# Stitch-Aware Routing for Multiple E-Beam Lithography

Iou-Jen Liu, Shao-Yun Fang, *Member, IEEE*, and Yao-Wen Chang, *Fellow, IEEE*

***Abstract*—Multiple e-beam lithography (MEBL) is one of the most promising next generation lithography technologies for high volume manufacturing, which improves the most critical issue of conventional single e-beam lithography, throughput, by simultaneously using thousands or millions of e-beams. For parallel writing in MEBL, a layout is split into stripes and patterns are cut by stripe boundaries, which are defined as stitching lines. Critical patterns cut by stitching lines could suffer from severe pattern distortion or even yield loss. Therefore, considering the positions of stitching lines and avoiding stitching line-induced bad patterns are required during layout design. In this paper, we propose the first work of stitch-aware routing framework for MEBL based on a two-pass bottom-up multilevel router. We first identify three types of stitching line-induced bad patterns which should not exist in an MEBL-friendly routing solution. Then, stitch-aware routing algorithms are, respectively, developed for global routing, layer/track assignment, and detailed routing. Experimental results show that our stitch-aware routing framework can effectively reduce stitching line-induced bad patterns and thus may not only improve the manufacturability but also facilitate the development of MEBL.**

***Index Terms*—Algorithms, design, manufacturability, multiple e-beam lithography (MEBL), performance, routing, stitch.**

## I. Introduction

E-BEAM lithography (EBL) is one of the most expected next generation lithography technologies for overcoming the manufacturing limitations of conventional optical lithography. However, the relatively low throughput due to the maskless direct write process constrains EBL from high volume manufacturing. Thus, EBL was only applied to few applications such as photomask fabrication [18]. In recent years, the concept of multiple e-beam lithography (MEBL) has been proposed, which utilizes massively parallel exposure with thousands or even millions of beams to dramatically improve the throughput. Also, several innovative MEBL systems have been under development and have shown very promising lithography performance and cost effectiveness [13], [16], [17], [20].

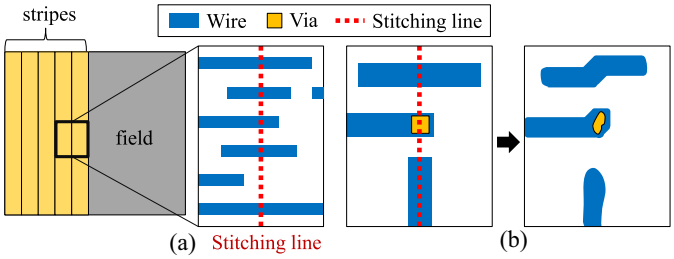


Fig. 1. Layout division and overlay error in MEBL. (a) Layout is split into stripes and the stripe boundaries are defined as the stitching lines. (b) Features cut by stitching lines suffer from different degrees of pattern distortion.

Due to the deflection limitation of each beam and parallel writing strategies in MEBL, a layout (a main field) is split into stripes (subfields) as shown in Fig. 1(a), and we define the stripe boundaries as the stitching lines. Since patterns in different stripes are written by different beams or in different writing passes, a pattern cut by a stitching line suffers from overlay error between two beams or two writing passes [9], [19]. Note that the overlay error could cause different impacts on different types of patterns cut by stitching lines. As illustrated in Fig. 1(b), a horizontal wire can be patterned well even if the overlay error exists. On the other hand, some patterns with critical dimension, such as vias or vertical wires, can have severe pattern distortion and electrical variation due to the overlay error. Therefore, designing MEBL-friendly layouts by considering stitching lines is desirable for enhancing manufacturability. However, to the best of our knowledge, no previous work has addressed the stitching line-induced printability problems during physical design for MEBL.

In current semiconductor manufacturing, metal layers become one of the most critical parts with respect to reliability, manufacturability, and circuit performance, and thus routing plays a crucial role in the VLSI design flow. In MEBL, routing without considering stitching lines may cause stitching line-induced bad patterns. As shown in Fig. 2(a), without stitching line consideration, a via is cut by the stitching line on the wire $A$, and a part of the wire $B$ is vertically routed on the

Manuscript received March 1, 2014; revised June 5, 2014 and September 5, 2014; accepted October 29, 2014. Date of publication January 13, 2015; date of current version February 17, 2015. This work was supported in part by Genesys Logic, IBM, MediaTek, SpringSoft, TSMC, Academia Sinica, and National Science Foundation of Taiwan under Grant 103-2221-E-002-259-MY3, Grant NSC 102-2923-E-002-006-MY3, Grant NSC 102-2221-E-002-235-MY3, Grant NSC 101-2221-E-002-191-MY3, Grant NSC 100-2221-E-002-088-MY3, Grant NSC 99-2221-E-002-207-MY3, and Grant NSC 99-2221-E-002-210-MY3. A preliminary version of this paper was presented at the Proceedings of the ACM/IEEE Design Automation Conference (DAC) in June 2013 [8]. This paper was recommended by Associate Editor Z. Li.

I.-J. Liu is with the Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan.

S.-Y. Fang is with the Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei 10607, Taiwan.

Y.-W. Chang is with the Department of Electrical Engineering, Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 106, Taiwan (e-mail: ywchang@ntu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2014.2385761

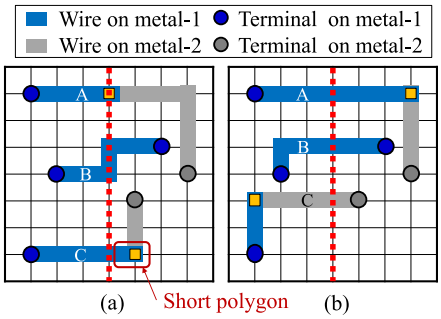


Fig. 2. Routing with and without stitching consideration. (a) Stitching line-induced bad patterns are generated without considering stitching lines during routing. (b) Better routing solution derived from a stitch-aware router.

stitching line. Another undesired pattern occurs on the wire *C*, which is a short wire segment cut by the stitching line with a landing via. We define this type of patterns as short polygons. Short polygons may also cause severe manufacturing defects, which will be explained in Section II-A. Fig. 2(b) shows a better routing result, where no stitching line-induced bad pattern is produced. Avoiding vias cut by stitching lines and avoiding wires vertically routed on stitching lines are not difficult. For example, removing routing tracks vertically overlapped with stitching lines can prevent wires from vertically routing on stitching lines. However, avoiding the generation of short polygons is not trivial. In fact, considering this type of bad patterns could significantly increase the design complexity.

In this paper, we propose the first work of stitch-aware routing framework for MEBL, which minimizes the number of stitching line-induced bad patterns during routing. The framework is based on a two-pass bottom-up multilevel router (similar to [3] for double via optimization), and the stitch-aware routing algorithms are, respectively, proposed in each routing stage: 1) global routing; 2) layer/track assignment; and 3) detailed routing. The major contributions of this paper are listed as follows.

1) Three types of stitching line-induced bad patterns are identified and the three corresponding stitch-aware routing constraints are established. These constraints are then used to guide the router through the framework to generate an MEBL-friendly routing solution.
2) The difference in resource estimation in global routing between MEBL and conventional optical lithography is distinguished. Also, the stitch-aware global routing cost is proposed and integrated into the router.
3) A new layer assignment algorithm based on a segment conflict graph is proposed. Compared to an existing algorithm, our approach has better layer assignment performance with the increasing number of routing layers.
4) Two short polygon-avoiding track assignment algorithms are proposed: 1) an integer linear programming (ILP)-based algorithm and 2) a graph-based heuristic algorithm. Experimental results show that the graph-based heuristic approach is much more efficient and appropriate for the iterative track assignment process.
5) A stitch-aware detailed routing algorithm, which includes the stitch-aware weighted cost of routing grids and the stitch-aware net ordering, is proposed. Experimental results show that the stitch-aware detailed routing algorithm can further reduce the number of short polygons by 80% with zero hard constraint violations.

The rest of this paper is organized as follows. Section II introduces the three routing constraints and the multilevel routing framework. In Section III, the stitch-aware routing algorithms in global routing, layer/track assignment, and detailed routing are respectively presented. Section IV reports our experimental results. Finally, we conclude this paper in Section V.

## II. Preliminaries

In this section, preliminaries of the stitch-aware routing framework are given: the three routing constraints due to stitching lines in MEBL are introduced in Section II-A, and the multilevel routing framework used in this paper is presented in Section II-B.

### A. Stitch-Aware Routing Constraints

As mentioned in Section I, patterns cut by stitching lines suffer from overlay errors between two different beams or two different writing passes. Although these pattern segmentations are inevitable during circuit design, stitching lines should avoid cutting critical patterns to reduce severe pattern distortion or even yield loss. For example, as mentioned in Section I, vias should not be cut by stitching lines and wires should not vertically route on stitching lines.

Another type of stitching line-induced bad patterns, short polygons, is due to the data preparation flow in MEBL. Because of the maskless lithography process, rasterization is required to transform a layout into a pixel-based black/white bitmap, and thus patterns can be exposed on a wafer by controlling each independent beam to be "on" or "off" [9], [12]. Rasterization consists of two major steps: 1) rendering followed by and 2) dithering with error diffusion. In rendering, a layout is sliced into grids, and patterns are converted into pixel-based gray-level data with intensity proportional to the pattern coverage in each pixel. Then, in dithering, the resulting gray-level bitmap is transformed into a black–white bitmap. The error of each pixel due to dithering is not neglected but diffused to its neighboring unprocessed pixels. As illustrated in Fig. 3, to transform the gray-level bitmap in Fig. 3(a) into a black–white bitmap, a dithering algorithm distributes the error of the grid marked by the red point to its right and lower grids. However, the dithering process may cause irregular pixels on feature edges, as shown in Fig. 3(b).

A short polygon may cause a severe defect after the rasterization process. Fig. 4 shows an example. The short polygon cut by the stitching line undergoes rendering and dithering during the data preparation flow. Due to the error diffusion process, the short polygon has irregular pixels on the bottom-right corner. These few error pixels, however, account for a large percentage of the pixels of the short polygon and thus can result in serious pattern distortion after e-beam exposure.

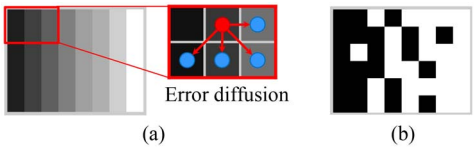


Fig. 3. Dithering with error diffusion. (a) Gray-level bitmap is transformed into a black/white bitmap with a dithering algorithm. (b) Irregular pixels on a feature edge due to dithering.

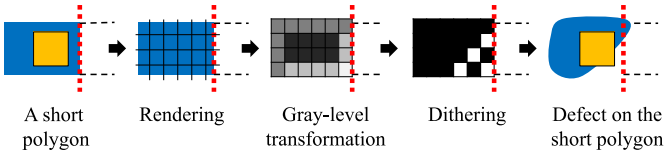


Fig. 4. Rasterization process of a short polygon. A severe defect occurs due to dithering with error diffusion.

Then, the misalignment between the polygon and the via becomes a circuit defect or causes unacceptable electrical variation. Therefore, short polygons with landing vias should be avoided in a routing solution for better MEBL control.

Hence, given a set of stitching lines, we define the following three routing constraints.

1) *Via Constraint:* Vias cannot be cut by stitching lines [see Fig. 5(a)].
2) *Vertical Routing Constraint:* Wires cannot vertically route on stitching lines [see Fig. 5(b)].
3) *Short Polygon Constraint:* Vias should not land on short polygons. As illustrated in Fig. 5(c), we define the area within the distance $\epsilon$ from a stitching line as the stitch unfriendly region of the stitching line. A horizontal wire has a short polygon violation if it satisfies the following two conditions: 1) the wire is cut by a stitching line and 2) at least a line end of the wire lies in the corresponding stitch unfriendly region with a landing via. Therefore, in Fig. 5(c), the upper wire has a short polygon violation, and the lower wire is a preferred routing instance without any violation.

The via constraint and the vertical routing constraint are hard constraints because via violations and routing violations typically cause severe pattern distortions [see Fig. 1(b)] and have direct negative impact on manufacturability. The short polygon constraint is a soft constraint because short polygons do not always lead to defects and thus yield loss. Our routing framework minimizes the number of short polygons, forbids the routing violation, and allows via violations only on given fixed pins. Our stitch-aware routing problem is formulated as follows.

*Problem 1:* Given a netlist, routing planes, and the locations of stitching lines, perform stitch-aware routing to minimize the number of short polygons, the number of vias, and total wirelength such that no vertical routing violation occurs, and via violations only occur on fixed pins.

### B. Multilevel Routing Framework

Various multilevel frameworks have been widely adopted for large-scale routing, in which coarsening and uncoarsening stages are applied according to different optimization

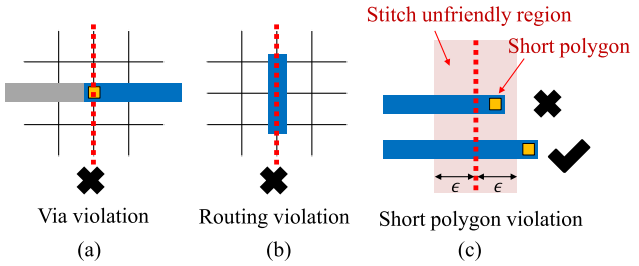


Fig. 5. Three routing constraints for stitch-aware routing. (a) Vias cannot be cut by stitching lines. (b) Wires cannot vertically route on stitching lines. (c) Vias cannot land on short polygons.

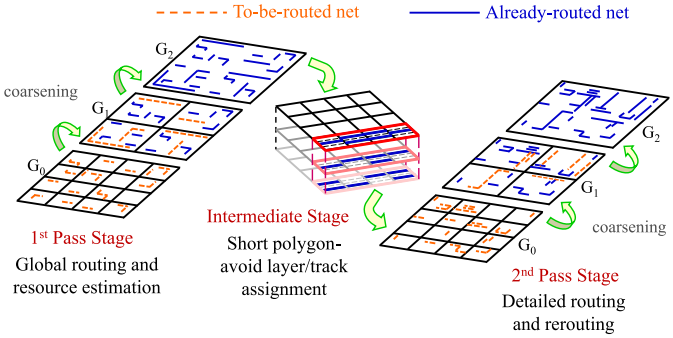


Fig. 6. Two-pass, bottom-up routing framework.

objectives [3], [4], [11]. For local circuit effects (e.g., congestion and via minimization), bottom-up (coarsening) approaches show better optimization capability since the approaches route local nets first [3]. On the other hand, top-down (uncoarsening) manners can handle global electrical effects (e.g., timing and layout uniformity) well, in which longer nets are routed prior to shorter nets [4]. Since the three routing constraints induced by stitching lines are all local effects, a two-pass bottom-up multilevel framework [3] is adopted in this paper. The bottom-up coarsening scheme iteratively groups a set of routing tiles into larger tiles. A net whose pins are in the same tile is referred to as a local net. The router would try to find paths for local nets in the routing tiles to which the nets belong until every local net is tried. Then the coarsening scheme would be performed again to form larger tiles. This routing and coarsening scheme would be repeated until only one tile is left and every net is tried.

Fig. 6 shows our stitch-aware routing framework. The first bottom-up routing pass finds the global route of each local net in the routing graph $G_i$ of level $i$. Then, an intermediate stage of stitch-aware layer/track assignment is performed to facilitate stitch-aware pattern optimization in detailed routing. Finally, pin-to-segment/segment-to-segment detailed routing and failed net rip-up/rerouting are performed in the second bottom-up routing pass.

## III. Stitch-Aware Routing Framework

In this section, stitch-aware global routing, layer/track assignment, and detailed routing are presented in the following subsections.

### A. Stitch-Aware Global Routing

In the global routing stage, a routing plane is first divided into global tiles and transformed into a routing graph, in which a vertex represents a global tile and each pair of adjacent global tiles is connected by an edge, as shown in Fig. 7(a). Then, nets sequentially find their global routing paths on the graph with minimized routing costs. The routing cost of a routing path is usually computed according to the routing congestion on the path.

Resource estimation in global routing for MEBL is quite different from conventional routing problems due to the existence of stitching lines. For example, in Fig. 7(b), the capacity of each boundary (the maximum number of wires that can pass through the boundary) of the global tile is originally six without considering stitching lines. However, the capacities of the top boundary and the bottom boundary are reduced by one since no wire can route on the track occupied by the stitching line due to the vertical routing constraint. Furthermore, it is undesirable that many line ends of vertical segments lie in the same tile. As shown in Fig. 7(b), only two vertical tracks are not in stitch unfriendly regions. If there are three vertical segments whose line ends lie in the tile, at least one line end will lie in the stitch unfriendly region, and the line end may cause a short polygon violation on the connected horizontal wire, as the segment $C$ in Fig. 7(b).

To consider both of the situations, in a global routing graph, each edge is assigned an edge capacity indicating the maximum number of wires that can pass through the tile boundary without overflow, and each vertex is also assigned a vertex capacity denoting the number of tracks not in stitch unfriendly regions. Then, the cost of an edge $e_i$ ($\psi_e(i)$) and the cost of a vertex $v_j$ ($\psi_v(j)$) are respectively defined as follows:

$$\psi_e(i) = 2^{d_e(i)/c_e(i)} - 1 \tag{1}$$

$$\psi_v(j) = 2^{d_v(j)/c_v(j)} - 1 \tag{2}$$

where $c_e(i)$ is the capacity of $e_i$, $c_v(j)$ is the capacity of $v_j$, $d_e(i)$ is the demand of $e_i$, which is the number of segments that have routed on $e_i$, and $d_v(j)$ is the demand of $v_j$, which is the number of line ends that have lain on $v_j$. Thus, for a global routing path $P = (V, E)$ composed of a set $V$ of vertices and a set $E$ of edges, to simultaneously minimize edge and vertex congestions during global routing, the cost of $P$ can be defined as follows:

$$\Psi(P) = \sum_{e_i \in E} \psi_e(i) + \sum_{v_j \in V} \psi_v(j). \tag{3}$$

### B. Stitch-Aware Layer Assignment

Layer assignment [14] and track assignment [1], [7] have been proven as effective intermediate stages between 2-D global routing and detailed routing for improving the routing quality of high complexity designs.

In addition, many manufacturability issues can be optimized during layer assignment and track assignment, such as crosstalk, antenna effect, and wire density uniformity [4], [11], [15], [21]. In this paper, stitch-aware layer assignment and stitch-aware track assignment algorithms

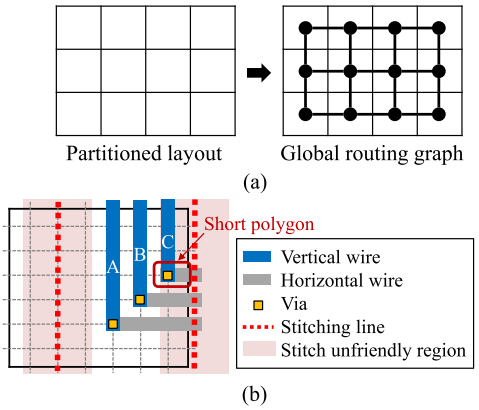


Fig. 7. Global routing model and routing resource estimation for MEBL. (a) Layout is divided into global tiles and transformed into a graph model. (b) Routing resource in the vertical direction is less than that in the horizontal direction due to the stitching lines. In addition, the line end of the segment $C$ lying in the stitch unfriendly region causes a short polygon on the connected horizontal wire.

are also proposed for optimizing stitching line-induced bad patterns.

In layer assignment, we assign the vertical (horizontal) segments in a column (row) panel to different vertical (horizontal) routing layers. A column (row) panel is defined as a column (row) of global tiles in a global routing graph. The conventional objective in layer assignment is to minimize the number of vias without violating the congestion constraints [14]. However, as mentioned in Section III-A, line ends of segments should be also scattered to different layers to avoid generating short polygons.

To solve the stitch-aware layer assignment problem, we first construct a segment conflict graph for each panel, in which a vertex $v_i$ represents a segment $i$ and an edge connecting two vertices if the two segments intersect in some tiles. For a column panel, we set an edge weight $w(v_i, v_j)$ for each edge $(v_i, v_j)$ as follows:

$$w(v_i, v_j) = D_{\text{segment}}(v_i, v_j) + D_{\text{end}}(v_i, v_j) \tag{4}$$

where $D_{\text{segment}}(v_i, v_j)$ is the maximum segment density in the rows where the segment $i$ and the segment $j$ are overlapped, and $D_{\text{end}}(v_i, v_j)$ is the maximum line-end density in the rows where the line ends of $i$ and $j$ are overlapped. [Note that we simply remove the second item in (4)] for row panels since we consider line-end densities only in column panels. Fig. 8(b) shows a conflict graph for the segments in a column panel shown in Fig. 8(a). To uniformly distribute segments and line ends to $k$ layers, the layer assignment problem can be solved by finding a maximum-cut $k$-coloring solution [6] of the segment conflict graph, which is equivalent to finding a $k$-coloring solution with the minimum total edge weight [6]. Fig. 8(c) shows a three-coloring solution with the minimum total edge weight of the segment conflict graph.

Since the maximum-cut $k$-coloring problem is NP-complete [6], previous work has proposed a heuristic approach that first constructs a maximum spanning tree on a conflict graph and then solves the $k$-coloring problem on the tree. Note that a tree is always $k$-colorable when $k \geq 2$. This

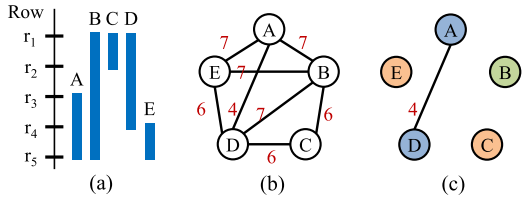


Fig. 8. Layer assignment considering segment and line-end uniformities. (a) Set of segments in a vertical panel. (b) Corresponding segment conflict graph. (c) Layer assignment solution by solving the maximum-cut $k$-coloring problem.

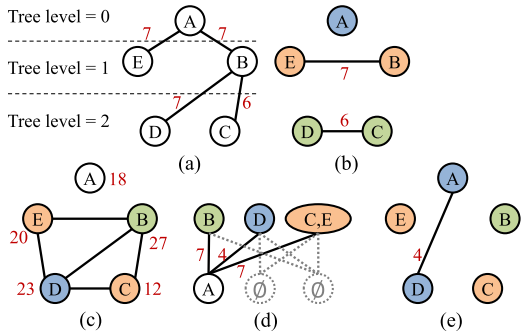


Fig. 9. Heuristics for solving the maximum-cut $k$-coloring problem. (a) and (b) Maximum spanning tree approach. (c)–(e) Our algorithm that can generate a better solution.

heuristic can solve the maximum-cut $k$-coloring problem well as $k$ equals two; however, as $k$ is greater than two, solving the maximum-cut $k$-coloring problem with the maximum spanning tree approach may degrade the solution quality since more edges can be simultaneously considered as more colors are available. As illustrated in Fig. 9(a) and (b), if three vertical layers are available, after constructing a maximum spanning tree and three-coloring the tree according to the tree level of each vertex, a layer assignment solution is generated with total edge weight equal to 13.

In this paper, we propose another heuristic algorithm to get better solutions. We first compute the vertex weight for each vertex by summing the weights of the incident edges. Then, we find a set of $k$-colorable vertices with the maximum total vertex weight. Although this problem is NP-complete in general graphs, it can be solved in polynomial time for segment conflict graphs, which are interval graphs, by using a minimum cost flow algorithm [2]. As shown in Fig. 9(c), $V_1 = \{v_B, v_C, v_D, v_E\}$ is a three-colorable vertex set in the segment conflict graph with the maximum total vertex weight, and $\{v_B\}$, $\{v_D\}$, and $\{v_C, v_E\}$ are the three-coloring groups of $V_1$. The algorithm then finds the next $k$-colorable vertex set with the maximum total vertex weight on the remaining graph. To merge the coloring groups of the two vertex sets, a perfect bipartite matching algorithm is applied to minimize the total conflict edge weight. As illustrated in Fig. 9(d), two pseudo coloring groups $\emptyset$ are first created since only the vertex $v_A$ remains, and thus the three-coloring groups of the second vertex set $V_2$ are $\{v_A\}$, $\emptyset$ and $\emptyset$. To combine the coloring groups of $V_1$ and $V_2$, a complete bipartite graph is constructed and the edge weights are set as the total conflict edge weight between two groups. By solving the minimum weight perfect bipartite matching problem, coloring groups are merged with the minimum conflict edge weight. The above process is performed iteratively until no vertex is left. Fig. 9(e) shows the layer assignment result with smaller total edge weight equal to 4. After obtaining coloring groups, to complete the layer assignment, we have to assign coloring groups to different layers. To minimize the number of vias, we adopt the assignment method proposed by [4] to make coloring groups with more segments of the same nets assigned to closer layers.

### C. Short Polygon-Avoid Track Assignment

In track assignment, segments of the same layer in a panel are assigned exact track numbers, which is a crucial stage for short polygon avoidance. A desired track assignment solution which can avoid short polygon generation is a track assignment without bad ends. A bad end is a line end of a vertical wire segment lying in the stitch unfriendly region of a stitching line, and the connected horizontal wire is cut by the stitching line. For example, the lower end of the wire segment $C$ in Fig. 7(b) is a bad end. To derive a track assignment solution without bad ends, an ILP-based algorithm and a graph-based algorithm are proposed, which are detailed in the following subsections. Note that the short polygon-avoiding track assignment algorithms are only applied to column panels. Segments in row panels can be assigned by using conventional layer assignment algorithms.

*1) ILP-Based Approach:* First, the short polygon-avoiding track assignment problem can be intuitively transformed into a multicommodity flow model, which is a directed graph $G = (V, E)$. Fig. 10(a) shows a track assignment instance. To find an exact track number for the segment $A$, for example, the multicommodity flow graph model is constructed as shown in Fig. 10(b), where a track vertex represents a track in a global tile, a forbidden vertex is a track occupied by a stitching line, and the source vertex $s_A$ and the target vertex $t_A$ are the top end and the bottom end of the segment $A$. The source edges connect $s_A$ to the track vertices of the tile where the top end of $A$ lies. Similarly, the target edges connect the track vertices of the tile where the bottom end of $A$ lies to $t_A$. A source/target edge is removed if the line end becomes a bad end on the corresponding track. For example, $s_A$ causes a bad end if it starts on the second track, and thus the second source edge is removed from the graph. Also, track vertices of adjacent tiles are connected with track edges, and the edge weight of a track edge is set to be the difference of the track numbers of the two track vertices to minimize wirelength and the number of routing bends. The whole multicommodity flow graph model of the four segments is shown in Fig. 10(c). Then, we find a track assignment solution with an ILP formulation. The notation used in our ILP formulation is listed as follows.

1) $K$: A set of segments in a track assignment problem.
2) $s(k)$: The source vertex of the segment $k$.
3) $t(k)$: The target vertex of the segment $k$.
4) $V_{\text{track}}$: A set of track vertices.
5) $w(u, v)$: The weight of the directed edge $(u, v)$.

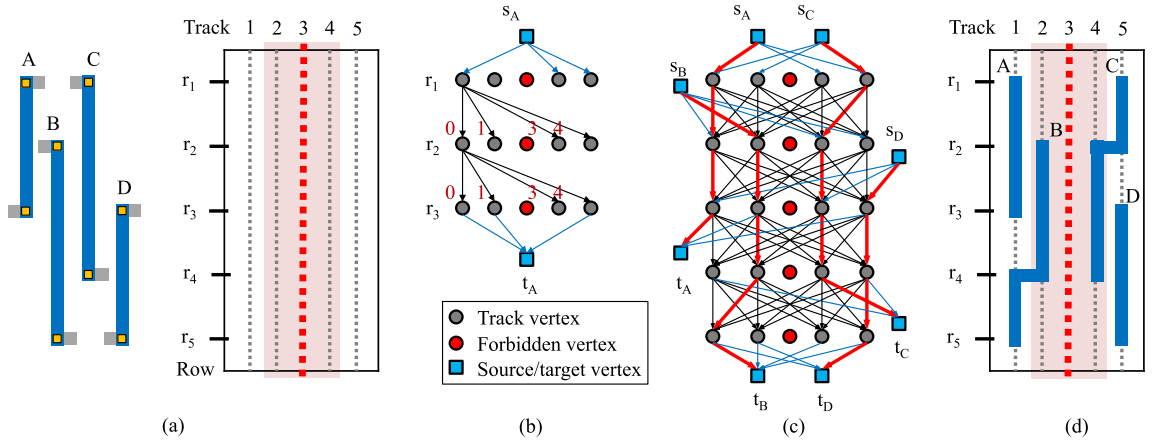


Fig. 10. ILP-based track assignment approach. (a) Track assignment instance. (b) Multicommodity flow model of the segment $A$. (c) Multicommodity flow model of all segments and the solution derived from the ILP formulation. (d) Corresponding track assignment solution.

6) $f_k(u, v)$: 0–1 integer variable that denotes if the segment $k$ is routed through the directed edge $(u, v)$.
7) $\mathcal{C}$: A set of crossed edge pairs.

Based on the notations, the short polygon-avoiding track assignment problem can be formulated as follows:

$$\text{minimize} \quad \sum_{(u,v)\in E}\left(w(u,v)\times\sum_{k\in K}f_k(u,v)\right)$$

subject to

$$\sum_{(s(k),v)\in E} f_k(s(k),v)=1,\forall k\in K \quad (5)$$

$$\sum_{(u,t(k))\in E} f_k(u,t(k))=1,\forall k\in K \quad (6)$$

$$\sum_{u\in V} f_k(u,v)=\sum_{w\in V} f_k(v,w),\forall k\in K,\forall v\in V_{\text{track}} \quad (7)$$

$$\sum_{u\in V}\sum_{k\in K} f_k(u,v)\le 1,\forall v\in V_{\text{track}} \quad (8)$$

$$\sum_{k\in K} f_k(u_1,v_1)+\sum_{k\in K} f_k(u_2,v_2)\le 1 \quad \forall((u_1,v_1),(u_2,v_2))\in\mathcal{C}. \quad (9)$$

The objective of the ILP formulation is to minimize the total edge weight of a flow solution such that the wirelength and the number of wire bends can be minimized. Constraints (5) and (6) ensure that each segment can find a unique path from its source vertex to the target vertex. Constraint (7) is used to guarantee that the number of paths flowing into a node equals that draining from the node. Constraint (8) guarantees that a track in a tile is occupied by at most one segment. Finally, (9) prevents segments from crossing with each other. Let $T$ be the number of tracks in a column panel and let $R$ be the number of rows in the global routing graph, the number of ILP variables is $O(T^2R)$ and the number of ILP constraints is $O(TR|K| + T^4R)$, which is dominated by (7) and (9).

Using "doglegs" to avoid short polygon generation is one of the advantage of the ILP-based approach. However, since the short polygon-avoiding track assignment process is performed for every panel in all vertical layers, the runtime of iteratively solving the ILP formulation may be prohibitively long as the chip size increases. Therefore, we propose another graph-based track assignment heuristic, which can efficiently utilize doglegs for short polygon avoidance.

*2) Graph-Based Approach:* The graph-based short polygon-avoiding track assignment algorithm first determines the segment order in a panel, and then tries to resolve bad ends with doglegs by using a graph-based algorithm.

The approach starts from assigning longer segments next to stitching lines. As shown in Fig. 11(a) and (b), the segments $B$, $C$, and $E$ are placed adjacent to stitching lines. Longer segments have larger flexibility to avoid short polygon generation by applying doglegs. Then, some bad ends of those longer segments will be generated if the segments do not change their track numbers. For example, the bottom end of $B$, the bottom end of $C$, and the top end of $E$ are currently bad ends. After that, segments not overlapped with the bad ends are assigned next to those longer segments such that the bad ends can be easily resolved with doglegs. Therefore, as illustrated in Fig. 11(b), the segment $A$ is assigned next to $B$ and the segment $D$ is assigned next to $E$. For the remaining segments having less impact on bad ends, the track numbers are arbitrarily assigned.

After determining the segment order, doglegs are used to resolve bad ends. A set of segments between two stitching lines are considered at a time. Each segment is first divided into intervals according to global tiles, as the segments $C, D$, and $E$ shown in Fig. 11(c). Then, two constraint graphs are constructed to record the geometry relationship among these intervals. As illustrated in Fig. 11(d), the first one is the minimum track constraint graph, where a vertex represents an interval and an directed edge $(v_i, v_j)$ indicates that the two intervals are overlapped in the $x$-direction and the interval $i$ is left to the interval $j$. A dummy vertex $d$ is created and connected to a vertex $v_i$ if the interval $i$ should not be assigned to the leftmost track. For example, the interval $c_3$ has a bad end if it is assigned to the leftmost track between the two stitching lines, and thus a dummy vertex is created and connected to $v_{c_3}$ through the edge $(d, v_{c_3})$ in the minimum track constraint graph. After creating a source vertex $s$ and connecting $s$ to the vertices of the leftmost intervals and the dummy

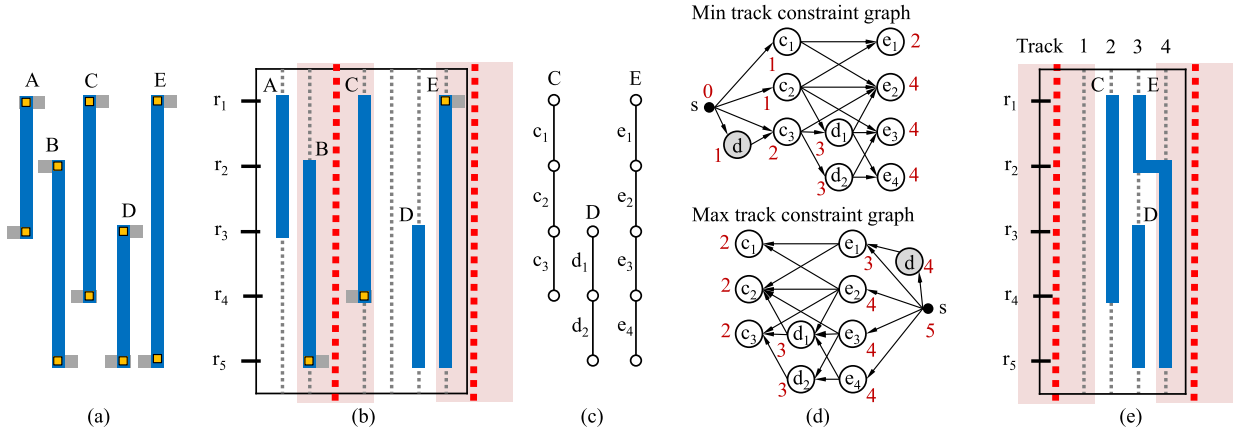


Fig. 11. Graph-based track assignment approach. (a) Track assignment instance. (b) Segment order is first determined. (c) Segments $C, D$, and $E$ between two stitching lines are simultaneously considered and are divided into intervals. (d) Feasible track assignment solution space of each interval is computed by using the minimum and maximum track assignment constraint graphs. (e) Final track assignment solution of $C, D$, and $E$.

vertices, a longest path algorithm is applied to compute the minimum track number $m$ of each interval, which indicates the leftmost feasible track number. The weight of every edge is one, except for the edge between the source vertex and the dummy vertex. The weight of the edge between the source vertex and the dummy vertex is the number of vertical tracks in the stitch unfriendly region on one side of a stitching line. In Fig. 11, each stitch unfriendly region contains one vertical track on one side of a stitching line, and thus the weight of the edge between the source vertex and the dummy vertex is one. For the maximum track constraint graph, the construction is almost the same except that an edge $(v_i, v_j)$ connects the vertex of the interval $i$ right to the vertex of the interval $j$ and a dummy vertex connected to a vertex $v_i$ if the interval $i$ should not be assigned to the rightmost track. A similar algorithm is applied to compute the maximum track number $M$ of each interval. As shown in Fig. 11(d), the two numbers of a vertex in the minimum and maximum track constraint graphs give a feasible solution space $[m, M]$ of track assignment for each corresponding interval.

Finally, we sequentially determine the track numbers from the leftmost segment to the rightmost segment between the two stitching lines according to their feasible solution spaces. The wirelength and the number of bends of each segment are greedily optimized during track assignment. For example, all intervals of the segment $C$ are assigned to the second track for wirelength and bend optimizations, and the final assignment solution is shown in Fig. 11(e).

### D. Stitch-Aware Detailed Routing

The final stage of our routing framework is stitch-aware detailed routing, which finds pin-to-segment and segment-to-segment detailed routes based on a conventional A$^\star$-search routing algorithm [10]. To illustrate the direction of the routing path more easily, we define the $x$-direction and $y$-direction as the horizontal direction and vertical direction, respectively. And routing in the $z$-direction is defined as routing from one layer to a neighboring layer. To satisfy the via and vertical routing constraints, wires passing through stitching lines can only route in the $x$-direction. For minimizing the number of generated short polygons, the stitch-aware weighted cost of routing grids and the stitch-aware net ordering play important roles.

*1) Stitch-Aware Weighted Cost:* The cost of a routing grid is adjusted to minimize the number of generated short polygons when A$^\star$-search is performed. Observed that the generation of short polygons is due to the shortage of routing resources in the region near stitching lines, we define the four tracks that are nearest to a stitching line as escape region. Obviously, for any path crossing the stitching line, if the escape region is not occupied by other routed nets, the path does not generate any short polygon. Therefore, to reserve the routing resources of the escape region for pathes that are potential short polygon generators, we add escape cost to grids in the escape region. Fig. 12 illustrates the effect of the escape cost. There are two pairs of pins, $(A_1, A_2)$ and $(B_1, B_2)$, need to be connected. In Fig. 12(b), where escape cost is not considered, $(A_1, A_2)$ is routed in the escape region. As a result, $(B_1, B_2)$ has to be routed in the $z$-direction in the stitch unfriendly region, and thus a short polygon is generated. On the contrary, in Fig. 12(c), where the escape cost is considered, $(A1, A2)$ has a detour and is not routed in the escape region. Therefore, the routing path of $(B1, B2)$ can be routed in the $z$-direction outside the stitch unfriendly region, and thus no short polygon is generated. In addition to the escape cost, we give a large cost, named via in stitch unfriendly region cost, if a wire in a stitch unfriendly region is routed in the $z$-direction. Therefore, a detailed path with the minimum number of line ends lying in stitch unfriendly regions will be found. As illustrated in Fig. 13, a detailed route is constructed from the source point $s$ to the target point $t$ and no short polygon is generated.

Considering a path from a routing grid $i$ to a neighboring routing grid $j$, the detailed router computes the routing cost $C_{\text{grid}}(j)$ of grid $j$ as follows:

$$C_{\text{grid}}(j) = C_{\text{grid}}(i) + \alpha C_{\text{wl}}(i,j) + \beta C_{\text{vsu}}(i,j) + \gamma C_{\text{esc}}(j) \tag{10}$$

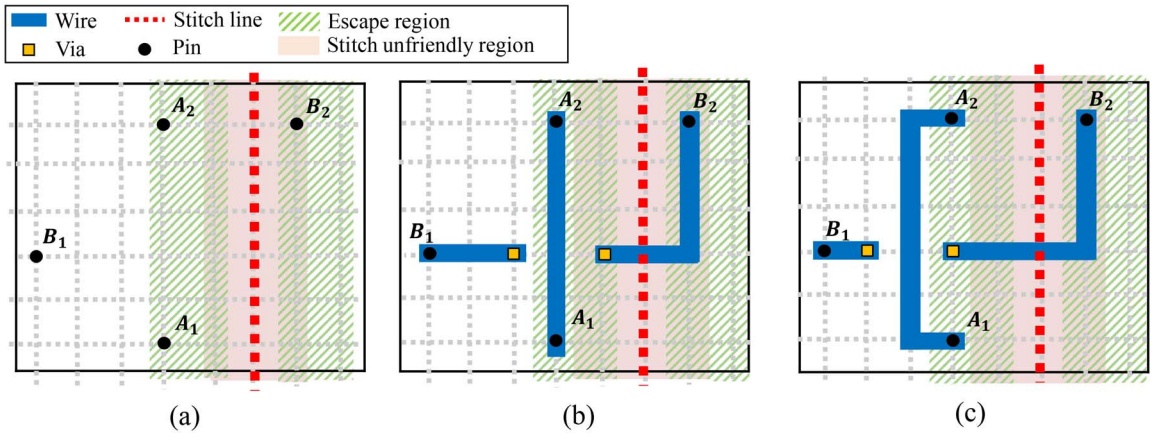


Fig. 12. (a) Pin $A_1$ should be connected to pin $A_2$; pin $B_1$ should be connected to pin $B_2$. (b) Routing without escape cost. The routing path of $(A_1, A_2)$ occupies the escape region of $(B_1, B_2)$, and thus the routing path of $(B_1, B_2)$ generates a short polygon. (c) Routing with escape cost. $(A_1, A_2)$ is not routed in the escape region, and thus the routing path of $(B_1, B_2)$ does not generate any short polygon.

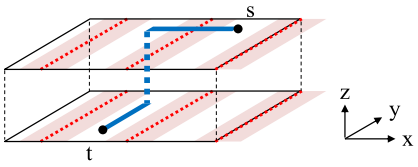


Fig. 13. Stitch-aware detailed routing. A wire can only route in the $x$-direction as passing through stitching lines. In addition, a detailed route with the minimum number of line ends lying in stitch unfriendly regions will be found.

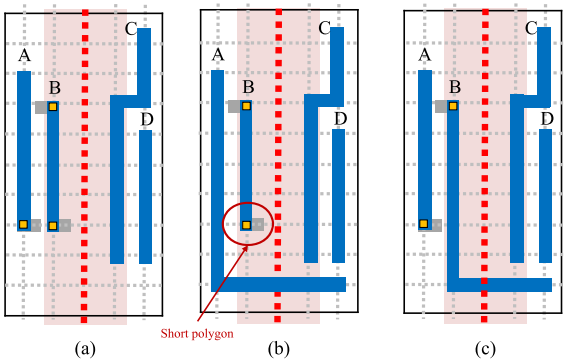


Fig. 14. (a) Lower end of segment $A$ and segment $B$ should be connected to the pins or segments in the right panel. (b) Segment $A$ is routed first, and thus a short polygon is generated in the lower end of segment $B$. (c) Segment $B$ is routed first, and no short polygon is generated.

where $C_{\text{wl}}(i, j)$ is the extra wirelength caused by the path from grid $i$ to grid $j$, $C_{\text{vsu}}(i, j)$ is the via in the stitch unfriendly region cost, $C_{\text{esc}}(j)$ is the escape cost, and $\alpha$, $\beta$, and $\gamma$ are user defined parameters. Note that, to minimize the number of short polygons, $\beta$ is set to be much larger than $\gamma$.

*2) Stitch-Aware Net Ordering:* For panels whose segment densities are equal to the number of tracks, bad ends are inevitable. Therefore, to reduce the number of short polygons generated by bad ends, the segments with bad ends should be given sufficient routing resources during detailed routing, and thus the nets with more bad ends are given higher priority in the order for detailed routing. Fig. 14 illustrates the importance of the stitch-aware net ordering. Fig. 14(a) shows a result of the track assignment, where the lower end of segment $A$ and and the lower end of segment $B$, which is a bad end, should be connected to pins or segments in the right panel. If the net with segment $A$ is routed first, obviously, we do not have enough routing resources to connect the lower end of segment $B$ to pins or segments in the right panel without generating a short polygon, as shown in Fig. 14(b). In contrast, in Fig. 14(c), segment $B$ is routed prior to segment $A$. Because there are enough routing resources, the lower end of segment B is connected to the right panel by a horizontal path in the same layer. Therefore, no short polygon is generated.

TABLE I
MCNC BENCHMARK CIRCUITS

| Circuit | Size ($\mu m^2$) | #Layers | #Nets | #Pins |
|---|---|---|---|---|
| Struct | 4903×4904 | 3 | 1920 | 5471 |
| Primary1 | 7522×4988 | 3 | 904 | 2941 |
| Primary2 | 10438×6488 | 3 | 3029 | 11226 |
| S5378 | 435×239 | 3 | 1694 | 4818 |
| S9234 | 404×225 | 3 | 1486 | 4260 |
| S13207 | 660×365 | 3 | 3781 | 10776 |
| S15850 | 705×389 | 3 | 4472 | 12793 |
| S38417 | 1144×619 | 3 | 11309 | 32344 |
| S38584 | 1295×672 | 3 | 14754 | 42931 |

TABLE II
FARADAY BENCHMARK CIRCUITS

| Circuit | Size ($\mu m^2$) | #Layers | #Nets | #Pins |
|---|---|---|---|---|
| DMA | 408.4×408.4 | 6 | 13256 | 73982 |
| DSP1 | 706×706 | 6 | 28447 | 144872 |
| DSP2 | 642.8×642.8 | 6 | 28431 | 144703 |
| RISC1 | 1003.6×1003.6 | 6 | 34034 | 196677 |
| RISC2 | 959.6×959.6 | 6 | 34034 | 196670 |

## IV. EXPERIMENTAL RESULTS

Our algorithm was implemented in the C++ programming language on a 2.93 GHz Linux workstation with 48 GB memory. The minimum cost flow problem and the minimum weight bipartite matching problem in layer assignment

TABLE III
COMPARISON OF THE PROPOSED ROUTER WITH THE BASELINE ROUTER

| Circuit | Baseline | | | | Stitch-aware routing framework | | | |
|---|---|---|---|---|---|---|---|---|
| | Rout. (%) | #VV | #SP | CPU (s) | Rout. (%) | #VV | #SP | CPU (s) |
| Struct | 100.00 | 418 | 1216 | 1 | 100.00 | 418 | 0 | 1 |
| Primary1 | 100.00 | 226 | 869 | 1 | 100.00 | 226 | 0 | 1 |
| Primary2 | 99.90 | 808 | 3207 | 5 | 100.00 | 811 | 0 | 4 |
| S5378 | 96.77 | 923 | 351 | 1 | 98.21 | 838 | 8 | 1 |
| S9234 | 98.99 | 755 | 261 | 1 | 98.56 | 686 | 7 | 1 |
| S13207 | 98.16 | 46 | 726 | 2 | 99.24 | 59 | 61 | 2 |
| S15850 | 97.68 | 51 | 981 | 2 | 99.08 | 68 | 58 | 3 |
| S38417 | 98.63 | 24 | 2354 | 5 | 99.08 | 35 | 122 | 6 |
| S38584 | 98.50 | 67 | 3221 | 11 | 98.81 | 96 | 156 | 12 |
| Dma | 97.13 | 1219 | 1904 | 7 | 99.17 | 1293 | 4 | 7 |
| Dsp1 | 97.51 | 1769 | 1720 | 6 | 99.61 | 1826 | 2 | 7 |
| Dsp2 | 97.59 | 2043 | 1818 | 5 | 99.59 | 2108 | 49 | 6 |
| Risc1 | 96.87 | 2301 | 3186 | 11 | 99.48 | 2418 | 10 | 14 |
| Risc2 | 97.27 | 2240 | 3073 | 10 | 99.51 | 2381 | 2 | 11 |
| Comp. | 1.000 | −* | 1.000 | 1.0 | 1.011 | −* | 0.023 | 1.1 |

TABLE IV
EXPERIMENTAL RESULTS THAT SHOW THE EFFECTIVENESS AND EFFICIENCY OF THE STITCH-AWARE GLOBAL ROUTING ALGORITHM

| Circuit | w/o line end consideration | | | | w/ line end consideration | | | |
|---|---|---|---|---|---|---|---|---|
| | TVOF | MVOF | WL | CPU (s) | TVOF | MVOF | WL | CPU (s) |
| S5378 | 28 | 3 | 7550 | 0.110 | 0 | 0 | 7594 | 0.111 |
| S9234 | 37 | 4 | 6468 | 0.109 | 0 | 0 | 6535 | 0.110 |
| S13207 | 251 | 8 | 16193 | 0.264 | 0 | 0 | 16561 | 0.226 |
| S15850 | 231 | 6 | 19120 | 0.338 | 1 | 1 | 19406 | 0.340 |
| S38417 | 445 | 6 | 48484 | 1.060 | 0 | 0 | 49227 | 1.070 |
| S38584 | 820 | 6 | 64175 | 4.700 | 0 | 0 | 65445 | 4.710 |
| Comp. | 1.000 | 1.000 | 1.000 | 1.000 | 0.001 | 0.028 | 1.015 | 1.007 |

TABLE V
CHARACTERISTICS OF THE LAYER ASSIGNMENT INSTANCES

| #Instance | Segment density | | Line end density | |
|---|---|---|---|---|
| | Max | Avg. | Max | Avg. |
| 50 | 11.68 | 5.72 | 6.06 | 2.00 |

TABLE VI
COMPARISON OF LAYER ASSIGNMENT RESULTS BETWEEN TWO HEURISTIC ALGORITHMS PROPOSED IN [4] AND IN THIS PAPER

| Heuristic | #Layers | | | |
|---|---|---|---|---|
| | 2 | 3 | 4 | 5 |
| Max. Spanning Tree [4] | 1371.56 | 965.10 | 760.48 | 681.98 |
| Ours | 1181.42 | 672.60 | 421.72 | 276.98 |
| Improvement | 13.86% | 30.31% | 44.55% | 59.39% |

are solved by adopting the LEDA package [23], and the ILP formulation in track assignment is solved by using the CPLEX12.3 library [22]. Two suites of benchmarks, the MCNC benchmarks and the real industry Faraday benchmarks, were used. Tables I and II list the information of the benchmarks. To reflect the sub−20 nm technology nodes, the minimum feature sizes of the MCNC benchmarks and Faraday benchmarks were shrunk to 36 and 32 nm, respectively. In our routing framework, the distance between two stitching lines was set to be 15 times the width of routing pitch, and the stitching lines are uniformly distributed in a layout. In addition, the tracks adjacent to stitching lines fall into stitch unfriendly regions.

### A. Comparison With Baseline Router

We first compare our stitch-aware router with a baseline router, which uses the global routing results generated from NTUgr [5] and performs layer assignment, track assignment, and detailed routing with conventional routing objectives such as wirelength and routability. To fix routing violations caused by the segments assigned to the tracks on stitching lines in the track assignment stage, we simply rip up those segments and directly route the corresponding nets in detailed routing. In addition, wires in the baseline router can only route in the $x$-direction on stitching lines during detailed routing to avoid via and routing violations. Thus, the baseline router can also generate routing results without any routing violation and with the same number of via violations as our stitch-aware router if they both have 100% routability.

The comparison of our router with the baseline router is shown in Table III, where "Rout." gives the routability, "#VV" reports the number of via violations, and "CPU" lists the runtime in second. Routability is defined as the ratio of the number of successfully routed nets to the number of total nets, that is, Routability $=$ # routed nets/# total nets $\times 100\%$. The user-defined parameters in (10) are set as follows: $\alpha = 1$, $\beta = 10$, and $\gamma = 5$. Compared with the baseline router, our router dramatically reduces the number of short polygons, slightly improves the routability, and has 10% runtime overhead. The baseline router has worse routability because many nets whose segments are ripped up during the track assignment stage are all directly routed in detailed routing. Those nets suffer from higher failure rate due to the lack of wire planning. As a

TABLE VII
EXPERIMENTAL RESULTS THAT COMPARE THE EFFECTIVENESS AND EFFICIENCY AMONG DIFFERENT TRACK ASSIGNMENT ALGORITHMS

| Circuit | w/o stitch consideration | | | | ILP-based Approach | | | | Graph-based Approach | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Rout. (%) | #VV | #SP | CPU (s) | Rout. (%) | #VV | #SP | CPU (s) | Rout. (%) | #VV | #SP | CPU (s) |
| Struct | 100.00 | 418 | 1202 | 1 | 100.00 | 418 | 0 | 553 | 100.00 | 418 | 0 | 1 |
| Primary1 | 100.00 | 226 | 863 | 1 | 100.00 | 226 | 1 | 11129 | 100.00 | 226 | 0 | 1 |
| Primary2 | 99.89 | 805 | 3149 | 5 | 100.00 | 811 | 2 | 76144 | 100.00 | 811 | 0 | 4 |
| S5378 | 96.74 | 924 | 299 | 1 | 97.98 | 831 | 6 | 3027 | 98.21 | 838 | 8 | 1 |
| S9234 | 98.99 | 755 | 228 | 1 | 98.50 | 686 | 2 | 2230 | 98.56 | 686 | 7 | 1 |
| S13207 | 98.17 | 46 | 641 | 2 | 99.40 | 59 | 84 | 5797 | 99.24 | 59 | 61 | 2 |
| S15850 | 97.68 | 51 | 863 | 2 | 99.39 | 68 | 45 | 9687 | 99.08 | 68 | 58 | 3 |
| S38417 | 98.62 | 24 | 2063 | 5 | NA | NA | NA | $>100000$ | 99.08 | 35 | 122 | 6 |
| S38584 | 98.50 | 67 | 2892 | 11 | NA | NA | NA | $>100000$ | 98.81 | 96 | 156 | 12 |
| Dma | 97.03 | 1212 | 1751 | 7 | 99.20 | 1293 | 6 | 1687 | 99.17 | 1293 | 4 | 7 |
| Dsp1 | 97.46 | 1759 | 1602 | 6 | 99.68 | 1824 | 5 | 2473 | 99.61 | 1826 | 2 | 7 |
| Dsp2 | 97.45 | 2022 | 1703 | 5 | 99.76 | 2114 | 6 | 2089 | 99.59 | 2108 | 49 | 6 |
| Risc1 | 96.83 | 2295 | 2949 | 11 | 99.49 | 2403 | 9 | 5423 | 99.48 | 2418 | 10 | 14 |
| Risc2 | 97.18 | 2220 | 2846 | 10 | 99.57 | 2353 | 8 | 4810 | 99.51 | 2381 | 2 | 11 |
| Comp. | 1.000 | −* | 1.000 | 1.0 | 1.015 | −* | 0.019 | 3623.3 | 1.012 | −* | 0.026 | 1.1 |

TABLE VIII
EXPERIMENTAL RESULTS THAT SHOW THE EFFECTIVENESS AND EFFICIENCY OF THE STITCH-AWARE DETAILED ROUTING

| Circuit | w/o stitch consideration | | | | w/ stitch consideration | | | |
|---|---|---|---|---|---|---|---|---|
| | Rout. (%) | #VV | #SP | CPU (s) | Rout. (%) | #VV | #SP | CPU (s) |
| Struct | 100.00 | 418 | 1 | 1 | 100.00 | 418 | 0 | 1 |
| Primary1 | 100.00 | 226 | 0 | 1 | 100.00 | 226 | 0 | 1 |
| Primary2 | 100.00 | 811 | 0 | 4 | 100.00 | 811 | 0 | 4 |
| S5378 | 98.11 | 833 | 61 | 1 | 98.21 | 838 | 8 | 1 |
| S9234 | 98.56 | 686 | 28 | 1 | 98.56 | 686 | 7 | 1 |
| S13207 | 99.71 | 59 | 195 | 2 | 99.24 | 59 | 61 | 2 |
| S15850 | 99.51 | 68 | 229 | 3 | 99.08 | 68 | 58 | 3 |
| S38417 | 99.44 | 36 | 427 | 6 | 99.08 | 35 | 122 | 6 |
| S38584 | 99.18 | 96 | 572 | 12 | 98.81 | 96 | 156 | 12 |
| Dma | 99.25 | 1289 | 37 | 7 | 99.17 | 1293 | 4 | 7 |
| Dsp1 | 99.66 | 1826 | 13 | 6 | 99.61 | 1826 | 2 | 7 |
| Dsp2 | 99.65 | 2108 | 62 | 6 | 99.59 | 2108 | 49 | 6 |
| Risc1 | 99.55 | 2417 | 53 | 13 | 99.48 | 2418 | 10 | 14 |
| Risc2 | 99.89 | 2379 | 34 | 10 | 99.51 | 2381 | 2 | 11 |
| Comp. | 1.000 | −* | 1.000 | 1.00 | 0.998 | −* | 0.200 | 1.02 |

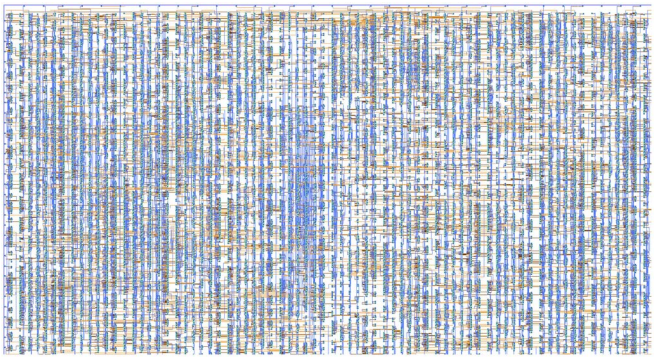

Fig. 15. Routing result of the circuit S38417.

result, the baseline router generates fewer via violations in some cases because the via violations of those failed nets are not counted.

### B. Comparison of Global Routing Algorithms

To show the effectiveness and efficiency of the stitch-aware routing algorithms in each stage, we first conducted an experiment to compare the proposed global routing approach that considers the line-end densities of global tiles with the approach that considers only wire densities. We define "vertex overflow" to be the number of line ends in a global tile that exceeds the line-end capacity of the global tile (i.e., the number of tracks not in stitch unfriendly regions). The experimental results are shown in Table IV, where "TVOF" gives the total vertex overflow, "MVOF" shows the maximum vertex overflow among all global tiles, "WL" reports the total wire length, and CPU lists runtime in second. Note that we only report results of six "hard" benchmarks because both approaches have no vertex overflow in other benchmarks. The results show that our global routing approach with line-end consideration achieves zero vertex overflows in most of the benchmarks, with only 1.5% wirelength overhead.

### C. Comparison of Layer Assignment Algorithms

In the second experiment, we compare the two heuristic algorithms solving the maximum-cut $k$-coloring problem on a segment conflict graph in layer assignment. The first algorithm proposed by [4] solves the problem by finding a maximum spanning tree on the conflict graph. The second one we proposed is based on iteratively finding a set of $k$-colorable vertices of maximum total weight. We let the cost of a layer assignment solution be the total conflict edge weight in the

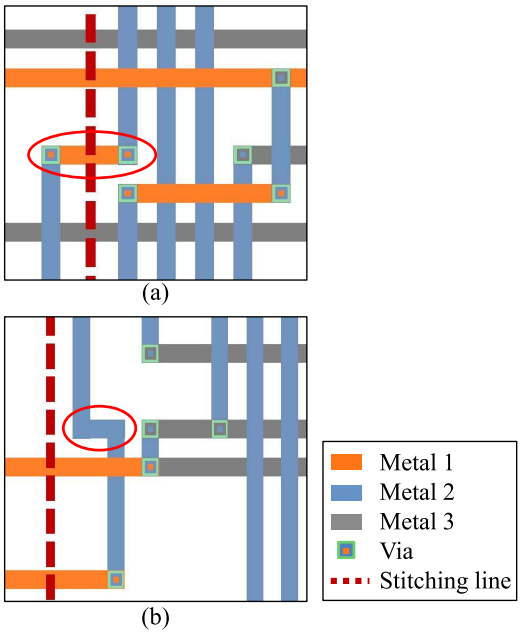


Fig. 16. Short-polygon avoiding by considering stitching lines. (a) Without stitching line consideration, the short polygons are generated. (b) By using the dogleg in track assignment, no short polygon is generated.

$k$-coloring solution, and thus a smaller cost indicates a better solution. In the experiment, 50 layer assignment instances with the same numbers of intervals and global tiles were randomly generated. The average line-end density and the average segment density of the 50 layer assignment instances are listed in Table V. We ran the two algorithms on the layer assignment instances by setting the number of available vertical layers from 2 to 5. Table VI shows the average layer assignment cost derived from the two algorithms. Compared to the maximum spanning tree method, our algorithm generates better layer assignment results no matter how many routing layers with the same preferred direction are available. Furthermore, the improvement becomes more significant as the number of available layers increases, which shows that our algorithm is better for the current IC designs whose numbers of routing layers increase with design complexity.

### D. Comparison of Track Assignment Algorithms

We also show the effectiveness of avoiding short polygon generation by applying three different track assignment approaches: 1) track assignment without considering stitching lines (track assignment in the baseline router); 2) track assignment by solving the ILP formulation; and 3) track assignment by applying the graph-based algorithm. Note that the same stitch-aware algorithms in other routing stages are used for fair comparison. The experimental results are shown in Table VII. Track assignment without considering stitching lines causes worse routability due to the same reason as we explained in Section IV-A. Also, the three approaches have similar numbers of via violations due to fixed pins. On the other hand, the results show that by considering stitching lines, both the ILP-based approach and the graph-based approach can effectively reduce the number of short polygon violations by more than 97% and slightly improve the routability. Since the ILP-based approach is too time-consuming to generate a routing solution in a reasonable runtime, the graph-based approach is more appropriate for large scale routing instances.

### E. Comparison of Detailed Routing Algorithms

Finally, we show the effectiveness of the proposed stitch-aware detailed routing algorithm by comparing: 1) detailed routing without stitch consideration and 2) stitch-aware detailed routing, on the results of graph-based track assignment. The results in Table VIII show that, compared with detailed routing without stitch consideration, our stitch-aware detailed routing approach reduces the number of short polygons by 80% with only 0.2% loss of routability. Therefore, the stitch-aware routing is a critical stage to further reduce the number of short polygons after stitch-aware track assignment.

The routing result of the circuit S38417 by applying our stitch-aware routing is shown in Fig. 15. In addition, Fig. 16 shows the local views of routing wires. As shown in Fig. 16(a), routing without considering stitching lines can cause short polygons. On the other hand, by applying our short polygon-avoiding track assignment algorithms, the number of short polygons are dramatically reduced by using doglegs, as the example shown in Fig. 16(b).

## V. Conclusion

In this paper, we propose the first work of stitch-aware routing framework for MEBL. We first identify three types of stitching line-induced bad patterns which could cause severe pattern distortion, electrical variation, or even yield loss. Then, we provide solutions to avoid generating these bad patterns during each routing stage. Experimental results show that our algorithms can efficiently and effectively reduce the number of short polygons. In addition, to remove the via violations due to the fixed pin positions of nets, stitch-aware algorithms should be also desirable in the placement stage. Thus, stitch-aware placement could be our future work to further improve the manufacturability and facilitate the development of MEBL.

## References

[1] S. Batterywala, N. Shenoy, W. Nicholls, and H. Zhou, "Track assignment: A desirable intermediate step between global routing and detailed routing," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, 2002, pp. 59–66.

[2] M. C. Carlisle and E. L. Lioyd, "On the k-coloring of intervals," *Discrete Appl. Math.*, vol. 59, no. 3, pp. 225–235, 1995.

[3] H.-Y. Chen, M.-F. Chiang, Y.-W. Chang, L. Chen, and B. Han, "Full-chip routing considering double-via insertion," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 5, pp. 844–857, May 2008.

[4] H.-Y. Chen, S.-J. Chou, S.-L. Wang, and Y.-W. Chang, "A novel wire-density-driven full-chip routing system for CMP variation control," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 2, pp. 193–206, Feb. 2009.

[5] H.-Y. Chen, C.-H. Hsu, and Y.-W. Chang, "High-performance global routing with fast overflow reduction," in *Proc. 2009 IEEE/ACM Asia South Pac. Design Autom. Conf.*, Yokohama, Japan, pp. 582–587.

[6] J.-D. Cho, S. Raje, and M. Sarrafzadeh, "Fast approximation algorithms on maxcut, k-coloring, and k-color ordering for VLSI applications," *IEEE Trans. Comput.*, vol. 47, no. 11, pp. 1253–1266, Nov. 1998.

[7] J. Cong, J. Fang, and K. Y. Khoo, "DUEN—A multilayer gridless routing system," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 5, pp. 633–647, May 2001.

[8] S. Y. Fang, I. J. Liu, and Y. W. Chang, "Stitch-aware routing for multiple e-beam lithography," in *Proc. 2013 ACM/IEEE Design Autom. Conf.*, Austin, TX, USA, pp. 1–6.
[9] E. A. Hakkennes *et al.*, "Demonstration of real time pattern correction for high throughput maskless lithography," *Proc. SPIE*, vol. 7970, Apr. 2011, Art. ID 79701A.
[10] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.
[11] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D.-T. Lee, "Crosstalk- and performance-driven multilevel full-chip routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 24, no. 6, pp. 869–878, Jun. 2005.
[12] D. Hung, O. Meijer, and A. Zepka, "Bottlenecks in data preparation flow for multi-beam direct write," *Proc. SPIE*, vol. 8166, Oct. 2011, Art. ID 81662C.
[13] C. Klein *et al.*, "PML2: The maskless multibeam solution for the 22nm node and beyond," *Proc. SPIE*, vol. 7271, Mar. 2009, Art. ID 72710N.
[14] T.-H. Lee and T.-C. Wang, "Congestion-constrained layer assignment for via minimization in global routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 9, pp. 1643–1656, Sep. 2008.
[15] T.-H. Lee and T.-C. Wang, "Simultaneous antenna avoidance and via optimization in layer assignment of multi-layer global routing," in *Proc. 2010 IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, pp. 312–318.
[16] B. J. Lin, "Future of multiple-e-beam direct-write systems," *Proc. SPIE*, vol. 8323, Mar. 2012, Art. ID 832302.
[17] M. A. McChord *et al.*, "REBL: Design progress toward 16 nm half-pitch maskless projection electron beam lithography," *Proc. SPIE*, vol. 8323, Mar. 2012, Art. ID 832311.
[18] S. Rizvi, *Handbook of Photomask Manufacturing Technology*. Boca Raton, FL, USA: Taylor & Francis, 2005.
[19] K. Ronse, "E-beam maskless lithography: Prosoects and challenges," *Proc. SPIE*, vol. 7637, Apr. 2010, Art. ID 76370A.
[20] M. J. Wieland *et al.*, "MAPPER: High throughput maskless lithography," *Proc. SPIE*, vol. 7637, Apr. 2010, Art. ID 76370F.
[21] D. Wu, J. Hu, and R. N. Mahapatra, "Antenna avoidance in layer assignment," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 4, pp. 734–738, Apr. 2006.
[22] (Sep. 1, 2012). *IBM ILOG CPLEX Optimizer.* [Online]. Available: http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/
[23] (Sep. 1, 2012). *The LEDA Package.* [Online]. Available: http://www.algorithmic-solutions.com/leda

**Iou-Jen Liu** received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, and the M.S. degree from the Graduate Institute of Electronics Engineering, National Taiwan University, in 2012 and 2014, respectively.

His current research interests include physical design and design for manufacturability.

**Shao-Yun Fang** (S'11–M'13) received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, and the Ph.D. degree from the Graduate Institute of Electronics Engineering, National Taiwan University, in 2008 and 2013, respectively.

She is currently an Assistant Professor with the Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei. Her current research interests include physical design and design for manufacturability for integrated circuits.

Dr. Fang was the First Place Winner of the 2012 ACM/SIGDA Student Research Competition (Graduate Student Category), and the Silver Award Winner of the 2012 TSMC Outstanding Student Research Award (Category I: Circuit Design Technologies), the Best Paper Award from the 2010 VLSI/Design CAD Symposium (EDA Category), and two best paper nominations from the 2012 and 2013 International Symposium on Physical Design.

**Yao-Wen Chang** (S'94–A'96–M'96–SM'12–F'13) received the B.S. degree from the National Taiwan University, Taipei, Taiwan, in 1988, and the M.S. and Ph.D. degrees in computer science from the University of Texas at Austin, Austin, TX, USA, in 1993 and 1996, respectively.

He is currently an Associate Dean with the College of Electrical Engineering and Computer Science and the Distinguished Professor with the Department of Electrical Engineering, National Taiwan University. He has served as an Independent Board Director of Genesys Logic, Taipei, and the Technical Consultant of Faraday, Hsinchu, Taiwan, MediaTek, Hsinchu, and RealTek, Hsinchu. His current research interests include physical design and manufacturability for integrated circuits. He has co-edited one textbook on electronic design automation and co-authored one book on routing and over 240 ACM/IEEE conference and journal papers in the above areas.

Dr. Chang was the recipient of the four awards at the 50th ACM/IEEE Design Automation Conference (DAC) in 2013 for the First Most Papers (34 DAC papers) in the Fifth Decade, the Most Prolific Author in a Single Year (seven papers in 2012 and 2013 each), the DAC Prolific Author Award (40 Club), the Top-5 Longest Publication Streaks (recent 15 years), the First Place Winner of five recent contests such as the 2013 Placement Contest at the International Conference on Computer-Aided Design (ICCAD), the 2012 DAC Placement Contest, the 2012 ISPD Discrete Gate Sizing Contest, the 2011 PATMOS Timing Analysis Contest, and the 2009 ISPD Clock Tree Synthesis Contest, 15-time Winner of the DAC/ISPD/ICCAD contests on placement, global routing, clock network synthesis, discrete gate sizing, process defect detection, and mask optimization, six Best Paper Awards, 23 Best Paper Award nominations from DAC (five times), ICCAD (four times), and ISPD (six times), the Distinguished Research Award (highest honor) from the Ministry of Science and Technology of Taiwan (three times), the IBM Faculty Awards (three times), the CIEE Distinguished EE Professorship, the MXIC Young Chair Professorship, and the Distinguished (highest honor)/Excellent Teaching Awards from NTU (eight times). He has served on the editorial boards of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS and the IEEE DESIGN AND TEST OF COMPUTERS. He has served as the General Chairs of ICCAD and ISPD, the Program Chairs of ASP-DAC, FPT, ICCAD, and ISPD, and the Steering Committee Chair of ISPD. He is currently the IEEE Council on Electronic Design Automation Vice President of technical activities. He has also served on the Technical Program Committees of all major EDA conferences and the Chair of the EDA Consortium of the Ministry of Education, Taiwan.