1. 数字逻辑概论

1. 数字逻辑基础

教学基本要求

1、了解数字信号与数字电路的基本概念

2、了解数字信号的特点及表示方法

3、掌握常用二～十、二～一十六进制数的转换

4、了解常用二进制码，熟悉8421 BCD码

5、掌握基本逻辑运算及逻辑函数的表示方法

1.1 数字电路与数字信号

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 模拟信号与数字信号

1.1.4 数字信号的描述方法

1.1数字电路与数字信号

1.1.1数字技术的发展及其应用

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 ，1 0 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-- 芯片内部的布线细微到纳米量级(0.05μm以下)

微处理器的时钟频率超过3GHz(10^9Hz)，高达6.8GHz

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

电路设计方法伴随器件变化从传统走向现代

a)传统的设计方法：

采用自下而上的设计方法；由人工组装,经反复调试、验证、修改完成。所用的元器件较多，电路可靠性差,设计周期长。

b)现代的设计方法：

现代EDA技术实现硬件设计软件化。采用从上到下设计方法，电路设计、 分析、仿真 、修订 全通过计算机完成。

1.1.2、数字集成电路的分类及特点

1、数字集成电路的分类

根据电路的结构特点及其对输入信号的响应规则的不同，

--数字电路可分为组合逻辑电路和时序逻辑电路。

从电路的形式不同，

--数字电路可分为集成电路和分立电路

从器件不同

--数字电路可分为TTL 和 CMOS电路

从集成度不同

--数字集成电路可分为小规模、中规模、大规模、超大规模和甚大规模五类。

集成度:每一芯片所包含的门个数

分类	门的个数	典型集成电路
小规模	最多12个	逻辑门、触发器
中规模	12~99	计数器、加法器
大规模	100~9999	小型存储器、门阵列
超大规模	10,000~99,999	大型存储器、微处理器
甚大规模	10^6以上	可编程逻辑器件、多功能专用集成电路

2、数字集成电路的特点

1) 稳定性高, 抗干扰能力强

2) 易于设计–对0和1表示的信号进行逻辑运算和处理

3) 体积小, 通用性好, 成本低，便于集成.

4)具可编程性, 可实现硬件设计软件化

5)高速度 低功耗

6)便于存储、传输和处理

3、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。

(2) 数字电路的设计方法

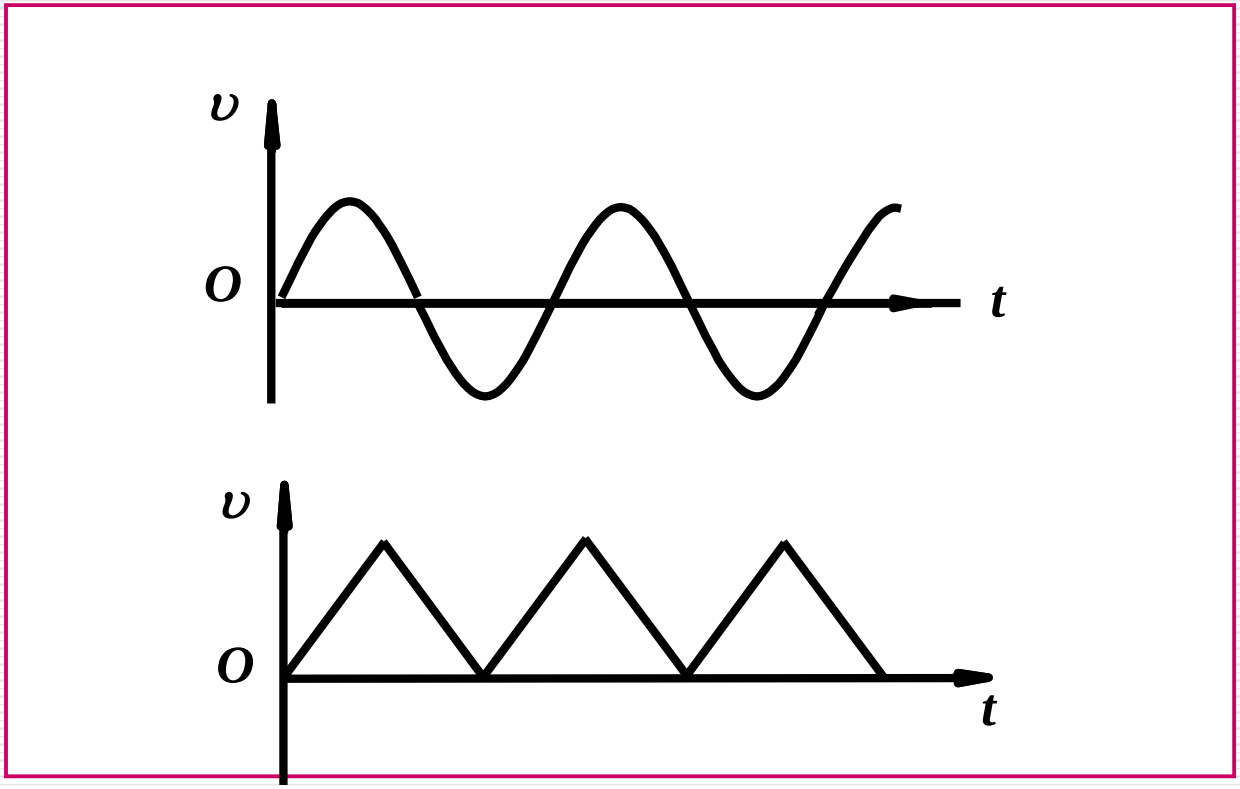
数字电路的设计:从给定的逻辑功能要求出发，选择适当的逻辑器件，设计出符合要求的逻辑电路。

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 模拟信号与数字信号

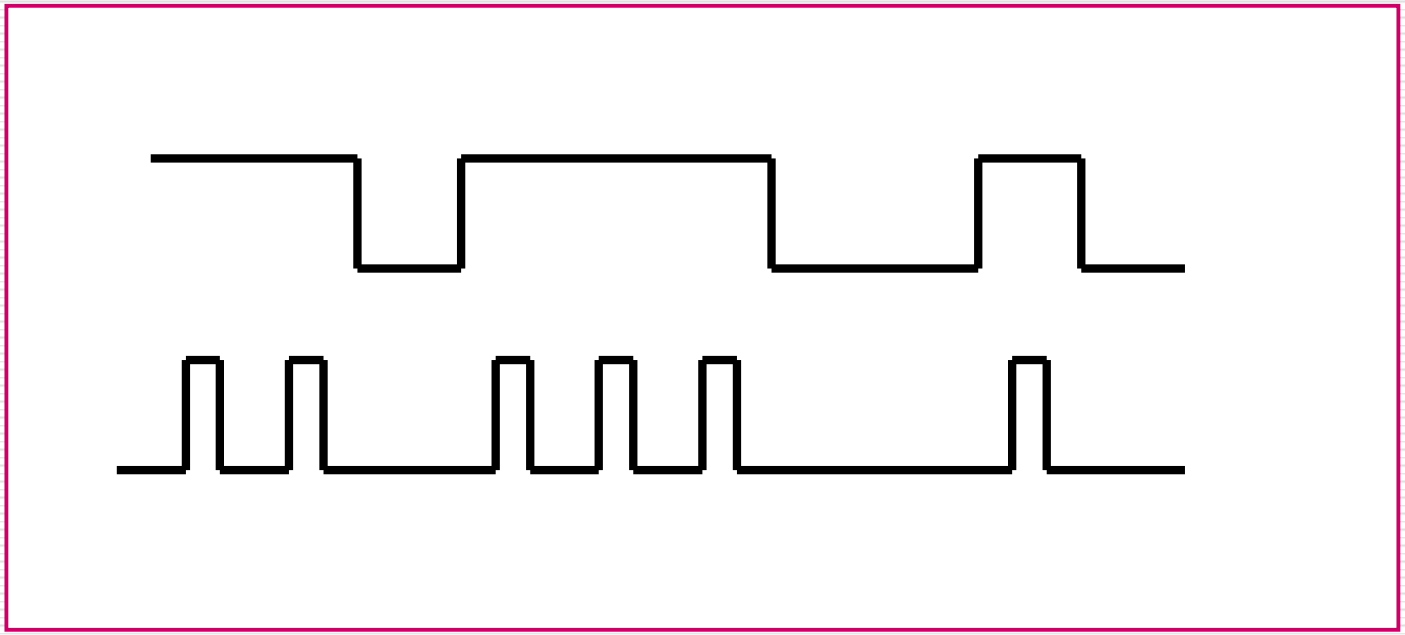
1. 模拟信号

---时间和数值均连续变化的电信号，如正弦波、三角波等



2、数字信号

---在时间上和数值上均是离散的信号。

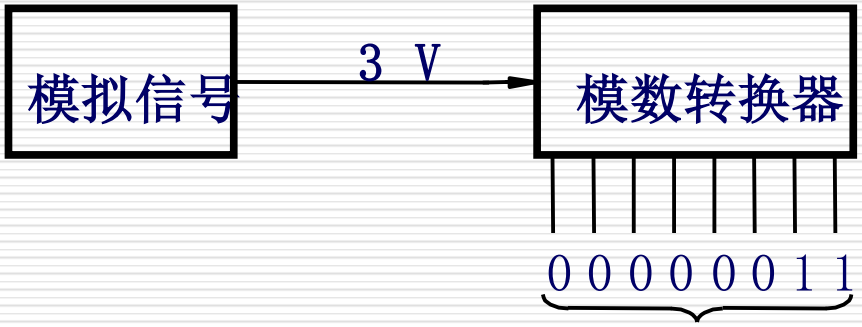


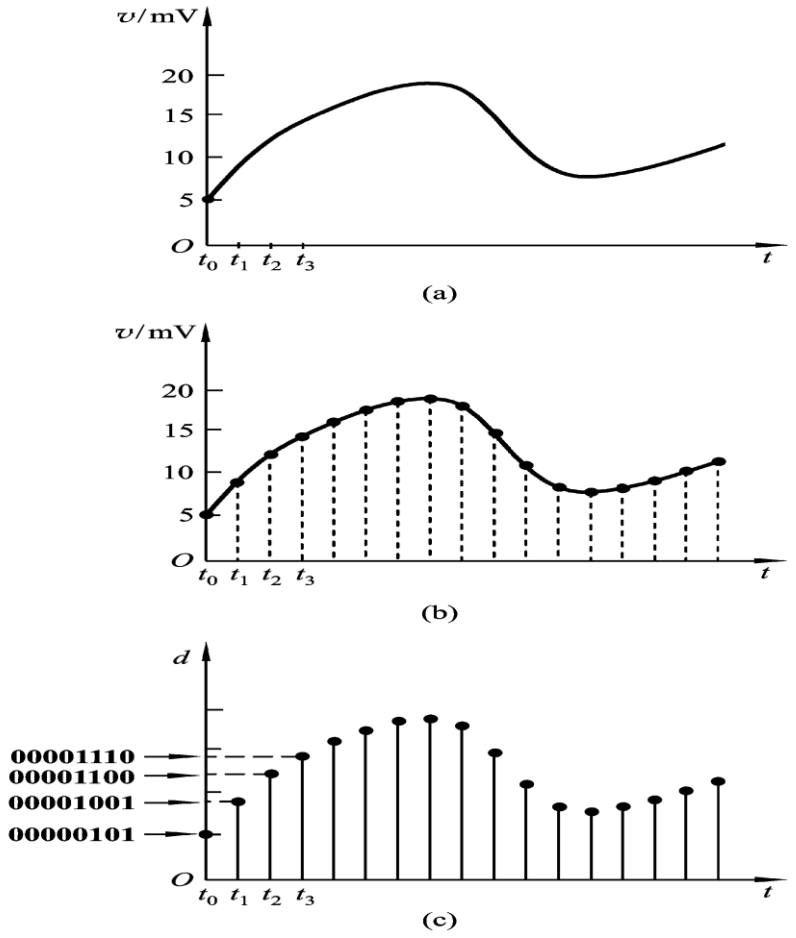
•数字电路和模拟电路：工作信号，研究的对象不同，分析、设计方法以及所用的数学工具也相应不同

3、模拟信号的数字表示

由于数字信号便于存储、分析和传输，通常都将模拟信号转换为数字信号.

模数转换的实现





1.1.4 数字信号的描述方法

1、二值数字逻辑和逻辑电平

- 二值数字逻辑

0、1数码---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

a 、在电路中用低、高电平表示0、1两种逻辑状态

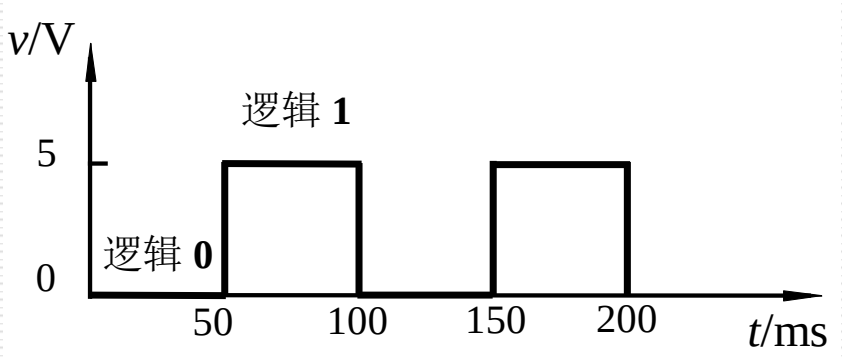
逻辑电平与电压值的关系（正逻辑）

电压(V)	二值逻辑	电 平
+5	1	H(高电平)
0	0	L(低电平)

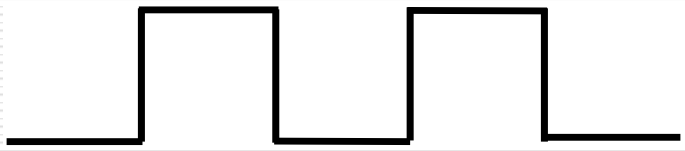
2、数字波形

数字波形-------是信号逻辑电平对时间的图形表示.

(a) 用逻辑电平描述的数字波形

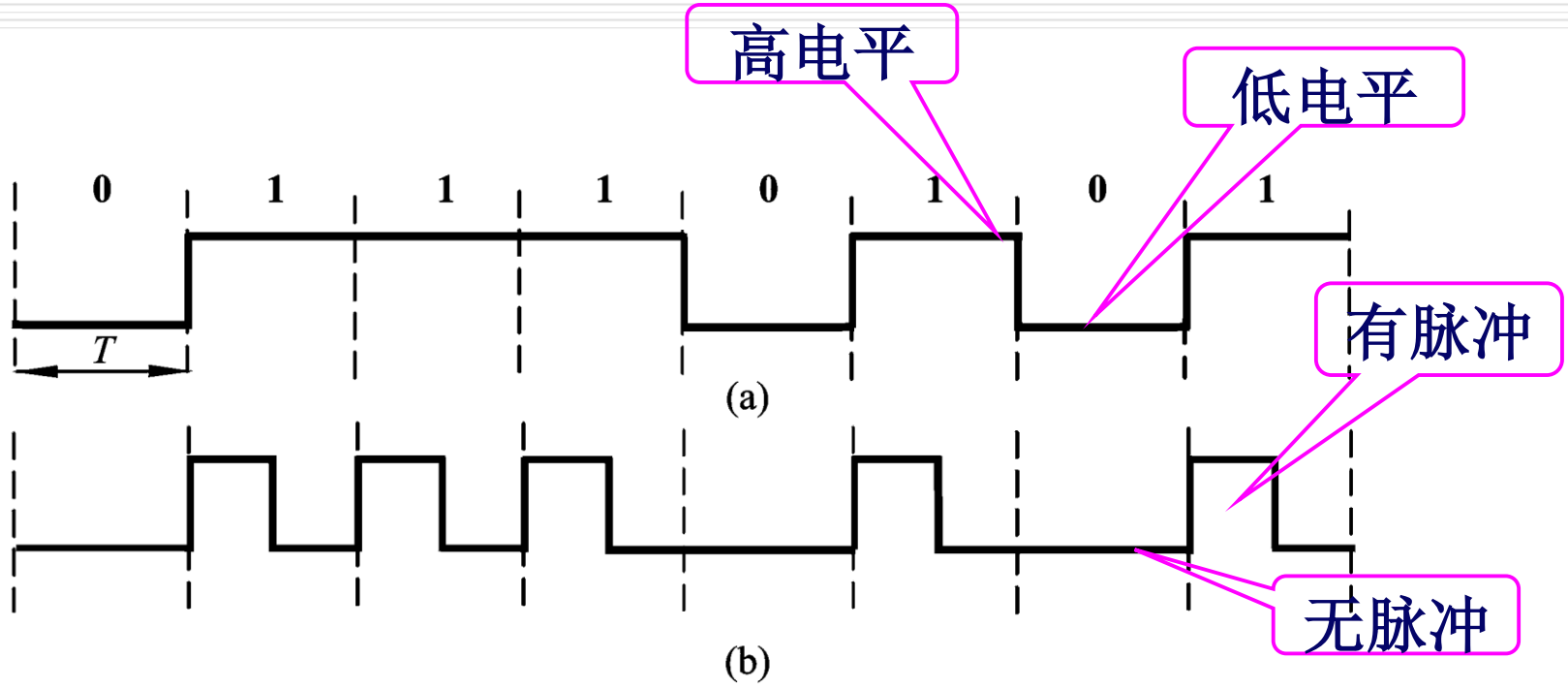


(b) 数字波形的常规表示



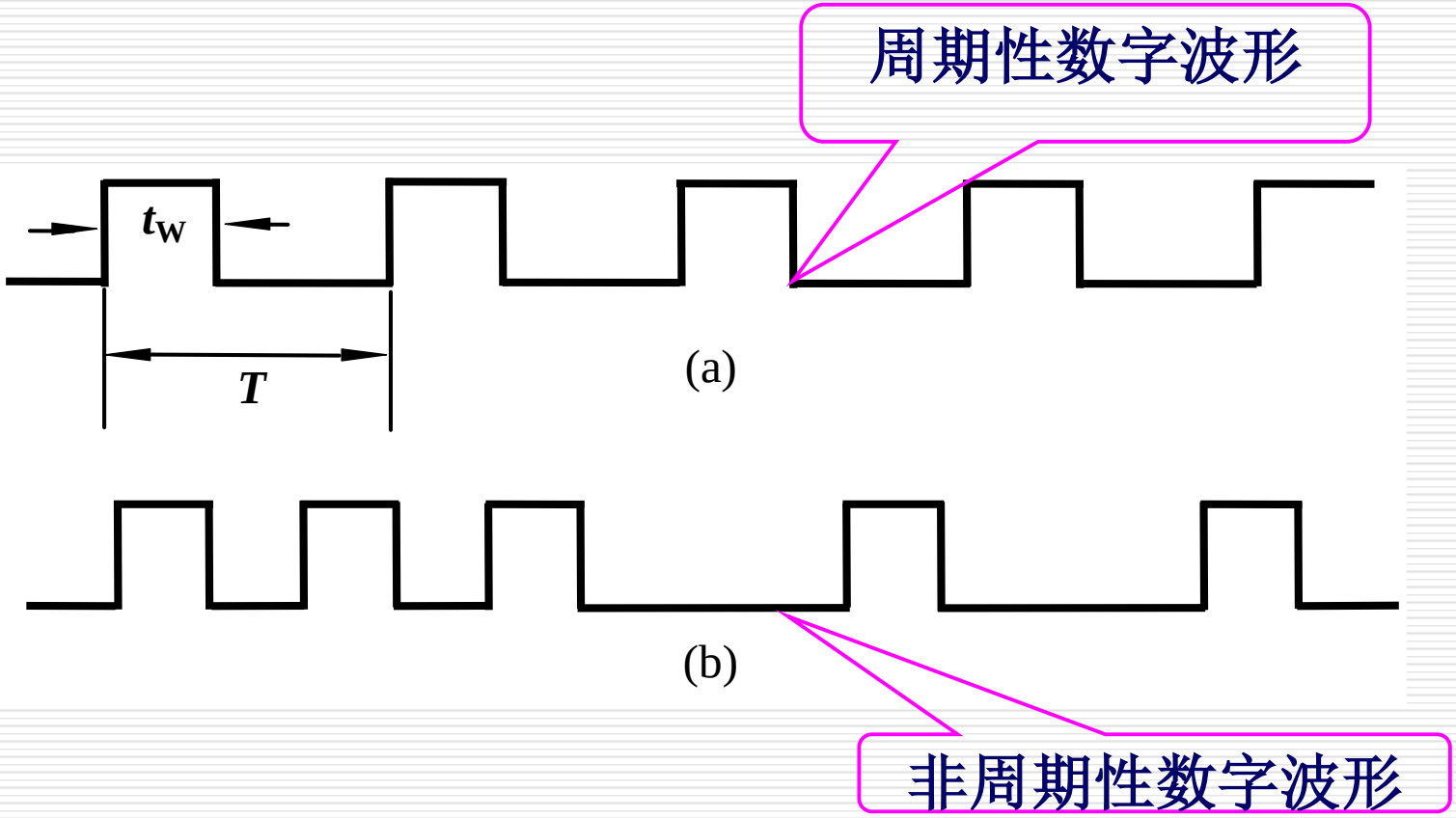
(1)数字波形的两种类型：

*非归零型　　*归零型



比特率 ---------- 每秒钟转输数据的位数

(2) 周期性和非周期性



$$q(\%) = \frac{t_W}{T} \times 100\%$$

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

$$\left[\frac{1.544\times10^{6}}{\mathrm{s}}\right]^{-1}=647.67\times10^{-9}\,\mathrm{s}=648\mathrm{ns}$$

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。

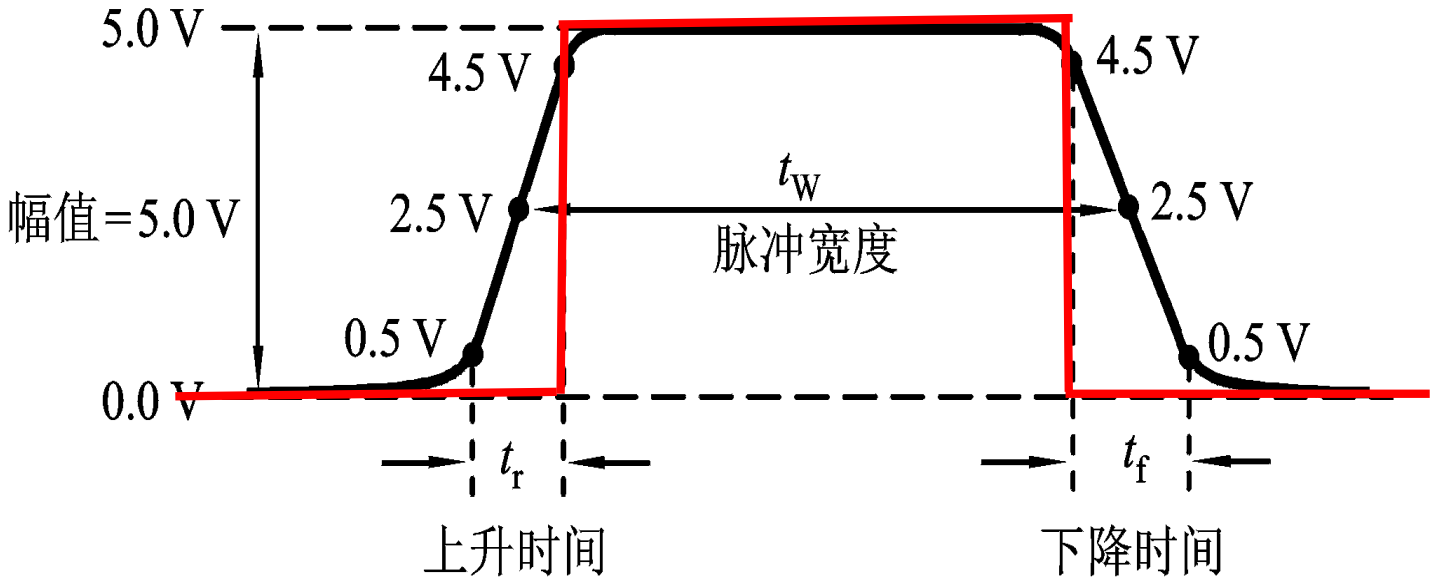
解：因数字波形的脉冲宽度t_w=6ms，周期T=6ms+10ms=16ms。

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

(3) 实际脉冲波形及主要参数

理想脉冲波形

非理想脉冲波形



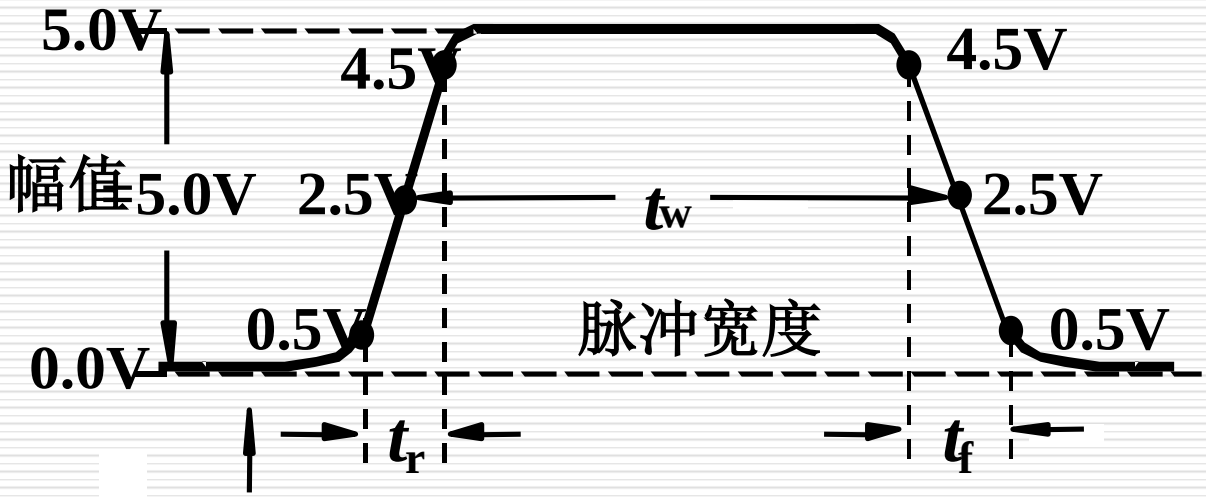
几个主要参数：

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 ）



1.2 数制

1.2 数制

数制:多位数码中的每一位数的构成及低位向高位进位的规则

1.2.1十进制

十进制采用0, 1, 2, 3, 4, 5, 6, 7, 8, 9十个数码，其进位的规则是“逢十进一”。

$$4587.29=4\times10^{3}+5\times10^{2}+8\times10^{1}+7\times10^{0}+2\times10^{-1}+9\times10^{-2}$$

系数　　位权

一般表达式:

$$(N)_D=\sum_{i=-\infty}^{\infty}K_i\times10^{i}$$

各位的权都是10的幂。

任意进制数的一般表达式为:

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

1.2.2　二进制

1、二进制数的表示方法

二进制数只有0、1两个数码，进位规律是：“逢二进一”.

例如：1+1= 10 $= 1\times 2^1 + 0\times 2^0$

二进制数的一般表达式为：

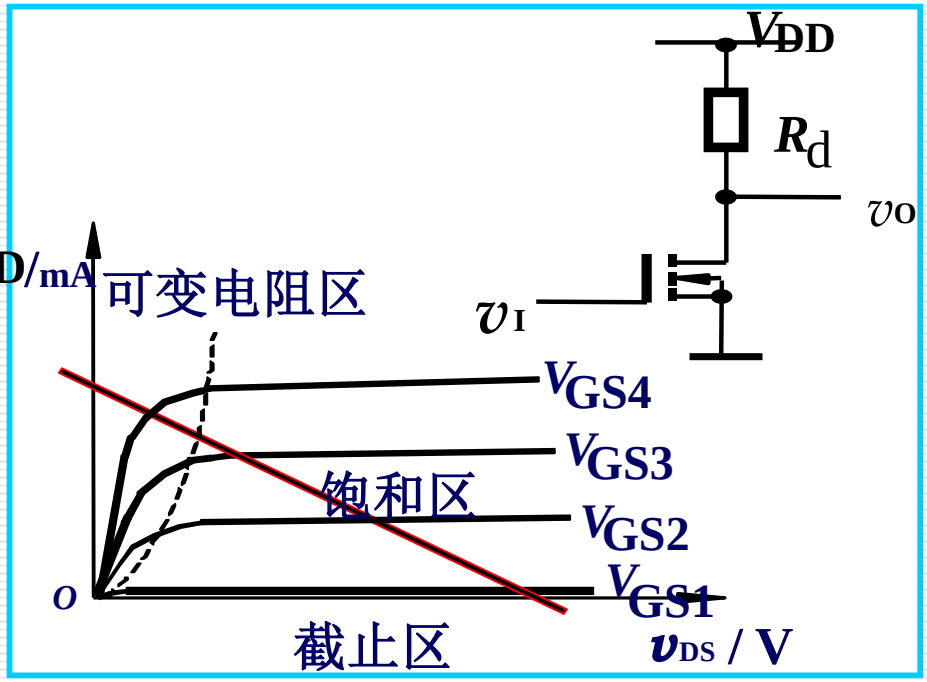
系数　　位权

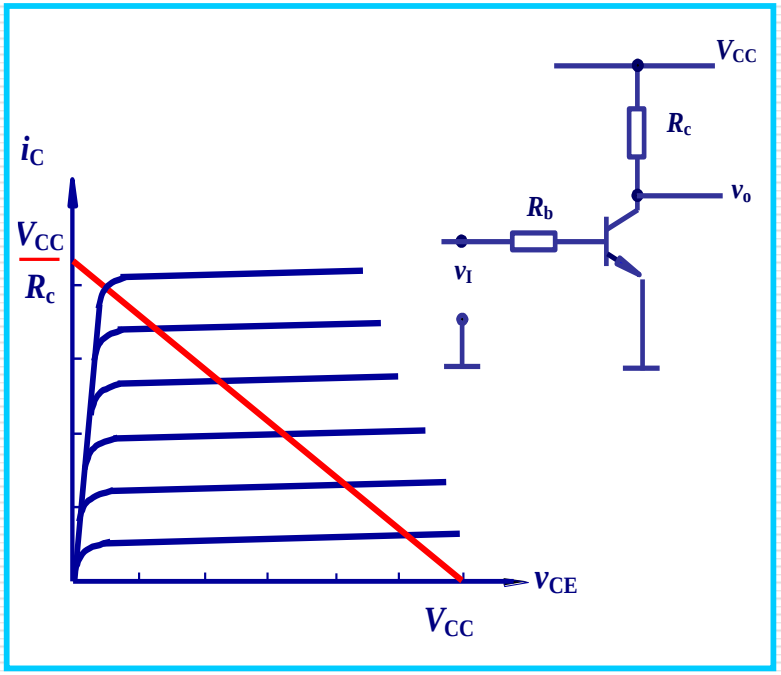
$$(N)_B = \sum_{i=-\infty}^{\infty} K_i \times 2^i$$

各位的权都是2的幂。

2、 二进制的优点

（1）易于电路表达---0、1两个值，可以用管子的导 通或截止，灯泡的亮或灭、继电器触点的闭合或断开来表示。

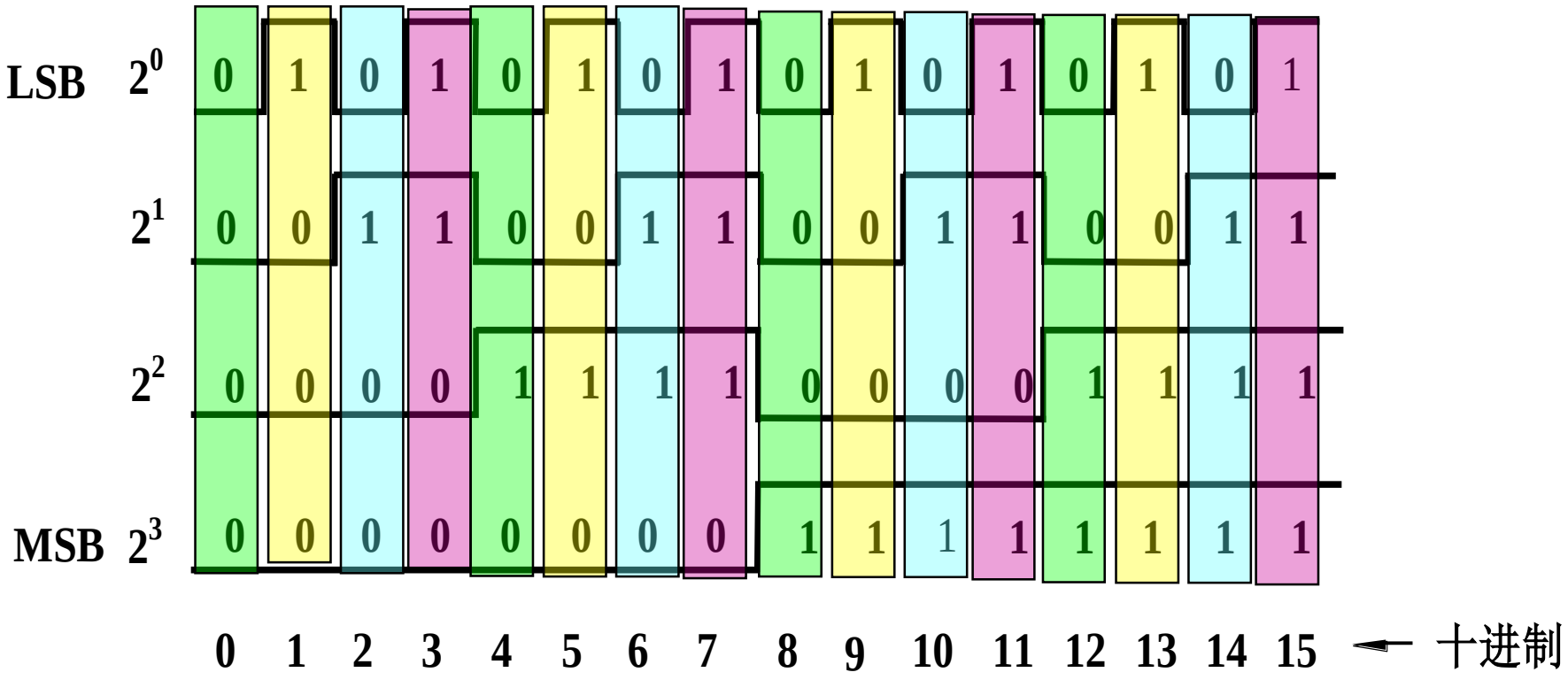




（2）二进制数字装置所用元件少,电路简单、可靠 。

（3）基本运算规则简单, 运算操作方便。

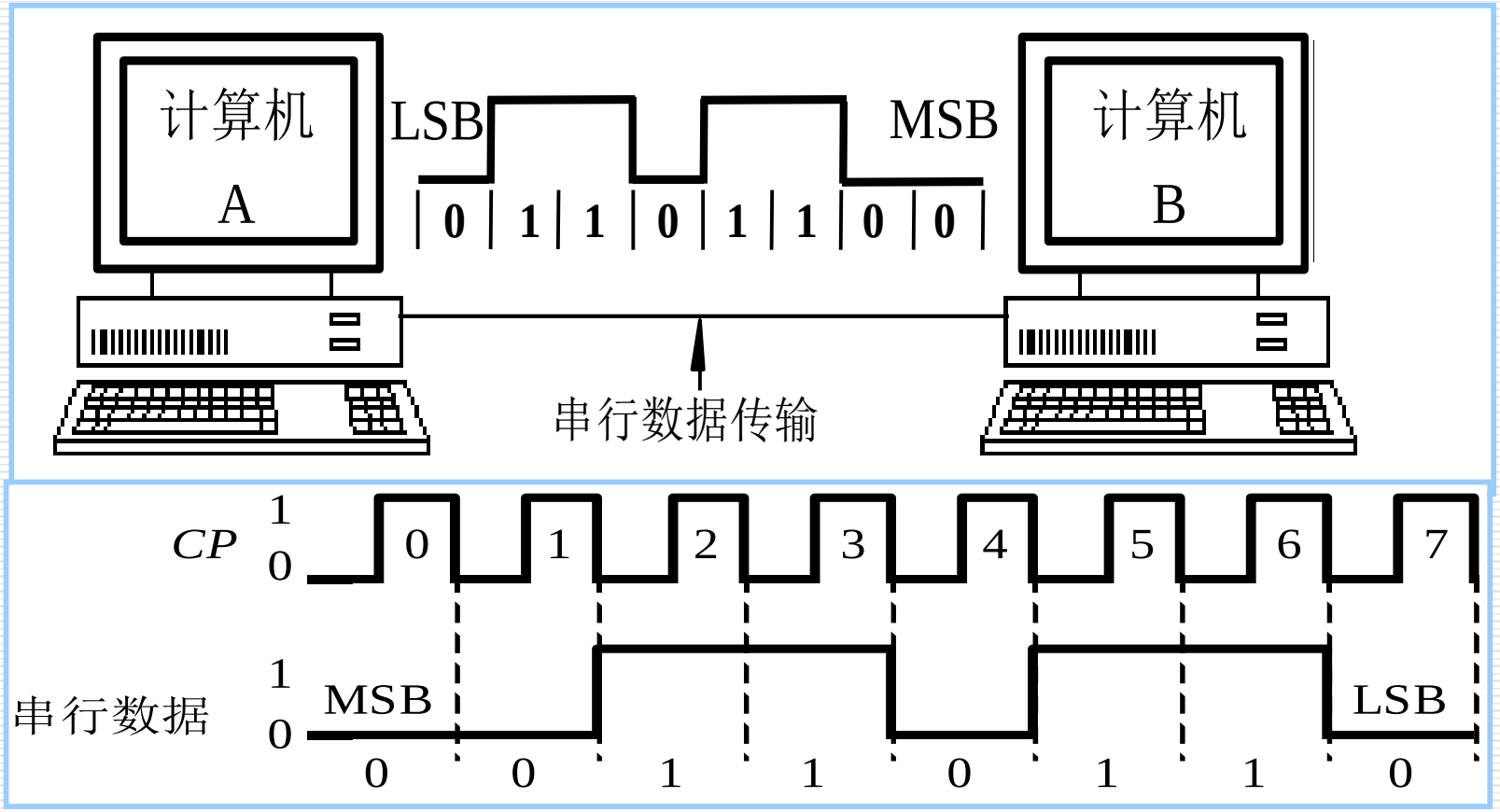
3、二进制数波形表示



4、 二进制数据的传输

（1）二进制数据的串行传输

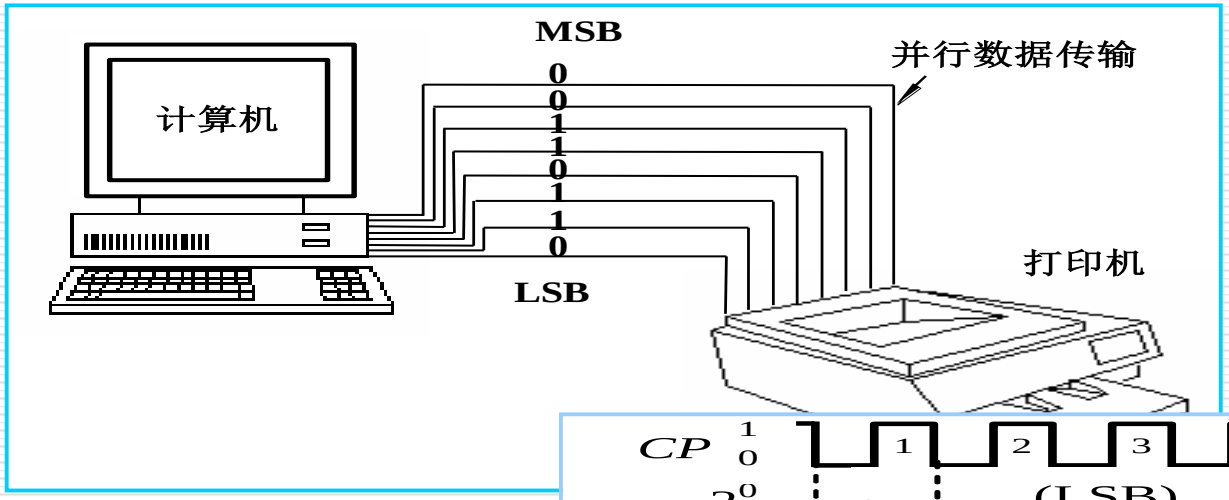
在时钟脉冲CP控制下，数据由最高位MSB到最低位LSB 逐位送。

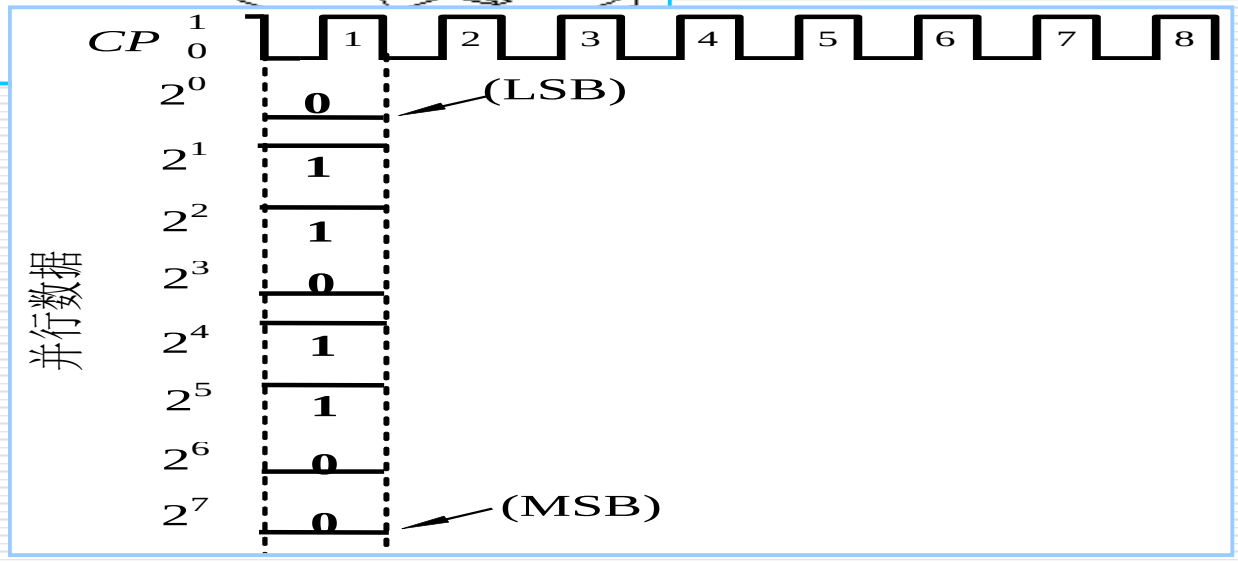


（2）二进制数据的并行传输

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 二-十进制之间的转换

1)、十进制数转换成二进制数：{ 整数部分 / 小数部分

a. 整数的转换：

“辗转相除”法:将十进制数连续不断地除以2 ， 直至商为零，所得余数由低位到高位排列，即为所求二进制数

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
2	18	…… 余	……b_1
2	9	…… 余	……b_2
2	4	…… 余	……b_3
2	2	…… 余	……b_4
2	1	…… 余	……b_5
	0		

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，所以得

$(133)_D=(10000101)_B$

b. 小数的转换：

对于二进制的小数部分可写成

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

将上式两边分别乘以2，得

$$2 \times (N)_{\mathrm{D}} = b_{-1} \times 2^{0} + b_{-2} \times 2^{-1} + \cdots + b_{-(\mathrm{n}-1)} \times 2^{-(\mathrm{n}-2)} + b_{-n} \times 2^{-(\mathrm{n}-1)}$$

由此可见，将十进制小数乘以2，所得乘积的整数即为 b_{-1}

不难推知，将十进制小数每次除去上次所得积中的整数再乘以2，直到满足误差要求进行“四舍五入”为止，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$(0.39)_D$转换成二进制数,要求精度达到1%

解　由于精度要求达到1%，需要精确到二进制小数7位，即$1/2^7=1/128$。

$0.39\times2=0.78$　$b_{-1}=0$

$0.78\times2=1.56$　$b_{-2}=1$

$0.56\times2=1.12$　$b_{-3}=1$

$0.12\times2=0.24$　$b_{-4}=0$

$0.24\times2=0.48$　$b_{-5}=0$

$0.48\times2=0.96$　$b_{-6}=0$

$0.96\times2=1.92$　$b_{-7}=1$

$0.92\times2=1.84$　$b_{-8}=1$

计算时要多算1位，然后考虑“4舍5入”。$b_{-8}=1$产生进位。

所以　$(0.39)_D=(0.0110010)_B$

1.2.4 十六进制和八进制

1.十六进制

十六进制数中只有0, 1, 2, 3, 4, 5, 6, 7, 8, 9 , A、B、C、D、E、F十六个数码，进位规律是“逢十六进一”。各位的权均为16的幂。

例如 $(A6.C)_H = 10\times16^1 + 6\times16^0 + 12\times16^{-1}$

一般表达式：$$(N)_H = \sum_{i=-m}^{n-1} a_i \times 16^i$$

各位的权都是16的幂。

2、二--十六进制之间的转换

二进制转换成十六进制：

因为16进制的基数16=2^4，所以，可将四位二进制数表示一位16进制数，即 0000～1111 表示 0-F。

例 $(1111000101011110)_B$ = $(78AE)_H$

十六进制转换成二进制：

将每位16进制数展开成四位二进制数，排列顺序不变即可。

例 $(BEEF)_H$ = $(1011\ 1110\ 1110\ 1111)_B$

3.八进制

八进制数中只有0, 1, 2, 3, 4, 5, 6, 7八个数码，进位规律是“逢八进一”。各位的权都是8的幂。

八进制就是以8为基数的计数体制。

一般表达式 $$(N)_8 = \sum_{i=-m}^{n-1} a_i \times 8^i$$

4、二-八进制之间的转换

- 因为八进制的基数$8=2^3$，所以，可将三位二进制数表示一位八进制数，即 000～111 表示 0～7

- 转换时，由小数点开始，整数部分自右向左，小数部分自左向右，三位一组，不够三位的添零补齐，则每三位二进制数表示一位八进制数。

例 $(10110.011)_B = (26.3)_O$

将每位八进制数展开成三位二进制数，排列顺序不变即可。

例 $(752.1)_O = (111\ 101\ 010.001)_B$

5.十六进制的优点：

1、）与二进制之间的转换容易；

2、）计数容量较其它进制都大。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$(FFFF)_H = (65535)_D$，即64K。其容量最大。

3、）书写简洁。

1.3　二进制的算术运算

1.3.1　无符号二进制的数算术运算

1.3.2　有符号二进制的数算术运算

1.3　二进制的算术运算

1.3.1　无符号数算术运算

1、二进制加法

无符号二进制的加法规则：

0+0=0，0+1=1，1+1=10。

例1.3.1 计算两个二进制数1010和0101的和。

解：

$$\begin{array}{rcccc} & 1 & 0 & 1 & 0 \\ + & 0 & 1 & 0 & 1 \\ \hline & 1 & 1 & 1 & 1 \end{array}$$

2．二进制减法

无符号二进制数的减法规则：

0-0=0， 1-1=0，1-0=1 0-1=11

例1.3.2 计算两个二进制数1010和0101的差。

解：

$$\begin{array}{rcccc} & 1 & 0 & 1 & 0 \\ - & 0 & 1 & 0 & 1 \\ \hline & 0 & 1 & 0 & 1 \end{array}$$

3、乘法和除法

例1.3.3　计算两个二进制数1010和0101的积。

解：

$$
\begin{array}{rcccccccc}
 & & & & 1 & 0 & 1 & 0 \\
 & & & \times & 0 & 1 & 0 & 1 \\
\hline
 & & & & 1 & 0 & 1 & 0 \\
 & & & 0 & 0 & 0 & 0 & \\
 & & 1 & 0 & 1 & 0 & & \\
 & 0 & 0 & 0 & 0 & & & \\
\hline
 & & 1 & 1 & 0 & 0 & 1 & 0
\end{array}
$$

例1.3.4 计算两个二进制数1010和111之商。

解：

$$
\begin{array}{r}
\quad 1.\ 0\ 1\ 1 \\
\hline
111\overline{)\,1\ 0\ 1\ 0} \\
\quad\ 1\ 1\ 1 \\
\hline
\quad\ 1\ 1\ 0\ 0 \\
\quad\ \ 1\ 1\ 1 \\
\hline
\quad\ \ 1\ 0\ 1\ 0 \\
\quad\ \ \ 1\ 1\ 1 \\
\hline
\quad\ \ \ 1\ 1
\end{array}
$$

……余数

1.3.2　带符号二进制的减法运算

有符号的二进制数表示 ：

二进制数的最高位表示符号位，且用0表示正数，用1表示负数。其余部分用原码的形式表示数值位。

$$(+11)_D = (0\ 1011)_B$$
$$(-11)_D = (1\ 1011)_B$$

1. 二进制数的补码表示

补码或反码的最高位为符号位，正数为0，负数为1。

当二进制数为正数时，其补码、反码与原码相同。

当二进制数为负数时，将原码的数值位逐位求反，然后在最低位加1得到补码。

4位二进制原码、反码、补码对照表

n位带符号二进制数的原码、反码和补码的数值范围：

原码 $-(2^{n-1}-1)\sim+(2^{n-1}-1)$

反码 $-(2^{n-1}-1)\sim+(2^{n-1}-1)$

补码 $-2^{n-1}\sim+(2^{n-1}-1)$

十进制数	二进制数		
	原码	反码	补码
−8	—	—	1 0 0 0
−7	1 1 1 1	1 0 0 0	1 0 0 1
−6	1 1 1 0	1 0 0 1	1 0 1 0
−5	1 1 0 1	1 0 1 0	1 0 1 1
−4	1 1 0 0	1 0 1 1	1 1 0 0
−3	1 0 1 1	1 1 0 0	1 1 0 1
−2	1 0 1 0	1 1 0 1	1 1 1 0
−1	1 0 0 1	1 1 1 0	1 1 1 1
−0	1 0 0 0	1 1 1 1	0 0 0 0
+0	0 0 0 0	0 0 0 0	0 0 0 0
+1	0 0 0 1	0 0 0 1	0 0 0 1
+2	0 0 1 0	0 0 1 0	0 0 1 0
+3	0 0 1 1	0 0 1 1	0 0 1 1
+4	0 1 0 0	0 1 0 0	0 1 0 0
+5	0 1 0 1	0 1 0 1	0 1 0 1
+6	0 1 1 0	0 1 1 0	0 1 1 0
+7	0 1 1 1	0 1 1 1	0 1 1 1

2. 二进制补码的减法运算

减法运算的原理:减去一个正数相当于加上一个负数

$A-B=A+(-B)$，对$(-B)$求补码，然后进行加法运算。

例1.3.7　试用4位二进制补码计算5–2。

解：因为 $(5-2)_{补}=(5)_{补}+(-2)_{补}$

$=0101+1110$

$=0011$

所以 5–2=3

$$
\begin{array}{rcccc}
 & 0 & 1 & 0 & 1 \\
+ & 1 & 1 & 1 & 0 \\
\hline
\boxed{1} & 0 & 0 & 1 & 1
\end{array}
$$

自动丢弃

3. 溢出

例1.3.8 试用4位二进制补码计算5+7。

解：因为$(5+7)_{补}=(5)_{补}+(7)_{补}$

=0101+0111

=1100

$$\begin{array}{rcccc} & 0 & 1 & 0 & 1 \\ + & 0 & 1 & 1 & 1 \\ \hline & [1] & 1 & 0 & 0 \end{array}$$

解决溢出的办法:进行位扩展.

4. 溢出的判别

如何判断是否产生溢出？

+ 4	0 1 0 0	− 5	1 0 1 1
+) + 3	+ 0 0 1 1	+) − 3	+ 1 1 0 1
+ 7	[0] 0 1 1 1	− 8	[1] 1 0 0 0
+ 2	0 0 1 0	− 3	1 1 0 1
+) + 6	+ 0 1 1 0	+) − 6	+ 1 0 1 0
+ 8	[0] 1 0 0 0	− 9	[1] 0 1 1 1

当方框中的进位位与和数的符号位（即b_3位）相反时，则运算结果是错误的，产生溢出。

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 二进制代码

码制:编制代码所要遵循的规则

二进制代码的位数(n),与需要编码的事件（或信息）的个数(N)之间应满足以下关系：　　$2^{n-1} \leq N \leq 2^n$

1. 二—十进制码 (数值编码)

(BCD码----- Binary Code Decimal）

用4位二进制数来表示一位十进制数中的0~9十个数码。

从4 位二进制数16种代码中,选择10种来表示0~9个数码的方案有很多种。每种方案产生一种BCD码。

1.4.1二-十进制码

（1）几种常用的BCD代码

BCD码十进制数码	8421码	2421 码	5421 码	余3码	余3循环码
0	0000	0000	0000	0011	0010
1	0001	0001	0001	0100	0110
2	0010	0010	0010	0101	0111
3	0011	0011	0011	0110	0101
4	0100	0100	0100	0111	0100
5	0101	1011	1000	1000	1100
6	0110	1100	1001	1001	1101
7	0111	1101	1010	1010	1111
8	1000	1110	1011	1011	1110
9	1001	1111	1100	1100	1010

（2）各种编码的特点

有权码：编码与所表示的十进制数之间的转换容易

如$(10010000)_{8421BCD}=(90)_D$

余 3 码的特点:当两个十进制的和是10时，相应的二进制正好是16，于是可自动产生进位信号,而不需修正.1和9, 2和8,…..6和4的余 3 码。便于求10的补码。

余3码循环码：相邻的两个代码之间仅一位的状态不同。按余3码循环码组成计数器时，每次转换过程只有一个触发器翻转，译码时不会发生竞争一冒险现象。

(4)求BCD代码表示的十进制数

对于有权BCD码，可以根据位权展开求得所代表的十进制数。例如：

$$[0111]_{8421BCD} = 0\times 8 + 1\times 4 + 1\times 2 + 1\times 1 = (7)_D$$

$$[1101]_{2421BCD} = 1\times 2 + 1\times 4 + 0\times 2 + 1\times 1 = (7)_D$$

(3)用BCD代码表示十进制数

对于一个多位的十进制数，需要有与十进制位数相同的几组BCD代码来表示。例如：

$$(463.5)_{10} = [\underset{4}{0100}\ \underset{6}{0110}\ \underset{3}{0011}.\ \underset{5}{0101}]_{8421BCD}$$

不能省略！

$$(863.2)_{10} = [\underset{8}{1110}\ \underset{6}{1100}\ \underset{3}{0011}.\ \underset{2}{0010}]_{2421BCD}$$

不能省略！

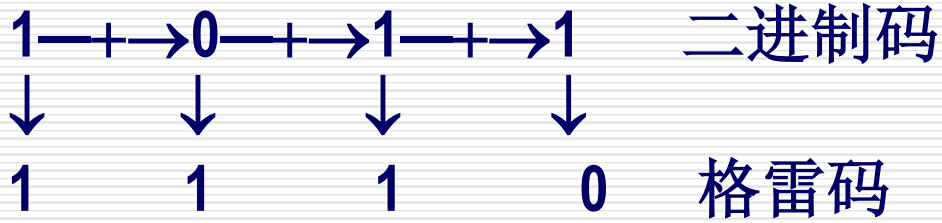
1.4.2 格 雷 码

- 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。

二进制码 $b_3b_2b_1b_0$	格雷码 $G_3G_2G_1G_0$
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

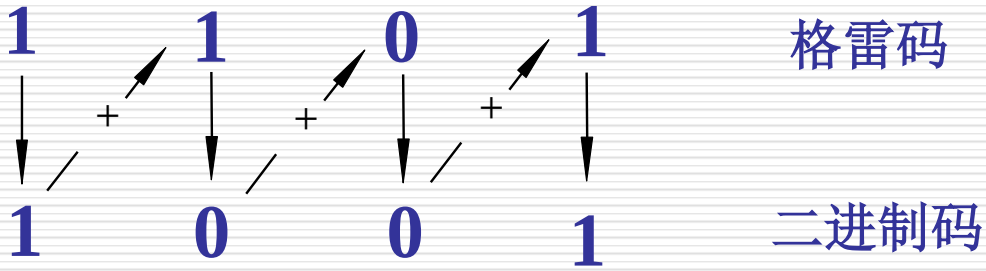
二进制码到格雷码的转换

（1）格雷码的最高位（最左边）与二进制码的最高位相同。
（2）从左到右，逐一将二进制码相邻的两位相加（舍去进位），作为格雷码的下一位。



格雷码到二进制码的转换

（1）二进制码的最高位（最左边）与格雷码的最高位相同。
（2）将产生的每一位二进制码，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。



1.4.3 ASCII 码(字符编码)

- 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。

1.5 二值逻辑变量与基本逻辑运算

*逻辑运算： 当0和1表示逻辑状态时，两个二进制数码按照某种特定的因果关系进行的运算。

逻辑运算使用的数学工具是逻辑代数。

* 逻辑代数与普通代数:与普通代数不同,逻辑代数中的变量只有0和1两个可取值，它们分别用来表示完全两个对立的逻辑状态。

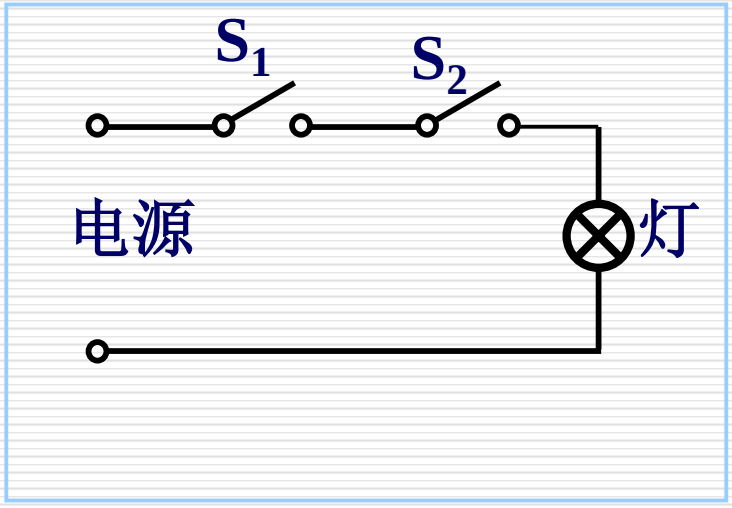
在逻辑代数中，有与、或、非三种基本的逻辑运算。

逻辑运算的描述方式:逻辑代数表达式、真值表、逻辑图、卡诺图、波形图和硬件描述语言（HDL) 等。

1．与运算

(1) 与逻辑:只有当决定某一事件的条件全部具备时，这一事件才会发生。这种因果关系称为与逻辑关系。

与逻辑举例



电路状态表

开关S_1	开关S_2	灯
断	断	灭
断	合	灭
合	断	灭
合	合	亮

1．与运算

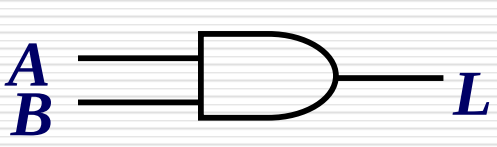
与逻辑举例状态表

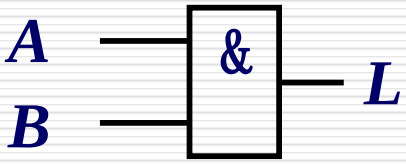
开关S_1	开关S_2	灯
断	断	灭
断	合	灭
合	断	灭
合	合	亮

逻辑真值表

A	*B*	*L*
0	0	0
0	1	0
1	0	0
1	1	1

与逻辑符号



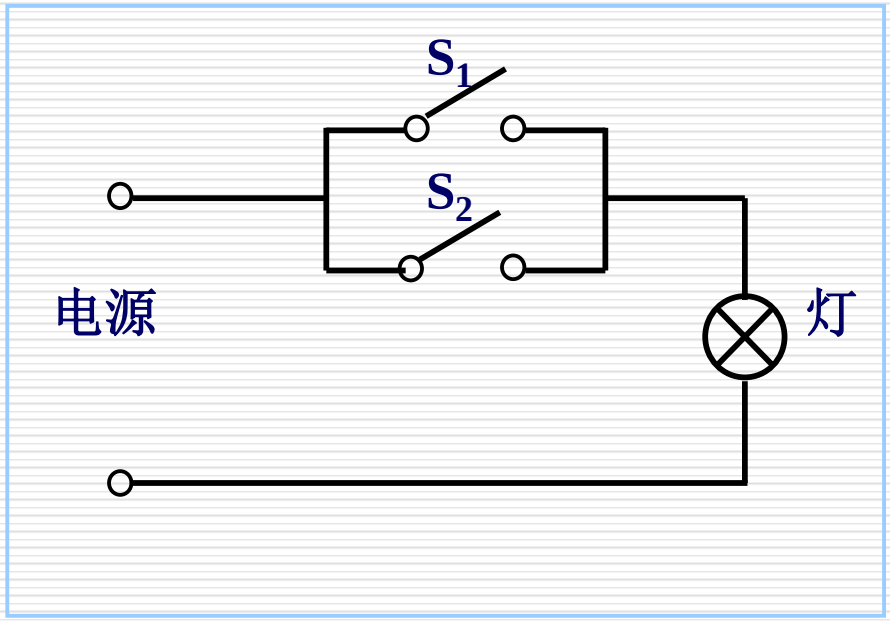


逻辑表达式　　与逻辑：　$L = A \cdot B = AB$

2、或运算

只要在决定某一事件的各种条件中，有一个或几个条件具备时，这一事件就会发生。这种因果关系称为或逻辑关系。

或逻辑举例



电路状态表

开关S_1	开关S_2	灯
断	断	灭
断	合	亮
合	断	亮
合	合	亮

2、或运算

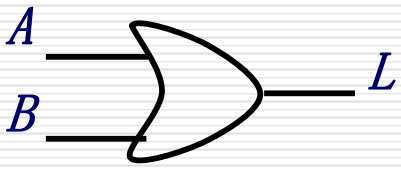
或逻辑举例状态表

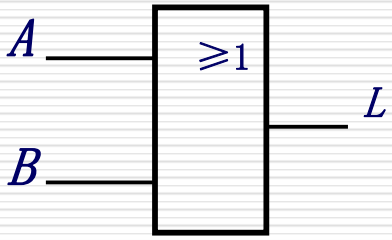
开关S_1	开关S_2	灯
断	断	灭
断	合	灭
合	断	灭
合	合	亮

逻辑真值表

A	*B*	*L*
0	0	0
0	1	1
1	0	1
1	1	1

或逻辑符号



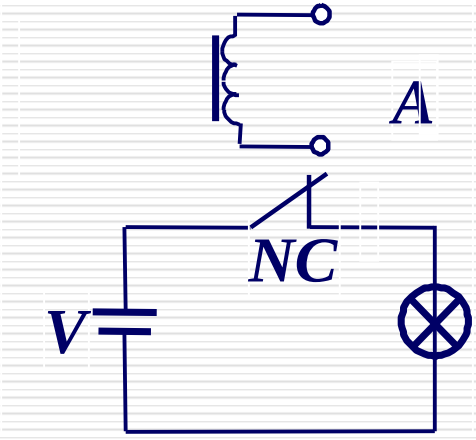


逻辑表达式　　或逻辑：　$L = A + B$

3. 非运算

事件发生的条件具备时，事件不会发生；事件发生的条件不具备时，事件发生。这种因果关系称为非逻辑关系。

非逻辑举例



非逻辑举例状态表

A	灯
不通电	亮
通电	灭

3. 非运算

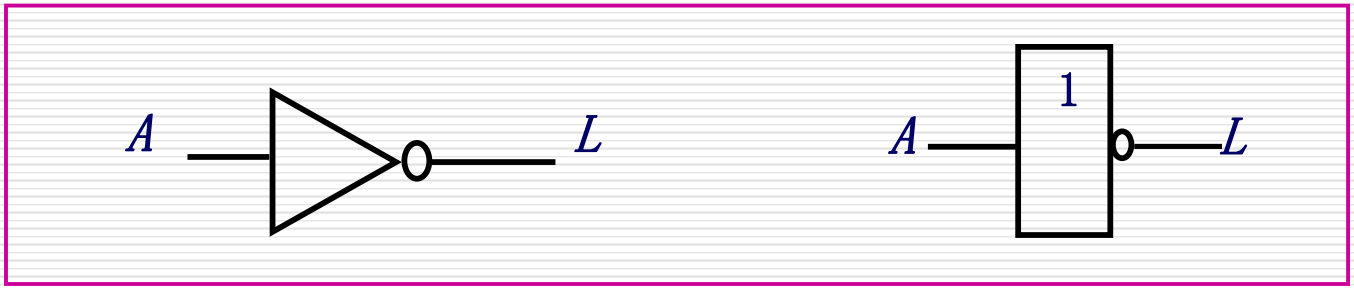
非逻辑举例状态表

A	灯
不通电	亮
通电	灭

非逻辑真值表

A	L
0	1
1	0

非逻辑符号



逻辑表达式 $L = \overline{A}$

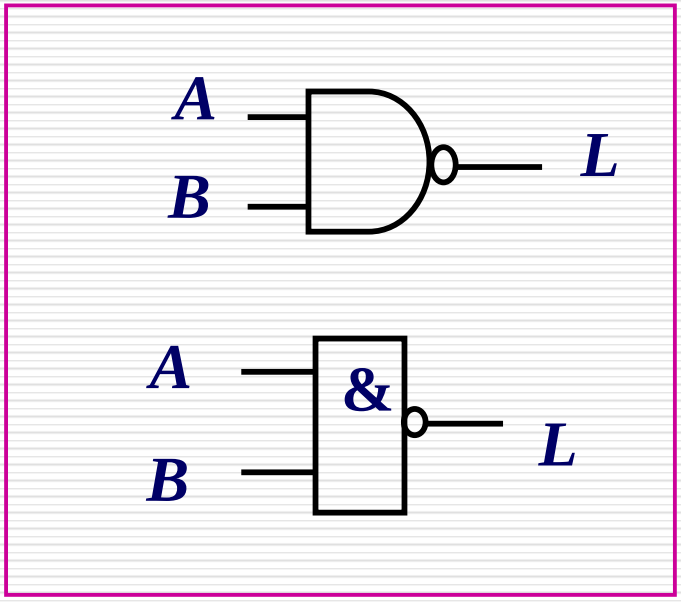
4. 几种常用复合逻辑运算

1)与非运算

两输入变量与非逻辑真值表

A	B	L
0	0	1
0	1	1
1	0	1
1	1	0

与非逻辑符号



与非逻辑表达式

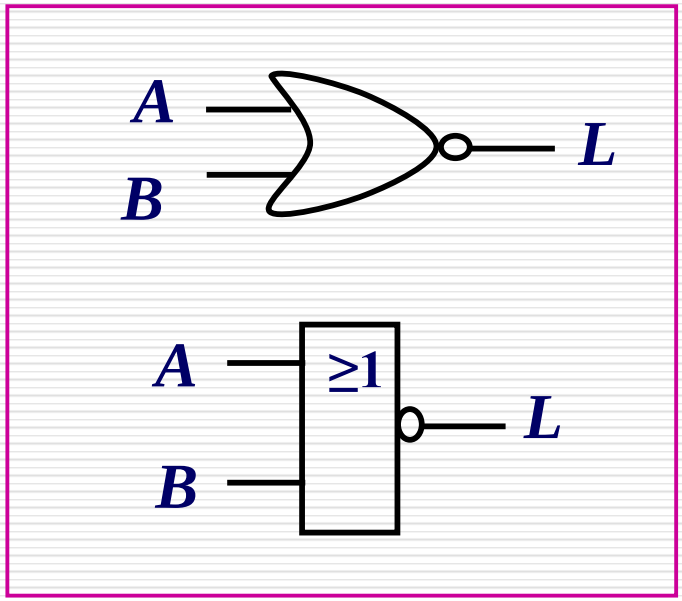
$$L = \overline{A \cdot B}$$

2)或非运算

或非逻辑符号

两输入变量或非逻辑真值表

A	B	L
0	0	1
0	1	0
1	0	0
1	1	0



或非逻辑表达式

$$L = \overline{A+B}$$

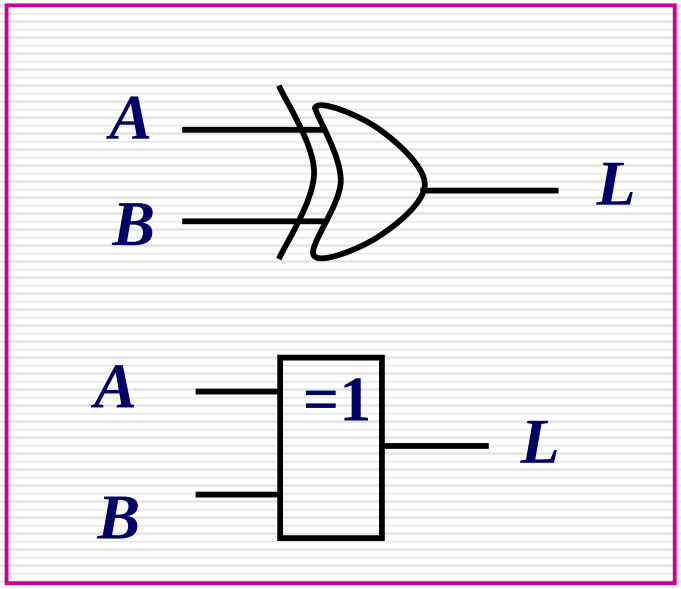
3） 异或逻辑

若两个输入变量的值相异，输出为1，否则为0。

异或逻辑真值表

A	B	L
0	0	0
0	1	1
1	0	1
1	1	0

异或逻辑符号



异或逻辑表达式 $L = A \oplus B = \overline{A}B + A\overline{B}$

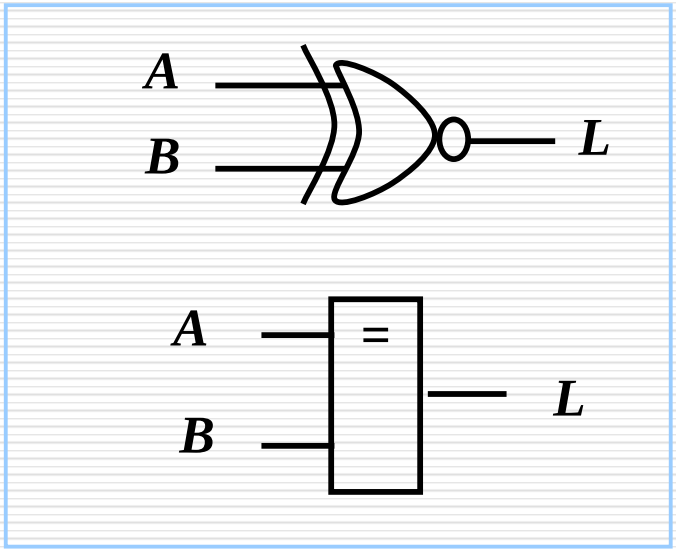
4） 同或运算

若两个输入变量的值相同，输出为1，否则为0。

同或逻辑真值表

A	B	L
0	0	1
0	1	0
1	0	0
1	1	1

同或逻辑逻辑符号



同或逻辑表达式

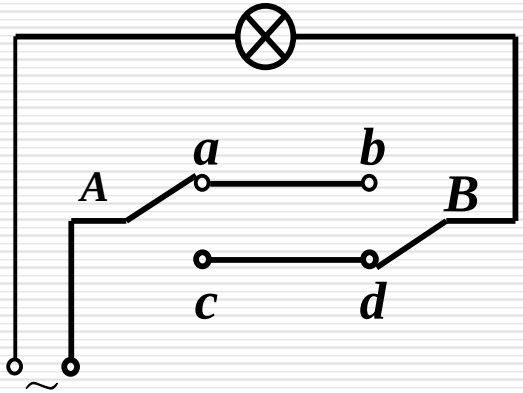
$$L=AB+\overline{A}\,\overline{B}=A\odot B$$

1.6 逻辑函数的建立及其表示方法

1.6.1 逻辑函数几种表示方法

1. 真值表表示

楼道灯开关示意图



确定变量、函数，并赋值

开关： 变量 A、B
灯 ： 函数 L

A、B: 向上—1 向下--0
L ：亮---1; 灭---0

逻辑抽象，列出真值表

开关状态表

开关 A	开关 B	灯
下	下	亮
下	上	灭
上	下	灭
上	上	亮

逻辑真值表

A	B	L
0	0	1
0	1	0
1	0	0
1	1	1

2、逻辑函数表达式表示。

逻辑表达式是用与、或、非等运算组合起来，表示逻辑函数与逻辑变量之间关系的逻辑代数式。

例：已知某逻辑函数的真值表，试写出对应的逻辑函数表达式。

$$L = \overline{A}\,\overline{B} + AB$$

逻辑真值表

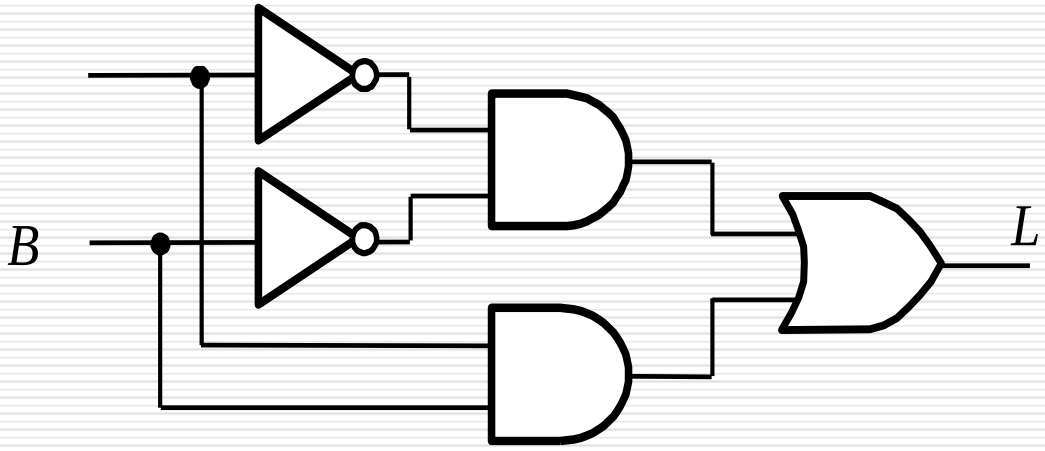
A	B	L
0	0	1
0	1	0
1	0	0
1	1	1

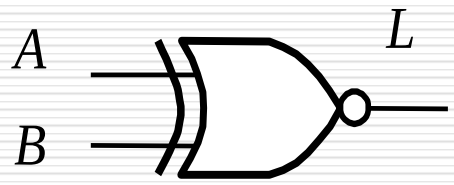
3. 逻辑图表示方法

用与、或、非等逻辑符号表示逻辑函数中各变量之间的逻辑关系所得到的图形称为逻辑图。

将逻辑函数式中所有的与、或、非运算符号用相应的逻辑符号代替，并按照逻辑运算的先后次序将这些逻辑符号连接起来，就得到图电路所对应的逻辑图

例：已知某逻辑函数表达式为 $L = \overline{A}\,\overline{B} + AB$ 试画出其逻辑图



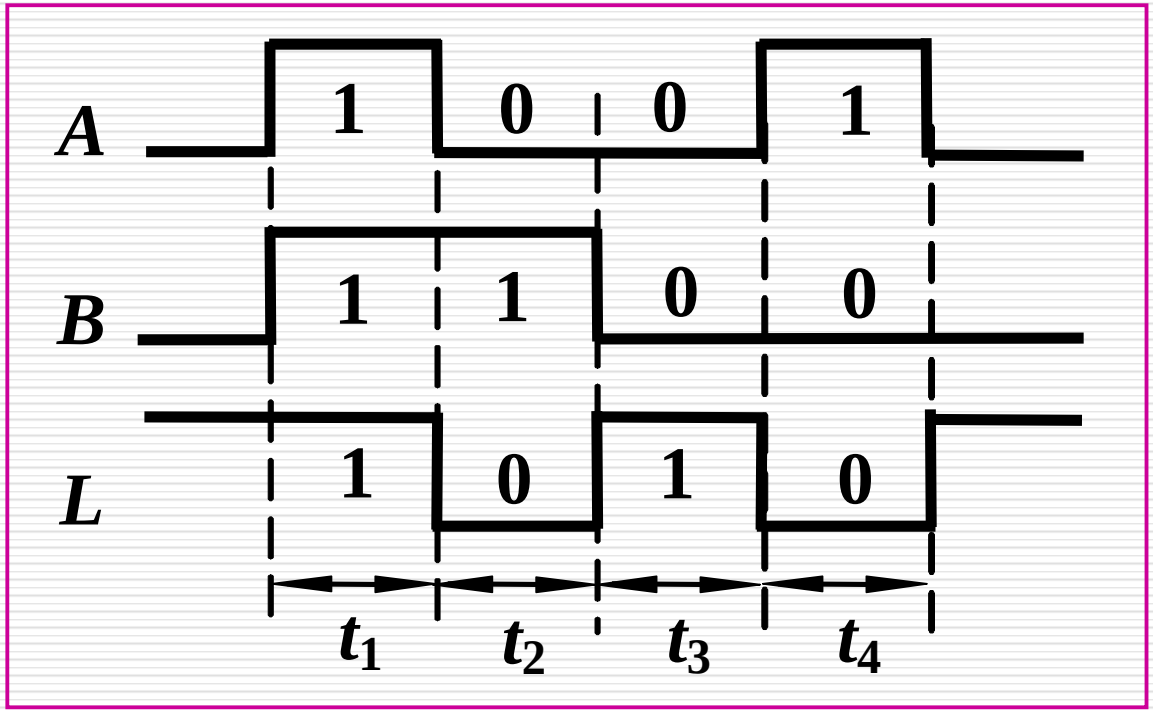


4. 波形图表示方法

用输入端在不同逻辑信号作用下所对应的输出信号的波形图，表示电路的逻辑关系。

真值表

A	B	L
0	0	1
0	1	0
1	0	0
1	1	1



1.6.2 逻辑函数表示方法之间的转换

逻辑函数的真值表、逻辑函数表达式、逻辑图、波形图、卡诺图及HDL描述之间可以相互转换。这里介绍两种转换。

1.真值表到逻辑图的转换

真值表如右表。

A	B	C	L
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

转换步骤：

(1)根据真值表写出逻辑表达式

$$L = \overline{A}BC + A\overline{B}C$$

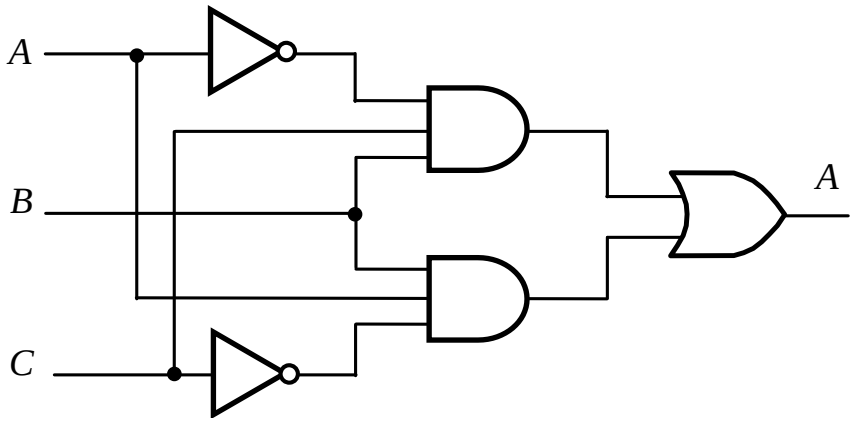
(2)化简逻辑表达式（第2章介绍）

上式不需要简化

(3)根据与或逻辑表达式画逻辑图

$$L = \overline{A}BC + AB\overline{C}$$

用与、或、非符号代替相应的逻辑符号，注意运算次序。



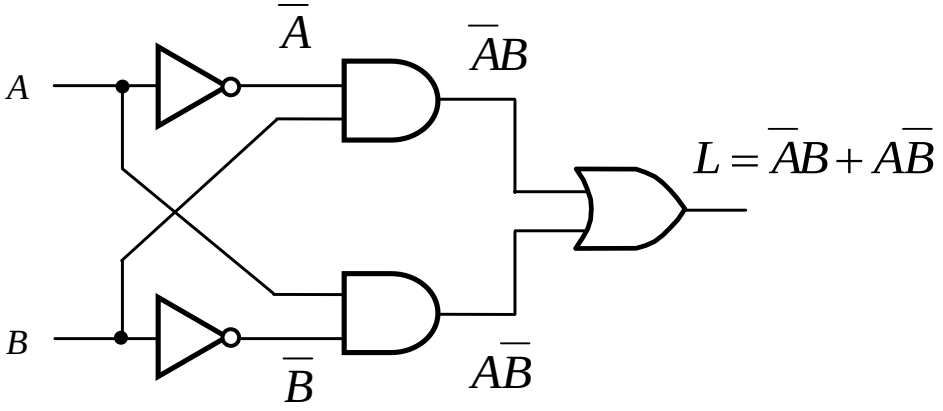
2. 逻辑图到真值表的转换

转换步骤：

(1)根据逻辑图逐级写出表达式

(2)化简变换求最简与或式

(3)将输入变量的所有取值逐一代入表达式得真值表



A	B	L
0	0	0
0	1	1
1	0	1
1	1	0

小　结

●用0和1可以组成二进制数表示是数量的大小，也可以表示对立的两种逻辑状态。数字系统中常用二进制数来表示数值。

● 在微处理器、计算机和数据通信中，采用十六进制。任意一种格式的数可以在十六进制、二进制和十进制之间相互转换。

● 二进制数有加、减、乘、除四种运算，加法是各种运算的基础。特殊二进制码常用来表示十进制数。如8421码、2421码、5421码、余三码、余三码循环码、格雷码等。

● 与、或、非是逻辑运算中的三种基本运算。数字逻辑是计算机的基础。逻辑函数的描述方法有真值表、逻辑函数表达式、逻辑图、波形图和卡诺图等。