i$$

即可得对应的十进制数，如

$$(1001)_2=1\times2^3+1\times2^0=9$$

十进制数	8421码
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

1. 8421码: 是最常用的十进制编码。其是一种恒权代码，每位的权为8、4、2、1。

1010-1111这6个编码在8421BCD码里面是禁用码

n位十进制数 的BCD码由 n组BCD码 构成。 如

$(234)_{10}=(0010\ 0011\ 0100)_{8421\text{-}BCD}$。

So，8421BCD码 转 二进制数？

十进制数	8421码
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

2. 2421码是有权码，其每位的权为2、4、2、1，如$(1100)_2=1\times2+1\times4=6$，其0和9、1和8、2和7...互为反码（反射特性）。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：便能从高位直接产生进位信号。

十进制数	2421码
0	0000
1	0001
2	0010
3	0011
4	0100
5	1011
6	1100
7	1101
8	1110
9	1111

3. 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。

十进制数	余3码
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

4. 5211码也是有权码，其每位的权为5、2、1、1，如$(0111)_2$=1×2+1×1+1×1=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，是一种错误最小化代码，使电路工作更稳定。

十进制数	5211码	余3循环码
0	0000	0010
1	0001	0110
2	0100	0111
3	0101	0101
4	0111	0100
5	1000	1100
6	1001	1101
7	1100	1111
8	1101	1110
9	1111	1010

二、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，避免过渡噪声，抗干扰能力强，主要用在计数器中。

编码顺序	自然码	格雷码	编码顺序	自然码	格雷码
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

余三循环码是取自4位典型格雷码的3-12这10个代码。

三、美国信息交换标准代码（ASCII）

ASCII是一组七位二进制代码，共128个（自学）

应用：计算机和通讯领域

作　业

- 补充:$(01111001)_{8421BCD码}$等值的二进制数是( )$_2$.
- 1.1,
- 1.4(4), 1.5(4),
- 1.9(1), (2)
- 1.12(1), (3)