Self-Aligned Double Patterning Lithography Aware Detailed Routing with Color Pre-Assignment

Yixiao Ding, *Student Member, IEEE,* Chris Chu, *Fellow, IEEE* and Wai-Kei Mak, *Member, IEEE,*

***Abstract*—As the technology nodes scale down to sub-22nm, double patterning lithography (DPL) has been considered as a practical solution for layout manufacturing. Compared with litho-etch-litho-etch (LELE), self-aligned double patterning (SADP) has better overlay control. To have a better SADP layout decomposability of routing patterns, we consider SADP during detailed routing stage. Two major types of SADP processes are considered: spacer-is-dielectric (SID) type and spacer-is-metal (SIM) type. Different from previous works, the idea of color pre-assignment is adopted for SADP-aware detailed routing. An elegant graph model is proposed to capture both routing and SADP manufacturing cost. They greatly simplify the problem to maintain SADP design rules in detailed routing. We apply a negotiated congestion based rip-up and reroute scheme to achieve better routability while maintaining SADP design rules. Compared with other state-of-the-art academic works, our approach does not produce any side overlay error and no SADP design rules violation is reported. Meanwhile, a better solution in terms of total wirelength, routability, and runtime is achieved.**

***Index Terms*—Self-aligned double patterning lithography, detailed routing, algorithm, manufacturability, color pre-assignment.**

I. Introduction

As the technology nodes scale down to 22nm and beyond, double patterning lithography (DPL) has been considered as a practical solution for layout manufacturing. There are two major types of DPL: litho-etch-litho-etch (LELE) and self-aligned double patterning (SADP). In LELE lithography, layout patterns are firstly decomposed into two halves and each half of patterns is assigned into a mask. Then, a process of exposure and etch is applied on each mask for layout manufacturing. In the decomposition step, adjacent patterns with space less than manufacturing limit are assigned into different masks. Thus, design rule violations are avoided and smaller chip features are obtained. However, LELE requires accurate alignment on the second mask exposure. Otherwise, the overlay error will cause yield loss.

Compared with LELE, SADP has less stringent overlay requirements. Thus, it is a promising technique to further push beyond the alignment constrained resolution limit of LELE. Two popular types of process are developed for SADP [1]. One is spacer-is-metal (SIM) in which the spacers directly define metal patterns. The other one is spacer-is-dielectric (SID) in which spacers ultimately define trenches. Both types of SADP utilize two masks: a core mask to make mandrel patterns and a cut/trim mask to get final layout patterns. Fig. 1 shows an example of target layout manufactured by two different SADP processes. In SIM type SADP as shown in Fig. 1(b), the spacer is firstly deposited around the pre-featured mandrel patterns. Since the spacer itself becomes the metal pattern, the cut mask is used to cut off unwanted portions of patterns formed by spacer. Thus, the regions covered by spacer but not covered by cut mask patterns produce the final layout. In addition, since the width of spacer is constant, it is difficult to vary the line-width of metal patterns by SIM type SADP [1]. In SID type SADP as shown in Fig. 1(c), after the similar spacer deposition, the mandrels are removed and sub-metals are deposited at both original mandrels locations and space between spacers. Then, the trim mask is used to include target layout patterns. Since the spacer is just dielectric, the regions not covered by spacer but covered by trim mask form the final patterns. In this paper, we assume the SIM type SADP applies cut mask, and SID type SADP uses trim mask for layout manufacturing. Note that our work can be potentially extended to handle some other SADP variants, e.g., trim-based SIM type SADP and cut-based SID type SADP.

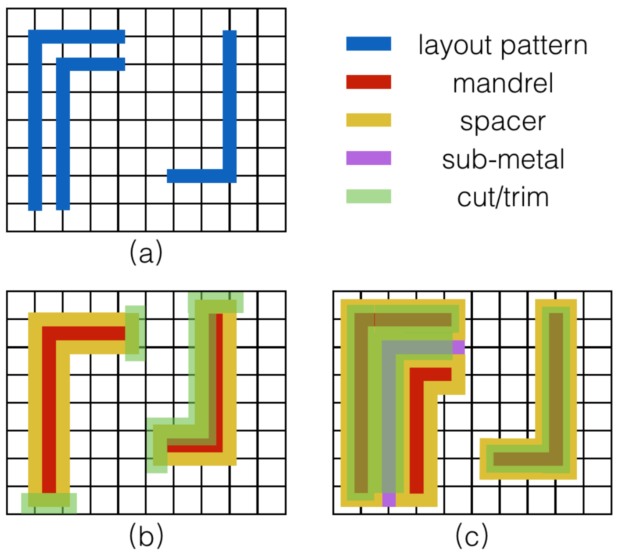

Fig. 1. Two types of SADP process. (a) Target layout. (b) SIM type SADP. (c) SID type SADP.

This work was supported in part by the Ministry of Science and Technology under grant MOST 104-2628-E-007-003-MY3.

Preliminary version of this paper is in the Proceedings of the 52nd ACM/IEEE Design Automation Conference (DAC), 2015

Yixiao Ding and Chris Chu are with the Department of Electrical and Computer Engineering, Iowa State University, IA, USA.

Wai-Kei Mak is with the Department of Computer Science, National Tsing Hua University, 101 Kuan Fu Rd. Sec. 2, Hsinchu, Taiwan 300 R.O.C.

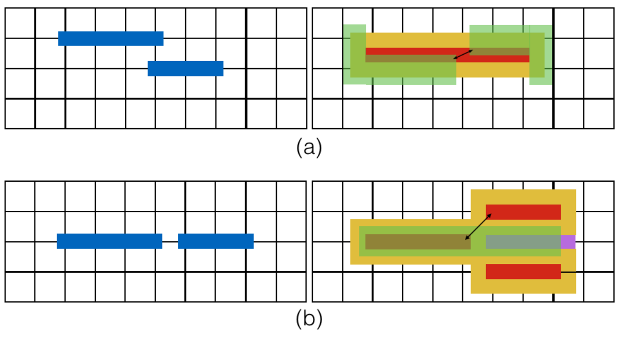

Fig. 2. SADP undecomposable layout configuration. (a) Design rule violation occurs on cut mask in SIM process. (b) Design rule violation occurs on core mask in SID process.

The LELE layout decomposition problem can be modeled as a two-coloring problem with stitch minimization [2], [3]. Two patterns are assigned with different colors if they have a conflict. A pattern may be split into two parts to resolve a conflict but results in a stitch. Then layout patterns with the same color are assigned into the same mask. Similar to LELE, SADP also requires a decomposition of the layout into core mask and cut/trim mask. However, SADP layout decomposition is more challenging since the resulting mandrel mask and cut/trim mask look significantly different from the target layout. A layout configuration without considering SADP layout decomposition will probably make the layout not manufacturable by SADP. As shown in Fig. 2(a), after layout decomposition, the two adjacent cut mask patterns are too close and cause design rule violation. Thus, the target layout is not manufacturable by SIM type SADP. Similarly, a design rule violation occurs on the core mask after layout decomposition in Fig. 2(b). Thus, the target layout cannot be manufactured by SID type SADP. In order to avoid these SADP unmanufacturable patterns, it is necessary to consider SADP layout decomposition in the earlier design stage, especially in detailed routing. This will greatly improve the decomposability of layout patterns during manufacturing time.

[4] proposed an LELE-aware detailed routing algorithm which applied a color pre-assignment idea. Each track in the routing grid is firstly assigned with a fixed color, then maze routing is performed on the grid. In this way, the mask assignment is known at the moment once a net is routed. Hence, the presence of stitches is foreknown during detailed routing and can be easily minimized. Besides, the way that the colors are pre-assigned enables the detailed router to generate only decomposable layout. We extend the idea of color pre-assignment to our SADP-aware detailed routing. [5], [6], [7] presented the SADP layout decomposition of arbitrary 2D patterns in post routing stage. However, the design flexibility is restricted in this stage and some layout patterns may be naturally undecomposable. Previous works on SADP-aware detailed routing only focus on SID type SADP [8], [9], [10], [11], [12]. To the best of our knowledge, there is no previous work considering SIM type SADP during detailed routing stage. [8] solved routing and layout decomposition problems simultaneously based on the correct-by-construction approach. However, the final routability and layout decomposability heavily depend on the net ordering. [9] performed routing on a grid structure where grid nodes are alternately colored by two colors or uncolored for SADP. However, their approach is unrealistic because it requires that every pin of each net must fall on the same colored grid nodes, otherwise it cannot route the net. [10] proposed an expanded routing graph model. Each point in the routing grid is split into four vertices in the routing graph, and further split into eight vertices to consider prohibited line-ends. The high complexity of routing graph will slow down the runtime and increase memory load of detailed router. In addition, how to handle via in the graph model is not mentioned. [11] maintained a overlay constraint graph during routing which is expensive. Meanwhile, the side overlay error cannot be avoided in the experimental results. [12] mainly focused on SADP-aware pin access for standard cell instead of SADP-aware detailed routing. Color pre-assignment is the key idea and major innovation for our SADP-aware detailed routing. Some primitive forms of color pre-assignment idea are proposed and applied in [13], [14], [15], [10]. [13] proposed a feature assignment method for SID type SADP layout decomposition of line-space array. They treated lines as main mandrel features alternately while remaining lines as sub-metals. [14], [15] applied the similar idea on their SADP-aware pin access for purely unidirectional patterns. [10] assigned routing tracks as main mandrel tracks and sub-metal tracks alternatively before detailed routing.

The contributions of this paper are summarized in the followings.

- This is the first work to systematically consider SIM type SADP lithography during detailed routing stage.
- We extend the idea of color pre-assignment to both SIM and SID types SADP-aware detailed routing. The idea greatly simplifies the problem to maintain SADP design rules in detailed routing.
- The novel graph models are proposed for both SIM and SID type SADP-aware detailed routing. They effectively capture both routing and SADP manufacturing cost.
- We offer the strong routing results in terms of wirelength, routability, and runtime. Furthermore, in the final routing solution, no side overlay error is guaranteed for SADP layout manufacturing.

The rest of the paper is organized as follows. Section 2 provides some preliminary information, especially the SADP design rules considered in the paper. Section 3 presents our problem formulation. Section 4 explains our proposed solution to the problem in details. Section 5 demonstrates our experimental results. Finally, Section 6 concludes the paper.

II. Preliminaries

As mentioned in the previous section, it is hard to print metal lines with mixed-width by SIM type SADP lithography. However, it is possible to manufacture patterns with mixed width by SID type SADP lithography. Since the focus of the SID type SADP-aware detailed routing in this paper is how to apply our key idea of color pre-assignment, we will not further consider mixed-width wires. Thus, we assume that all the

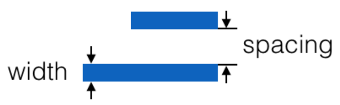

Fig. 3. Minimum spacing and minimum width rules.

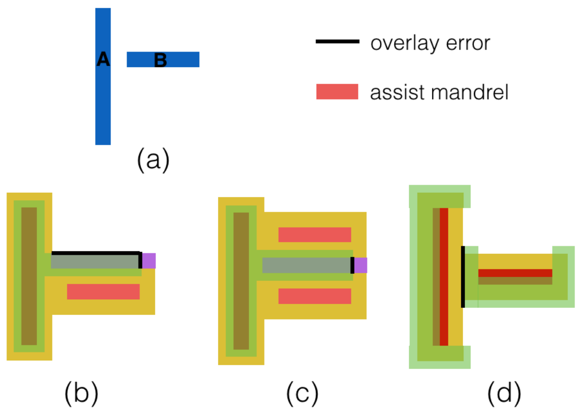

Fig. 4. SADP layout decomposition with overlay error. (a) Target layout. (b) SID type with side overlay error. (c) SID type with no side overlay error. (d) SIM type with side overlay error.

metal patterns in the layout are regular with same fixed width. To successfully manufacture the layout by SADP lithography, SADP design rules should be maintained. In this section, we will discuss the SADP design rules considered in this paper and the incurred routing constraints in detailed routing.

A. Core and cut/trim mask design rules

As mentioned before, core mask is used to make mandrel patterns in both types of SADP. A secondary mask, a cut mask in SIM process and a trim mask in SID process, is used to form the target layout patterns. Due to the optical resolution limits, several design rules should be enforced in the design of core and cut/trim masks. In this paper, we consider minimum spacing and minimum width rules as shown in Fig 3. The minimum spacing rule requires any two adjacent patterns on the mask should be separated with distance at least minimum spacing value. We define S_c and S_s as the minimum spacing values for core and secondary masks. The minimum width rule specifies the minimum width for every pattern on the mask. W_c and W_s denote minimum width values for core and secondary masks.

B. Overlay error

The major advantage of SADP over LELE is the better overlay control. However, misalignment of secondary mask sometimes could still cause overlay error, which results in pattern distortions. [11] defines a *side overlay error* as an overlay error occurs at a section of the side boundary of a layout pattern and *tip overlay error* as an overlay error occur at the line end of a layout pattern. [11] also points out tip overlays are considered as non-critical overlays which can be ignored while side overlays should be minimized to reduce yield loss. Fig. 4(b)(c) show two methods of SID type SADP layout decomposition for target layout in Fig. 4(a). For the method in Fig. 4(b), the pattern B will be generated with a side overlay error at upper boundary and a tip overlay error at the right line end. However, if side boundaries of pattern B are all surrounded by spacers as shown in Fig. 4(c), no side overlay error will occur in SADP layout manufacturing. Hence, an additional mandrel pattern is placed at the upper side of pattern B to provide spacer protection. We refer to the additional mandrel pattern as an assist mandrel. To obtain zero side overlay error in SID type SADP process, we require that all the patterns formed from sub-metals have spacer protection in layout decomposition. Fig. 4(d) shows the SIM type SADP layout decomposition for the same target layout. The pattern A will be generated with a side overlay error at its right side boundary. Therefore, we require the cut mask patterns not to overlap with side boundaries of spacers which used to form target layout patterns. In this way, zero side overlay error can be achieved for SIM type SADP layout manufacturing.

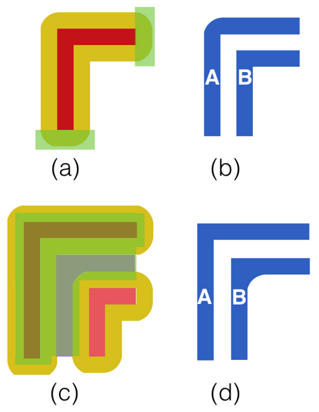

Fig. 5. Non-preferred turns caused by the spacer rounding issue. (a) Rounded spacers deposited at convex corners of a mandrel. (b) A non-preferred turn in SIM type SADP. (c) Large residue occurs at the concave corner of the sub-metal. (d) A non-preferred turn in SID type SADP.

C. Non-preferred turns

[13] observes the corner rounding of the spacer deposition around mandrel line ends in the simulations of the mandrel contours. [16] further observers that spacers get rounded at convex corners of mandrels while staying sharp at concave mandrel corners. Fig. 5(a) shows the rounding issue of spacers. As a result, additional constraints need to be considered for layout manufacturing. In SIM type SADP, if an L shape layout pattern is formed from the rounded spacer at a convex mandrel corner, yield loss will increase. Therefore, whenever L shape layout patterns are formed, we prefer using spacers deposited around concave corners of mandrels. In the case shown in Fig. 5(b), pattern B is preferred over pattern A by SIM type SADP. In SID type SADP, when an L-shape layout pattern is defined by sub-metal, large residue will occur at its concave corner due to the spacer rounding at the convex corner of the assist mandrel. As a result, the pattern is manufactured with distortion by SID type SADP. In order to obtain the clean layout patterns, we prefer to use mandrel patterns to directly define L-shape layout patterns in SID type SADP lithography.

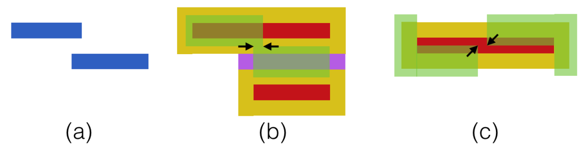

Fig. 6. Prohibited anti-parallel line-ends. (a) A layout contains anti-parallel line-ends. (b) Minimum width rule violation occurs by SID type. (c) Minimum spacing rule violation occurs by SIM type.

Fig. 5(c) shows the layout decomposition for two L-shape layout patterns manufactured by SID type SADP. As shown in Fig. 5(d), pattern A is preferred over pattern B by SID type SADP. [10] also identified this manufacturing challenge and called it **sm-jogs** minimization. In this paper, we refer to L-shape layout patterns manufactured by rounded spacer in SIM process or sub-metal in SID process as non-preferred turns. The number of non-preferred turns should be minimized in detailed routing.

D. Prohibited line-ends

The prohibited line-ends refer to a particular line-ends configuration in the layout pattern which is prohibited during detailed routing due to the manufacturing challenge. [10] identified the "anti-parallel line-ends", in which two layout patterns are on adjacent tracks and form a pair of line ends in opposite direction. Fig. 6(a) shows a layout pattern containing an anti-parallel line-ends. In the SID type SADP layout decomposition shown in Fig. 6(b), the minimum width rule is violated when two trim mask patterns are merged. Thus, the anti-parallel line-ends should be prohibited in SID type SADP-aware detailed routing. The anti-parallel line-ends in Fig. 6(a) should also be prohibited in SIM type SADP-aware detailed routing. This is due to the violation of minimum spacing rule on cut mask patterns, which is shown in Fig. 6(c). Therefore, the anti-parallel line-ends should either have enough horizontal overlapping length len_o or enough horizontal separation distance $dist_s$. $len_o = W_s$ and $dis_s = S_s$ for SID type, while $len_o = S_s$ and $dist_s = W_s$ for SIM type.

Besides, the line-ends configuration in Fig. 2(b) should be prohibited in SID type SADP-aware detailed routing due to the minimum spacing rule violation on core mask. We refer to this line-ends configuration as anti line-ends. Furthermore, the line-ends configuration in Fig. 4(d) should be prohibited in SIM type SADP-aware detailed routing due to the intolerable side overlay error during manufacturing.

III. Problem formulation

We assume that there is a preferred routing direction for each layer and the other direction perpendicular to the preferred routing direction is defined as non-preferred routing direction of the layer. We do not completely disable but strongly discourage routing in the non-preferred routing direction. We refer to the above problem as the restricted detailed routing problem. In addition, we assume all the multi-pin nets from netlist have been decomposed into 2-pin nets and each pin has several candidate locations. With such assumption and design rules mentioned in Section 2, we formulate the SADP lithography aware detailed routing problem.

Given a netlist, a multi-layer routing grid, a set of blockages, and design rules, restricted detailed routing with simultaneous pin location determination for all the nets is performed. The final routing patterns should be compliant to design rules of either SIM or SID type SADP lithography in layout manufacturing. The objective is to achieve 100% routability and to ensure zero side overlay error in SADP layout decomposition. Besides, design rule violations, total wirelength, the number of vias, and non-preferred turns should be minimized

IV. Proposed solution

A. Overall flow

The overall flow of our SADP-aware detailed routing is shown in Fig. 7. Assuming a netlist, a routing grid, a set of blockages, and design rules are given. The routing of each net should be along the grid lines. We firstly perform color pre-assignment on the routing grid, then we build a routing graph based on our proposed graph model. After that, we perform independent routing iterations. During this phase, we route all the nets *almost* independently in each iteration to minimize the negative impact of net ordering. Several heuristics are applied here in order to obtain better solution in fewer number of iterations. This phase will terminate if the congestion of the current iteration is no better than the previous one. Next, we treat the output of independent routing iterations as the initial routing solution for the negotiated congestion based rip-up and reroute (RNR) phase. The RND iterations will continue until there is no congestion in the routing solution or it reaches the pre-set maximum number of iterations. The last phase is design rule violation removal based RNR iterations. All the prohibited line-ends in the layout are firstly identified. Then line end extension or cut/trim mask pattern merge is performed to resolve violations. If violations still exist, RNR is called to legalize the routing solution. We also has a pre-set maximum number of RNR iterations for this phase. The iterations will stop once there is no violation or it reaches the maximum iteration count. Finally, SADP-friendly routing solution is generated, and congestions and design rules violations are reported if existed.

B. Color pre-assignment

Different from LELE, SADP layout decomposition cannot be simplified into a 2-coloring problem. Meanwhile, additional design rules should be maintained and side overlay errors need to be avoided during layout decomposition. Those make the consideration of SADP lithography in detailed routing an even more complicated problem. In this situation, we adopt the idea of color pre-assignment to simplify the problem. Before detailed routing, the routing grid is assigned colors to determine where mandrel patterns and cut/trim mask patterns can be formed. With such restriction, the layout decomposition is known at the moment when a net is routed. In this way, SADP design rule violations and side overlay errors can be easily avoided during detailed routing. In this subsection, we

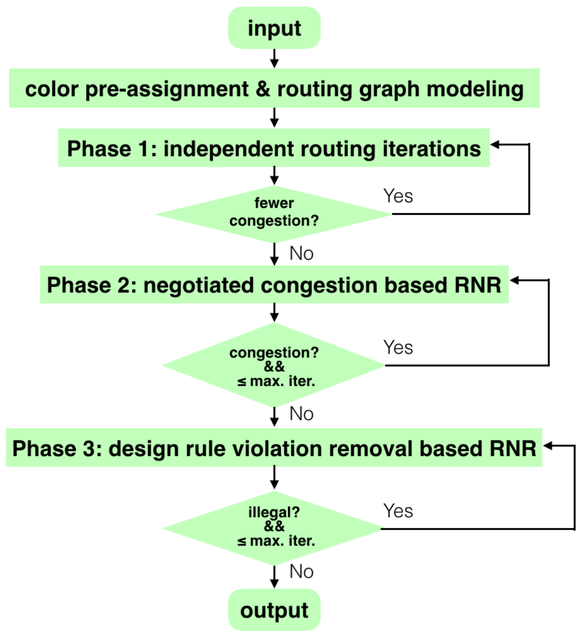
input
color pre-assignment & routing graph modeling
Phase 1: independent routing iterations
fewer congestion?
Yes
No
Phase 2: negotiated congestion based RNR
congestion? && ≤ max. iter.
Yes
No
Phase 3: design rule violation removal based RNR
illegal? && ≤ max. iter.
Yes
No
output

Fig. 7. SADP-aware detailed routing flow.

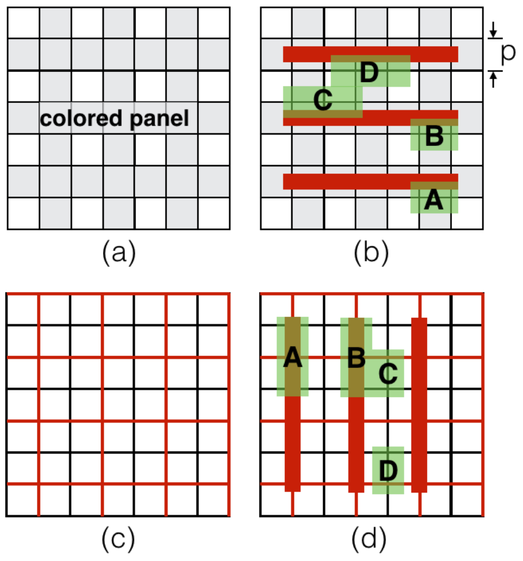

Fig. 8. Color pre-assignment for SIM and SID type SADP-aware detailed routing. (a)(b) SIM type. (c)(d) SID type.

will explain how we pre-assign colors over the routing grid. Since the color pre-assignment is performed on each layer individually, it can be applied to a multi-layer routing grid with different pitch sizes for each layer. Note that the way we do color pre-assignment and the resulting routing restrictions for SIM and SID types SADP-aware detailed routing are different, which will be described as follows.

1) SIM type: On each layer, we define a panel as the area between two adjacent horizontal (vertical) grid lines. We pre-assign colors to the panels alternately in both horizontal and vertical directions. Fig. 8(a) shows the colored routing grid prepared for SIM type SADP-aware detailed routing. The colored panels specify where the mandrel patterns may

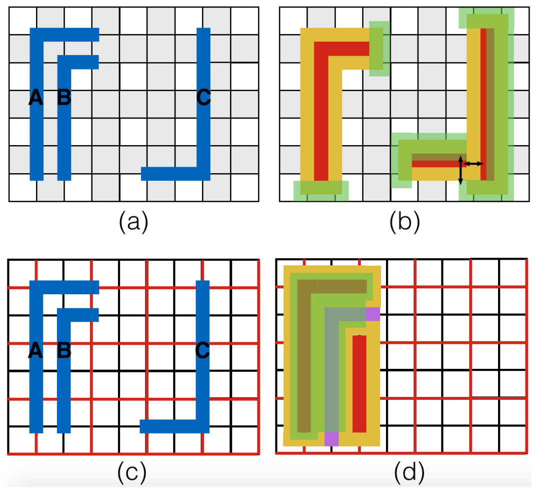

Fig. 9. SIM type and SID type SADP-aware detailed routing restrictions due to the color pre-assignment. (a)(b) SIM type. (c)(d) SID type.

be formed. Meanwhile, mandrel pattern is required to be aligned in the middle of colored panel which is shown in Fig. 8(b). Suppose the pitch size of routing tracks p is given. To align the metal patterns along the routing tracks, we assume $w_m + w_{sp} = p$ where w_m is the width of mandrel, and w_{sp} is the width of spacer. Meanwhile, $w_m \geq W_c$ is required to maintain the minimum width rule. If we require that the pitch size of colored panels, which is $2p$, is larger than or equal to $S_c + w_m$, the minimum spacing rule of core mask is also maintained. Since similar lithography processes are implemented for both core and cut mask pattern fabrication, we assume $S_c \approx S_s$ and $W_c \approx W_s$. To have better side overlay control, we require $w_c = p$ where w_c is the width of cut mask. The cut mask patterns are aligned along the routing tracks and Fig. 8(b) shows four possible locations of cut mask patterns A, B, C, and D. For the pair of patterns A and pattern B, their pitch size is $2p$, thus the design rules are maintained. Under the condition of no prohibited anti-parallel line-ends in the layout pattern, the pair of B and C are separated with enough spacing, while the pair of C and D can merge. With color pre-assignment and above assumption, the SADP layout decomposition becomes straightforward and SADP design rule are easy to maintain.

2) SID type: Different from SIM type, we pre-assign color to routing tracks alternately in both horizontal and vertical directions. Fig 8(c) shows the routing grid with color pre-assignment which prepares for SID type SADP-aware detailed routing. The mandrel patterns are required to be formed only along red tracks and should be aligned in the center, which is shown in Fig. 8(d). Thus, the routing layout patterns along red tracks are made directly from mandrels, while patterns along black tracks are made from sub-metals. With similar geometric assumption for SIM type, the design rules of core mask and trim mask can be maintained. Fig. 8(d) shows four possible locations of trim mask patterns A, B, C, and D. Under the condition of no prohibited anti-parallel line-ends and anti line-ends in the layout, any pair of trim mask patterns can be either separated with enough spacing or merged.

3) Routing restrictions: The color pre-assignment restricts where mandrel patterns can be formed for both SIM and SID types SADP layout decomposition. Under such restrictions, some routing patterns are not manufacturable due to the design rule violations, thus should be forbidden during detailed routing. Fig. 9(a) shows three L-shape routing patterns A, B, and C, Fig. 9(b) shows their corresponding SIM type SADP layout decomposition. A is a non-preferred turn due to the spacer rounding issue at a convex mandrel corner. B is formed from spacer deposited at a concave mandrel corner, thus can be manufactured without any degradation. We refer to it as a preferred turn. The layout decomposition for C cannot avoid design rule violations. As shown in Fig. 9(b), both core and cut mask violate the minimum spacing rule. Hence, we refer to the L-shape pattern C as a forbidden turn, and should be strictly avoided during detailed routing. Fig. 9(c)(d) show the same L-shape routing patterns and their corresponding SID type SADP layout decomposition. A is directly defined by mandrel which is free from spacing rounding issue. We refer to it as a preferred turn. B is a non-preferred turn since it is formed from sub-metal. The mandrel and trim mask patterns cannot even be designed for C under the restrictions of color pre-assignment. There, the pattern C is referred as a forbidden turn during detailed routing.

From the examples above, it is observed that how a routing pattern turns at the grid point determines its manufacturability. We have a forbidden turn which cannot be manufactured, and a preferred turn and a non-preferred turn which can be manufactured without degradation and with degradation, respectively. In the next subsection, we will introduce the graph model used in detailed routing which captures manufacturability of routing patterns exactly.

C. Graph model

In this section, we introduce our graph model which captures both routing cost and manufacturability of routing pattern in detailed routing. In addition, the graph model complexity is linear with the size of routing grid in which the constant factor can be kept small in practice. The graph models for SIM and SID type SADP-aware detailed routing are slightly different which will be described separately as follows.

1) SIM type: Suppose we are given a multi-layer routing grid with color pre-assignment and a preferred routing direction for each layer. We construct our routing graph G by viewing each grid segment or via as a vertex. An edge exists between two vertices if they are directly connected in the routing grid. A cost is associated with each edge to indicate the cost of traveling from the vertex in one end to the vertex in the other end. The construction of G is described below by considering graph models for pin, via, and grid segment separately. Fig. 10 shows the graph models together with five types of edges marked with different colors, including via, unit wirelength, preferred turn, non-preferred turn, and forbidden turn. Fig. 10 (a) shows the via model for a via accessible by eight grid segments from its upper routing layer and lower routing layer. In the via model, an edge exists between the vertex representing the via and the vertex representing an

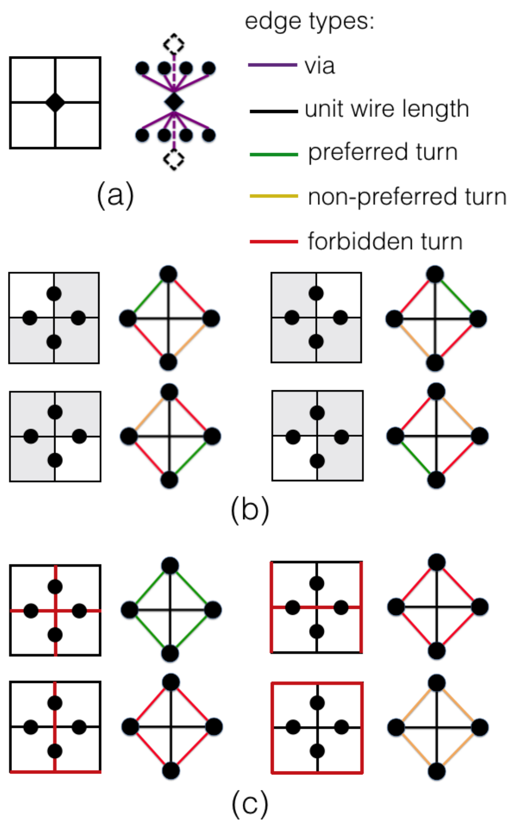

Fig. 10. Graph modeling for SADP-aware detailed routing. (a) via model. (b) SIM type grid segment models. (c) SID type grid segment models.

accessible grid segment. In addition, if a via layer exist above (below) the via, there will be an edge connecting the vertex representing the via and the vertex representing an accessible via from its upper (lower) via layer. The cost of the via edge is user-defined. In the routing grid with SIM type color pre-assignment, four types of grid point can be identified by the relative position of the uncolored grid square. Fig. 10(b) shows the grid segment models for the four grid segments incident to each type of grid point. In each grid segment model, four edge types can be assigned with different costs to capture relative routing and pattern manufacturability expenses. Note that the cost of the forbidden turn edges should be very high to avoid a routing pattern containing a forbidden turn.

2) SID type: The only difference between the graph models for SIM type and SID type are the grid segment models. In the colored routing grid prepared for SID type SADP-aware detailed routing, four types of grid points are identified by the colors of two intersected tracks. For a better illustration, we use a pair to represent each type of grid points, where its first member denotes the color of the horizontal track and the second member denotes the color of the vertical track. Fig. 10(c) shows the grid segment models which are characterized by four types of grid points, namely $\langle red, red\rangle$, $\langle red, black\rangle$, $\langle black, red\rangle$, and $\langle black, black\rangle$. The grid segment model characterized by$\langle red, red\rangle$ contains edges with the preferred turn type while the grid segment model characterized by$\langle black, black\rangle$ contains non-preferred turn type edges. For the grid segment models characterized by the other two

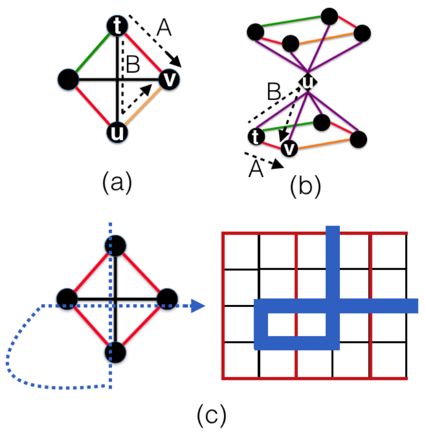

Fig. 11. Invalid routes. (a)(b) A route containing a forbidden turn. (c) A route containing a loop structure.

types, it contains edges with the forbidden turn type. Similar to the grid segment models for SIM type, different types of edges can be assigned with different costs to capture both routing and manufacturing expenses.

Algorithm 1 Modified Dijkstra's algorithm

```
Input: $net$ with source $s$ and sink $t$
Output: $path$ for $net$
  initialize($G$, $s$) and priority queue $PQ$
  while $dist(t)$==$\infty$ do
    vertex $u$ = PQ.dequeue()
    for each $v$ such that $(u, v) \in G.E$ do
      if $(pre(u), v) \notin G.E$ then
        relax($u$, $v$)
      end if
    end for
  end while
```

By applying the proposed graph model, Dijkstra's algorithm can be used to find a path for each net. Given a pin, the vertices in G representing the grid segments and vias which are accessible to it are treated as source/sink nodes. However, the path computed by Dijkstra's algorithm may not be a valid route in the routing grid. Fig.11 shows three cases of invalid routes. As shown in Fig.11(a), to avoid the path segment A which contains a forbidden turn edge, Dijkstra's algorithm will choose a path with the path segment B instead. The reason is that the cost of a unit wirelength edge together with a non-preferred turn edge is less than that of a forbidden turn edge. However, the corresponding route in the routing grid is not valid since it contains a forbidden turn. Similar issue occurs in Fig. 11(b). Thus, we resolve those issues by modifying Dijkstra's algorithm which is shown in Algorithm 1. Whenever we perform relaxation procedure on an edge (u, v) to update the minimum cost for vertex u, we check whether vertex presenting u's predecessor is connected to vertex v in G. If so, the relaxation is abandoned. In addition, the path computed by the modified Dijkstra's algorithm might contains a loop as shown in Fig. 11(c). Thus, once a loop structure is found in a routing path, we increase the cost to use the routing resources along the loop. Then, we rip-up and reroute the net by the modified Dijkstra's algorithm until no loop is found in the path. In this way, every path in the routing solution is ensured to be a valid route in the routing grid.

To speedup the modified Dijkstra's algorithm, a routing space-based heuristic is applied in our implementation. Given a net, a 3D routing space $L \times W \times H$ is determined before routing, where L and W are the length and width of the bounding box of its pins, and H is the number of routing layers. Then, the xy-dimension of the routing space is gradually enlarged during routing iterations. Specifically, $L \times = 1.2^n$ and $W \times = 1.2^n$, where n is the number of iterations in phase 1 or the number of RNR times of the net in phase 2 and 3. The modified Dijkstra's algorithm can only search path within the routing space. If the path computed by the modified Dijkstra's algorithm has a cost bigger than the pre-set value, the routing space will be further enlarged and the modified Dijkstra's algorithm will search the path for the net again. The iterations will continue until a path is obtained with a cost within the pre-set value.

D. *Overall routing scheme*

Negotiated congestion based routing scheme has been shown to be very effective on many routing problems, including FPGA routing [17] and IC global routing [18], [19], [20] Recently, [21] further demonstrates its success on the escape routing problem. In this paper, we apply the negotiated congestion based routing scheme for SADP-aware detailed routing. A congestion occurs when the paths of more than one routed nets go through the same grid point in the routing grid. We refer to it as the grid point congestion. The negotiated congestion based routing scheme targets to resolve all grid point congestions to obtain a congestion free routing solution. Every time a net is routed, if its path causes any grid point congestions with the previously routed nets, we increase the cost to use the routing resources corresponding to the grid point congestion. Thus, the net with more alternative routes tends to detour from the congested routing resources in the RNR. In this way, the grid point congestion can be potentially resolved. One major advantage of negotiated congestion based routing scheme is that the route of each net is never committed as final routing solution until the entire flow terminates. Nets causing congestions or design rule violations will be ripped-up and rerouted. All the nets will negotiate with each other for using the routing resources to find their own routes. This is the reason our routing scheme does not depend on a particular net ordering. To realize such scheme, we incorporate the other two cost components into our graph model.

$$Cost_e = BC_e + UC_e + HC_e^i$$
$$UC_e = \alpha \times Usage(p)$$
$$HC_e^i = HC_e^{i-1} + \beta \times Overflow(p)$$

$Cost_e$ is the total cost of an edge e in G. BC_e denotes the base cost of e, which is determined by the edge type. p is a grid point in the routing grid shared by two grid segments/vias corresponding to the two vertices of e. UC_e is the usage cost indicating occupation of routing resources. It is updated after a net is routed or ripped-up and equal to the weighted current usage at p. HC_e^i is the history cost after iteration i indicating historical congestion information. It is updated once a grid point congestion is detected at p and can be computed by accumulating HC_e^{i-1} with weighted overflow at p.

Algorithm 2 SADP-aware detailed routing

```
Input: netlist, a routing graph, and SADP design rules
Output: SADP friendly detailed routing solution
  Phase 1: independent routing iterations
  block routing resources occupied by all the pins from netlist
  while fewer grid point congestions do
    for each $net_i$ in netlist do
      unblock routing resources occupied by pins of $net_i$
      the modified Dijkstra's algorithm finds $path_i$ for $net_i$
      block routing resources occupied by pins of $net_i$
      update UC for $path_i$ ($\alpha$ is extremely small)
    end for
    update HC for all grid point congestions
    remove all UC in the $G$
  end while
  Phase 2: negotiated congestion based RNR iterations
  update UC for all the paths in netlist
  build a queue $Q$ containing all grid point congestions
  while !isEmpty(Q) && #iter. $\leq$ max. #iter. do
    a grid point congestion $c$ = Q.dequeue()
    chose rip-up net $net_j$ which causes the $c$
    update UC after removing $path_j$ of $net_j$
    the modified Dijkstra's algorithm finds $path'_j$ for $net_j$
    update UC for $path'_j$
    if reroute of $net_j$ causes a grid point congestion $c'$ then
      update HC for the $c'$
      Q.enqueue($c'$)
    end if
  end while
  Phase 3: design rule violation based RNR iterations
  build a priority queue $PQ$ containing all prohibited line-ends violations
  block vias within prohibited line-ends regions
  while !isEmpty(PQ) && #iter. $\leq$ max. #iter. do
    violation = PQ.dequeue()
    if violation is prohibited anti-parallel line-ends && line-end extension is allowed then
      do line-end extension
    else
      rip-up and reroute
      if reroute causes a grid point congestion $c$ then
        PQ.enqueue($c$)
      end if
    end if
  end while
```

As shown in Algorithm 2, our SADP-aware detailed routing has three major phases. The first phase is independent routing iterations. In this phase, the routing of each net does not consider the already routed nets, i.e., no additional penalty will be charged to use the routing resources occupied by routed nets when finding a path for a net. We refer to such

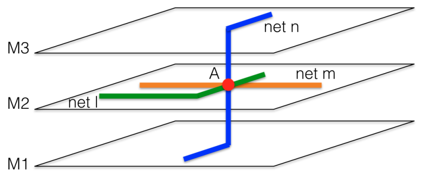

Fig. 12. Heuristic to find the rip-up net.

routing as independent routing. In each iteration, we route all the nets from the netlist by independent routing. The reason for independent routing is to minimize the negative effect of net ordering in sequential routing. A couple of heuristics are applied here to obtain a better routing solution in fewer number of iterations, which are explained as follows.

- In a valid detailed routing solution, the path of every net should not go through pins of any other nets. Thus, before the start of independent routing iterations, we block the routing resources occupied by all pins of netlist. Every time we route a net, we firstly unblock routing resources occupied by the pins of the net, then modified Dijkstra's algorithm is applied to find a path. After that, we re-block routing resources occupied by the pins of the net and prepare for the routing of the next net. In this way, potential routing congestions will be avoided.
- As shown in Algorithm 1, every net is routed by performing the modified Dijkstra's algorithm from source to sink nodes. It is possible that multiple optimal solutions exist for a net. Thus, choosing one optimal solution which causes fewest number of grid point congestions with previously already routed nets will probably reduce total number of grid point congestions in the end of iteration. Therefore, every time after we route a net, we will update usage cost for the path of the net. The value of α used to calculate usage cost is set extremely small such that multiple optimal solutions can be differentiated. By applying this heuristic, the routing of each net is not totally independent from the previously routed nets. However, the value of α is so small, the usage cost update will never make the original non-optimal solutions optimal.

After all the nets from the netlist are routed in one iteration, we update the history cost for all grid point congestions so that the congested routing resources are more expensive to use in the next iteration. This phase will terminate if the grid point congestion count of current iteration is more than that of the last iteration.

The next phase is negotiated congestion based RNR iterations. The target of this phase is to eliminate all the grid point congestions by RNR. Initially, a queue Q is built and it includes all the identified grid point congestions. At each iteration, one grid point congestion is popped up from Q, then a rip-up net causing this grid point congestion is chosen. After that, a reroute aiming to avoid grid point congestion in the new path is performed for the rip-up net. As a result, the grid point congestion will probably be resolved over iterations, and a detailed routing solution without any congestion could

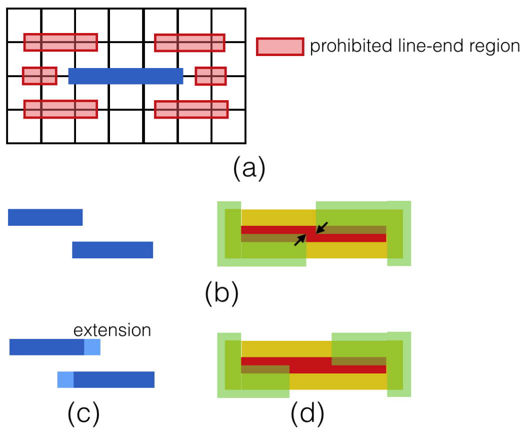

Fig. 13. (a) Prohibited line-end region. (b) Prohibited anti-parallel line-ends cause cut mask design rule violation in SIM type SADP lithography. (c) Line end extension is performed. (d) No design rule violation in SIM type SADP lithography.

be obtained. In practice, we find that the choice of rip-up net is critical to the effectiveness of RNR. Thus, we develop a heuristic to find a better rip-up net. As shown in Fig. 12, the paths of three routed nets m, l, and n all go through the grid point A, thus a grid point congestion is detected and targeted to be resolved. Suppose the costs of paths for net l, m, and n are c_m, c_l, and c_n, respectively. We firstly do a trial reroute to obtain the costs of new paths for nets m, l, and n, which are c'_m, c'_l, and c'_n. We define $\Delta reroute = c'_i - c_i$, where $i = \{m, l, n\}$. Then, we chose the net with smallest $\Delta reroute$ as the rip-up net. The heuristic greatly helps us to obtain a congestion free routing solution with fewer number of RNR iterations. Given the rip-up net, the modified Dijkstra's algorithm is performed to compute the new path for it and usage cost is updated due to the RNR. In the case of the new path obtained in the reroute could not avoid grid point congestion, we will update history cost for the grid point congestion, and push it to Q for later fix. This phase will continue until PQ is empty or current iteration count reaches the pre-set maximum iteration count. A congestion is reported if it exists.

The last phase is design rule violation based RNR iterations. The target of this phase is to resolve all prohibited line-ends while ensuring a congestion free routing solution. Given a metal pattern, we define a prohibited line-ends region for its line ends which is shown in Fig. 13(a). Other metal patterns within this region are checked if they form a prohibited line-ends with it. Thus, time complexity of identifying all the prohibited line-ends in the layout pattern is $O(n)$, where n is number of metal lines in the layout. Then, a priority queue PQ used to keep violations is built. The violation can either be a prohibited line-ends or a grid point congestion. Initially, the PQ contains all the identified prohibited line-ends in the layout. At each iteration, a violation is popped up from PQ and whether it is a prohibited anti-parallel line-ends is checked. Fig. 13(b) shows an example of prohibited anti-parallel line-ends in SIM type SADP layout manufacturing. [22] states that the line-end extension is an option to resolve the manufacturing challenging of the prohibited anti-parallel line-ends, as illustrate in Fig. 13(c)(d). Thus, if line-end extension is allowed, it is performed to resolve the violation. Otherwise, we will rip-up and reroute the net causing the violation. To avoid creating a new prohibited line-ends in the reroute, some routing resources around line ends are blocked. Specifically, all the vias within the prohibited line-ends region are blocked. This approach is a little restrictive. However, most of prohibited line-ends can be resolved by line-end extension, and not so many RNR iterations are called in this phase. In practice, all design rule violations can be resolved without degrading routing solution much. However, the reroute net could still cause a grid point congestion. If it is the case, the grid point congestion is pushed to PQ. To maintain a congestion free routing solution, the violation with the type of grid point congestion has higher priority in PQ than that of prohibited line-ends. This phase will continue until PQ is empty or current iteration count reaches the pre-set maximum iteration count. Finally, detailed routing solution compliant to SADP design rules is generated. Congestions or design rule violations will be reported if they exist.

V. Experimental results

Our SADP-aware detailed routing is implemented by C++ programming language and with an option to choose either SIM or SID type SADP lithography. We run all the experiments on a machine with a 2.4 GHz Intel Core i5 CPU and 8 GB memory. We compare with three previous works [4], [11], and [12]. [4] is LELE-aware detailed routing also applying the idea of color pre-assignment. [11] is the SID type SADP-aware detailed routing using cut process instead of trim process. [12] is the latest work on SID type SADP-aware detailed routing. The benchmark suite used in each comparison is provided by the authors of each paper. The statistics of each benchmark, including the number of nets and size of routing grid are listed in Table I, Table II, and Table III respectively. In our experiments, we set the p = 48nm, w_m = w_{sp} = 24nm, S_c = 72nm, and S_s = 64nm. Note that the parameters, including edge cost in the G and α and β values, are always kept the same values in the experiments for each benchmark suite. Table IV lists all the parameter values used in each set of experiments. and the results are listed in Table V, Table VI, and Table VII. Furthermore, we perform another three sets of experiments. Firstly, we show the effectiveness of our proposed RNR heuristic, and experimental results are in Table VIII. Secondly, we investigate the solution quality and convergence of SADP-aware detailed routing under different parameter settings, which is shown in Fig. 14. Finally, we analyze the advantages of our color pre-assignment approach, and experimental results are in Table IX.

A. Compare with Seong-I Lei et al. [4]

The motivation to compare our algorithm with [4] is that both of the works adopt the idea of color pre-assignment for detailed routing. In [4], each routing track is assigned with one of the two colors and adjacent tracks in horizontal (or vertical) direction are assigned with different colors.

TABLE I
STATISTICS OF BENCHMARKS FROM [4]

Benchmark	C1	C2	C3	C4	T1	T2	T3	T4
#Nets	1500	10000	1927	2400	869	1036	1763	3017
Grid size	100 × 100	300 × 300	400 × 400	400 × 400	500 × 500	600 × 600	800 × 800	1000 × 1000

TABLE II
STATISTICS OF BENCHMARKS FROM [11]

Benchmark	Test1	Test2	Test3	Test4	Test5	Test6	Test7	Test8	Test9	Test10
#Nets	1000	1800	4000	8000	12000	1000	1800	4000	8000	12000
Grid size	240 × 240	340 × 340	400 × 400	600 × 600	900 × 900	240 × 240	340 × 340	400 × 400	600 × 600	900 × 900

TABLE III
STATISTICS OF BENCHMARKS FROM [12]

Benchmark	ecc	efc	ctl	alu	div	top
#Nets	1671	2219	2706	3108	5813	22201
Grid size	436 × 446	406 × 421	496 × 503	406 × 408	636 × 646	1176 × 1179

TABLE IV
PARAMETER VALUES IN THE EXPERIMENTS

Parameters	Via	unit WL	Preferred turn	Non-preferred turn	Forbidden turn	α	β
Section V.A	4	1	16	20	$INT\_MAX/8$	4	1
Section V.B	12	1	1	1	$INT\_MAX/8$	4	2
Sections V.C, V.D, and V.E	4	1	$INT\_MAX/8$	$INT\_MAX/8$	$INT\_MAX/8$	4	8

The idea greatly reduces coloring conflicts in LELE layout decomposition. However, there are several intrinsic differences between LELE and SADP. Firstly, LELE allows using stitches to resolve conflicts in layout decomposition where SADP does not. Secondly, LELE has a major disadvantage that it has worse overlay control due to the easy misalignment of two masks. Finally, SADP targets at 10nm technology node while LELE targets at 16/14nm technology node. Thus, more design rules need to be considered in SADP-aware detailed routing. For example, we need to consider the non-preferred turn minimization which will restrict solution space of detailed routing. The prohibited line-ends is not allowed in the routing solution which will further restrict detailed routing. Each benchmark has four routing layers. In benchmarks C1-C4, each pin only covers one grid point while each pin might cover several grid points (usually 2-5 grid points) in benchmarks T1-T4. Table V shows the performance between [4] and our algorithm, where "#S" reports the number of stitches, and "#NPT" gives the number of non-preferred jogs. Compared with [4], our approach for both SIM and SID type SADP lithography can generate detailed routing solution with almost same quality in terms of total wirelength and via count. The non-preferred turn count can be minimized by our algorithm to a very small number. [4] uses a 3GHz Linux machine with 64 GB memory. Given the different machine speeds, our algorithm is estimated more than 3X faster.

B. Compare with Iou-Jen Liu et al. [11]

[11] is the overlay-aware detailed routing for SID type SADP lithography using cut process. [11] claims the cut process has higher design flexibility over trim process. However, it inevitably introduces side overlay error whenever a cut pattern overlaps with a section of a feature side boundary. In contrast, SIM type SADP does not introduce any side overlay since all features are formed by spacers. Moreover, we ensure all the features' side boundaries are protected by spacer in our SID type SADP layout decomposition. Table VI compares the performance of our algorithm and [11], where "VPN" denotes via count per net, and "OLL" reports the total side overlay length. In benchmarks Test1-Test5, the locations of the source and sink pins of each two-pin net are fixed while the source and sink pins of every two-pin net have multiple candidate locations in Test6 - Test10. There are three routing layers from M1 to M3 in each benchmark, and each layer does not have a preferred routing direction. [11] performs the detailed routing in all routing layers, but most of the wires are routed on M1. This is the reason why they have such small "VPN" in the detailed routing solution as shown in Table VI. For a fair comparison, we remove the penalty for routing in non-preferred routing direction and disable the non-preferred turn minimization in our SADP-aware detailed routing. Compared with [11], our detailed routing for SIM and SID type SADP lithography both reduce total wirelength by 21%. [11] tries to complete detailed routing using a single layer, thus "VPN" is very small in the experimental results. The "VPN" in our routing solution is slightly larger than that in [11]. However, the value is actually small enough, and on average 0.08 in SIM type and 0.12 in SID type. Furthermore, our algorithm achieves 100% routability while [11] cannot route all the nets successfully for all the benchmarks. It shall be noted that the total wirelength and "VPN" for [11] will increase if 100% routability could be achieved. Finally, although [11] tries to minimize the amount of side overlay error, they cannot completely eliminate it. On average 2.2% of total wirelength (462.8 over 20731.7) is generated with side overlay error. On the other hand, our approach produces no side overlay error at all. This greatly improves the yield and reduces circuit performance variability. Since it is a joint work with TSMC,

TABLE V
COMPARE WITH SEONG-I LEI ET AL. [4]

	Seong-I Lei et al. [4]					SIM type SADP-aware detailed routing					SID type SADP-aware detailed routing				
Benchmark	WL	#Vias	#S	%Rout.	CPU(s)	WL	#Vias	#NPT	%Rout.	CPU(s)	WL	#Vias	#NPT	%Rout.	CPU(s)
C1	9054	3900	0	100	11	9196	4568	1	100	2	9202	4544	2	100	1
C2	60325	23036	0	100	60	61039	24906	2	100	10	61019	24728	4	100	10
C3	64347	4084	0	100	42	64415	4108	0	100	22	64417	4106	0	100	24
C4	64331	5124	0	100	42	64401	5142	0	100	17	64415	5154	0	100	22
T1	99146	2092	0	100	122	98984	2002	0	100	79	98989	2000	0	100	70
T2	128429	2480	0	100	819	128250	2352	0	100	113	128251	2362	0	100	107
T3	215035	4060	0	100	539	214697	3936	0	100	174	214688	3940	0	100	178
T4	194716	6306	0	100	159	193940	6144	0	100	105	193948	6140	0	100	112
Average	104422.9	6385.3	0	100.0	224.3	104365.3	6644.8	0.4	100.0	65.3	104366.1	6621.8	0.8	100.0	65.5
Normalized	1.00	1.00		1.00	1.00	1.00	1.04		1.00	0.29	1.00	1.04		1.00	0.29

[11] is not able to release the binary. Thus, we have to compare the runtime of two algorithms on different machines. [11] uses a 2.93 GHz Linux work station with 48GB memory. Given the different machine speeds, our algorithm is estimated more than 2X faster. [11] maintains a constraint graph during detailed routing to minimize overlay error and resolve design rule violation. One of the major reasons that we can have such speedup is that we totally avoid it due to the color pre-assignment approach.

C. Compare with Xiaoqing Xu et al. [12]

[12] is the most recently published SID type SADP-aware detailed routing. Table VII are the experimental results of the comparison. All the benchmarks have three layers, where M1 is not allowed for routing. [12] performed the unidirectional routing, and routing direction for M2 and M3 are horizontal and vertical, respectively. In addition, [12] considers the via rule in which no two vias can be inserted within certain spacing. To have a fair comparison, we also disallow routing on M1 and routing not in routing direction on M2 and M3. Since our SADP-aware detailed routing only considers different-net via rule which requires no two vias from different nets can be inserted within certain spacing. Thus, we ask the author of [12] to disable the via rule and generate experimental results. Meanwhile, we disable the different-net via rule to generate experimental results. Furthermore, we ensure the way to measure wirelength and via count are same with [12]. Both [12] and our algorithm can generate routing solution with no side overlay error in SADP layout decomposition. Thus, we do not show OLL in Table VII. Compared with [12], both of our SIM type and SID type SADP-aware detailed routing reduce total wirelength by 23%. Both of our SIM and SID type SADP-aware detailed routing reduce via count by 10% comparing with [12]. Furthermore, we can achieve 100% routability for all the benchmarks while [12] fails to route all the nets in each benchmark. These unroutable nets will further enlarge the difference of total wirelength and via count between our algorithm and [12]. [12] uses a Linux machine with 3.4GHz Intel(R) Core and 32GB memory to generate experimental results. Given the different machine speeds, our algorithm is estimated more than 4X faster.

D. Demonstrate effectiveness of RNR heuristic

In this subsection, we will demonstrate the effectiveness of our proposed heuristic for choosing the rip-up net during RNR iterations. We choose our SIM type SADP-aware detailed routing on benchmarks from [12] as the baseline. To compare with it, we disable the heuristic and always randomly choose the rip-up net during RNR iterations. Table VIII shows the comparison results, where "#C" reports the number of grid point congestions, and "#DRV" gives the number of design rule violations in the layout. As shown in the Table VIII, compared with the baseline, the routability is reduced by more than 1% without proposed heuristic. Congestions can not be resolved for all the benchmarks, and design rule violations may occur in the layout. Meanwhile, the total wirelength and via count also increase. This is because a bad choice of rip-up net is more likely to have more detour in the reroute. Thus, quality of routing solution is degraded. However, our proposed heuristic has almost 20% runtime overhead.

E. Demonstrate the solution quality and convergence of our routing scheme

In this subsection, we will demonstrate the solution quality and convergence of our SADP-aware detailed routing under different parameter settings. Several user-defined parameters are used in our SADP-aware detailed routers, which are shown in Table IV. Based on our observation on all the experiments above, the unit wirelength cost value has little effect on the routing solution. Thus, we keep its value as one in all the experiments. Meanwhile, increasing (decreasing) via cost will reduce (raise) via count within a certain range in the routing solution. To investigate the impact of α and β, we choose benchmark "div" in Table VII and run our SADP-aware detailed routing with different values. Note that different from the experiments in Section V.C, we enable our different-net via rule. We use a pair (α, β) to denote the values of α and β in a run. Fig. 14 contains four plots showing solution quality and convergence of our SIM type SADP-aware detailed routing, include total wirelength, via count, total number of RNR iterations, and CPU time. Each point in the plots is a run with different α and β values. Since all the runs can achieve 100% routability, we do not show the routability plot here. Generally speaking, if the α and β are set smaller values, our SADP-aware detailed routing can achieve little better solution quality in terms of wirelength and via count. However, more RNR iterations are needed in order to achieve 100% routability, and total runtime increases accordingly. Note that the impact of α and β on solution quality is actually small. The run with maximum wirelength is only about 0.5% more than the run

TABLE VI
COMPARE WITH IOU-JEN LIU ET AL. [11]

	Iou-Jen Liu et al. [11]					SIM type SADP-aware detailed routing					SID type SADP-aware detailed routing				
Benchmark	WL	VPN	OLL	%Rout.	CPU(s)	WL	VPN	OLL	%Rout.	CPU(s)	WL	VPN	OLL	%Rout.	CPU(s)
Test1	4610	0.03	104	95.3	0.4	4186	0.12	0	100	1.0	3926	0.34	0	100	0.5
Test2	7318	0.02	134	96.7	1.6	6041	0.12	0	100	1.8	5889	0.14	0	100	1.7
Test3	12115	0.02	268	97.2	6.0	8650	0.13	0	100	2.7	8438	0.19	0	100	2.9
Test4	26745	0.02	503	97.4	23.1	20210	0.14	0	100	8.7	19656	0.18	0	100	6.8
Test5	40204	0.01	424	98.3	56.2	29598	0.05	0	100	23.5	28924	0.11	0	100	15.8
Test6	5198	0.06	209	96.1	0.3	3891	0.05	0	100	1.5	3868	0.04	0	100	0.7
Test7	9108	0.04	260	96.9	1.2	6580	0.02	0	100	1.7	6584	0.03	0	100	1.9
Test8	17397	0.03	642	95.6	5.4	13569	0.06	0	100	3.6	13717	0.08	0	100	4.9
Test9	33589	0.03	1040	96.2	24.5	33931	0.05	0	100	12.9	35534	0.08	0	100	19.2
Test10	51051	0.02	1044	97.8	54.8	37301	0.01	0	100	17.2	37336	0.03	0	100	17.3
Average	20731.7	0.03	462.8	96.8	17.4	16385.7	0.08	0	100.0	7.5	16387.2	0.12	0	100.0	7.2
Normalized	1.00	1.00		1.00	1.00	0.79	3.95		1.03	0.43	0.79	4.56		1.03	0.41

TABLE VII
COMPARE WITH XIAOQING XU ET AL. [12]

	Xiaoqing Xu et al. [12]				SIM type SADP-aware detailed routing				SID type SADP-aware detailed routing			
Benchmark	WL	#Vias	%Rout.	CPU(s)	WL	#Vias	%Rout.	CPU(s)	WL	#Vias	%Rout.	CPU(s)
ecc	45975	6441	96.41	21.6	35090	5490	100	9.1	35044	5492	100	9.9
efc	57458	8516	94.05	36.7	46477	7973	100	13.8	46463	7972	100	13.9
ctl	71869	10616	95.27	36.5	57739	9445	100	15.4	57690	9445	100	18.1
alu	74568	11525	94.18	48.3	57236	10402	100	14.9	57265	10404	100	15.4
div	155602	22967	94.51	99.5	121983	20787	100	40.1	122087	20789	100	39.3
top	506072	82132	94.54	664.5	379539	73999	100	121.2	380008	73992	100	114.9
Average	151924.0	23699.5	94.8	151.2	116344.0	21349.3	100.0	35.8	116426.2	21349.0	100.0	35.3
Normalized	1.00	1.00	1.00	1.00	0.77	0.90	1.06	0.24	0.77	0.90	1.06	0.23

TABLE VIII
DEMONSTRATE EFFECTIVENESS OF PROPOSED RNR HEURISTIC

	With RNR heuristic				Without RNR heuristic					
Benchmark	WL	#Vias	%Rout.	CPU(s)	WL	#Vias	%Rout.	#C	#DRV	CPU(s)
ecc	35090	5490	100	9.1	35094	5528	99.28	11	0	9.8
efc	46477	7973	100	13.8	46625	8042	98.02	36	1	10.9
ctl	57739	9445	100	15.4	57870	9554	99.22	13	0	14.8
alu	57236	10402	100	14.9	57441	10604	98.94	33	0	11.6
div	121983	20787	100	40.1	122468	21136	98.66	46	0	32.4
top	379539	73999	100	121.2	381468	75434	98.59	193	7	94.7
Average	116344.0	21349.3	100.0	35.8	116827.7	21716.3	0.99	55.3	1.3	29.0
Normalized	1.00	1.00	1.00	1.00	1.00	1.02	0.99			0.81

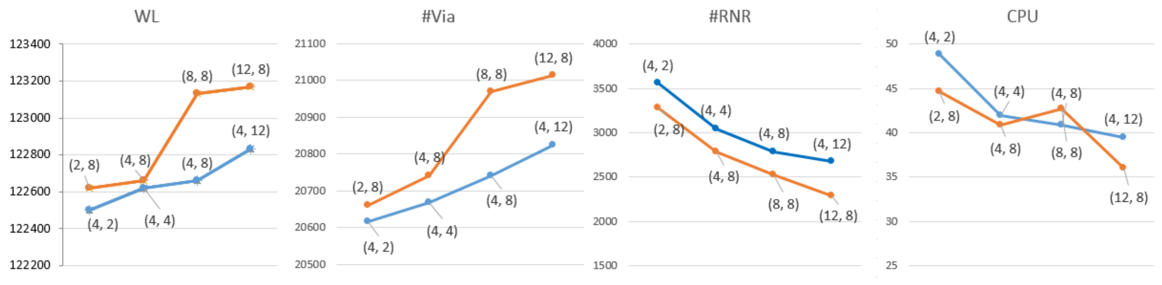

Fig. 14. The solution quality and convergence of our SADP-aware detailed routing with different parameter settings.

with minimum wirelength. At the same time, the run with maximum via count is only about 1.8% more than the run with minimum via count. Consequently, the solution quality of our SADP-aware detailed routing has little dependence on the setting of α and β values.

F. Demonstrate effectiveness of color pre-assignment approach

Color pre-assignment is the key idea and innovation of our SADP-aware detailed routing. To demonstrate its effectiveness, we choose the pure detailed routing without color pre-assignment as a baseline. In the baseline, we only run our routing scheme with phase 1 and 2, and report wirelength, via count, routability, and runtime. We compare our SADP-aware detailed routing with the baseline on benchmark suite from [12]. The experimental results are in Table IX. Compared with the baseline, SADP-aware detailed routing has 2% wirelength increase, 1%via count increase, and 28% runtime increase. Consequently, color pre-assignment approach helps us to achieve SADP decomposable routing solution without much overhead.

TABLE IX
DEMONSTRATE OF COLOR PRE-ASSIGNMENT APPROACH

	Pure detailed routing without color pre-assignment				SADP-aware detailed routing with color pre-assignment			
Benchmark	WL	#Vias	%Rout.	CPU(s)	WL	#Vias	%Rout.	CPU(s)
ecc	35549	5033	100	8.6	35853	5061	100	11.9
efc	46022	7828	100	13.4	46654	7883	100	17.0
ctl	57153	9260	100	14.3	57875	9323	100	18.4
alu	56958	10039	100	14.8	57888	10167	100	20.4
div	120890	20509	100	39.8	122660	20742	100	41.63
top	378970	70058	100	112.3	385368	70947	100	153.0
Average	115923.6	20454.5	100	33.9	117716.3	20687.1	100.0	43.76
Normalized	1.00	1.00	1.00	1.00	1.02	1.01	1.00	1.28

VI. CONCLUSION

In this paper, we propose a detailed routing algorithm for both SIM and SID type SADP lithography. We apply the color pre-assignment approach which greatly simplifies the SADP layout decomposition of routing pattern. The proposed graph model helps avoiding the design rule violation in detailed routing. Compared with other state-of-the-art SADP-aware detailed routing algorithms, we generate stronger experimental results in terms of total wirelength, routability, and runtime. Moreover, no side overlay error is ensured in our detailed routing solution for SADP layout manufacturing. This gives us an evidence that color pre-assignment is a highly effective approach to consider SADP lithography during detailed routing. For the future works, we have several directions. Firstly, this work assumes the SIM type SADP using cut process while SID type SADP using trim process. However, our work can be potentially extended to trim-based SIM type SADP and cut-based SID type SADP. Generally, the mask patterns in cut process are absolutely complement of mask patterns in trim process. Thus, several modifications should be made. For example, the definition of each prohibited line-ends configuration, and the way to identify the type of the edge in the graph model. Secondly, our SADP-aware detailed routing can be potentially speedup by parallelization, especially the independent routing iterations. Thirdly, our SADP-aware detailed routing can be potentially extended to non-uniform track structure. However, some modifications should be made. For example, the via model should be changed to handle the misalignment of routing tracks from different layers. Furthermore, we can consider mixed-width wires in our SID type SADP detailed routing, which will make our work more realistic to industrial designs. Finally, we want to further extend our color pre-assignment approach to detailed routing for self-aligned quadruple patterning (SAQP) lithography targeted at sub-10nm design.

ACKNOWLEDGMENT

This work was supported in part by the Ministry of Science and Technology under grant MOST 104-2628-E-007-003-MY3.

REFERENCES

[1] Yongchan Ban, Alex Miloslavsky, Kevin Lucas, Soo-Han Choi, Chul-Hong Park, and David Z. Pan, “Layout decomposition of self-aligned double patterning for 2D random logic patterning,” in *Proc. of Society of Photographic Instrumentation Engineers*, Feburary 2011.

[2] Kun Yuan, Jae-Seok Yang, and David Z. Pan, “Double patterning layout decomposition for simultaneous conflict and stitch minimization,” in *Proc. of International Symposium on Physical Design*, March 2009.

[3] Yue Xu and Chris Chu, “GREMA: Graph reduction based efficient mask assignment for double patterning technology,” in *Proc. of International Conference On Computer Aided Design*, November 2009.

[4] Seong-I Lei, Chris Chu, and Wai-Kei Mak, “Double patterning-aware detailed routing with mask usage balancing,” in *Proc. of International Symposium on Quality Electronic Design*, March 2014.

[5] Yongchan Ban, Kevin Lucas, and David Z. Pan, “Flexible 2D layout decomposition framework for spacer-type double pattering lithography,” in *Proc. of Design Automation Conference*, June 2011.

[6] Hongbo Zhang, Yuelin Du, Martin D. F. Wong, and Rasit Topaloglu, “Self-aligned double patterning decomposition for overlay minimization and hot spot detection,” in *Proc. of Design Automation Conference*, June 2011.

[7] Zigang Xiao, Yuelin Du, Hongbo Zhang, and Martin D. F. Wong, “A polynomial time exact algorithm for self-aligned double patterning layout decomposition,” in *Proc. of International Symposium on Physical Design*, March 2012.

[8] Jhih-Rong Gao and David Z. Pan, “Flexible self-aligned double patterning aware detailed routing with prescribed layout planning,” in *Proc. of International Symposium on Physical Design*, March 2012.

[9] Chikaaki Kodama, Hirotaka Ichikawa, Koichi Nakayama, Toshiya Kotani, Shigeki Nojima, Shoji Mimotogi, Shinji Miyamoto, and Atsushi Takahashi, “Self-aligned double and quadruple patterning-aware grid routing with hotspots control,” in *Proc. of Asia and South Pacific Design Automation Conference*, Janunary 2013.

[10] Yuelin Du, Qiang Ma, Hua Song, James Shiely, Gerard Luk-Pat, Alexander Miloslavsky, and Martin D. F. Wong, “Spacer-is-dielectric-compliant detailed routing for self-aligned double patterning lithography,” in *Proc. of Design Automation Conference*, June 2013.

[11] Iou-Jen Liu, Shao-Yun Fang, and Yao-Wen Chang, “Overlay-aware detailed routing for self-aligned double patterning lithography using the cut process,” in *Proc. of Design Automation Conference*, June 2014.

[12] Xiaoqing Xu, Bei Yu, Jhih-Rong Gao, Che-Lun Hsu, and David Z. Pan, “PARR: Pin access planning and regular routing for self-aligned double patterning,” in *Proc. of Design Automation Conference*, June 2015.

[13] Gerard Luk-Pat, Ben Painter, Alex Miloslavsky, Peter De Bisschop, Adam Beacham, and Kevin Lucas, “Avoiding wafer-print artifacts in spacer is dielectric (SID) patterning,” in *Proc. of Society of Photographic Instrumentation Engineers*, Feburary 2013.

[14] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Pan, “Self-aligned double patterning aware pin access and standard cell layout co-optimization,” in *Proc. of International Symposium on Physical Design*, March 2014.

[15] ——, “Self-aligned double patterning aware pin access and standard cell layout co-optimization,” *IEEE Trans. Computer-Aided Design Integrated Circuits Systems*, vol. 34, 2015.

[16] Yuelin Du, Hua Song, James Shiely, and Martin D. F. Wong, “Improved spacer-is-dielectric (SID) decomposition with model based verification,” in *Proc. of Society of Photographic Instrumentation Engineers*, 2013.

[17] L. McMurchie and C. Ebeling, “Pathfinder: A negotiation-based performance-driven router for fpgas,” in *Proc. of ISFPGA*, November 1995.

[18] Jarrod A. Roy and Igor L. Markov, “High-performance routing at the nanometer scale,” in *Proc. of International Conference On Computer Aided Design*, November 2007.

[19] Muhammet Mustafa Ozdal and Martin D. F. Wong, “Archer: a history-driven global routing algorithm,” in *Proc. of International Conference On Computer Aided Design*, November 2007.

[20] Minsik Cho, Katrina Lu, Kun Yuan, and David Z. Pan, "Boxrouter 2.0: architecture and implementation of a hybrid and robust global router," in *Proc. of International Conference On Computer Aided Design*, November 2007.

[21] Qiang Ma, Tan Yan, and Martin D. F. Wong, "A negotiated congestion based router for simultaneous escape routing," in *Proc. of International Symposium on Quality Electronic Design*, March 2010.

[22] Yixiao Ding, Chris Chu, and Wai-Kei Mak, "Throughput optimization for SADP and e-beam based manufacturing of 1D layout," in *Proc. of Design Automation Conference*, June 2014.

PLACE PHOTO HERE

Yixiao Ding (S'14) Yixiao Ding received the B.S. degree in electrical engineering and automation from Hunan University, Changsha, Hunan, China, in 2012. He is currently a Ph.D. candidate in Electrical and Computer Engineering Department at Iowa State University, Ames, Iowa, USA. His advisor is Prof. Chris Chu. His research interests include VLSI physical design and design for manufacturability.

PLACE PHOTO HERE

Chris Chu (M'99-F'12) Chris Chu received the B.S. degree in computer science from the University of Hong Kong, Hong Kong, in 1993. He received the M.S. degree and the Ph.D. degree in computer science from the University of Texas at Austin in 1994 and 1999, respectively. Dr. Chu is a Professor in the Electrical and Computer Engineering Department at Iowa State University. His area of expertises include CAD of VLSI physical design, and design and analysis of algorithms. Dr. Chu is currently an associate editor for IEEE TCAD. He has served on the technical program committees of several major conferences including DAC, ICCAD, ISPD, ISCAS, DATE, ASP-DAC, and SLIP. Dr. Chu received the IEEE TCAD best paper award at 1999 for his work in performance-driven interconnect optimization. He received another IEEE TCAD best paper award at 2010 for his work in routing tree construction. He received the ISPD best paper award at 2004 for his work in efficient placement algorithm. He received another ISPD best paper award at 2012 for his work in floorplan block shaping algorithm. He received the ASPDAC best paper award at 2014 for his work in stencil design for electron-beam lithography. He received the Bert Kay Best Dissertation Award for 1998-1999 from the Department of Computer Sciences in the University of Texas at Austin. He is a Fellow of IEEE.

PLACE PHOTO HERE

Wai-Kei Mak (M'98) Wai-Kei Mak received the B.S. degree from the University of Hong Kong, Hong Kong, in 1993, and the M.S. and Ph.D. degrees from the University of Texas at Austin, U.S., in 1995 and 1998, respectively, all in computer science. Dr. Mak is now a Professor with the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan. He was an Assistant Professor with the Department of Computer Science and Engineering, University of South Florida, U.S., from 1999 to 2003. His current research interests include VLSI physical design automation, and CAD for field-programmable technologies. Dr. Mak is a recipient of the IEEE/ACM Asia and South Pacific Design Automation Conference 2014 Best Paper Award for his work in e-beam lithography throughput optimization. His lab won the first place at the FPT 2008 Logic Block Clustering Contest, the third place at the IEEE CEDA PATMOS 2011 Timing Analysis Contest, and the second place at the TAU 2013 Variation-Aware Timing Analysis Contest. He has served on the Program and/or the Organizing Committee of Asia South Pacific Design Automation Conference, the International Conference on Field Programmable Logic and Applications, and the International Conference on Field-Programmable Technology (FPT). He was the Technical Program Chair of FPT in 2006 and was the General Chair of the same conference in 2008. Since 2009, he has been a Steering Committee Member of the International Conference on Field-Programmable Technology.