



《电子技术基础 数字部分》

第一章 数字逻辑概论

什么是数字电路？

- 就是用数字电子开关表示二进制的0和1，从而来实现**算术运算**和**逻辑运算**，并最终实现信息的存储、处理和传输。

数字电路的学习方法

- 掌握逻辑代数的基本原理（理论基础）
- 重视逻辑关系及其描述方法（思维方式）
- 熟悉逻辑电路的基本分析方法与设计方法（基本训练）
- 掌握器件的使用（实践能力）

□ 作业要求

- 抄题号，建议抄题
- 统一使用**B5活页纸**，自己装订成本，学期末检查
- 在规定时间内提交（**周二交，两周一次**）
- 每次只交当次作业，**用订书机或胶带纸固定**，每一页写**学号、姓名**

□ 自主学习

- Quartus II 9.1
- Verilog HDL基础

□ 学习交流

- QQ群：519817329 Email: jujube_yang@seu.edu.cn



□ 考核形式

- 平时（10%）+期中考试（30%）+期末考试（60%）
- 考试形式：闭卷

□ 参考书目

- 1、《数字电子技术基础》，阎石主编，清华大学
- 2、《计算机结构与逻辑设计》，黄正瑾主编，东南大学
- 3、《数字电子技术》[Digital Fundamentals], Thomas L. Floyd 著
- 4、搭建你的数字积木-数字电路与逻辑设计（Verilog HDL&Vivado 版），汤勇明等编著，清华大学出版社
- 5、线上资源：计算机结构与逻辑设计 东南大学 中国大学MOOC(慕课)
(icourse163.org)

课程安排

- 数字逻辑概论
 - 1.1、1.2、1.3、1.4、1.5
- 逻辑代数
 - 2.1、2.2、2.3、2.4
- 逻辑门电路
 - 3.1、3.2、3.3、3.7
- 组合逻辑电路
 - 4.1、4.2、4.3、4.4、4.5
- 锁存器和触发器
 - 5.1、5.2、5.3、5.4、5.5

课程安排

- **时序逻辑电路**
 - 6.1、6.2、6.3、6.4、6.5、6.6
- **半导体存储器**
 - 7.1、7.2
- **CPLD和FPGA**
 - 8.1、8.2、8.3
- **脉冲波形的变换与产生**
 - 9.1、9.2
- **数模与模数转换器**
 - 10.1、10.2
- **硬件描述语言基础**

第一章 数字逻辑概论

1.1 数字信号与数字电路

1.2 数制

1.3 二进制数的算术运算

1.4 二进制代码

1.5 二值逻辑变量与基本逻辑运算

1.1 数字信号与数字电路

1.1.1 数字技术的发展及其应用

1.1.2 数字集成电路的分类及特点

1.1.3 模拟信号和数字信号

1.1.4 数字信号的描述方法

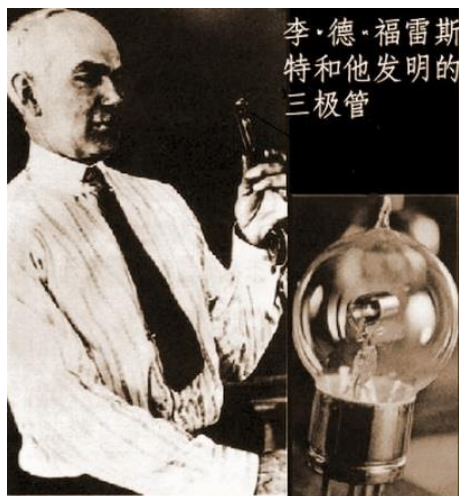
1.1.1 数字技术的发展及其应用

电子技术的发展特点:以电子器件的发展为基础

1906	李德. 福莱斯特	真空管（电子管）
1947	威廉. 肖克利	晶体管
1960-		半导体集成电路

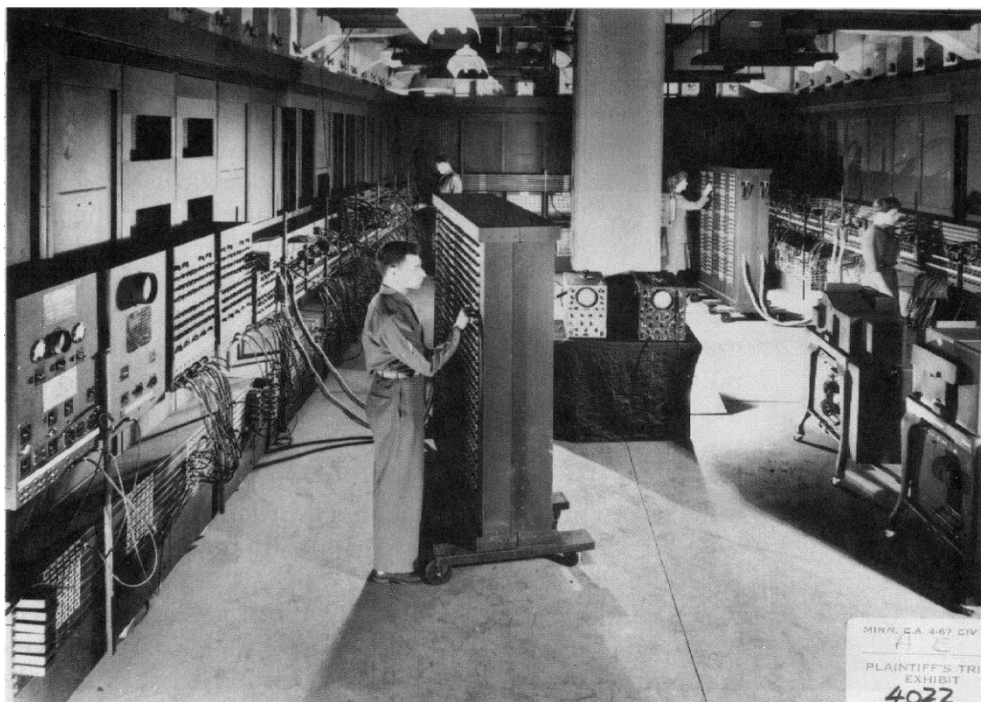
□ 真空管

1906年，**福莱斯特**等发明了电子管；电子管体积大、重量重、耗电大、寿命短。目前在一些大功率发射装置中使用。



可以说它是有三个电极（阴极，栅极和阳极）的灯泡，参与工作的电极被封装在一个真空的容器内。

□ 世界第一台通用电子计算机--ENIAC



1946年2月14日在美国宾夕法尼亚大学的莫尔电机学院诞生，由18,800多个电子管组成，重量30多吨，占地面积170多平方米。

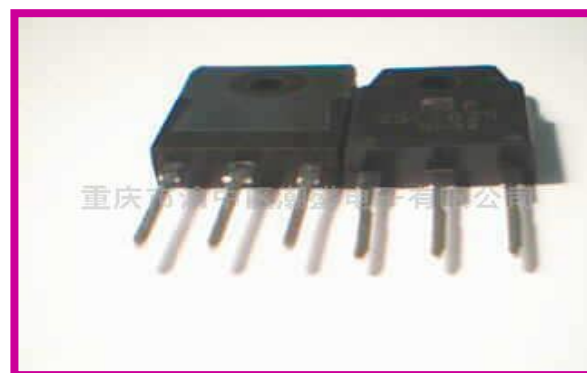


大小：长30.48m，宽6m，高2.5m；速度：5000次/sec

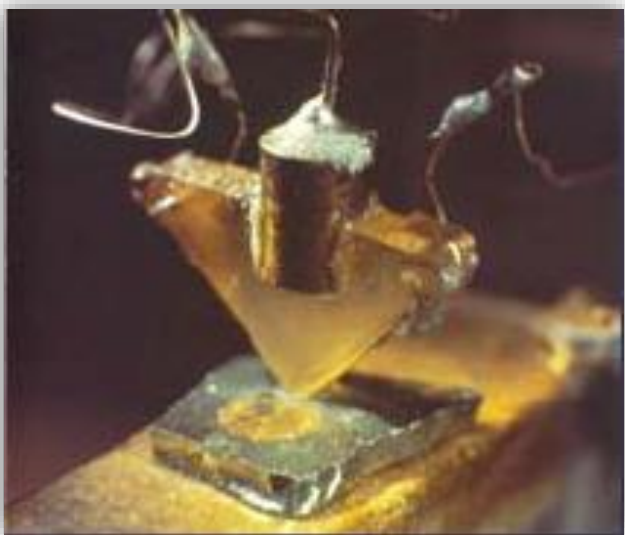
功率：150KW；平均无故障运行时间：7min

□ 晶体管

半导体二极管、三极管、场效应晶体管（主要指MOS管）



□ 第一个晶体管 (1947年12月16日)



- ❖ 威廉 肖克利, 沃特 布拉顿 & 约翰 巴顿(贝尔实验室)
- ❖ 半导体晶体进行声音信号放大实验 锗材料
- ❖ 1956年 诺贝尔物理学奖 (献给世界的圣诞节礼物)

电子管统治了20世纪的前半部分:

体积大、价格昂贵、功耗大、稳定性差

现在, 这一切改变了!



晶体管与电子管的比较

- ◆ 体积比电子管小很多
- ◆ 耗电大大降低
- ◆ 稳定性有很大提高



电子管实物

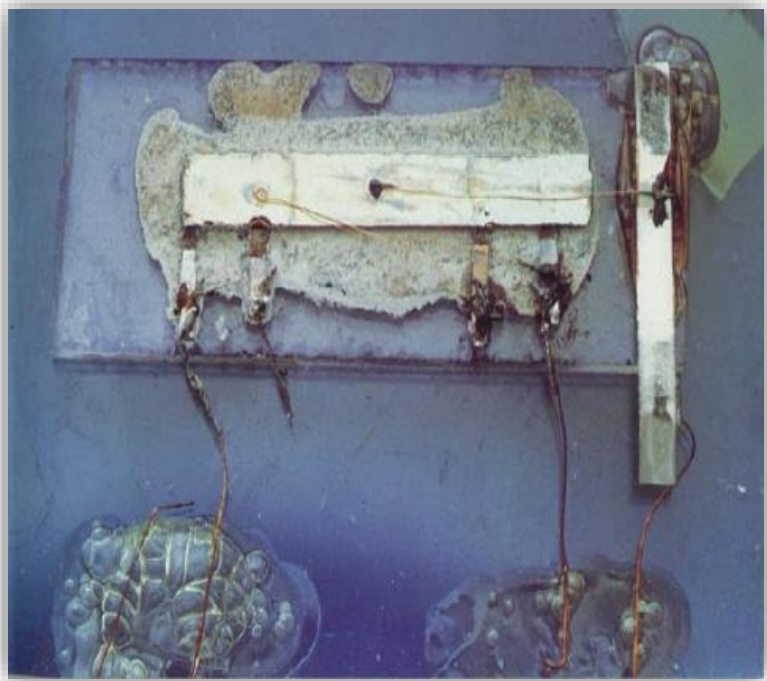


晶体管实物



1955年，贝尔实验室使用800只晶体管组装了世界上第一台晶体管计算机TRADIC。
(Transistor Digital Computer)

□ 第一块集成电路 (1958)

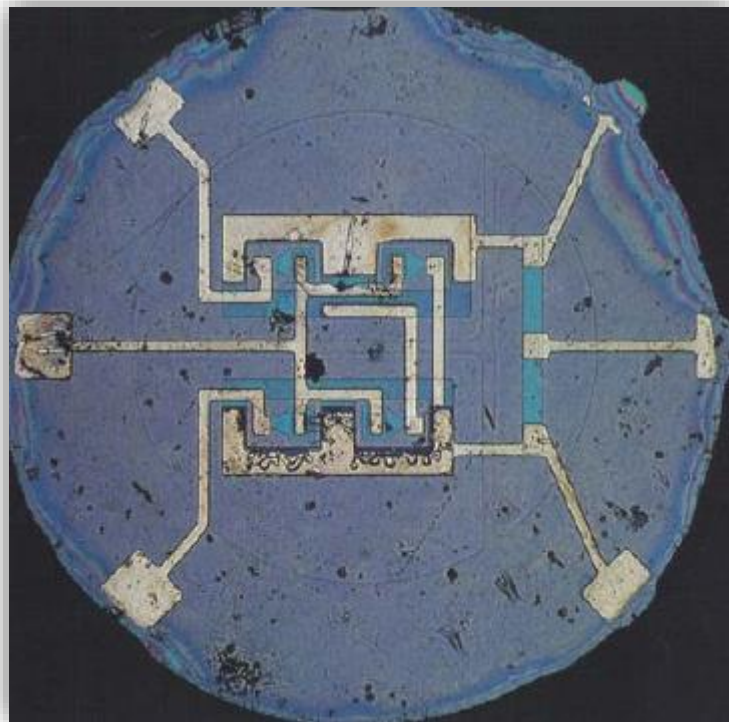


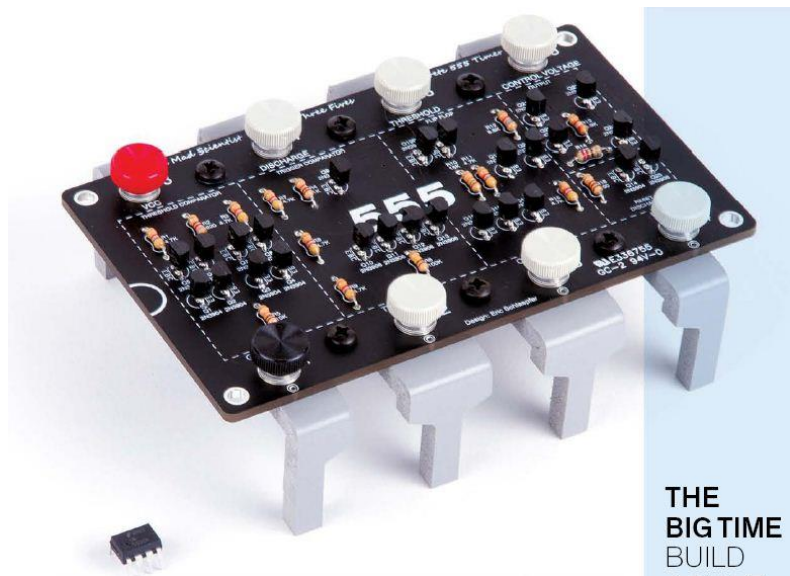
- ❖ 杰克 基尔比(TI)
- ❖ 5个器件
- ❖ 锗材料
- ❖ 2000年 诺贝尔物理学奖

采用一定的工艺，把一个电路中所需的晶体管、电阻、电容和电感等元件及布线互连一起，制作在一小块或几小块半导体晶片或介质基片上，然后封装在一个管壳内，成为具有所需电路功能的**微型结构**；其中所有元件在结构上已组成一个整体，它在电路中用字母“IC”表示。

□ 第一块商用集成电路 (1961)

- ❖ Fairchild (仙童)
- ❖ 1bit存储器
- ❖ 4个晶体管和5个电阻
- ❖ 小规模集成电路的时代开始了
- ❖ Fairchild被认为是硅谷人才摇篮





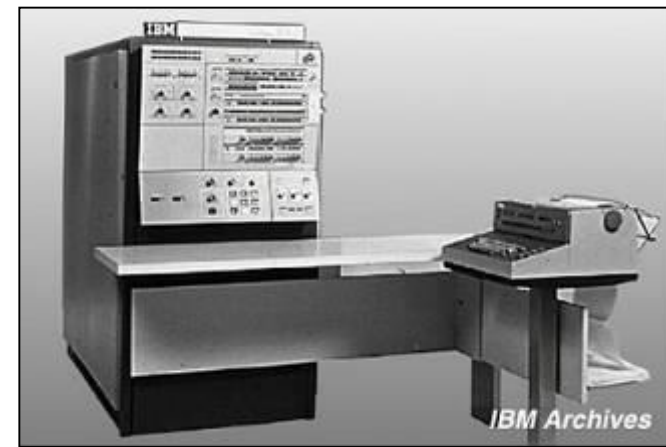
集成电路与晶体管的比较：



晶体管实物



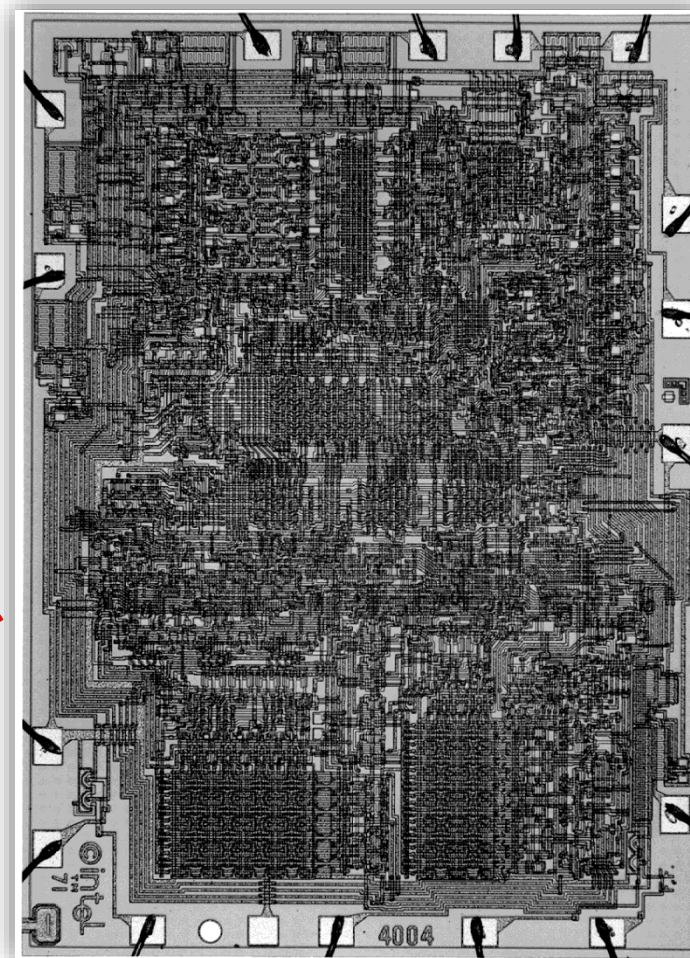
集成电路实物



1964年4月7日，在IBM公司成立50周年之际，由年仅40岁的吉恩·阿姆达尔 (Gene Amdahl)担任主设计师，历时四年研发的IBM360计算机问世，标志着第三代计算机的全面登场，这也是IBM历史上最为成功的机型之一。

□ 微处理器诞生 (1971)

- Intel 4004
- 2300个晶体管
- 第一个单芯片计算机
- 标志着大规模集成电路时代的开始
- 10 μ m工艺
- 4位数据位宽
- 108KHz主频



□ 半导体集成电路

60~70代- IC技术迅速发展：SSI、MSI、LSI、VLSI。

10万个晶体管/片。

80年代后- ULSI，10亿个晶体管/片、ASIC制作技术成熟。

90年代后- 97年一片集成电路上有40亿个晶体管，目前上百亿了

目前- 芯片内部的布线细微到纳米量级(90,45,22,14,8,7,5,3,1nm)

将来- 高分子材料或生物材料制成密度更高、三维结构的电路。

□ 数字技术的应用



1.1.2 数字集成电路的分类及特点

□ 根据电路的**结构特点**及其对输入信号的**响应规则**的不同

——**组合逻辑电路**和**时序逻辑电路**

□ 根据电路的**形式**不同

——**集成电路**和**分立电路**

□ 根据**器件**不同

——**TTL** 和 **CMOS**电路

□ 根据**集成度**不同

——**小规模**、**中规模**、**大规模**、**超大规模**和**甚大规模**

集成度：每一芯片所包含的晶体管个数

分类	晶体管的个数	典型集成电路
小规模SSI	小于100个	逻辑门、触发器
中规模MSI	$10^2 \sim 10^3$	计数器、加法器
大规模LSI	$10^3 \sim 10^5$	小型存储器、门阵列
超大规模 VLSI	$10^5 \sim 10^6$	大型存储器、微处理器
甚大规模 ULSI	10^6 以上	可编程逻辑器件、多功能专用集成电路

□ 数字集成电路的特点

- ◆ 可靠性、稳定性和精度高, 抗干扰能力强
- ◆ 易于设计
- ◆ 体积小, 通用性好, 成本低, 电路简单, 便于大规模集成
- ◆ 具可编程性, 可实现硬件设计软件化
- ◆ 高速度、低功耗
- ◆ 加密性好, 便于存储、传输和处理

□ 数字电路的分析、设计与测试

(1) 数字电路的分析方法

数字电路的分析:根据电路确定电路输出与输入之间的逻辑关系。

分析工具: 逻辑代数。

电路逻辑功能主要用真值表、功能表、逻辑表达式和波形图表示。

(2) 数字电路的设计方法

数字电路的设计:从给定的逻辑功能要求出发, 选择适当的逻辑器件, 设计出符合要求的逻辑电路。

设计方式:分为传统的设计方式和基于EDA软件的设计方式。

1.1.3 模拟信号和数字信号

□ 自然界物理量的表现形式

◆ 连续： 在一定的范围内有无穷多个取值可能

——无法用数字准确表示

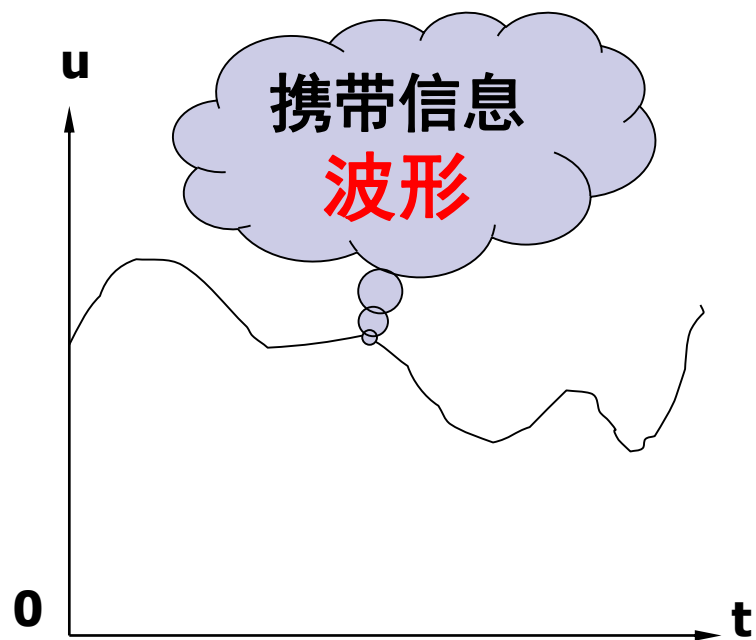
◆ 离散： 在一定的范围内只有某些特定取值

——可与数字相对应

□ 电子系统处理物理量的方法

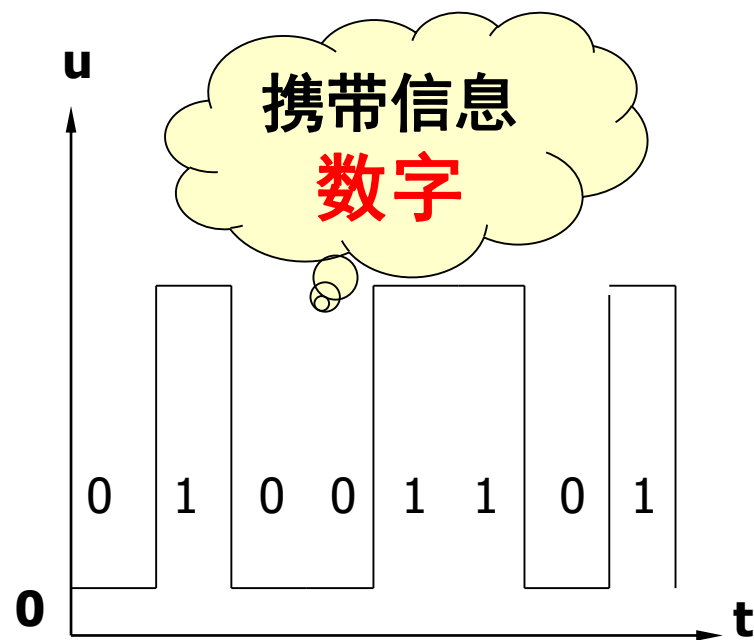
用某个电参量（电压、电流、频率、相位等）去描述

——电信号



模拟信号

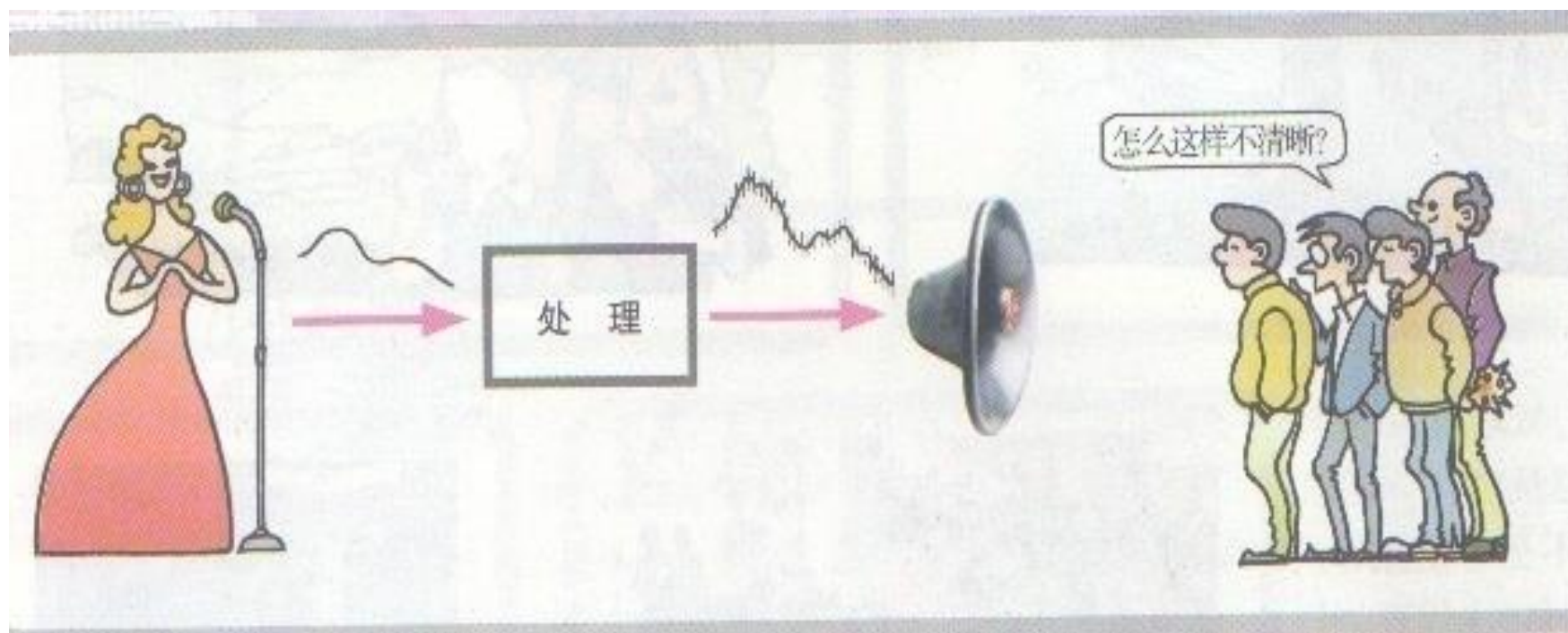
时间上和数值上均连续变化的信号



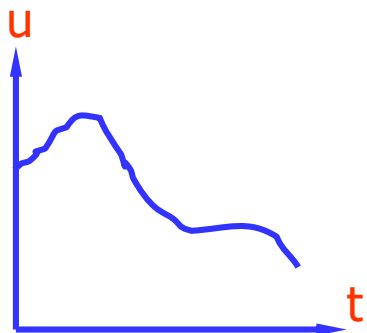
数字信号

时间上和数值上均是离散的信号

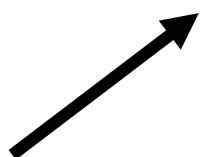
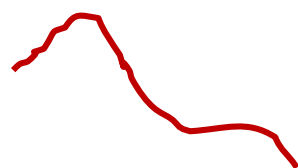
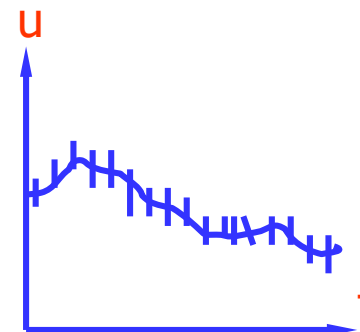
数字电路的优点



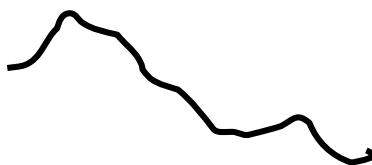
信息的载体——信号波形



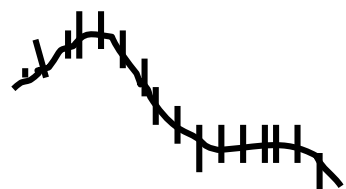
模拟通道



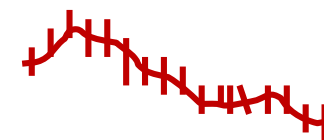
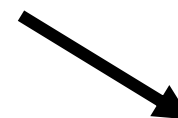
衰减



畸变



干扰



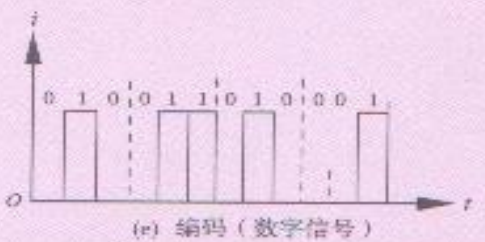
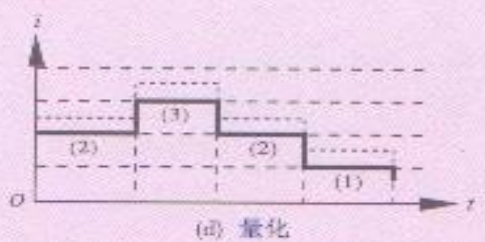
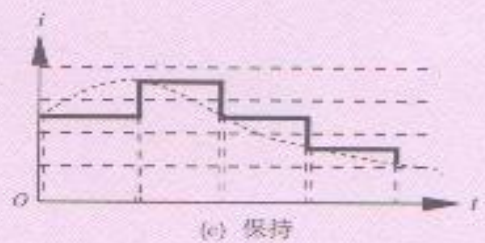
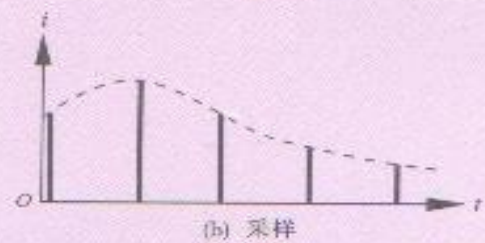
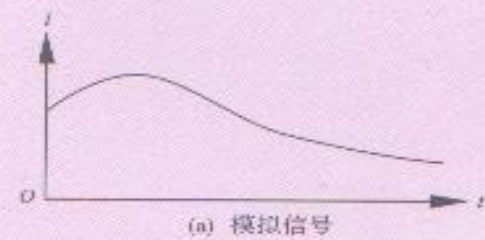
信息载体——数字

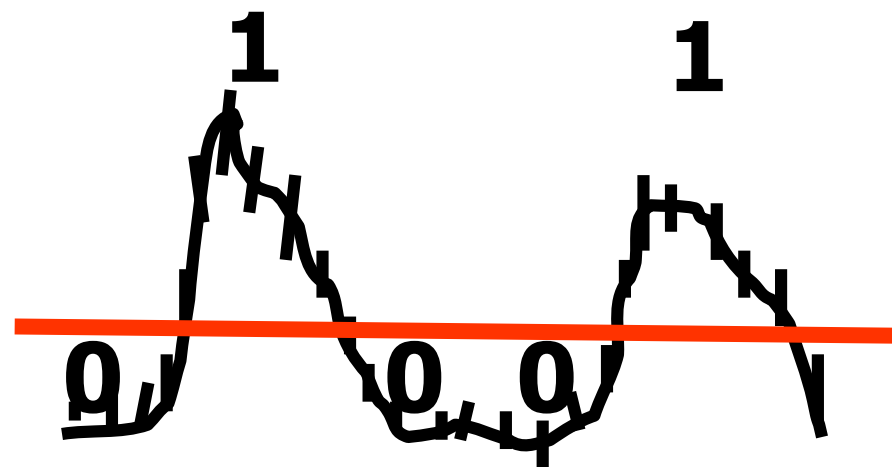
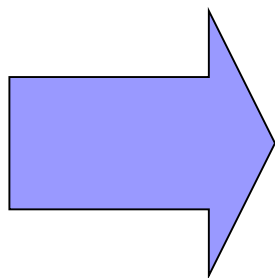
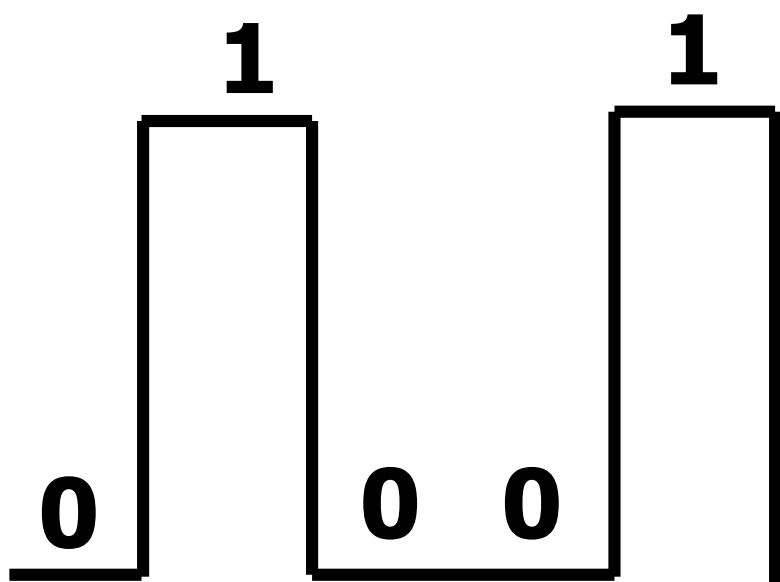
采样

保持

量化

编码





◆ 只要0和1不混淆，信息就不会丢失

1.1.4 数字信号的描述方法

□ 二值数字逻辑和逻辑电平

◆ 二值数字逻辑

0、1---表示数量时称**二进制数**

---表示事物状态时称**二值逻辑**

◆ 表示方式

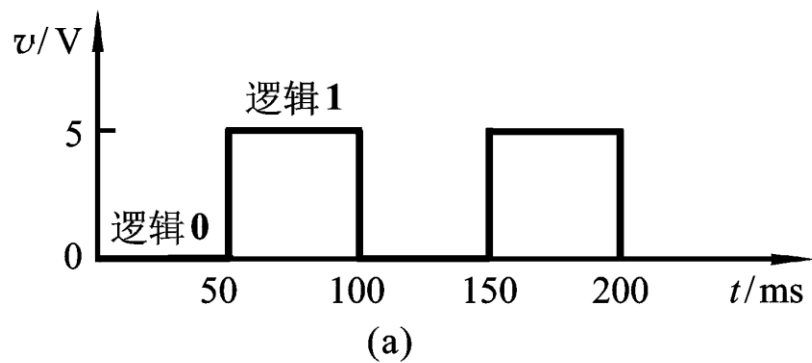
在电路图中用**低、高电平**表示**0、1**两种逻辑状态

逻辑电平与电压值的关系（正逻辑）

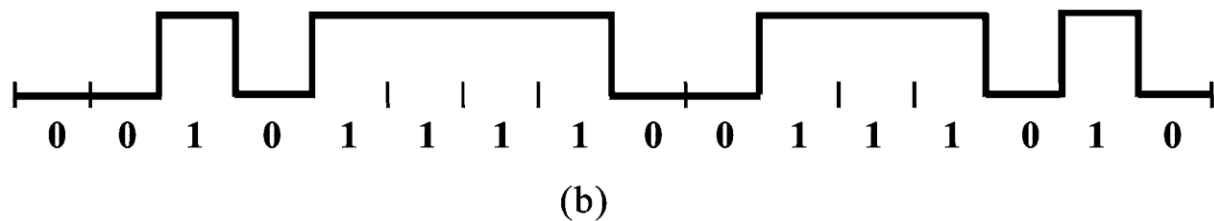
电压（V）	二值逻辑	电平
+5	1	H（高电平）
0	0	L（低电平）

□ 数字波形

是信号逻辑电平对时间的图形表示



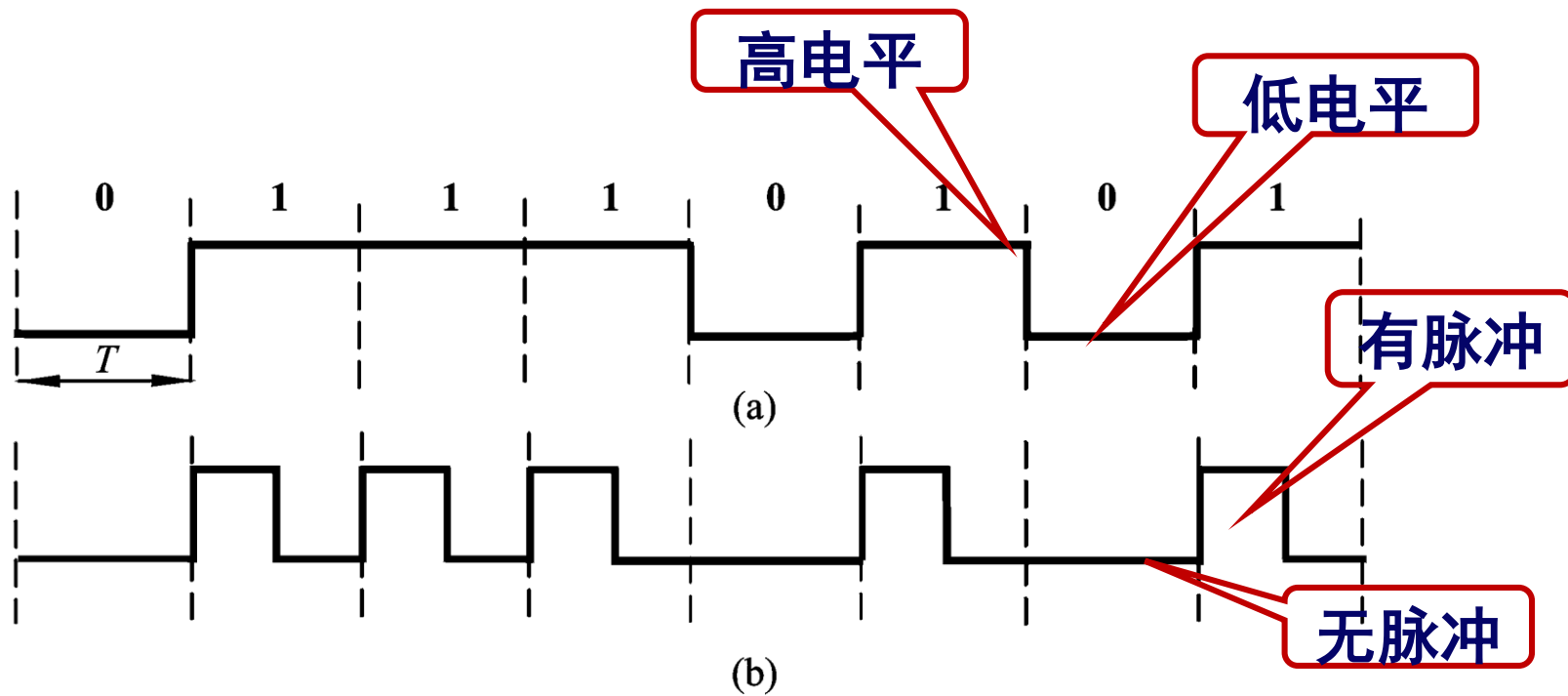
用逻辑电平描述的数字波形



16位数据的图形表示

□ 数字波形的两种类型

非归零型



归零型

比特率 ————— 每秒钟传输数据的位数

Kbps, Mbps, Gbps的含义?

例1.1.1 某通信系统每秒钟传输1544000位(1.544兆位)数据，求每位数据的时间。

解：按题意，每位数据的时间为：

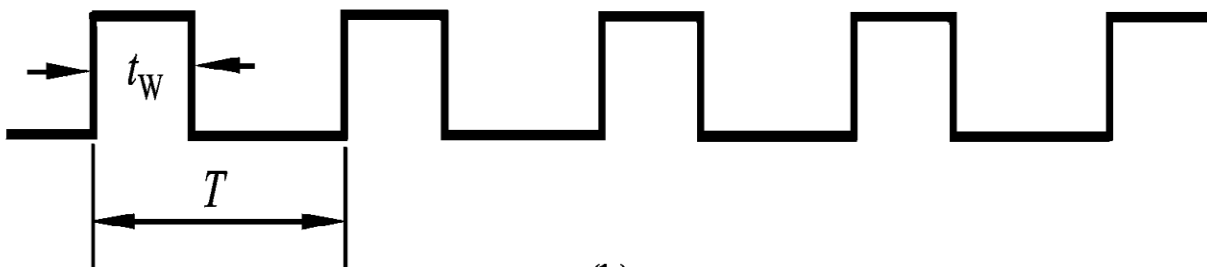
$$\left[\frac{1.544 \times 10^6}{1\text{s}} \right]^{-1} = 647.67 \times 10^{-9} \text{s} = 648 \text{ns}$$

□ 周期性和非周期性



(a)

非周期性数字波形



(b)

周期性数字波形

t_w : 脉冲宽度

T : 周期

占空比 $q(\%) = \frac{t_w}{T} \times 100\%$

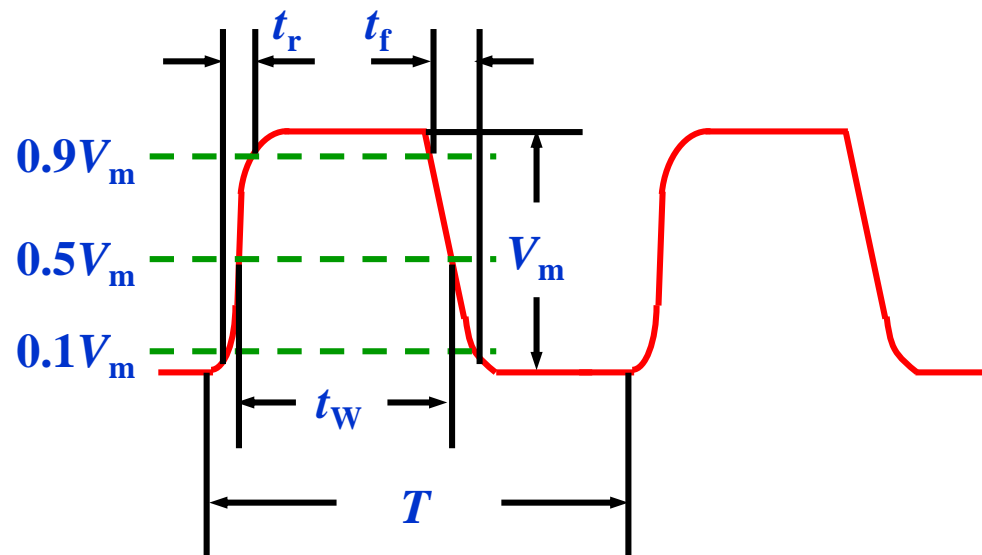
例1.1.2 设周期性数字波形的高电平持续6ms，低电平持续10ms，求占空比q。

解：因数字波形的脉冲宽度 $t_w=6\text{ms}$ ，周期 $T=6\text{ms}+10\text{ms}=16\text{ms}$ 。

$$q = \frac{6\text{ms}}{16\text{ms}} \times 100\% = 37.5\%$$

□ 实际脉冲波形及主要参数

非理想脉冲波形



周期 (T) ----表示两个相邻脉冲之间的时间间隔

脉冲宽度 (t_w) ---- 脉冲幅值的50%的两个时间所跨越的时间

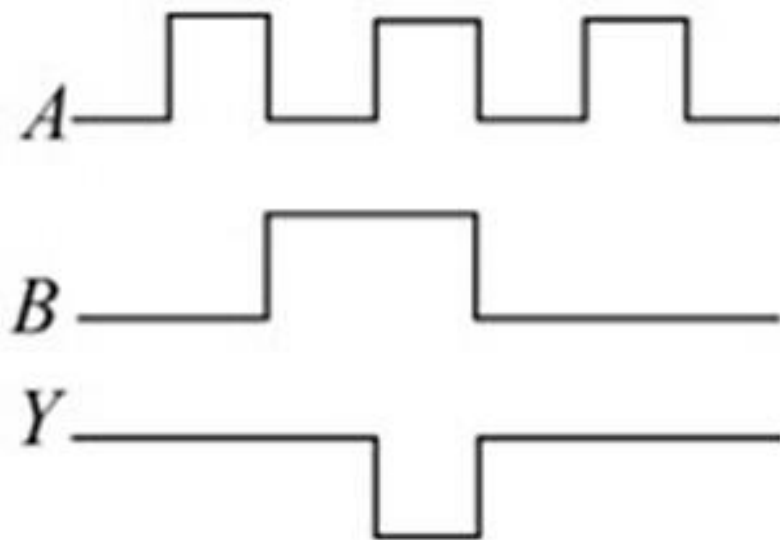
占空比 Q ----- 表示脉冲宽度占整个周期的百分比

上升时间 t_r 和下降时间 t_f ----从脉冲幅值的10%到90% 上升下降所经历的时间(典型值ns)

□ 时序图、波形图或定时图

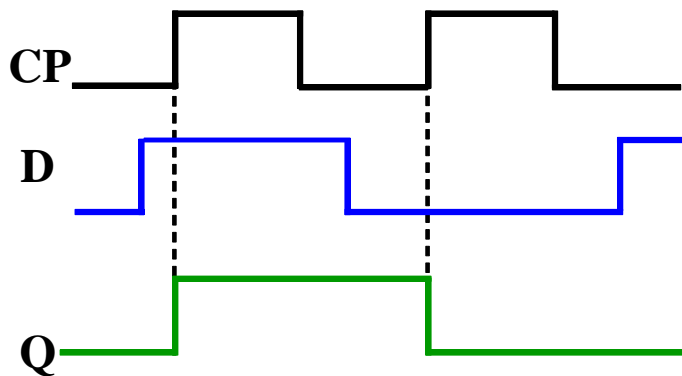
◆ 波形图

输入变量的每一种取值与相应的输出值按照时间顺序依次排列得到的图形。



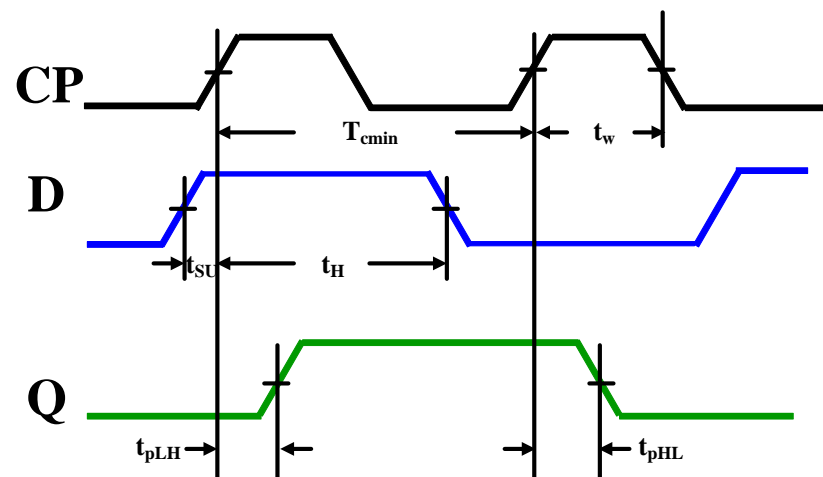
◆ 时序图或定时图

在时序电路中，电路的状态和输出对时钟脉冲序列和输入信号响应的波形图。



时序图

侧重于描述电路逻辑功能



定时图

侧重于各个信号的先后顺序及时间量

第一章 数字逻辑概论

1.1 数字信号与数字电路

1.2 数制

1.3 二进制数的算术运算

1.4 二进制代码

1.5 二值逻辑变量与基本逻辑运算

1.2 数制

1.2.1 十进制

1.2.2 二进制

1.2.3 十-二进制之间的转换

1.2.4 十六进制和八进制

1.2.1 十进制(Decimal)

□ 以十为基数的记数体制，用十个数码表示：

1、2、3、4、5、6、7、8、9、0

□ 遵循逢十进一、借一当十的计数规律

$$4587.29 = 4 \times 10^3 + 5 \times 10^2 + 8 \times 10^1 + 7 \times 10^0 + 2 \times 10^{-1} + 9 \times 10^{-2}$$

□ 十进制数的一般表达式：

$$(N)_D = \sum_{i=-\infty}^{\infty} K_i \times 10^i$$

系数 位权

□ 任意进制数的一般表达式：

$$(N)_r = \sum_{i=-\infty}^{\infty} K_i \times r^i$$

1.2.2 二进制(Binary)

□ 以二为基数的记数体制，用两个数码表示：**0、1**

□ 遵循逢**二进一**，**借一当二**的规律

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ 1\ 0\ 1\ 1\ 11 \\ +\ 1\ 0\ 0\ 1\ 9 \\ \hline 1\ 0\ 1\ 0\ 0\ 20 \end{array}$$

$$\begin{array}{r} 1\ 0\ 1 \\ 1\ 0\ 1\ 0\ 10 \\ -\ 0\ 1\ 0\ 1\ 5 \\ \hline 0\ 1\ 0\ 1\ 5 \end{array}$$

□ 二进制数的一般表达式为:

$$(N)_B = \sum_{i=-\infty}^{\infty} K_i \times 2^i$$

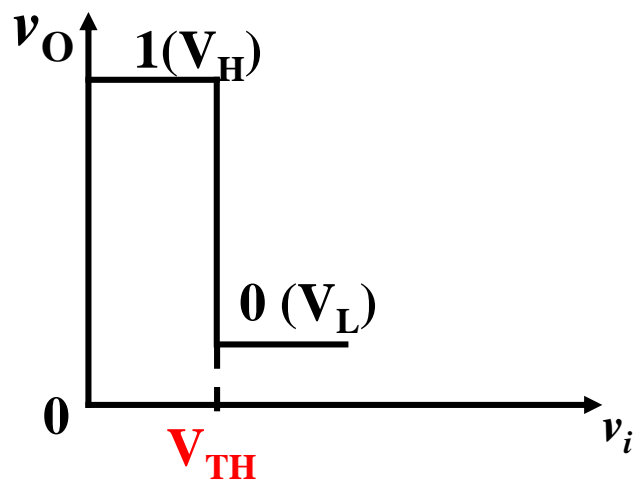
系数

位权

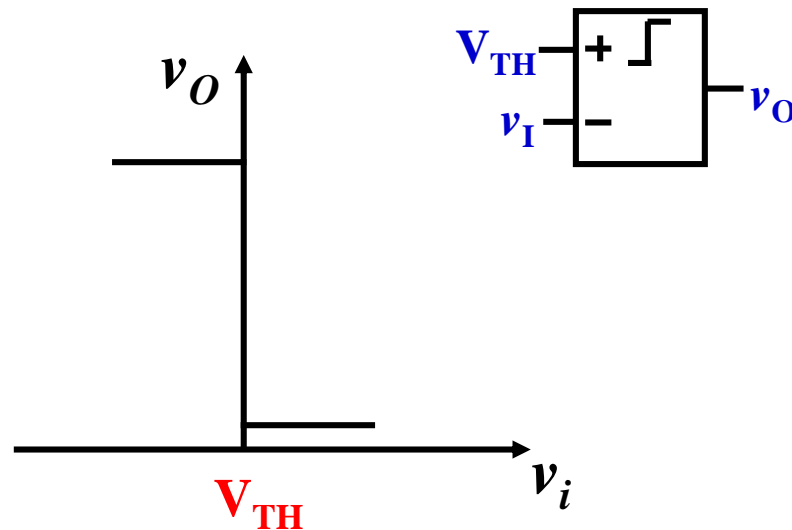
□ 二进制的特点



用电路的两个状态---有（1）和无（0）来表示二进制数，数码的产生，存储和传输简单、可靠。



电子器件典型传输特性

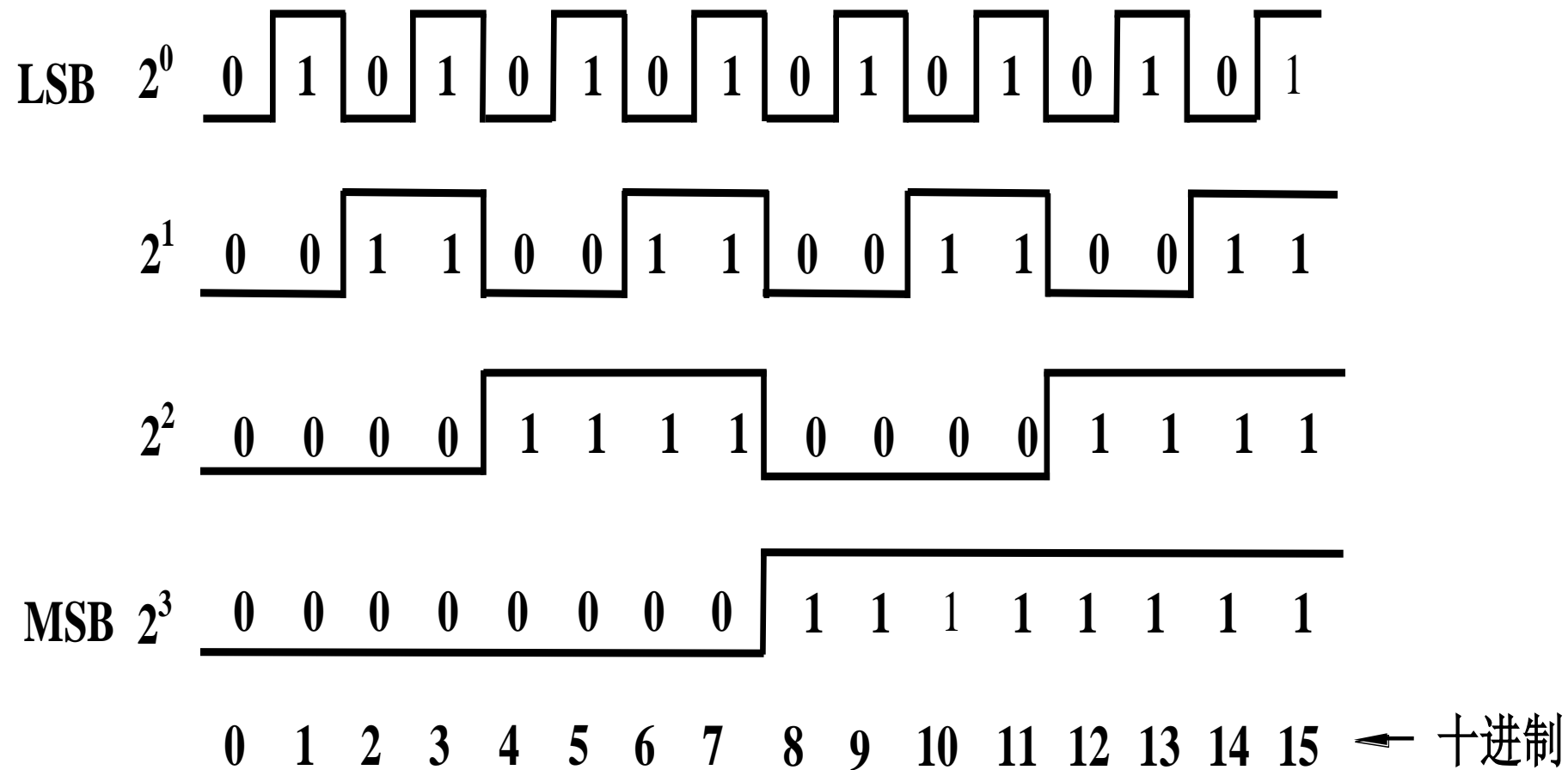


电压比较器

□ 二进制的特点

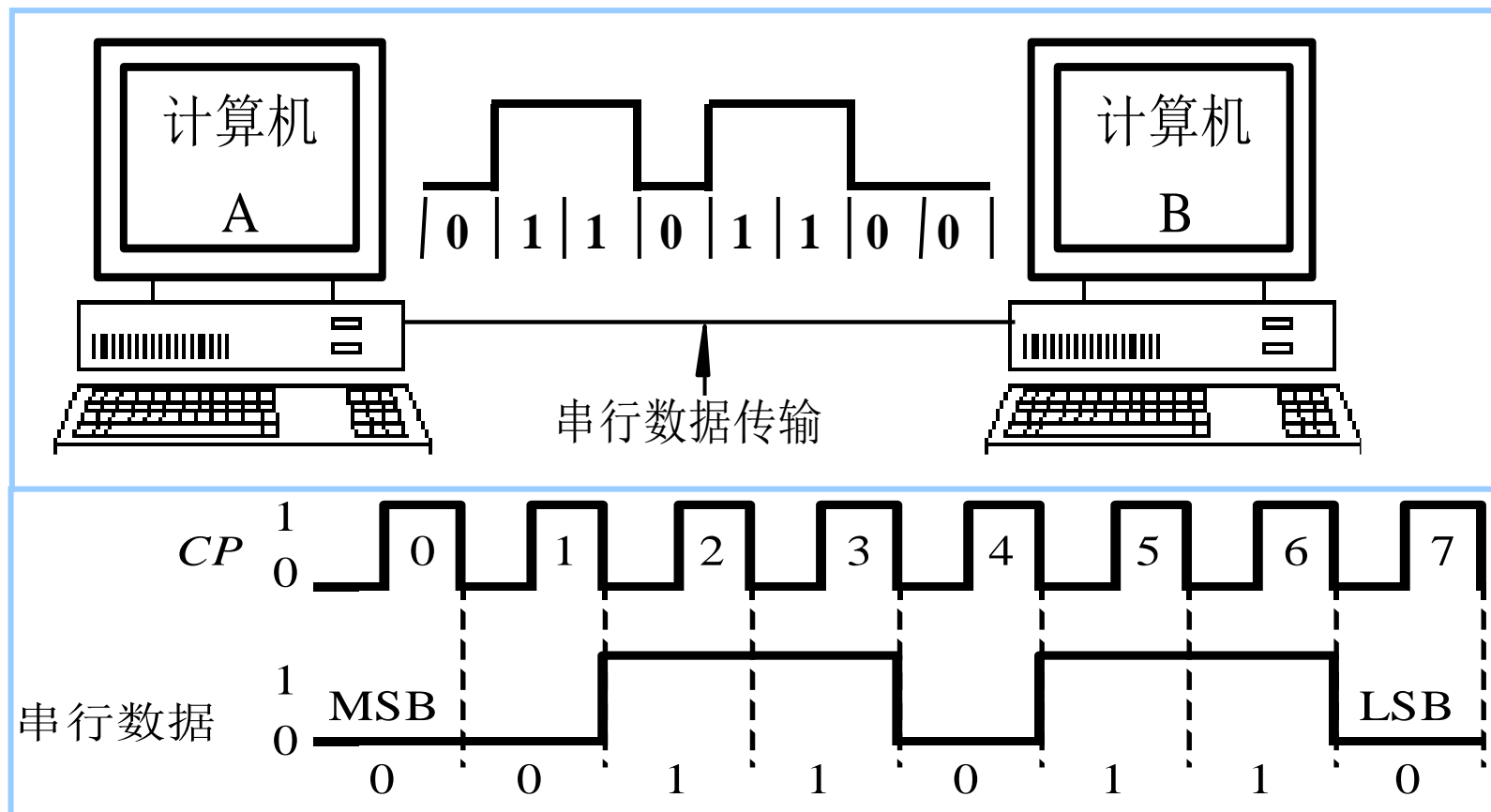
- ✓ 运算规则简单。
- ✓ 需要的设备量少。
- ✓ 可使用逻辑代数。
- ✓ 不符合人们的日常习惯，输入时将十进制转换成二进制，运算结果输出时再转换成十进制数。

□ 二进制数波形表示



□ 二进制数据的传输

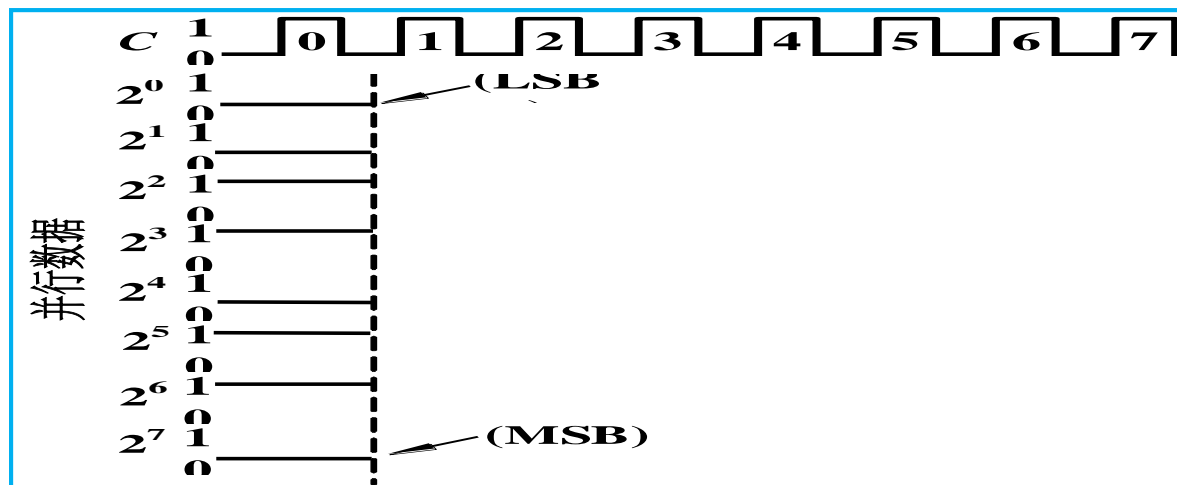
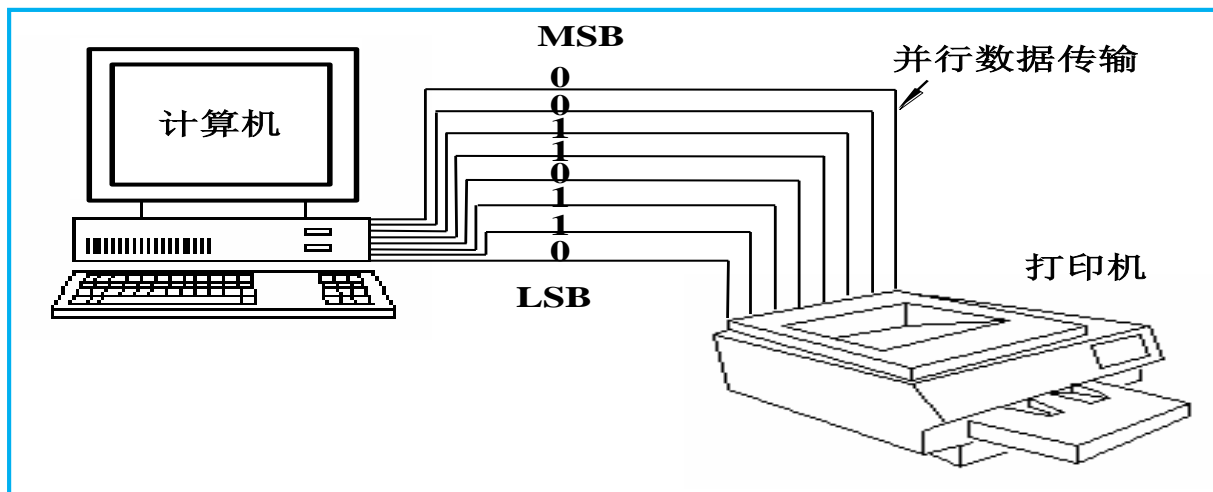
(1) 二进制数据的串行传输



一组数据在时钟脉冲的控制下逐位传送，只需一根数据传输线。

(2) 二进制数据的并行传输

- 将一组二进制数据所有位同时传送。
- 传送速率快, 但数据线较多, 而且发送和接收设备较复杂。



1.2.3 十-二进制之间的转换

□ 二进制转换成十进制：按权展开相加

□ 十进制转换成二进制

◆ 整数：除基取余

对于整数部分可写成

$$(N)_D = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \cdots + b_1 \times 2^1 + b_0 \times 2^0$$

将上式两边分别除以2，得

$$\frac{(N)_D}{2} = \underbrace{b_n \times 2^{n-1} + b_{n-1} \times 2^{n-2} + \cdots + b_1}_{\text{商}} + \boxed{b_0} \quad \text{余数}$$

每除一次2，得到的余数即为二进制数由低到高的1位数字。

连续除以2直到商为0，所有的余数即为十进制数对应的二进制数。

例1.2.2 将十进制数 $(37)_D$ 转换为二进制数。

解：根据上述原理，可将 $(37)_D$ 按如下的步骤转换为二进制数

2	37 余 b_0	1		低
2	18 余 b_1	0		
2	9 余 b_2	1		
2	4 余 b_3	0		
2	2 余 b_4	0		
2	1 余 b_5	1		
0						高

由上得 $(37)_D=(100101)_B$

当十进制数较大时，有什么方法使转换过程简化？

例1.2.3 将 $(133)_D$ 转换为二进制数

解：由于 2^7 为128，而 $133-128=5=2^2+2^0$ ，

所以对应二进制数 $b_7=1$ ， $b_2=1$ ， $b_0=1$ ，其余各系数均为0，

$$(133)_D=(10000101)_B$$

◆ 小数：乘基取整

对于小数部分可写成

$$(N)_D = b_{-1} \times 2^{-1} + b_{-2} \times 2^{-2} + \cdots + b_{-(n-1)} \times 2^{-(n-1)} + b_{-n} \times 2^{-n}$$

将上式两边分别乘以2，得

$$2 \times (N)_D = \boxed{b_{-1}} \times 2^0 + b_{-2} \times 2^{-1} + \cdots + b_{-(n-1)} \times 2^{-(n-2)} + b_{-n} \times 2^{-(n-1)}$$

将十进制小数乘以2，所得乘积的整数即为 b_{-1}

将十进制小数每次去掉上次所得积中的整数再乘以2，

直到满足误差要求，就可完成由十进制小数转换成二进制小数。

例：将十进制小数 $(0.706)_D$ 转换成二进制数,要求误差不大于 2^{-10} 。

解：

$$0.706 \times 2 = 1.412 \quad b_{-1} = 1$$

$$0.592 \times 2 = 1.184 \quad b_{-6} = 1$$

$$0.412 \times 2 = 0.824 \quad b_{-2} = 0$$

$$0.184 \times 2 = 0.368 \quad b_{-7} = 0$$

$$0.824 \times 2 = 1.648 \quad b_{-3} = 1$$

$$0.368 \times 2 = 0.736 \quad b_{-8} = 0$$

$$0.648 \times 2 = 1.296 \quad b_{-4} = 1$$

$$0.736 \times 2 = 1.472 \quad b_{-9} = 1$$

$$0.296 \times 2 = 0.592 \quad b_{-5} = 0$$

$$0.472 \times 2 = 0.944 \quad b_{-10} = 0$$

由于最后的小数大于0.5，根据“四舍五入”原则， b_{-10} 为1。

$$(0.706)_D = (0.1011010011)_B$$

注意：最后的小数小于0.5，不需要“进1”，若大于0.5，则需要“进1”处理，保证每一位都是有效位。

练习

将二进制整数化成十进制数

(1) 1011010

90

(2) 1000111

71

将二进制小数化成十进制数

(1) .1011

0.6875

(2) .0101

0.3125

将二进制数化成十进制数

11011.111

27.875

练习

将十进制整数化成二进制数

(1) 51 (2) 95

110011

1011111

将十进制小数化成二进制数

(1) .5625 (2) .47

0.100100

0.011110

将十进制数化成二进制数（保留6位小数）

77.54

1001101.100011

1001101.100010 最后的小数 >0.5 , “进1” 处理

1.2.4 十六进制和八进制

□ 十六进制 (Hexidecimal)

- ◆ 用十六个数码表示：0~9, A, B, C, D, E, F
- ◆ 遵循逢十六进一，借一当十六的规律
- ◆ 十六进制数的表达式

$$(N)_H = \sum_{i=-\infty}^{\infty} K_i \times 16^i$$

$$(A6.C)_H = 10 \times 16^1 + 6 \times 16^0 + 12 \times 16^{-1}$$

□ 十六-二进制之间的转换

二-十六进制数码表

二进制数	十六进制数	二进制数	十六进制数
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F

□ 十六-二进制之间的转换

◆ 二进制数转换为十六进制数

- 以小数点为界，分别往高、往低每4位为一组，最后不足4位时用0补充，然后写出每组对应的十六进制字符，即为对应十六进制数。

◆ 十六进制数转换成二进制数

- 将十六进制的各位数码分别用对应的4位二进制数代入。

例1：将二进制数 $(1111111000111.100101011)_b$ 转换成对应的十六进制数。

<u>000</u>	<u>1111</u>	<u>1110</u>	<u>0011</u>	<u>1</u>	<u>1001</u>	<u>0101</u>	<u>11000</u>
1	F	C	7	.	9	5	8

$$(1111111000111.100101011)_b = (1FC7.958)_h$$

例2：将十六进制数 $(3AB.4A)_h$ 转换成对应的二进制数。

3	A	B	.	4	A
↓	↓	↓		↓	↓
0011	1010	1011		0100	1010

$$(3AB.4A)_h = (1110101011.0100101)_b$$

□ 八进制 (Octal)

◆ 用八个数码表示：0, 1, 2, 3, 4, 5, 6, 7

◆ 遵循逢八进一，借一当八的规律

◆ 八进制数的表达式

$$(N)_O = \sum_{i=-\infty}^{\infty} K_i \times 8^i$$

□ 八-二进制之间的转换

二-八进制数码表

二进制数	八进制数
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

□ 八-二进制之间的转换

◆ 二进制数转换为八进制数

- 以小数点为界，分别往高、往低每3位为一组，最后不足3位时用0补充，然后写出每组对应的八进制字符，即为对应八进制数。

◆ 八进制数转换成二进制数

- 将八进制的各位数码分别用对应的3位二进制数代入。

例1：将二进制数 $(10100101.01011101)_b$ 转换成对应的八进制数

$(245.272)_o$

例2：将八进制数 $(367.505)_o$ 转换成对应的二进制数

$(11110111.101000101)_b$

□ 十进制数与二、八、十六进制数对照表

十进制	二进制	八进制	十六进制	十进制	二进制	八进制	十六进制
0	0000	00	0	11	1011	13	B
1	0001	01	1	12	1100	14	C
2	0010	02	2	13	1101	15	D
3	0011	03	3	14	1110	16	E
4	0100	04	4	15	1111	17	F
5	0101	05	5	16	10000	20	10
6	0110	06	6	17	10001	21	11
7	0111	07	7	18	10010	22	12
8	1000	10	8	19	10011	23	13
9	1001	11	9	20	10100	24	14
10	1010	12	A				

□ 十六进制的优点

- 1) 与二进制之间的转换容易；
- 2) 计数容量较其它进制大，假如同样采用四位数码：

二进制最多可计至 $(1111)_B = (15)_D$

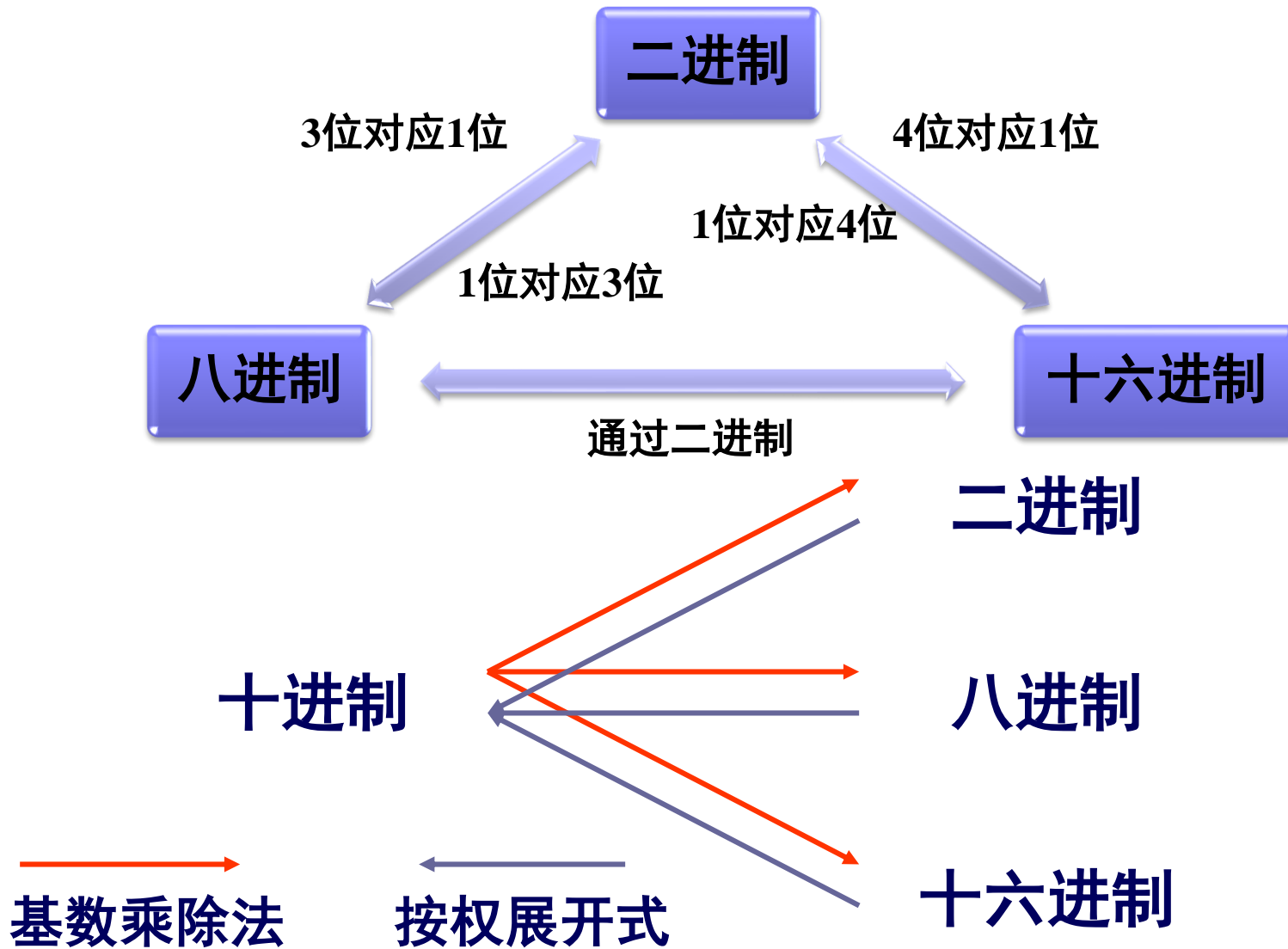
八进制可计至 $(7777)_O = (2800)_D$

十进制可计至 $(9999)_D$

十六进制可计至 $(FFFF)_H = (65535)_D$

- 3) 书写简洁。

□ 常用数制间的转换小结



第一章 数字逻辑概论

1.1 数字信号与数字电路

1.2 数制

1.3 二进制数的算术运算

1.4 二进制代码

1.5 二值逻辑变量与基本逻辑运算

1.3 二进制数的算术运算

1.3.1 无符号二进制数的算术运算

1.3.2 带符号二进制数的减法运算

1.3.1 无符号二进制数的算术运算

□ 无符号二进制加法规则

$$0+0=0, 0+1=1, 1+1=\textcolor{red}{10}$$

□ 无符号二进制减法规则

$$0-0=0, 1-1=0, 1-0=1, 0-1=\textcolor{red}{11}$$

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ 1\ 0\ 1\ 1\ \textcolor{red}{11} \\ +\ 1\ 0\ 0\ 1\ \textcolor{red}{9} \\ \hline 1\ 0\ 1\ 0\ 0\ \textcolor{red}{20} \end{array}$$

$$\begin{array}{r} 1\ 0\ 1 \\ 1\ 0\ 1\ 0\ \textcolor{red}{10} \\ -\ 0\ 1\ 0\ 1\ \textcolor{red}{5} \\ \hline 0\ 1\ 0\ 1\ \textcolor{red}{5} \end{array}$$

□ 无符号二进制乘法规则

$0 \times 0 = 0, \quad 0 \times 1 = 1 \times 0 = 0, \quad 1 \times 1 = 1$

□ 无符号二进制除法规则

$0 \div 1=0, \quad 1 \div 1=1$

$$\begin{array}{r}
 1101 \\
 \times 1011 \\
 \hline
 1101 \\
 1101 \\
 0000 \\
 + 1101 \\
 \hline
 10001111
 \end{array}$$

$$\begin{array}{r}
 1.011 \\
 \hline
 111 \overline{) 1010} \\
 \underline{111} \\
 1100 \\
 \underline{111} \\
 1010 \\
 \underline{111} \\
 11
 \end{array}$$

1.3.2 带符号二进制数的减法运算

□ 实数在计算机中的表示

- ◆ **真值**：真值是指在数值前面用“+”号表示正数，“-”号表示负数的带符号二进制数。
- ◆ **机器数**：机器数是指在数字系统中用“0”表示符号为“+”，用“1”表示符号为“-”，即把符号“数值化”后的带符号二进制数。



00001101 表示数+13

10001101 表示数-13

□ 原码

在二进制原码表示法中，符号位为0表示正数，符号位为1表示负数，尾数不变。

真值	原码
$+X_{n-1} X_{n-2} \dots X_1$	$0X_{n-1} X_{n-2} \dots X_1$
$-X_{n-1} X_{n-2} \dots X_1$	$1X_{n-1} X_{n-2} \dots X_1$

整数0的原码有两种形式

□ 原码的优缺点

◆ 原码的优点是容易理解。它和代数中的正负数的表示方法很接近。

◆ 原码的加法规则：

判断被加数和加数的符号是**同号**还是**异号**：

➤ **同号时**，做加法，结果的符号就是被加数的符号。

➤ **异号时**，先比较被加数和加数的数值(绝对值)的大小，然后由大值减去小值，结果的符号取大值的符号。

为了简化加法器的设计，必须寻找其他表示负数的方法。

□ 补码

整数补码的定义（带符号位）

设二进制整数 $X = \pm x_{n-1} \dots x_1$, 则其补码定义为:

$$[X]_{\text{补}} = \begin{cases} X & 0 \leq X < 2^{n-1} \\ 2^n + X & -2^{n-1} < X \leq 0 \end{cases} \quad \longrightarrow \quad [X]_{\text{补}} = 2^n + X$$

整数0的补码只有一种形式

对8位码来说, 这里 $n=8$, 7位表示真值, 1位表示符号

□ 补码与原码的关系

◆ 符号位与原码相同，0表示正数，1表示负数。

◆ 数值位与符号相关。

□ 正数补码的数值位与原码的数值位相同

□ 负数补码的数值位是原码的数值位按位取反，再在最低位加1。

□ 补码的加减法运算

◆ 补码的加法运算

$$\underline{[X]_{\text{补}} + [Y]_{\text{补}} = [X + Y]_{\text{补}}}$$

注意：符号位的进位需丢弃

例1：已知 $X = +0000111$ ， $Y = -0010011$ ，求 $X + Y$

例2：已知 $X = -0011001$ ， $Y = -0000110$ ，求 $X + Y$

在计算机中进行两个带符号数的加法运算，**只要将给定的真值用补码表示，就可以直接进行加法运算。**在运算过程中不必判断加数和被加数的正负，一律作加法，最后将结果转换为真值即可。

◆ 补码的减法运算

$$[X]_{\text{补}} - [Y]_{\text{补}} = [X - Y]_{\text{补}} = [X + (-Y)]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$$

◆ 在补码中，减法统一于加法。关键是如何很方便地从 $[Y]_{\text{补}}$ 求出 $[-Y]_{\text{补}}$ ，然后做加法。

◆ 从 $[Y]_{\text{补}}$ 求 $[-Y]_{\text{补}}$ 的方法是：符号位连同数值位一起取反加1。

例1：已知 $X = +1100000$ ， $Y = +0010011$ ，求 $X - Y$

例2：已知 $X = -0111000$ ， $Y = -0010001$ ，求 $X - Y$

□ 反码

整数反码的定义（带符号位）

设二进制整数 $X = \pm x_{n-1} \dots x_1$, 则其反码定义为:

$$[X]_{\text{反}} = \begin{cases} X & 0 \leq X < 2^{n-1} \\ (2^n - 1) + X & -2^{n-1} < X \leq 0 \end{cases}$$

整数0的反码有两种形式

□ 反码与原码的关系

◆ 符号位与原码相同，0表示正数，1表示负数

◆ 数值位与符号相关

□ 正数反码的数值位与原码的数值位相同

□ 负数反码的数值位是原码的数值位按位取反

□ 原码、反码、补码的关系

当X为正数时

$$[X]_{\text{原}} = [X]_{\text{补}} = [X]_{\text{反}}$$

当X为负数时

$$[X]_{\text{补}} = [X]_{\text{反}} + 1$$

□ 4位二进制数原码、反码、补码对照表

十进制数	二进制数		
	原码	反码	补码
-8	--	--	1000
-7	1111	1000	1001
-6	1110	1001	1010
-5	1101	1010	1011
-4	1100	1011	1100
-3	1011	1100	1101
-2	1010	1101	1110
-1	1001	1110	1111
-0	1000	1111	0000
+0	0000	0000	0000
+1	0001	0001	0001
+2	0010	0010	0010
+3	0011	0011	0011
+4	0100	0100	0100
+5	0101	0101	0101
+6	0110	0110	0110
+7	0111	0111	0111

n位带符号二进制数的码的数值范围：

原码: $-(2^{n-1}-1) \sim +(2^{n-1}-1)$

反码: $-(2^{n-1}-1) \sim +(2^{n-1}-1)$

补码: $-2^{n-1} \sim +(2^{n-1}-1)$

□ 反码的加减法运算

◆ 反码的加法运算

$$\underline{[X]_{\text{反}} + [Y]_{\text{反}} = [X+Y]_{\text{反}}}$$

运算时，符号位和数值位一样参加运算。当符号位有进位产生时，应将进位加到运算结果的最低位，才能得到最后结果。

例1：已知 $X=+0000111$ ， $Y=-0010011$ ，求 $X+Y$

例2：已知 $X=-0011001$ ， $Y=-0000110$ ，求 $X+Y$

在计算机中进行两个带符号数的加法运算，只要将给定的真值用反码表示，就可以直接进行加法运算。在运算过程中不必判断加数和被加数的正负，一律作加法，最后将结果转换为真值即可。

◆ 反码的减法运算

$$\underline{[X]_{\text{反}} - [Y]_{\text{反}} = [X - Y]_{\text{反}} = [X + (-Y)]_{\text{反}} = [X]_{\text{反}} + [-Y]_{\text{反}}}$$

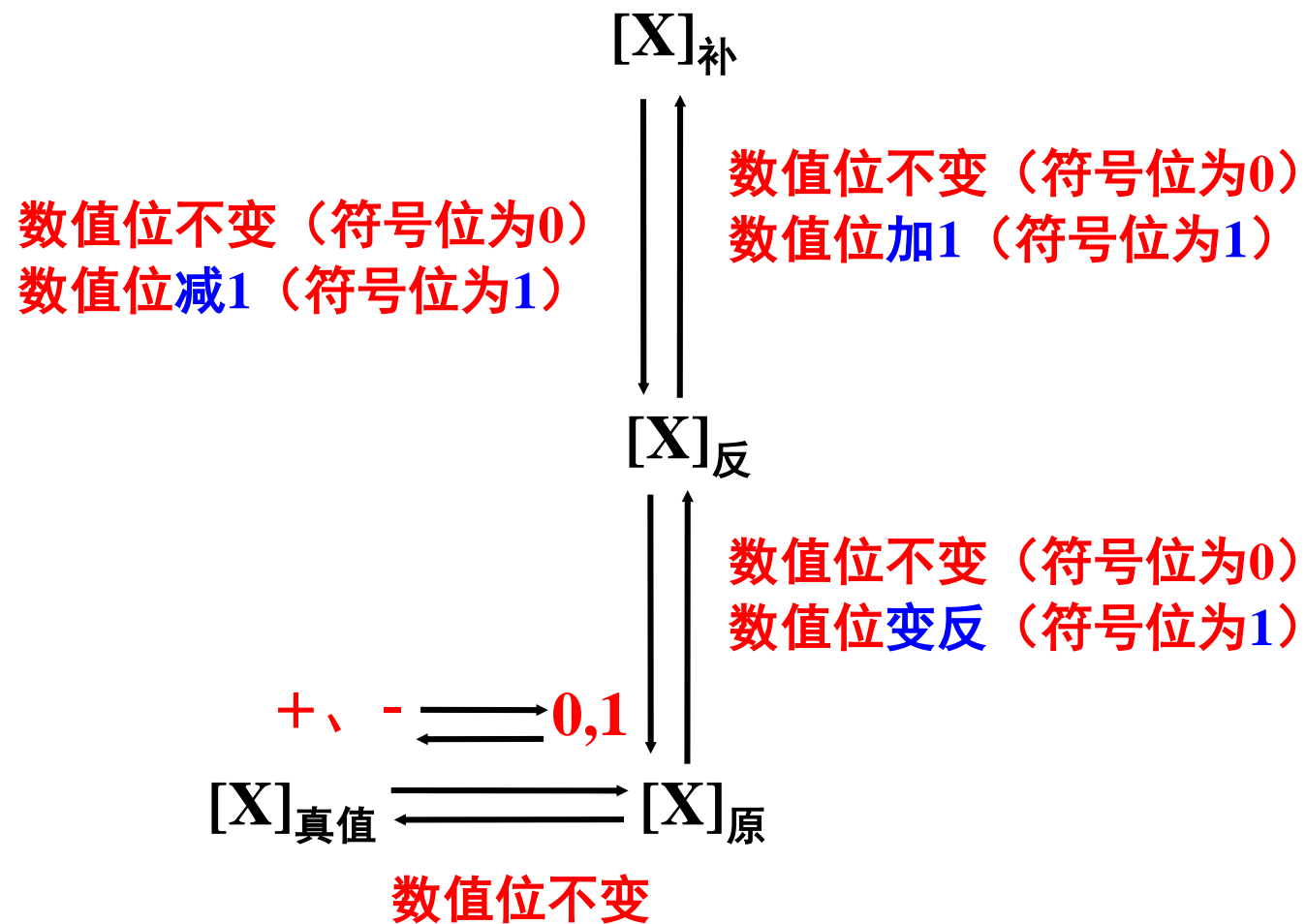
- ◆ 在反码中，减法统一于加法。关键是如何很方便地从 $[Y]_{\text{反}}$ 求出 $[-Y]_{\text{反}}$ ，然后做加法。
- ◆ 从 $[Y]_{\text{反}}$ 求 $[-Y]_{\text{反}}$ 的方法是：符号位连同数值位一起取反。

例1：已知 $X=+1100000$ ， $Y=+0010011$ ，求 $X-Y$

例2：已知 $X=-0111000$ ， $Y=-0010001$ ，求 $X-Y$

注意：利用反码进行运算，如果结果溢出了1，一定要将运算结果再补上1，才能得到正确的结果。

□ 原码、补码和反码间的相互转换



求解原码，反码，补码等相关问题的关键第一步是什么？

- 1) 判断此数是正是负
- 2) 正数的反码、补码与原码相同
- 3) 负数的反码是符号位不变，（原码）其余各位取反；负数的补码是符号位不变，（原码）其余各位取反+1

小窍门：一个数的补码再次求补即为其原码

□ 溢出和溢出的判别

例：用4位二进制补码计算5+7

$$\begin{array}{r} +5 \\ +) +7 \\ \hline +12 \end{array}$$

$$\begin{array}{r} 0101 \\ +) 0111 \\ \hline 1100 \\ -4 \end{array}$$

□ 溢出和溢出的判别

$$\begin{array}{r} +4 \quad 0100 \\ +) +3 \quad +) 0011 \\ \hline +7 \quad \underline{00111} \end{array}$$

$$\begin{array}{r} -5 \quad 1011 \\ +) -3 \quad +) 1101 \\ \hline -8 \quad \underline{11000} \end{array}$$

无溢出

$$\begin{array}{r} +2 \quad 0010 \\ +) +6 \quad +) 0110 \\ \hline +8 \quad \underline{01000} \end{array}$$

$$\begin{array}{r} -3 \quad 1101 \\ +) -6 \quad +) 1011 \\ \hline -9 \quad \underline{10111} \end{array}$$

有溢出

判别：如果两个加数的符号相同，而和的符号与它们相反时，产生溢出
(符号位的进位与和数的符号位相反时，产生溢出)

练习

1、将下列二进制数用8位原码表示

(1) 11011

(2) -111100

2、将下列二进制数用8位补码表示

(1) 10101

(2) -1110111

3、将下列二进制数用8位反码表示

(1) 10011

(2) -1110111

练习

1、写出与下列二进制原码对应的补码和反码

(1) 00110110

(2) 10101001

2、写出与下列二进制补码对应的反码和原码

(1) 01001110

(2) 11001111

3、写出与下列二进制反码对应的补码和原码

(1) 00011101

(2) 11101000

4、写出下列用十六进制书写的补码对应的反码和原码

(1) BC

(2) 7D

第一章 数字逻辑概论

1.1 数字信号与数字电路

1.2 数制

1.3 二进制数的算术运算

1.4 二进制代码

1.5 二值逻辑变量与基本逻辑运算



1.4 二进制代码

1.4.1 二-十进制码

1.4.2 格雷码

1.4.3 ASCII码

1.4.1 二-十进制码

□ 编码

□ 以一定的规则编制代码，用以表示十进制数值、字母、符号等过程。

□ 译码

□ 将代码还原成所表示的十进制数、字母、符号等过程。

□ 需要编码的信息有N项，对应的二进制数码的位数为n，N和n需满足

$$\square 2^n \geq N$$

二-十进制码（BCD码）：4位二进制代码编码1位十进制数字

□ 常用BCD码

十进制数N	NBCD (8421) 码	2421码	余3码	余3循环码
0	0000	0000	0011	0010
1	0001	0001	0100	0110
2	0010	0010	0101	0111
3	0011	0011	0110	0101
4	0100	0100	0111	0100
5	0101	1011	1000	1100
6	0110	1100	1001	1101
7	0111	1101	1010	1111
8	1000	1110	1011	1110
9	1001	1111	1100	1010

- ◆ 四种BCD码都是用四位二进制代码表示一位十进制数字。
- ◆ 四种BCD码与十进制数之间的转换是以四位对应一位，直接进行变换。一个n位十进制数对应的BCD码一定为4n位。

十进制数N	NBCD (8421) 码	2421码
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	1011
6	0110	1100
7	0111	1101
8	1000	1110
9	1001	1111

有权码(weighted code): 如8421码,2421码。其特点是, 当知道权值和代码时, 就可计算出它代表的十进制值。

8421码的0111, 代表
 $0 \times 8 + 1 \times 4 + 1 \times 2 + 1 \times 1 = 7$

$$(a_3 a_2 a_1 a_0)_{8421 \text{码}} = (8a_3 + 4a_2 + 2a_1 + a_0)_{10}$$

2421码的1110, 代表
 $1 \times 2 + 1 \times 4 + 1 \times 2 + 0 \times 1 = 8$

$$(a_3 a_2 a_1 a_0)_{2421 \text{码}} = (2a_3 + 4a_2 + 2a_1 + a_0)_{10}$$

十进制数N	NBCD (8421) 码	2421码
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	1011
6	0110	1100
7	0111	1101
8	1000	1110
9	1001	1111

8421码的特点：

- 编码简单**直观**，同十进制互换是直接按位转换。
- 具有**奇偶性**，凡是对应十进制数是奇数的码字，最低位皆为1；凡是对应十进制数是偶数的码字，最低位皆为0。

2421码的特点：

- 是一种具有自补性的BCD码，简称**自补码**。即十进制数0~9十个数符的2421码自身按位取反，得到的另一个2421码为原数的9补码的2421码表示，即以9为模的补码。

十进制数N	余3码	余3循环码
0	0011	0010
1	0100	0110
2	0101	0111
3	0110	0101
4	0111	0100
5	1000	1100
6	1001	1101
7	1010	1111
8	1011	1110
9	1100	1010

余3码的特点：

- 自补码
- 无权码
- 由8421码加3得到

余3循环码的特点：

- 无权码
- 相邻性

BCD码虽然形式上和二进制数一样，但各位之间并不遵循“逢二进一”的进位关系，不能按二进制运算法则运算

练习

1、用8421BCD码表示下列各数

(1) 248

2、将下列用NBCD码表示的数还原为十进制数

(1) 100101010011

3、分别用2421码和循环码表示下列各十进制数

(1) 25

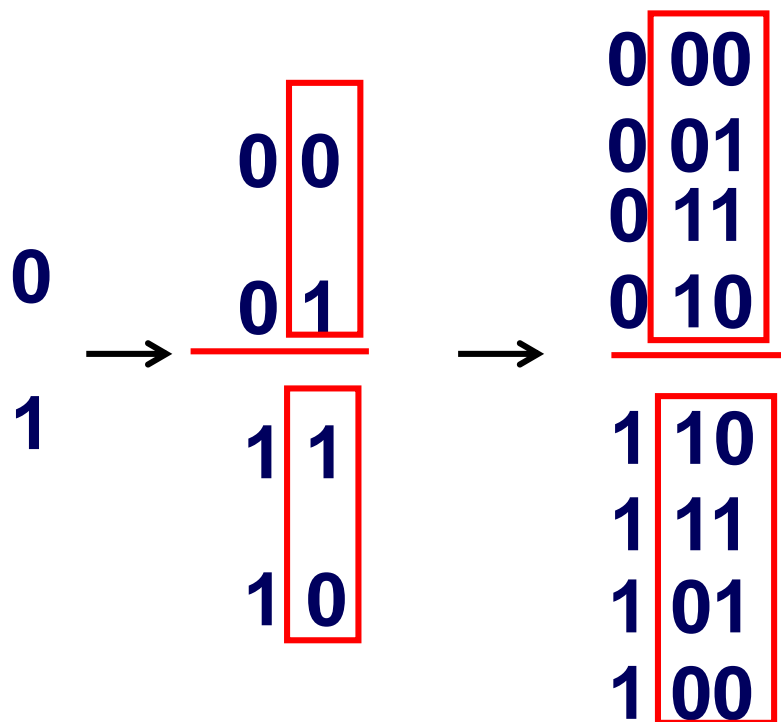
1.4.2 格雷码

- 格雷码是一种**无权码**。
- 编码特点是：任何两个**相邻代码**之间**仅有一位不同**。
- 该特点常用于模拟量的转换。当模拟量发生微小变化，格雷码仅仅改变一位，这与其它码同时改变2位或更多的情况相比，更加可靠，且容易检错。
- **不能直接进行算术运算**。

十进制数N	二进制码 ($b_3 b_2 b_1 b_0$)	格雷码($G_3 G_2 G_1 G_0$)
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

□ 格雷码的构成方法

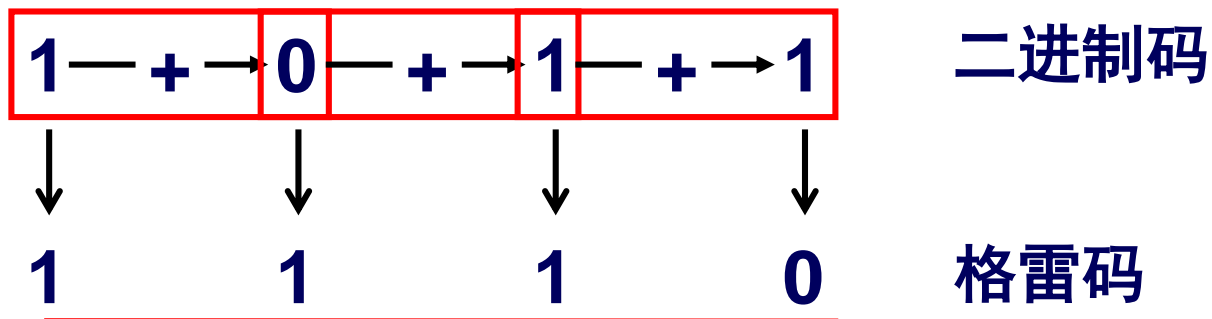
- ◆ 1位格雷码有两个代码，**0和1**
- ◆ $(n+1)$ 位格雷码有 2^{n+1} 组代码，其中前 2^n 组代码是将 n 位格雷码按顺序排列，最高位补0。
- ◆ $(n+1)$ 位格雷码的后 2^n 组代码是将 n 位格雷码按倒序排列，最高位补1。



□ 二进制码转换成格雷码

- ◆ 格雷码的最高位与二进制码的最高位相同
- ◆ 从左到右，逐一将二进制码相邻的2位相加（舍去进位），作为格雷码的下一位。

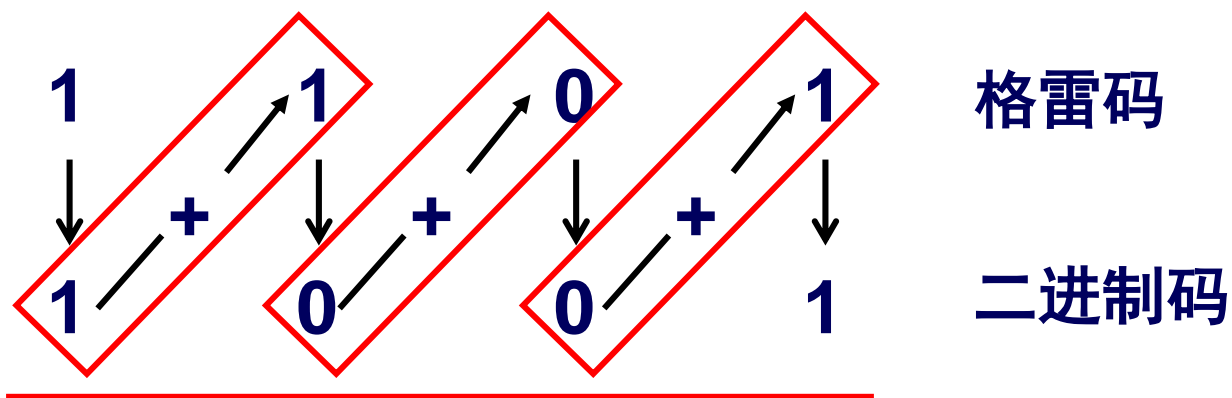
例1.4.1 将二进制码1011转换成格雷码



□ 格雷码转换成二进制码

- ◆ 二进制码的最高位与格雷码的最高位相同
- ◆ 将产生的每一个二进制码，与下一位相邻的格雷码相加（舍去进位），作为二进制码的下一位。

例1.4.2 将格雷码1101转换成二进制码



□ 格雷码的应用

- ◆ 格雷码是一种错误最小化的编码，在模数转换、汽车制动、数控机床、伺服电机、机器人等领域有广泛的应用。
- ◆ 在需要将模拟量转换成用相邻二进制数序列表示数字量的系统中。
- ◆ 光电码盘用于需要指示角度位置的系统。

练习

将下列二进制数转化为格雷码

(1) 11010

将下列格雷码转化为二进制数

(1) 10110

1.4.3 ASCII码

- ASCII码即美国标准信息交换码。
- 它共有128个代码，可以表示大、小写英文字母、十进制数、标点符号、运算符号、控制符号等，普遍用于计算机的键盘指令输入和数据等。

第一章 数字逻辑概论

1.1 数字信号与数字电路

1.2 数制

1.3 二进制数的算术运算

1.4 二进制代码

1.5 二值逻辑变量与基本逻辑运算

1.5 二值逻辑变量与基本逻辑运算

□ **逻辑**——希腊语：词语，规律，推理，关系

此处指事物之间的因果关系

□ **命题**——能用真和假来判断的陈述句

太阳从东边升起

$$3+2=8$$

疑问句与感叹句不是命题

□ **命题运算**——逻辑前提与逻辑结论

如星期天天晴就去公园开展活动

逻辑前提

逻辑结论

□ 逻辑代数（布尔代数，开关代数）

◆ 英国数学家乔治.布尔1854年提出

◆ 用符号代替文字，数学代替推理

A——星期天
B——天晴
F——去公园开展活动

} 逻辑前提

——逻辑结论

$$F = f(A, B)$$

A, B, F 非真即假，非假即真

若 $A \neq 1$, 则 $A=0$; 若 $A \neq 0$, 则 $A=1$

A, B, F ——逻辑变量

1, 0 ——逻辑常量

$$F=AB$$

逻辑代数是一个由逻辑变量集 K ，常量1和0以及“与”、“或”、“非”3种基本运算构成的一个封闭的代数系统，记为 $L=\{K, +, \cdot, -, 0, 1\}$ 。它是一个二值代数系统。常量1和0表示真和假，无大小之分。

□ 逻辑代数的基本运算

◆ 非(NOT) 逻辑求反 非门

◆ 与(AND) 逻辑乘法 与门

◆ 或(OR) 逻辑加法 或门

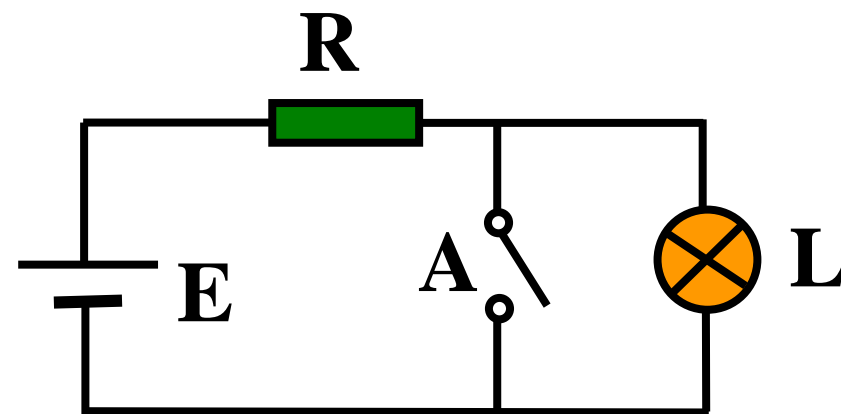
□ 非逻辑和非运算

非逻辑：决定事件发生的条件只有一个，条件不具备时事件发生（成立），条件具备时事件不发生。

非逻辑真值表

A	$L = \bar{A}$
0	1
1	0

特点：1则0, 0则1

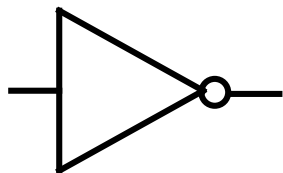
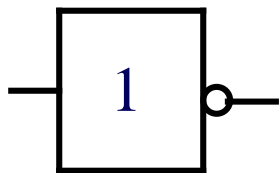


A	接通	1
	不接通	0
L	亮	1
	不亮	0

◆ 函数表达式为: $L = f(A) = \overline{A}$

◆ 逻辑符号:

(a) 国标GB4728.12-85符号 (b) MIL符号

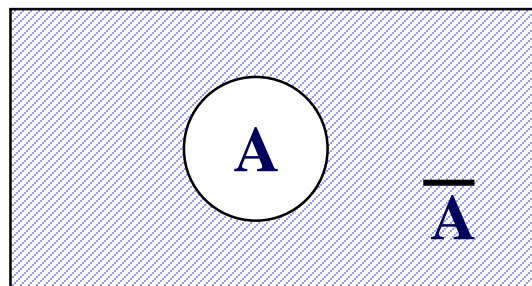


◆ 运算规则:

$$\overline{0} = 1$$

$$\overline{1} = 0$$

$$\overline{\overline{A}} = A$$

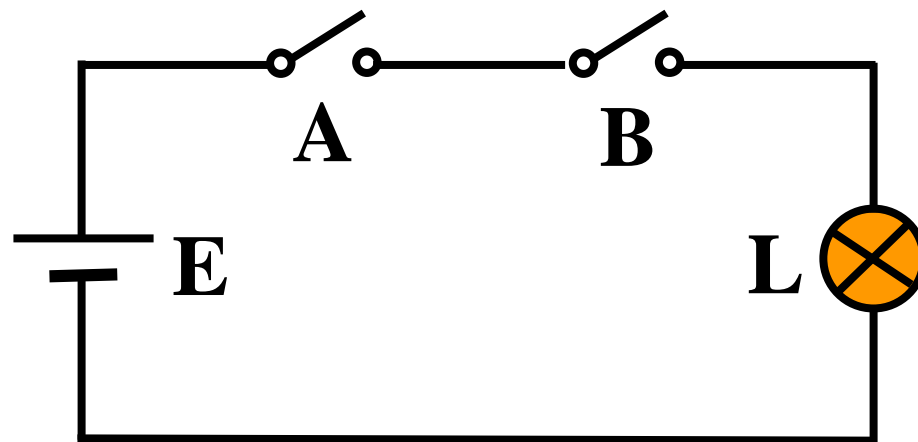


□ 与逻辑和与运算

与逻辑： 决定事件发生的各条件中，所有条件都具备，事件才会发生（成立）。

与逻辑真值表

A	B	$L=A \times B$
0	0	0
0	1	0
1	0	0
1	1	1

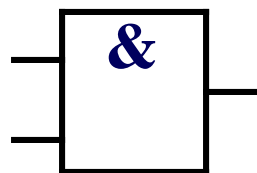


特点： 任0 则0, 全1则1

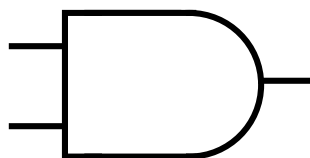
◆ 逻辑表达式: $L=AB$

◆ 逻辑符号:

(a) 国标GB4728.12-85符号



(b) MIL符号



◆ 运算规则:

$$0 \cdot 0 = 0$$

$$0 \cdot 1 = 1 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

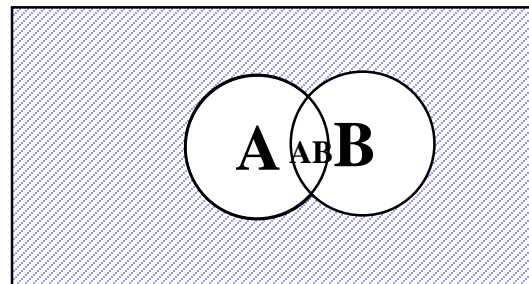
$$A \cdot 0 = 0$$

$$A \cdot 1 = A$$

$$A \cdot A = A$$

$$A \cdot \bar{A} = 0$$

$$A \cdot A \cdot A \cdots A = A$$

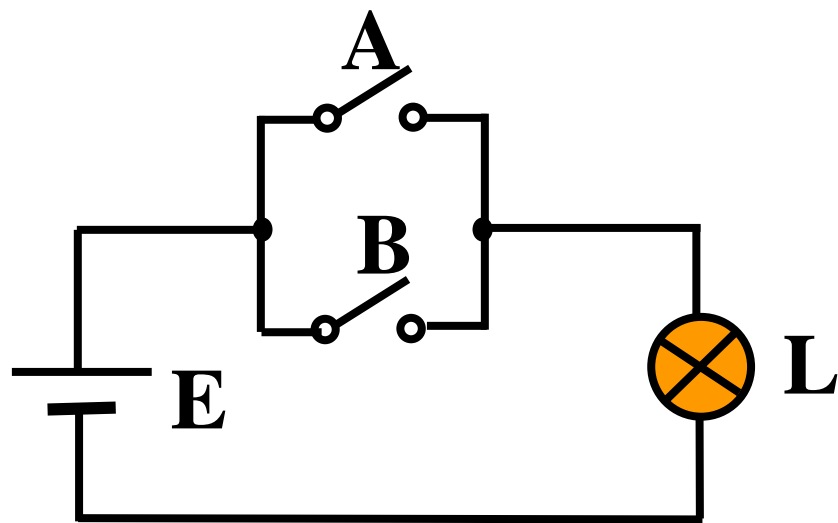


□ 或逻辑和或运算

或逻辑： 决定事件发生的各条件中，有一个或一个以上的条件具备，事件就会发生（成立）。

或逻辑真值表

A	B	$L=A+B$
0	0	0
0	1	1
1	0	1
1	1	1

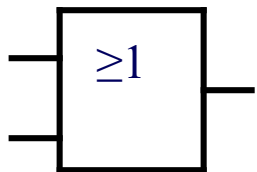


特点：任1则1,全0则0

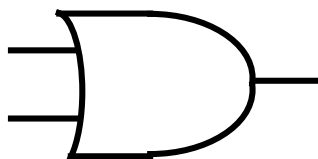
◆ 逻辑表达式: $L=A+B$

◆ 逻辑符号:

(a) 国标GB4728.12-85符号



(b) MIL符号



◆ 运算规则:

$$0+0=0$$

$$1+0=0+1=1$$

$$1+1=1$$

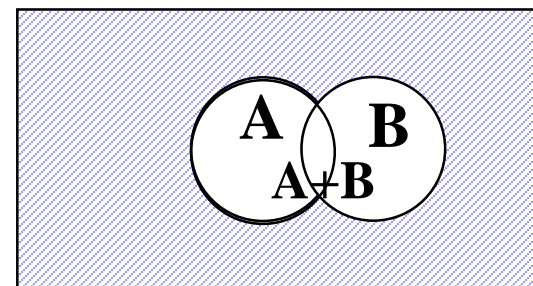
$$A+0=A$$

$$A+1=1$$

$$A+A=A$$

$$A+\bar{A}=1$$

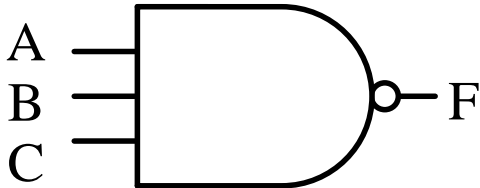
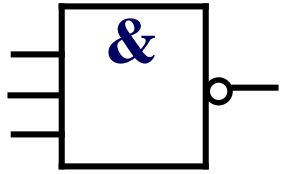
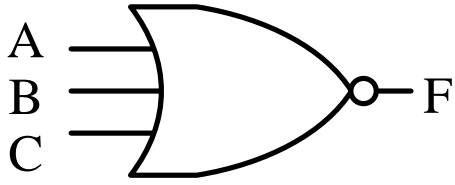
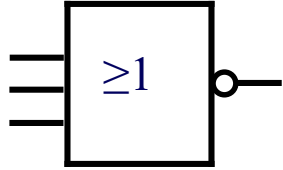
$$A+A+\dots+A=A$$



□ 以上三种基本逻辑运算如在逻辑运算式中同时出现时，其优先顺序为：**非、与、或**，必要时还可用括号加以提前。

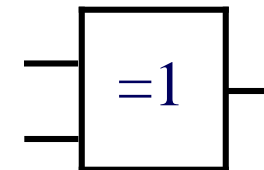
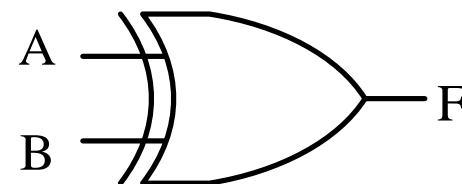
□ 几种常用的逻辑关系

“与”、“或”、“非”是三种基本的逻辑关系，任何其它的逻辑关系都可以以它们为基础表示。

<p>与非：条件A、B、C都具备，则F不发生。</p>	$F = \overline{A \cdot B \cdot C}$ <p>任0则1，全1则0</p>		
<p>或非：条件A、B、C任一具备，则F不发生。</p>	$F = \overline{A + B + C}$ <p>任1则0，全0则1</p>		

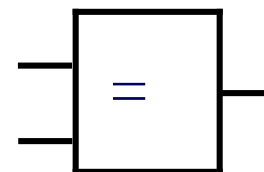
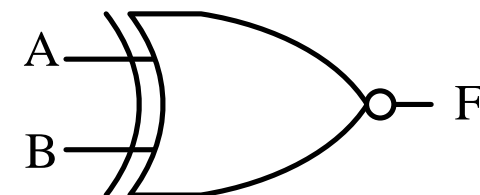
异或： 输入A、B状态相同，输出0； 状态不同，输出1。

$$F = A\bar{B} + \bar{A}B \\ = A \oplus B$$



同或： 输入A、B状态相同，输出1； 状态不同，输出0。

$$F = \bar{A}\bar{B} + AB \\ = \overline{A \oplus B} = A \odot B$$



本章小结

- ◆ 概念和表示方法(十进制、二进制 *、八进制、十六进制)
- ◆ 二进制的优点
- ◆ 数制间的转换*
- ◆ 码制间的转换*
- ◆ BCD码的概念(二进制码与循环码)
- ◆ 基本逻辑运算

□ 作业

◆ 1.1.2

◆ 1.2.1; 1.2.2(4); 1.2.3(3); 1.2.6(3)); 1.2.7(3) ;1.2.8(3); 1.2.9

◆ 1.3.1(2,4); 1.3.2(2,4); 1.3.3(2,4)); 1.3.4(4)

◆ 1.4.1(4); 1.4.2(4); 1.4.3(4); 1.4.4(4)

◆ 1.5.3