

数字电子技术基础

授课教师：梁亮

G栋513室

E-mail: liangl@hit.edu.cn



成绩计算方式:

平时成绩20%+期末考试80%

平时成绩包括作业、考勤和课上问答

上课时间地点:

1-15周 周三1-2节 T3201 周五3-4节 T3501

1-15周 周三3-4节 H508 周五1-2节 H308

固定答疑时间地点:

周四下午 G513



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



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

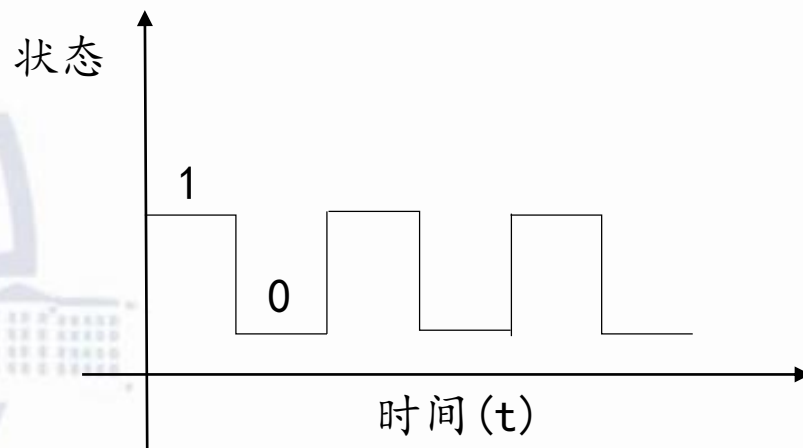
微机原理、嵌入式系统

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

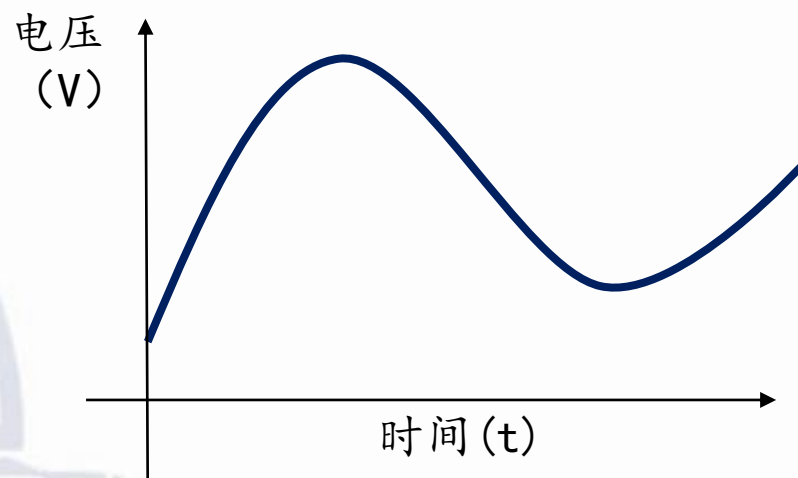
1. 模拟信号与数字信号

信号可分为模拟信号和数字信号

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。



除数字信号以外的信号称为模拟信号，把工作在模拟信号下的电子电路叫模拟电路。



下列信号哪些是连续的，哪些是离散的

颜色 (Color)

声音 (Sound)

车辆 (Cars)

狗 (Dogs)

电压、电流信号 (Voltage and Current)

数字电路VS模拟电路

工作信号、研究的对象、分析/设计方法以及所用的数学工具都具有明显的不同



电路原理

模拟电子技术基础
数字电子技术基础

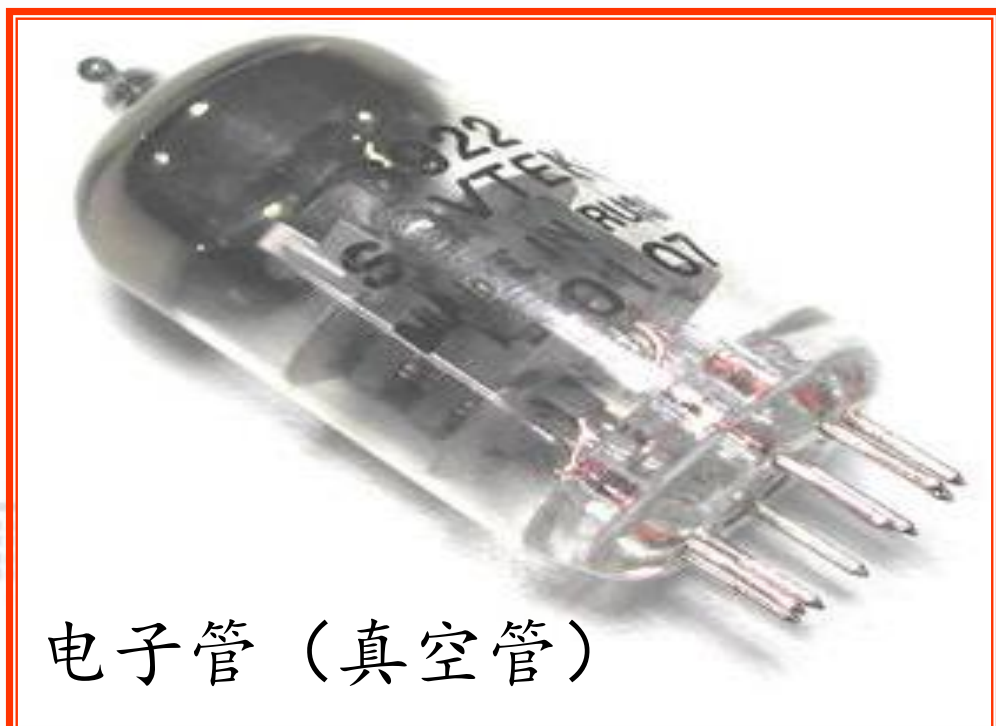
模拟电子技术实验
数字电子技术实验

微机原理、嵌入式系统

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

2. 电子技术的发展过程

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）。



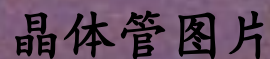
电子管（真空管）



19世纪80年代，爱迪生发现了“爱迪生效应”，由此英国物理学家和电气工程师弗莱明发明了二极管。美国发明家德福雷斯特，在二极管的灯丝和板极之间巧妙地加了一个栅板，从而发明了第一只真空三极管。许多人都将三极管的发明看作电子工业真正的诞生起点。电子管的问世，推动了无线电电子学的蓬勃发展。

缺点：电子管十分笨重，能耗大、寿命短、噪声大，制造工艺也十分复杂

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，在其它如测量领域得到应用



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

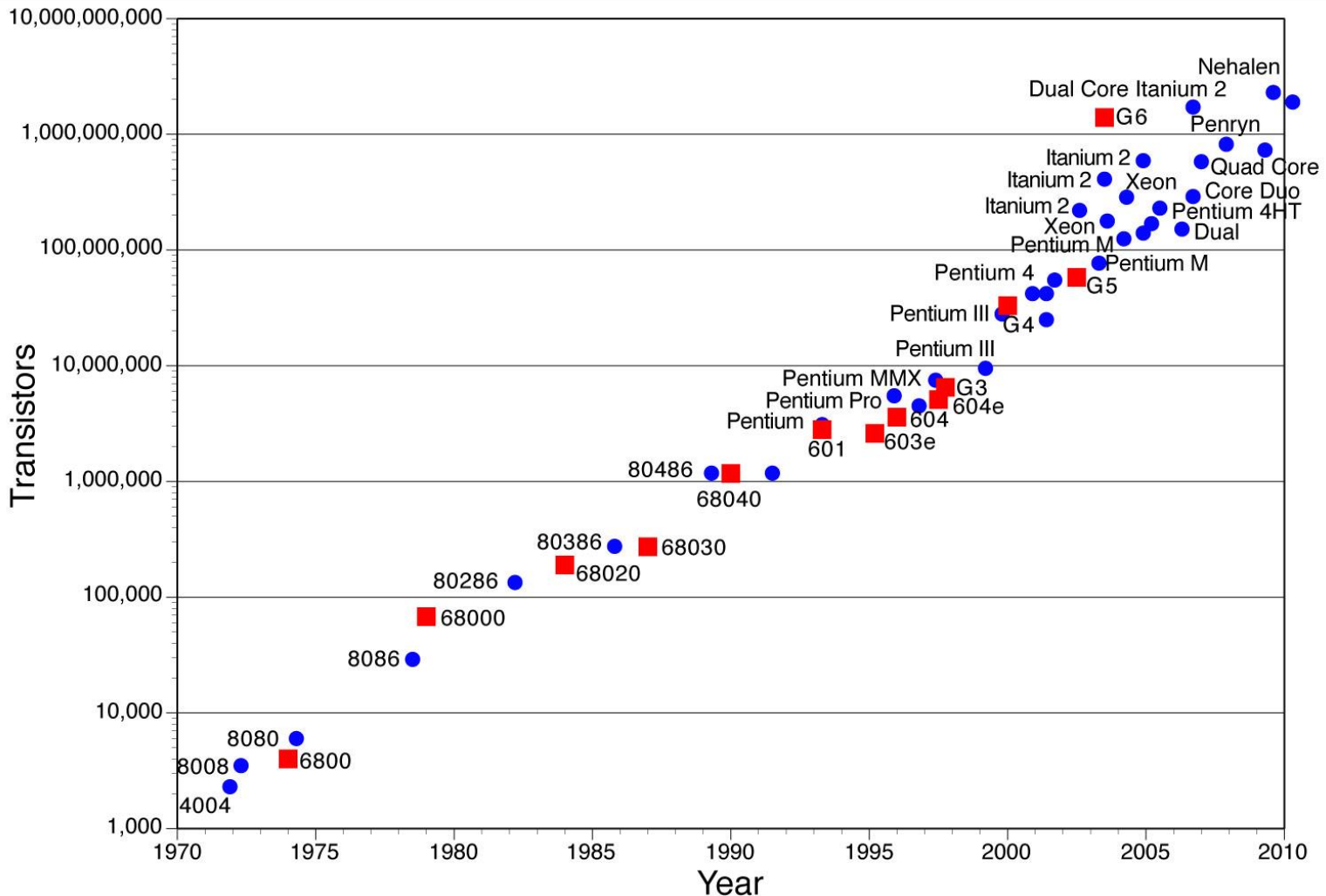


第四阶段：20世纪70年代中期到80年代中期，微电子技术的发展，使得数字技术得到迅猛的发展，产生了大规模和超大规模的集成数字芯片，应用在各行各业和我们的日常生活

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。



绪论



摩尔定律 (Moore's Law)

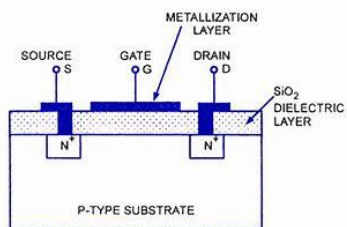
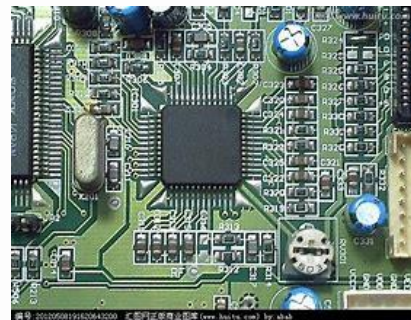
电子电路的功能？

处理信息+能量转换

- 模拟电路：用连续的模拟电压、电流值来表示信息
- 数字电路：用一个离散的电压序列来表示信息



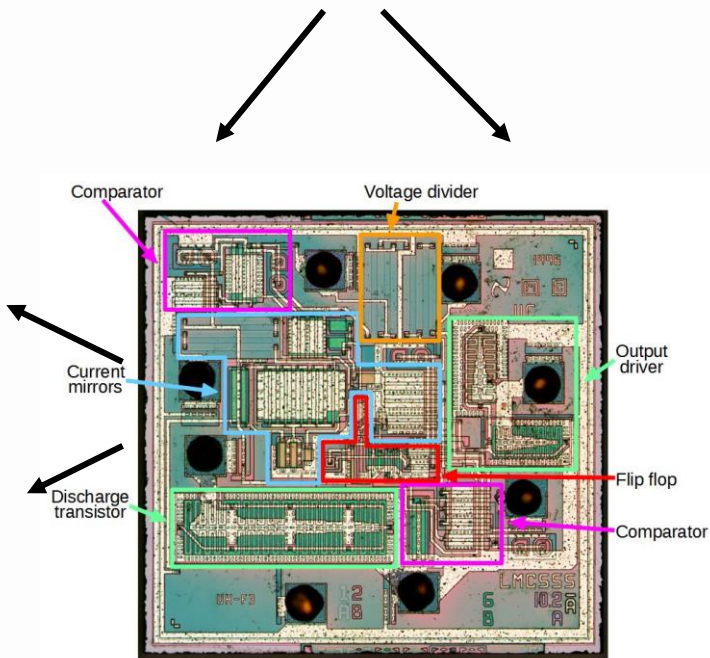
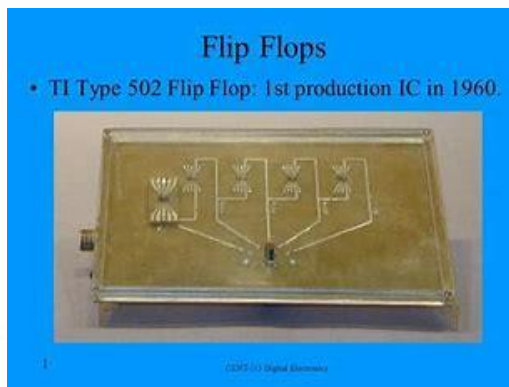
绪论



N-Channel E-MOSFET Structure

+

0111001010



结构

- 结构化设计
- 每一级有限的复杂度
- 模块复用

接口

- 系统工程的关键元素
- 良好的隔离特性
- 扩展和升级特性

如何实现一个好的系统设计？

- 最小的代价实现最大的功能
- 实现一个可靠的系统
- 未来技术的兼容性

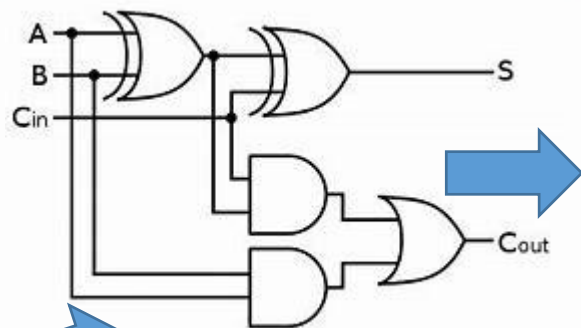


学习目的:

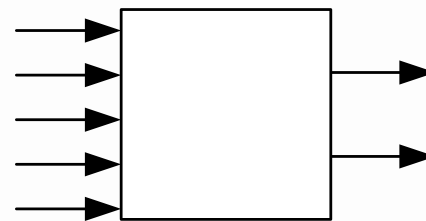
- 理解底层系统如何工作 Bottom-up
- 学习数字电路的理论基础
- 学习数字电路设计和分析的原则（抽象方法，接口等）
- 学习各个层级上设计电路的方法
 - 学科历史
 - 系统方法
 - 算法
 - 电路调试方法

绪论

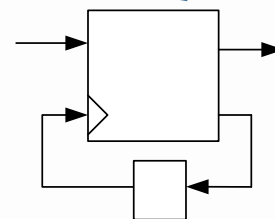
学习对象:



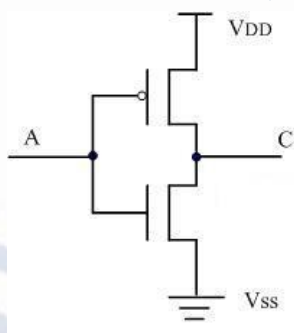
逻辑门电路



组合逻辑电路



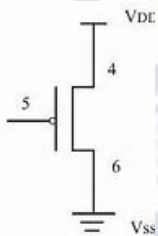
时序逻辑电路



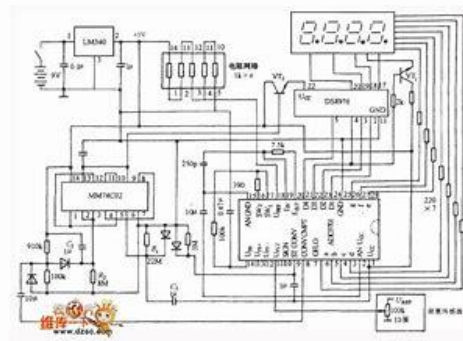
单体MOSFET



N型MOS管



P型MOS管



复杂系统

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



内容提要

本章首先介绍有关数制和码制的一些基本概念和术语，然后给出数字电路中常用的数制和编码。此外，还将具体讲述不同数制之间的转化方法和二进制数算术运算的原理和方法。



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，其遵循的原则称为数制。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。下面就介绍这几种常用的数制。



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进位的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

一、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

$$(D)_{10} = k_{n-1}k_{n-2}\cdots k_0k_{-1}\cdots k_{-m}$$
$$= k_{n-1} \times 10^{n-1} + \cdots + k_0 \times 10^0 + k_{-1} \times 10^{-1} + \cdots + k_{-m} \times 10^{-m} = \sum_{i=-m}^{n-1} k_i \times 10^i$$



$$(D)_{10} = k_{n-1}k_{n-2}\cdots k_0k_{-1}\cdots k_{-m}$$
$$= k_{n-1} \times 10^{n-1} + \cdots + k_0 \times 10^0 + k_{-1} \times 10^{-1} + \cdots + k_{-m} \times 10^{-m} = \sum_{i=-m}^{n-1} k_i \times 10^i$$

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 $0 \sim 9$ 十个数， i 取值从 $(n-1) \sim 0$ 的所有正整数到 $-1 \sim -m$ 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



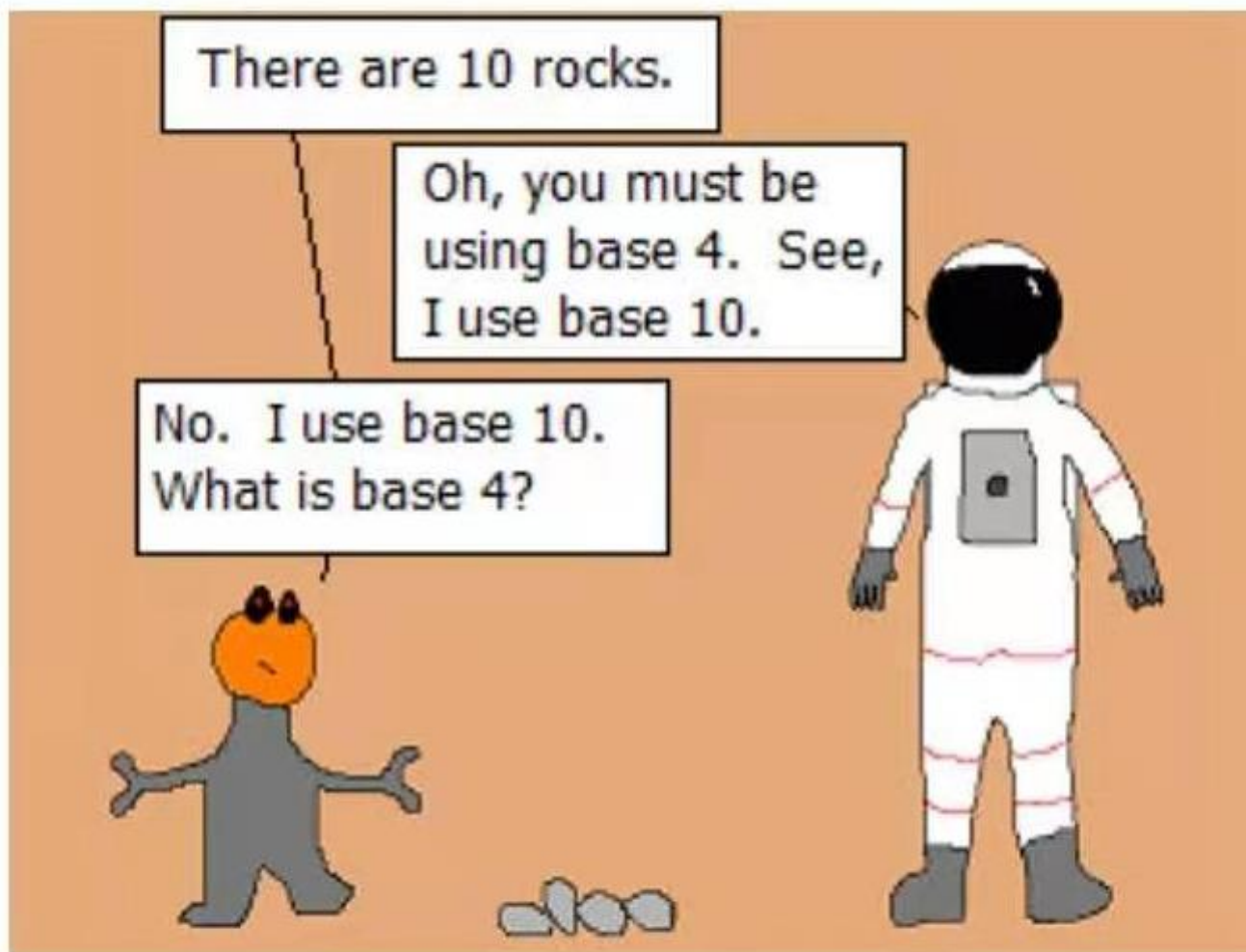
例如: $(249.56)_{10} = 2 \times 10^2 + 4 \times 10^1 + 9 \times 10^0$
 $+ 5 \times 10^{-1} + 6 \times 10^{-2}$

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

$$(D)_N = k_{n-1}k_{n-2} \cdots k_0k_{-1} \cdots k_{-m}$$
$$= k_{n-1} \times N^{n-1} + \cdots + k_0 \times N^0 + k_{-1} \times N^{-1} + \cdots + k_{-m} \times N^{-m} = \sum_{i=-m}^{n-1} k_i \times N^i$$

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



Every base is base 10.



二、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

$$(D)_2 = k_{n-1}k_{n-2}\cdots k_0k_{-1}\cdots k_{-m}$$
$$= k_{n-1} \times 2^{n-1} + \cdots + k_0 \times 2^0 + k_{-1} \times 2^{-1} + \cdots + k_{-m} \times 2^{-m} = \sum_{i=-m}^{n-1} k_i \times 2^i$$

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

$$\text{如 } (11011.101)_2 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$
$$+ 1 \times 2^{-1} + 0 \times 2^{-2}$$

$$+ 1 \times 2^{-3}$$

$$= (27.625)_{10}$$



第一章 数制和码制

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；



三、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

$$(N)_8 = k_{n-1}k_{n-2}\cdots k_0k_{-1}\cdots k_{-m}$$
$$= k_{n-1} \times 8^{n-1} + \cdots + k_0 \times 8^0 + k_{-1} \times 8^{-1} + \cdots + k_{-m} \times 8^{-m} = \sum_{i=-m}^{n-1} k_i \times 8^i$$

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



四、十六进制

进位规则是“逢十六进一”，其基数为16。任意一个n位整数、m位小数的十六进制可表示为

$$(N)_{16} = k_{n-1}k_{n-2} \cdots k_0k_{-1} \cdots k_{-m}$$
$$= k_{n-1} \times 16^{n-1} + \cdots + k_0 \times 16^0 + k_{-1} \times 16^{-1} + \cdots + k_{-m} \times 16^{-m} = \sum_{i=-m}^{n-1} k_i \times 16^i$$

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)

16ⁱ—为十六进制的权，基数为16

n、m—为正整数



$$\begin{aligned}\text{如 } (F9.1A)_{16} &= 15 \times 16^1 + 9 \times 16^0 + 1 \times 16^{-1} + 10 \times 16^{-2} \\ &= (249.1015625)_{10}\end{aligned}$$

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



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

$$\begin{aligned}(D)_N &= k_{n-1}k_{n-2} \cdots k_0k_{-1} \cdots k_{-m} \\ &= k_{n-1} \times N^{n-1} + \cdots + k_0 \times N^0 + k_{-1} \times N^{-1} + \cdots + k_{-m} \times N^{-m} = \sum_{i=-m}^{n-1} k_i \times N^i\end{aligned}$$



例如：

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

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

$$(2AF.CE)_H = 2 \times 16^2 + 10 \times 16^1 + 15 \times 16^0 + 12 \times 16^{-1} + 14 \times 16^{-2} \\ = 512 + 160 + 15 + 0.75 + 0.0546875 = (688.81)_D$$



二、十进制数转换成二进制数：

即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



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

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 $(173)_D = (10101101)_B$

2	173	1(k_0)
2	86	0(k_1)
2	43	1(k_2)
2	21	1(k_3)
2	10	0(k_4)
2	5	1(k_5)
2	2	0(k_6)
2	1	1(k_7)
	0		



b. 小数部分

由于精度要求为1%，故应该令

$$2^{-m} \leq 1\%$$

$$2^{-m} \leq 1\% = 10^{-2}$$

$$2^m \geq 100$$

取对数，可得

$$m \lg_{10} 2 \geq \lg_{10} 100 = 2$$

$$m \geq 6.6$$

取 $m=7$ 满足精度要求，
过程如下

$$\text{即 } (0.39)_D = (0.0110001)_B$$

$$\text{故 } (173.39)_D$$

$$= (10101101.0110001)_B$$

$$0.39 \times 2 = 0.78 \dots\dots 0 (k_{-1})$$

$$0.78 \times 2 = 1.56 \dots\dots 1 (k_{-2})$$

$$0.56 \times 2 = 1.12 \dots\dots 1 (k_{-3})$$

$$0.12 \times 2 = 0.24 \dots\dots 0 (k_{-4})$$

$$0.24 \times 2 = 0.48 \dots\dots 0 (k_{-5})$$

$$0.48 \times 2 = 0.96 \dots\dots 0 (k_{-6})$$

$$0.96 \times 2 = 1.92 \dots\dots 1 (k_{-7})$$



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：

$$\begin{aligned}(1011110.1011001)_B &= (001 \ 011 \ 110.101 \ 100 \ 100)_2 \\ &= (136.544)_O\end{aligned}$$

$$\begin{aligned}(1011110.1011001)_B &= (0101 \ 1110.1011 \ 0010)_2 \\ &= (5E.B2)_H\end{aligned}$$

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

解：

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

$$(9F12.04A)_H = (1001111100010010.00000100101)_B$$



☺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

$$\begin{aligned}(87)_D &= (1010111)_B = (001\ 010\ 111)_B \\ &= (0101\ 0111)_B = (127)_D \\ &= (57)_H\end{aligned}$$

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

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

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

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

$$\begin{array}{r} 1.11\ldots \\ 0101 \overline{)1001} \\ \underline{0101} \\ 1000 \\ \underline{0101} \\ 0110 \\ \underline{0101} \\ 0010 \end{array}$$



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、补码：

-2^{N-1}	2^{N-2}	2^3	2^2	2^1	2^0
------------	-----------	-----	-----	-------	-------	-------	-------



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

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。



第一章 数制和码制

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。

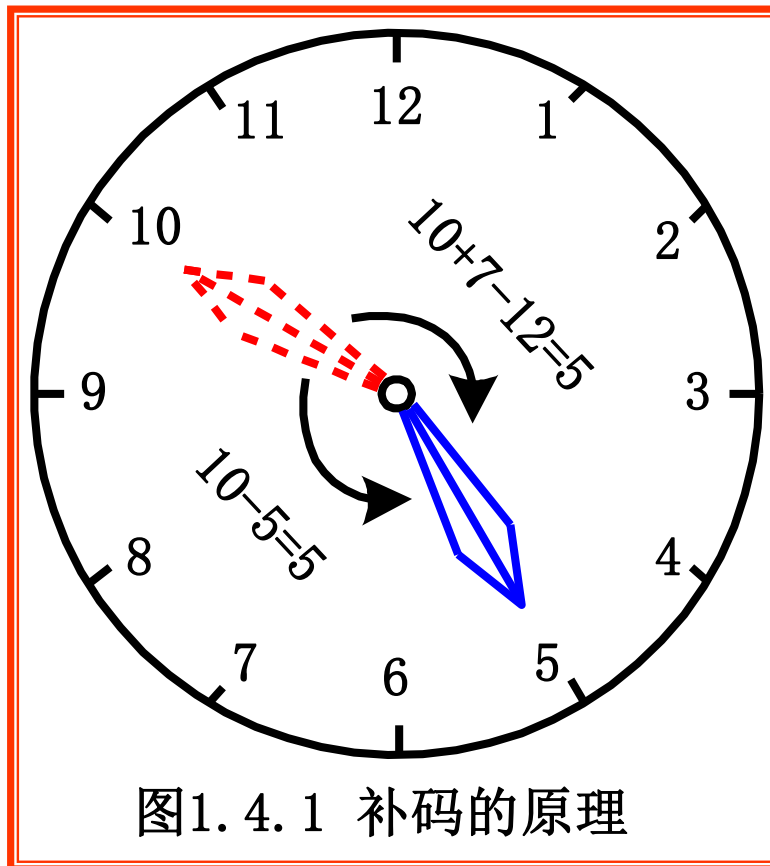


图1.4.1 补码的原理

第一章 数制和码制

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

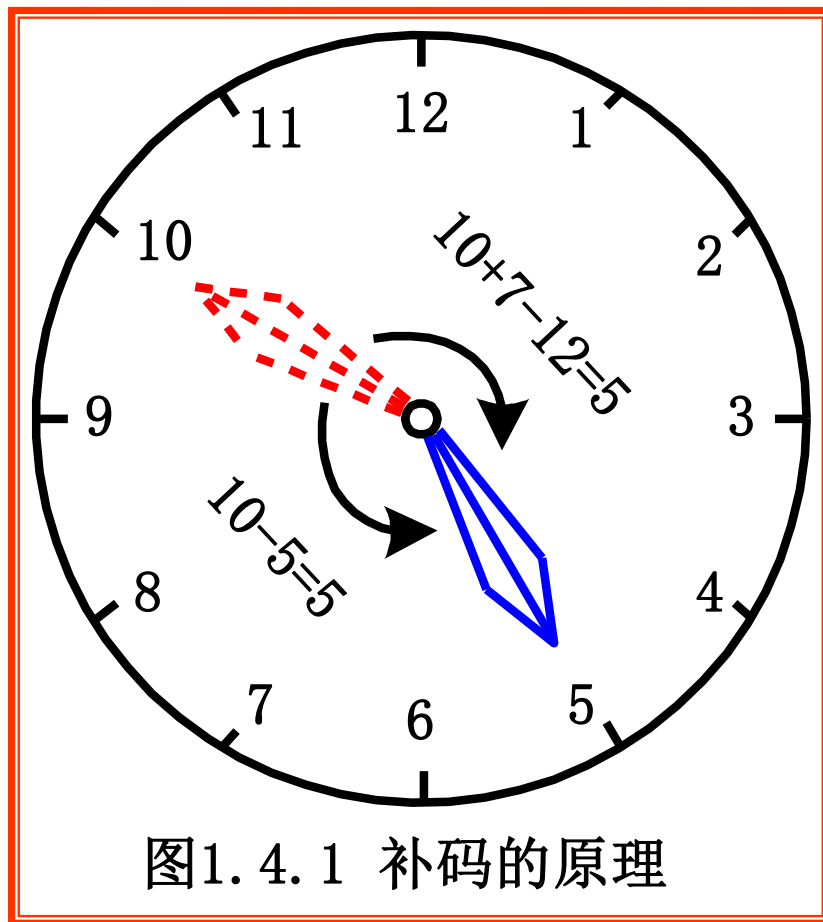


图1.4.1 补码的原理



三、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



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. 正数的补码是它所表示的数的真值，负数的补码部分不是它所示的数的真值。
 3. 与原码和反码不同，“0”的补码只有一个，即
(00000000)_B
 4. 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1100100。



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，但对于二进制数来说，先减“1”取反和先取反再加“1”的结果是一样的。故由负数的补码求原码就是数值位取反加“1”。


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 $(-2^{n-1} \sim 2^{n-1}-1)$ ，如 $n=8$ ，则可表示 $(-128 \sim 127)$ ，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  $\left\{ \begin{array}{l} (+75)_D = (01001011)_B \\ (+28)_D = (00011100)_B \\ (-75)_D = (11001011)_B \\ (-28)_D = (10011100)_B \end{array} \right.$

补码  $\left\{ \begin{array}{l} (-75)_D = (10110101)_B ; \\ (-28)_D = (11100100)_B ; \end{array} \right.$

$$\begin{array}{r} 75 \\ + 28 \\ \hline 103 \end{array} \quad \xrightarrow{\text{red arrow}} \quad \begin{array}{r} 01001011 \\ + 00011100 \\ \hline 01100111 \end{array}$$



第一章 数制和码制

$$\begin{array}{r} 75 \\ - 28 \\ \hline 47 \end{array} \quad \longrightarrow \quad \begin{array}{r} 01001011 \\ + 11100100 \\ \hline 10010111 \end{array}$$

溢出

$$\begin{array}{r} -75 \\ + 28 \\ \hline -47 \end{array} \quad \longrightarrow \quad \begin{array}{r} 10110101 \\ + 00011100 \\ \hline 11010001 \end{array}$$

补码

$$\begin{array}{r} -75 \\ - 28 \\ \hline -103 \end{array} \quad \longrightarrow \quad \begin{array}{r} 10110101 \\ + 11100100 \\ \hline 11001001 \end{array}$$

溢出

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

几种常用的十进制代码

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: 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

$$(1001)_2 = 1 \times 2^3 + 1 \times 2^0 = 9$$

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。



3. 2421码是有权码，其每位的权为2、4、2、1，如 $(1100)_2 = 1 \times 2 + 1 \times 4 = 6$ ，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 $(0111)_2 = 1 \times 2 + 1 \times 1 + 1 \times 1 = 4$ ，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。

二、格雷码

特点：1. 每一位的状态变化都按一定的顺序循环。

2. 编码顺序依次变化，按表中顺序变化时，相邻代码

只有一位改变状态。

应用：减少过渡噪声

编码顺序	二进制	格雷码	编码顺序	二进制码	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



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

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000011111110000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。



三、美国信息交换标准代码 (ASC II)

ASC II 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)