第2章
逻辑代数基础

Leng Ximo

逻辑代数基础

- 逻辑代数中的基本运算
- 逻辑代数的公式与定理
- 逻辑函数及其表示方法
- 逻辑函数的化简

逻辑代数中的基本运算

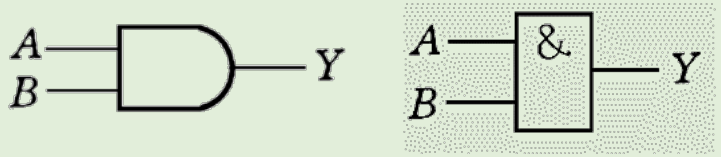
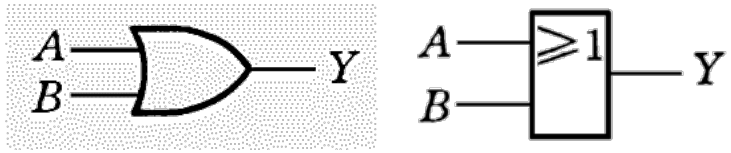
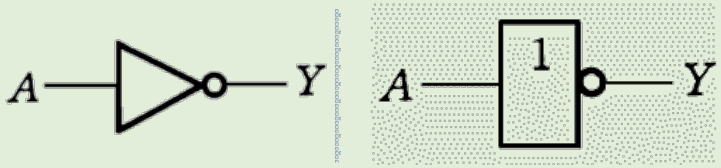
逻辑代数概述 —— 本章要解决的问题

“逻辑”指的是事物之间的因果关系，在数字逻辑电路中我们采用 0 和 1 来描述一个事物（事件）的两种不同逻辑状态，即一般通用的二值逻辑；

本章关注的内容是，在已经将现实世界中的客观事物抽象为 0 和 1 后，如何根据逻辑代数的知识确定各个事物的逻辑关系（具体至数学表达式）；至于具体采用什么电路来实现这些逻辑运算，则是第三章门电路研究的内容而非本章的关注点；

逻辑代数中的基本运算

三种基本运算

	表达式	逻辑关系含义	符号
与 AND	$A \cdot B$ 一般简写为 AB	决定事物结果的全部条件同时具备， 结果才发生	
或 OR	$A + B$	决定事物结果的条件至少有一个满足， 结果就发生	
非 NOT	A' 或 $\overline{A}$	条件具备结果不发生， 条件不具备结果发生， 对立关系	

逻辑代数中的基本运算

真值表

表征逻辑事件输入（条件）和输出（结果）之间全部可能状态的表格；

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

与

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

或

A	Y
0	1
1	0

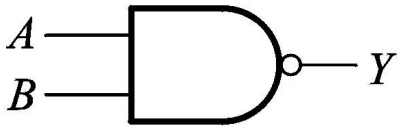
非

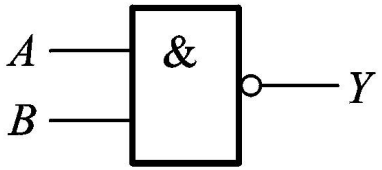
逻辑代数中的基本运算

要记住这些逻辑运算命名的规则和意义
例如后面学习会用到的“与或式” 和“或与式”！

其他复合逻辑运算

与非 —— “先与再非”

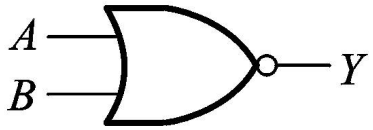


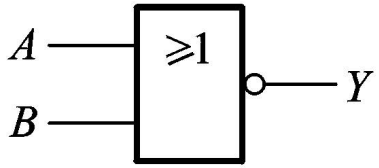


与非

$Y=(A\cdot B)'$

或非 —— “先或再非”

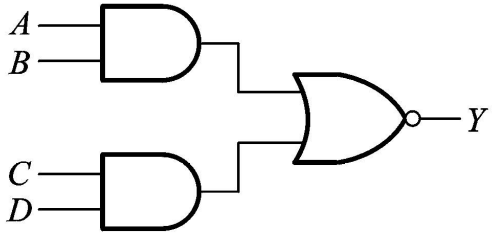


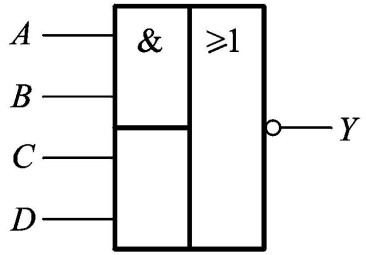


或非

$Y=(A+B)'$

与或非 —— “先与再或再非”





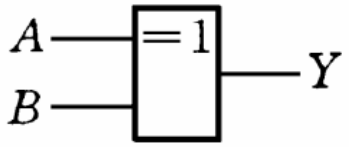
与或非

$Y=(A\cdot B+C\cdot D)'$

逻辑代数中的基本运算

其他复合逻辑运算

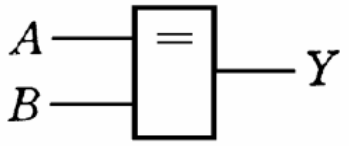
- 异或 —— 不同时输出 1，相同时输出 0；同或 —— 相同时输出 1，不同时输出 0；
- 异或和同或两种运算关系是对立（互补）的，即异或取反即为同或；
- 异或和同或都可以用基本的与、或、非运算关系表示；



异或

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

$$Y = A \oplus B = A'B + AB'$$



同或

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

$$Y = A \odot B = AB + A'B'$$

逻辑代数中的基本运算

在后续的微机原理课程
以及汇编语言程序设计等
仍会用到！

补充：与、或、非、异或的用途

- 与运算 AND ，可以用来将指定的位置 0 ，例如想将 10111101 的第五位和第六位（从右往左）置零，则可以 (10111101) AND (11001111)，因为任何变量和 1 相与不改变此变量，任何变量和 0 相与都为 0；
- 或运算 OR ，可以用来将指定的位置 1 ，和与运算刚好相反，任何变量和 0 相或都不改变此变量，任何变量和 1 相或都为 1；
- 非运算 NOT ，用于取反，但是只能同时将整体取反，不能将指定位取反；
- 异或运算 XOR ，用于指定位的取反，任何变量与 0 异或都不改变此变量，任何变量与 1 异或得到的是反变量；例如想将 10111101 的低四位取反，则可以(10111101) XOR (00001111)；

逻辑代数的公式和定理

思考：
怎样证明一个逻辑公式的正确性？
（根据逻辑关系推演
或
用真值表遍历）

逻辑代数的基本公式

序号	公　　式	序号	公　　式
		10	$1' = 0;\ 0' = 1$
1	$0 \cdot A = 0$	11	$1 + A = 1$
2	$1 \cdot A = A$	12	$0 + A = A$
3	$A \cdot A = A$	13	$A + A = A$
4	$A \cdot A' = 0$	14	$A + A' = 1$
5	$AB = BA$	15	$A + B = B + A$
6	$A(BC) = (AB)C$	16	$A + (B + C) = (A + B) + C$
7	$A(B + C) = AB + AC$	17	$A + BC = (A + B)(A + C)$
8	$(AB)' = A' + B'$	18	$(A + B)' = A'B'$
9	$(A')' = A$		

必须熟练掌握

初学时、复习时
认真过一遍
不要刻意去死记硬背！

注意几个特别强调的公式

逻辑代数的公式和定理

需要强调的基本公式 —— 分配律

$$A(B+C)=AB+AC \qquad A+BC=(A+B)(A+C)$$

用途：

将“混杂”在一起的变量分离，将逻辑函数表达式化成“与或”形式和“或与”形式；在后面学习逻辑函数的“最小项之和”和“最大项之积”表示时非常有用！

逻辑代数的公式和定理

需要强调的基本公式 —— 反演律（德·摩根定律）

$$(AB)' = A' + B' \qquad (A + B)' = A'B'$$

记忆方法：“与的非等于非的或，或的非等于非的与”

与在概率论课程中学习的德摩根定律一致，即“交的反等于反的并，并的反等于反的交”

用途：

实现与运算和或运算之间的转换，在后面对函数式进行特定形式的变换时非常有用；

逻辑代数的公式和定理

通过多练习来帮助记忆和熟练应用!

逻辑代数的常用公式

序　号	公　式	说　明
21	$A + AB = A$	吸收律
22	$A + A'B = A + B$	消去律
23	$AB + AB' = A$	在写成最小项之和形式时经常逆向使用
24	$A\,(A + B) = A$	——
25	$AB + A'C + BC = AB + A'C$ $AB + A'C + BCD = AB + A'C$	多余项定律：○□+○′△+(□△…) =○□+○′△ 在解决含有多个变量（例如超过5个）的化简问题时常用
26	$A(AB)' = AB'$; $A'(AB)' = A'$	可以看作是公式21和22的反演式 不用单独记忆

逻辑代数的公式和定理

理解每一个定理的含义而非去背!

逻辑代数的基本定理

- 代入定理：

在任何一个包含 A 的逻辑等式中，若以另外一个逻辑式代入式中 A 的位置，则等式依然成立；

- 反演定理：

对任一逻辑式 Y ，在求解它的反时（即已知 Y 求解 Y′ ），只需要在保证运算优先次序不变的原则下，将其中所有的 · 换成 +，+ 换成 ·，原变量换成反变量，反变量换成原变量（不属于单个变量上的反号不动），1 变成 0，0 变成 1，得到的结果就是 Y′ ；

（不推荐基础较差的同学使用，容易错！建议逐次用德·摩根定律更稳妥！）

逻辑代数的公式和定理

理解每一个定理的含义而非去背!

逻辑代数的基本定理

- 对偶定理

若两逻辑表达式相等，则它们的对偶式也相等；

所谓的对偶式指的是在保持运算优先顺序不变的原则下，将表达式中所有的 · 换成 +，+ 换成 ·，0 换成 1，1 换成 0；（注意与反演定理的区别！变量不取反！）

主要应用于证明题当中，证明两个形式较复杂的逻辑式相等，可以尝试两边同时取对偶式，证明它们的对偶式相等；或者是，先取对偶式，对对偶式进行化简之后，再取对偶就可以得到原逻辑表达式的化简结果；

逻辑函数及其表示方法

逻辑函数

以逻辑变量为输入，运算结果为输出，则输入变量值确定以后，输出的取值也随之而定，因此输入—输出之间是一种函数关系，称作逻辑函数；

在我们学习的二值逻辑中，输入变量和输出变量只有 0 和 1 两种状态；

注意：
在求解实际应用问题中的逻辑函数关系时，首先必须做的事情是将具体事物抽象为逻辑变量并规定其不同状态（0 或 1），这是很重要的解题习惯，类似于在概率论答题时首先要记不同的事件为 A、B.....；本章只研究现有的抽象后的逻辑函数式；

逻辑函数及其表示方法

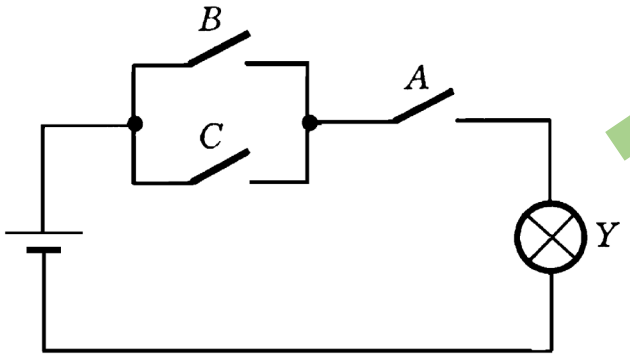
逻辑函数的不同表示方法

- 真值表 —— 遍历输入变量所有取值下的输出值，列成表格；
- 逻辑函数式 —— 将输出写成输入变量的与、或、非等逻辑运算的组合式；
- 逻辑图 —— 逻辑函数式用图形符号表示；
- 波形图 ——输入变量所有取值可能与对应输出的时域波形；
- 卡诺图（见下一节）
- 计算机软件中的描述方式（例如 Verilog 语言）

逻辑函数及其表示方法

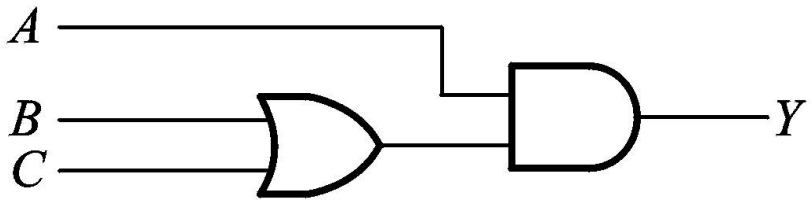
逻辑函数的不同表示方法示例

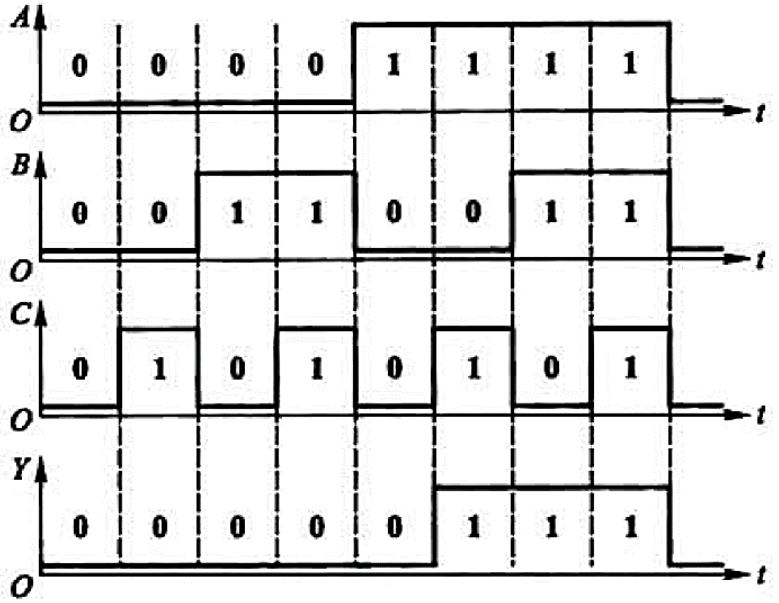
$$Y = A \cdot (B + C)$$



记开关 A、B、C
闭合为 1，断开为 0；
记灯 Y 亮为 1，灭为 0；

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





逻辑函数及其表示方法

逻辑函数不同表示方法间的转换

- 从真值表得到逻辑函数式（重点）

Step 1：找出真值表中使逻辑函数输出 Y = 1的输入变量取值的组合即使 Y = 1的行；

Step 2：使 Y = 1的每组输入变量取值的组合（即每一行）对应一个乘积项，

取值为 1 写入原变量，取值为 0 写入反变量；

Step 3：将各行对应的乘积项相加，即最终结果为“与或”形式；

从真值表得到逻辑函数式是几种不同表示方法转换中最重要的一个，因为在后面数字电路的设计的学习中，很多时候输入和输出的逻辑关系比较复杂，不能直接看出逻辑函数式，这个时候可以通过遍历得到真值表，再将真值表转化为对应的逻辑函数式，其他表示方法也就自然而然地得到了；而且根据真值表转化得到的这种与或形式的逻辑函数式在下一节逻辑函数的化简时也有很大用处，因为恰好每一行对应一个最小项；

逻辑函数及其表示方法

不用去刻意背文字内容！
对一个例子具体操作一遍熟练即可！

逻辑函数不同表示方法间的转换

- 逻辑函数式⟷逻辑图

逻辑函数式→逻辑图： 用图形符号代替逻辑函数式中的逻辑运算符；
逻辑图→逻辑函数式： 从输入到输出逐级写出每个图形符号对应的逻辑运算式；

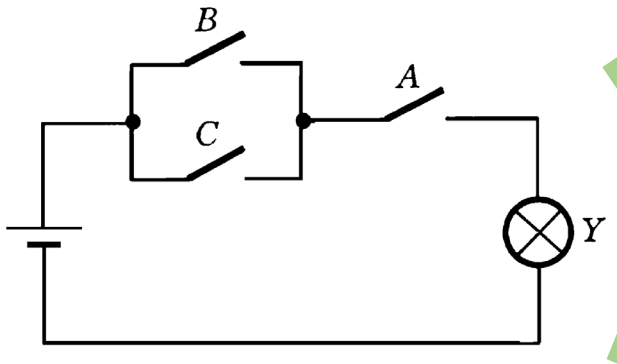
- 真值表⟷波形图

真值表→波形图： 将所有输入变量与对应输出变量的取值依次排列，以时间为横轴；
波形图→真值表： 在波形图中找出每个时间段里输入变量与输出变量的取值列成表格；
（对于 N 个输入变量要对应 2^N 个组合即 2^N 行）

逻辑函数及其表示方法

逻辑函数的标准形式

$$Y = A \cdot (B + C)$$



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

寻找一个标准的，唯一的
对于任何逻辑函数都通用的形式？

$$Y = AB'C + ABC' + ABC$$

逻辑函数及其表示方法

不用去刻意背文字定义！
自己过一遍 2、3、4、5变量的真值表和最小项！

最小项

- 最小项 m 的定义：

m 是乘积项，包含 n 个因子，n 个变量均以原变量和反变量的形式在 m 中出现一次；

事实上最小项就是在列写 n 个输入变量的逻辑函数的真值表时每一行对应的输入变量的乘积项，取值为 1 写入原变量，取值为 0 写入反变量；使最小项取值为 1 的变量取值，按照 n 位自然二进制数排列后对应的十进制数就是这个最小项的编号；

逻辑函数及其表示方法

最小项的特点

① 全体最小项之和为 1；

② 各个最小项之间是互斥的，即任何两个最小项之积为 0；
换句话说，对于任何一个特定的输入变量取值，有且仅有一个最小项为 1；

③ 两个相邻的最小项之和可以合并，消去一对因子，只留下公共因子；
这里的“相邻”不是按照自然二进制数编号后的相邻，而是指逻辑上的相邻，
规定仅一个变量不同的最小项彼此称为相邻；
例如 $A'B'C$、$A'BC'$、$A'BC$ 即 m_1、m_2、m_3，m_1 和 m_2 不是相邻，
m_1 和 m_3 才是相邻；m_1 和 m_3 可以合并消去因子 $B+B'$；

逻辑函数及其表示方法

逻辑函数的最小项之和形式

任何一个逻辑函数式都可以化为若干个最小项之和的形式；

前面学习的真值表到逻辑函数式的转化实质上就是在用最小项之和表示逻辑函数；

当给定的逻辑函数式不是标准的最小项之和的形式时，一般不断地利用 $A + A' = 1$ 补全缺少的因子，并结合分配律 $A(B+C) = AB + AC$ 将或与形式转化为与或形式；

逻辑函数及其表示方法

最大项

- 最大项 M 的定义：

M 是相加项，包含 n 个因子，n 个变量均以原变量和反变量的形式在 M 中出现一次；

可以把最大项看作是对每一个最小项应用反演定理，因此，最大项的编号原则是反的，即原变量记为 0，反变量记为 1，n 位自然二进制数排列后对应的十进制数为编号；
例如 $A+B+C'$ 的编号是 001 即 M_1；
（记忆方法：对于与运算即乘积项，得 1 “容易”，即得 1 时输入变量组唯一确定；
对于或运算即相加项，得 0 “容易”，即得 0 时输入变量组唯一确定；）

逻辑函数及其表示方法

最大项的特点

① 全体最大项之积为 0；

② 任何两个最大项之和为 1；
换句话说，对于任何一个特定的输入变量取值，有且仅有一个最小项为 0；

③ 两个相邻的最大项之积保留公共变量，这里的相邻和前面最小项的相邻一致，都是指逻辑上的相邻，即只有一个变量不同，例如 $A+B+C'$ 和 $A+B'+C'$ 为相邻，即 M_1 和 M_3 相邻，两者的乘积项即相与的结果是 $A+C'$；

逻辑函数及其表示方法

逻辑函数的最大项之积形式

任何一个逻辑函数式都可以化为若干个最大项之积的形式；

当给定的逻辑函数式不是标准的最大项之积的形式时，一般不断地利用 $A \cdot A' = 0$
补全缺少的因子，并结合分配律 $A + BC = (A + B)(A + C)$ 将与或形式转化为或与形式；

逻辑函数及其表示方法

直接变换得到最大项之积过程比较繁琐，
因此通常应用最小项解决问题！

最小项之和与最大项之积示例

$$Y = A'B + AC$$

$$\begin{aligned} Y &= A'B(C+C') + A(B+B')C \\ &= A'BC + A'BC' + ABC + AB'C \\ &= m_3 + m_2 + m_7 + m_5 \\ &= \sum m(2,3,5,7) \end{aligned}$$

$$\begin{aligned} Y &= (A'B + A)(A'B + C) \\ &= (A + B)(A'B + C) \\ &= (A + B)(A' + C)(B + C) \\ &= (A + B + CC')(A' + BB' + C)(AA' + B + C) \\ &= (A + B + C)(A + B + C')(A' + B + C)(A' + B' + C)(A + B + C)(A' + B + C) \\ &= (A + B + C)(A + B + C')(A' + B + C)(A' + B' + C) \\ &= M_0 \cdot M_1 \cdot M_4 \cdot M_6 \\ &= \prod M(0,1,4,6) \end{aligned}$$

逻辑函数及其表示方法

最小项与最大项的关系

根据最大项和最小项的定义以及德·摩根定理，易推导得到：

$$M_i = m_i'$$

如果将逻辑函数写成最小项之和的形式：$\sum m_i (i \in R_1)$

再将其写成最大项之积的形式：$\sum M_j (j \in R_2)$

两者的下标编号集合 R_1 和 R_2 是**互补的**；

所以很多时候不需要直接展开求解最大项之积！先求解最小项之和更方便！

逻辑函数的化简

逻辑函数的化简的目标

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

$$Y = AB'C + ABC' + ABC$$

两个非门，三个与门，两个或门

$$Y = AB + AC$$

两个与门，一个或门

$$Y = A(B + C)$$

一个与门，一个或门

逻辑函数的化简

逻辑函数的化简的目标

根据上一页的例子，逻辑函数的最小项之和虽然是通用的标准形式，但不一定就是最简的；在实际数字逻辑电路设计时，需要考虑在逻辑功能相同的条件下使用尽可能简单的电路，即电子器件数量足够少；

事实上，没有绝对的最简，由于我们习惯性将逻辑函数写成最小项之和即与或形式，因此对于与或形式的逻辑函数，包含的乘积项最少，且每个乘积项里的因子也不能再减少时，称其为最简形式即最简与或形式；其他形式的最简同理；例如最简或与式、最简与非—与非式、最简或非—或非式、最简与或非式；

逻辑函数的化简

公式化简法

使用前面学习的基本公式和常用公式；对基础要求较高，没有固定的步骤，难以确定结果正确与否，不通用但有时很快速好用；

对公式化简法，要求掌握但不必深入研究，熟悉教材上的例题和课后题即可；

（一个可能会用到公式化简法的情况是，当逻辑函数含有变量数很多时，即很难直接采用卡诺图去进行化简时，一般题目的设计会使解题时经常需要用 $AB + A'C + BCD = AB + A'C$ 这个公式来消去多余的变量，接着再用卡诺图去进行化简）

逻辑函数的化简

卡诺图化简法（重点）

卡诺图化简法的本质 —— 将最小项之和用图形描述，将逻辑上的相邻表现为几何图形的相邻，通过合并相邻的几何图形，合并后对应仅保留公共因子的最小项，最终结果就是最简与或式；

卡诺图：

以 2^n 个小方块分别代表 n 变量的所有最小项，并将它们排列成矩阵，而且使几何位置相邻的两个最小项在逻辑上也是相邻的（只有一个变量不同），就得到表示 n 变量全部最小项的卡诺图；

逻辑函数的化简

卡诺图的绘制

A \ B	0	1
0	$A'B'$ m_0	$A'B$ m_1
1	AB' m_2	AB m_3

A \ BC	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

不是按照自然二进制数排列，
按照格雷码排列！

AB \ CD	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

AB \ CDE	000	001	011	010	110	111	101	100
00	m_0	m_1	m_3	m_2	m_6	m_7	m_5	m_4
01	m_8	m_9	m_{11}	m_{10}	m_{14}	m_{15}	m_{13}	m_{12}
11	m_{24}	m_{25}	m_{27}	m_{26}	m_{30}	m_{31}	m_{29}	m_{28}
10	m_{16}	m_{17}	m_{19}	m_{18}	m_{22}	m_{23}	m_{21}	m_{20}

（目的：使逻辑上相邻对应几何上相邻，
保证相邻的方块只有一个变量不同）

逻辑函数的化简

卡诺图的绘制

A \ B	0	1
0	$A'B'$ m_0	$A'B$ m_1
1	AB' m_2	AB m_3

A \ BC	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

AB \ CD	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

AB \ CDE	000	001	011	010	110	111	101	100
00	m_0	m_1	m_3	m_2	m_6	m_7	m_5	m_4
01	m_8	m_9	m_{11}	m_{10}	m_{14}	m_{15}	m_{13}	m_{12}
11	m_{24}	m_{25}	m_{27}	m_{26}	m_{30}	m_{31}	m_{29}	m_{28}
10	m_{16}	m_{17}	m_{19}	m_{18}	m_{22}	m_{23}	m_{21}	m_{20}

记忆方法：

增加一个变量看作是折纸一样翻折，

轴对称位置上的最小项也满足相邻！
（把卡诺图看作是闭合的图形）

（教材上的卡诺图并不是唯一的，
只要能够保证逻辑对称性与
几何对称性一致即可，
最好去理解卡诺图绘制的过程）

逻辑函数的化简

不用去刻意背文字内容！
多通过题目练习熟练过程！

用卡诺图化简逻辑函数的步骤

Step 1 —— 将逻辑函数写成最小项之和的形式；

Step 2 —— 根据最小项之和绘制对应的卡诺图，逻辑函数式中含有的最小项的对应位置填 1，其余的位置填 0；

Step 3 —— 合并相邻的最小项，对应最终结果；

（注意：相邻表示逻辑相邻，轴对称位置上的方块也可合并！）

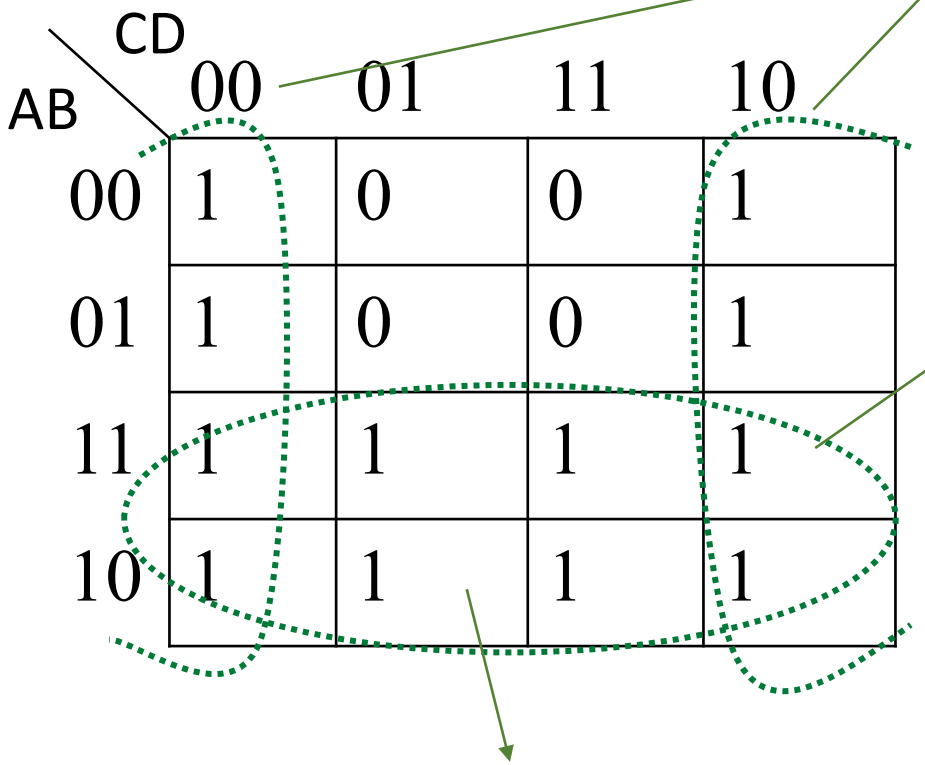
（注意：合并时，每个方块可以重复使用，因为 A+A=A）

（注意：合并的原则是“乘积项的数目最少，每个乘积项的因子最少”，即画的圈要尽量大（消去尽量多的不同的因子），画圈的数目尽量少）

（注意：结果不唯一，但乘积项的数目和每个乘积项的因子个数一致）

逻辑函数的化简

卡诺图相邻项合并示例



轴对称位置上的也可以合并，也叫做“相邻”

每个矩形可以重复使用！

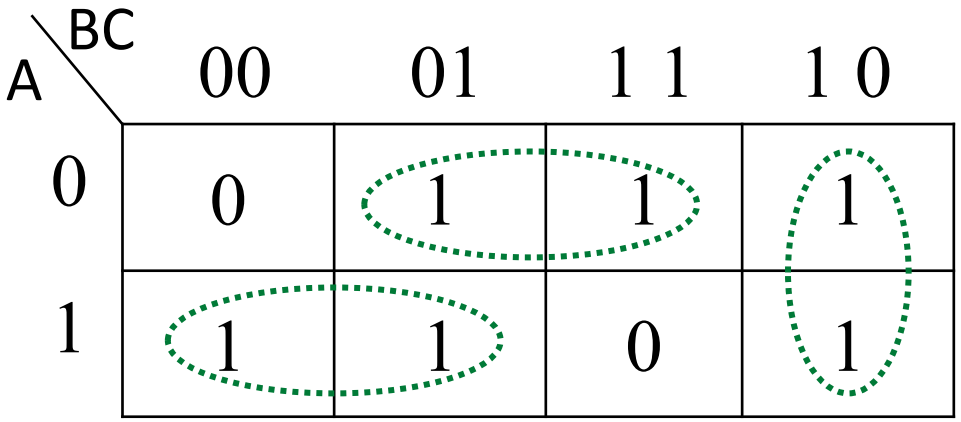
保证“画的圈最大，圈的个数最少”

2^n 个相邻的方块可以消去 n 个相异的变量

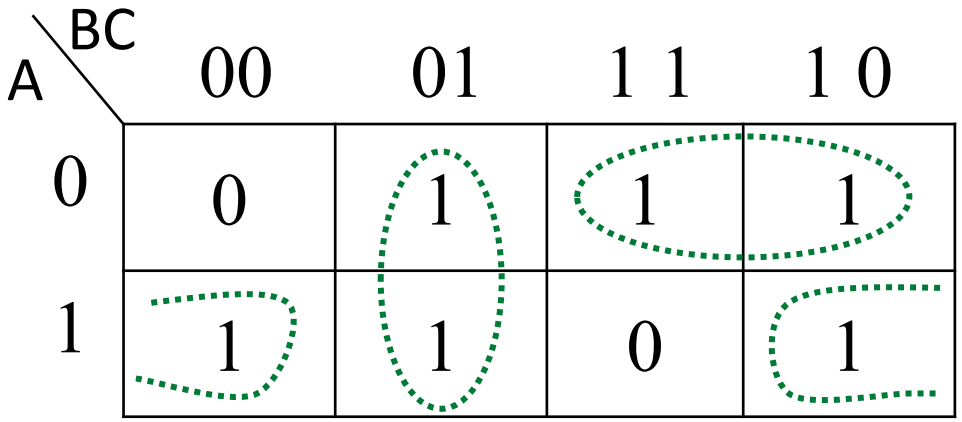
逻辑函数的化简

思考：
哪种表示方法是唯一的？
（最小项之和与最大项之积）
（真值表）

卡诺图相邻项合并示例



$$AB' + A'C + BC'$$



$$AC' + A'B + B'C$$

结果不唯一

但乘积项数目和每个乘积项的因子数一致（这也是检查答案是否正确的依据）

即圈的个数不能再减少，也找不到更大的圈；

逻辑函数的化简

逻辑函数的其他变换形式

实际的逻辑电路设计可能会受到电子器件种类与数量的限制，因此可能要进行变换；
前面化简得到的最简与或形式并不一定是设计时需要的逻辑函数形式；

- 与非—与非形式（只能用与非门实现）：
  将得到的与或形式取两次反，即逆向利用德·摩根定理；

- 与或非形式 / 或与形式：
  根据全部最小项之和为 1，将互补的其余的最小项求和后取反，即与或非形式；
  或者直接在卡诺图中合并为 0 的方块；对与或非形式应用德·摩根定理即或与形式；

- 或非—或非形式（只能用或非门实现）：
  将与或非形式中的每个乘积项再利用德·摩根定理，即变为或非形式；

逻辑函数的化简

含有无关项的逻辑函数的化简

约束项：实际应用问题中不可能出现（不允许出现）的输入组合对应的最小项称为约束项，约束条件表示为这些最小项均为 0，即和为 0；

任意项：实际应用问题中当输出的值是任意的（我们不关心的）、对电路功能无影响时的输入组合，对应的最小项称为任意项；

无论是约束项还是任意项，都是在实际应用问题背景下引入的概念；

约束项和任意项统称为无关项，无关项既可以出现在逻辑函数表达式当中也可以剔除出去，换句话说，它是否出现在逻辑函数表达式中无关紧要，对结果没有影响；

逻辑函数的化简

含有无关项的卡诺图化简

Step 1 —— 根据题意确定无关项；

一般第二章的题目会直接给出无关项，例如 “约束条件为 xxx +xxx +..... =0 ”以及 “∑d(...) = 0 ” （无关项 集合用 d 表示）这类已知条件；
后面在基于实际应用设计逻辑电路时应自行根据实际情况来分析确定无关项；

Step 2 —— 绘制卡诺图时，无关项的方块用 × 表示；

Step 3 —— 化简时根据需要，将无关项填入 1 或 0 ，原则是使化简后的项数最少，每项的因子最少，即“画的圈最大，圈的个数最少”；

逻辑函数的化简

含有无关项的卡诺图化简示例

AB \ CD	00	01	11	10
00	0	1	×	0
01	0	×	1	0
11	×	0	×	×
10	1	×	0	×

AB \ CD	00	01	11	10
00	0	0	0	1
01	1	×	0	1
11	×	×	×	×
10	1	0	×	×

“画的圈最大，圈的个数最少”

逻辑代数基础——习题

本章习题小结

本章的习题考察形式有很多种，包括选择填空、证明题、分析题（化简变换），包括其他章节的设计题也会间接考察本章的知识点，因此非常重要；

涉及知识点：

- 逻辑代数的公式和定理；
- 逻辑函数不同表示方法的转换；（重点：从真值表得到逻辑函数式）
- 逻辑函数的最小项和最大项；
- 卡诺图化简逻辑函数；
- 特定形式逻辑函数表达式的变换；

逻辑代数基础——习题

例1：

证明：

(1) $AC + B'C + BD' + CD' + A(B + C') + A'BCD' + AB'DE = A + B'C + BD'$

(2) $(A + B + D)' + A'BC'D + A'BC + CD' + ((A' + B + C)(A' + C'))' = AB'D + BC + B'D' + A'BD$

证明题可以用到的分析方法：

①不断利用基本公式和常用公式去变换和化简；

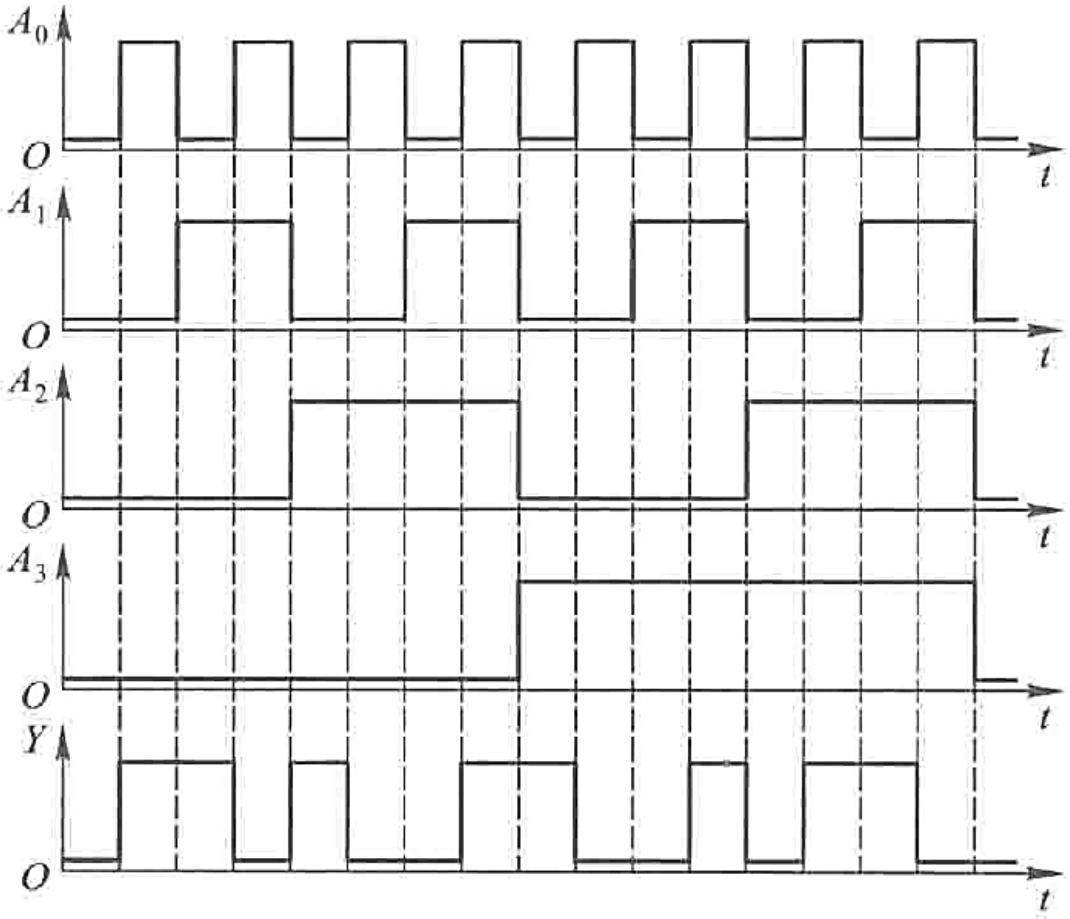
②两边取对偶式，根据对偶定理变为证明对偶式相等；

③真值表遍历，列举所有的可能的输入输出两侧比对；

④都写成最小项之和的形式（画出卡诺图但不化简），因为最小项之和是唯一的；

逻辑代数基础——习题

例2：给定逻辑函数 Y 的波形图如图所示，请写出其逻辑函数式；



分析思路：

波形图→真值表→逻辑函数式

逻辑代数基础——习题

例3：将下列逻辑函数式化成最小项之和、最大项之积的形式；

(1) $((A \odot B)(C \odot D))'$

(2) $A + B + CD$

(3) $AB' + BC' + CA'$

重点：最小项和最大项之间的关系！

写成最小项之和后如何快速得到最大项之积？—— 互补

逻辑代数基础——习题

一般不会考察超过五个变量的卡诺图化简，如果有第六个变量，考虑 $AB + A'C + BCD = AB + A'C$ 公式；

例4：用卡诺图将下列逻辑函数式化为最简与或形式；

(1) $Y = ABC + ABD + C'D' + AB'C + A'CD' + AC'D$

(2) $Y = A'B'C'D'E' + A'C'E' + BCDE' + A'BDE + A'C'D'E + AB'DE + ACD + ABCD'$

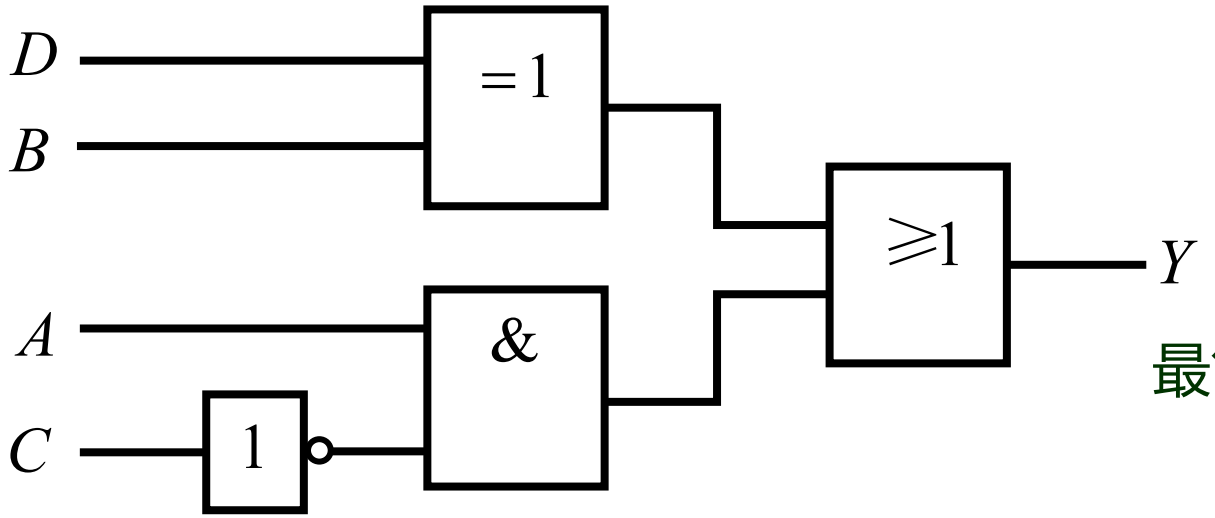
(3) $Y = A'BC + C'DE + ACDE + A'BDEF$

(4) $Y = CD'(A \oplus B) + A'BC' + A'C'D$,给定约束条件为$AB + CD = 0$

(5) $Y(A,B,C,D) = \sum m(2,3,7,8,11,14) + d(0,5,10,15)$

逻辑代数基础——习题

例5：已知逻辑图如下,请写出逻辑函数的最简与或式、最简或与式、最简与非—与非式、最简或非—或非式和最简与或非式；



分析思路：

最简与或 —— 卡诺图合并1

最简与或非 —— 卡诺图合并0

最简或与 —— 对最简与或非应用德·摩根定理

最简与非—与非 —— 对最简与或取两次反

最简或非—或非 —— 对最简与或非应用德·摩根定理

(第 ③ 个是把括号外面的非号去掉，
第 ⑤ 个是把相与的项写成它们反变量的或非，
一个是正着用一个是反着用)

逻辑代数基础——习题

补充习题：应用卡诺图化简复杂的逻辑函数式

如果给定的要求化简的逻辑函数式很复杂，并不是常见的与或形式，但是可以表示为例如 $Y = Y_1 \cdot Y_2$，$Y = Y_1 \oplus Y_2$，$Y = Y_1 \odot Y_2$ 等即两个或多个变量的逻辑运算，那么可以分别画出这些变量的卡诺图，（要求表格的规模和每一个位置的方块对应的最小项一致），再将多个卡诺图上每一个方块上的 1 和 0 按照给定的逻辑运算关系代入，就可以得到最终的目标输出变量的卡诺图；

例如：若要求化简 $Y = (ABC + CD) \oplus (AB' + A'C + BC + C'D)$，那么没有必要直接展开 Y，可以令 $Y_1 = (ABC + CD)$，$Y_2 = (AB' + A'C + BC + C'D)$，画出 Y_1 和 Y_2 的 4 变量卡诺图，将两张卡诺图对应位置上的 0 和 1 按照 $\oplus$ 运算符运算后得到的结果就是 Y 的卡诺图；
其他同类型题目同理；

逻辑代数基础——小结

- 逻辑代数的基本运算 —— 与、或、非；
- 逻辑代数的基本公式和常用公式；
- 逻辑函数的不同表示方法以及相互转换；
- 逻辑函数的最小项和最大项；
- 逻辑函数的化简与形式变换；