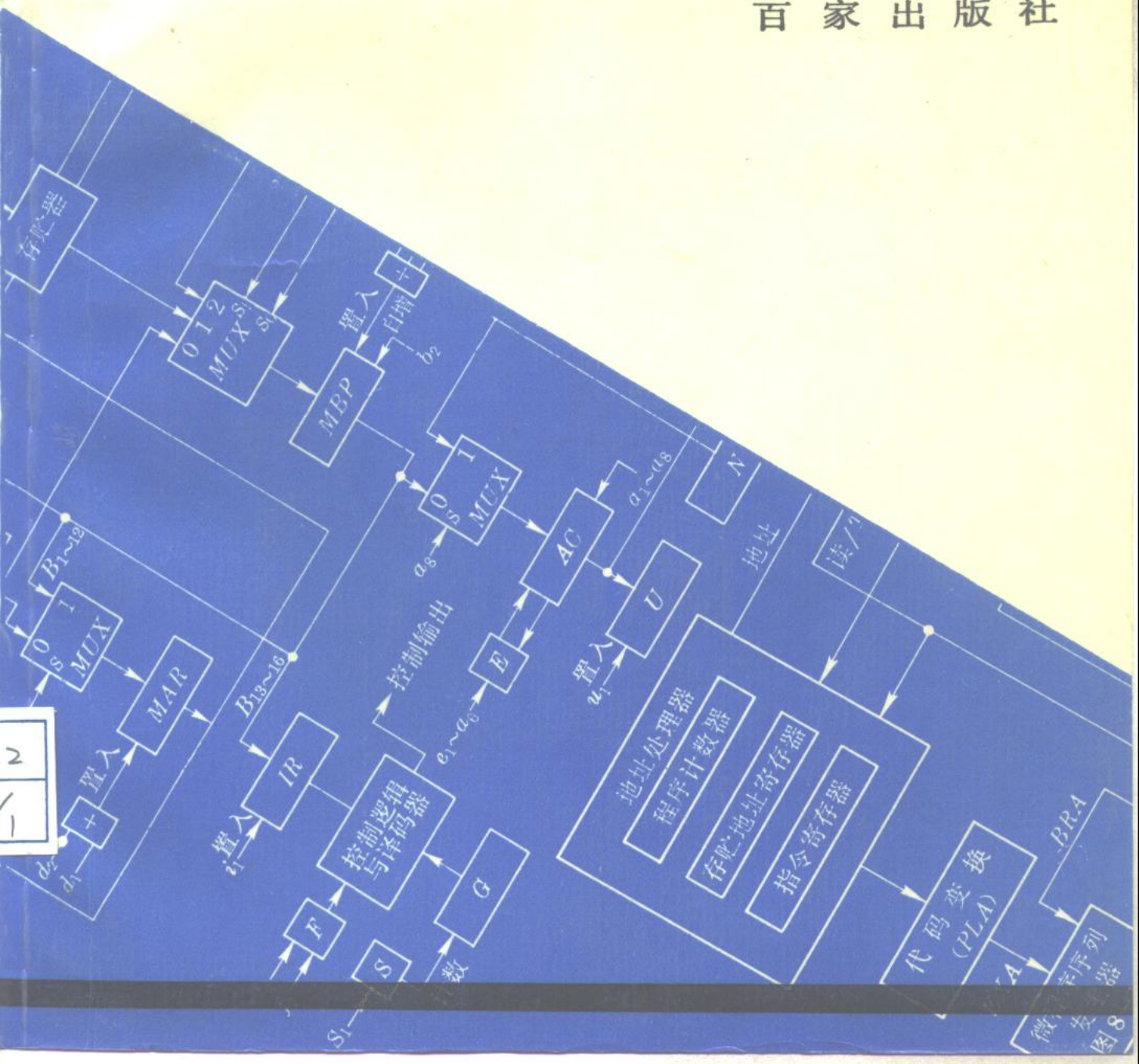
JISUANJILUOJISHEJIJICHU

计算机逻辑设计基础

张礼平 编著

百家出版社



计算机逻辑设计基础

张礼平　编著

中国纺织大学编辑

百家出版社

内 容 提 要

本书是电子工程和计算机专业的基础教材。它以小型数字计算机为背景，全面介绍了经典逻辑设计和现代逻辑设计的基础知识。全书共分九章，即：组合逻辑电路的分析和设计；同步时序电路的分析；同步时序电路的设计；异步时序电路的分析和设计；中、大规模集成电路与逻辑设计；寄存器传送逻辑；处理器逻辑设计；控制器逻辑设计；计算机的设计。本书可用作大专院校计算机工程专业、计算机软件专业的教材，也可供从事计算机应用的科学工作者和工程技术人员阅读、参考。

计算机逻辑设计基础

张礼平 编著

中国纺织大学图书出版编辑部编辑

百家出版社出版

商务印书馆上海印刷厂印刷

新华书店上海发行所发行　　各地新华书店经售

开本 787×1092 1/16　印张 14　字数 339,000

1989 年 9 月第 1 版　1989 年 9 月第 1 次印刷

印数 1—3,500 本

书号：ISBN 7-80576-045-4/TP·02　　定价：4.75 元

前　言

随着计算机科学和技术的发展，计算机逻辑设计出现了硬件和软件两种逻辑设计方法．软件逻辑设计方法本质上是硬件支持下的程序设计．近年来，软件逻辑设计方法的研究和实践已受到了越来越广泛的重视，取得了一定的成果．硬件逻辑设计方法是以集成电路为基础的逻辑设计．硬件逻辑的操作速度快，灵活，成本低，是软件逻辑无法相比的．所以，硬件逻辑设计方法是计算机逻辑设计的最重要的手段；熟练地掌握硬件逻辑设计的理论和技巧，对于计算机硬件设计和维护，以及从事计算机应用的人员来说，是至关重要的．

1980年以来，我国各大专院校普遍开设了“数字逻辑”课程．作者曾编写《计算机逻辑设计》教材，并多次对各种不同层次的学生讲授．本书是在该教材的基础上进一步修改、提炼而成的，以奉献给大专院校师生，也希望受到从事计算机应用的科学工作者、工程技术人员的欢迎．

本书可作为计算机工程专业的“数字逻辑和计算机设计”课程的教材，也可作为计算机软件专业“计算机结构”课程的教材。全书共九章．

第一章以布尔代数的图解法和列表法为工具，讲解组合逻辑电路的设计和分析方法，介绍了一些计算机中的基本部件，如加法器、代码变换器等．

第二章以状态表和状态图为基础，讲解同步时序逻辑电路的分析方法，并列举大量的分析实例．

第三章论述同步时序逻辑电路的设计方法，对复杂的不完全给定同步时序逻辑电路的设计，也作了适当的介绍．

第四章介绍了脉冲异步时序逻辑电路和电平异步时序逻辑电路的设计方法．

第五章介绍常用的MSI和LSI功能部件，如并行加法器、译码器、多路器、只读存贮器、可编程逻辑阵列等．讨论如何以MSI和LSI功能部件为基础，进行逻辑设计．

第六章详细地介绍描述数字系统的“寄存器传送逻辑”方法．通过一台简单的计算机的设计，讲解如何用“寄存器传送逻辑”方法进行逻辑设计．

第七章讲解计算机处理器单元，讨论了组成处理器单元的总线和高速暂存器两种方案．进一步深入地讨论算术逻辑单元(ALU)的设计，并且延伸论述了设计各种ALU的方法．

第八章讲解控制器逻辑设计的四种方法．每态一位法和序列寄存-译码法构成“硬连接”控制器．微程序控制法和可编逻辑阵列法中引入微程序设计思想，可构成固件和PLA控制器．

第九章讨论小型数字计算机的设计．讨论中以寄存器传送逻辑方法描述计算机指令系统、体系结构和运行方法；确定计算机功能部件的原则和方法．

本书在内容叙述上力求深入浅出，通俗易懂，使初学者便于自学．在讨论设计方法时，力求物理概念清楚，设计步骤明确，使读者能够自行设计、组织所需的功能部件和计算机系统．各章后附有练习题，以求加深对课程内容的理解．

最后，借本书出版之际，我要感谢邹海明教授，因为他对本书的编写方向和内容组织提出了宝贵的指导性意见．毛法尧和肖道举对本书的初稿提出过大量的改进意见．本书出版过程中，中国纺织大学给予了大力支持，我想在此一并表示对他们的谢意．本书的第五章和第七章由孙志仁最后修改和审订，第六章由归瑶琼编写．

由于作者业务水平和实践经验有限，书中的缺点和错误在所难免，恳请读者予以指正．

编　者

1988年5月

目　录

第一章　组合逻辑电路的分析和设计

1.1　概　　述

数字系统中的逻辑电路可分成两类：一类叫做组合逻辑电路，另一类叫做时序逻辑电路.

组合逻辑电路是由逻辑门组成的，任一瞬时的输出仅仅由该瞬时的输入变量组合确定，而与以前的输入状况无关. 一个具体的组合逻辑电路所实现的某种特定的逻辑功能，在逻辑上完全可以用一组布尔函数加以充分描述.

时序逻辑电路是由逻辑门和记忆元件（如触发器、延迟元件等）组成的，电路的输出不仅取决于现在的各输入状态，而且还取决于过去的各输入状态.

为了研究组合逻辑电路，图 1-1 给出了组合逻辑电路的一般方框图.

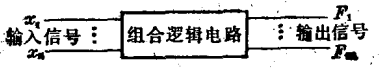


图　1-1

组合逻辑电路的输入信号记作 $x_1, \cdots, x_n$，输出信号记作 $F_1, \cdots, F_m$. 如果用一般的表达方式来描述输入、输出之间的关系，输出信号就可以用输入信号的逻辑函数来表示：

$$F_i = f_i(x_1, \cdots, x_n), \quad \text{其中 } i = 1, \cdots, m.$$

n 个二进制输入变量共有 2^n 种可能的组合，m 个输出变量可以用 m 个逻辑函数来描述.

本章在二进制、布尔代数等基本知识的基础上，解决以下三个问题：

1. 组合逻辑电路的分析：对于给定的组合逻辑电路，寻求一个逻辑函数，来描述它的工作状况和分析它的逻辑功能.

2. 组合逻辑电路的设计：对于给定的逻辑要求，或者给定描述某一逻辑功能的逻辑函数，决定用何种元件和何种结构来实现.

3. 组合逻辑电路竞争和险象产生的原因，如何发现？如何消除？

1.2　组合逻辑电路的分析方法

我们研究、讨论某一给定的逻辑电路图时，经常会遇到这样一类问题：有时需要推敲某个逻辑电路的设计思想；或者要更换逻辑电路的某些子电路；或者要评价逻辑电路的经济指标是否合理，等等. 这样就要求我们对已知的逻辑电路进行分析，以求解决上述问题. 所谓分析，就是要找出组合逻辑电路输入和输出之间的关系. 找出在什么输入组合情况下输出为“1”状态，什么输入组合情况下输出为“0”状态.

组合逻辑电路（以下简称组合电路）的分析方法，一般可分为以下四个步骤（图 1-2 给

出了组合电路分析的方框图)：

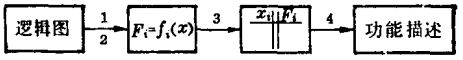


图 1-2

第一步．根据已给出的组合逻辑电路图，先根据电路由输入逐级判断，写出其逻辑函数表达式．

第二步．对得到的逻辑函数表达式进行化简，化简的一般方法有布尔(G. Boole)代数法、卡诺(Karnaugh)法、奎恩-麦克拉斯基(Quine-Mccluskey)法．卡诺法是一种图解法，适合于手算．奎恩-麦克拉斯基法是一种列表法，适合机器运算．

第三步．由简化的逻辑函数表达式，可以直接列出真值表．

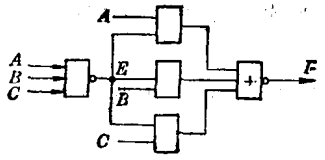


图 1-3

第四步．由真值表，判断该逻辑电路所能完成的逻辑功能．

【例 1】 试分析图 1-3 所示的组合电路的逻辑功能．

第一步．写出该电路的逻辑函数表达式．该组合电路是三级门电路．第一级是一个与非门，显然，E 点的输出为 $E=\overline{ABC}$．进一步，即可直接写出电路的逻辑函数表达式

$$F=\overline{A\,\overline{ABC}}+\overline{B\,\overline{ABC}}+\overline{C\,\overline{ABC}}.$$

第二步．对上述逻辑函数表达式进行化简

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

第三步．根据简化的逻辑函数，列出表 1-1 所示的真值表．

表 1-1

A	B	C	F
0	0	0	1
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

第四步．根据真值表，进行该组合电路的功能分析．

由真值表可见，只有在 m_0、m_7 两种情况下，输出 F 才为 1．也就是说，只有 A、B、C 三个输入一致时，输出为 1；否则，输出为 0．所以，这个组合电路具有检查“输入不一致”的逻辑功能，称之为“不一致电路”．

在一些运行可靠性要求非常高的系统中，往往采用三套设备同时工作．只要其中有

不正常工作时，“不一致电路”便可发出信号，告诉操作人员进行维护，以保证系统可靠地运行.

在讨论该电路经济合理性时，发现该电路在理论上不是最佳的. 假若对上述的逻辑函数进行逻辑变换：

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

根据上述逻辑函数表达式可画出“不一致电路”的另一形式. 如图 1-4 所示.

比较图 1-3 和图 1-4，我们发现它们的逻辑功能相同. 但是，图 1-4 的电路形式简单，清晰.

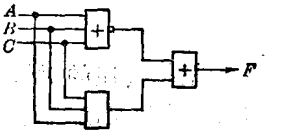


图 1-4

根据集合论和布尔代数可知：“与”、“或”、“非”三个基本逻辑是一个“完全集”，而且“与非”或者“或非”本身就是一个“完全集”. 因此，一切组合电路可以用“与”门，“或”门和“非”门组成；也可以用“与非”门或者“或非”门组成. 一般说来，完全用“与非”门或者完全用“或非”门组成组合电路比较经济、方便. 一般以“与-或”表达式描述的逻辑函数，用“与非”门构成比较方便.

【例 2】 $F=AB+CD+HG$.

这是一个以“与-或”表达式描述的逻辑函数，由摩根定理，可以将上述逻辑函数表达式变换为：

$$F_1=AB+CD+HG=\overline{\overline{AB}\,\overline{CD}\,\overline{HG}}.$$

于是，此逻辑表达式便可方便地用“与非”门构成，如图 1-5 所示.

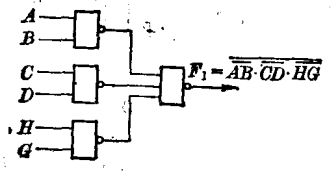


图 1-5

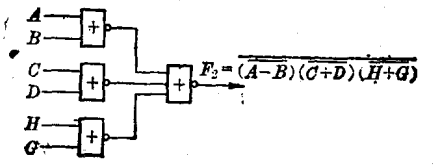


图 1-6

【例 3】 $F_2=(A+B)(C+D)(H+G)$.

这是一个以“或-与”表达式描述的逻辑函数，由摩根定理，可以将上述逻辑函数表达式变换为：

$$F_2=(A+B)(C+D)(H+G)=\overline{\overline{(A+B)}+\overline{(C+D)}+\overline{(H+G)}}.$$

于是，此逻辑函数表达式便可方便地用“或非”门来构成，如图 1-6 所示.

从例 2 和例 3 可以看出：逻辑函数 F_1 和逻辑函数 F_2 间有“对偶”关系. 图 1-5 和图 1-6 所表示的逻辑电路结构完全相同，只是一个用“与非”门，另一个用“或非”门.

逻辑电路的分析和设计中，常常要把“与-或”表达式描述的逻辑函数用“与非”门实现. 用逻辑函数的对偶法则就可以很方便地实现这一点.

“与-或”表达式描述的逻辑函数要用“或非”门实现，可以先作出其对偶函数的“与非”门结构的逻辑电路，然后用“或非”门代替电路中的“与非”门即可.

【例 4】 求“不一致电路”函数的“或非”门结构的逻辑电路.

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

$$F'=(A+B+C)(\overline{A}+\overline{B}+\overline{C})=\overline{A}B+\overline{A}C+A\overline{B}+\overline{B}C+A\overline{C}+B\overline{C}.$$

$$=\bar{A}B+\bar{B}C+A\bar{C}.$$

于是，F' 可以很方便地用“与非”门构成，如图 1-7 所示.

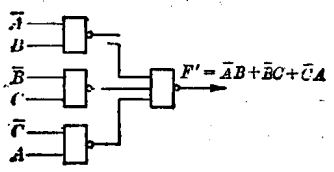


图 1-7

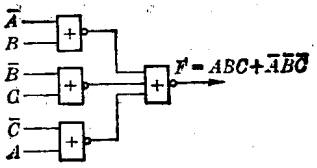


图 1-8

根据前述的对偶法则，只要将图 1-7 中的“与非”门全部用“或非”门来代替，我们就得到了 F' 的对偶函数 F 的“或非”门结构形式. 如图 1-8 所示.

同样，以“或-与”表达式描述的逻辑函数若要用“与非”门实现，可先求出其对偶函数的“或非”门结构的逻辑电路，然后再用“与非”门代替电路中的“或非”门即可.

我们知道，所谓简化，就是寻找逻辑函数具有最少的项数和最少字母的形式. 但在实际应用中，最简化形式的逻辑函数不一定能得到最简单的逻辑电路. 而逻辑函数的某个较复杂的形式却能得到最简单的逻辑电路. 这种较为复杂的逻辑函数形式，称为“最佳式”.

【例 5】 函数 $F_1=A\bar{B}+\bar{A}B$ 已是最简化的形式. 用“与非”门构成的逻辑电路如图 1-9 所示.

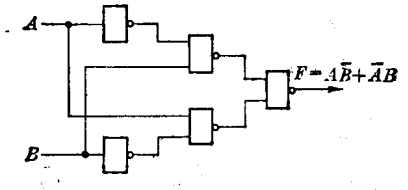


图 1-9

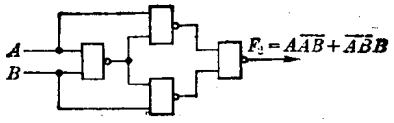


图 1-10

而函数 $F_2=A\,\overline{A\bar{B}}+B\,\overline{\bar{A}B}$ 是函数 F_1 的较复杂形式，但其用“与非”门构成的逻辑电路却比 F_1 用“与非”门实现的逻辑电路简单. 如图 1-10 所示.

由图 1-9 和图 1-10 可以看出：函数的最简式不一定是最佳式. 但在许多情况下，最简式就是最佳式.

1.3　组合逻辑电路的设计方法

组合电路的设计，是分析的逆过程. 设计也可以分成四个基本步骤(图 1-11 给出了组合电路设计过程的方框图)：

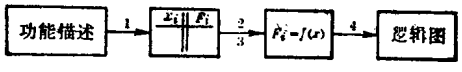


图 1-11

第一步. 根据设计的逻辑要求，分析有几个逻辑输入？有几个逻辑输出？然后列出真值表.

第二步. 根据真值表，写出逻辑函数的“与-或”式.

第三步. 根据使用的门电路的类型及实际问题的要求，用任何一种简化方法将函数

简化成所需的表达式.

第四步. 根据所得的函数表达式,画出逻辑电路图.

当然,根据问题的难易及设计者的熟练程度,可以跳过第一步、第二步甚至第三步.第三步中,实际问题是比较复杂的,这要依靠设计者根据具体问题灵活掌握.

必须指出:经上述四步得到的逻辑电路,不一定是最佳的. 实际上,电路的逻辑设计将是多次进行的. 根据实际工作的要求,根据所选用的元件,本着性能可靠、经济、简单等原则,反复比较,最后确定最佳方案. 一般说来,最佳式是个理论问题,最佳设计是个实际工程问题.

【例 6】 用"与非"门设计一个三变量的多数表决电路.

第一步. 关于逻辑功能的分析:这个多数表决电路具有三个输入,一个输出. 当三个输入全为"1",或者两个输入为"1"时,输出为"1". 列出真值表(如表 1-2 所示),设输入变量为 A、B、C,输出为 F.

表 1-2

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

第二步. 由真值表(表 1-2 所示)即可写出逻辑函数 F 的表达式

$$F=\sum m(3,\ 5,\ 6,\ 7).$$

第三步. 化简

经卡诺图(图 1-12 所示)将逻辑函数 F 简化为

$$F=AB+AC+BC.$$

进一步,将上式写成"与非-与非"表达式:

$$F=\overline{\overline{AB}\ \overline{AC}\ \overline{BC}}.$$

第四步. 画出逻辑电路图,如图 1-13 所示.

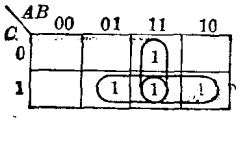


图 1-12

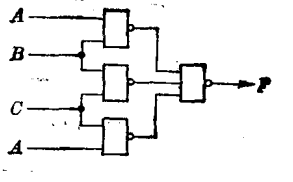


图 1-13

【例 7】 设 $x=x_1x_2$, $y=y_1y_2$ 是两个二进制正整数. 自选门电路, 设计"$x>y$"时 F

的值为"1"，否则为"0"的逻辑电路.

第一步. 关于逻辑功能的分析：我们把 x_1、x_2、y_1、y_2 看作是四个逻辑变量. 这样，电路有四个输入和一个输出.

下面来分析 F 和输入之间的关系. 我们知道，比较两数时，总是先从高位开始比较. 如果 $x_1=1$，而 $y_1=0$，那么无论 x_2、y_2 的值是什么，"$x>y$"都成立，即 F 的值都是为"1"；还有，当 $x_1=y_1$ 同时 $x_2=1$，而 $y_2=0$ 时，亦满足要求. 为此，列出简化的真值表，如表 1-3 所示.

表 1-3

x_1	y_1	x_2	y_2	F
1	0	—	—	1
0	0	1	0	1
1	1	1	0	1

第二步. 由表 1-3 所示的真值表，即可写出逻辑函数 F 的表达式.

$$F=x_1\bar{y}_1+\sum m(2,\ 14).$$

第三步. 化简：经卡诺图(图 1-14 所示)将逻辑函数 F 简化为：

$$F=x_1\bar{y}_1+x_1x_2\bar{y}_2+\bar{y}_1x_2\bar{y}_2.$$

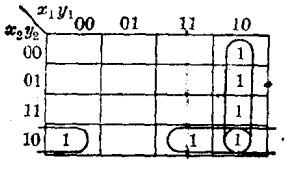


图 1-14

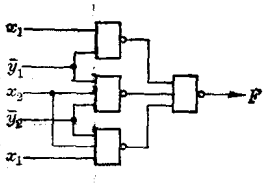


图 1-15

第四步. 画出逻辑电路图，如图 1-15 所示. 选用"与非"门实现.

【例 8】 设计一排队电路. 输入 A、B、C 为电位信号，通过排队电路分别由 F_A、F_B、F_C 输出. 在同一时间只能有一个信号通过，如果同时有两个以上信号出现时，则依 A、B、C 的优先顺序通过. 写出 F_A、F_B、F_C 的逻辑表达式.

第一步. 关于逻辑功能的分析：这个电路有三个输入，三个输出(如图 1-16 所示). 根据逻辑要求，列出真值表(如表 1-4 所示).

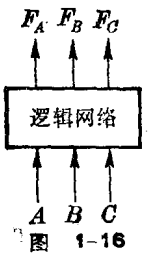


图 1-16

表 1-4

A	B	C	F_A	F_B	F_C
1	—	—	1	0	0
0	1	—	0	1	0
0	0	1	0	0	1

第二步. 由真值表(表 1-4)，可写出逻辑函数表达式：

$$\begin{cases} F_A=A, \\ F_B=\bar{A}B, \\ F_C=\bar{A}\bar{B}C. \end{cases}$$

这是一个具有三个输出的逻辑网络，要化简这样的网络，考虑的问题是多方面的．假如我们把每个函数都看作是相互无关的来进行化简，其结果往往不是最好的，这个问题将在第 1.4 节讨论．

1.4 逻辑函数化简中的几个问题

前面我们化简逻辑函数的方法，只是针对孤立的单个函数的，同时假定输入端既有原变量，也有反变量．但在实际设计中，情况往往要复杂得多．

有的实际逻辑设计中，它的变量之间往往不是孤立的，它们之间经常有一个互相制约的关系．如代表操作(+、−、×、÷)的变量 A、B、C、D 的组合，就不能同时为"1"信号．如果按照相互无关的变量来化简，不会得到最佳的结果．有的实际逻辑设计中，对于一组输入变量，要求有多个输出，而这种多个输出的逻辑函数之间又并非孤立的．如果把每一个逻辑函数都看作相互无关的进行化简，其结果常常也不是最好的．而有的实际逻辑设计中，输入只有原变量，而反变量不能提供．为了得到反变量，往往需要加一级反门，这对于大量变量来说，是很不经济的．

以上这些，都向我们提出了一个问题，就是在一定条件和要求下，怎样化简，才能使所设计的逻辑网络结构简单合理．本节就是要讨论化简中的这样一些问题．

一． 没有反变量输入时，逻辑网络化简问题

用"与非"门进行逻辑设计的时候，特别是设计中、大规模集成电路的时候，为了减少输入端数量，常常要求每个输入信号只用一根线而不用两根线，即只有输入原变量而无输入反变量．一种直接的办法是加反向器将原变量变成其反变量来实现．

对比例 5 的图 1-9 和图 1-10，可以看出，这两种方案虽然都没有反变量输入，可图 1-10 的方案比图 1-9 的方案节省了一个"与非"门．在只有原变量输入的条件下，我们取这个方案．

对于例中的逻辑表达式 $F=A\bar{B}+\bar{A}B$，图 1-9 的方案用了两个"与非"门，以产生 $\bar{A}$ 和 $\bar{B}$．图 1-10 的方案是用一个共同的函数来代替 $\bar{A}$ 和 $\bar{B}$，因而不必再用两个"与非"门．因为在 $\bar{A}\bar{B}$ 这一项中，$\bar{B}$ 可以用 $(\bar{A}+\bar{B})$ 来代替，即 $A(\bar{A}+\bar{B})=A\bar{B}$；在 $\bar{A}B$ 这一项中，$\bar{A}$ 也可以用 $(\bar{A}+\bar{B})$ 来代替，即 $B(\bar{A}+\bar{B})=\bar{A}B$．所以，可以将 F 变换成：

$$F=A\bar{B}+\bar{A}B=A(\bar{A}+\bar{B})+(\bar{A}+\bar{B})B$$
$$=A\overline{AB}+\overline{AB}B.$$

写成"与非-与非"表达式，就是：

$$F=\overline{A\overline{AB}\ B\overline{AB}}.$$

从这个例子中可以看出：用 $(\bar{A}+\bar{B})$ 分别代替 $\bar{A}$ 和 $\bar{B}$ 这一步是最关键的．选择 $(\bar{A}+\bar{B})$ 是由于它具有两条性质：

1．它可以取代两个需要用"反"门产生的反变量 $\bar{A}$ 和 $\bar{B}$．

2．它可以变换成 $\overline{AB}$，因而只用一个"与非"门就能实现．

用来代替 $\bar{A}$、$\bar{B}$ 的 $(\bar{A}+\bar{B})$ 或者 $\overline{AB}$，称之为"代替因子"．如果找不到合适的"代替因子"，则只能用增加"反"门的办法，来得到所需要的每个反变量．

进行没有反变量输入的逻辑网络化简时，遇到下述逻辑函数表达式类型，可以直接提因子简化：

(1)
$$\begin{aligned}&\overline{A}f(x_i)+\overline{B}f(x_i)+\overline{C}f(x_i)+\cdots\\&=(\overline{A}+\overline{B}+\overline{C}+\cdots\cdots)f(x_i)\\&=\overline{ABC\cdots\cdots}f(x_i).\end{aligned}$$

其中 $f(x_i)$ 表示任何变量的与函数，x_i 中也可有反变量存在(但不包含 A、B、C、…… 任一变量)．例如：

$$\begin{aligned}F&=\overline{A}C+\overline{B}C+\overline{D}F+\overline{E}F=(\overline{A}+\overline{B})C+(\overline{D}+\overline{E})F\\&=\overline{AB}\,C+\overline{DE}\,F.\end{aligned}$$

对上式二次求"反"，就可以用"与非"门来实现，这是一个没有反变量输入的三级"与非"逻辑网络，其逻辑电路图从略．

(2)
$$\begin{aligned}&(\overline{A}BC\cdots MN)f_1(x_i)+(A\overline{B}C\cdots MN)f_2(x_i)+\cdots+(ABC\cdots M\overline{N})f_n(x_i)\\&=(\overline{A}+\overline{B}+\overline{C}+\cdots\cdots+\overline{M}+\overline{N})\cdot(BC\cdots MN)f_1(x_i)\\&\quad+(\overline{A}+\overline{B}+\overline{C}+\cdots\cdots+\overline{M}+\overline{N})\cdot(AC\cdots MN)f_2(x_i)\\&\quad+\cdots\cdots\\&\quad+(\overline{A}+\overline{B}+\overline{C}+\cdots\cdots+\overline{M}+\overline{N})\cdot(ABC\cdots M)f_n(x_i)\\&=\overline{ABC\cdots MN}\,[(BC\cdots MN)f_1(x_i)+(AC\cdots MN)f_2(x_i)+\cdots\cdots\\&\quad+(ABC\cdots M)f_n(x_i)].\end{aligned}$$

其中，函数 $f_1(x_i)$，$f_2(x_i)$，…，$f_n(x_i)$ 分别表示 x_i 变量的"与"函数，各个函数彼此间可以不相同，即对于给定的一个变量可以在某个函数中以原变量方式出现，在另一个函数中以反变量方式出现，而在又一个函数中也可以不出现．

例如：

$$\begin{aligned}F&=B\overline{C}E+ABCD\overline{E}\,\overline{F}+A\overline{B}EF+\overline{B}C\overline{D}E\\&=(B\overline{C}E)+(BC\overline{E})AD\overline{F}+A\overline{B}EF+(\overline{B}CE)\overline{D}\\&=\overline{BCE}\,(CE\overline{D}+BE+BCAD\overline{F})+A\overline{B}EF\\&=\overline{BCE}\,C\overline{D}E+\overline{BCE}\,BE+\overline{BCE}\,ABCD\overline{F}+A\overline{B}EF.\end{aligned}$$

采用"与非"门实现上述逻辑函数，得到一个三级"与非"逻辑网络．比直接用一个与非门实现将一个原变量变成其反变量来实现的方案节约了一个"与非"门．逻辑电路图略．

对于一个逻辑函数，往往同时存在上述两种表达式类型．这时，可以同时应用两种方法进行处理．必须指出：上述方法虽然简单，容易掌握，但是有时不能保证得到最简的结果．

没有反变量输入的逻辑网络化简的方法，许多人进行了大量的研究，提出了许多方法．如"并项-代替因子"法，卡诺图法，等等．可是，至今尚未发现系统而又简单的有效方法．这里，我们不准备介绍，读者可以参考有关资料，自己研究．

二．具有多个输出的逻辑网络的化简问题

一个具有相同输入变量而有多个输出的逻辑网络，如果只是孤立地将单个输出函数简化，然后直接拼在一起，在多数情况中并不能保证这个多输出网络为最简，这是因为对于这种网络有时存在能够共享的部分．

为了讨论这种简化问题，首先通过一个例题(利用卡诺图来求解)以说明这个方法的实质．

【例 9】 一个具有四个输入变量、三个输出变量的译码电路，其输出函数为：

$$F_1=\sum m(2,\ 4,\ 10,\ 11,\ 12,\ 13),$$
$$F_2=\sum m(4,\ 5,\ 10,\ 11,\ 13),$$
$$F_3=\sum m(1,\ 2,\ 3,\ 10,\ 11,\ 12).$$

用卡诺图，很容易求出每一个输出函数的最简逻辑表达式．如图 1-17 所示．

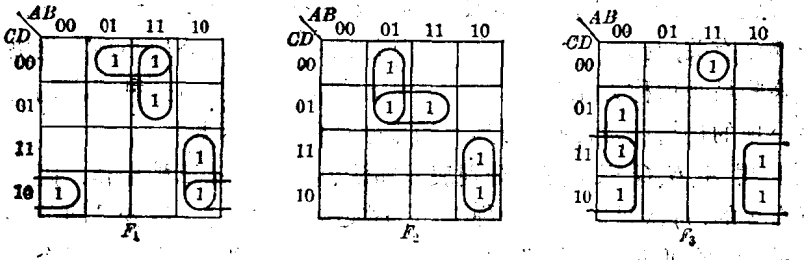


图 1-17

$$F_1=\bar{B}C\bar{D}+B\bar{C}\bar{D}+A\bar{B}C+AB\bar{C},$$
$$F_2=B\bar{C}D+\bar{A}B\bar{C}+A\bar{B}C,$$
$$F_3=AB\bar{C}\bar{D}+\bar{A}\bar{B}D+\bar{B}C.$$

由此可以看出，这里只有 $A\bar{B}C$ 这一项是 F_1 和 F_2 的公用项．总共有九个本原蕴涵项．但当我们用卡诺图重新化简时，虽然不是求得 F_1、F_2 和 F_3 的最简表达式，却可以得到一些三个函数的公用项．如图 1-18 所示．

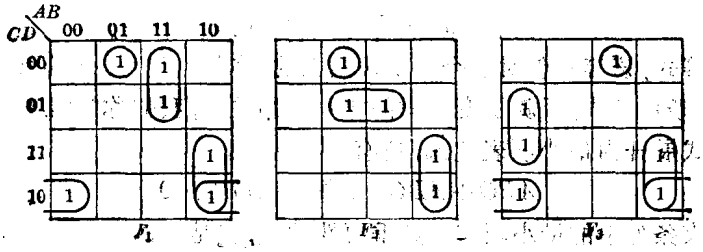


图 1-18

$$F_1=A\bar{B}C+AB\bar{C}+\bar{B}C\bar{D}+\bar{A}B\bar{C}\bar{D},$$
$$F_2=A\bar{B}C+\bar{A}B\bar{C}\bar{D}+B\bar{C}D,$$
$$F_3=A\bar{B}C+\bar{A}\bar{B}D+\bar{B}C\bar{D}+AB\bar{C}\bar{D}.$$

这里的三个函数总计有七个蕴涵项．三个函数的公用项为 $A\bar{B}C$，两个函数的公用项为 $\bar{B}C\bar{D}$ 和 $\bar{A}B\bar{C}\bar{D}$．只有四项是 F_1，F_2 和 F_3 单独要求的．

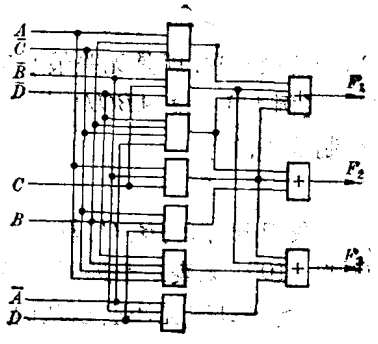


图 1-19

显然尽管每一个输出不是最简化的，但从译码器电路整体看，这却是一个最简化网络．逻辑电路图如图 1-19 所示．

我们从利用卡诺图简化多输出的译码电路中可以看出：简化的要点在于充分利用公用项．但是，用卡诺图就要依赖人们对图形的直观能力，并且一般不能保证求得的逻辑网络是最简的．对这一点，请读者给予充分地注意．

三、具有"约束"条件的逻辑网络化简问题

直到现在，我们所讨论的一组逻辑变量，都假定它们是相互独立的，或者说它们的取值是互不依赖的：即一个变量的取值是"1"还是"0"，与其它变量的取值无关．假设一组变量共有 n 个，那末对于这组相互独立的变量来说，就有 2^n 种不同的取值组合．也就是说，n 个变量的逻辑函数与 2^n 个最小项都有关．

但是，在某些实际问题中，一个 n 变量的函数并不一定与 2^n 个最小项都有关，而仅仅与其中的一部分有关，与另一部分无关．也就是说，这另一部分最小项不能决定函数的取值．我们把这些最小项称为无关(Don't Care)最小项．把具有这种特点的逻辑函数称为包含无关最小项的逻辑函数，或称为具有约束条件的逻辑函数．

【例 10】用 A、B、C 三个变量分别表示加法、乘法、除法三种操作．因为机器是按顺序逐条执行指令的，每次只能执行一种操作，因此 A、B、C 这一组变量间就有一个重要的相互制约关系，即任何两个变量都不能同时取值为"1"．也就是说，这三个变量的取值只可能出现四种情况："000"，"001"，"010"，"100"，而不可能出现另外四种情况："011"，"101"，"110"，"111"．我们称这样的一组变量 A、B、C 是一组有"约束"的变量．

可以用逻辑式来描写它们之间的相互制约关系，即：

$$\begin{cases} A\cdot B=0, \\ B\cdot C=0, \\ C\cdot A=0. \end{cases}$$

将三个等式合并成一个等式，即：

$$AB+BC+CA=0.$$

这是一个条件等式，该等式仅对这个例子是成立的，在其它场合一般说是不成立的．这样一个等于"0"的条件等式，反映了变量间的相互制约关系，我们称为"约束条件"．

把条件等式展开为最小项表达式，则有：

$$ABC+AB\bar{C}+A\bar{B}C+\bar{A}BC=0.$$

包含在约束条件中的最小项有一条重要性质，那就是它的值永远不可能是"1"．也就是说，一个最小项的条件等式，还可以化为一组最小项为"0"的条件等式．即：

$$\begin{cases} ABC=0, \\ AB\bar{C}=0, \\ A\bar{B}C=0, \\ \bar{A}BC=0. \end{cases}$$

这样的最小项，就是无关最小项．

【例 11】设 $ABCD$ 是一位十进制数码的二进制编码，其中 A、B、C、D 表示四个二进制数码，即：

$$x=8A+4B+2C+D.$$

我们分析 A、B、C、D 这一组变量．由于它们是用来表示一位十进制数的，所以它用 0000～1001 来表示十进制数码 0～9，而其它六组取值"1010"、"1011"、"1100"、"1101"、"1110"、"1111"是不会出现的．因此，对应于这六组取值的六个最小项 $A\bar{B}C\bar{D}$、$A\bar{B}CD$、$AB\bar{C}\bar{D}$、$AB\bar{C}D$、$ABC\bar{D}$、$ABCD$ 不可能为"1"，即：$\sum m(10, 11, 12, 13, 14, 15)=0$．所以 A、B、C、D 是一组具有"约束"的变量，m_{10}、m_{11}、m_{12}、m_{13}、m_{14}、m_{15} 都是无关最小项．