2022 35th International Conference on VLSI Design and 2022 21st International Conference on Embedded Systems (VLSID) | 978-1-6654-8505-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/VLSID2022.2022.00037

# Stitch-avoiding Global Routing for Multiple E-Beam Lithography

Kritanta Saha*1, Sudipta Paul†2, Pritha Banerjee†3 and Susmita Sur-Kolay*4
*Advanced Computing and Microelectronics Unit, Indian Statistical Institute, Kolkata, India
†Department of Computer Science and Engineering, University of Calcutta, Kolkata, India
[1]kritanta09@gmail.com, [2]sudiptapaul2@gmail.com, [3]banerjee.pritha74@gmail.com, [4]ssk@isical.ac.in

***Abstract*—In Multiple E-Beam Lithography (MEBL), a layout of an IC is partitioned into vertical stripes whose boundaries are called stitch-lines. Routing patterns in each stripe are written by different beams and/or in different writing passes. However, patterns cut by stitch-lines suffer from overlay errors resulting in severe pattern distortions. These distortions are mainly due to Via violation, Vertical Routing violation, or Short Polygon. In this paper, we have proposed a Stitch-avoiding Global Router. The proposed method selects the best possible global routing path for each net so that even a traditional detailed router can generate a Detailed Routing solution with fewer violations. Experimental results show that our proposed global router followed by a conventional detailed router generates solutions with 17.3%, 6.21% and 1.89% reduction in Via violations, Short polygon violations and routed wirelength respectively with no vertical routing violations, compared to those by traditional global routing.**

***Index Terms*—Multiple e-beam lithography (MEBL), global routing, stitch-line, via violation, short polygon**

## I. Introduction

The decrease in feature size of modern ICs has led to Next-Generation Lithography (NGL) techniques such as Electron Beam Lithography (EBL) [1], Multiple E-Beam Lithography (MEBL) [2], Extreme Ultra-Violet Lithography (EUVL) to overcome the limitations of 193 nm immersion lithography. By using thousands or even millions of e-beams in parallel, MEBL overcomes the low throughput of EBL [1]. A layout is divided into vertical stripes (Fig.1(a)), where the boundaries are called *stitch-lines* [2]. Routing patterns in different stripes are printed by different beams simultaneously and/or in different printing passes. Routing comprises global routing, layer assignment and detailed routing, and hence poses challenges in MEBL, as deflection of two adjacent e-beams produces overlay errors [2] on the patterns cut by stitch-lines (Figs. 1(b) and (c)). Patterns suffering from overlay errors are called critical patterns. Overlay errors cause severe pattern distortions and electrical violations, thereby poor circuit performance or even fabrication defects.

During routing in MEBL, utmost attention should be paid to critical patterns. A *stitch-unfriendly region* $\tau$ [2] is the area within a horizontal distance $\epsilon$ from a vertical stitch-line. Authors in [2] identified three types of violations causing bad patterns, namely (Fig.1(d)) 1) Via violation: vias on stitch-lines, 2) Vertical Routing violation: vertical routing on stitch-lines, and 3) Short Polygon violation: a horizontal wire segment cutting a stitch-line and landing on a stitch-unfriendly region with a leading via is defined as a short polygon [2]. Severe pattern distortions can occur due to the existence of vias and vertical routing violations, hence these are considered as hard constraints, whereas short polygon as a soft constraint as it may not always lead to pattern distortion [2]. Hence, during routing, these violations need to be minimized.

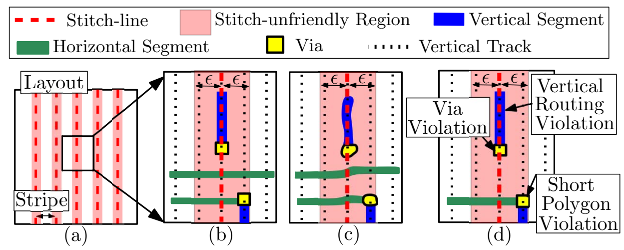



Fig. 1. (a) Stitch-lines in a layout, (b) patterns cut by stitch-lines, (c) pattern distortion due to overlay error, and (d) violations due to a stitch-line.

Wire perturbation-based approach of rerouting nets in [3] reduces stitch-line violations, but in the post routing phase on a pre-routed layout. The perturbations may not avoid all stitch-line violations for a congested and large circuit. The approach in [2] used a traditional global router with a new congestion constraint and reduced the violations mainly in the detailed routing phase. As detailed routing is largely dependent on global routing, it is pertinent to design a dedicated global router for MEBL in order to obtain a detailed routing with no stitch-line violations.

In this paper, we propose a stitch-avoiding global router for MEBL that considers stitch-lines while obtaining a global routing solution such that even a traditional detailed router can generate routing with minimum stitch-line violations. Experimental results of our proposed global router on the Faraday [4] benchmark suite show a significant reduction in the stitch-line violations.

The rest of the paper has the preliminaries, problem formulation and framework of our proposed method in Section II. Sections III and IV present Phase I and II of our global router. Section V provides the experimental results and Section VI the concluding remarks.

## II. Stitch-avoiding Global Routing

### A. Preliminaries

A *grid graph model* represents an instance of global routing by a grid graph $G = (V, E)$ where $V$ is the set of vertices such

978-1-6654-8505-0/22/$31.00 ©2022 IEEE
DOI 10.1109/VLSID2022.2022.00037

that a vertex $v_r \in V$ represents the global grid cell (*G-cell*) $c_r$, and $E = \{e_{rs}\}$ is the set of edges. If G-cells $c_r$ and $c_s$ are adjacent to each other, then $e_{rs} = \{v_r, v_s\} \in E$. Each $e_{rs}$ has routing capacity $C_{rs}$, the maximum number of segments that are allowed to cross the boundary between G-cells $c_r$ and $c_s$.

A G-cell containing stitch-line is a *stitch holding cell*.

*Stitch-avoiding routing capacity* $C_{rs}^{Stitch}$ of edge $e_{rs}$ is defined as $C_{rs}^{Stitch} = (C_{rs} - |L|)$ if $e_{rs}$ is a vertical edge and $c_r$, $c_s$ both are stitch holding cells, where $|L|$ is the number of vertical layers. $C_{rs}^{Stitch} = C_{rs}$ otherwise.

*Available stitch-avoiding routing capacity* $A_{rs}^{Stitch}$ of edge $e_{rs}$ is defined as $(C_{rs}^{Stitch} - U_{rs})$ where $U_{rs}$ is the number of net segments that pass through the G-cell boundary corresponding to edge $e_{rs}$.

*Stitch-avoiding vertex capacity* $C_r^{Stitch}$ is the maximum number of pins or vias that are allowed in the G-cell $c_r$.

*Available stitch-avoiding vertex capacity* $A_r^{Stitch}$ of vertex $v_r$ is defined as $(C_r^{Stitch} - U_r)$ where $U_r$ is the number of pins or vias in G-cell $c_r$.

Let the netlist to be routed be $\mathcal{N} = \{N_1, N_2, ...., N_n\}$. Depending on the number of pins in a net $N_i$, the nets are considered in two categories, namely (a) *multi-pin Nets*, and (b) *two-pin Nets*. Depending on the position of the pins, two sub-categories are identified for two-pin nets as follows:

- A two-pin net $N_i$ is called a *Flat net* if the two pins $P_1^i$ and $P_2^i$ are in G-cells $v_r$ and $v_s$ respectively and the G-cells corresponding to $v_r$ and $v_s$ are in the same row or column of G-cell grid.
- A two-pin net $N_i$ is called a *Bend net* if the two pins $P_1^i$ and $P_2^i$ are in G-cells $v_r$ and $v_s$ respectively that are not in same row or column and there are several paths consisting of a sequence of horizontal and vertical segments to connect the pins present in $v_r$ and $v_s$.

### B. Problem Formulation

The proposed stitch-avoiding global routing problem using grid graph model with $C_{rs}^{Stitch}$ and $C_r^{Stitch}$ is formulated as follows:

*Given a netlist $\mathcal{N} = \{N_1, N_2, ...., N_n\}$, locations of stitch-lines $S = \{S_1, S_2, ...., S_l\}$ and the routing grid graph $G = (V, E)$, find a set of stitch-avoiding rectilinear paths $\Gamma = \{\Gamma_1, \Gamma_2, ..., \Gamma_n\}$ for every net in the netlist $\mathcal{N}$ such that the constraints for stitch-avoiding global routing and vertex capacity are not violated, i.e., $U_{rs} \leq C_{rs}^{Stitch}$ $\forall e_{rs} \in E$, $U_r \leq C_r^{Stitch}$ $\forall v_r \in V$, and the total path length $\Sigma_{i=1}^{n}|\Gamma_i|$ is minimized, where $|\Gamma_i|$ is the length of the path of net $N_i$.*

### C. Proposed Global Routing Framework

A state-of-the-art global router with only non-uniform G-cell capacities cannot ensure reductions in stitch-line violations as these depend on the position of the stitch-lines, pins, Steiner points, vias and segments. These factors need to be considered during global routing to assign global routing paths efficiently which would consequently guide the detailed router to minimize violations. Fig. 2 shows the framework of our proposed stitch-avoiding global router inspired by *BoxRouter* [5]. It has two phases, namely, (a) Stitch-avoiding Steiner Minimum Tree (SSMT) generation followed by (b) multistage stitch-avoiding global routing, which are presented in the next two sections.

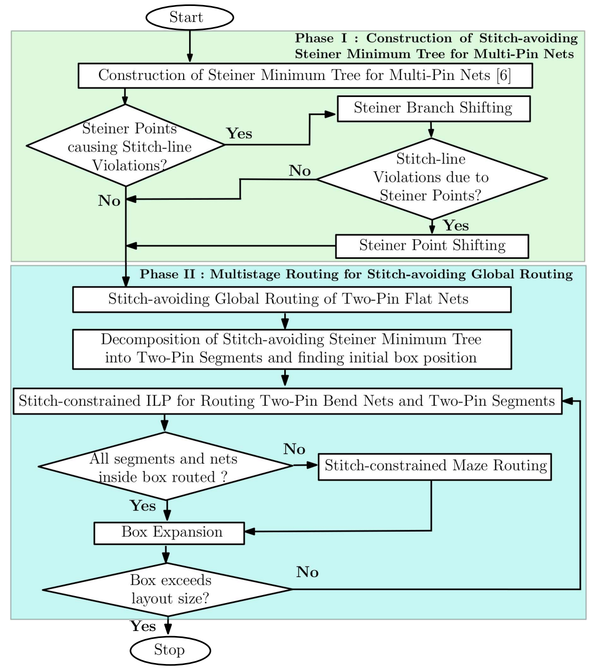



Fig. 2. Framework of Stitch-avoiding Global Routing

## III. Stitch-avoiding Steiner Minimum Tree (SSMT)

For each multi-pin net, we use FLUTE [6] to generate an initial Steiner Minimum Tree (SMT). This SMT may contain Steiner points on the stitch-lines producing via and vertical routing violations, or on the stitch-unfriendly region causing short polygon violation. Two shifting mechanisms namely *Steiner Branch Shifting (SBS)* and *Steiner Point Shifting (SPS)* are performed to avoid these violations. Shifting techniques are used in [7] to get a congestion-aware SMT. However, in this work, stitch-line violations are minimized by shifting a Steiner branch or point out of the stitch-unfriendly region only if it causes a stitch-line violation.

In SBS, a branch of SMT whose both endpoints are Steiner points and lies in a stitch-unfriendly region is a candidate for shifting (see Figs. 3(a) and (b)). In case of SPS, a Steiner point is considered for shifting if it is on a stitch-line or in a stitch-unfriendly region, and one of its connected branches crosses a stitch-line (see Figs. 3(c) and (d)).

As stitch-lines are assumed to be vertical, both SBS and SPS search horizontally for a suitable location to shift a candidate Steiner tree branch or point outside the stitch-unfriendly region. SBS searches for a routing region with sufficient routing capacity outside the nearer boundary of the stitch-unfriendly region. If no such routing region can be found, then it searches towards the farther boundary of the stitch-unfriendly region. If no suitable routing region exists which can hold the branch due to capacity constraints, this

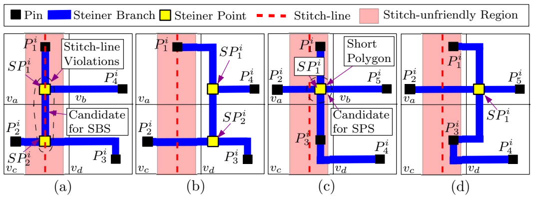



Fig. 3. (a) Steiner branch between $SP_1^i$ and $SP_2^i$ causing via and vertical routing violations, (b) violations removed after SBS, (c) Steiner Point $SP_1^i$ causing short polygon violation, and (d) short polygon removed after SPS.

violation cannot be eliminated during global routing. SPS uses line probing method to shift a candidate Steiner point.

## IV. Multistage Stitch-avoiding Global Routing

Once SSMTs have been generated, the ordering of the nets for global routing is first the two-pin flat nets which are less flexible, then the two-pin bend nets and multi-pin nets.

### *A. Stitch-avoiding Global Routing of Two-Pin Flat Nets*

Two-pin flat nets have only one shortest path between the two pins, either horizontally or vertically spanned. These nets are routed using a shortest path algorithm on $G$ with available routing resource $A_{ij}^{stitch}$ as edge weight. However, some of the vertically spanned nets may reside in stitch holding cells, which need careful resource utilization inside them. These vertically spanned two-pin flat nets are routed by using our proposed stitch-constrained maze routing described below in Section IV.C.

### *B. Stitch-avoiding Global Routing of Two-Pin Bend Nets (TPBN) and Multi-Pin Nets*

The SSMT for each multi-pin net is decomposed into two-pin segments (TPS). These two-pin segments can be either Two-pin Flat Segment (TPFS) or Two-pin Bend Segment (TPBS). Along the lines of *BoxRouter* [5], we define a box on the layout containing multiple $G$-cells. In our method, the four most densely populated contiguous $G$-cells comprise the initial box. A stitch-constrained Integer Linear Programming (ILP) followed by Stitch-constrained Maze Routing are used to route the TPBNs and TPSs residing in the current box. Next, the box is iteratively expanded to include new $G$-cells and new nets. In the next iteration, the newly included nets and only the previously unrouted nets are considered for stitch-avoiding global routing by solving a corresponding stitch-constrained ILP followed by Stitch-constrained Maze Routing. This process is repeated until the box covers the entire layout.

In each iteration, only the TPBNs and TPSs (both TPFSs and TPBSs) that are present inside the current box are routed, so the size of the ILP is small. Hence, solving ILPs for each box does not become a time-consuming task. We can also control the amount of expansion of the current box depending on the density of nets present in the circuit.

A number of constraints and weights are used in the ILP to generate a solution that minimizes stitch-line violations. Let us consider routing of the TPBNs and TPSs of $k^{th}$ box which contains the following three categories of nets and segments:

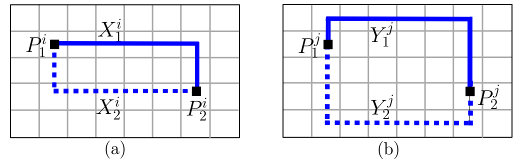



Fig. 4. Two patterns for a net of (a) $Category$ 1 (b) $Category$ 2

- $Category$ 1: TPBNs and TPSs newly included in the $k^{th}$ box.
- $Category$ 2: Unrouted TPBNs and TPSs in the ILP solution for the $(k-1)^{th}$ box.
- $Category$ 3: Unrouted TPBNs and TPSs in the ILP solution for the $(k-2)^{th}$ and $(k-1)^{th}$ boxes.

Let $NET_k$ be the set of $n_1$ Category 1 and $n_2$ Category 2 TPBNs and TPBSs present inside the $k^{th}$ box. Let $NET_k^{Flat}$ be the set of $n_3$ Category 1 TPFSs present inside the $k^{th}$ box.

As shown in Fig. 4(a), for $Category$ 1, only two L-shaped patterns are considered for each TPBN and TPBS to reduce routing bends in the stitch-constrained ILP, because new vias introduced by bends may increase stitch-line violations. Let $X_1^i$ and $X_2^i$ denote the two L-shaped routing patterns of net $N_i$. However, only one horizontal or vertical pattern is considered for TPFS and is denoted by $X_1^i$. In our proposed stitch-constrained ILP, weights are assigned to $X_1^i$ and $X_2^i$ in the objective function as follows:

*a) Weight depending on length and type of net:* Small nets are less flexible, hence need to be routed first. We consider the Manhattan Distance (MD) between the endpoints of each net in the weight function. TPBNs are less flexible than TPSs as both of their endpoints are fixed pins. Hence, a higher weight is assigned to TPBNs. The weight $\gamma^i$ for net $N_i$ depending on its length and category, is defined as

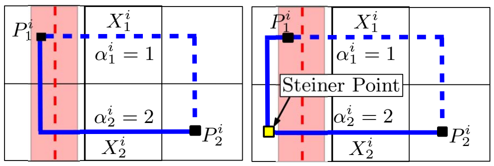
$$\gamma^i = \left\lceil \frac{\sum_{N_j \in NET_k \cup NET_k^{Flat}} MD(N_j)}{(n_1+n_2+n_3).MD(N_i)} \right\rceil + FPins(N_i) \quad (1)$$

where $FPins(N_i)$ is the number of fixed pins in net $N_i$.

*b) Weight depending on routing congestion and the position of endpoints with respect to stitch-lines:* To improve routability, the ratio of $MinAvailableCapacity$, the minimum available capacity and $MaxDemand$, the maximum demand along a path $X_f^i$ ($f = 1$ $or$ $2$), is the weight to route the net through less congested region. Depending on the position of the endpoints of the TPBNs and TPBSs with respect to stitch-lines, different priorities are given to $X_1^i$ and $X_2^i$. The priorities of $X_f^i$ for $f = 1$ or 2, is denoted by $\alpha_f^i$.

Let $P_1^i$ and $P_2^i$ be the two endpoints of the net $N_i$. Depending on the position of stitch-lines with respect to the pin positions, Algorithm 1 computes $\alpha_1^i$ and $\alpha_2^i$ for TPBNs and TPBSs. If $\alpha_1^i$ is assigned a higher value than $\alpha_2^i$, then path $X_1^i$ is more suitable than path $X_2^i$ in the presence of stitch-lines and vice-versa. No preference exists for TPFSs, as only one path is considered. Hence, for TPFSs, we set $\alpha_1^i = 1$.

Figure 5(a) shows an example of a situation that falls under Case I of Algorithm 1. It shows two candidate L-shaped paths for net $N_i$ denoted as $X_1^i$ and $X_2^i$. The pin position with respect




(a) (b)

Fig. 5. (a) Value of $\alpha_1^i$ and $\alpha_2^i$ under a situation; (b) Making path $X_2^i$ violation free by introducing new Steiner Point

to stitch-line satisfies the condition $t_1 = true \wedge t_2 = false \wedge x(P_1^i) < S_1$ of Case I where $S_1 = xStitch(P_1^i)$ . Now, if we choose the path $X_1^i$, then a short polygon is generated at $P_1^i$. Two cases can occur: $P_1^i$ can be either a fixed pin or a Steiner point. The presence of a Steiner point inside the stitch-unfriendly region indicates that both SBS and SPS were unable to move the Steiner point out of the stitch-unfriendly region due to the unavailability of required space outside the stitch-unfriendly region. Under this situation, we can also consider the Steiner point as unmovable. Hence, whether $P_1^i$ is a fixed pin or an unmovable Steiner point, we cannot eliminate the short polygon at $P_1^i$.

If we choose the path $X_2^i$, then there is no short polygon at $P_1^i$. However, during detailed routing, we may need to introduce a via at the junction of horizontal and vertical segment of $X_2^i$. Further, if the vertical segment of $X_2^i$ is assigned to a track inside the stitch-unfriendly region, then the newly introduced via generates a short polygon violation. We can avoid this situation in the global routing phase itself. We introduce a Steiner point outside the stitch-unfriendly region as shown in Fig. 5(b) and thus prevent the short polygon violation from occurring. Hence, we assign $\alpha_1^i = 1$ and $\alpha_2^i = 2$ to give higher priority to path $X_2^i$ than path $X_1^i$.

The total weight of $X_f^i$ for $f = 1$ or 2, is defined as

$$w_f^i = \gamma^i + \alpha_f^i \left\lceil \frac{MinAvailableCapacity(X_f^i)}{MaxDemand(X_f^i)} \right\rceil \qquad (2)$$

$Category$ 2 TPBNs and TPBSs not being routed during $(k-1)^{th}$ iteration, implies that the routing resources are not enough for routing with only the L-shaped patterns. More bends are necessary to route these TPBNs and TPBSs. We consider two patterns with two bends as shown in Fig. 4(b). These two patterns are denoted by $Y_1^j$ and $Y_2^j$ for net $N_j$. These paths have one more bend and a wirelength overhead over $Category$ 1 TPBNs and TPBSs. Hence, a penalty $\varphi$ is subtracted from the weight of $Y_1^j$ and $Y_2^j$.

TPBNs and TPSs in $Category$ 3 are not considered in the $k^{th}$ box for routing and are routed later by stitch-constrained maze routing described in Section IV.C.

The generalized stitch-constrained ILP to route the nets inside $k^{th}$ box is presented in Fig. 6. Consider $E_{Box^k}$ and $V_{Box^k}$ to be the set of edges and vertices of $G$ that are present in the $k^{th}$ box respectively. The objective function specified in Eqn. 3 is to maximize the number of routed nets by choosing candidate paths $X_f$, $Y_q$ and $X_l$ for Category 1 and Category 2 TPBNs, TPBSs and TPFSs present in the $k^{th}$ Box such that

**Algorithm 1:** Finding $\alpha_1^i$ and $\alpha_2^i$

**Input :** $NET_k$: TPBNs and TPBSs inside box k.
**Output:** Weights $\alpha_1^i$, $\alpha_2^i$ for all $N_i \in NET_k$

**for** *each net* $N_i \in NET_k$ **do**
  **Set** $t_1 = \tau Check(P_1^i)$, $t_2 = \tau Check(P_2^i)$ ;
  /* $\tau Check(P_f^i)$ returns $true$ if $P_f^i$ is inside a stitch-unfriendly region and false otherwise for f=1 or 2 */
  **Set** $S_1 = xStitch(P_1^i)$, $S_2 = xStitch(P_2^i)$ ;
  /* $xStitch(P_f^i)$ returns the $x$-coordinate of the stitch-line present in the cell where pin $P_f^i$ resides. */
  **Case I : Set** $\alpha_1^i = 1, \alpha_2^i = 2$ **If**
  $(t_1 = true \wedge t_2 = false \wedge x(P_1^i) < S_1) \vee$
  $(t_1 = false \wedge t_2 = true \wedge x(P_2^i) < S_2) \vee$
  $(t_1 = true \wedge t_2 = true \wedge x(P_1^i) < S_1 \wedge x(P_2^i) < S_2)$;
  /* $x(P_f^i)$ returns the x-coordinate of $P_f^i$ and assume w.l.o.g. that $x(P_1^i) \leq x(P_2^i)$ */
  **Case II : Set** $\alpha_1^i = 2, \alpha_2^i = 1$ **If**
  $(t_1 = true \wedge t_2 = false \wedge x(P_1^i) > S_1) \vee$
  $(t_1 = false \wedge t_2 = true \wedge x(P_2^i) > S_2) \vee$
  $(t_1 = true \wedge t_2 = true \wedge x(P_1^i) > S_1 \wedge x(P_2^i) > S_2)$;
  **Case III : Set** $\alpha_1^i = 1, \alpha_2^i = 1$ otherwise;
**end**

the constraints are satisfied. Eqns. 4 and 5 restrict the ILP solver to choose at most one candidate path for each TPBN and TPBS of Category 1 and 2 respectively. Eqn. 6 allows the ILP solver to select a particular TPFS depending on the value of the objective function and other constraints. Eqns. 7 and 8 in the ILP impose routing and vertex capacity constraints.

### C. Stitch-constrained Maze Routing

After solving the ILP for the $k^{th}$ box, the unrouted two-pin flat nets and the Category 3 TPBNs and TPSs in the $k^{th}$ box are routed. More bends are needed for routing these unrouted nets. Hence, we use stitch-constrained Maze Router

$$\max \sum_{i=1}^{n_1}\sum_{f=1}^{2} w_f^i X_f^i + \sum_{j=1}^{n_2}\sum_{q=1}^{2}(w_q^j - \varphi)Y_q^j + \sum_{l=1}^{n_3} w_1^l X_1^l \quad (3)$$

subject to

$$X_1^i + X_2^i \leq 1 \qquad \forall N_i \in (NET_k \setminus NET_{k-1}) \qquad (4)$$

$$Y_1^j + Y_2^j \leq 1 \qquad \forall N_j \in (NET_k \cap NET_{k-1}) \qquad (5)$$

$$X_1^l \leq 1 \qquad \forall N_l \in NET_k^{Flat} \qquad (6)$$

for each $e_{rs} \in E_{Box^k}$, $X_f^i, Y_q^i$ intersects $e_{rs}$

$$\sum_{e_{rs}}(X_f^i + Y_q^i) \leq A_{rs}^{stitch} \qquad (7)$$

for each $v_r \in V_{Box^k}$, Bend of $X_f^i, Y_q^i$ lies in $v_r$

$$\sum_{v_r}(X_f^i + Y_q^i) \leq A_r^{stitch} \qquad (8)$$

$$X_1^i, X_2^i, Y_1^j, Y_2^j \in \{0, 1\} \qquad (9)$$

Fig. 6. Generalized stitch-constrained ILP to route two-pin bend nets and two-pin segment present inside the $k^{th}$ box

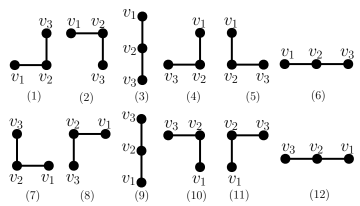


Fig. 7. Twelve possible patterns considered during Retrack

based on Hadlock's algorithm [8] which uses *detour number* defined as the number of grid cells directed away from its target. In our method, for each vertex $v \in G$, instead of using only the *detour number* of the neighbouring vertices of $v$, we use four parameters, namely, detour number, available routing resource of the edge between $v$ and its neighbour, available vertex capacity and *Stitch Penalty*. If the $G$-cells $c_r$ and $c_s$ are both stitch holding cells, then the Stitch Penalty $\rho(v_r, v_s)$ = number of tracks in stitch-unfriendly region $\tau$, and 0 otherwise. While exploring $v_r$, we define a cost for each of its neighbours $v_s$ as follows

$$cost(v_s) = A_{rs}^{stitch} + A_{s}^{stitch} - \rho(v_r, v_s) \quad (10)$$

Our method stores the detour number and the cost of each explored vertex $v_r$. Then our maze router explores the neighbouring vertices $v_s$ in non-increasing order of their cost.

The Retrack phase of our Stitch-constrained Maze router traces the path starting from the target vertex to the source vertex. In order to reduce bends and select the path with maximum available routing resources, we consider at a time these three vertices: vertex $v_1$, its neighbour $v_2$ and $v_2$'s neighbour $v_3$. Initially, $v_1 = t$. The neighbours are selected according to their detour numbers, i.e., $D[v_1] \geq D[v_2] \geq D[v_3]$. If one of the neighbours, either $v_2$ or $v_3$, is the source vertex then we terminate and return the path obtained.

The 12 patterns in which $v_1, v_2$, and $v_3$ can appear in $G$ are shown in Fig. 7. But not all the possibilities can appear always as some of the neighbouring vertices may not be visited during the exploration phase. For the patterns with bends, we discard a pattern if $A_r^{stitch} = 0$. For each possible pattern, its bottleneck cost is $\min(cost(v_1), cost(v_2), cost(v_3))$. For the patterns with a bend, we subtract a bend penalty from $cost(v_2)$. For patterns with bends and $v_2$ representing a stitch holding cell, we subtract a stitch penalty from $cost(v_2)$ also. We select the pattern with maximum bottleneck capacity. Next, we consider $v_3$ as the initial vertex and iterate until the source vertex is reached.

## V. Experimental Results

Our proposed stitch-avoiding global router is implemented in Java on a Linux workstation with 2.40 GHz Intel(R) Xeon(R) CPU and 32 GB RAM. We use FLUTE [6] to construct SMT and GLPK as the ILP solver. We consider the Faraday suite (Table I) in ICCAD'04 Mixed-size Placement Benchmarks [4] [9].

We perform placement by NTUplace3 [10] and then test our global router on its outputs. The distance between two stitch-lines is taken as $15\times$ the routing pitch.

Table II shows the number of Steiner branches and Steiner points before and after performing SBS and SPS. The number of Steiner branches and points on stitch-lines are denoted by $\#SB^{SL}$ and $\#SP^{SL}$ respectively, and $\#SB^{\tau}$ and $\#SP^{\tau}$ correspond to the numbers in the stitch-unfriendly regions. $HPWL$ gives the total half-perimeter wirelength. While SBS reduces $\#SB^{SL}$, $\#SP^{SL}$, $\#SB^{\tau}$ and $\#SP^{\tau}$ by about 70%, 59%, 44% and 31% respectively with an overhead of 0.14% increase in HPWL, SPS reduces these four parameters by 100%, 100%, 35% and 25% respectively with an overhead of 0.07% increase in HPWL.

Table III compares the performance of our proposed global router with a traditional global router called *BoxRouter* [5] and *BoxRouter* with reduced routing capacities for Stitch holding cells. Due to the unavailability of code for BoxRoute, we have implemented it in JAVA. $\#TVOF$ denotes the total number of vertex capacity overflow and $\#MVOF$ the maximum overflow of vertex capacity in a G-cell. $WL$ denotes the total wirelength of global routing paths. We set $\varphi$ to 1 in Eqn. 3 of stitch-constrained ILP, and bend penalty in our maze router as 1. Table III shows that our proposed method reduces $\#SB^{SL}$ and $\#SP^{SL}$ by 100%. Due to SBS and SPS, the density of Steiner points in the cells around the stitch holding cells increased. However, the values of $TVOF$ and $MVOF$ indicate that these increments did not exceed the vertex capacities. Moreover, our approach did not generate any overflow for RISC2 as in *BoxRouter*. Our proposed global router achieved 100% routability for all the nets with 0.28% and 16% overhead in wirelength and runtime respectively.

The effect of our global router on reduction of stitch-line induced violations is evaluated by performing detailed routing with the traditional detailed router *RegularRoute* [11]. Here, we block vertical routing on the tracks that coincide with the stitch-lines to avoid vertical routing violations. As the source code of *RegularRoute* is unavailable, we have implemented it in JAVA. Table IV shows the comparison of our proposed stitch-avoiding global router followed by *RegularRoute* with Baseline Router I composed of traditional *BoxRouter* followed by *RegularRoute* and Baseline Router II composed of *BoxRouter* with reduced routing capacities of Stitch holding cells followed by *RegularRoute*. $\#VV$, $\#SHP$ and $\#VRV$ denote the number of via violations, short polygons and vertical routing violations respectively. $\%Rout.$ denotes the routability in percentage. Due to the blocking of vertical routing on stitch-lines, $\#VRV$ is zero for both the Baseline routers and our proposed router.

Table IV demonstrates that the Baseline Router II does not reduce the stitch-line violations as much as Baseline

TABLE I
Faraday benchmark Circuits of ICCAD04 [4]

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

TABLE II

NUMBER OF STEINER BRANCHES(SB) AND STEINER POINTS(SP) BEFORE AND AFTER PERFORMING SBS AND SPS

| Circuits | Initial: Before SBS and SPS | | | | | After SBS | | | | | | After SBS and SPS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\#SB^{SL}$ | $\#SP^{SL}$ | $\#SB^{r}$ | $\#SP^{r}$ | HPWL($\mu$m) | $\#SB^{SL}$ | $\#SP^{SL}$ | $\#SB^{r}$ | $\#SP^{r}$ | HPWL($\mu$m) | CPU(ms) | $\#SB^{SL}$ | $\#SP^{SL}$ | $\#SB^{r}$ | $\#SP^{r}$ | HPWL($\mu$m) | CPU(ms) |
| DMA | 1522 | 1240 | 3904 | 3207 | 544183.62 | 494 | 525 | 2307 | 2327 | 545049.31 | 19 | 0 | 0 | 1500 | 1769 | 545577.02 | 14 |
| DSP1 | 2184 | 1696 | 6157 | 4746 | 1143573.09 | 610 | 654 | 3258 | 3165 | 1145058.71 | 24 | 0 | 0 | 2103 | 2357 | 1145740.70 | 26 |
| DSP2 | 2070 | 1591 | 6295 | 4803 | 1034884.39 | 552 | 597 | 3450 | 3255 | 1036381.16 | 29 | 0 | 0 | 2366 | 2468 | 1037014.39 | 23 |
| RISC1 | 3854 | 3091 | 10082 | 8075 | 1669906.97 | 1195 | 1288 | 5702 | 5586 | 1672344.89 | 41 | 0 | 0 | 3712 | 4216 | 1673591.58 | 47 |
| RISC2 | 3949 | 3183 | 10108 | 8105 | 1688466.52 | 1201 | 1307 | 5759 | 5709 | 1690820.61 | 44 | 0 | 0 | 3732 | 4312 | 1692085.39 | 41 |
| Ratio | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.2984 | 0.4047 | 0.5605 | 0.6926 | 1.0014 | - | 0.00 | 0.00 | 0.6550 | 0.7545 | 1.0007 | - |

TABLE III

COMPARISON OF OUR PROPOSED STITCH-AVOIDING GLOBAL ROUTER WITH TRADITIONAL GLOBAL ROUTER

| Circuits | BoxRouter [5] | | | | | | BoxRouter [5] with Reduced Capacities | | | | | | Our Proposed Stitch-avoiding Global Router | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\#SB^{SL}$ | $\#SP^{SL}$ | TVOF | MVOF | WL($\mu m$) | CPU(s) | $\#SB^{SL}$ | $\#SP^{SL}$ | TVOF | MVOF | WL($\mu m$) | CPU(s) | $\#SB^{SL}$ | $\#SP^{SL}$ | TVOF | MVOF | WL($\mu m$) | CPU(s) |
| DMA | 1522 | 1240 | 0 | 0 | 556443.08 | 20.52 | 1522 | 1240 | 0 | 0 | 556443.08 | 20.52 | 0 | 0 | 0 | 0 | 558491.43 | 25.79 |
| DSP1 | 2184 | 1696 | 0 | 0 | 1163966.87 | 45.72 | 2184 | 1696 | 0 | 0 | 1163974.86 | 49.19 | 0 | 0 | 0 | 0 | 1166902.24 | 53.19 |
| DSP2 | 2070 | 1591 | 0 | 0 | 1057605.85 | 45.45 | 2070 | 1591 | 0 | 0 | 1057635.54 | 46.85 | 0 | 0 | 0 | 0 | 1060317.83 | 53.20 |
| RISC1 | 3854 | 3091 | 0 | 0 | 1705038.18 | 81.83 | 3854 | 3091 | 0 | 0 | 1705102.18 | 90.68 | 0 | 0 | 0 | 0 | 1709656.30 | 91.80 |
| RISC2 | 3949 | 3183 | 1 | 4 | 1720034.83 | 77.29 | 3949 | 3183 | 0 | 0 | 1720050.82 | 87.54 | 0 | 0 | 0 | 0 | 1724911.15 | 92.39 |
| Ratio | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.1005 | 0.00 | 0.00 | 0.00 | 0.00 | 1.0028 | 1.1682 |

TABLE IV

EFFECTIVENESS OF THE PROPOSED STITCH-AVOIDING GLOBAL ROUTER AFTER DETAILED ROUTING

| Circuits | Baseline Router I : BoxRoute [5] and RegularRoute [11] | | | | | Baseline Router II : BoxRoute [5] with Reduced Capacities and RegularRoute [11] | | | | | Our Proposed Stitch-avoiding Global Router and RegularRoute [11] | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\#VV$ | $\#SHP$ | $\#VRV$ | %Rout. | CPU(Sec.) | $\#VV$ | $\#SHP$ | $\#VRV$ | %Rout. | CPU(Sec.) | $\#VV$ | $\#SHP$ | $\#VRV$ | %Rout. | CPU(Sec.) |
| DMA | 3589 | 1626 | 0 | 95.94 | 421 | 3587 | 1629 | 0 | 95.85 | 406 | 2990 | 1513 | 0 | 96.90 | 406 |
| DSP1 | 7469 | 3496 | 0 | 96.56 | 1419 | 7453 | 3502 | 0 | 96.59 | 1418 | 6225 | 3246 | 0 | 97.11 | 1413 |
| DSP2 | 7410 | 3462 | 0 | 96.69 | 1296 | 7408 | 3480 | 0 | 96.71 | 1304 | 6227 | 3220 | 0 | 97.74 | 1303 |
| RISC1 | 10799 | 5246 | 0 | 95.92 | 3016 | 10798 | 5264 | 0 | 95.95 | 3039 | 8778 | 4993 | 0 | 96.09 | 3035 |
| RISC2 | 10934 | 5565 | 0 | 96.09 | 2944 | 10939 | 5537 | 0 | 96.18 | 2986 | 8891 | 5218 | 0 | 96.48 | 2955 |
| Ratio | 1.00 | 1.00 | - | 1.00 | 1.00 | 1.002 | 1.0008 | - | 1.004 | 1.006 | 0.8236 | 0.9378 | - | 1.006 | 1.0017 |

Router I. This indicates that it is not possible to reduce stitch-line violations only by the traditional *BoxRouter* with reduced routing capacities of the stitch holding cells. However, Table IV also demonstrates that our proposed Stitch-avoiding global router followed by detailed routing with *RegularRoute*, achieved on average a reduction of $17.3\%$ and $6.21\%$ in $\#VV$ and $\#SHP$ respectively as compared to the Baseline Router I. Further, at termination, the proposed method attained a reduction of $1.89\%$ on average in total routed wirelength with $0.6\%$ improvement in the $\%Rout.$ compared to the results of the Baseline router I. In [2] and [3], the placement tools used to place the circuits of Faraday benchmark suite were not disclosed. As placement affects the number of stitch-line violations, we cannot compare our results with those by [2] and [3] without this information.

## VI. CONCLUDING REMARKS

This paper presents a stitch-avoiding global router for MEBL. Different shifting mechanisms are imposed along with stitch-constrained ILP and stitch-constrained maze routing to route nets with minimum stitch-line violations. Experimental results show significant reductions in the stitch-line violations due to our stitch-avoiding global routing. The performance of the proposed Stitch-avoiding global router followed by other state-of-the-art commercial and academic detailed routers needs to be analyzed for further improvements. A stitch-avoiding detailed router can further reduce the stitch-line violations. Hence, designing an efficient stitch-avoiding detailed router is in the future scope of this work.

## REFERENCES


[1] B. J. Lin, “Future of multiple-e-beam direct-write systems,” *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 11, no. 3, p. 033011, 2012.

[2] I.-J. Liu, S.-Y. Fang, and Y.-W. Chang, “Stitch-aware routing for multiple e-beam lithography,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 3, pp. 471–482, 2015.

[3] S. Paul, P. Banerjee, and S. Sur-Kolay, “Post-layout perturbation towards stitch friendly layout for multiple e-beam lithography,” in Proceedings 2017 *IEEE International Conference on Computer Design* (ICCD). IEEE, 2017, pp. 411–414.

[4] “Faraday benchmarks,” ICCAD, 2004. [Online]. Available:http://vlsicad.eecs.umich.edu/BK/ICCAD04bench/FARADAYICCAD04Bench.tar.gz

[5] M. Cho and D. Z. Pan, “BoxRouter: A new global router based on box expansion and progressive ILP,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 12, pp. 2130–2143, 2007.

[6] C. Chu, “Flute: Fast lookup table based technique for rsmt construction and wirelength estimation.” [Online]. Available: http://home.eng.iastate.edu/~cnchu/flute-3.1/flute-3.1.tgz

[7] M. Cho, K. Lu, K. Yuan, and D. Z. Pan, “BoxRouter 2.0: A hybrid and robust global router with layer assignment for routability,” *ACM Transactions on Design Automation of Electronic Systems* (TODAES), vol. 14, no. 2, pp. 1–21, 2009.

[8] F. Hadlock, “Finding a maximum cut of a planar graph in polynomial time,” *SIAM Journal on Computing*, vol. 4, no. 3, pp. 221–225, 1975.

[9] S. Adya, S. Chaturvedi, J. Roy, D. Papa, and I. Markov, “Unification of partitioning, floorplanning and placement,” in *Proceedings of International Conference on Computer-Aided Design* (ICCAD), 2004, pp. 550–557.

[10] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen, and Y.-W.Chang, “NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 7, pp. 1228–1240, 2008.

[11] Y. Zhang and C. Chu, “RegularRoute: An efficient detailed router applying regular routing patterns,” *IEEE Transactions on Very Large Scale Integration* (VLSI) systems, vol. 21, no. 9, pp. 1655–1668, 2012.