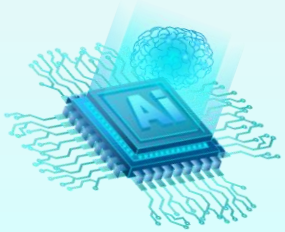
第4章 数字电路基础

数字电子计算机主要由庞大的数字电路构成，它以数字0和1为基础。数字电路主要研究对象是电路的输出与输入之间的逻辑关系，使用的数学工具主要是逻辑代数。本章主要讲述数字电路的基础知识，包括数制及数制之间的转换方法、计算机中信息的基本表示方式、逻辑代数基础、逻辑函数的表示与化简等。



4.1 数制及数制之间的转换方法

信息（Information）一词是一个严谨的科学术语，其定义不统一，含义十分广泛。人们利用这个词来描述客观世界各种事物的运动状态、相互联系与相互作用。可以从信息的表现形式、类别等直观理解信息，信息的表现形式如数字、文字、语言、图片、温度、体积、颜色等，信息的类别如电子信息、财经信息、天气信息、生物信息等。

现在我们使用的计算机，其雏形源于1946年美国宾夕法尼亚大学研制的电子数字积分器和计算器（Electronic Numerical Integrator And Calculator，ENIAC），简称电子计算机，或直接称为计算机。它是利用电子元器件及电子线路制作的可以进行数值计算、逻辑计算，并具有存储记忆功能，能够按照程序运行的电子设备。

计算机中的信息均是用数字“0”、“1”表示，涉及到数制问题。

4.1.1 数制

1．数制的概念

通俗地说，数制（Number system）就是计数的法则，它用一组固定的数码和一套统一的规则来表示数字的大小。例如，人们日常生活中使用的数制是十进制（Decimal system），它使用0、1、2、3、4、5、6、7、8、9这十个数码，并定义以下规则：自然界中所有的数字都用这十个数码表达，满十进一，且规定同一个数码在从左到右不同的位置上所表示的数值大小不同。人类普遍使用十进制，可能与远古时代用十指记数这个习惯有关。

2．基数计数法

数制中的基数（Radix number）表示基本符号的个数。例如，十进制的基数就是10，二进制的基数就是2，十六进制的基数为16。

数制中的位权（Position weight）表示某一位上的1所表示数值的大小（所处位置重要性的度量）,一般简称权（weight）。

有了基数与权概念，任意一个数x可表示成按权展开：

$$x = \sum_{i=n-1}^{-m} \alpha_i R^i = \alpha_{n-1}R^{n-1} \ldots + \alpha_1 R^1 + \alpha_0 R^0 + \alpha_{-1}R^{-1} + \cdots + \alpha_{-m}R^{-m}$$

其中R表示某一进制的基数，n表示整数部分位数，m为小数部分位数。

例如，对任意一个平时使用的十进制数，可按照权展开为：

$$836.207 = \sum_{i=2}^{-3} \alpha_i R^i = 8 * 10^2 + 3 * 10^1 + 6 * 10^0 + 2 * 10^{-1} + 0 * 10^{-2} + 7 * 10^{-3}$$

【课堂练习】自行举一个十进制例子，按照权展开

3．计算机中常用的数制

数制	数码	数码个数	基数	进位规则	借位规则	书写前缀	书写后缀
二进制	0、1	2	2	逢二进一	借一当二	0b	B
十进制	0、1、2、3、4、5、6、7、8、9	10	10	逢十进一	借一当十	（无）	D
十六进制	0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F	16	16	逢十六进一	借一当十六	0x	H
说明	十六进制数码中的 A、B、C、D、E、F 分别对应十进制的 10、11、12、13、14、15						

注意：书写的前后缀

【课堂练习】将二进制数101.1101及十六进制数8BD.A6F按权形式展开

4.1.2 数制之间的转换方法

1．其他进制数与十进制数之间的转换

1）其他进制数转为十进制数：

“按权展开求和”

2）十进制数转为其他进制数

十进制数转为其他进制数的方法，一般采用“乘除法”。

【例 4-2】 将十进制数 89.86 转为二进制制数。

解：89.86=0b1011001.1101111，计算方法如下：

整数部分：

被除数	除数	商	余数	
89	÷2=	44	1	整数部分最低位
44	÷2=	22	0	
22	÷2=	11	0	
11	÷2=	5	1	
5	÷2=	2	1	
2	÷2=	1	0	
1	÷2=	0	1	整数部分最高位

小数部分 ：

被乘数	乘数	乘积	整数	
0.86	*2=	1.72	1	小数部分最高位
0.72	*2=	1.44	1	
0.44	*2=	0.88	0	
0.88	*2=	1.76	1	
0.76	*2=	1.52	1	
0.52	*2=	1.04	1	
0.04	*2=	0.08	0	小数部分最低位
0.08	*2=	0.16	0	
0.16	*2=	0.32	0	……（取决于期望的精度）

【课堂练习】把十进制数56.23转为二进制数和十六进制数。

2．二进制数与十六进制数之间的转换

表4-2 十六进制数与二进制数的对应关系

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

二进制数转换为十六进制数的基本方法：以小数点为界，整数部分向左，每4位二进制数为一组，不足4位的，高位补0，然后用1位十六进制数码表示对应的二进制数即可；小数部分向右，每4位二进制数为一组，不足4位的，低位补0，然后用1位十六进制数码表示对应的二进制数即可。

十六进制数转换为二进制数的基本方法：把每位十六进制数码用4位二进制数表示，书写时根据具体情况去除不影响结果的整数部分的前置0与小数部分的后置0，使之符合平时书写习惯即可。

【课后自行练习】

3．利用工具查看进制转换结果（编程时可用）

4.2 计算机中信息的基本表示方式

在数学中有正数、负数、小数、实数等概念，它们计算机中是如何表示的？

4.2.1 计算机中信息表示的相关基本概念

1．位、字节、机器字长

“位”（bit）是单个二进制数码的简称，是可以拥有两种状态的最小二进制值，分别用“0”和“1”表示。8位二进制数称为一个“字节”（byte），是计算机中信息的基本度量单位。机器字长是指计算机在运算过程中一次能吞吐的二进制数据位数，等于数据总线条数，与CPU内数据寄存器的宽度是一致的。

计算机中使用二进制，可做如下理解：第一，二进制只取两个数码0和1，物理上可以用两个不同的稳定状态的元器件来表示；第二，它的运算规则简单，基数为2，进位规则是“逢二进一”，借位规则是“借一当二”；第三，计算机的理论基础是逻辑和代数，当二进制与只使用“真”和“假”两个值与逻辑代数建立联系后，就为计算机的逻辑设计提供了便利的工具，如集成电路中门电路的设计。

2．机器数与真值

计算机中规定存储时最高位为符号位，0表示正数，1表示负数。一个数学中实际的数，符号书写用“±”号表达（“+”号通常省略），称为真值。在规定了用0表示正数、1表示负数之后，以二进制形式形式存储于计算机内部，称为机器数。

4.2.2 整数在计算机中的补码表示方法

在数学中，把像-3，-2，-1，0，1，2，3，10等这样的数，称为整数（integer），在整数系中，零和正整数被称为自然数，-1、-2、-3、…、-n、…（n为非零自然数）为负整数。那么整数如何存储在计算机的存储器中的呢？结论：整数在计算机使用补码表示

1．补码的定义与求法

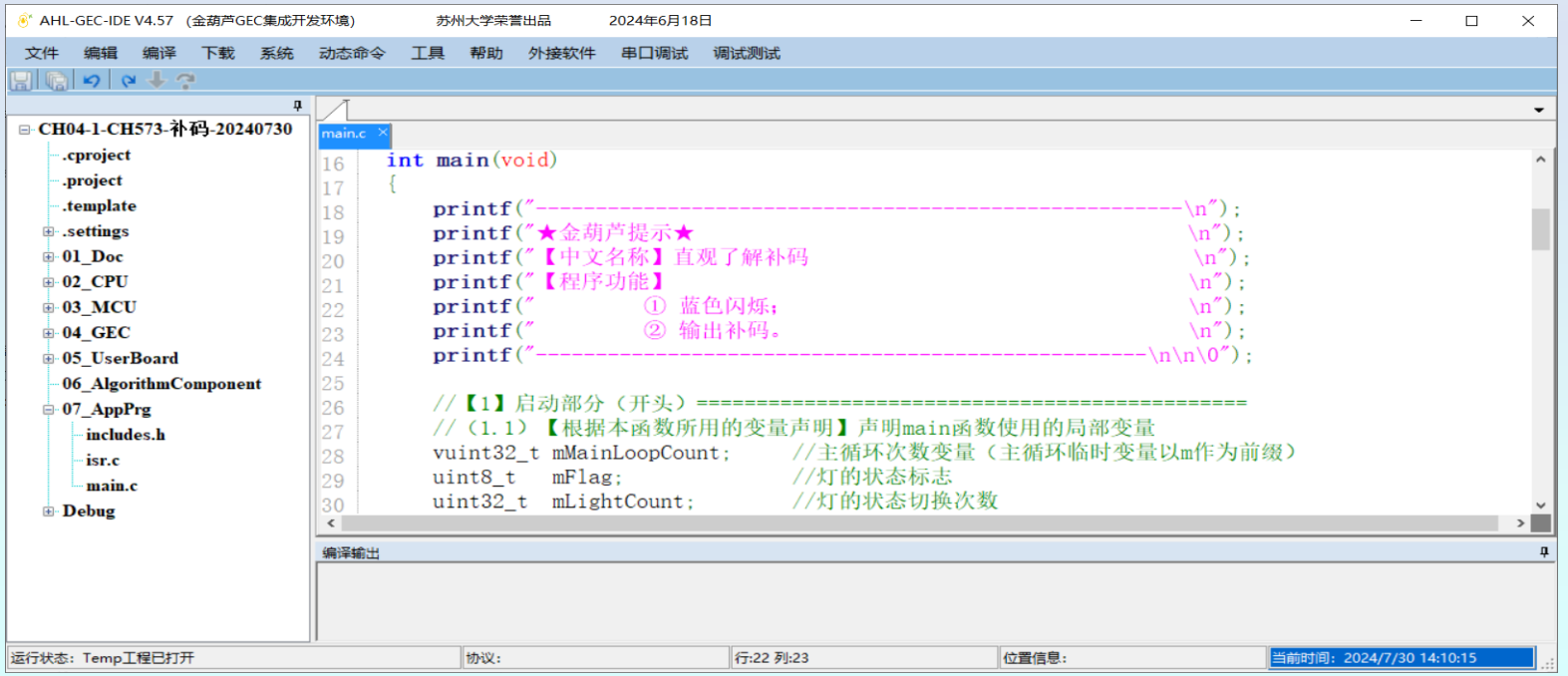
补码主要用于解决负整数在计算机中的存储问题，对应n位字长，模$m=2^n$，整数表达范围是：$-2^{n-1}\sim(2^{n-1}-1)$，设真值记为x_z，对于在此范围内的任意正数，$x_z\geq0$，其补码：$x_b=x_z$，对于在此范围内的任意负数，$x_z\leq0$，其补码：$x_b=2^n-|x_z|$。简单地说，一个正整数的补码就是自身，一个负整数的补码是模减去这个数的绝对值。

课堂练习：对于8位字长的计算机

对于16位字长的计算机

2．利用程序直观了解补码【03-Software\CH04\补码-CH573】

资源下载：https://sumcu.suda.edu.cn/jsjyjjc/list.htm，附录A




AHL-GEC-IDE V4.57 (金葫芦GEC集成开发环境)
苏州大学荣誉出品
2024年6月18日
文件 编辑 编译 下载 系统 动态命令 工具 帮助 外接软件 串口调试 调试测试
金葫芦AHL-GEC-IDE：串口更新 (程序下载与运行)
设备连接
重新连接GEC
COM28： CH573 BIOS RTOS V1.7 20240702
自动搜索
更新操作与进度提示
一键自动更新
暂停传输
退出串口更新
100.0%
导入机器码.Hex文件
请选择机器码Hex文件
选择文件
机器码文件： CH04-1-CH573-补码-20240730.hex
:04C000006F10004A73
:10C0040008000200000000006F0080086F0080089E
:10C014000000000000000000000000000000000001C
:10C024000000000000000000000000000000000000C
:10C034006F008006000000006F0040060000000052
:10C044006F0000066F0000066F0000066F00000618
:10C054006F0000066F0000066F000006000000007D
:10C064006F00C0056F00C0056F00C0056F40AF8A48
更新与运行提示信息
清空
灯闪烁次数mLightCount = 8
==========
数据（用十进制有符号数表达）g_x1=-16
实际存储情况（用十六制表达）g_x1=FFFFFFF0
LIGHT_BLUE:OFF
==========
数据（用十进制有符号数表达）g_x1=-17
实际存储情况（用十六制表达）g_x1=FFFFFFEF
LIGHT_BLUE:ON
灯闪烁次数mLightCount = 9
==========
数据（用十进制有符号数表达）g_x1=-18
运行状态：程序整体更新成功
协议：串口
行:22 列:23
协议信息：端口COM28,波特率115200
当前时间：2024/7/30 14:11:27

3．在计算机中整数使用补码表示的缘由

1）原码与反码表示不合适

为了容易理解，下面以8位计算机为例阐述这个问题。

（1）所谓原码表示，就是用最高位表示整数的±号，0表是正，1表示负，数值位不变。这种表示出现了-0（1000 000）问题，0就是0，哪还有+0、-0如何理解？如何参与运算？

（2）所谓反码表示，就是用最高位表示整数的±号，0表是正，1表示负，数值位逐位取反。这种表示出现了-0（1111 1111）问题。

（3）原码与反码表示解决不了符号位变成了数字之后参与运算问题。在原码表示中，计算：1+（-1）=$(0000\ 0001)_{原}$+ $(1000\ 0001)_{原}$=$(1000\ 0010)_{原}$ = -2，这是不对的。在反码表示中，计算：（-1）+（-2）=$(1111\ 1110)_{反}$+$(1111\ 1101)_{反}$=$(1111\ 1011)_{反}$=$(1000\ 0100)_{原}$ = -4，这也是不对的。

2）补码表示合适

（1）补码表示可以解决以上问题。首先，没有+0、-0问题了，而且可以用原码中-0(1000 0000)，在补码中表示为-128，形成了-128，-127，…，-1，0，1，…，127，共256个8位有符号数的完整表达。其次，在补码表示中，计算：1+（-1）=0000 0001+ 1111 1111=0000 0000=0，这是对的。又用补码表示计算：（-1）+（-2）=$(1111\ 1111)_{补}$+$(1111\ 1110)_{补}$=$(1111\ 1101)_{补}$=$(1000\ 0011)_{原}$= -3，这也是对的。

（2）使用补码表示，可以将真值的减法运算变为机器中加法运算，使得CPU内部不需要设计减法器。例如，1-2=1+(-2)= $(0000\ 0001)_{补}$+$(1111\ 1101)_{补}$=$(1111\ 1111)_{补}$=$(1000\ 0001)_{原}$= -1，正确。

3）补码设计的基本数学原理

以上可以看出，通过设计机器数的补码表示，有效地解决了符号位参与运算问题，且将减法变成了加法，简化了CPU内运算器的设计。那么，补码设计的基本数学原理是什么呢?

从直观理解模的概念说起。生活中具有1～12小时指针的机械闹钟，到12小时后，又从0开始（12就是0），即超过12就溢出了。若说是18点，即6点，18/12的余数是6，数学上称之为模运算，符号“mod”，即18 mod 12 = 6，读做“18模12的结果为6”。

接下来看，若机械闹钟指针指向8点，要把它拨到指向5点，有两种方法：

方法一：回拨，即逆时针拨3小时，即用减法：8-3=5；

方法二：正拨，即顺时针拨9小时，即用加法：8+9=5（不对啊，8+9怎么等于5？可对于这个闹钟，这样的操作是对的）。看看实际数学过程：(8+9) mod 12 =5，即8-3与8+9具有等同效果。减法运算变成了加法运算。同时，注意这个9=12-3，给出了顺时针拨多少小时的一个求法，可以表示成：8-3与8+(12-3)是等效的。

类比一下，在计算机中，若用8位表示机器数，超过256就溢出了（256就是0）。类似上面方法：要计算1-2，通过类比，1-2与1+(256-2)是等效的，从8位机器数来看，这个254就称为-2的“补码”。利用这种方法，减法不见了，减法变成了用加法替代。这种分析，也给出了负数补码的一种简便求法，例如-2的补码：256-2=254。特别看看-128的补码：256-128=128，-128就是8位机器数（补码）能表示的最小数字了。

从数学角度看，5、17、29、…对12来说，余数是相同的，这些余数相同的数被称之为“同余数”。不失一般性表述：两个整数a，b，若它们除以整数m所得的余数相等，则称a，b对于模m同余，记作 $a \equiv b \pmod{m}$，读作 a 与 b 关于模 m 同余。同余数理论被用于机器数的补码设计，解决了减法变加法及符号位参与运算问题。

4.2.3 实数在计算机中的浮点数表示方法

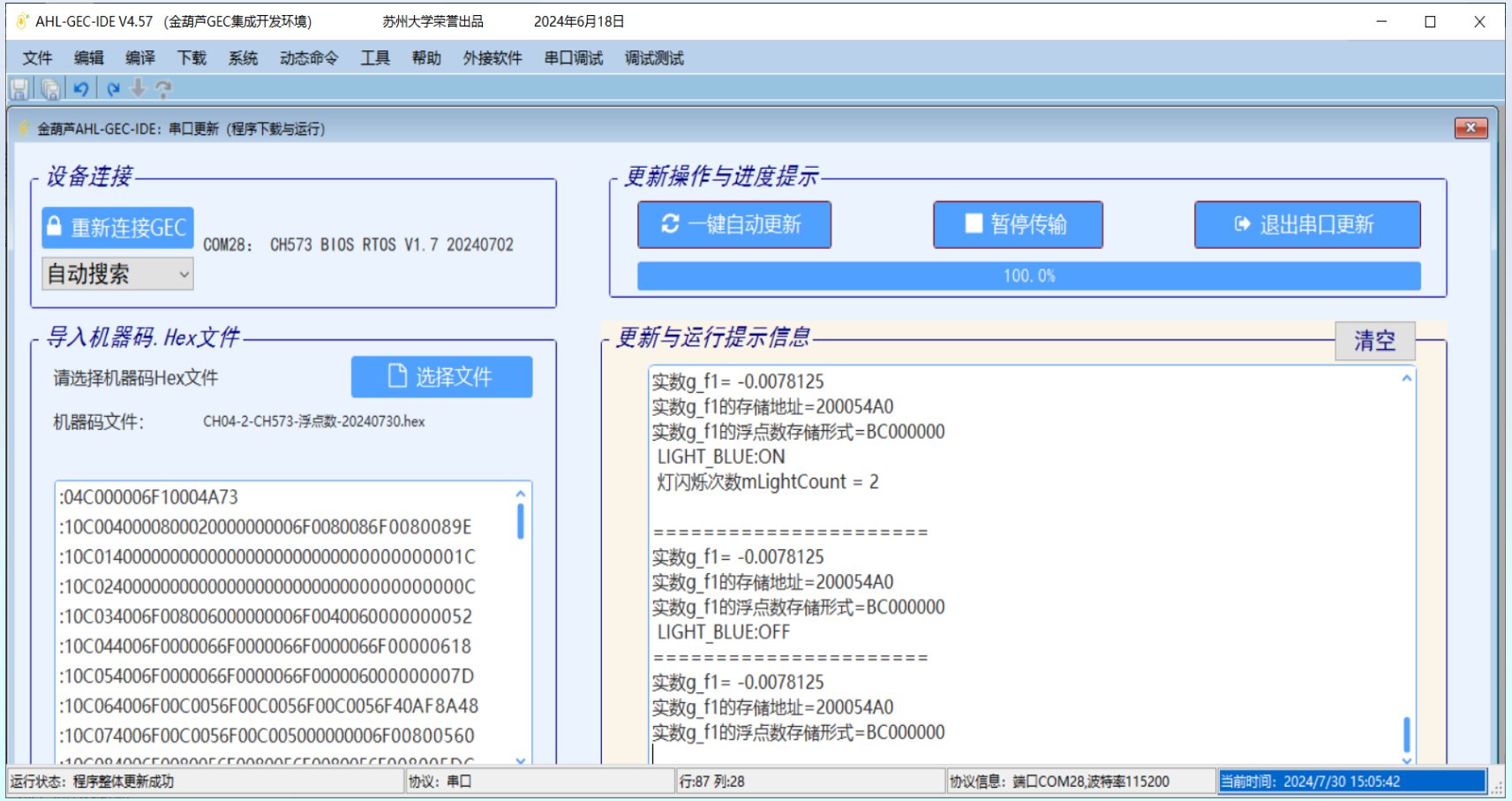
1．浮点数表示方法

数学上，实数是带有小数点数，高级计算机语言中大多采用浮点数存储数学中带小数点的数。IEEE的IEC 60559标准规定：从逻辑上用三元组{S，E，M}来表示一个数V的，其中符号位S决定数是正数（S＝0）还是负数（S＝1），占1位；M是二进制小数，是通过把二进制数中的小数点向左移n位，直到小数点的左边只有一位且为1而得到的，占23位，被称为尾数位；指数位E=127+n，占8位，它决定了小数点的实际位置。

表4-3 单精度浮点数（32位）存储格式

	字段1（1位）	字段2（8位）	字段3（23位）	存储值（十六进制）
数据位	D31	D30 …… D23	D22 …… D0	
用途	符号位	指数位	尾数位	
记号	S	E	M	
例1：32.125	0	10000100	00000001000000000000000	0x42008000
例2：-64.75	1	10000101	00000011000000000000000	0xC2818000
例3：0.015625	0	01111001	00000000000000000000000	0x3C800000
例4：-0.0078125	1	01111000	00000000000000000000000	0xBC000000

2. 直观理解浮点数表示方法【03-Software\CH04\浮点数-CH573】



4.3 逻辑代数基础

导引：

什么是逻辑（logic）？逻辑就是思维的规律，从哲学角度来看，就是因果规律，即从某些已知条件出发推出合理的结论。

逻辑代数？逻辑代数是一种用于描述客观事物逻辑关系的数学方法，由英国科学家乔治·布尔（George·Boole）于19世纪中叶提出，因而又称布尔代数；逻辑变量的取值仅为“0”和“1”，基本逻辑运算有“与”、“或”、“非”等；它有一套完整的运算规则，包括公理、定理和定律，是计算机硬件设计的有力工具，被广泛地应用于开关电路和数字逻辑电路的变换、分析、化简和设计上，因此也被称为开关代数。

逻辑关系是事物或事理的内部联系，其类型包括因果关系、层递关系、主次关系、总分关系、并列关系等。把反映“条件”和“结果”之间的关系称为因果逻辑关系。如果以电路的输入信号反映“条件”，以输出信号反映“结果”，此时电路输入、输出之间也就存在确定因果逻辑关系。数字电路就是实现特定逻辑关系的电路，又称为逻辑电路。逻辑电路的基本单元是逻辑门，它们反映了基本的逻辑关系。

4.3.1 常用逻辑关系的电路及符号

1．非逻辑及非门

非逻辑是指：决定某事件的唯一条件不满足时，该事件就发生；而条件满足时，该事件反而不发生的一种因果关系。$Y=\overline{A}$

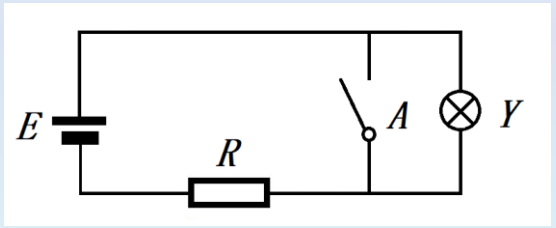
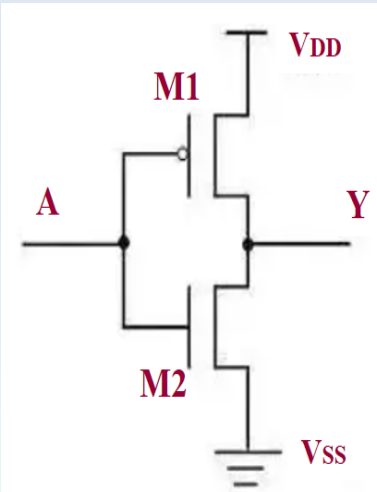
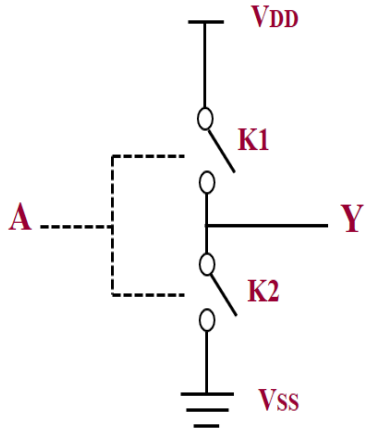


表4-5 非门真值表

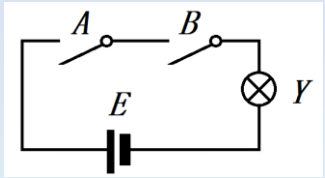
A	Y
0	1
1	0





2．与逻辑及与门

与逻辑指的是：只有当决定某一事件的全部条件都具备之后，该事件才发生，否则就不发生的一种因果关系。Y=A·B



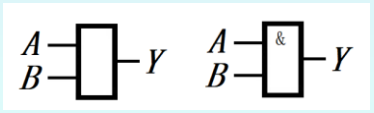
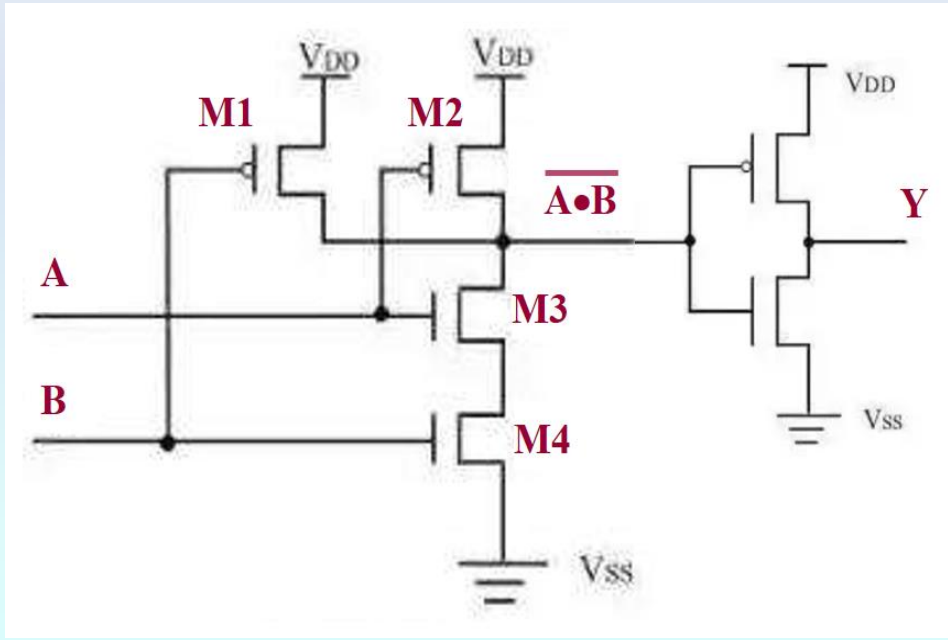


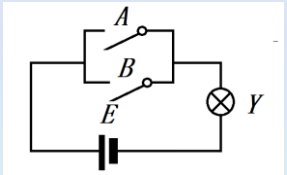
表4-5 与门真值表

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



3．或逻辑及或门

或逻辑指的是：在决定某事件的诸条件中，只要有一个或一个以上的条件具备，该事件就会发生；当所有条件都不具备时，该事件才不发生的一种因果关系。Y=A+B



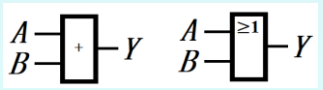
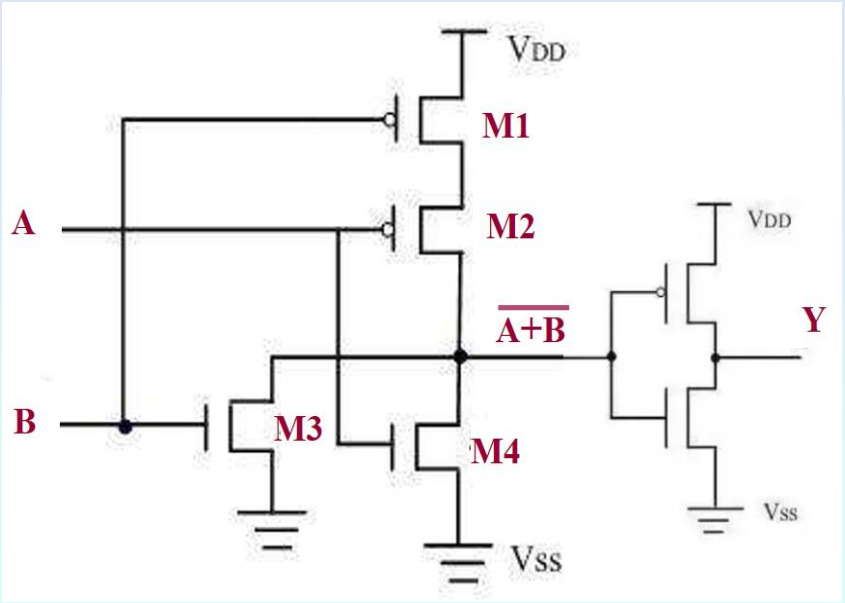


表4-6 或门真值表

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



4. 异或及同或逻辑

异或逻辑指的是：参与异或逻辑运算的多个变量中，若有奇数个变量值为1，则运算结果为1；反之若有偶数个变量值为1，则运算结果为0。异或逻辑符号表示为⊕。

同或逻辑指的是：参与同或逻辑运算的多个变量中，若有奇数个变量值为1，则运算结果为0；反之若有偶数个变量值为1，则运算结果为1。同或逻辑符号表示为⊙。

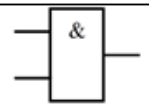
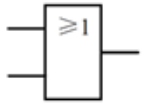
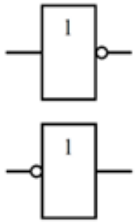
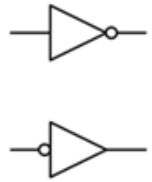
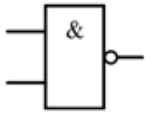
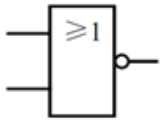
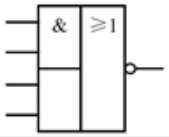
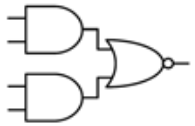
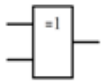
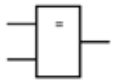
表4-7 两变量的异或及同或逻辑真值表

A	B	$A\oplus B$	$A\odot B$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

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

异或及同或的性质（课本：P82）

4. 常用的逻辑门及符号表示

序号	名称	国标图形符号	国外流行符号	表达式
1	与门			Y=AB
2	或门			Y=A+B
3	非门			$Y=\overline{A}$
4	与非门			$Y=\overline{AB}$
5	或非门			$Y=\overline{A+B}$
6	与或非门			$Y=\overline{AB+CD}$
7	异或门			$Y=A\overline{B}+\overline{A}B=A\oplus B$
8	同或门			$Y=AB+\overline{A}\,\overline{B}=A\odot B$

4.3.2 逻辑代数基本定律

1．逻辑代数的基本定律和公式（理解记忆）

表4-9 逻辑代数的基本定律和公式

名称	公式1	公式2
0-1律	$A \cdot 1 = A$ $A \cdot 0 = 0$	$A + 0 = A$ $A + 1 = 1$
互补率	$A\bar{A} = 0$	$A + \bar{A} = 1$
重叠率	$AA = A$	$A + A = A$
交换率	$AB = BA$	$A + B = B + A$
结合律	$A(BC) = (AB)C$	$A + (B + C) = (A + B) + C$
分配律	$A(B + C) = AB + AC$	$A + BC = (A + B)(A + C)$
反演律	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\,\bar{B}$
对合律	$\bar{\bar{A}} = A$	
吸收律	$A(A + B) = A$ $A(\bar{A} + B) = AB$ $(A + B)(\bar{A} + C)(B + C) = (A + B)(\bar{A} + C)$	$A + AB = A$ $A + \bar{A}B = A + B$ $AB + \bar{A}C + BC = AB + \bar{A}C$

吸收率可以用其他基本公式推导出来

2．逻辑代数基本运算规则（理解记忆）

逻辑代数三个规则：

（1）代入规则。在任一逻辑等式中，如果将等式两边所有出现的某一变量都代之以一个逻辑函数，则此等式仍然成立，这一规则称之为代入规则。

（2）反演规则。已知一逻辑函数F，求其反函数时，只要将原函数F中所有的原变量变为反变量，反变量变为原变量；“＋”变为“·”，“·”变为“＋”；“0”变为“1”；“1”变为“0”。

（3）对偶规则。已知一逻辑函数F，只要将原函数F中所有的“＋”变为“·”，“·”变为“＋”；“0”变为“1”；“1”变为“0”，而变量保持不变、原函数的运算先后顺序保持不变，那么就可以得到一个新函数，这新函数就是对偶函数F'。其对偶与原函数具有如下特点：①原函数与对偶函数互为对偶函数；②任两个相等的函数，其对偶函数也相等。

4.4 逻辑函数的表示与化简

4.4.1 逻辑函数的表示方法

在数字系统的逻辑电路中，如果某一输出变量与一组输入变量存在着一定的对应关系，当输入变量取任意一组确定的值，输出变量的值也就唯一地被确定，则称这种关系为逻辑函数关系。即用有限个与、或、非逻辑运算符，按某种逻辑关系将逻辑变量a、b、c、...连接起来，所得的表达式 $f=f$（a、b、c、...）称为逻辑函数。

逻辑函数有如下特点：

（1）逻辑变量和逻辑函数的取值只有0和1两种可能；（2）逻辑函数和逻辑变量之间的关系是由“或”、“与”、“非”三种基本逻辑运算决定的。

描述逻辑函数的常用方法有5种表示形式：布尔代数法、真值表、逻辑图、波形图及卡诺图。

1．布尔代数法

布尔代数法产生函数的逻辑表达式：是由逻辑变量和与、或、非三种运算符连接起来所构成的式子。逻辑函数表达形式不是唯一的。其优点是：书写简洁方便，易用公式和定理进行运算、变换。缺点是：逻辑函数较复杂时，难以直接从变量取值看出函数的值。表达式列写方法：可利用真值表列出表达式，取真值表中F=1的组合，输入变量值为1的表示成原变量，值为0的表示成反变量，然后将各变量相乘，最后将各乘积项相加，即得到函数的与或表达式。例如：$F=AB+BC+AC$。

2．真值表

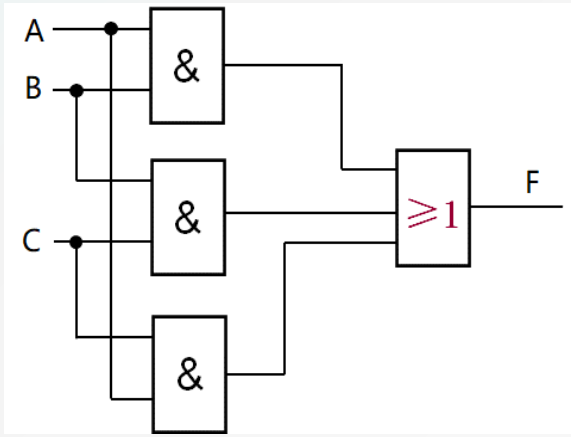
真值表定义为：输入变量不同取值组合与函数值间的对应关系列成表格。真值表具有唯一性。其优点是：直观明了，便于将实际逻辑问题抽象成数学表达式。缺点是：难以用公式和定理进行运算和变换；量较多时，列函数真值表较繁琐。

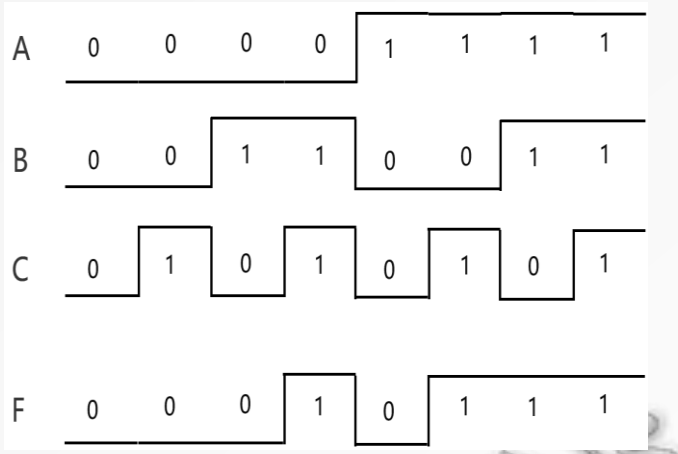
3．逻辑图

逻辑图是由基本门或复合门等逻辑符号及它们的连线构成的图。同一种逻辑功能可用不同的逻辑电路图表示，因此逻辑图不具有唯一性。其优点是：最接近实际电路。缺点是：不能进行运算和变换，所表示的逻辑关系不直观。

4．波形图

波形图是由输入变量的所有可能取值组合的高、低电平及其对应的输出函数值的高、低电平所构成的图形，即输入变量和对应的输出变量随时间变化的波形。其优点是：形象直观地表示了变量取值与函数值在时间上的对应关系，且具有唯一性。缺点是：难以用公式和定理进行运算和变换，当变量个数增多时，画图较麻烦。





5．卡诺图

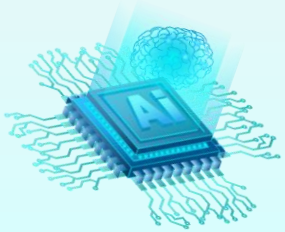
卡诺图是将逻辑函数真值表中的各行排列成矩阵形式，在矩阵的左方和上方按照格雷码的顺序写上输入变量的取值，在矩阵的各个小方格内填入输入变量各组取值所对应的输出函数值，这样构成的图形就是卡诺图。（下一小节阐述）

AB / C	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$F=AB+BC+AC$ 的卡诺图

4.4.2 逻辑函数的代数化简方法（掌握基本化简方法）

一个逻辑函数可以有多种表达方式，虽然繁简程度有所差异，但是逻辑功能完成等效，将较繁的逻辑表达式变换为与之等效的最简表达式称之为逻辑函数的化简。逻辑函数和实现逻辑函数的数字电路是对应的，逻辑函数简化了，则对应的数字电路也相应变简单了。常用的两种化简方式为逻辑代数法和卡诺图法。逻辑代数法化简没有规定方法步骤，结果是否最简也难以判断；卡诺图法有固定步骤，易于掌握，按步骤化简后可得最简表达式，本节将重点介绍卡诺图化简法。



逻辑代数化简法是利用逻辑代数的基本定理和公式对逻辑函数进行等式变换来获得最简结果的方法。

（课堂练习）

【例4-5】 试化简$F = AC + ACD + \overline{D} + \overline{A}BC\overline{D}$。

解：

$$F = AC + ACD + \overline{D} + \overline{A}BC\overline{D}$$ 分配律

$$= AC(1 + D) + \overline{D}(1 + BC\overline{D}) = AC + \overline{D}$$ 0-1律

【例4-6】 试化简$F = AB + A\overline{B} + AD + \overline{A}C + BD + ACEF + \overline{B}EF$。

解：

$$F = AB + A\overline{B} + AD + \overline{A}C + BD + ACEF + \overline{B}EF$$

$$= A(B + \overline{B}) + AD + \overline{A}C + BD + ACEF + \overline{B}E$$ 互补律

$$= A(1 + D + CEF) + \overline{A}C + BD + \overline{B}EF$$ 0-1律

$$= A + \overline{A}C + BD + \overline{B}EF = A + C + BD + \overline{B}EF$$ 吸收律

4.4.3 逻辑函数的卡诺图化简方法（难点，掌握）

1. 逻辑函数的最小项及其表示

逻辑函数的最小项是一个包括所有输入变量的乘积项，每个变量均以原变量或反变量的形式出现一次。每个最小项都包括所有变量且每个变量出现且仅出项1次，含有n个输入变量的函数共有2^n个最小项。

如：$Y_1=F_1(A,B)$，函数$F_1(A,B)$有4个最小项$\overline{A}\overline{B}$，$\overline{A}B$，$A\overline{B}$和AB。

$Y_2=F_2(A,B,C)$，函数$F_2(A,B,C)$共有8个最小项$\overline{A}\overline{B}\overline{C}$，$\overline{A}\overline{B}C$，$\overline{A}B\overline{C}$，$\overline{A}BC$，$A\overline{B}\overline{C}$，$A\overline{B}C$，$AB\overline{C}$，$ABC$。最小项例。

最小项的性质：任一最小项，只有一组对应变量取值使其值为1；任意两个不同的最小项的乘积为零；全体最小项之和为1。

最小项的编号及表达式（从0开始编号）：

m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7

最小项表达式（标准与或式）：最小项之和的形式。

逻辑公式展开法获取逻辑函数的标准与或式：第一步，将逻辑函数展开成与或表达式；第二步，利用$(\overline{X}+X)$为表达式中与式项补充缺项变量X；第三步，再次展开逻辑函数为与或表达式；第四步，消除重复项即可得逻辑函数的标准与或式。

课堂练习：【例4-8】

【例4-9】

$$
\begin{aligned}
Y &= \overline{AB + AD + \overline{B}C} && \text{反演律}\\
&= (\overline{A} + \overline{B})(\overline{A} + \overline{D})(B + \overline{C}) && \text{分配律、交换律}\\
&= (\overline{A} + \overline{AD} + \overline{AB} + \overline{BD})(B + \overline{C}) && \text{吸收律}\\
&= (\overline{A} + \overline{BD})(B + \overline{C}) && \text{分配律、互补律}\\
&= \overline{A}B + \overline{AC} + \overline{BCD}\\
&= (\overline{A}BC + \overline{A}B\overline{C}) + (\overline{A}B\overline{C} + \overline{A}BC) + (A\overline{BCD} + \overline{A}BCD)\\
&= \overline{A}BC + \overline{A}B\overline{C} + \overline{A}BC + A\overline{BCD} + \overline{A}BCD\\
&= \overline{A}BCD + \overline{A}BC\overline{D} + \overline{A}B\overline{C}D + \overline{A}B\overline{C}\overline{D} + \overline{A}BCD + \overline{A}BCD + A\overline{B}CD\\
&= m_7 + m_6 + m_5 + m_4 + m_1 + m_0 + m_8\\
&= \sum m_{(0,1,4,5,6,7,8)}
\end{aligned}
$$

【例4-10】利用真值表写出逻辑函数$Y = A + BC$的标准与或式。

①列出$Y = A + BC$的真值表：

A	B	C	A+BC
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

②写出$Y = A + BC$的标准与或式：

$$Y = A + BC = \overline{A}BC + A\overline{BC} + A\overline{B}C + AB\overline{C} + ABC = \sum m_{(3,4,5,6,7)}$$

2. 卡诺图的构成

预备知识：**格雷码**

在一组数的编码中，若任意两个相邻的代码只有一位二进制数不同，则称这种编码为**格雷码**（Gray Code）。

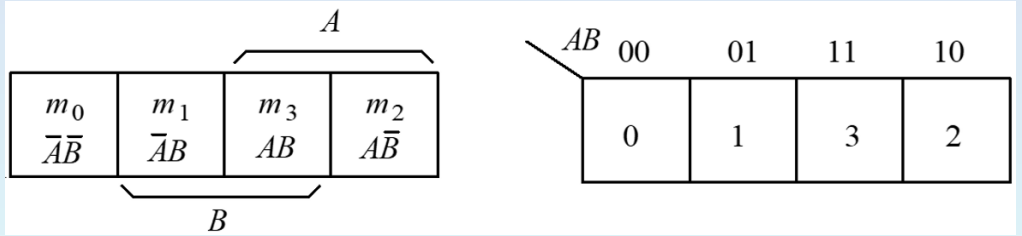
十进制数	4位自然二进制码	4位典型格雷码
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100

卡诺图的构成：卡诺图是用直角坐标来划分逻辑平面，形成棋杆式方格，每个方格相当于输入变量的一种组合。方格中所填逻辑值即为对应输出函数值，方格编号是输入变量按照二进制权重的排序。坐标的划分应使变量在相邻方格间按格雷码排列，以便函数在相邻方格间的吸收合并，最终达到最简化简的目的。

（注意：$\overline{X}$对应0，X对应1）

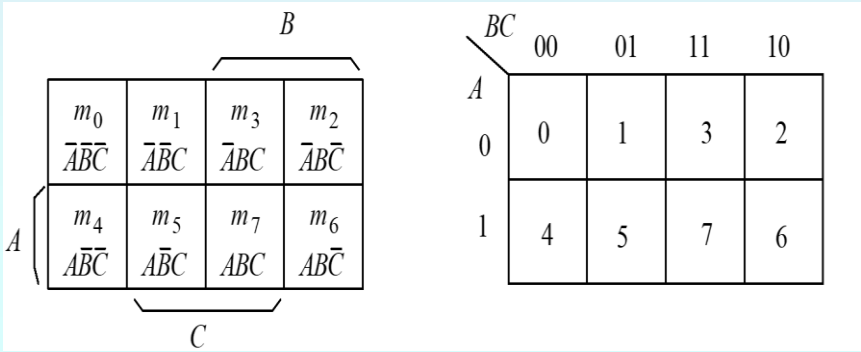
二变量卡诺图：

（数字序号记忆一下）



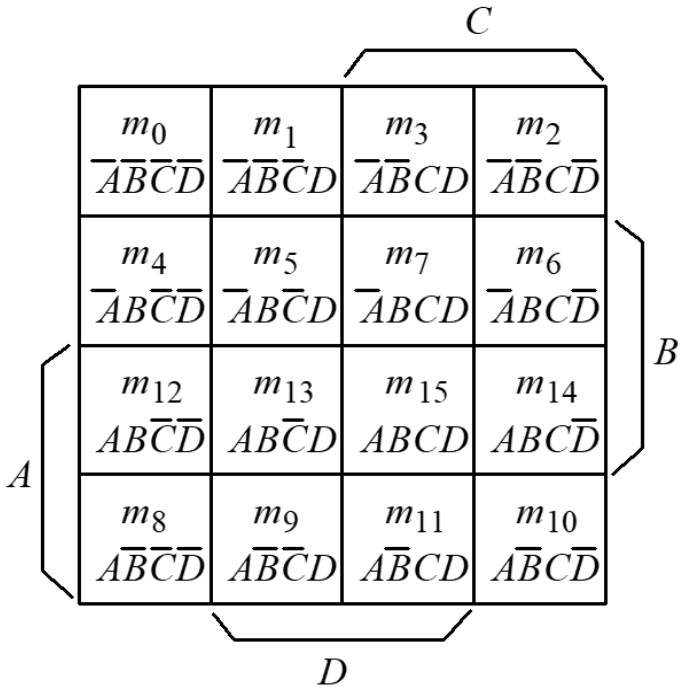
三变量卡诺图：

（数字序号记忆一下）



四变量卡诺图：

（数字序号记忆一下）



AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

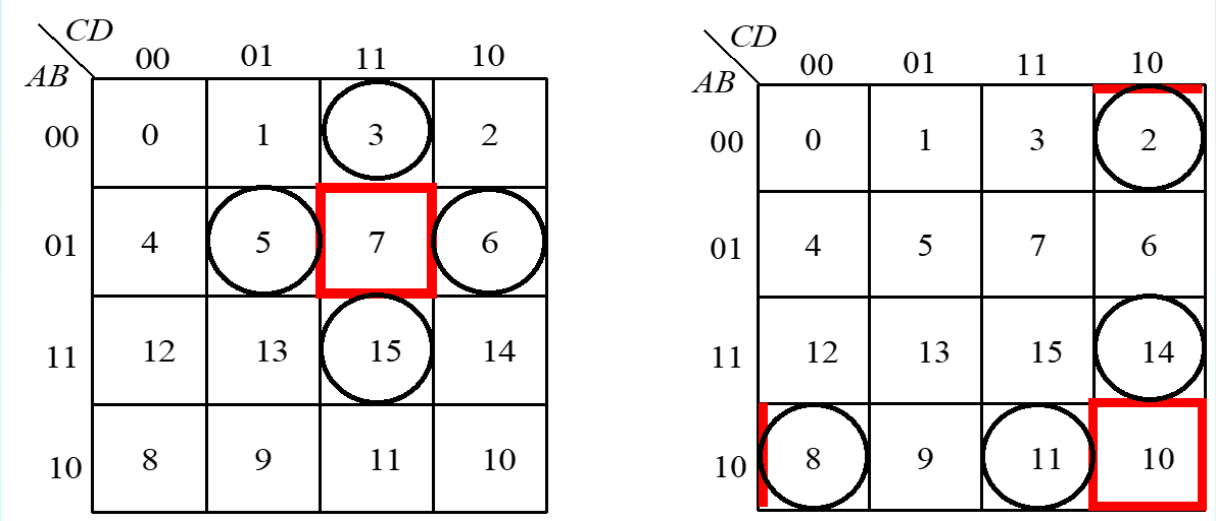
四变量卡诺图（课堂练习）

卡诺图原则： 最小项个数2^N,几何相邻的必须逻辑相邻

逻辑相邻： 只有一个变量取值不同其余变量均相同。逻辑相邻的最小项可以合并。

几何相邻： ①相邻——紧挨；②相对——任一行或一列的两头；③相重——对折起来后位置相重。

四变量最小项m_7和m_{10}的相邻项



3. 卡诺图化简方法

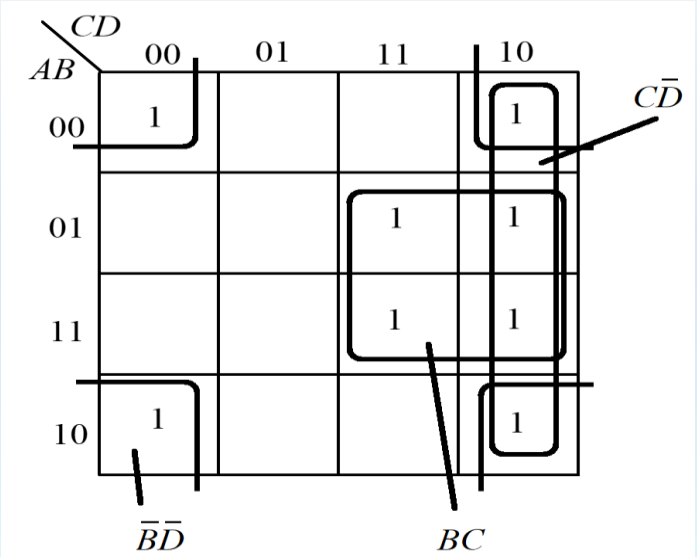
卡诺图化简逻辑函数的基本原理，是依据关系式$A\overline{B}+AB=A$，即两个“与”项中，如果只有一个变量相反，其余变量均相同，则这两个“与”项可以合并成一项，消去其中互反的变量。

相邻最小项用倒角矩形圈(或椭圆形圈)圈起来，称为卡诺圈。在合并项(卡诺圈)所处位置上，若某变量的代码有0也有1，则该变量被消去，否则该变量被保留，并按0为反变量，1为原变量的原则写成乘积项形式的合并项中。

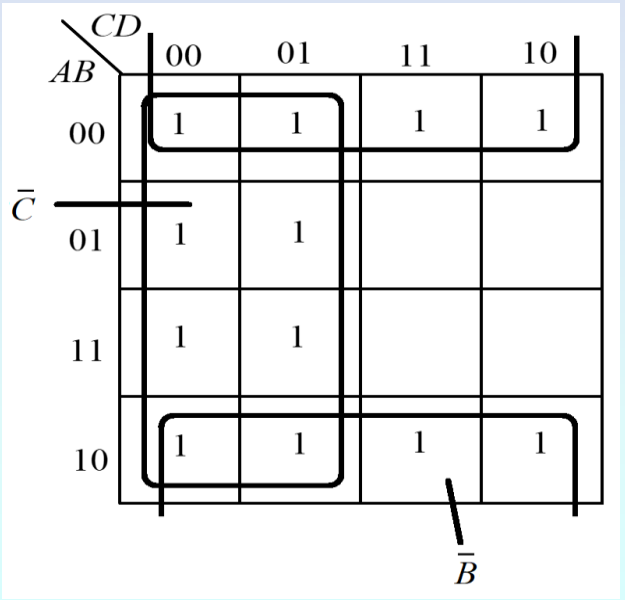
1）卡诺图化简逻辑函数的原理

（1）2个相邻的最小项结合，可以**消去1个取值不同的变量**而合并为1项

（2）4个相邻的最小项结合，可以**消去2个取值不同的变量**而合并为1项



（3）8个相邻的最小项结合，可以**消去3个取值不同的变量**而合并为1项



2）卡诺图化简步骤

（1）画出逻辑函数的卡诺图。

（2）合并相邻的最小项，画卡诺圈。

（3）写出化简后的表达式。每一个圈写一个最简与项，规则是，取值为l的变量用原变量表示，取值为0的变量用反变量表示，将这些变量相与。然后将所有与项进行逻辑加，即得最简与—或表达式。

3）画卡诺圈的原则

（1）尽量画大圈，但每个圈内只能含有2^n（n=0,1,2,3.....）个相邻项。要特别注意对边相邻性和四角相邻性。

（2）圈的个数尽量少。

（3）卡诺图中所有取值为1的方格均要被圈过，即不能漏下取值为1的最小项。

（4）在新画的包围圈中至少要含有1个未被圈过的1方格，否则该包围圈是多余的。

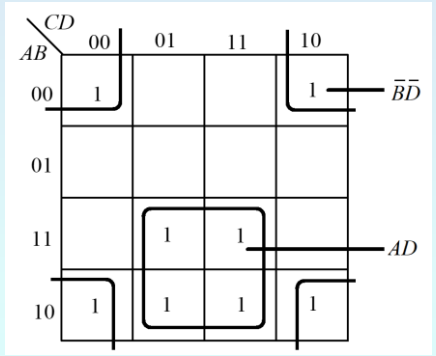
【例4-11】 用卡诺图化简逻辑函数$F = AD + A\overline{B}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D}$。

解：

（1）逻辑函数表达式分解获得最小项

$$F = AD + A\overline{B}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D}$$
$$= A(B+\overline{B})(C+\overline{C})D + A\overline{B}\,(C+\overline{C})\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D}$$
$$= ABCD + AB\overline{C}D + A\overline{B}\,CD + A\overline{B}\,\overline{C}D + A\overline{B}\,C\overline{D} + A\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D}$$
$$= \sum m_{(15,13,11,9,10,8,0,2)}$$

（2）填卡诺图，画卡诺圈合并最小项



（3）读图得最简与或表达式 $F = AD + \overline{B}\,\overline{D}$

4. 无关项的逻辑函数化简 （难点）

在实际数字电路中，当逻辑变量被赋予特定含义时，有一些变量的取值组合根本就不会出现，或者对应于变量的某些取值，其函数值可以是任意的（0或1），我们将变量取这些值对应的最小项称之为无关项、约束项或任意项。

有无关项的逻辑函数用$F=\sum m+\sum d$表示，其中$\sum m$为相关的最小项，$\sum d$为无关最小项。无关项在逻辑函数的输出中一般用“×”表示，化简过程中既可以当“0”不作处理，也可以当作“1”，参与卡诺圈的合并销项。

包含无关最小项的逻辑函数称为不完全确定的逻辑函数，此类逻辑函数化简时适当利用无关项可进一步优化简化结果。

【例4-12】 要求设计一个逻辑电路，能够判断一位十进制数是奇数还是偶数。当十进制数为奇数时，电路输出1；当十进制数为偶数时，电路输出0。

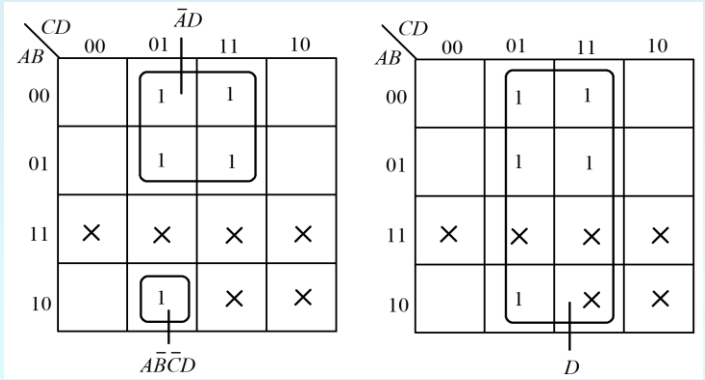
解：

（1）列出真值表获得最小项

ABCD	F	ABCD	F	ABCD	F	ABCD	F
0000	0	0100	0	1000	0	1100	×
0001	1	0101	1	1001	1	1101	×
0010	0	0110	0	1010	×	1110	×
0011	1	0111	1	1011	×	1111	×

无关项：不能有大于等于10的，所以所有大于等于10的画“×”

（2）填卡诺图，画卡诺圈合并最小项



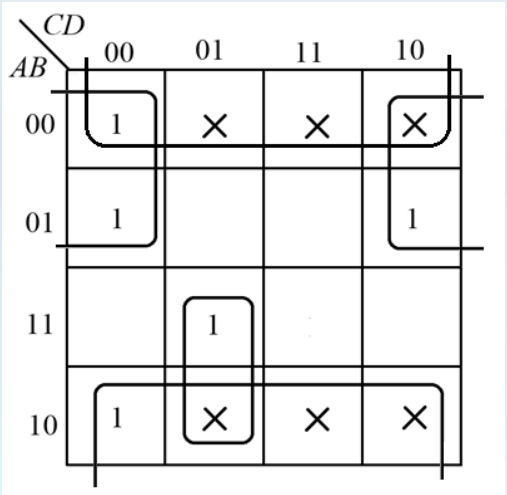
（3）读图得最简与或表达式

无关项不参与简化 $F = \overline{A}D + A\overline{B}\,\overline{C}D$　　无关项参与化简 $F = D$　　（A、B、C任意）

【例4-13】化简$F(A,B,C,D)=\sum m_{(0,4,6,8,13)}+\sum d_{(1,2,3,9,10,11)}$。

解：

（1）填卡诺图，画卡诺圈合并最小项



对无关项$\sum d_{(1,2,3,9,10,11)}$直接标记“×"

（2）读图得最简与或表达式

$F(A,B,C,D)=\overline{B}+\overline{A}\,\overline{D}+A\overline{C}D$

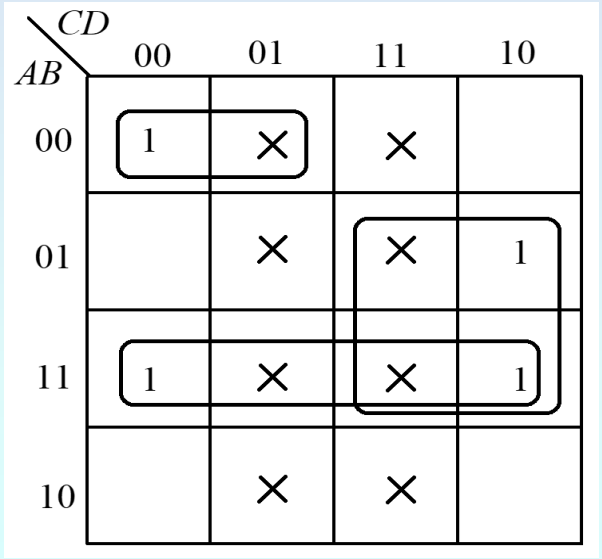
【例4-14】 化简$F=\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}+AB\overline{C}\,\overline{D}+\overline{A}\,BC\overline{D}+ABC\overline{D}$，约束条件$D=0$。

解：

（1）计算约束项。约束条件D = 0 表示 D值必须为0，满足约束条件的ABCD的所有组合为：0000 0010 0100 0110 1000 1010 1100 1110，即项第0，2，4，6，8，10，12，14项，除此之外的项为约束项，即第1，2，5，7，9，11，13，15项。

填卡诺图时将约束项和原本已经标为1的地方之外的空格画 X。

（2）填卡诺图，画卡诺圈合并最小项



（3）读图得最简与或表达式

$F=AB+BC+\overline{A}\,\overline{B}\,\overline{C}$

课堂练习：习题5(2)中的约束条件$AB+AC=0$，应该标哪几个为×？
（过去部分同学标错）

AB不能为1，即A、B不能同时为1
因此：12、13、15、14
打×

AC不能为1，即A、C不能同时为1

因此：15、14、11、10
打×

所以，打×有：12、13、15、14，
11、10

AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

Thank you