

可编程逻辑器件PLD

可编程逻辑器件PLD

黄 德 季晓东 编著

PLD

人民邮电出版社

TP332.1

H83

382792

可编程逻辑器件 PLE

黄 德 季晓东 编著

人民邮电出版社

登记证号(京)143号

图书在版编目(CIP)数据

15228 / 30

可编程逻辑器件 PLE/黄德,季晓东编著,一北京:人民邮电出版社,1995.6

ISBN 7-115-05615-3

I. 可… II. ①黄… ②季… III. 可编程序控制器, PLE IV. TP332.3

内 容 提 要

可编程逻辑器件 PLE 是 80 年代中期问世的新一代 PROM(可编程只读存储器)器件,与 PAL、GAL 等器件相似,具有大量的乘积项,用于实现各种逻辑功能、高速运算、复杂的状态机、保密通信与数据传输。它可取代中、小规模集成电路,扩大 PAL 的用途以及实现其他 ASIC 部件所不宜的功能,具有集成度高、延时小、可靠、灵活等优点。本书介绍 PLE 的原理与应用,并给出了一些应用实例。

本书可作为从事工业过程控制、计算机工程、通信工程、电子技术、仪表测量等专业的工程技术人员及科研工作者的参考读物。

可编程逻辑器件 PLE

黄 德、季晓东 编著

责任编辑 王亚明

*

人民邮电出版社出版发行

北京朝阳门内南竹杆胡同 111 号

北京顺义振华印刷厂印刷

新华书店总店科技发行所经销

*

开本:787×1092 1/16 1995 年 7 月 第 一 版

印张:6.5 1995 年 7 月 北京第1次印刷

字数:158 千字 印数:1-4 000 册

ISBN 7-115-05615-3/TP·176

定价:9.20 元

前 言

PLE(可编程逻辑器件)是新一代的高速 PROM(可编程只读存储器)产品。其主要特点是与阵列固定、或阵列可编程并具有大量的输入乘积项,故也适用于 PAL、GAL 所不宜的场合。因此,该产品与 PAL、GAL 器件同是风靡国际电子技术市场的新型产品。

本书介绍 PLE 器件的原理与实际应用。共分为七章:第一章介绍 PLE 器件基本知识;第二至四章介绍 PLE 器件的电路原理、品种、技术规范、软硬件开发工具,以及 PLE 的开发过程;第五章说明 PLE 器件在实现逻辑功能、编码与译码、多路开关、比较器、数码转换等方面的应用;第六章介绍 PLE 器件在高速运算方面的应用;第七章介绍 PLE 器件在实现复杂运算方面的功能。本书涉及到的几十个应用实例,所含的资料与说明均非常详尽。

PAL、GAL、PLE 以及其它 ASIC 部件的引进和应用正在不断深入。希望本书能对读者有所帮助。

在本书编写过程中,承蒙中国科技大学计算机系主任、博士生导师陈国良教授指导,刘振安副教授及周建民副教授审阅了全稿并提出许多宝贵意见,特此表示感谢。

限于编者水平,不当之处请读者指正。

作者

目 录

第一章 逻辑设计基本知识	(1)
第一节 布尔代数基本运算法则.....	(1)
第二节 布尔表达式的演算和简化.....	(2)
第三节 卡诺图.....	(5)
第四节 时序电路器件.....	(9)
第五节 状态机	(10)
第二章 PLE 概述	(12)
第一节 PLE 内部电路原理	(12)
第二节 PLE、PLA、PAL 之比较	(14)
第三节 PLE 的优点和用途	(16)
第四节 PLE 的简单应用举例	(16)
第三章 PLE 产品的型号、品种和技术规范	(20)
第一节 PLE 产品的型号意义及品种	(20)
第二节 PLE 器件系列产品的技术规范	(24)
第三节 PLE9R8 技术规范	(27)
第四节 PLE10R8、11RA8、11RS8 技术规范	(28)
第五节 PLE 器件系列电路方框图	(30)
第四章 PLE 器件的开发过程	(34)
第一节 开发硬件简介	(34)
第二节 ABEL 软件开发工具	(34)
第三节 CUPL 软件	(37)
第四节 PLE 器件编程说明及其参数	(41)
第五章 PLE 基本应用实例	(44)
第一节 基本逻辑门电路	(44)
第二节 存储器地址译码器	(45)
第三节 逻辑功能选择	(47)

第四节	可扩展 3~8 多路分配器	(48)
第五节	二路 2~1 多路开关	(51)
第六节	带极性控制的四路 2~1 多路开关	(53)
第七节	16 进制数的 7 段译码器	(55)
第八节	4 位 BCD 码和格雷码的相互转换	(57)
第九节	八位优先权编码器	(59)
第十节	4 位数值比较器	(60)
第十一节	带极性控制的 4 位数值比较器	(61)
第十二节	8 位滚动移码器	(62)
第十三节	输出极性可编程的 4 位右移码器	(65)
第六章	高速查表运算	(68)
第一节	概述	(68)
第二节	4 位查表运算乘法器	(69)
第三节	反正切查表	(70)
第四节	直角三角形斜边查表	(72)
第五节	圆周长查表	(74)
第六节	摆周期查表	(77)
第七节	算术逻辑运算单元	(79)
第七章	用 PLE 实现复杂运算功能	(81)
第一节	华莱士树形压缩算法	(81)
第二节	剩余算法	(87)
第三节	分布式算法	(94)
第四节	流水线算法	(97)
参考文献		(98)

第一章 逻辑设计基本知识

第一节 布尔代数基本运算法则

各种逻辑门电路的功能是实现把单输入或多输入信号转换为按一定法则生成的输出信号,最基本的门电路有三种:AND(与)、OR(或)、INVERTER(非),如图 1.1 所示。利用这三种基本门电路可实现所需要的任意其它逻辑功能。

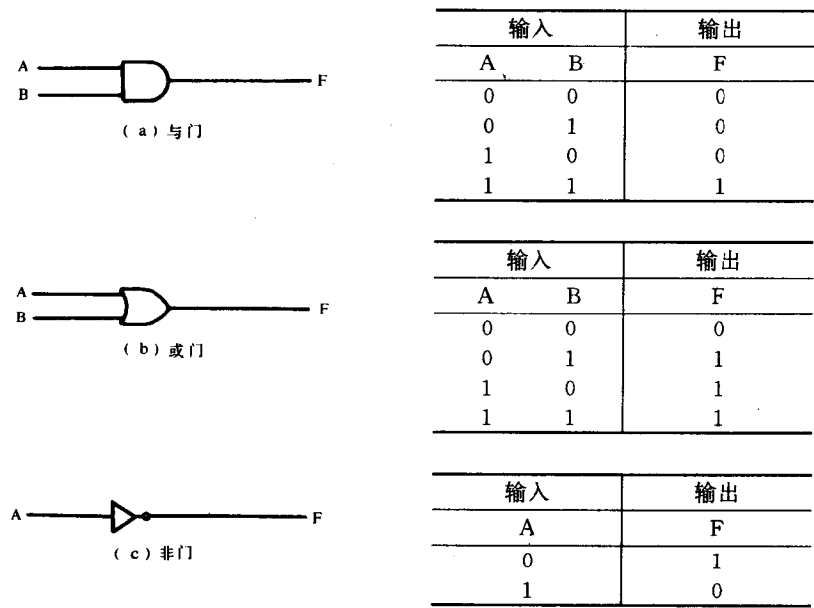


图 1.1 基本门电路

为能用布尔代数方法表达逻辑门电路的上述功能,定义以下布尔算符:

- = 逻辑等于;
- 负(非,反相,求补);
- + 或(和);
- 与(积)。

这样图 1.1 中的与门功能就可表示为: $F=A \cdot B$;或门表示为: $F=A+B$;非门表示为: $F=\overline{A}$ 。

布尔算符等价于逻辑算符,但与算术算符有别,例如:“+”是逻辑求和,“•”是逻辑求

积,上述等式为布尔等式或逻辑等式(或方程),它与算术等式(含未知数的代数方程)是有区别的。

布尔方程的运算和简化遵循以下法则:

法则 1:	$A+0=A$
法则 2:	$A \cdot 0=0$
法则 3:	$A+1=A$
法则 4:	$A \cdot 1=A$
法则 5:	$A+A=A$
法则 6:	$A \cdot A=A$
法则 7:	$A+\bar{A}=1$
法则 8:	$A \cdot \bar{A}=0$
法则 9:	$\bar{\bar{A}}=A$
法则 10:	$A+A \cdot B=A$
法则 11:	$A(A+B)=A$
法则 12:	$(A+B)(A+C)=A+B \cdot C$
法则 13:	$A+\bar{A} \cdot B=A+B$
交换律:	$A+B=B+A$ $A \cdot B=B \cdot A$
结合律:	$A+B+C=(A+B)+C=A+(B+C)$ $A \cdot B \cdot C=(A \cdot B) \cdot C=A \cdot (B \cdot C)$
分配律:	$A+(B \cdot C \cdot D)=(A+B) \cdot (A+C) \cdot (A+D)$ $A \cdot (B+C+D)=A \cdot B+A \cdot C+A \cdot D$
摩根(DeMorgan)定理:	$\overline{(A+B+C)}=\bar{A} \cdot \bar{B} \cdot \bar{C}$ $\overline{(A \cdot B \cdot C)}=\bar{A}+\bar{B}+\bar{C}$

对任何布尔表达式或部分表达式的求补运算皆可按两步进行:

1. 或算符用与算符代替,或者与算符用或算符代替。
2. 表达式中的每一项用其互补项代替。

上述摩根定理在逻辑设计中是很有用的,因为它能十分简捷地完成“积之和”与“和之积”两种表达式之间的转换。

第二节 布尔表达式的演算和简化

任何复杂的逻辑表达式都可简化为“二级”形式,即可表达为“积之和”(SOP)或“和之积”(POS)。为此,我们先来定义“SOP”、“POS”以及“项”的概念:

1. 乘积项——乘积项是一个简单的变量或多个变量的逻辑求积。所说的变量也可求补变量。
2. 和项——和项是一个简单的变量或多个变量的逻辑求和。所说的变量也可求补变

量。

3. 标准项——标准项是一个乘积项或和项,其中每一个变量不能使用多次。

4. 最小项——最小项是一个乘积项,它包含每一个变量一次且只有一次(可以是真值或求补)。

5. 最大项——最大项是一个和项,它包含每一个变量一次且只有一次(可以是真值或求补)。

6. 积之和表达式——它是一个乘积项或多个乘积项的逻辑求和。

7. 和之积表达式——它是一个和项或多个和项的逻辑求积。

例如: $A \cdot B \cdot C$ 是一个乘积项; $A+B+C$ 是一个和项; A 既是一个乘积项,也是一个和项; $A+B \cdot C$ 既不是乘积项,也不是和项; $A+\bar{B}$ 是一个和项; $A \cdot \bar{B} \cdot \bar{C}$ 是一个乘积项; $\bar{B}$ 既是一个和项,也是一个乘积项。

$\bar{A} \cdot B + A \cdot \bar{B}$ 是一个 SOP 表达式; $(A+B) \cdot (\bar{A}+\bar{B})$ 是一个 POS 表达式。

SOP 或 POS 表达式能直接转换为非常简单的门电路,最简单的电路可分为两级,信号从输入端到输出端的最长路径是两个逻辑门电路。

在设计逻辑电路时,设计者可把已知变量分为输入、输出两组。改变输入变量的状态,看看每个输入状态下输出信号如何变化,从而推演出逻辑表达式。可按以下步骤:

1. 构造输入、输出值的真值表,如表 1.1 所示。

表 1.1 真 值 表

输 入			输 出	乘 积 项	和 项
A	B	C	F		
0	0	0	0	$\bar{A} \bar{B} \bar{C}$	$A+B+C$
0	0	1	0	$\bar{A} \bar{B} C$	$A+B+\bar{C}$
0	1	0	1	$\bar{A} B \bar{C}$	$A+\bar{B}+C$
0	1	1	1	$\bar{A} B C$	$A+\bar{B}+\bar{C}$
1	0	0	0	$A \bar{B} \bar{C}$	$\bar{A}+B+C$
1	0	1	0	$A \bar{B} C$	$\bar{A}+B+\bar{C}$
1	1	0	1	$A B \bar{C}$	$\bar{A}+\bar{B}+C$
1	1	1	0	$A B C$	$\bar{A}+\bar{B}+\bar{C}$

2. 推出 SOP 表达式。

在“乘积项”一栏中,按输入信号 A、B、C 的值列表,然后将每个输出信号为 1 的那些行的乘积项相加,则为 SOP。即:

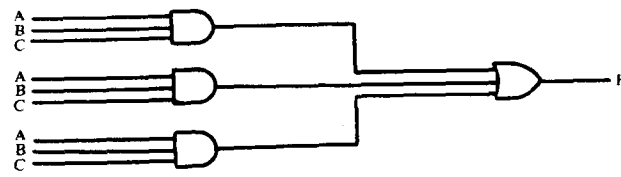
$$F = \bar{A} \cdot B \cdot \bar{C} + \bar{A} \cdot B \cdot C + A \cdot B \cdot \bar{C}$$

若推导 POS 表达式,则按下述方法:

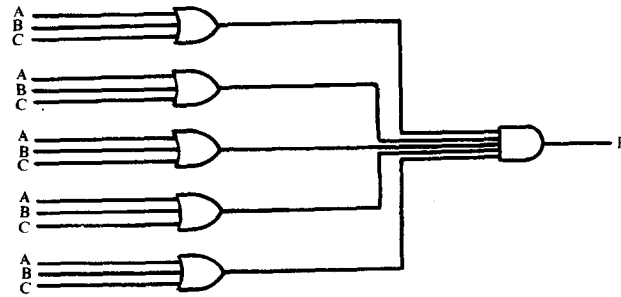
在“和项”一栏中按输入信号 A、B、C 的值列表,然后将每个输出信号为“0”的行求积,则为 POS。即:

$$F = (A+B+C) \cdot (A+B+\bar{C}) \cdot (\bar{A}+B+C) \cdot (\bar{A}+B+\bar{C}) \cdot (\bar{A}+\bar{B}+\bar{C})$$

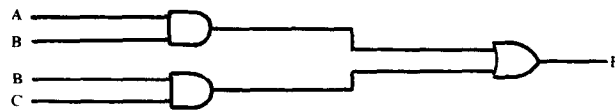
3. 将上述 SOP 表达式转换为电路图 1.2(a),或将 POS 表达式转换为电路图 1.2(b)。



(a)



(b)



(c)

图 1.2 逻辑实现电路

简化 SOP 方程:

$$\begin{aligned}
 F &= \bar{A} \cdot B \cdot \bar{C} + \bar{A} \cdot B \cdot C + A \cdot B \cdot \bar{C} \\
 &= \bar{A} \cdot B \cdot (\bar{C} + C) + A \cdot B \cdot \bar{C} \\
 &= \bar{A} \cdot B + A \cdot B \cdot \bar{C} \\
 &= B \cdot (\bar{A} + A \cdot \bar{C}) \\
 &= B \cdot (\bar{A} + \bar{C}) \\
 &= \bar{A} \cdot B + B \cdot \bar{C}
 \end{aligned}$$

或简化 POS 方程:

$$\begin{aligned}
 F &= (A+B+C) \cdot (A+B+\bar{C}) \cdot (\bar{A}+B+C) \cdot (\bar{A}+B+\bar{C}) \cdot (\bar{A}+\bar{B}+\bar{C}) \\
 &= (A+B) \cdot (\bar{A}+B) \cdot (\bar{A}+\bar{C}) \\
 &= B \cdot (\bar{A}+\bar{C}) \\
 &= \bar{A} \cdot B + B \cdot \bar{C}
 \end{aligned}$$

简化后的逻辑方程可由图 1.2(c)所示电路来实现,这一结果表达式称为最简表达式,简化过程称为最简化过程,最简表达式的电路实现是最经济、最可靠的。

任何逻辑电路可用逻辑表达式来表达,称为逻辑方程。把逻辑方程简化就得到简化电路的目的。例如,电路图 1.3(a)可简化为 1.3(b)。简化过程如下:

$$\begin{aligned}
 F &= (A \cdot B \cdot C) \cdot (B + D) + A \cdot \bar{C} \cdot (B + D) \\
 &= A \cdot B \cdot C + B \cdot D + A \cdot B \cdot C \cdot D + A \cdot \bar{C} \cdot B + A \cdot \bar{C} \cdot D \\
 &= A \cdot B \cdot C(1 + D) + D(B + 1) + A \cdot \bar{C} \cdot B + A \cdot \bar{C} \cdot D \\
 &= A \cdot B \cdot C + D + A \cdot \bar{C} \cdot B + A \cdot \bar{C} \cdot D \\
 &= A \cdot B \cdot (C + \bar{C}) + D(1 + A \cdot \bar{C}) \\
 &= A \cdot B + D
 \end{aligned}$$

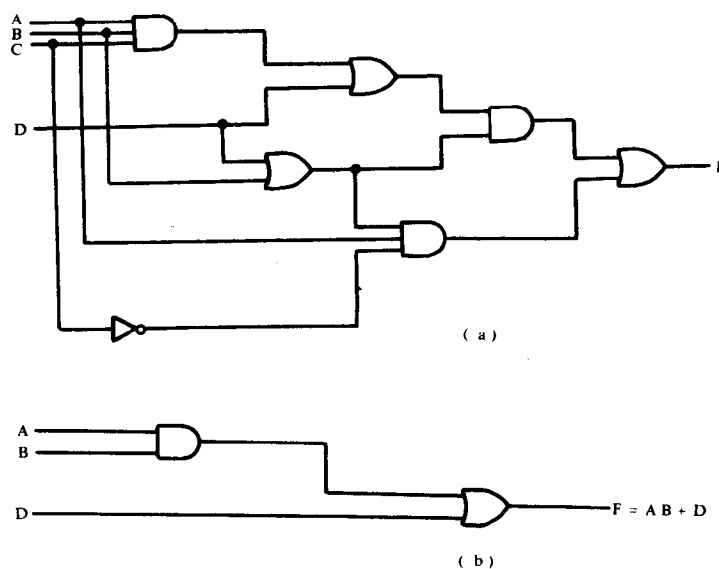


图 1.3 逻辑电路的简化

第三节 卡 诺 图

卡诺图是表达逻辑功能的图解法,图解法比真值表简单,卡诺图的排列方法有许多种,其基本特点是:

1. 在卡诺图中,相应于输入变量的每一种组合的功能,其真(或假)值占唯一位置。
2. 在采用前述简化方法时,位置的排列要易读易懂。

对于逻辑设计者来说,卡诺图是十分有效的工具,也是手工简化逻辑的常用方法。下面举例说明卡诺图的建立方法。

假设一个两变量的真值表,输入信号的组合最多可能为四种,相应的与、或功能如表 1.2 所示。

表 1.2a

两变量的与真值表

A	B	$A \cdot B$
0	0	0
0	1	0
1	1	1
1	0	0

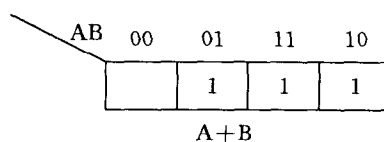
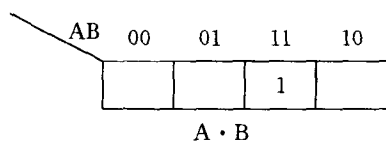
表 1.2b

两变量的或真值表

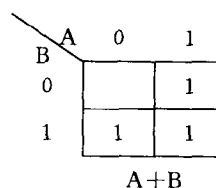
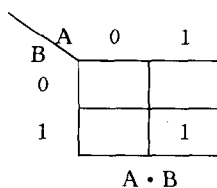
A	B	$A + B$
0	0	0
0	1	1
1	1	1
1	0	1

现在换一种方法,画一个由四个小方块组成的一块图,每一个小方块代表一种逻辑组合,在相应组合逻辑功能为1的小方块里填写“1”。其它小方块则应填写“0”,表示相应于这些组合的逻辑功能为0,通常可将“0”省略不填。

以上所述如图 1.4(a)所示,它是一种合法的卡诺图形式,通常排列为方阵,如图 1.4(b)所示。



(a)

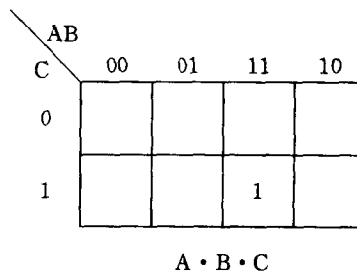


(b)

图 1.4 两变量卡诺图

几个变量的卡诺图为 2^n 块,不管变量为多少,总可根据其真值表的图表形式(图 1.5(a))表达为唯一的区域交错的图(如图 1.5(b))所示。

A	B	C	$A \cdot B \cdot C$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



(a)

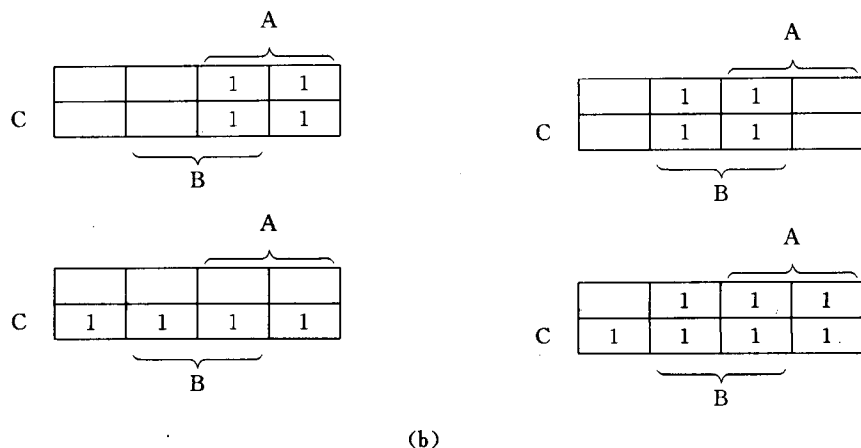


图 1.5 三变量卡诺图

图 1.6(a), (b), (c), (d) 中分别表示另外一些逻辑函数的卡诺图。

要注意的是, 图 1.5(a) 和图 1.6(b) 中的逻辑表达式是最小项, 八个方块中的每一个相应于三变量八个最小项中的每一项。所谓最小项, 它是布尔函数的一种形式, 在卡诺图中相应于不为 0 的最小区域; 类似地, 最大项在卡诺图中相应于不为 1 的最大区域, 图 1.5(b)、1.6(c) 是最大项。

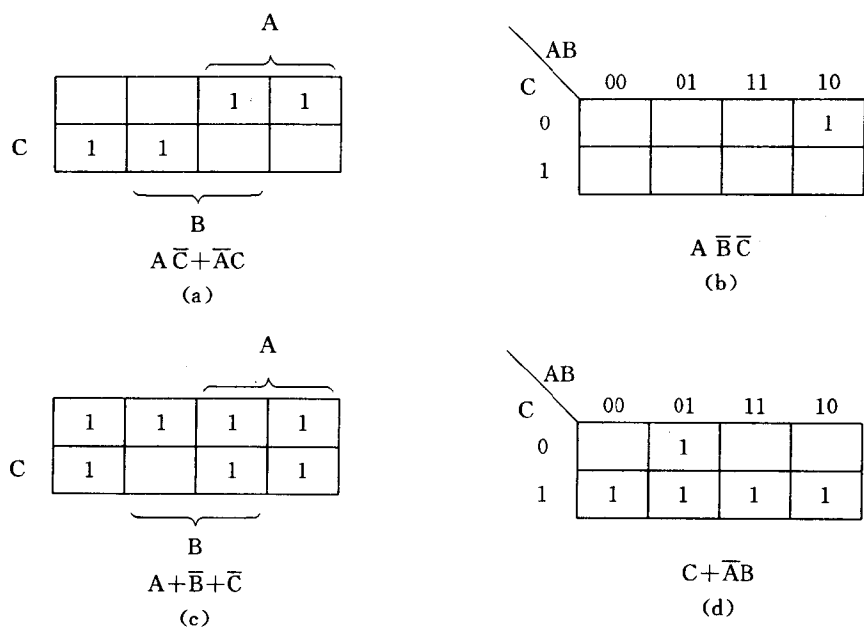


图 1.6 三变量卡诺图举例

由于卡诺图的每一方块相应于真值表的一行, 所以方块数等于真值表的行数。两变量和三变量的标准卡诺图如图 1.7 所示。若逻辑函数表示为最小项的列表形式, 则在相应的方块中填写“1”就构造了卡诺图。

	A	0	1
B	0	0	2
	1	1	3

	AB	00	01	11	10
C	0	0	2	6	4
	1	1	3	7	5

图 1.7 两变量和三变量卡诺图

若逻辑函数表示为最大项的列表形式,则在相应的方块中填写“0”,或在其它方块中填写“1”。

例如 $f(A,B,C)=m(0,2,3,7)$ 的卡诺图如图 1.8(a) 所示, $f(A,B,C)=M(0,1,5,6)$ 的卡诺图如图 1.8(b) 所示,其中 m 为最小项, M 为最大项。

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

(a)

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

(b)

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

图 1.8 卡诺图举例

前面已说明根据真值表可按最小项表达式或者最大项表达式建立真值表。以下说明如何用卡诺图来简化最小项表达式。

简化方法的基本原则是:卡诺图中邻近的 n 个变量的一对最小项可结合为一个乘积项,从而 n 个变量减少到 $n-1$ 个变量。这里所说的“邻近”,也包括每行中两端的项。

例如,对图 1.9 所示的逻辑函数:

$$f(A,B,C)=m(0,1,4,6)$$

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

可简化为:

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

图 1.10 为一个多变量的卡诺图,每行、每列两端的项也是邻近项,可把几个最小项结合在一起,以加快简化步骤。

	AB	00	01	11	10
C	0	1		1	1
	1	1		3	7

图 1.9 逻辑函数图

在逻辑设计时,经常遇到某些输入变量的组合形式在表达式中不存在。对这种情况,在卡诺图中用“无关”符号“X”表示。它可以为 1,也可以为 0,这取决于是否能将逻辑函数简化得更好些。图 1.11 表示对卡诺图中“无关”情况的处理方法。

对于更多变量的卡诺图方法,可参阅有关开关理论与代数设计方法的图书。

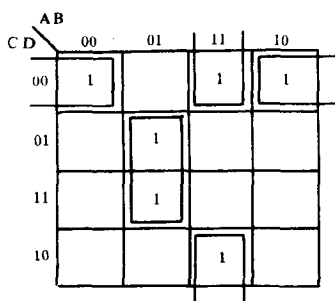


图 1.10 简化举例

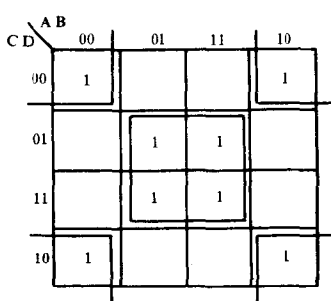
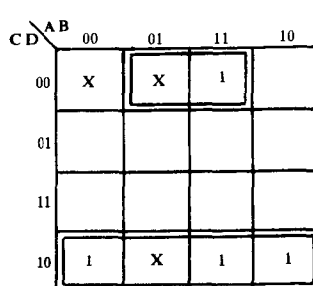


图 1.11 “无关”情况的处理方法



第四节 时序电路器件

逻辑设计分为时序逻辑设计与简单组合逻辑设计两大类,这两种设计方法的结合为复合逻辑设计。简单组合逻辑系统中不使用存储器,其输出由现行输入信号确定。因此,分析和解决的问题比较容易些。而时序逻辑系统则包含有存储器,它的输出是现行输入信号及先前输出信号的函数。多路开关、编码/译码器、加法器、比较器等电路都是简单的组合逻辑电路,而移位器、计数器、状态机、存储器、控制器等电路则属于时序逻辑电路。

逻辑门电路是最基本的组合逻辑电路单元,而触发器则是最基本的时序逻辑电路单元。触发器是一种存储器件,可以存储、记忆二进制编码信息。触发器可分为四类基本触发器:即 D 触发器、T 触发器、R-S 触发器、J-K 触发器,如图 1.12 所示。

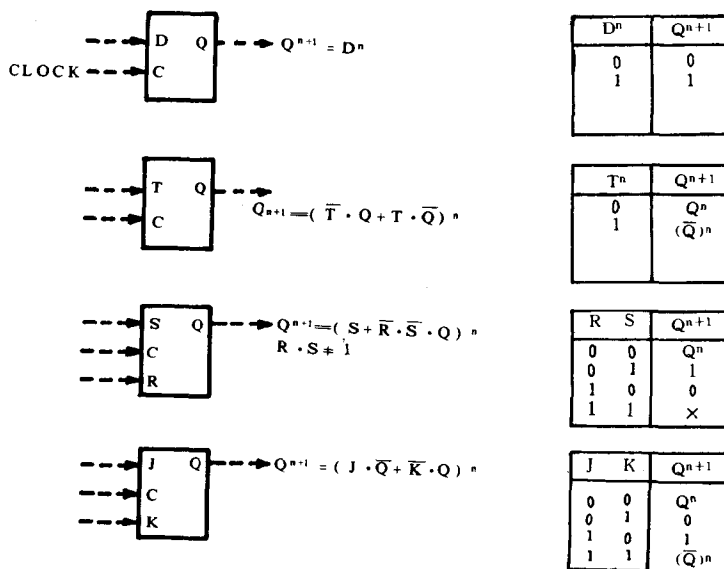


图 1.12 四种基本触发器

这些存储器件的输出信号在时钟信号翻转之前的一段时间内不会发生变化。因此,应注

意在 $n+1$ 时钟周期内,输出信号是在 n 时钟周期内输入信号的函数。

D 触发器的输入信号 D 在该触发器中“寄存”,一当时钟到来,这一信号则在下一个 $(n+1)$ 时钟周期内在输出端(Q)建立,它类似于一位 RAM。D 触发器在数据存储和其它方面非常有用。

其它三类触发器也是一位存储器。不过,不是简单地寄存输入信号,而是按照不同的逻辑关系,相应于输入信号来改变自身的状态,在时钟到来前能保持先前状态。例如:T 触发器在时钟到来前,若输入 T 为假,则保持先前状态;在时钟到来时则改变到相反状态。用它作二进制计数时,每当计数的低位数据有进位时,高位改变一次状态。

R-S 触发器在输入端 S 为真时置位,而在 R 输入端为真时则复位。

J-K 触发器在 J 端为真时置位,而在 K 端为真时复位。这类似于 R-S 触发器;不同的是,在 J、K 两端同时为真时,输出改变到相反的状态。若把 J、K 连接起来,则它可用作 T 触发器。若把 J-K 触发器的 K 端连接一个反相器,如图 1.13 所示,则它可代替 D 触发器。

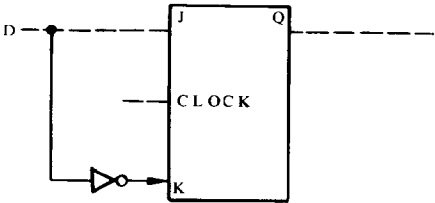


图 1.13 用 J-K 触发器实现 D 触发器

锁存器也是一种时序器件,它在同步逻辑电路中不能使用。

第五节 状态机

现行状态变量、先前输入变量、下一状态变量、先前输出变量之间的关系,可采用多种方法定义。图 1.14 是一个简单的时序系统。

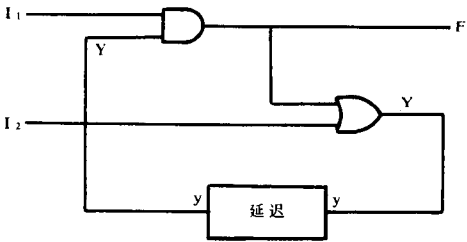


图 1.14 时序系统

这一系统有两个先前输入信号(即四种组合值)一个先前输出信号,一个状态变量。系统使用一个延时电路作为存储器,现行状态可能为两个不同的值: $Y=0$ 或 $Y=1$,再与四种输入组合信号组合,定义八种现行状态。必须按每一现行状态定义下一状态变量 Y 和先前输出变量 F。表 1.3 描述了上述全部信息,称之为状态表。

表 1.5

状 态 表

现 行 状 态	下 一 状 态 Y				输 出 F					
Y	I ₁ I ₂ =	00	01	10	11	I ₁ I ₂ =	00	01	10	11
0		0	1	0	1		0	0	0	0
1		0	1	1	1		0	0	1	1

描述时序系统功能的第二种方法是采用状态图,它是现行状态与下一状态的图形表示方法,常用来表示时序器件。图中箭头表示状态变化,箭头旁注明先前输入、输出条件。图1.14的状态图如图 1.15 所示,图中用斜杠把输入、输出值隔开。

状态表和状态图是分析、设计时序系统和数字系统的基本工具。

由于时序系统中存在着输出到输入的反馈信号,因此可能发生某种不稳定性或不确定性。这有可能使电路运行发生故障,甚至不能运行。这些问题可归结为以下几类情形:

1. 系统的输入或输出状态不确定。
2. 系统输出状态不稳定,甚至在外来输入信号不变时也是如此。

3. 系统输出状态虽然稳定,但无法按照先前输入状态来确定系统的下一状态。

为防止上述问题发生,在设计和使用时序系统时应做某些限制,例如增加如下限制条件:

1. 不稳定时禁止继续使用;
2. 按照基本模式运行;
3. 按照脉冲模式运行。

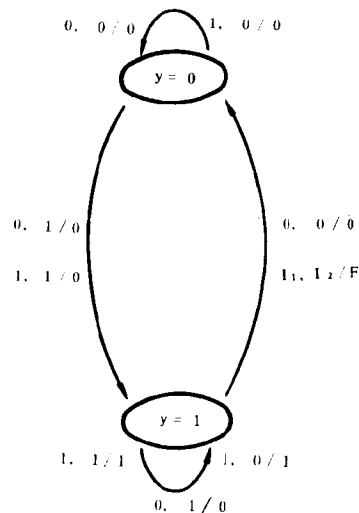


图 1.15 状态图