第10卷 第9期 半 导 体 学 报 Vol. 10, No. 9
1989 年 9 月 CHINESE JOURNAL OF SEMICONDUCTORS Sept., 1989

一种可编程序逻辑阵列的布局算法

薄 建 国
(中国科学院半导体研究所,北京)

1988 年 7 月 5 日收到

本文提出了一种解决可编程序逻辑阵列 (PLA) 布局的新算法. 这个算法综合分析考虑"与" (AND) 和"或"(OR)平面所形成的"组" (GROUP) 之间的互连关系,在此基础上,形成了初始布局. 本算法同时又考虑了总体的合理性,还照顾到局部的合理性,对初始布局进行了改善. 实验结果表明,本算法在减少 PLA 的面积方面,具有明显的优越性.

主题词: 可编程序逻辑阵列, PLA 特征语言, 布图, 布局及布线, PLA 折迭法,通道区布线,线网及其权重,超大规模集成电路(VLSI)

一、引 言

可编程序逻辑阵列 (PLA) 在开关和时序逻辑电路中得到越来越广泛的应用. PLA 的特征语言 (PLA-personality language) 不仅准确描述了其功能,而且指明了它的布图特性,因而使有关的布图设计可以在较高的层次上进行,这就为缩短布图设计周期创造了极为有利的条件. 由于通常的 PLA 特征语言所描述的 PLA 布图特性不能充分利用芯片面积, 因之人们提出了一些算法来改变 PLA 的结构以求在不改变原 PLA 功能的条件下,尽可能减少芯片面积. 一种最常用的方法就是折迭算法 (folding). 通常, 构成 PLA 的"与" (AND) 和"或" (OR) 平面都是很稀疏的,即使沿行和列方向各折迭一次, 面积仍然不能充分利用. 多次折迭, 又会引入布局和布线问题. 为了尽可能提高面积利用率,我们提出了一个新算法.

如果一个 PLA 有器件处的位置用一个"元件" (cell) 代换,而把没有器件处的位置用一个"线条" (wire) 代换,则 PLA 的 AND 和 OR 平面就可分别被转换成"元件-线条"平面 (cell-wire). 由于"线条"比"元件"的尺寸要小得多,因而这个 PLA 所占芯片面积就会减小. 这种基于尽量压缩 PLA 所占芯片面积, 而又尽可能不打乱原 PLA 的有序结构的想法是 1985 年提出的. 这个算法,一共包括有三个部分的工作. 第一部分是完成通道区布线及行、列的交换算法[1]; 第二部分就是本文提出的布局新算法. 由于原 PLA 的有序结构基本上没有被打乱, 故使得重新布局和布线变得比较容易解决. 在第一部分的工作里, 我们已经把某些"行" (row) 形成了固定的"组". 这些组又可分为两类: 一类是由 AND 平面形成的;另一类是由 OR 平面形成的. 现在的问题是,如何对这些组进行布局,才能使整个 PLA 所占芯片面积达到最小. 本文提出的算法,就是为解决这个问题所设计的.

二、连接矩阵

我们用 AND-1，AND-2，…来代表 AND 平面的组号；使用 OR-1，OR-2，…来表示 OR 平面的组号．每个组中所包含的行（row）号用 1,2,3…… 来表征．那么，要实现原 PLA 的功能，显然必须把在 AND 平面中和 OR 平面中具有相同行号的行连接起来．为了准确表达它们的连接关系，可以使用连接矩阵．我们采用下述方法来建立 AND 和 OR 平面中各组的连接矩阵．首先，列出各组中所包含的行号，如表 1(a)所示．然后，按照各组中所包含行数的多少为顺序，把 AND 组填在连接矩阵的首列，把 OR 组填在这个矩阵的首行，在各行和各列的交点处分别写上相应的 AND 组（例如 AND-*i*）和 OR 组（例如 OR-*i*）相连线的根数．如果不相连，就填上数 0．对于表 1(a) 例子的

表 1(a)　各 AND 及 OR 组所含行号的实例之一

AND		OR	
AND-1	3 6 9	OR-1	18 17 19
AND-2	12		
AND-3	18 17 19	OR-2	2 5 4 6 1 3
AND-4	2 5 4 1	OR-3	9 11 10 12 8 7
AND-5	8 11 10 7	OR-4	13 16 15 14
AND-6	13 14 16 15		

表 1(b)　表 1(a) 的连接矩阵

	OR-2	OR-3	OR-4	OR-1
AND-4	4	0	0	0
AND-5	0	4	0	0
AND-6	0	0	4	0
AND-1	2	1	0	0
AND-3	0	0	0	3
AND-2	0	1	0	0

表 1(c)

	OR-2	OR-3	OR-4	OR-1
AND-4	4	0	0	
AND-5	0	4	0	
AND-6	0	0	4	
AND-1	2	1	0	
AND-2	0	1	0	
AND-3				3

表 1(d)

	OR-2	OR-3	OR-4	OR-1
AND-4	4	0		
AND-5	0	4		
AND-1	2	1		
AND-2	0	1		
AND-6			4	
AND-3				3

连接矩阵，如表 1(b) 所示.

根据连接矩阵，我们可以立即找出 AND 组和 OR 组之间的连接状况. 从实例表 1(b)中，AND-1 和 OR-1 是不相连的，AND-4 和 OR-2 有四条线要连接.

不难看出，连接矩阵的某行或某些行(某列或某些列)可以和另行或另外某些行(另列或另外某些列)自由交换. 交换之后，原 PLA 的连接关系不变. 例如，表 1(b) 中，AND-4 这行可以和任何行，比如 AND-3 这行进行位置交换.

如果一个连接矩阵的某行或某些行 (某列或某些列)的元素不为 0，而在这行或这些行(这列或这些列)上的其余元素均为 0；并且在元素不为 0 的这行或这些行(这列或这些列)对应的列(或行)上的其余元素均为 0，那么，这个连接矩阵便可分裂成两个独立的子连接矩阵. 所谓"独立"，就是这两个子连接矩阵之间没有 row 向的连线. 例如，表 1(b) 中，AND-3 和 OR-1 就可分裂出来，如表 1(c) 所示，重复此过程，表 1(c)可再次分裂，成为表 1(d).

连接矩阵的行和列实际上是没有首尾之分的. 即，连接矩阵的最后一行是和第一行(注意，不是首行!)连在一起的，而其第一列(注意，不是首列!)是和最后一列连接在一起的. 所以，连接矩阵可以被想象成是一个首尾连在一起的实体.

三、初始布局及总体改善

1. 评价函数

为了评价一个布局的好坏，首先应确定什么样的布局才是好的. 通常，我们认为，如果采用一种布线方案，使完成布局之后，所占用的面积最小，同时又能使连线总长及其中

最长的那条连线也最短，那么这种布局就是最好的．由此看出，布局和布线实际上应统一考虑．可是，由于布局和布线问题的复杂性，人们通常是把它们分开来处理的．如果一个布局方案要等到布线完成之后，才能和另一种布局方案作比较，那么要对各种可能的布局方案作比较，就得花费大量的机时．对于复杂的布线问题，这实际上是不可取的．这里，我们提出一种十分简捷的评价一个布局方案好坏的方法：计算评价函数．

对于我们所面临的实际问题，评价函数可以这样来选取：当连接矩阵中的元素存在着 0 的情况下，分别沿行向和列向计算所有不为 0 的元素之间跨越 0 的个数；对于每个这样的 0，都加上一个"权重"；然后求其总和，就得到了此评价函数之值．这里所谓的"权重"，可以按如下方法来求得：如果是沿行(列)向计算的话，则它是这个 0 对应的列(行)向的那个组的高度之值．此高度包括组内的所有行的元件高度及各行元件之间的布线高度．此值，我们已在[1]中求得．不难看出，这种方法是十分简便的．如果我们希望得到更为精确的值的话，那么可以在此之前，进行组间的通道区布线，以得到各组间的布线高度，把它加到上述有关值之中．

2. 布局的改善

基于连接矩阵的行(列)可交换性及连续性，可以对其进行交换，形成不同的布局方案．每次交换，都要设法把矩阵元素为 0 的区域往"表面"移动．对于连接矩阵中各元素都不为 0 的情况，则交换的目的是把那些元素值大的尽量往"核心"移动．在程序中，我们把使最长连线最短化作为次要的目标来考虑．具体的处理过程可以简述如下：

```
if (any element is zero)
  (
    calculate its evaluation function
    calculate the longest connective net's length
    previous value = evaluation function
    previous length = the length of the longest connective net
    for all possible row exchange
      for all possible columnexchange
        calculate the evaluation function
        calculate the current longest connective net's length
        current value = the evaluation function
        current length = current longest connective net's length
        if (current value < previous value)
          result = current placement
        else if (current value = previous value)
             if (current length < previous length)
               result = current placement
)
else
  do
```

(

exchange the element which has large value near the core

)

while (no any element which has large value than the element which has small value is on the near surface)

四、布局的进一步改善

1. 对称变换

计算每个 AND 和 OR 组的互连线的相对长度. 这里，连线长度以曼哈顿距离计算. 对每一个组，找出它的"中线"位置.

定义：正负代价. 自一个组的某端点引一条线至另个组的某个端点. 若完成这条连线时，连线不跨越这个组的中线，则这条连线对这个组讲，具有正的代价；否则称为具有负的代价.

根据上述得到的改善布局方案，对每个组都计算每条连线对这个组的代价，然后将它们加起来以求得总的代价. 注意，在计算中要考虑组内各行间的布线高度. 如果事先做了组间通道区布线的话，组间的布线高度也要考虑进去. 从结果中找出总代价最小且为负的那个组，将它做沿其"中线"的 x 轴对称变换. 重复这整个过程，使不再出现总代价为负的组为止. 注意，当进行了某个组的 x 轴对称变换后，组间的通道区布线必须在上述改变过的地方重做. 物理意义是，考虑到组间的相互影响，要把那些应当做 x 对称的组，做沿其"中线"的 x 轴对称，以缩短 AND 和 OR 间的连线长度，从而使面积减小. 从图 1 中，不难看出它的作用.

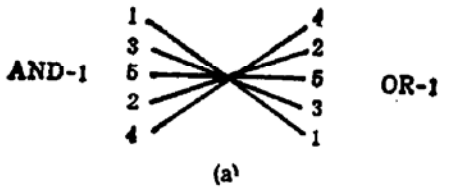

图 1(a) 原来的布局方案

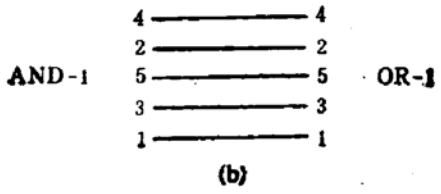

图 1(b) AND-1 做 x 轴对称之后

2. 局部改善

综上所述，我们都是在做总体上的改善. 在局部看来，仍然存在某些不合理性，需要做进一步的改善.

(1) 从较多的实例研究中，我们发现，在某些情况下，一些 AND 和 OR 组的连线很多，它们形成了"子块". 而块内和块外的连线却很少，这就是所谓的成团结构，这从连接矩阵中也是可以发现的. 可以把一个团看成是一个"准独立子连接矩阵"进行相应的处理，可使 AND-OR 间的连线进一步缩短.

(2) 在许多 PLA 的实例中，有些行含有相同的元素，而且这些元素的排列顺序也是完全相同的. 在进行组的划分中，它们可能存在于同一个组内，也可能存在于不同的组

表 2　实验结果

NAME	I/O	PT	DENS(%)	MIN(%)	FOLD (%)	RES(%)	TIME(s)	cell/wire
alcom	15/38	47	8.1	85.1			384.5	5/1
alu1	12/8	19	9.9	100.0		35.1	89.2	5/1
br1	12/8	34	48.2	55.9		65.3	145.4	5/1
br2	12/8	35	48.7	37.1		62.7	166.2	5/1
BFS	23/15	21	20.8		56.8	44.2	120.3	5/1
BFS	23/15	21	20.8		56.8	38.5	121.2	10/1
clpl	11/5	20	13.9	100.0		53.5	92.9	5/1
dekoder	4/7	10	59.3	90.0		78.7	27.9	5/1
ex1	20/14	10	48.2			56.3	63.0	5/1
log8mod	8/5	46	38.1	82.6		61.9	173.4	5/1
misj	35/14	48	3.1	72.9			499.9	5/1
newapla	12/10	17	17.8	100.0		56.7	113.1	5/1
newapla1	12/7	10	24.5	100.0		47.2	43.4	5/1
newapla2	6/7	7	36.8	100.0		51.6	23.3	5/1
newcpla2	7/10	19	28.3	100.0		65.6	86.3	5/1
newcwp	4/5	11	35.0	100.0		52.3	27.1	5/1
p82	5/14	24	34.9	87.5		64.6	102.1	5/1
pope. rom	6/48	64	52.0	92.2			622.0	5/1
p ex3							54.7	5/1
p. ex4	12/9	21	10.4	85.7		34.3	84.6	5/1
p. ex5		32				73.3	52.0	5/1
p ex6		34				58.9	96.6	5/1
p. ex7							53.9	5/1
rd53	5/3	31	48.9	100.0		63.1	81.8	5/1
risc	8/31	74	11.7	37.8		48.2	523.5	5/1
SCA	23/19	52	11.1		59.5	32.0	314.8	5/1
sex	9/14	23	15.4	91.3		50.3	158.8	5/1
sqr6	6/12	63	42.1	74.6		70.7	221.1	5/1
sqr6	6/12	63	42.1	77.8		70.7	221.1	5/1
wim	4/7	10	60.7	90.0		82.7	28.1	5/1

NAME——所用实例的名字
I/O——PLA 的(输入/输出)数
PT——PLA 的 product term
DENS——原 PLA 的元件所占密度
MIN——使用 minimization 所得面积/原 PLA 面积
FOLD——用折迭算法所用面积/原 PLA 面积
RES——本算法所用面积/原 PLA 面积
TIME——本算法所用机时(包括 PLA 输入，通道布线、优化及布局)
cell/wire——本算法采用的(元件/线条)面积之比

内. 显然，这些行是可以互换的. 交换的目的，是使 AND 和 OR 间的连线缩短.

(3) 在一些实例中，行间是不需要通道的. 由于我们假定布局是自下而上进行的，这些无需通道的行应当尽可能多的放到底部去. 否则，当下部某处由于 AND 中的行间通道数和 OR 中的行间通道数不同造成 AND 和 OR 间连线发生弯曲时，在其上部的那些原来不需要通道的也跟着发生弯曲，使 AND 和 OR 间的布线区面积增大.

五、组间布线通道的改善

如果组内含有不同数目的元件，那么组的长度也就不同．较短组的长度和较长组的长度之差，对 AND 和 OR 间的布线影响不大．但对确定组间布线通道数，影响一般是较大的．可以使短的在适当处加长，以减少组间的通道区布线所需通道数，从而减少面积．在[1]中的通道区布线中，也存在着类似的情况．由于这个问题已超出本文所讨论的范围，我们就不再赘述了．

六、结 语

表 2 中举出了近三十个例子的结果，并和其它方法做了比较．图 2 画出了一个实例的布局结果．不难看出，这个算法具有一定的优越性．其中 BFS 是一种折迭算法的例子[3]，假设原 PLA 的面积为 100，应用这种折迭算法后，可使面积缩为原来的 56.8%．应用我们提出的算法，若使用(元件/线条)的面积比为 5/1，面积缩为 44.2%；(元件/线条)面积比为 10/1 时，面积仅为原来的 38.5%，例子 SCA 选自[2]．而例子 p.ex3 至 p.ex7

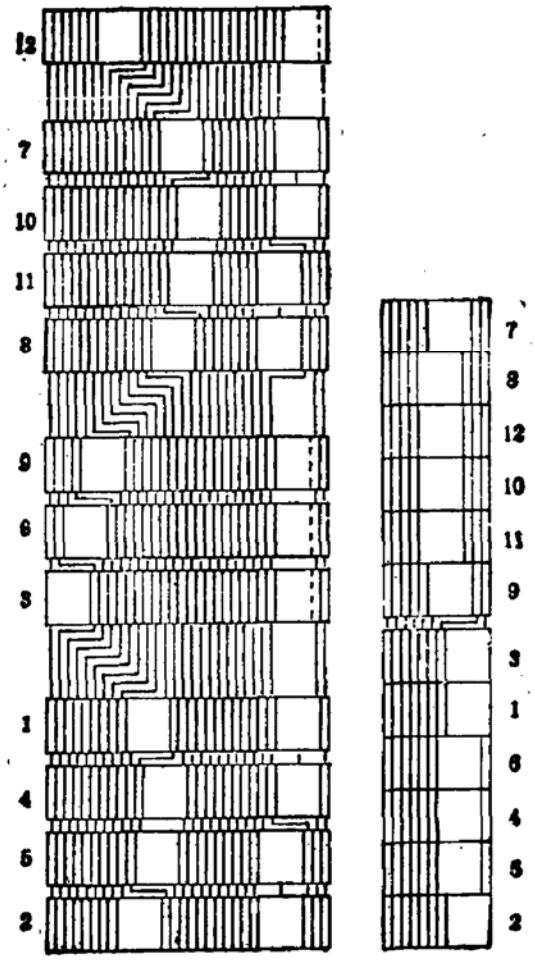

图 2 一个实例的结果示意图

选自菲利普公司的实例，其余是从美国加州大学伯克利分校提供的供全美 PLA 测试用的三种类型的测试标准例子中选出的(工业实用化的、数学上及随机产生的三种类型)。对于大部分的例子，面积都有不同程度的缩小。表 2 中的 TIME，给出了自原始数据的输入直至布局的完成所化费的机时 (VAX-11/780)，它包括了文献[1]的处理所化费的机时(分组、组内布线、择优等)。不难看出，此方法对 PLA 的压缩是有效的。

参　考　文　献

[1] C. A. Papachristou and J. G. Bo, *IEEE ICCAD-87*, November 1987, pp. 432—435.

[2] G. D. Hachtel, A. R. Newton and A. L. Sangiovanni-Vin Centelli, *IEEE Trans. on CAD.*, **CAD-1**, 63 (1982).

[3] S. Y. Hwang, R. W. Dutton and T. Blank, *IEEE Trans. on CAD.*, CAD-5, 433(1986).

[4] 庄文君，王守觉，电子学报，13，1(1985).

A Placement Algorithm for PLA

Bo Jianguo

(Institute of Semiconductors, Academia Sinica, Beijing)

Abstract

A placement for PLA based on the connective relation among the groups which are formed from AND and OR plane is described. It includes the initial placement and some improvements. It has both overall consideration and local reasonability. The algorithm is successfully applied to some benchmarks provided by Berkeley, several application circuits provided by Philips and some examples. The comparisons with other methods are given.

KEY WORDS: Programmable Logic Array (PLA), PLA Personality Language, Layout, Placement and Routing, PLA Folding, Channel Routing, Net and its Weight, VLSI