高等职业技术学校教学用书

数字电子技术基础

刘培宇　主编

GAODENG ZHIYE JISHU XUEXIAO JIAOXUE YONGSHU

冶金工业出版社

高等职业学校教学用书

数字电子技术基础

主　编　刘培宇
副主编　丁有生

北京
冶金工业出版社
2001

内容提要

本书是参照教育部工科电工类和计算机类中等专业学校《电子技术基础》教学大纲编写的，同时考虑到高等职业技术学校的教学需要，并结合当代数字电子技术的发展和模块化教学改革的需要，对课程体系进行了适当的调整。

全书共分7章。内容主要有基础知识、基本门电路、组合电路、时序电路、脉冲技术、信息的传输变换与存储、计算机辅助设计与制作。

本书着重介绍物理概念和基本原理，避免繁琐的数学推导和计算。在内容编排上以中小规模的数字集成电路为主，增加GAL、CMOS—LED组合器件、语音电路、多功能门等新颖器件，反映了CMOS电路新的技术成就。讲述电路时，注重逻辑功能和器件的使用，介绍必要的实践操作，并增加适量的操作练习题。最后介绍了电子制作中必用的软件工具PROTEL，GAL编程必用的MF语言及编程器硬件，附录中还提供了足够的器件资料，为突出技能本位的教学提供条件。

本书可作为高等职业技术学校、中等专业学校、技工学校、职业高中相关专业的教材，也可供电子爱好者自学。

图书在版编目(CIP)数据

数字电子技术基础/刘培宇主编. —北京：冶金工业出版社，2000.4 (2001.6 重印)
高等职业技术学校教学用书
ISBN 7-5024-2329-X

Ⅰ.数… Ⅱ.刘… Ⅲ.电子电路-高等学校；技术学校-教材 Ⅳ.TN710

中国版本图书馆 CIP 数据核字(2000)第 03513 号

出版人 卿启云(北京沙滩嵩祝院北巷 39 号，邮编 100009)
责任编辑 俞跃春 美术编辑 李心 责任校对 王贺兰
北京顺义兴华印刷厂印刷；冶金工业出版社发行；各地新华书店经销
2000 年 4 月第 1 版，2001 年 6 月第 2 次印刷
787mm×1092mm 1/16；23.5 印张；563 千字；366 页；3001～5000 册
29.50 元
冶金工业出版社发行部 电话：(010)64044283 传真：(010)64044283
冶金书店 地址：北京东四西大街 46 号(100711) 电话：(010)65289081
(本社图书如有印装质量问题，本社发行部负责退换)

前　言

本教材在编写过程中，力图使教材内容突出职业技能本位的思想，突出应用，有利于提高学生制作与解决实践问题的能力。在介绍每一个电路时，都注重基本概念、基本原理和基本方法，着眼于器件的功能和使用，避免繁琐的数学推导与计算，在文字上力求通俗易懂，便于自学。本书以 CMOS 集成电路为主兼及 TTL 和其他类型数字集成电路，避免了复杂 TTL 集成电路的分析，易于学生掌握和理解，同时也适应了 CMOS 主流的趋势。为突出实际应用，每章配有例题与习题，包括实践题，而且还介绍各类数字集成电路的型号和参数，管脚图，器件功能，以备查阅。书中配备有习题课和讨论课的内容，以启发学生思考，引导创新，辅助教学，便于安排新颖实验制作设计。

鉴于当前集成电路的飞速发展和广泛应用，本教材删减了分立元件的内容，而以中小规模集成电路为主，并注意介绍常用大规模集成电路和新颖集成电路，如 GAL、CMOS—LED 组合件、语音电路、电子表、光耦合器件等。

本书与欧伟民主编的《模拟电子技术基础》一书配套使用，可作为全国高职、中专、技校、职高工科电工类、电子类、计算机类、机电类专业的教材，也可作为有关人员的参考书。

本书由天津市工业学校副教授刘培宇担任主编，吉林省冶金工业学校高级讲师丁有生任副主编。具体编写人员包括：刘培宇（第 4、7 章），丁有生（第 5 章），河北省冶金工业学校王国贞（第 2、3 章），天津市工业学校张涛（第 6 章），攀钢职业技术教育中心徐奉弟（第 1 章）。全书由株洲高等职业技术学院、贵州省冶金学校、黑龙江省理工学校、新疆钢铁学校、山东省工业学校、安徽省冶金工业学校、河南省三门峡黄金工业学校、宝钢工业技术学校等校教师集体审阅。在审稿期间，与会教师们指出了本书许多不足之处，并提出了宝贵意见，在此致以衷心感谢。

由于编者水平所限，书中难免存在缺点和错漏之处，恳请各兄弟学校的师生和读者给予批评指正。

编　者

1999 年 7 月

目　录

1 基础知识

1.1 概述

1.1.1 数字信号和模拟信号

信号有多种。如轮船上使用的灯光信号，是一种光信号；学校中的上课铃声，是一种声音信号；电报是一种电信号，电台发射的电波也是一种电信号。

电信号有两大类，一类叫做模拟信号，其特点是电压值连续变化，如电视机内传输的电视信号；另一类叫做数字信号，其特点是电压值不连续变化，是一种具有一定宽度、一定幅度、瞬时出现的电压，如图 1-1 所示，称为电脉冲，如电报信号。

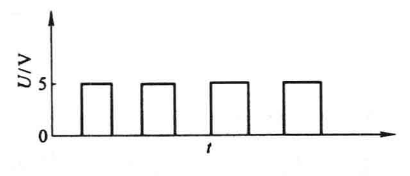


图 1-1　电脉冲之一种

模拟信号的电压值模拟了物质的变化过程，如温度的变化，空间位置的变化等。因为这些变化过程是连续的，所以模拟信号的电压值也是连续变化不可能间断的。数字信号则用于反映自然界中的二元状态，如有与无，开与关，动与停等，因此又可叫做开关信号。在电路中用有电脉冲代表开状态，无电脉冲代表关状态。在分析电路时，用 1 代表有电脉冲（或高电平），称为逻辑 1；用 0 代表无电脉冲（或低电平），称为逻辑 0。可见，数字信号的电压值应该是间断的。

处理模拟信号的电路，称为模拟电子电路，是“模拟电子技术”研究的对象。对数字信号进行加工、变换、输送、存储的电路称为数字电子电路，是“数字电子技术”研究的对象。本教材仅介绍数字电路中的基本电路，如基本门电路、编码译码电路、触发器、计数器、脉冲的产生与变换等。数字电路也称为开关电路或逻辑电路。

1.1.2 数字电路及其应用

数字电路可以用来处理开关过程，对机械的启停实行有序控制。数字信号也可以反映量的变化，但不是用电压的幅度来表征量的变化，而是用脉冲的数目来表征。不过，若纯以脉冲个数来表达相应的数，对很大的数则不便处理。科学家将 0 和 1 的不同组合（实际就是脉冲的组合）来代表 0～9 的 10 个数字，这便巧妙地解决了数字信号对量的表示。由此得到启发，0 和 1 的不同组合不但可以表示数字，还可以表示文字，这种组合称为码。这就提供了一个利用数字电路处理抽象信息的可能，即把信息表达为文字和数字，变为码，变为数字信号，送入数字电路进行计算或处理。这种数字电路构成的设备就是数字电子计算机，它是数字电子技术（简称数字技术）的集大成。由于数字电子计算机的出现，集成电路技术的发展，计算机可以做得很小，如火柴盒大小。这就使计算机有了植入各种设备的可能，于是出现了纯数字化电视、数字电话、数字照相机、数字收音机、VCD、数字仪表、数控机床、PLC、机器人、CT（医用）、传真机、复印机、绘图仪、计费器、电子秤、自动售货机、游戏机等等。几乎遍及社会生活的各个领域。近年来，随着国际互联网的出现，数字化技术还可以实现电子邮件，网上购物，越洋聊天，网上存款，网上取物（照片、图纸、书刊）等等。数字电子计算机的出现不但使原有的某些设备有了重大的变革，而且还出现了许多新事物。正如科学家尼葛洛庞蒂在《数字

化生存》一书中所写的那样，“数字不仅用于计算，还决定我们的生存”。至少可以说，数字技术正在改变人们的生活方式。

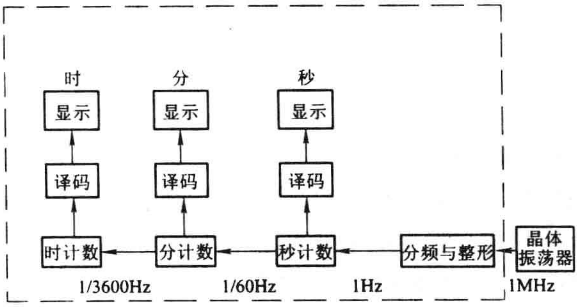


图 1-2　电子表方框图

数字电路究竟是怎样工作的呢？现以电子表为例来说明。图 1-2 为电子表的方框图，它可以显示时、分、秒的时间。其中晶体振荡器用来产生 1MHz 的高频方波脉冲，经过分频与整形电路把它变成 1Hz 的方波脉冲，实际就是一个秒定时器。秒脉冲送秒计数器累计秒值，计数至 60s 便产生一脉冲送分计数器，故秒计数器实为一分定时器，每 60s 便输出一脉冲(频率为 1/60Hz)。分计数器累计分值，即统计分脉冲的个数。分计数器每计满 60 个脉冲便向时计数器送出一个脉冲，它实际是一个小时定时器。秒计数器和分计数器每输出一个脉冲时都自动将计数值清 0。时计数器统计小时脉冲的个数，即统计时值，它是计满 24 个脉冲时便自动将计数值清 0。时、分、秒计数值分别送译码器译为时、分、秒的字形码，然后由显示器显示相应的字形。只要时、分、秒计数器的值发生变化，显示的字形也随之改变。做电子表时，虚线框内的 7 个电路都集成在一个集成电路中，另加晶体振荡器及时、分调整小按钮即可。

由上述中可以看到，数字电路包括的内容十分广泛，它包括由脉冲的产生、整形、分频、控制，直到计数、译码、显示等典型的数字单元电路。

上述的各种单元数字电路，可以由双极型三极管构成，也可以由 MOS 管(即场效应晶体管)构成。无论哪种晶体管，它们在电路中只有两种工作状态，即导通与截止。导通和截止之间的转换过程是很快的。这是数字电路的一大特点。数字电路只研究输出与输入之间的逻辑关系，这也是和模拟电路截然不同的另一特点。由于这一特点，数字电路又称为逻辑电路。数字电路的分析方法也与模拟电路截然不同，它要以逻辑代数为工具。每一种数字电路的功能称为逻辑功能，表达逻辑功能的方式主要有真值表、逻辑式、波形图。

1.2　数制

如果说用码来表示数字，使得数字电路处理量化的信息成为可能的话，将二进制引入数字电路则大大简化了运算规律，使这种可能成为现实，本节的主要内容便是阐述二进制及其转换为其他计数制的方法。

1.2.1　数的表示方法

1.2.1.1　十进制

十进制是我们十分熟悉的计数体制，这种以 10 为基数的计数制可以总结为两条：

(1)十进制有 10 个数字符号，0、1、2、3……9。位权为 $10^{位序-1}$。

(2)在计数及运算时“逢十进一”,“借一当十”,基数为 10。

以上两条可以推广到其他计数制,即:

(1)N 进制有 N 个数字符号。

(2)逢 N 进一,借一当 N,基数为 N。

1.2.1.2　二进制

根据上面的叙述,对以 2 为基数的二进制可以得出:

(1)二进制有两个数字符号,规定为 0(零)和 1(壹)。

(2)逢二进一,借一当二,基数为 2。

既然二进制只有两个数符,那么 2、3、……、9 如何表示?9 以上的数又如何写?请研究下面二进制的加算式:

```
      1……加数 A
  +). 1……被加数 B
  ─────
    1 0……二
    ↑ ↑
    │ └本位和 S
    └──进位 C
     (a)

    1 0
 +)   1
 ──────
    1 1……三
     (b)

      1 1
 +). .  1
 ────────
    1 0 0……四
     (c)

    1 0 0
 +)     1
 ────────
    1 0 1……五
     (d)

    1 0 1
 +)   . 1
 ────────
    1 1 0……六
     (e)

    1 1 0
 +)     1
 ────────
    1 1 1……七
     (f)

      1 1 1
 +). . .  1
 ──────────
    1 0 0 0……八
     (g)

    1 0 0 0
 +)       1
 ──────────
    1 0 0 1……九
     (h)
```

其中算式(a)的 1+1 得二,按逢二进一的法则向第二位进一位,本位记 0,不能写成 2,因为二进制无此数符。进入高位的这一位记为 1,但这个 1 不代表拾,它不是十进制。这一位代表二,逢二进一嘛。也就是说,二进制记数时第二位的 1 之值为二,第一位的 1 才是 1。再看算式(c),第二位的 1 和进位位相加时,仍按逢二进一的法则向第三位进一,本位记 0,得结果 100。这个数是多少呢?当然不应该是 100。已知第二位的 1 之值是二,记入第二位的进位位也是二,它们之和即向第三位的进位位应该是四,故二进制中 100 之值为四。以后书写这种进制数时以 $(100)_2$ 表示,下角之 2 表示()内之数为二进制记数法。再看算式(g)中第三的运算,被加数第三位的 1 和进位位的 1 加,仍按逢二进一规则进行,向第四位进一,本位记 0,得结果 $(1000)_2$,这个数又是多少?如前分析,第三位的 1 值为四,四加四得八,故第四位之 1 的值为八。按上面推出的各项结论,考察二进制数

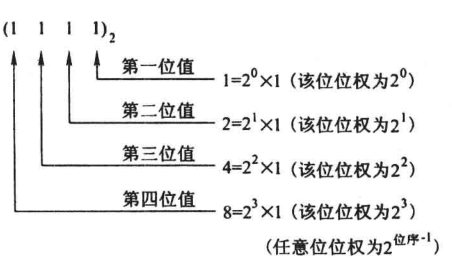


图 1-3　二进制位权分析图

$(1111)_2$,如图 1-3 所示。

故$(1111)_2$之值应为各位之值的和,即 15。由此发现二进制数每一位中的 1 之值,除了第一位是代表 1 之外,以后各位中的 1 之值是前一位中的 1 之值的 2 倍。由此,不难把 9 以上的数写成二进制数,也不难把一个二进制数写成十进制数。

上面介绍的二进制加法表明,二进制数是一连串的 0 和 1,这可以很容易地用脉冲串来表示它;还说明二进制加法比十进制加法运算步骤要简单。只要注意到借一当二便可看懂下面的二进制减法。式(i)中被减数第一位是 0,比减数中的 1 小,不够减需借,借一当二,减 1 余 1,故本位差记 1。第二位原为 0,被借去 1,比减数中的 0 小,不够减需借位,借一当二,还掉被减之 1 余 1,故第二位的差也记 1。第三位的差同理记 1。第四位被减数原为 1,被借走余 0,故本位差记为 0。可见减法也很简单。

$$
\begin{array}{rl}
\underset{\cdot}{1}\ \underset{\cdot}{0}\ \underset{\cdot}{0}\ 0 & \cdots\cdots\text{八} \\
-)\qquad\quad 1 & \cdots\cdots\text{一} \\
\hline
0\ 1\ 1\ 1 & \cdots\cdots\text{七}
\end{array}
$$

(i)减法算式

至于乘除法,比十进制的乘除法更简单。请看算式(j)中的乘法,运算过程不难理解。可以总结出二进制的乘法运算步骤实际只有两步:第一步是移位,第二步是加法。此处的移位共移了四次,这是由乘数的位数决定的。按一般运算的习惯,由低位向高位求积,故移位是向左进行。若乘数为 1,则用被乘数原数实行左移;若乘数为 0,则用 0 实行左移。最后将各部分积求和,即得最后积。除法的运算规则也很简单,由(k)式可以总结为如下三步:(1)由高位至低位按位截取被除数与除数比较;(2)若截取部分比除数小则商 0,并继续右移一位,若截取部分大于或等于除数则商 1;(3)商 1 时要实行截取部分减除数之操作。简言之,第一步右移,第二步比较,第三步减法。

$$
\begin{array}{rl}
1\ 1\ 0\ 1 & \cdots\cdots\text{十三} \\
\times\qquad 1\ 0\ 0\ 1 & \cdots\cdots\text{九} \\
\hline
\left.\begin{array}{r}
1\ 1\ 0\ 1 \cdots\cdots\text{第一位不移位} \\
0\ 0\ 0\ 0\ \ \cdots\cdots\cdots\text{第二位移一位} \\
0\ 0\ 0\ 0\ \ \ \ \cdots\cdots\cdots\cdots\text{第三位移二位} \\
1\ 1\ 0\ 1\ \ \ \ \ \ \cdots\cdots\cdots\cdots\cdots\text{第四位移三位}
\end{array}\right\} & \text{向左移位} \\
\hline
1\ 1\ 1\ 0\ 1\ 0\ 1 & \cdots\cdots\text{一百一十七}
\end{array}
$$

(j)

总之,二进制的引入,使数在数字电路中的表示更简单,也使数的四则运算可以用一些简单的逻辑功能(加、减、左移、右移、比较)来完成,数字电路对量的运算成为可能。本书的全加器、移位寄存器、比较器就可完成这类逻辑操作。

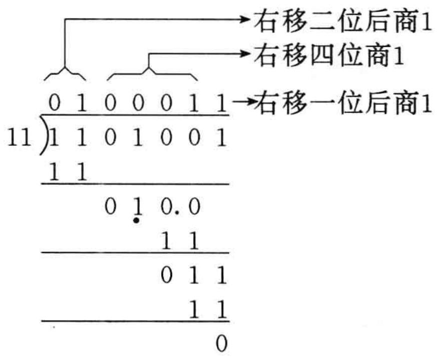
1.2.1.3 八进制

八进制可以写出:

(1)8 个数符,0、1、2、……、7。

(2)逢八进一,借一当八,基数为 8。

按逢八进一的规则可知 7+1=10,此处 10 之值也不为 10,而应为 8。八进制数中各位



(k)

的位权为 $8^{位序-1}$。

【例 1-1】 求八进制数$(407)_8$以十进制表示时的值。

解 $(407)_8=4\times8^2+0\times8^1+7\times8^0=(263)_{10}$

1.2.1.4 十六进制

十六进制可以得到：

(1)有 16 个数符，即 0、1、2、……、9、A、B、C、D、E、F。

(2)逢十六进一，借一当十六，以 16 为基数。

$A\sim F$ 为新的数字符号，其中 $A=10$、$B=11$、$C=12$、$D=13$、$E=14$、$F=15$，$F+1=16$。各位位权为 $16^{位序-1}$。

【例 1-2】 求十六进制数$(2ABC)_{16}$之值。

解 $(2ABC)_{16}=2\times16^3+10\times16^2+11\times16^1+12\times16^0$

$=(10940)_{10}$

几种数制对照见表 1-1。

表 1-1 几种数制对照表

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	0	0	0	13	1101	15	*D*
1	1	1	1	14	1110	16	*E*
2	10	2	2	15	1111	17	*F*
3	11	3	3	16	10000	20	10
4	100	4	4	17	10001	21	11
5	101	5	5	18	10010	22	12
6	110	6	6	19	10011	23	13
7	111	7	7	20	10100	24	14
8	1000	10	8	32	100000	40	20
9	1001	11	9	64	1000000	100	40
10	1010	12	*A*	127	1111111	177	*7F*
11	1011	13	*B*	255	11111111	377	*FF*
12	1100	14	*C*				

1.2.2 不同进制数的转换

1.2.2.1 二进制、十六进制转换为十进制

【例 1-3】 将二进制数$(10011)_2$转换为十进制数。

解 $(10011)_2 = 1\times2^4+0\times2^3+0\times2^2+1\times2^1+1\times2^0$

$= (19)_{10}$

十六进制数的转换见例 1-2。

1.2.2.2 十进制数转换为二进制、十六进制

十进制正整数转换为二进制数可以采用除 2 取余法，其步骤：

第一步，把给定的十进制数连续以 2 除之，直到商 0 为止。每除一次，取其余数(0 或 1)，记于右方。除第一次外，其余各次均为除前次之商。

第二步，按由后至前的顺序排列余数，即得所求的二进制数。

【例 1-4】 将$(158)_{10}$转换成二进制数。

解

$$
\begin{array}{r|l l l}
2 & 158 & \cdots\cdots\text{余}0 & \uparrow\text{低位} \\
2 & 79 & \cdots\cdots\text{余}1 & \\
2 & 39 & \cdots\cdots\text{余}1 & \\
2 & 19 & \cdots\cdots\text{余}1 & \\
2 & 9 & \cdots\cdots\text{余}1 & \\
2 & 4 & \cdots\cdots\text{余}0 & \\
2 & 2 & \cdots\cdots\text{余}0 & \\
2 & 1 & \cdots\cdots\text{余}1 & \text{高位} \\
 & 0 & &
\end{array}
$$

答：$(158)_{10}=(10011110)_2$

十进制数转换为十六进制数可类似地采取除 16 取余法。

【例 1-5】 将$(158)_{10}$转换为十六进制数。

解

$$
\begin{array}{r|l l l l}
16 & 158 & \cdots\cdots\text{余}14 & \text{即}E & \uparrow\text{低位} \\
16 & 9 & \cdots\cdots\text{余}9 & 9 & \text{高位} \\
 & 0 & & &
\end{array}
$$

答：$(158)_{10}=(9E)_{16}$

十进制数转换为八进制数，读者可自己思考。

1.2.2.3 二进制与十六进制的相互转换

将二进制数(正整数)转换为十六进制数的方法是：从最低位开始，每四位分成一组，每组对应转换为一位十六进制数。

【例 1-6】 将$(10011110)_2$转换为十六进制数

解

$$
\begin{array}{cc}
\underline{1001} & \underline{1110} \\
\downarrow & \downarrow \\
9 & E
\end{array}
$$

答：$(10011110)_2=(9E)_{16}$

【例 1-7】 将$(1010011110)_2$转换为十六进制数。

解

$$\begin{array}{ccc} \underline{0010} & \underline{1001} & \underline{1110} \\ \downarrow & \downarrow & \downarrow \\ 2 & 9 & E \end{array}$$

答：$(1010011110)_2=(29E)_{16}$

十六进制正整数转换为二进制数，就是上例之逆过程。

【例 1-8】 将$(29E)_{16}$转换为二进制数。

解

$$\begin{array}{ccc} 2 & 9 & E \\ \downarrow & \downarrow & \downarrow \\ 0010 & 1001 & 1110 \end{array}$$

答：$(29E)_{16}=(1010011110)_2$

1.2.3 二-十进制码(BCD 码)

前面介绍了几种数制的表示方法，说明了数字技术中量的表示方法。数字技术中往往也采用一定位数的二进制数来表示各种文字、符号，通常称之为代码。这与电报业中用数字代表文字，使之一一对应的做法相类似。将文字变为代码称为编码，代码变为文字则称为译码（电报业中称为译电）。所谓 BCD 码或二-十进制码，指的是用 4 位二进制数来表示十进制数中 0～9 的 10 个数符。由于 4 位二进制数有 16 种不同的组合，而表示 0～9 的 10 个数符只需要 10 种组合，其余 6 种组合便是无效的。因此，按选取方式的不同，可以得到不同的 BCD 码。表 1-2 列出了常见的几种 BCD 码。

表 1-2 常见的 BCD 码

<table>
<tr><th>十进制数</th><th>8421 码</th><th>十进制数</th><th>2421 码(A)</th><th>十进制数</th><th>2421 码(B)</th><th>十进制数</th><th>5421 码</th><th>十进制数</th><th>余 3 码</th><th>十进制数</th><th>格雷码</th></tr>
<tr><td>0</td><td>0000</td><td>0</td><td>0000</td><td>0</td><td>0</td><td>0</td><td>0000</td><td rowspan="3">不出现</td><td>0000</td><td>0</td><td>0000</td></tr>
<tr><td>1</td><td>0001</td><td>1</td><td>0001</td><td>1</td><td>1</td><td>1</td><td>0001</td><td>0001</td><td>1</td><td>0001</td></tr>
<tr><td>2</td><td>0010</td><td>2</td><td>0010</td><td>2</td><td>10</td><td>2</td><td>0010</td><td>0010</td><td>2</td><td>0011</td></tr>
<tr><td>3</td><td>0011</td><td>3</td><td>0011</td><td>3</td><td>11</td><td>3</td><td>0011</td><td>0</td><td>0011</td><td>3</td><td>0010</td></tr>
<tr><td>4</td><td>0100</td><td>4</td><td>0100</td><td>4</td><td>100</td><td>4</td><td>0100</td><td>1</td><td>0100</td><td>4</td><td>0110</td></tr>
<tr><td>5</td><td>0101</td><td>5</td><td>0101</td><td rowspan="6">不出现</td><td>101</td><td rowspan="3">不出现</td><td>0101</td><td>2</td><td>0101</td><td>5</td><td>0111</td></tr>
<tr><td>6</td><td>0110</td><td>6</td><td>0110</td><td>110</td><td>0110</td><td>3</td><td>0110</td><td>6</td><td>0101</td></tr>
<tr><td>7</td><td>0111</td><td>7</td><td>0111</td><td>111</td><td>0111</td><td>4</td><td>0111</td><td>7</td><td>0100</td></tr>
<tr><td>8</td><td>1000</td><td rowspan="6">不出现</td><td>1000</td><td>1000</td><td>5</td><td>1000</td><td>5</td><td>1000</td><td>8</td><td>1100</td></tr>
<tr><td>9</td><td>1001</td><td>1001</td><td>1001</td><td>6</td><td>1001</td><td>6</td><td>1001</td><td>9</td><td>1000</td></tr>
<tr><td rowspan="6">不出现状态</td><td>1010</td><td>1010</td><td>1010</td><td>7</td><td>1010</td><td>7</td><td>1010</td><td></td><td></td></tr>
<tr><td>1011</td><td>1011</td><td>5</td><td>1011</td><td>8</td><td>1011</td><td>8</td><td>1011</td><td></td><td></td></tr>
<tr><td>1100</td><td>1100</td><td>6</td><td>1100</td><td>9</td><td>1100</td><td>9</td><td>1100</td><td></td><td></td></tr>
<tr><td>1101</td><td>1101</td><td>7</td><td>1101</td><td rowspan="3">不出现</td><td>1101</td><td rowspan="3">不出现</td><td>1101</td><td></td><td></td></tr>
<tr><td>1110</td><td>8</td><td>1110</td><td>8</td><td>1110</td><td>1110</td><td>1110</td><td></td><td></td></tr>
<tr><td>1111</td><td>9</td><td>1111</td><td>9</td><td>1111</td><td>1111</td><td>1111</td><td></td><td></td></tr>
<tr><td>权</td><td>8421</td><td></td><td>2421</td><td></td><td>2421</td><td></td><td>5421</td><td></td><td>无权</td><td></td><td>无权</td></tr>
</table>

在二-十进制编码中，一般分为有权码和无权码两大类。例如 8421BCD 码是一种最基本的、应用十分普遍的 BCD 码。它是一种有权码，8421 就是指这种编码中各位的权分别为 8、4、2、1。属于有权码的还有 2421BCD 码，5421BCD 码等，而余 3 码、格雷码则是无权码。对于有权码来说，由于各位场有固定的权，所以这些码所表示的值就很容易识别。

二-十进制的表示方法也很简单，将十进制数的各位数字分别用 4 位二进制数按上述编码表示出来即可。

【例 1-9】 将$(149)_{10}$用 8421BCD 码表示。

解 $(149)_{10}=(0001\ 0100\ 1001)_{BCD}$

$$\begin{array}{ccc} \downarrow & \downarrow & \downarrow \\ \text{一百} & \text{四十} & \text{九} \end{array}$$

在例 1-9 中，$(000101001001)_{BCD}$的各组 4 位数是二进制，而组与组之间却是十进制的关系。还要指出，每组中的高位上的 0 都不能省略。又如$(109)_{10}=(000100001001)_{BCD}$，中间的(即十位上)0 应为 4 个 0。总之，每组都是 4 个 0 或 1 的组合。

1.3 逻辑代数

1.3.1 基本逻辑关系和基本逻辑运算

1.3.1.1 与逻辑

在图 1-4 的电路中，灯 Y 亮这一事件发生的条件是开关 A 闭合，同时开关 B 也闭合。只要有开关不闭合，即 A 不闭合(条件 A 不具备)，或 B 不闭合(条件 B 不具备)，或 A 和 B 都不闭合(全部条件都不具备)，事件 Y(灯亮)就不能发生。事件和条件之间的关系称为逻辑关系。研究逻辑关系的数学称为逻辑代数或布尔代数。在布尔代数中用逻辑变量来反映条件，如此例中的条件 A 和条件 B 在布尔代数中就称变量 A 和变量 B。逻辑变量的取值只有两种。即逻辑 1 和逻辑 0。逻辑 1 或称 1，代表条件成立(此例中为开关闭合)，逻辑 0 或称 0，代表条件不成立(此例中为开关不闭合)。布尔代数用逻辑函数来反映事件，如此例中的事件 Y，在逻辑代数中用逻辑函数 Y 来反映，逻辑函数的取值也只有两种，即 1 和 0。此处的 1 和 0 也同样没有数量的内涵，只有一种逻辑结果的符号。1 表示事件 Y 能发生(灯能亮)，0 表示事件 Y 不会发生(灯不亮)。

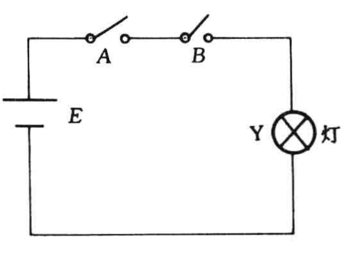


图 1-4 与逻辑关系

事件 Y 和条件 A、B 之间的逻辑关系是：只有全部条件具备事件才会发生，逻辑代数中把这种逻辑关系称为与逻辑。还可以找到许多具有与逻辑关系的实例。如甲、乙两人用一根扁担抬水，则必须两人齐用力，缺一不可。事件 Y(抬水)发生，必须甲乙两个条件都成立，即全部条件都成立。所以，此处事件 Y 和条件(甲与乙)之间的逻辑关系是与逻辑。再如，某单位要支取百万元储备金(事件 Y)，必须甲、乙、丙 3 位领导都签字才行。此处，事件 Y 与条件甲、乙、丙之间的关系也是与逻辑关系。可见，与逻辑关系是自然界中大量同类关系的一种数学抽象。在逻辑代数中，用下面的逻辑式来表达逻辑函数 Y(事件 Y)和逻辑变量 A、B、C(条件甲、乙、丙)之间与逻辑关系(或称与运算关系)：

$$Y=A\cdot B\cdot C \tag{1-1}$$

简写为 $$Y=ABC \tag{1-1'}$$

式中"·"称为与运算符。根据上述与逻辑的定义，对变量和逻辑函数的取值 0，1 的约定，下面的运算结果是必然的：

A,B,C 任一为 0　　$Y=0\cdot1\cdot1=1\cdot0\cdot1=1\cdot1\cdot0=0$

A,B,C 任二为 0　　$Y=0\cdot0\cdot1=0\cdot1\cdot0=1\cdot0\cdot0=0$

A,B,C 全为 0　　$Y=0\cdot0\cdot0=0$

A,B,C 全为 1　　$Y=1\cdot1\cdot1=1$

上面的运算结果表明一点，只有 ABC 3 个条件全部成立，事件 Y 才能发生，这正是与运算的内涵。

1.3.1.2　或逻辑

在图 1-5 的电路中，Y 灯亮，即事件 Y 发生的条件有两个，开关 A 及开关 B。事件 Y 与条件 A 和 B 的关系是：开关 A 闭合，或开关 B 闭合。并不一定要求两个条件必须同时成立，当然两个条件都成立事件 Y 也能发生。若有一个或一个以上条件成立，事件就发生，则事件 Y 和条件之间的这种逻辑关系称为或逻辑。或逻辑式如下：

$$Y=A+B \tag{1-2}$$

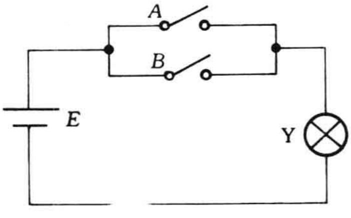


图 1-5　或逻辑关系

此处"＋"号为或逻辑运算符。显然有：

A,B 中任一为 1　　$Y=0+1=1+0=1$

A,B 都为 1　　$Y=1+1=1$

A,B 都为 0　　$Y=0+0=0$

上面的运算结果说明：(1)只要有一个或一个以上条件成立，事件 Y 就能发生，即 Y 取 1；(2)$1+1=1$，而不等于 2，也不等于二进制中的 10，因为此处 1 不具备量的内涵，只表示条件成立，表示事件 Y 能发生，而"＋"号也不是加的意思，而是"或"的意思。

或逻辑的实例还可举出许多。如一屋同住 4 人，每人各持一把锁，把 4 把锁 A,B,C,D 串联起来形成锁链，再把门锁上，A 挂一门扣，D 挂另一门扣，则 4 人中任一人回屋都能开锁(事件 Y)进屋。开锁，这一事件的发生，只要 A,B,C,D4 把锁(4 个条件)任一把或一把以上的锁打开即可。这就是一个或逻辑实例，表达为逻辑式便是：

$$Y=A+B+C+D$$

可见，或逻辑也是自然界中同类关系的数学抽象。

1.3.1.3　非逻辑

在图 1-6 中，灯 Y 发亮(事件 Y)的条件是灯 A 不亮，若灯 A 亮则 Y 不亮，Y 与条件 A 状态始终相反，这里事件 Y 与条件间的逻辑关系，在逻辑代数中称为非逻辑。其逻辑式如下所示：

$$Y=\overline{A} \tag{1-3}$$

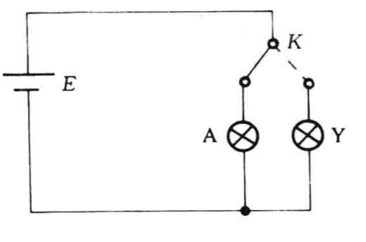


图 1-6　非逻辑关系

A 上方的"—"号称为非逻辑运算符。非运算如下：

A 为 0 时　　$Y=\overline{A}=\overline{0}=1$

A 为 1 时　　$Y=\overline{A}=\overline{1}=0$

非逻辑的实例也是很多的，如两队争足球冠军的决赛，则结果必然只有一队是冠军，另一队不是冠军。设为 Y 队和 A 队，则有：

$$Y=\overline{A}$$

A 是冠军，$A=1$　$Y=\overline{A}=\overline{1}=0$　Y 不是冠军　A 不是冠军，$A=0$　$Y=\overline{A}=\overline{0}=1$　Y 是冠军　非逻辑也是自然界中同类逻辑关系的数学抽象。与、或、非是自然界中 3 种最基本的逻辑关系。

1.3.2 逻辑函数的表示方式

1.3.2.1 逻辑表达式

逻辑表达式简称逻辑式，在逻辑式中，变量 A,B,C……的取值确定后，逻辑函数 Y 的取值应该是惟一的。上面已介绍了 3 种基本的逻辑函数式，如下所示：

与函数　　$Y=A\cdot B$

或函数　　$Y=A+B$

非函数　　$Y=\overline{A}$

逻辑式便于计算和表达逻辑关系，但在自然界中，某些事件是别的事件的条件，所以大量存在的是由基本逻辑关系构成的复杂的逻辑关系，逻辑式也就显得复杂。

1.3.2.2 逻辑图

用逻辑图表达逻辑关系。它的特点是形象直观，便于分析，如图 1-7 所示。

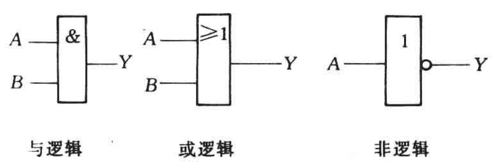


图 1-7　基本关系逻辑符号

1.3.2.3 真值表

将逻辑变量所有可能的取值列出，再写出逻辑函数 Y 的对应取值，以表列出即得。如表 1-3～表 1-5 所示。

表 1-3　与真值表

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

表 1-4　或真值表

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

表 1-5　非真值表

A	Y
0	1
1	0

真值表便于将实际问题表达为逻辑式，后面将逐步介绍。

1.3.2.4 常见逻辑关系的表示

A　与非逻辑

与非逻辑就是先与后非的逻辑关系，即：

$$Y'=A\cdot B$$

$$Y=\overline{Y'}=\overline{A\cdot B} \tag{1-4}$$