第 22 卷第 4 期　　半 导 体 学 报　　Vol. 22, No. 4
2001 年 4 月　　CHINESE JOURNAL OF SEMICONDUCTORS　　Apr., 2001

MARS: A General Multilayer Area Router

MA Qi[1] and YAN Xiao-lang[2]

(1 *IC-CAD Research Center, Hangzhou Institute of Electronic Engineering, Hangzhou* 310037, *China*)
(2 *Institute of Electric Engineering, Zhejiang University, Hangzhou* 310027, *China*)

Abstract: Based on a ripped-up and rerouted methodology, a multilayer area detailed router is presented by using simulated evolution technique. A modified maze algorithm is also performed for the single net.

Key words: multilayer area detailed router; simulated evolution; modified maze algorithm
EEACC: 7410D; 5120
CLC number: TN405.97　　**Document code**: A　　**Article ID**: 0253-4177(2001)04-0516-04

1 Introduction

Area routing presents some unusual constraints to the routing problem compared with channel routing and switchbox routing models. In an arbitrary rectilinear shape, area touting has terminals that locate anywhere on or inside the boundary specified, and in which, pre-routed nets and obstacles are allowed. Mighty[1] and Beaver[2] used to be regarded as area routers, but actually they are switchbox routers. SILK[3] and Echelon[4] are rip-up and re-routed area routers. The above algorithms, all neglecting the crosstalk, only support either reserved layer modes or unreserved ones, without any optimization of via number or support to the stacked via.

In this paper, a VLSI general multilayer area detailed router, named MARS, is presented. It has the following features: supporting both reserved and unreserved laver modes; stacked via having been supported and via minimization been carried out as part of routing process; critical nets having a priority routing and the crosstalk being taken into consideration.

Based on a rip-up and reroute methodology, MARS is described by using the simulated evolution technique. A feasible initial solution is first found and then optimized for the rip-up and reroute. For the single net, a modified maze algorithm is also performed.

2 Rip-up and Reroute

In MARS, with the simulated evolution technique, the set of nets to be ripped-up and rerouted has been determined. The simulated evolution is to score the creature in the current generation and determine its survival rate over the next generation (iteration), and at last to obtain a lower score and a higher probability to survive and then replace those who fail to survive with new ones that have different characteristic.

In this case, the creatures are nets. To optimize wire length, via number and crosstalk simultaneously, we define the "score" of a net as follows:

score= $\alpha\times$(the number of vias− the number of vias in the lower bound) + $\beta\times$(ac-

MA Qi male, was born in 1968, Ph. D, interested in design automation of integrated circuit.
YAN Xiao-lang male, was born in 1947, professor and Ph. D tutor, interested in design automation.
Received 6 May 2000, revised manuscript received 13 October 2000

tual wire length/lower bound) + $\gamma \times$ (wire length that possibly causes crosstalk if the net is critical) (1)

where α, β and γ are the constants standing for the optimizing strength for via counter, wire length and crosstalk, respectively. The lower bound for a net is the length routed by modified maze router when no other net is routed. The lower bound on the wire length has been measured for each net before the iteration starts.

Ensuring that all the nets are routed, MARS is to minimize:

$$\sum_i (\text{the weight of net } i) \times (\text{the score of net } i) \quad (2)$$

Greedy or other heuristic approaches is always to choose the nets with high score, i. e., bad nets, however, it may get trapped in a local optimal. MARS' simulated evolution technique copes with these weaknesses by using a probabilistic methodology, which decides whether a net will be ripped-up or not by comparing the net's score with a randomly generated value. In practices, MARS normalizes the score within the range between 0.1 and 0.9, and a random number generator is used to produce a uniformly distributed random number ranging between 0 and 1. A net is marked by ripped-up when its normalized score is greater than the randomly generated number. After all the comparisons are done, all nets marked by ripped-up queue to be rerouted. The order of the nets in the queue is determined by their respective score, with the worst net in the front. The queued nets are then rerouted one by one.

3 Modified Maze Algorithm

The modified maze algorithm in MARS is a multilayer, multiterminal maze routing one. The search starts from every component of the net, which is a heap (priority queue) of routing points. The point with the smallest cost among all the heaps is chosen for expansion. Components and the paths found are merged into a single component when their expansions meets each other. The procedure will not stop until there is only one component left.

In this algorithm, the expansion wave is propagated with different costs for one unit of connection in the following cases:

1) In reserved layer model: The cost is 1 for one unit connection in preferred direction and the expansion in non-preferred direction is forbidden.

2) In unreserved layer model: To improve the routability, we assume the preferred direction for each layer. Although the preferred direction is not strict, it can improve the routability and quality of routing results, especially largely reduce the via counter in comparison with the reserved layer model. The cost is also 1 for one unit connection in preferred direction and $a(a > 1)$ for non-preferred direction. If an expansion changes its present expansion direction (results in a bend), the cost of this expansion has to add an extra value, in this way, the number of bends of a routed net is reduced as possible.

3) In some special cases, some nets are required to route in one or several specified layer(s) as possible. So, the cost is 1 for one unit connection in these specified layer(s) and $b(b > 1)$ in the other non-specified layer(s). Of cause, the cost of direction-changed expansion has to increase.

4) Custom cell synthesis has special constraints on the routing problem, whose intra-cell routing is so-called limited unreserved layer model, that is to say, Poly layer is not routed as possible, while Metal1 layer is used as possible, and Metal2 layer is used as little as possible and has a preferred direction. So the cost is 1 for one unit expansion in both vertical and horizontal directions in Metal1 layer, $c(c > 1)$ both directions in Poly layer, $d(d > 1)$ in preferred direction and $e(d < e < c)$ in non-preferred direction in Metal2 layer.

5) The cost for a via connection is $f(f > 1)$. If stacked via is not supported, the one followed the via expansion can't be a via one, and these two via

expansions can't be merged.

6) When a wire segment (without bend) in a critical net is laid out parallel with another routed wire segment, which is longer than a fixed length $L1$, the cost of the expansion in line direction will increase. When a line is laid out parallel with another router line for longer than another fixed length $L2(L2> L1)$, the expansion in line direction is forbidden. This method is to reduce the crosstalk.

7) To obtain a smaller routing area in compaction, the real spacing of adjacent wire segment should be as small as possible, which is at most the minimum wire-to-via spacing provided so that the adjacent vias are not allowed in routing (in this case, the via-to-via spacing can define the spacing of the adjacent wire segment). So, in the modified maze algorithm in MARS, a via expansion causing adjacent vias are added by an extra value $g(g> 1)$.

Our multilayer modified maze algorithm is described as follows:

```
Procedure modi_Maze_algorithm(net)
begin
compCounter= the number of components;
init heaps for WF of all component;
repeat
    m= the heap whose top element is the
       smallest among the heaps;
    point= heapRemoveTop(m);
    direction= preMazeSpread(m, point);
    if (direction= 0) (* spreading from
       point cannot cause m meet another
       component* )
        for (six directions) do
            spreadPoint = mazeSpread
            (point, direction);
            spreadPoint.father= point;
            heapAddElement(m, spread-
            Point);
    else (* spreading from
point in direction cause m meet one component* )
        componentMerge (m, point,
        direction);
    decrease compCounter;
until(compCounter= 1)
end
```

In preMazeSpread(m, point), if component m can't meet other component after spreading in all six directions, 0 is returned; if m can meet another component in one direction, this direction will be returned; and if m can meet other component in more than one direction, the direction, in which the cost is smallest when meeting, is returned. mazeSpread(point, direction) does a maze expansion according to the corresponding expansion cost. componentMerge(m, point, direction) merges two met components.

4 Experimental Results

We have implemented our algorithm in C++ language on a Sun Ultra-Sparc 10 workstation under Solaris 2.6. Table 1 lists the size of some detail routing benchmarks, and table 2 presents the results of these benchmarks.

Table 1 Benchmarks

Name	# Net	Row	Col.	Critical Net Id
Burstein's Difficult[2]	24	16	24	15, 9
More Difficult[2]	24	16	23	15, 9
Augmented Dense[2]	19	19	17	18, 1
Modified Dense[2]	19	19	17	18, 1
Terminal Intensive[2]	24	17	24	19, 9, 1
With Obstacles*	14	11	15	5

* Fig. 23 Instance in Ref. [2] with Terminals' Layer Modified

Table 2 Routing Results in Case Crosstalk not Considered and Crosstalk Considered

(Unreserved Layer Mode, a= 5, c= 10, d= 20, Stacked via Supported)

Benchmark	3 Layers[2]		4 Layers[2]		3 Layers[3]		4 Layers[3]	
Name	Length	# via	Length	# via	Length	# via	Length	# via
Burstein's Difficult	607	55	546	37	568	40	552	43
More Difficult	531	40	528	36	550	46	555	42

Benchmark Name	3 Layers[2]		4 Layers[2]		3 Layers[3]		4 Layers[3]	
	Length	# via	Length	# via	Length	# via	Length	# via
Augmented Dense	550	36	540	32	561	42	555	44
Modified Dense	522	40	517	34	522	40	562	56
Terminal Intensive	631	58	642	50	631	58	649	55
With Obstacles	/	/	179	24	/	/	179	24

2　Crosstalk not Considered　　3　Crosstalk Considered, $L1=8$, $L2=24$

References

[1] Hyunchul Shin and Alberto Sangiovanni-Vincentelli, A Detailed Router Based on Incremental Routing Modification: Mighty, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 1987, **CAD-6**(11): 942—949.

[2] J. P. Cohoon and P. L. Heck, Beaver: A Computational-Geometry-Based Tool for Switchbox Routing, IEEE Trans. Comput.-Aided Des. Integr. Ciruits Syst., 1988, **7**(6): 684—697.

[3] Youn-Long Lin, Yu-Chin Hsu and Fur-Shing Tsai, SILK: A Simulated Evolution Router, IEEE Trans. Comput.-Aided Des. Integr. Ciruits Syst., 1989, **8**(10): 1108—1114.

[4] Mohan Guruswamy and D. F. Wong, Echelon: A Multilayer Detailed Area Router, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 1996, **15**(9): 1126—1136.

MARS: 一个通用多层区域布线算法

马　琪[1]　严晓浪[2]

(1 杭州电子工业学院微电子 CAD 研究所, 杭州　310037)

(2 浙江大学电气工程学院, 杭州　310027)

摘要: 提出一个 VLSI 多层区域详细布线算法, 算法使用模拟进化技术进行拆线重布线, 对单个线网则使用改进型多层迷宫算法进行布线。

关键词: 多层区域详细布线; 模拟进化; 改进型多层迷宫布线算法

EEACC: 7410D; 5120

中图分类号: TN405.97　　**文献标识码**: A　　**文章编号**: 0253-4177(2001)04-0516-04

马　琪　男, 1968 年出生, 博士, 主要从事集成电路自动化设计.

严晓浪　男, 1947 年出生, 博士研究生导师, 主要从事自动化设计.

2000-05-06 收到, 2000-10-13 定稿