第 5 章　可编程逻辑器件与组合逻辑电路设计

数字系统的总成本取决于电路板、电源、互连部件和芯片模块，还包括设计、测试等其他生产开发成本。由于数字逻辑器件通常只占了系统总成本中的一小部分，因此，数字系统的设计人员主要关注如何最小化系统中电路模块的数量，这样就可以最小化电路板的面积，同时减少系统功耗以及其他相关的系统成本。

本章将讨论可编程逻辑器件(Programmable Logic Device，PLD)，这种数字逻辑器件内含由设计人员进行配置和编程的电路，可以实现通常可能需要十几个 SSI 电路模块才能完成的逻辑功能。我们将讨论其基本电路结构，以及常用的三种 PLD 配置，并分析它们是如何应用于实现组合逻辑电路，也将涉及用于 PLD 开发的计算机辅助设计工具。

5.1　半定制逻辑器件

随着集成度的提高，单片电路模块(芯片)中的门数不断增加，使得电路板所使用的芯片数目减少了。在过去的 20 年中，单片芯片的门数从 7400 系列的 SSI 逻辑器件的几个门的规模，增加到目前高性能大规模集成电路芯片(VLSI)中的百万以上的门数。高度集成使得电路板(PCB)的空间大幅减少，同时也减少了功耗。

有三种方法可以实现一个数字电路系统：采用标准的 SSI/MSI/LSI 器件，使用全定制的 VLSI 器件，以及使用半定制的器件。标准的 SSI/MSI/LSI 器件的特点是应用方便，电路板可以采用现成的器件快速地装配完成，但是总的器件数量，包括平均每个门的成本将大到不可接受的程度，把设计集成到一个或多个全定制或半定制的器件中，可以显著地减少器件的数量，从而降低总的成本。

在全定制集成电路设计中，采用绘制每一个门电路的实际版图以及互连线来完成设计。虽然该设计过程周期漫长，代价高昂，通过应用 CAD 工具，可以优化芯片的性能和面积。半定制集成电路设计由于采用了预先设计的门阵列，标准单元或可编程的逻辑器件，可以减少设计的时间。门阵列(Gate Array)是一种包含大量不相互连的门电路的集成芯片，设计者只需要根据功能要求，完成门阵列中各个门电路的连接即可。这样，器件的生产厂商就把器件的生产过程分成了两个阶段：在开始阶段时，先大量生产出只带门阵列的芯片，然后存储起来备用，在最后阶段，只需要把芯片中少量用于完成互联功能的“层(Layer)”完成即可，而不是需要生产整个芯片，从而大幅地减少了生产制造的时间。通过这种方式大量的只剩最后少量互联层空白的门阵列芯片被生产出来，以备未来系统应用时，由设计者进一步定制。

“标准单元”是一些专门开发出来，并保存在标准库中的通用电路模块。设计者通过选择标准库中的标准单元，放置在芯片中的特定位置上，并进一步确定这些单元的互联关系来完成芯片的设计。这个过程与用 SSI/MSI/LSI 芯片设计电路板并无不同。标准单元库中

单元功能从分立的门电路到整个完整的处理器模块，以及其他的复杂电路模块。采用标准单元设计往往比全定制设计在性能优化上要差一些，但大幅度地减少了设计时间。采用标准单元设计出来的芯片必须与全定制芯片一样从头完成所有层的生产过程，以实现专门的系统功能。

使用“可编程逻辑器件(PLD)”可以去掉上述芯片生产中的与功能定制相关的步骤，它包括了一个可灵活可变的互联“层”，可以由电子方法改变连接关系，以实现特定的功能，同时，这种电子化的定制可以由最终用户自己完成。PLD 器件内含可以实现基本的两级逻辑开关表达式的资源，同时也包括其他逻辑单元，因此，在单片的 PLD 芯片上可以实现数千门逻辑门的功能。PLD 电路可以先用逻辑表达式设计，然后转换成 PLD 器件的格式，并用 PLD 编程器烧录到 PLD 器件中来实现，这样，数分钟内即可完成从设计到芯片生产的过程，而不是像在用门阵列或标准单元设计芯片时，需要花费在生产中几天或数周的时间，如果需要，设计也能在数小时甚至数分钟内快速和便宜地完成更改，而标准单元或门阵列设计往往需要花费数天或数周时间，重新进行生产。

5.2 逻辑阵列电路

可编程逻辑阵列电路由一些相同的模块排成阵列来实现，这些模块通过配置可以实现逻辑与和逻辑或的操作。在这一节中，将讨论这些电路的基本结构以及其基本功能，包括设计者可以实现编程，完成定制逻辑功能的机制。

5.2.1 二极管数字逻辑电路

可编程逻辑器件可以用半导体二极管、三极管开关或类似的单元组合在一起来实现。

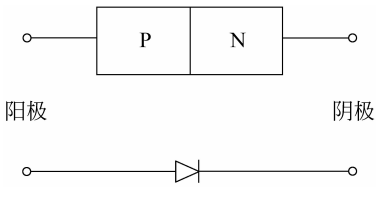

图 5.1 PN 结二极管及其逻辑符号

二极管是一种由两种半导体材料(P 型材料和 N 型材料)，连接放在一起来形成的一种电路元件，如图 5.1 所示。“半导体”是一种比绝缘体(比如橡胶)导电能力强，但比导体(比如铜)导电能力差的材料。“N”型和“P”型分别表示负电荷和正电荷的导电机制。二极管的功能与一个“理想”的开关类似，当加在二极管“P”端和“N”端之间的电压，使“P”端(正极：Anode)比“N”端(负极：Cathode)的电压更高时，二极管处于“正偏”状态。在这个模式中，二极管表现出闭合的开关或短路的功能，并保持一个比较小的导通电压(二极管两端的电压接近于 0)。当负极的电压比正极的电压显著高时，二极管处于“反偏”状态，实际上表现出一个开路或断开的开关的功能(全部的电压保持在二极管的两端)。

在数字电路设计中，每个二极管的一端通过一个电阻与电源或“地”相连，如图 5.2(a)和图 5.2(d)所示，二极管的另一端在一个数字逻辑信号控制下，使得处于二极管“正偏”或者“反偏”的状态。考察如图 5.2(a)所示电路的工作情况，当信号 A 是逻辑 1 时(正电压)，二极管处于“反偏”状态，相当于如图 5.2(b)中所示开路情况，这时，信号线 B 由电阻拉高到电源电压，使得信号 B 处于逻辑 1；当信号 A 处于逻辑 0 时(0 伏)，二极管处于“正偏”状态

而导通，相当于如图 5.2(c)所示的短路情况，使得信号 B 的电压拉低为逻辑 0。图 5.2(d)和图 5.2(e)显示了把电阻连接到地时的类似效果。

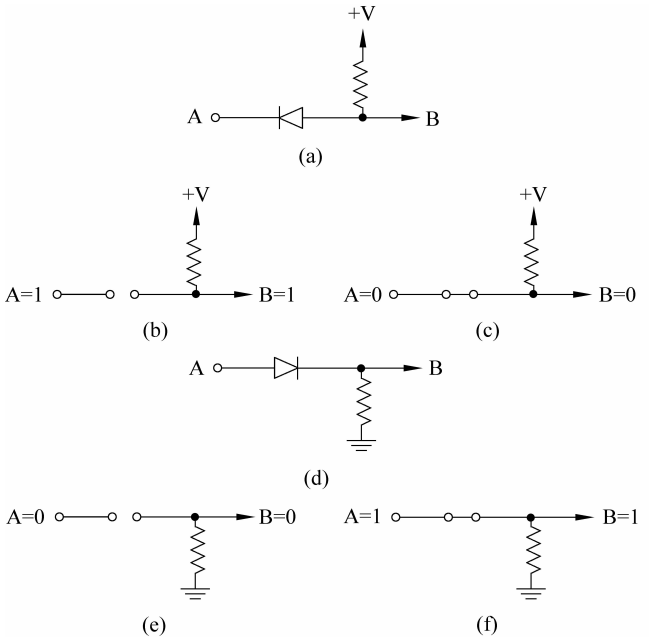

图 5.2　PN 结二极管在数字电路中的用法

(a) 带上拉电阻；(b) 反偏＝二极管开路，B 上拉到逻辑 1；

(c) 正偏＝二极管短路，B 下拉到逻辑 0；(d) 带下拉电阻；

(e) 反偏＝二极管开路，B 下拉到逻辑 0；(f) 正偏：二极管开路，B 上拉到逻辑 1

在可编程逻辑器件中，一般大量采用其他电子器件，比如三极管而不是二极管。虽然本章中主要讨论二极管组成的逻辑电路，但其中的数字逻辑设计原理是一样的。读者可以参考其他文献进一步了解。

5.2.2　“与”逻辑阵列和“或”逻辑阵列

上一节中描述的二极管逻辑电路表明我们可以利用 PN 结二极管实现各种开关逻辑函数的功能。例如，图 5.3(a)中的电路实现了三个输入(A、B 和 C)的与门功能。我们可以通过推导该电路的真值表来验证这一点。当 ABC＝111 时，所有三个二极管都处于开路状态，如图 5.3(b)所示电路的输出被拉高到逻辑 1；当 A 变为逻辑 0 时，对应的二极管变成短路状态，如图 5.3(c)所示，使得电路的输出拉低到逻辑 0；由于其他两个二极管保持开路状态，它们对输出没有影响。当 ABC＝101 和 ABC＝110 时，情况与前面类似。当多个输入为逻辑 0 时，每个对应的二极管均处于短路状态，同样输出也将拉低到逻辑 0，如图 5.3(d)所示，因此，这个电路最终实现了一个三输入与门的功能。推而广之，实际上可以用 K 个二极管实现 K 个输入的与门电路，读者可以自行验证这一点。

或门可以采用图 5.4 所示的电路来实现。在这个电路中，当 ABC＝000 时，所有的二极管开路，输出为逻辑 0，如图 5.4(b)所示。当 ABC＝100 时，如图 5.4(c)所示，与 A 相连的二极管导通(短路)使得输出为逻辑 1。由于另外两个二极管保持开路，它们不会影响输出

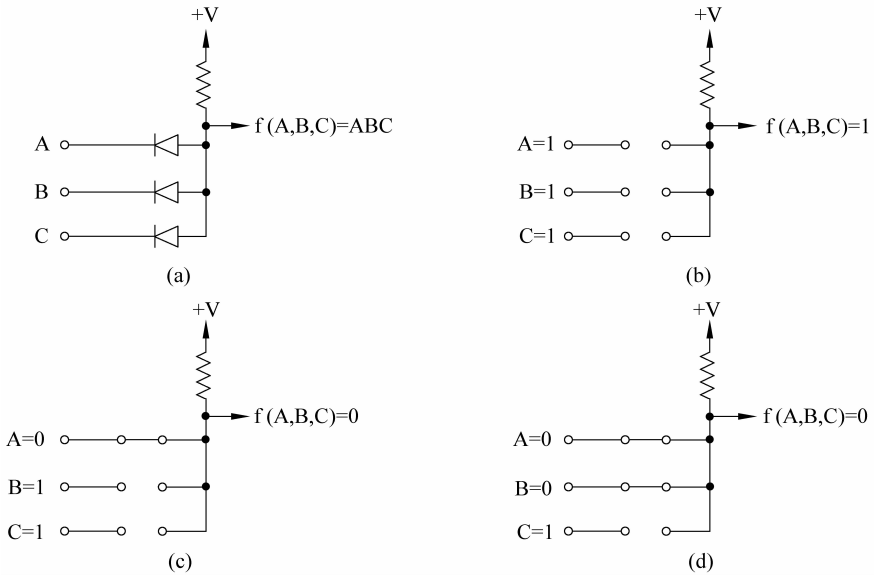

图 5.3 用二极管阵列实现与门功能

(a) 基本电路形式；(b) 所有二极管开路，f 拉高到 1；

(c) 一个二极管短路，f 拉低到 0；(d) 多个二极管短路，f 拉低到 0

的电平。同样，当任意其他的输入或其他输入中有一个或多个为逻辑 1 时，输出也为逻辑 1，因此，该电路实现了三输入或门的功能。这个电路同样可以推广到用 K 个二极管实现 K 输入的或门。

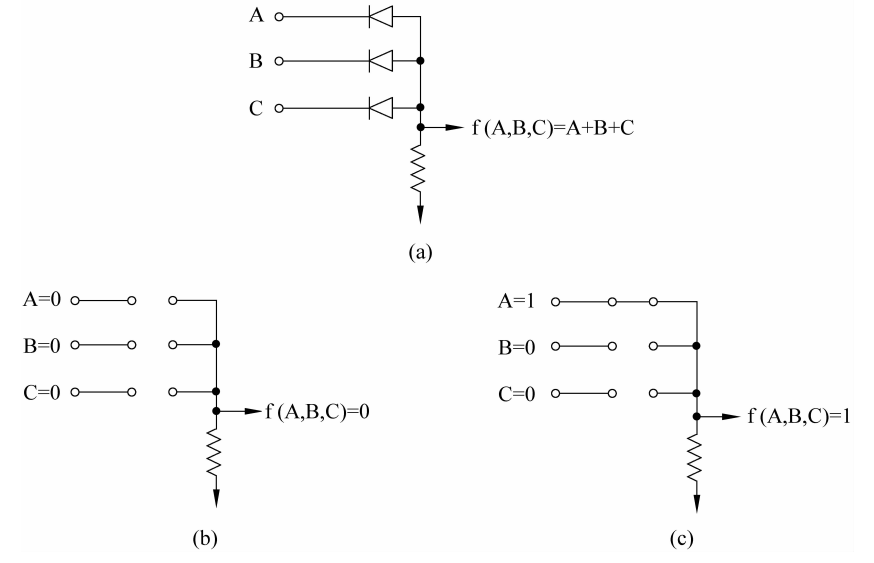

图 5.4 用二极管阵列实现或门功能

(a) 基本电路形式；(b) 所有二极管开路，f 拉低到 0；(c) 一个二极管短路，f 拉高到 1

5.2.3　两级与-或阵列

上述的“与”电路和“或”电路可以作为基本逻辑门电路采用类似的方式互连，从而实现任意的开关逻辑函数，例如，对于下面的这个两级积之和形式的逻辑函数：

$$f(a,b,c) = ab\bar{c} + \bar{b}c$$

图 5.5(a)中给出了一种使用二极管逻辑阵列实现该函数功能的电路。图 5.5(b)中给出了另一种更简略的形式，通常在用图表示二极管阵列电路时采用，逻辑“与”功能用图中的竖线连接一个代表上拉电阻的“与”门符号来表示，而逻辑“或”功能用图中的横向连接一个代表下拉电阻的“或”门符号来表示，而图中的“×”代表了各个二极管，有时候为了简洁起见，会不画“与”符号。

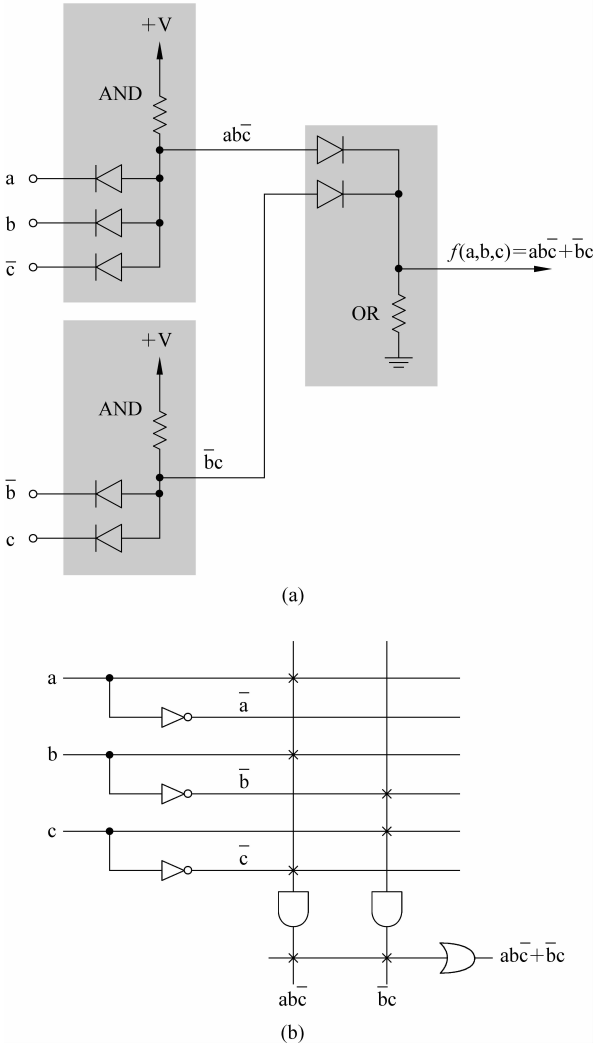

图 5.5　用与-或阵列(AND-OR)实现的积之和逻辑

(a) 与阵列与或阵列相连；(b) 简略形式

图 5.6 中显示了如何通过增加更多的 OR 阵列电路来实现多个函数功能，其中实现了下式中的两个函数：

$$f_1(a,b,c) = ab + \bar{c}$$

$$f_2(a,b,c) = ab + \bar{b}c$$

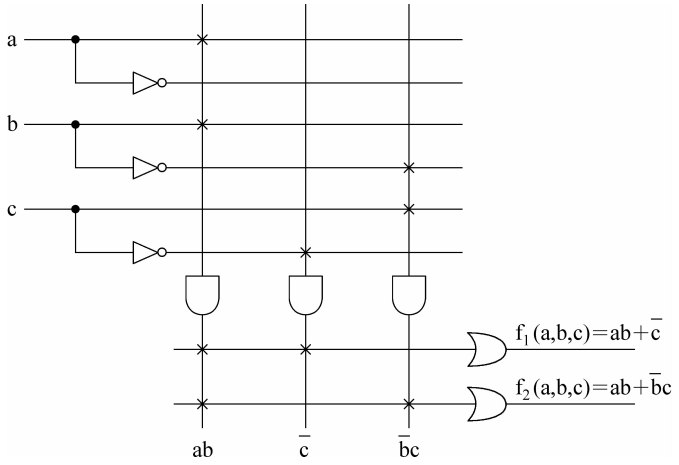

图 5.6 一个与-或阵列实现多个逻辑函数功能

注意到图中的积项 ab 在两个函数中都用到了，也就是说，它被两个或门“共享”了。因此，通过确定阵列中二极管的具体位置，可以实现对该阵列的定制或编程，以完成预定的函数功能。

这种把可编程的“与”阵列与可编程“或”阵列前后组合连接在一起的电路，常常称为“可编程逻辑阵列”(Programmable Logic Array，PLA)。PLA 可以通过定制或编程其中的各个二极管的组合，从而实现任意的逻辑函数。

例 5.1 设计一个实现下列三个函数的 PLA 电路，并画出简略图。

$$f_1(A,B,C,D,E) = \bar{A}\bar{B}\bar{D} + \bar{B}C\bar{D} + \bar{A}BCD\bar{E}$$

$$f_2(A,B,C,D,E) = \bar{A}BE + \bar{B}C\bar{D}E$$

$$f_3(A,B,C,D,E) = \bar{A}\bar{B}\bar{D} + \bar{B}C\bar{D}E + \bar{A}BCD$$

解：由于共有 5 个变量，因此，PLA 需要 5 个输入，每一个输入同时提供两个信号，其中一个与输入信号逻辑相同，另一个与输入信号逻辑互补。三个函数共有 7 个不同的积项，因此，PLA 需要提供至少 7 个积项。最后，由于要同时实现三个函数，因此，PLD 必须含有三个和项(OR)。

PLA 的结构如图 5.7 所示。表 5.1 中列出了 AND 阵列和 OR 阵列需要连接的位置。表中，积项的编号对应图 5.7 中与门的编号，每个与门连接的竖线上的节点可以产生一个积项。在表中 AND 阵列的部分，0 代表变量的互补信号连接到积项的线上，1 代表该变量直接连接到该积项线上，×代表变量的互补信号和原信号均连接到积项的线上。对于 OR 阵列，1 代表连接，0 代表不连接。

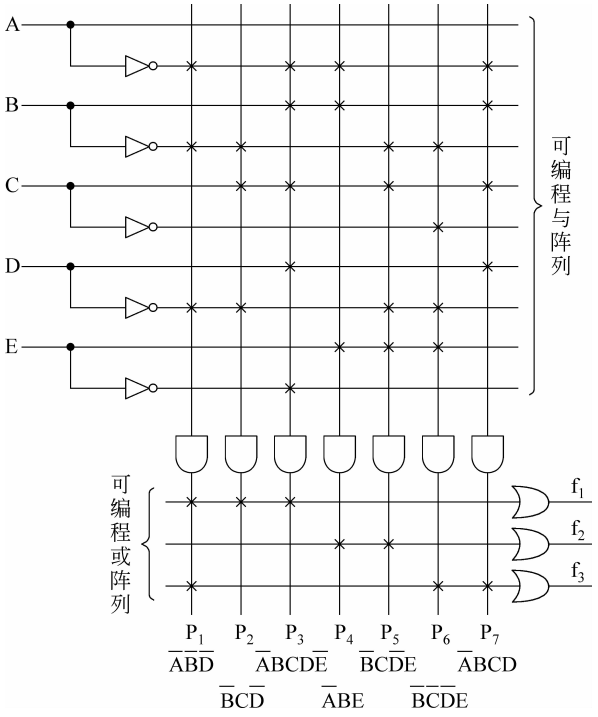

图 5.7　例 5.1 的 PLA 电路图

表 5.1　例 5.1 的 PLA 表格

	积　项	与阵列输入 ABCDE	或阵列输出 f_1 f_2 f_3
1	$\bar{A}\bar{B}\bar{D}$	00×0×	1 0 1
2	$\bar{B}C\bar{D}$	×010×	1 0 0
3	$\bar{A}BCD\bar{E}$	01110	1 0 0
4	$\bar{A}BE$	01××1	0 1 0
5	$\bar{B}C\bar{D}E$	×0101	0 1 0
6	$\bar{B}\bar{C}\bar{D}E$	×0001	0 0 1
7	$\bar{A}BCD$	0111×	0 0 1

例 5.2　图 5.8(a)中所示的一个五输入“多数表决器”，当输入多数为 1 时，该表决器的输出为 1。利用 PLA 设计该电路。

解：依题意，当最小项中的 1 的个数等于三或大于三时，这些最小项对应的输出为 1。因此，对应的函数表达式如下：

$$f(a,b,c,d,e) = \sum m(7,11,13,14,15,19,21,22,23,25,\cdots,31)$$

利用卡诺图法或其他方法，可以得到写成最小项之和形式的函数表达式，如下式所示：

$$f(a,b,c,d,e) = abc + abd + abe + acd + ace + ade + bcd + bce + bde + cde$$

该函数的 PLA 实现方式如图 5.8(b)所示。

当设计人员采用全定制或标准单元法设计 VLSI 电路时，他们常常利用 PLA 模块而不

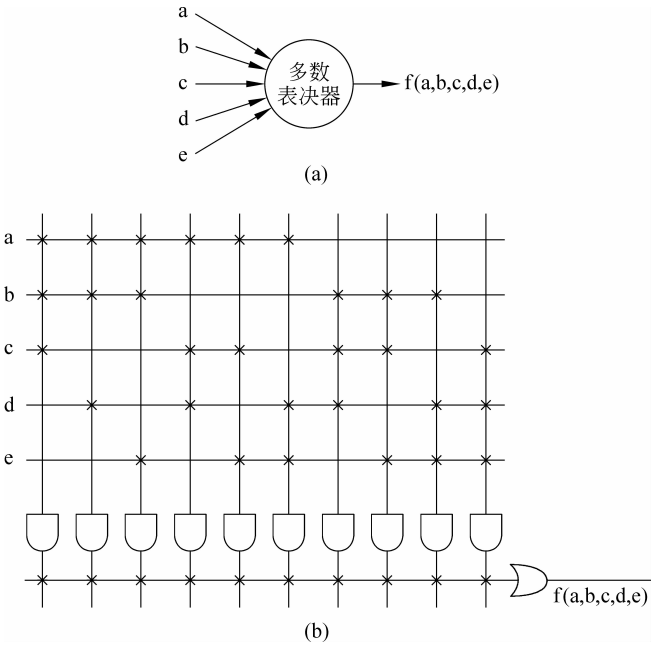

图 5.8 例 5.2 的多数表决器

(a) 如果多数输入为 1 时,表决器输出为 1;(b) PLA 实现

是分立的逻辑门电路来实现他们设计中的组合逻辑电路部分,PLA 中“与”阵列和“或”阵列的连接关系一般会在器件制造过程中完成,也称为“掩膜编程”。每个 PLA 的结构包括输入信号的数量,积项数以及和项数都根据具体的要求进行定制。使用 PLA 设计具有很多优点:首先,PLA 制造时比分立门电路更紧凑,因此占用更少的芯片面积。第二,CAD 设计工具可以直接从逻辑方程式中,自动生成 PLA 的结构和版图,从而减少设计时间。最后,目前开发出来的 PLA 电路的测试算法,相比分立门电路的测试算法的效率要更高些。

5.2.4 现场可编程“与”阵列和“或”阵列

“现场可编程”(Field-programmable)逻辑器件是一种带有未完成定制的与/ 或阵列模块的器件,这种器件可以由设计者自行完成与/或阵列的编程或配置,而不是由芯片厂商来实现定制。大多数标准的现场可编程(或简称为可编程)的与/或阵列,均可通过让设计者确定或编程阵列中的二极管连接方式,实现需要的积项和和项,来实现任意开关函数功能。

为了实现器件的可编程,一根由镍铬、钛钨或类似的合金构成的金属线与一个二极管串联在一起,放在该二极管和输出积项之间,具体参见图 5.9(a)。未熔断的熔丝如同短路,把对应的二极管与输出相连。熔丝可以采用通过加载一个大的电流的方法来熔断去除掉,使得输出与对应的输入没有关系。

一个具体的可编程与阵列具有如图 5.9(a)所示的内部结构。每个输入变量和它的互补输入变量,均通过二极管和熔丝与输出相连。通过把选择好的熔丝去除掉,可以实现变量 A、$\overline{A}$、B、$\overline{B}$、C 和 $\overline{C}$ 的任意组合的积项。例如,下面的开关函数:

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

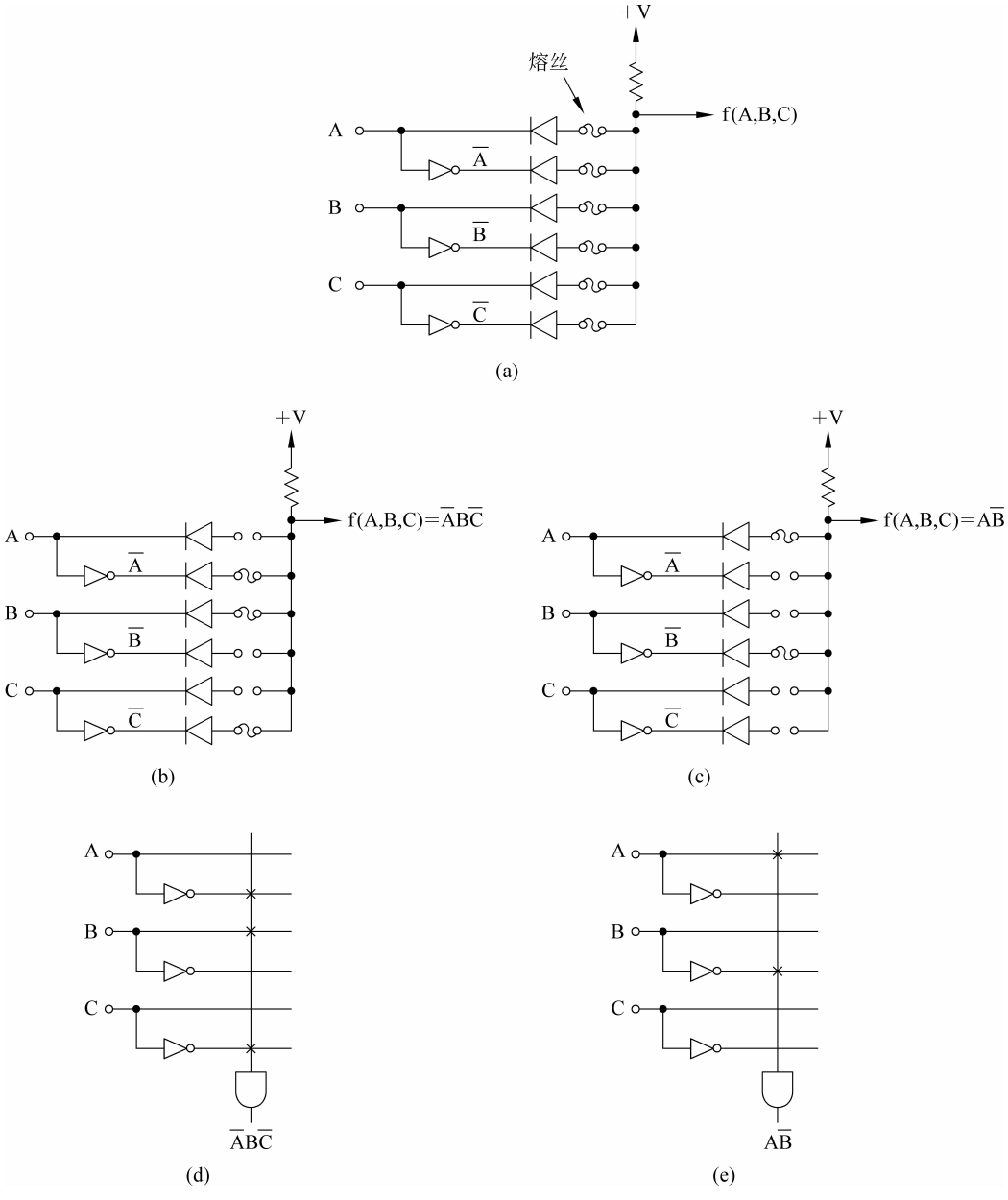

图 5.9　熔丝编程与阵列

(a) 未编程的与阵列；(b) f(A,B,C)=$\overline{A}B\overline{C}$；(c) f(A,B,C)=$A\overline{B}$；

(d) 图(b)的简略表示法；(e) 简略表示法

如图 5.9(b)中所示，通过把那些与输入信号 A、$\overline{B}$ 和 C 相连的二极管去掉连接，同时保持与信号 $\overline{A}$、B 和 $\overline{C}$ 相连的二极管保持不变，来实现这个函数的功能。类似地，开关函数：

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

可以通过去掉如图 5.9(c)中的对应四条熔丝来实现。

我们通常采用如图 5.9(d)和图 5.9(e)所示的简单示意方法，来表示可编程逻辑电路的

编程配置，而不用具体画出每个二极管及其熔丝的图形。图中节点处的×代表存在一根熔丝，如果没有×，则代表没有熔丝，也即该熔丝已熔断。读者可以自己对照一下图 5.9(d)和图 5.9(b)，以及图 5.9(e)和图 5.9(c)来验证这一点。

图 5.10(a)中是一个可编程或阵列的电路图，采用与前述的可编程与阵列同样的方法来实现。或阵列的输入 P_1、P_2 和 P_3 通常是与阵列所产生的积项。下列函数的输出如图 5.10(b)所示来实现，对应简略表示成图 5.10(c)。

$$f(P_1,P_2,P_3) = P_1 + P_3$$

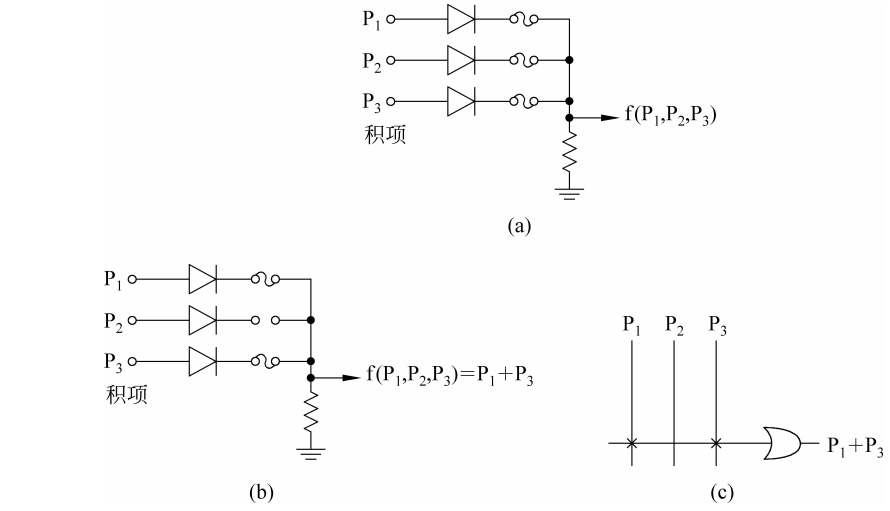

图 5.10 熔丝编程或阵列电路

(a) 未编程的或阵列；(b) $f(P_1+P_2+P_3)=P_1+P_3$；(c) 简略图示法

我们把从一片可编程逻辑器件中去除那些特定熔丝的过程，称为对这个器件“编程”。具体实现的方法一般是：先使用 CAD 软件把需要实现的逻辑函数(比如开关逻辑表达式)转化成熔丝熔断的图样，然后把这个图样交给一个称为“编程器”的设备，由该设备按照图样选择指定的熔丝，并加载电流，完成该熔丝的熔断，从而实现编程。

5.2.5 输出极性控制

标准可编程逻辑器件，除了产生积项和和项外，还会增加一些其他的特性，例如可编程的输出极性，信号反馈和双向信号控制管脚。图 5.11(a)中画出了一般 PLD 器件中在输出属性上的各种选项：高电平有效，低电平有效，双极性输出以及可编程的极性输出。

输出极性的控制是通过由一根熔丝作为一个异或门(XOR)的输入来实现的。当熔丝保持不动时，该输入信号为逻辑 0，而当熔丝熔断时，该输入信号为逻辑 1。根据 XOR 门的功能，当熔丝不断时，如图 5.11(b)所示，输出 $O_i = S_i \otimes 0 = S_i$，因此，输出为高电平有效。当熔丝熔断时，如图 5.11(c)所示，输出 $O_i = S_i \otimes 1 = \overline{S_i}$，因此，输出为低电平有效。要注意的是，×代表保持完整的熔丝，而没有×代表熔丝熔断了，这一点与与或阵列中的情形是一致的。

除了能够控制输出信号的高电平有效和低电平有效之外，输出极性控制也使得积之和