Post-Route Optimization for Improved Yield Using a Rubber-Band Wiring Model

Jeffrey Z. Su and Wayne W. Dai
Dept. of Computer Engineering, University of California, Santa Cruz, CA 95064

Abstract

This paper presents a unique approach to improve yield given a routed layout. Currently after routing has been completed and compacted, it generally proceeds to verification without further modifications. However, to improve manufacturability, we introduce a concept called even wire distribution, a key element of the SURF physical design tool. To alleviate congestion, we first move vias and wires towards less dense areas in a manner which preserves the existing wiring paths. Depending on the locally available area, we then increase wire spacing to reduce defect sensitivity, without changing the area of the design. Carafe, an inductive fault analysis tool is used to evaluate the new layout.

1.0 Introduction

Design for manufacturability (DFM) is becoming increasingly more important as feature sizes have shrunk to the submicron level and die sizes have increased dramatically for high performance microprocessor designs. Although progress has been made in improving manufacturing process capabilities, this alone cannot provide acceptable yield levels for cost sensitive products. As a result, DFM plays an increasing role in the physical design process as solutions require routing and layout changes to optimize for yield. This paper presents a strategy to improve yield by reducing a class of faults commonly known as bridging faults. Bridging faults are shorts between conductors and represent a significant amount of defects responsible for chip failure.

We propose a new strategy called *even wire distribution* (EWD), which reduces defect sensitivity of a routed design by adjusting movable objects and increasing interwire spacing without changing the chip size. Areas of the design with high wiring density are particularly vulnerable to faults. To alleviate congestion in this area, we adjust locations of movable elements, such as vias, to free up area before attempting to increase the spacing. Initial via placement during the routing phase may be sub-optimal from a yield perspective since the router is generally aiming to generate a minimal length route at minimum design rules. EWD is implemented within SURF, an area based router. The EWD optimization takes place after a design rule correct layout has been generated by SURF. Results are then streamed out in GDSII. We then use Carafe, an inductive fault analysis tool, to read the GDSII layout and generate critical area measurements[1]. This critical area measurement determines the defect sensitivity of the layout.

1.1 SURF

EWD is unique in that it operates on a canonical form of topological wiring, called a *rubber-band sketch* (RBS), which is generated by the SURF router[6]. The RBS, is gridless with a more compact data representation of a routed layout, than a typical geometric representation. A RBS represents a single layer of interconnect as flexible rubber-bands which have elastic properties. These rubber-band wires are initially considered to have zero width and zero spacing requirements with other objects in the design. Therefore the rubber-band geometry of a wire path can be envisioned as having minimum length for a specified planar path (Fig. 1a). A set of RBS represents a multilayer area based layout, called a *rubber-band layout* (RBL).

From an RBS, we enforce width and spacing constraints in a process called *spoke creation*, which will be covered in more detail later in this paper. Basically the spoke creation process pushes wires away from nearby objects creating an *extended rubber-band sketch* ERBS (Fig. 1b)[12]. The ERBS naturally represents all-angle wiring and can be constrained to a 45° (octilinear) or a

0-89791-993-9/97 $10.00 © 1997 IEEE

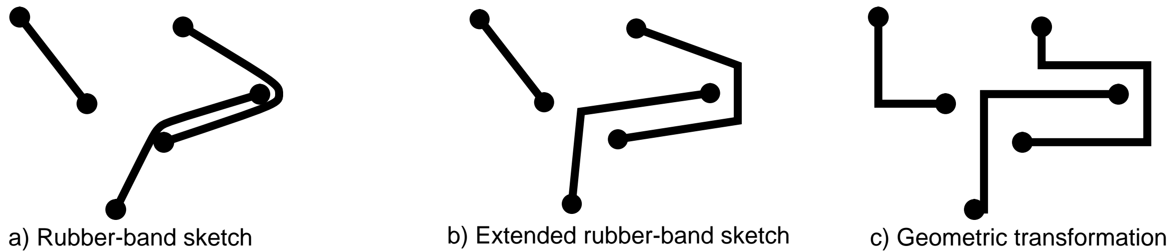

Figure 1: RBS and geometric transformation

Manhattan (rectilinear) wiring pattern (Fig. 1c). A set of multiple ERBS, which compose a layout, is called an *extended rubber-band layout* (ERBL).

The flexibility of the RBS is provided by a key operation called *rubber-band updating* (RBU). RBU is a continuous operation which models the rubber-band behavior by allowing objects within the layout to move while always maintaining wire connectivity. This operation alone provides the mechanism to adjust object locations and manipulate wiring[2].

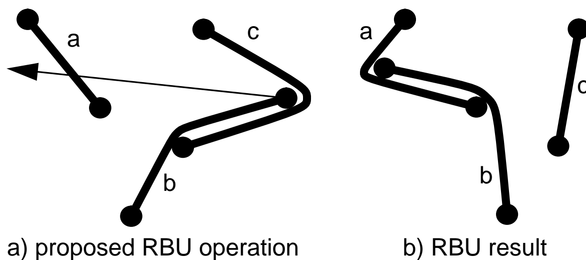

Figure 2: Rubber-Band Updating Operation

A RBS is composed of *terminals* and *branches*. Terminals generally represent pins, via contacts, junction points, or pads. Branches represent the rubber-band wire which connects two terminals. In turn each branch is composed of a polyline where each linear segment of the branch is called a *wire segment*. In Fig. 2a we depict a proposed move of a terminal. Fig. 2b illustrates the result of the move with the branches automatically updated as if they were rubber-bands. Branch *a* which was obstructing the terminal's path, automatically flexes to accommodate the move. Meanwhile branch *c,* originally stretched by the move, snaps to a single wire segment as the terminal moves away.

Since RBU is a layer based operation, we can extend RBU to handle objects which span multiple layers. A *stack* is a set of one or more terminals which exist on different signal layers, share the same x-y coordinates, and must be moved as an entity. However not all stacks, such as I/O pads or pins, can be moved. We define a *movable stack* as any stack whose position is subject to optimization. For the most part, the movable stack is generally a via or a junction point.

The rubber-band sketch concept was originally proposed by Maley and was used to determine if a topological embedding of wiring paths was design rule correct while generating a legal layout (similar to the ERBS) in the process[7]. Chen and Lee implemented a batch mode design rule check which is asymptotically faster than Maley's work[8]. Other works used a topological based layout representation for compaction strategies[9][10][11].

1.2 Previous Work

There are several works which improve yield by spacing or compacting the layout. Chiluvuri and Koren introduced a compaction strategy which took a compacted channel and incrementally shifted wire locations to reduce layout defect sensitivity[3]. A strategy was also presented in this work which increased wire width to reduce the possibility of breaks. Another post-route optimization strategy is called the LocDes spacer[4]. This work introduced local design rules which were used to adjust elements in an IC layout wherever possible to improve yield. The Yield Optimizing Router (YOR), a channel router, introduced the concept of net burying and net floating[5]. Although this approach is primarily focused on routing, the post route optimization aspect includes a via shifting strategy (sliding vias in a horizontal or vertical direction) and net bumping, which shifted wires to unused wiring tracks.

In general, the previous works take a geometric layout and improve yield by manipulating wire and feature geometry. The shortcoming of this approach is that the optimization only considers the nearest geometry to improve the layout without considering the congestion of the surrounding area. Layout geometry is also complex to manipulate for pushing or plowing operations especially if design rules and connectivity are maintained. The amount of memory resources required to edit the layout geometry is also becoming a factor with larger designs.

Current trends in today's IC designs indicate a shift towards high density area-based layouts with a higher number of signal layers. Two works are based on a

channel based layout [3][5] which is becoming less popular. Area based layouts are not exclusive to IC design but also applicable to various packaging technologies such as MCMs, and board layouts such as PCMCIA. Also, octilinear wiring is also becoming more popular as designers attempt to reduce wire length. In addition to this, some packaging technologies can even accommodate all-angle wiring patterns. However the previous works are only applicable to rectilinear layouts.

SURF is a versatile area based physical design tool which has been used to route IC and various packaging technologies[6]. One key advantage of using rubber-band wiring, is that for any layout, a topologically equivalent RBS has minimum wire length. Since the RBS naturally represents all-angle wiring, it can make the most efficient use of the layout for all-angle processes with a simple optional conversion to either a rectilinear or octilinear geometry.

Two works have demonstrated the ability to relocate vias[3][5]. However neither has the ability to incrementally move a via and propogate the results to the surrounding geometry. Only the RBU operation within SURF can move objects while preserving the connectivity of all wires affected by the move and maintain the RBS minimum wiring length property.

2.0 Background

In this paper, we propose a strategy to minimize bridging faults caused by spot defects. Spot defects occur during the manufacturing process, resulting in spots of extra or missing material on the wafer. The size of spot defects are comparable to the feature sizes of the layout such that their presence can alter the intended functionality or performance of the circuit. Another type of defect is a pinhole defect, which are generally much smaller than a micrometer. Dielectric pinholes often occur in chip insulators resulting in missing dielectric material. This defect has the potential to cause a short between overlapping wires on different metal layers. Both spot defects and pinholes have the capability of causing a *bridge fault*. Bridging faults occur when the defect shorts nearby wires. Bridging faults caused by pinholes will not be covered by this paper.

Spot defects are characterized by a circle having a given radius χ. Yet not all spot defects will cause faults. The presence of faults depends on the area where the defect appears, called the *critical area*. The size of the critical area depends on the size of the defect and the spacing between the wires, therefore the larger the defect, the larger the critical area. Since the probability of having a fault depends on the size of the critical area, the goal of EWD is to minimize the critical area for all defect sizes. We first present some commonly known formulae for critical area calculation based on previous work. We then present an overview of the spoke creation process which will introduce some terminology used in the cost computation before defining a global cost function of an ERBL, based on the estimated critical area.

2.1 Critical Area Definitions

The critical area is defined as a function of the defect size. Stapper's work characterized the critical area for short circuits of two conductors parallel for length *L*, separated by spacing *s*, and a defect diameter of χ [13].

$$A(\chi) = \begin{cases} 0 & \text{for } 0 \le \chi \le s \\ L(\chi - s) & \text{for } s \le \chi \le \infty \end{cases} \quad (1)$$

Stapper described a defect density function which is characterized by two parts. The first part describes a positively sloping linear function from $0 \le \chi \le \chi_0$, where χ_0 is the resolution of the photolithography process. Defects $\chi \ge \chi_0$ are characterized by the second part of the function which falls off by a factor of $1/\chi^n$. Experimentation at IBM found that $n = 3$ produces a reasonably accurate fit of the observed defects. With this value the distributions are

$$D(\chi) = \begin{cases} (\chi\bar{D})/\chi_0^2 & \text{for } 0 \le \chi \le \chi_0 \\ (\chi_0^2\bar{D})/\chi^3 & \text{for } \chi_0 \le \chi \le \infty \end{cases} \quad (2)$$

where $\bar{D}$ is the average defect density. We can use these functions to calculate λ, the average number of faults .

$$\lambda = \int_0^\infty A(\chi)D(\chi)d\chi \quad (3)$$

Since design rules are set at some value $s > \chi_0$ defects with a diameter $\chi \le \chi_0$ are not expected to create intralayer shorts. Substituting functions 1 and 2 and evaluating yields the following equation.

$$\lambda = (L\chi_0^2\bar{D})/(2s) \quad (4)$$

From this we can define the average critical area for intralayer shorts between two conductors.

$$\bar{A} = (L\chi_0^2)/(2s) \quad (5)$$

2.2 Design Rule Enforcement in an RBS

The quality of a layout can be determined by calculating the critical area for the design. With an

ERBL, initial wire geometry is known, which is used to provide a estimate of critical area. We use the estimated critical area of the ERBL as a global cost to guide the EWD. However, to understand how the EWD determines wiring density and increases interwire spacing, we first describe the spoke creation process. The spoke creation process checks the minimum design rules and creates a design rule correct embedding using RBU.

The spoke creation is an incremental layer based operation, where each RBS of the layout will be converted to a corresponding ERBS[12]. Each terminal of the layer is processed in a sequential order. For each terminal we first need to perform a design rule check before enforcing the spacing constraints. Design rules of an RBS are checked between pairs of terminals (Fig. 3). A *cut* is a line segment $c = (a, b)$ between two terminals, which has two attributes, *flow* and *capacity*. The flow captures the minimum amount of spacing and routing resources required by branches crossing the cut. The capacity indicates the total amount of available spacing resources between the terminals.

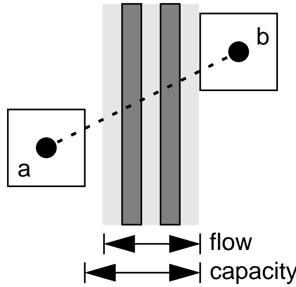

Figure 3: Properties of a cut

A cut (a,b) is routable if $flow(a, b) \le capacity(a, b)$ and is unroutable otherwise. An unroutable cut indicates the presence of a design rule violation (DRV). For a given terminal of an RBS, we check the cuts with respect to other nearby terminals. If no DRVs are found, we then enforce the spacing constraints.

The minimum separation between objects are enforced at the terminals by line segments called *spokes*. Spokes radiate out from a terminal pushing away and separating nearby wires. The spokes take into account the terminal geometry, spacing rules, and branch width. Depending on the final wiring pattern, either four spoke directions are used for rectilinear wiring or eight directions are used for octilinear and all-angle wiring. Once the required spokes have been created for each terminal, we have an ERBS. The ERBS was proven to be routable (design rule correct) by Maley and guarantees a legal geometric transformation[7].

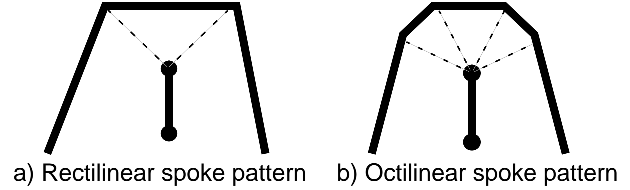

Figure 4: Spoke creation

2.3 Critical area estimation

The critical area estimation is used as the global cost of the ERBL during the EWD optimization process. For any stack move the EWD considers, the critical area estimation can determine the impact on critical area without having to actually move. However to generate critical area estimation for an ERBL, we need to identify parallel wire segments as well as the parallel length L and the spacing s (Eq. 5)

In order to determine relative positioning of objects within an ERBS, we construct an adjacency graph $G_{adj}(V, E)$ by applying a Delaunay Triangulation of the terminals. In a Delaunay Triangulation, the layout area is partitioned into triangular regions where each terminal has edges with the closest neighboring terminals. The edges of this triangulation also represent cuts which will be used in the EWD.

For a given cut, we define the *cut order* $(w_1, ..., w_n)$ as the order in which the wire segments cross the cut Wire segments w_i, w_j of the cut are defined to be *adjacent* if they are a consecutive pair in the cut order. Each wire segment in turn intersects a *sequence* of ordered cuts $S = (c_1, ..., c_m)$. Wire segments w_1, w_2 which are adjacent for one or more consecutive cuts, form a *common sequence* $C_{12} = (c_i, ..., c_{i+k})$ (Fig. 5). We use this common sequence to determine an *interval* during which both w_1 and w_2 are roughly parallel. For each wire segment, this interval is bounded by the cut immediately preceding and immediately following the common sequence $(c_{i-1}, ..., c_{i+k+1})$ (see arrows for w_1). Note that if c_{i-1} or c_{i+k+1} does not exist for a wire segment, the endpoint of the wire segment is used as the interval endpoint (see arrows for w_2). We average the length of both intervals to estimate L for w_1 and w_2.

The spacing of parallel wire segments is defined by available space from each cut of the common sequence. We assume the EWD will evenly distribute wires across the cut and enforce some maximal spacing s between each pair of wire segments and the terminals. The spacing of a cut (p,a) is calculated from the flow and capacity. The flow is composed of two components, one

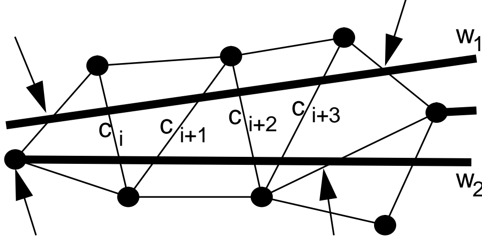

Figure 5: Parallel length approximation

contributed by the sum of the wire widths f_b, and another contributed by the minimum spacing requirements f_s, $f_{ttl}(p, a) = f_b + f_s$. The capacity however, can be expressed as a Euclidean distance function

$$capacity(p, a) = ((x_p - x_a)^2 + (y_p - y_a)^2)^{1/2} - (r_p + r_a) \quad (6)$$

where r represents the respective radii of r and a. From here we can compute the maximal spacing s for this cut with $|W|$ indicating the number of wire segments crossing the cut.

$$s = \frac{((x_p - x_a)^2 + (y_p - y_a)^2)^{1/2} - (r_p + r_a + f_b)}{|W| + 1} \quad (7)$$

The estimated critical area cost, ω_w, for two wire segments is Eq. 5 without the constants

$$\omega_w(w_i, w_j) = \begin{cases} 0 & \text{if not adjacent} \\ L/s & \text{otherwise} \end{cases} \quad (8)$$

The global cost of an ERBL is defined as the sum of the estimated critical area for every pair of wire segments

$$\Omega_g = \sum_i \sum_j \omega_w(w_i, w_j) \quad (9)$$

3.0 Problem Formulation

We divide the EWD procedure into two stages. The first stage is s*tack optimization* stage, which starts from the adjacency graph, and iteratively optimizes each movable stack location in multiple passes based on a local cost function $\Omega_l(p, A_T)$ determined from critical area estimated from neighboring wires. The second stage is *wire spacing* phase. For each terminal, we calculate the amount of increased separation for nearby wire segments based on the surrounding area. We then enforce this larger spacing value by increasing spoke lengths which has the effect of spreading parallel wire segments apart.

3.1 Stack Optimization

Given a set of all possible locations for a movable stack T and a set of neighbors A_T, we assign a cost of moving T to p. We will then use this cost function to evaluate a number of prospective locations prior to moving the stack. The *neighbors* of a terminal t, are the set of terminals $A_t = (a_1, a_2, \ldots, a_n)$ where there exists an edge $(t, a_i) \in E$. The *star region* of t, R_t is the area defined as the union of all triangular areas formed by t and its neighbors Fig. 6.

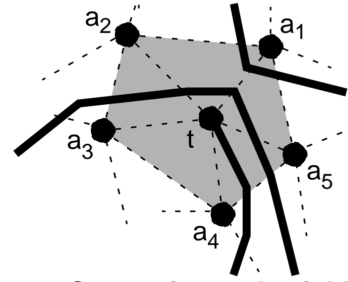

Figure 6: Star region and neighbors

The neighbors of stack T are defined as $\bigcup A_t, \forall(t \in T)$. The *local area* of T is defined as $\bigcap R_t, \forall(t \in T)$ (Fig. 7) for convex star regions (concave star regions need to be trimmed such that they are convex). We will use this local area to restrict the possible locations of the movable stack. Finally we will refer to p as an x-y coordinate located within the local area of which represents a prospective location for the movable stack. By restricting the move to the local area, the triangulation is kept planar. Non-planar triangulation will introduce inaccuracy in the local cost.

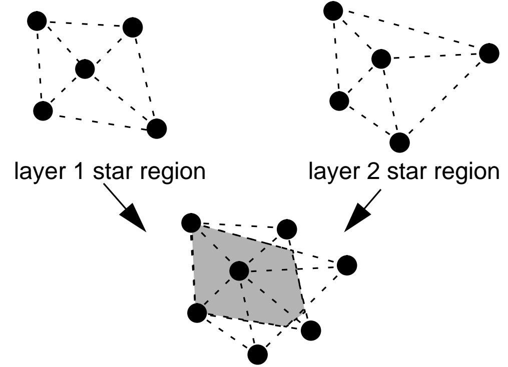

Figure 7: Local area of *T*

A local cost function Ω_l captures the critical area of the star regions for all $t \in T$. This cost, Ω_l will be used to estimate the critical area if T were to move to p. The cost Ω_l is defined as the sum of the cut costs $\omega_c(p, a_j)$ with respect to each neighbor in A_t.

$$\Omega_l(p, A_t) = \sum_{t \in T} \sum_{j=1}^{|A_t|} \omega_c(p, a_j) \quad (10)$$

Each pair of adjacent wires (w_i, w_{i+1}) of the cut

contributes some fraction of ω_w to the overall cut cost ω_c. By considering a move to p, we need recalculate each ω_w contributing to cut (p, a_j). In this case we assume the spacing s will be dictated by (p, a_j) rather than the average spacing specified in Eq. 8. s for each ω_w is recomputed using Eq. 7. Since ω_w includes the cost of span multiple cuts, we divide ω_w by $|C|$ the number of cuts in the common sequence to determine the fraction which contributes to (p, a_j). By summing these fractions for each adjacent wire segment pair (w_i, w_{i+1}), we compute ω_c. In Fig. 5 there are 4 common cuts between w_1 and w_2, therefore $|C| = 4$

$$\omega_c(p, a_j) = \sum_{\forall(w_i, w_{i+1})} \frac{\omega_w(w_i, w_{i+1})}{|C_{w_j, w_{j+1}}|} \tag{11}$$

The overall strategy is to apply the stack positioning optimization for all movable stacks. The optimization will only move the stack if the cost decreases. After each pass, the global cost is recalculated to measure the rate of improvement. This process of processing all stacks and evaluating the cost continues until the global cost converges. Since the local cost is a fraction of global cost, for a given localized area, the algorithm will terminate.

4.0 Implementation

To minimize Ω_g, we minimize the local cost for each stack. The most challenging aspect of this problem is choosing p. The ideal solution is to choose p an infinite distance from A_t where the cost will reach zero. To prevent this, we constrain the domain of p to the local area. There are several problems which may occur if p is allow to be any legal location within the sketch. First if p is located sufficiently far from A_t, there may be other terminals $u \notin A_t$ closer to t, violating key assumptions since these terminals were not factored into the original cost equation. Second, a long distance via move has a higher probability of creating large increases in wire length. Finally p must not cause a design rule violation. In Eq. 7, a sufficiently small distance would create a negative spacing value. To prevent undesired behavior, we constrain the domain of p to locations within the local area where

$$((x_p - x_a)^2 + (y_p - y_a)^2)^{1/2} > r_p + r_a + f_{ttl} \tag{12}$$

with r_p representing the radius of t, r_a the radius of neighbor terminal a, and f_{ttl} which is the total flow between p and a.

In Fig. 8 we used MATLAB to graph a topographical map based on cost for the star region. The polygon indicates the boundary of the combined area of all star regions of t, with the corners indicating the neighboring terminals for all layers. Curves in the figure indicate places of equal cost. The blank areas near the corner of the polygon indicate areas where p would cause a design rule violation.

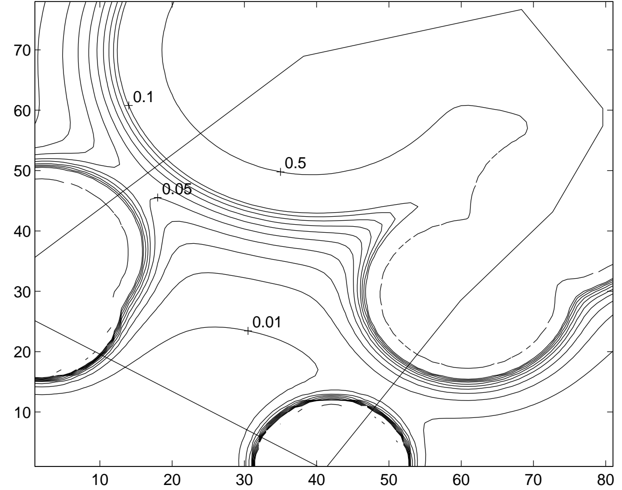

Figure 8: Cost curves of a star region

The EWD algorithm iterates through multiple passes of each movable stack in the design. For each stack, it first calculates an optimal location before moving. Once the global cost converges, the new spacing rules are calculated and enforced by the spoke creation process.

To solve stack optimization problem, we formulate it as a nonlinear constrained optimization. Equation 10 is used as the objective function. The constraints are the line equations which define the local region and Equation 12, to prevent DRVs. We use the Optimal Steepest Descent, using the gradient to find the initial search direction. From the search direction we use a Golden Section Search, a popular 1-D search technique, to find a new point with minimum cost. From this new minimum we calculate a new search direction. This process continues for some predetermined number of iterations or unless the distance between successive iterations reaches some minimum threshold η.

The wire spacing algorithm iterates through the set of neighbors on the appropriate layer for the terminals t of each via. For the given pair of corresponding terminals, the algorithm will calculate the amount of space the cut can support using (Eq. 10). The algorithm chooses and enforces the smallest space s from among the terminal pairs checked. The minimum spacing is set by the design rules while the ideal wire spacing rule is a value set by the user. However, enforcing arbitrarily large spacing values will not have a significant impact on

yield while resulting in increased wire length.

5.0 Experiments

The experiments were run on a Sun Sparc 20 running SunOS 4.1.3. The designs represent sections of MCM-D substrates. For the purposes of the experiment we set the maximum enforced spacing to 2X the minimum wire-to-wire design rule. Each design, was written out to GDSII from SURF both before and after the optimization process. The results were then streamed into CARAFE, which evaluated the critical area of each layout. We also compared the change in wire length resulting from EWD (Table 1).

	Nets	Crit. area before	Crit. area after	Pct. reduction	Change in Wire length
ex1	20	1.42e+08	6.12e+07	57%	+7%
ex2	25	2.61e+08	1.69e+08	35%	+8%
ap1	43	1.26e+09	8.14e+08	35%	+6%
ap2	70	2.18e+09	6.60e+08	69%	+10%
ap3	68	1.75e+09	5.32e+08	69%	+9%
ap4	61	6.53e+08	5.29e+08	9%	+3%

Table 1: Carafe critical area measurements

In each of the designs, we assume a high quality of routing. That is each trace is routed with minimal wire length and a minimal number of vias. We believe that this will allow EWD to make the most significant reductions in critical area. However the largest reduction in critical area also results in the largest increase in wire length although layout area is unaffected by our approach. In future work, we plan to take a closer look at the relationship between the critical area and the wire length

6.0 Conclusion

This work demonstrates the potential of EWD based on a rubber-band wiring model. The ERBL, a data efficient layout representation, has enough information about the routing to provide a critical area estimation. This estimation is used as cost function to determine an ideal location for a movable stack. The RBU operation provides EWD with the ability to manipulate the layout in a unique manner, allowing the optimization to find a local solution which minimizes critical area. Overall, much effort has been put into making RBU a robust operation and because of its versatility, it is an essential asset in improving the quality of the design.

7.0 Acknowledgments

Special thanks to Eric Thorne and F. Joel Ferguson for providing Carafe and much appreciated help.

8.0 References

[1] Alvin Jee and F. Joel Ferguson, “Carafe: An inductive fault analysis tool for CMOS VLSI circuits,” *IEEE VLSI Test Symposium*, pp. 92-98, 1993.

[2] W. Dai, R. Kong, J. Jue, and M. Sato. “Rubber band routing and dynamic data representation,” *ICCAD*, Santa Clara, CA, Nov. 1990, pp. 45-48.

[3] V. Chiluvuri, and I. Koren, “Layout-Synthesis Techniques for Yield Enhancement,” *IEEE Trans. on Semiconductor Manufacturing*, vol. 8, no. 2, pp. 178-187, May 1995.

[4] G. Allan, A. Walton, and R. Holwill, “An Yield Improvement Technique for IC Layout Using Local Design Rules,” *IEEE Trans. on CAD*, vol. 11, no. 11, pp. 1355-1362, Nov. 1992.

[5] Sy-Yen Kuo, “YOR: A Yield-Optimizing Routing Algorithm by Minimizing Critical Areas and Vias,” *IEEE Trans. on CAD*, vol. 12, no. 9, pp. 1303-1311, Sep.r 1993.

[6] D. Staepelaere, J. Jue, T. Dayan, and W. Dai, “Surf: a rubber-band routing system for multichip modules,” IEEE Design & Test, pp. 18-26, Dec. 1993.

[7] F. Miller Maley, “Single-layer wire routing and compaction,” Cambridge, Massachusetts, The MIT Press, 1990.

[8] H. Chen and D. T. Lee, “A faster algorithm for rubber-band equivalent transformation for planar VLSI layouts,” *IEEE Trans. on CAD*, 15(2):217-227, Feb. 1996.

[9] Kurt Mehlhorn and Sefan Näher, “A faster compaction algorithm with automatic jog insertion,” *IEEE Trans. on CAD*, 9(2):158-166, Feb. 1990.

[10] P.l deDood, J. Wawrzynek, E. Liu, and R. Suaya, “A two-dimensional topological compactor with octagonal geometry,” *28th DAC*, June 1991, pp. 727-731.

[11] Hiroshi Murata and Yoji Kajitani, “Interactive terminal sliding algorithm for hybrid IC layout,” Trans. on Information Processing Society of Japan, Dec. 1994, vol. 35, no. 12, pp. 2806-15 (in Japanese).

[12] W. Dai, R. Kong, and M. Sato, “Routability of a rubber-band sketch,” *28th DAC*, June 1991, pp. 45-48.

[13] C. H. Stapper, “Modeling of defect in integrated circuit photolithographic patterns,” IBM Journal of Research and Development, vol. 28, no. 4, pp. 461-475, July 1984.