Self-Aligned Double Patterning-Aware Detailed Routing with Double Via Insertion and Via Manufacturability Consideration

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

***Abstract*—In 10nm technology node, self-aligned double patterning (SADP) and triple patterning lithography (TPL) allow us to achieve minimum wiring pitch of around 45nm. While metal layers can be printed by SADP, via layer manufacturing requires TPL to maintain design rules. SADP-aware detailed routing is proposed to ensure decomposability of metal layer patterns. However, its routing solution does not automatically guarantee TPL decomposable via layers. Vias have an inherently low reliability and via failure causes a great yield loss. Double via insertion (DVI) is an effective means to increase yield by reducing via failures. With the restriction of SADP design rules and consideration of TPL decomposability for via layers, DVI becomes a more challenging problem. In this paper, we consider DVI and via layer TPL manufacturability simultaneously in SADP-aware detailed routing. Both spacer-is-metal (SIM) and spacer-is-dielectric (SID) types of SADP are considered. Furthermore, we tackle the TPL-aware DVI in post-routing stage. Both ILP and high-performance heuristic solutions are proposed. The experimental results demonstrate our router can obtain 100% routability and TPL decomposable via layers with reduced dead via count. Meanwhile, compared with the ILP approach to solve TPL-aware DVI problem, the heuristic approach can achieve similar solution quality and significant speedup.**

I. Introduction

Due to the various delays and setbacks, the next-generation lithography (NGL), e.g, extreme ultraviolet (EUV) and electron beam lithography (EBL), is not ready for volume production. Multiple patterning lithography (MPL) has emerged as a key solution for layout manufacturing in advanced technology nodes. The two main choices for 10nm technology node are triple patterning lithography (TPL) and self-aligned double patterning (SADP). TPL is involved with three exposure-etch steps, and a mask is used for patterning in each step. This process flow is known as litho-etch-litho-etch-litho-etch (LELELE). Layout decomposition assigns layout patterns into three masks to resolve conflicts, and patterns assigned to the same masks are processed at once. It is a crucial design step which determines whether layout is manufacturable by TPL. The TPL layout decomposition can be transformed into a 3-coloring problem which is NP-complete. [1], [2], [3], [4] are major works on TPL layout decomposition. The Figure 1(b) shows an example of TPL layout decomposition for target layout in Figure 1(a). The layout patterns are assigned with three colors (orange, green, and blue), and patterns with the same color are in the same mask. Different from TPL, only two masks are used in SADP to produce the final layout patterns: a core mask and a cut/trim mask. In SADP, the mandrel patterns are firstly formed by a core mask. Then, spacers are deposited around mandrels. By utilizing the cut/trim mask, we can obtain the target layout patterns. Two popular types of processes are developed for SADP [5]. One is Spacer-Is-Metal (SIM) in which the spacers directly define layout patterns. The other one is Spacer-Is-Dielectric (SID) in which spacers ultimately define trenches between layout patterns. SADP layout decomposition generates both core and cut/trim masks to form the target layout while maintaining mask design rules, e.g., minimum spacing and minimum width constraints. [6], [7], [8], [9] are selected works on SADP layout decomposition. Based on how the second mask is utilized, we have either cut or trim approach in SADP process. In the cut approach, patterns defined on the cut mask are not included in the final layout patterns. In the trim approach, region not covered by the trim mask patterns is not included in the final layout patterns. In this paper, we focus on the SIM type SADP with cut approach and SID type SADP with trim approach. Note that our approach can be easily adapted to other SADP variants, e.g., SIM type SADP with trim approach. Figure 1(b)(c) show SIM type and SID type SADP layout decomposition for the target layout in Figure 1(a), respectively.

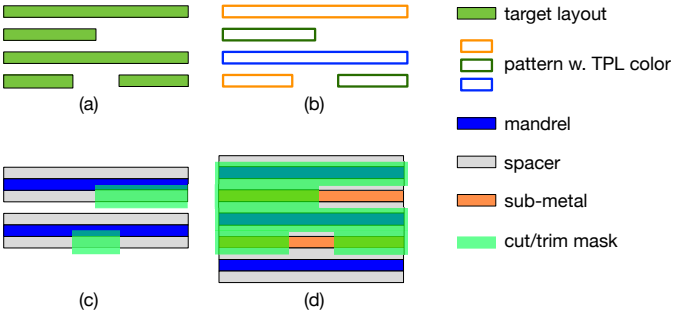

Fig. 1. Layout decomposition. (a) Target layout. (b) TPL layout decomposition. (c) SIM type SADP with cut approach layout decomposition. (d) SID type SADP with trim approach layout decomposition.

Compared with TPL, SADP has a couple of advantages. Firstly, SADP uses one fewer mask, which results in less manufacturing cost. Secondly, LELELE based TPL has misalignment error among the three exposure-etch steps. In contrast, SADP has a self-alignment property and less stringent overlay accuracy. It is popularly used for manufacturing unidirectional dense patterns with good pitch control [7]. However, to print

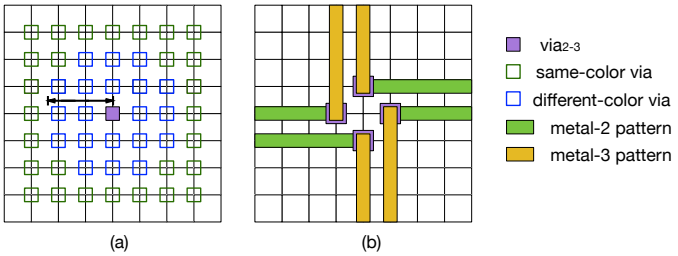

Fig. 2. (a) Same-color via pitch. (b) A via pattern forms a TPL violation in SADP-aware detailed routing solution.

two adjacent via patterns in 10nm technology node, we note that TPL is required. It is because the minimum width and minimum spacing constraints of the cut/trim mask in SADP prohibit printing two tiny features so close to each other. Thus, to manufacture layout in 10nm technology node, we assume to use SADP to print metal layer patterns and TPL to print via layer patterns in this paper.

The *same-color via pitch* is defined as the minimum center-to-center distance of a pair of via patterns from the same via layer that can be assigned to the same mask in TPL layout decomposition. By applying SADP on metal layers and TPL on via layers in layout manufacturing, the same-color via pitch is slightly larger than two times of routing track pitch size [10]. As shown in Figure 2(a), suppose a via from a routed net is inserted in the center of the grid. The *same-color via* is a via location where a via can be inserted, and assigned with the same color with existing via in TPL layout decomposition. On the contrary, a via should be assigned with a different color if it is inserted at the *different-color via*. We observe that SADP-aware detailed routing, which targets to ensure SADP decomposable metal layers, does not automatically guarantee via layers are TPL decomposable. Figure 2(b) shows an example of SADP-aware detailed routing solution which contains a TPL violation on the via layer between metal 2 and metal 3. Therefore, in 10nm technology node, it is necessary for SADP-aware detailed routing to consider the TPL decomposability of via layers.

With the shrinking size of technology nodes, the yield and reliability of integrated circuits (ICs) are more sensitive to process variation. Due to various reasons, e.g., cut misalignment, eletromigration, and thermal stress, vias may fail partially or completely [11]. A partial failed via will increase contact resistance and the parasitic capacitance which may induce timing problem. A complete failed via will leave an open net, and affect circuit functionality. It is identified as one of the major factors to cause chip failure and yield loss[12]. Double via insertion (DVI), which inserts a redundant via adjacent to a single via, is an effective method to increase yield and improve reliability. We call the single via that cannot have a redundant via without violating design rules a *dead via*. DVI in post-routing stage is limited by the inherent dead vias in the layout after detailed routing. To effectively reduce the dead via count, considering DVI during detailed routing stage is shown very helpful by previous works. In Figure 3(a), a redundant via is inserted for via b while via a is a dead via in post-routing DVI. On the contrary, if the detailed routing considers DVI as shown in Figure 3(b), all three vias can be protected by redundant vias. However, with the restriction of SADP design rules on metal layers and TPL design rules on via layers, considering DVI in detailed routing becomes an even more challenging problem. In this work, we investigate the SADP-aware detailed routing problem that simultaneously considers DVI and via layer manufacturability in 10nm technology node.

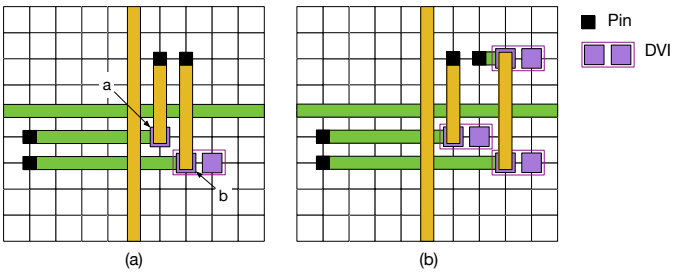

Fig. 3. (a) Detailed routing without DVI consideration. (b) Detailed routing considering DVI.

[13], [14], [15], [16], [17], [18], [19], [20] are selected works on SADP-aware detailed routing. Selected works on TPL-aware detailed routing are [21], [22], [23]. However, all these previous works ignore via layer manufacturability in 10nm technology node. Both [18] and [19] consider a via spacing rule, in which several via positions around an inserted via are forbidden. However, the via spacing rule does not ensure via layer patterns are compliant to design rules of a specific lithography technology. DVI in post-routing stage is studied by [11], [24], [25], [26], [27], [28]. However, in this stage only a slight layout modification is allowed, this methodology will restrict the DVI feasibility of some vias. Thus, considering DVI in detailed routing is proposed in [29], [30], [31], [32]. [29] formulates the problem as a multi-constrained shortest path problem solved by a Lagrangian relaxation technique. The approach has high time complexity, which limits the feasible problem size to be within hundreds of nets. In addition, the hard constraint which controls the dead via count in each net greatly reduces the routability. Both [32] and [30] consider DVI within a gridless routing model while grid based detailed routing is applied in our paper. [31] considers DVI both in double patterning lithography (DPL)-aware detailed routing and in post-routing stages. The DPL-aware detailed routing targets at 32/22nm technology nodes. Moreover, the exact function to compute each cost introduced to consider DVI in detailed routing is not given. Finally, the modification on coloring solution of existing layout is disabled during post-routing DVI. It greatly restricts flexibility of DVI, and potentially increases dead via count.

In this paper, we study both SIM and SID type SADP-aware detailed routing considering DVI and via layer manufacturability by TPL. Our major contributions are summarized as follows:

- This is the first work to consider DVI in both SIM and SID types SADP-aware detailed routing.
- This is the first work to consider via layer manufacturability by TPL in detailed routing. Each via layer in our

routing solution is ensured to be TPL decomposable.
- This is the first work to consider TPL design rules when performing DVI in post-routing stage. With the inserted redundant vias in post-routing DVI, each via layer is still TPL decomposable.
- The experimental results demonstrate the effectiveness and efficiency of our algorithm. Furthermore, the overheads of considering both DVI and via layer manufacturability in detailed routing is kept minimal.

The rest of the paper is organized as follows. Section 2 presents our problem formulation and some preliminaries. The overall flow and details of our proposed solution are presented in Section 3. Section 4 shows the experimental results, and finally Section 5 concludes the paper.

II. Preliminaries

A. Problem formulation

In the SIM type SADP lithography, the spacer whose width is constant will form the final metal pattern. Thus, it is hard to vary the line-width of layout pattern. On the contrary, it is potentially possible to print metal patterns with mixed-width by SID type SADP lithography. Since the focus of this paper is to consider double via insertion and via layer manufacturability, we will not further consider mixed-width wires. Thus, we assume that all the metal patterns in the layout are regular with same fixed width. Unidirectional routing has become a major trend in IC industry, and SADP is an excellent option for 1D layout manufacturing [33]. However, SADP is not restricted to unidirectional layout, and a careful choice of layouts makes it possible to define bidirectional features using SADP [15], [13], [34]. Therefore, we assume that there is a preferred routing direction on each routing layer and the other direction perpendicular to the preferred routing direction is defined as the non-preferred routing direction. We strongly discourage instead of completely disable routing in the non-preferred routing direction. We refer to this routing behavior as the restricted detailed routing.

The following is the formal problem definition.

Given a placed netlist, a multi-layer routing grid, and a set of design rules, we perform restricted detailed routing to generate a legal routing solution. The objective is to minimize the total wirelength and via count while achieving 100% routability. Meanwhile, the dead via count should be minimized in post-routing DVI. The constraints are that metal layer patterns are compliant to SADP design rules, and via layers are TPL decomposable.

B. Color pre-assignment approach

SADP-aware detailed routing is a challenging problem due to the non-intuitive SADP layout decomposition. As shown in Fig. 1(c)(d), the two mask patterns used to form the final layout have only minor similarities with the target layout patterns. Meanwhile, additional design rules, e.g., overlap error minimization [15], should be considered. The idea of color pre-assignment is applied in [17] to simplify the problem of maintaining SIM type SADP design rules in detailed routing. This approach is extended to handle SID type SADP-aware detailed routing [20]. In this paper, we adopt this approach for our SADP-aware detailed routing. Before the detailed routing, the multi-layer routing grid is assigned with colors. On each metal layer, a *panel* is defined as the area between two adjacent horizontal (vertical) grid lines. In the SIM type SADP, we pre-assign colors (grey and white) to the panels alternately in both horizontal and vertical directions as shown in Figure 4(a). In the SID type SADP, routing tracks are assigned with colors (black and grey) alternately in both horizontal and vertical directions as shown in Figure 4(c). The colored grid specify where the mandrel patterns and cut/trim mask patterns may be formed. Th mandrel pattern is required to be aligned in the middle of grey panel in SIM type, while mandrel patterns are required to be formed only along black tracks and should be aligned in the center in SID type. In this way, the SADP layout decomposition is known at the moment when a net is routed. Hence, the occurrence of a design rule violation is foreknown in detailed routing and can be easily minimized. However, the restriction of where mandrel patterns may be formed in the layout decomposition imposes additional constraints on detailed routing. [20] defines a preferred turn, a non-preferred turn, and a forbidden turn according to their SADP layout decomposability. If an L-shape metal layer pattern can be decomposed without any layout degradation, it is a preferred turn. Otherwise, it is a non-preferred turn. A forbidden turn is an L-shape metal layer pattern not allowed in detailed routing since the pattern is undecomposable. Each type of turn can be identified based on the location of the turning point in the colored routing grid and the turning direction. Figure 4(a)(b) shows four L-shape metal layer patterns in SIM type SADP-aware detailed routing and their corresponding mandrel and cut mask patterns after layout decomposition. As shown in Figure 4(b), a is preferred turn. b and c are forbidden turns due to the violation of minimum spacing rule in layout decomposition. d is a non-preferred turn since a degradation occurs due to the spacer rounding issue. Similarly, Figure 4(c) also shows four L-shape metal layer patterns in SID type SADP-aware detailed routing, in which a is a preferred turn, b and c are forbidden turns, and d is a non-preferred turn. As shown in Figure 4(d), no mandrel and trim mask patterns are drawn for b and c, since they are undecomposable. In sum, by applying color pre-assignment approach for our SADP-aware detailed routing, we need to strictly avoid forbidden turn to ensure layout is SADP manufacturable.

C. Double via insertion feasibility

Double via insertion is to add a redundant via beside a single via on the same via layer without a design rule violation. Given a single via, we assume a redundant via can be inserted at one of the four locations beside it. Figure 5(a) shows a partial layout of SIM type SADP-aware detailed routing containing a single via v which connects layout patterns from metal 2 and metal 3. Four locations a, b, c, and d on the same via layer of via v are candidate locations to insert a redundant via. We define these four candidate locations as DVI candidates (DVICs) of via v. To connect to the inserted

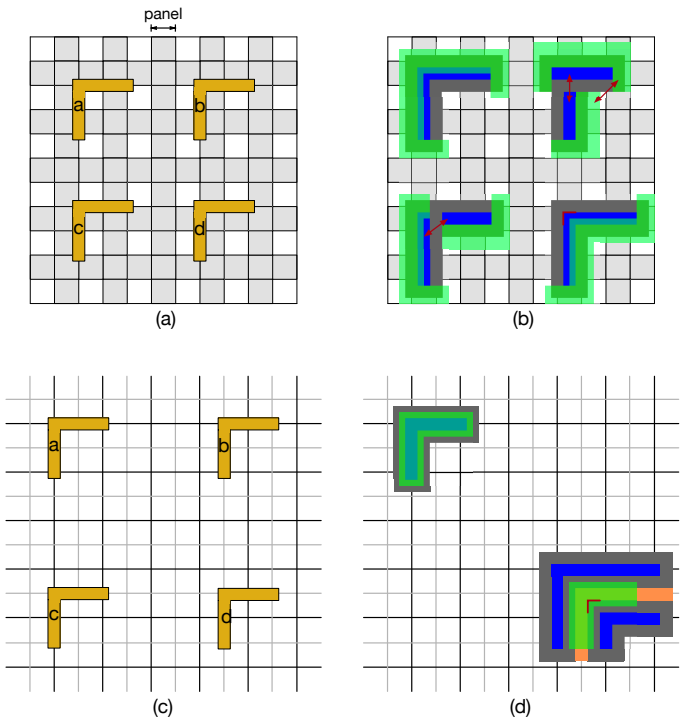

Fig. 4. Color pre-assignment for SADP-aware detailed routing. (a)(b) L-shape metal layer patterns and their corresponding layout decomposition in SIM type SADP-aware detailed routing. (c)(d) L-shape metal layer patterns and their corresponding layout decomposition in SID type SADP-aware detailed routing.

redundant via, the two metal patterns connected by v need to extend to a DVIC location as shown in Figure 5(b). This short connection together with the original metal pattern may form an L-shape pattern. This L-shape pattern needs to be examined under the constraints of our SADP-aware detailed routing. In Figure 5(b), the L-shape pattern on metal 3 is a forbidden turn. Thus, the DVIC d is not feasible for double via insertion. For the same reason, DVIC c is also not feasible due to the occurrence of a forbidden turn on metal 2. Furthermore, DVIC b is not feasible since the space is occupied by a metal layer pattern from another routed net. In this case, only DVIC a is feasible.

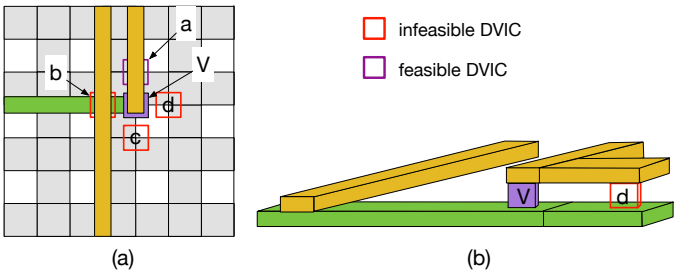

Fig. 5. Double via insertion. (a) Each single via has four DVICs. (b) The DVIC d is infeasible.

From above example, the DVI feasibility of a single via is affected by our SADP-aware detailed routing constraints. Without considering the case that the DVIC is occupied by the layout pattern from a different routed net, the DVI feasibility

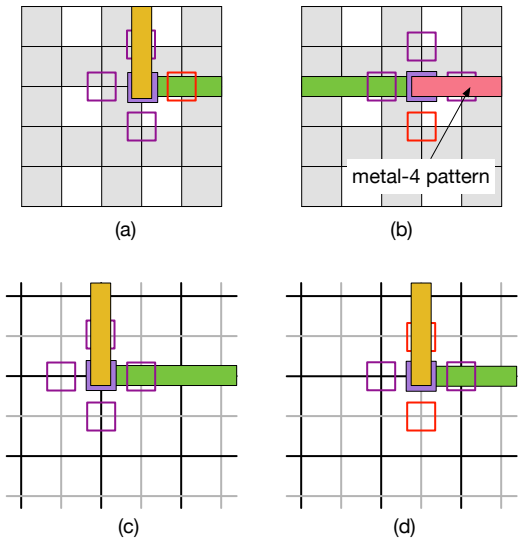

Fig. 6. (a)(b) DVI feasibility in SIM type SADP-aware detailed routing. (c)(d) DVI feasibility in SID type SADP-aware detailed routing.

of a single via is determined by two factors. One is the type of grid point where the single via locates in the colored grid, and the other is how the two metal patterns connected by the single via are oriented. Hence, given a single via from a routed net, it is easy to identify all its feasible and infeasible DVICs. Figure 6(a)(b) shows two examples of DVI feasibility of a single via in SIM type SADP-aware detailed routing. In Figure 6(a), to insert a redundant via at upper DVIC, the metal-2 pattern needs to extend towards the inserted via which forms an L-shape pattern. However, we observe that since the extension is only one unit grid length, the L-shape pattern which is a forbidden turn is actually decomposable by SIM type SADP. Thus, this particular DVIC is feasible in this case. Different from the upper DVIC, the right DVIC is infeasible since a forbidden turn occurs on metal 3. Even with only one unit grid length extension on metal 3, the L-shape pattern is undecomposable. In Figure 6(b), the single via is actually a stacked via which connects two layout patterns from metal 2 and metal 4. By comparing the two single vias in Figure 6(a)(b), they are located at the same type of grid point in the colored grid [20]. However, the orientation of two metal patterns connected by the single via in Figure 6(a) is different from that in Figure 6(b). Thus, the DVI feasibility of two single vias is not the same. Figure 6(c)(d) shows two examples of DVI feasibility of a single via in SID type SADP-aware detailed routing. Different from the previous examples, the orientations of two metal patterns connected by the single via in Figure 6(c)(d) are the same. However, the types of grid points where the two single vias locates in the colored grid are different [20]. As a result, the feasibility of the two single vias is different.

D. *Forbidden via pattern*

The via layer TPL layout decomposition can be transformed to a 3-coloring problem on the decomposition graph [2]. The graph is constructed by viewing each via pattern as

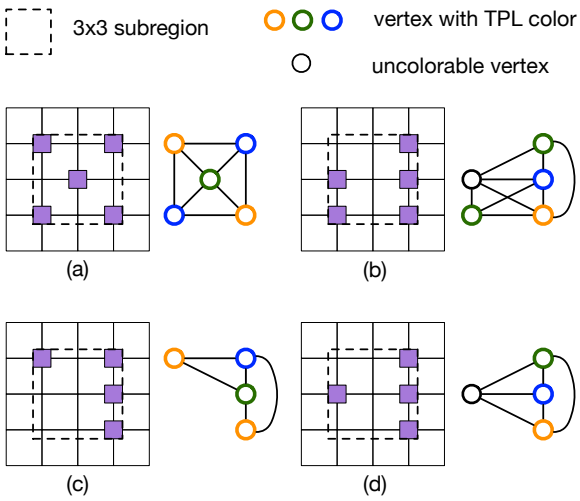

Fig. 7. Via patterns in 3x3 subregion and its decomposition graph. (a) A via pattern with 5 vias which is not an FVP. (b) An FVP with 5 vias, in which one via is uncolorable. (c) A via pattern with 4 vias which is not an FVP. (d) An FVP with 4 vias, in which one via is uncolorable.

a vertex. An edge exists between two vertices if the two via patterns are within same-color via pitch. However, maintaining a decomposition graph in detailed routing is expensive in terms of runtime and memory usage. Moreover, 3-coloring problem is NP-complete, and ensuring the decomposition graph is always 3-colorable in routing is difficult. Alternatively, we propose to examine each subregion of size 3×3 on each layer of routing grid and extract the via pattern within it. Then, determining if the via pattern is 3-colorable can be done in $O(1)$ time. If the via patterns in any 3×3 subregions in the routing grid are 3-colorable, then the decomposition graph is highly likely to be 3-colorable. Figure 7 shows several examples of the via pattern within a 3×3 subregion and its corresponding colored decomposition graph. We define a *forbidden via pattern* as a via pattern within a 3×3 subregion which is not 3-colorable. For simplicity, we refer to it as the FVP. An FVP can be identified by the via count and how vias are distributed within 3×3 subregion as follows:

1) Via patterns with 6 or more vias are all FVPs.
2) For via patterns with via count equal to 5, unless 4 of 5 the vias are on four corners of the 3×3 subregion, they are FVPs. Figure 7(a) shows an non-FVP with 5 vias and Figure 7(b) shows an FVP with 5 vias.
3) For via patterns with via count equal to 4, unless 2 of 4 the vias are on diagonally opposite corners of the 3×3 subregion, they are FVPs. Figure 7(c) shows an non-FVP with 4 vias and Figure 7(d) shows an FVP with 4 vias.
4) Via patterns with 3 or fewer vias are not FVPs.

III. Proposed Solution

A. Overall flow

The overall flow is shown in Figure 8. The inputs are a placed netlist, a multi-layer routing grid, and a set of design rules. The routing graph modeling, independent routing iterations, and negotiated congestion based rip-up and reroute (R&R) are explained with details in [20]. To consider DVI and via layer manufacturability by TPL in detailed routing, we develop a cost assignment scheme. The scheme assigns costs to the routing graph after routing of each net. The major advantage of this approach is the overheads of those additional considerations in detailed routing can be minimized. Thus, the SADP-aware detailed routing can keep high performance as before. Then, we propose a via layer TPL violation removal based R&R to eliminate all FVPs on the via layers. After that, a decomposition graph is constructed based on via layer patterns, and a fast 3-colorability check is performed. If not 3-colorable, the R&R is called to fix any remaining coloring conflicts. If 3-colorable, DVI considering via layer TPL decomposability is performed in post-routing stage. The output is SADP-aware detailed routing solution with DVI, in which via layers are guaranteed to be TPL decomposable.

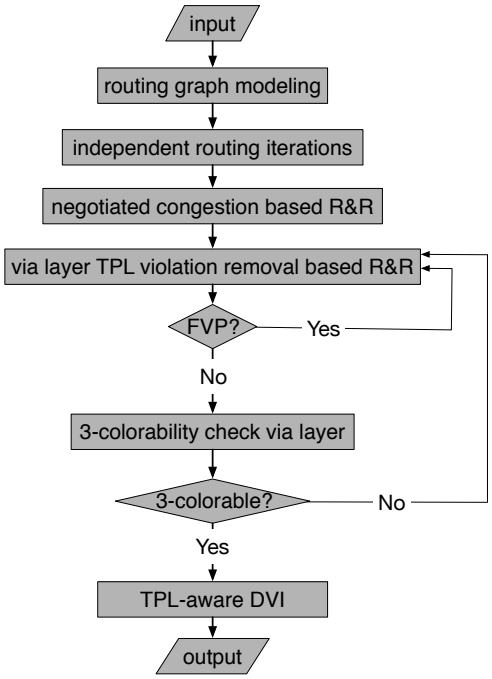

Fig. 8. Overall flow

B. Single net routing considering DVI and via layer TPL

Similar to the graph model in [20], we view each grid segment and 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 expense of routing from vertex in one end to the vertex on the other end. To consider DVI and via layer manufacturability by TPL during routing stage, the potential dead via and via pattern with TPL violation should be penalized in single net routing. Thus, we develop a cost assignment scheme which introduces various costs to the routing graph G. Figure 9 gives an example helping to explain how the cost assignment scheme works. Algorithm 1 is the pseudocode describing how costs are added to the G after routing of each net.

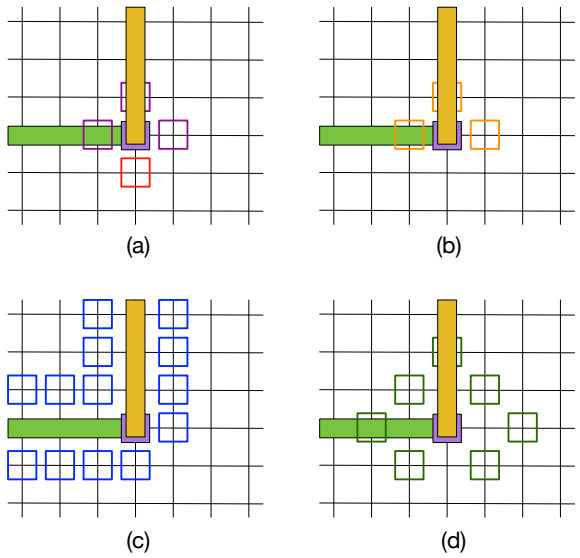

Fig. 9. An example to illustrate how cost assignment scheme works for single net routing (colored routing grid is not shown for clarity). (a) via_u of net_i has three feasible DVICs. (b) Block-DVIC via locations. (c) Along-metal via locations. (d) Conflict-DVIC via locations.

During sequential routing, the routing of a new net may affect the DVI feasibility of the vias in already routed nets. Meanwhile, the already routed nets may also affect the DVI feasibility of the vias in the net to be routed. We observe that the bi-directional effect is not symmetric, which will be analyzed as follows. Suppose via_u of a routed net_i connects metal lines on metal 2 and metal 3, and it has three feasible DVICs as shown in Figure 9(a). net_j is the new net to be routed. We firstly look into how routing of a new net affects the routed nets in terms of DVI feasibility. Given a single via in a routed net, block-DVIC via locations are defined as the its feasible DVICs. As shown in Figure 9(b), the block-DVIC via locations of via_u are marked by orange empty-fill squares. As long as net_j is routed across/through the block-DVIC via locations, the feasible DVIC count of via_u is reduced. The chance that via_u will have a redundant via in the post-routing DVI becomes lower. To penalize routing net_j across/through block-DVIC via locations, a penalty cost is added to the G after routing of net_i. The penalty cost can be computed by $\frac{\alpha}{\text{\# of feasible DVICs of } via_u}$, which we refer to it as block-DVIC cost (BDC). In this way, routing resources used for DVI by the via with smaller feasible DVIC count are assigned with higher costs, and vias from routed nets can be prevented from becoming dead vias.

Now, we look into how routed nets affect the net to be routed in terms of DVI feasibility. There are two scenarios which are shown in Figure 9(c)(d). We define the via locations along the metal patterns in the routed net as along-metal via locations, which are marked by blue empty-fill squares in Figure 9(c). If the net_j is routed with a via_v at any along-metal via location, the feasible DVIC of via_v may be reduced since the space is occupied by the metal pattern. To penalize routing net_j using vias at along-metal locations, a penalty cost is added to the G after routing of net_i. The penalty cost is referred as along-metal cost (AMC), which is a constant. In the other scenario, if net_j is routed with a via_v such that the via_v's DVIC shares the same via location with a feasible DVIC of via_u. We say these two DVICs are in conflict. In addition, the fewer number of feasible DVICs of via_u, the more chance that via_u or via_v becomes a dead via due to the conflicting DVIC. We define the via location like via_v as a conflict-DVIC via location which is marked by green empty-fill square in Figure 9(d). To prevent vias becoming dead vias due to conflicting DVIC in post-routing DVI, a penalty cost is added to the G after routing of net_i. The penalty cost is computed by $\frac{\beta}{\text{\# of feasible DVICs of } via_u}$, which we refer to it as conflict-DVIC cost (CDC). In summary, the cost assignment introduce three new kinds of costs to consider DVI in detailed routing, namely BDC, AMC, and CDC. In this way, the DVI feasibility of vias in both routed nets and the new net to be routed can be protected.

To avoid TPL violation on via layer, the cost assignment scheme also introduces another penalty cost. Given a via in a routed net, a penalty cost is assigned to each of its different-color via locations, which is shown in Figure 2(a). For each different-color via location, we find all existing vias that are within same-color via pitch with it, and refer to them as coloring conflicts. The penalty cost can be computed by $\gamma \times$ (# of coloring conflicts), which we refer to it as TPL cost (TPLC).

for *each* via_u *of routed* net_i **do**
  for *each* via_k *at feasible DVIC location of* via_u **do**
    find $vertex_k$ in G represents via_k;
    for *each* $vertex_m$ *adjacent to* $vertex_k$ *in* G **do**
      for *each edge* e *incident to* $vertex_m$ **do**
        $cost(e) += \frac{\alpha}{(\#\ of\ feasible\ DVICs\ of\ via_u)}$;
      end
    end
  end
  for *each* via_k *at conflict-DVI via location of* via_u **do**
    find $vertex_k$ in G represents via_k;
    for *each edge* e *incident to* $vertex_k$ **do**
      $cost(e) += \frac{\beta}{(\#\ of\ feasible\ DVICs\ of\ via_u)}$;
    end
  end
  for *each* via_k *at different-color via location of* via_u **do**
    find $vertex_k$ in G represents via_k;
    for *each edge* e *incident to* $vertex_k$ **do**
      $cost(e) += \gamma \times \#\ of\ coloring\ conflicts\ of\ via_k$
    end
  end
end
for *each* via_k *at along-metal via location of* net_i **do**
  find $vertex_k$ in G represents via_k;
  for *each edge* e *incident to* $vertex_k$ **do**
    cost(e) += AMC;
  end
end

Algorithm 1: Cost assignment scheme.

C. Via layer TPL violation removal based rip-up and reroute

To ensure via layers are TPL decomposable is a hard constraint for our SADP-aware detailed routing. The cost assignment scheme only discourages the occurrence of a TPL violation on via layers by adding TPLC to G. Thus, we introduce a new phase to our SADP-aware detailed routing flow: via layer TPL violation removal based R&R. Its pseudocode is presented in Algorithm 2. Similar to the negotiated congestion based R&R in [20], a base cost (BC), a usage cost (UC), and a history cost (HC) are applied to G. As mentioned before, maintaining a 3-colorable decomposition graph during R&R is expensive and difficult. Alternatively, we target to remove TPL violation on via layers by eliminating all FVPs through R&R iterations. The major advantage of this approach is detecting all the FVPs on via layers is $O(n)$, where n is the size of routing grid. In addition, updating the detected FVPs after a R&R iteration is $O(m)$, where m is the total number of vias in the rip-up and reroute net.

```
1  initialize a priority queue (PQ) and push all FVPs;
2  block via locations that will potentially generate FVP;
   while !PQ.emptry() do
       violation = PQ.pop();
       choose rip-up net N to resolve violation;
       update UC, BDC, AMC, CDC, and TPLC after removing N;
       update blocked via locations after removing N;
       the modified Dijkstra's algorithm reroutes N;
       update UC, BDC, AMC, CDC, and TPLC after rerouting N;
       update blocked via locations after rerouting N;
       if reroute creates congestion then
           update HC for congested routing resource;
           PQ.push(congestion);
       else if reroute creates FVP then
           update HC for vias in that FVP;
           PQ.push(FVP);
   end
```

Algorithm 2: Via Layer TPL violation removal based R&R.

Several techniques are applied in order to obtain a faster convergence of R&R iterations in this phase. In [20], a queue is kept during R&R iterations, which contains all the congestions to be resolved. A congestion occurs when the paths of more than one routed nets go through the same grid point in the routing grid. Differently, the via layer TPL violation removal based R&R targets to eliminate all the FVPs on via layers while maintaining a congestion-free routing solution. Thus, we use a priority queue to keep both congestion and FVPs during R&R iterations in line 1 of Algorithm 2. Specifically, each element in the priority queue contains the type of violation to be resolved, and the nets cause this violation. The type can either be a congestion or an FVP. In each R&R iteration, we check the violation in the top element from the priority queue, and find a rip-up net among nets causing this violation. We set the associated priority of a congestion is higher than that of an FVP in the priority queue. Hence, a congestion is always resolved first if it exists. Secondly, in the beginning of R&R iterations, some via locations are blocked in line 2 of Algorithm 2 to prevent reroute from creating FVPs. The blocked via locations are updated in lines 7 and 10 after a R&R iteration. Figure 10 shows several examples of how via locations are blocked. Given a 3×3 subregion, for each unused via location, if an FVP is created after inserting a via at that location, then it should be blocked. Even with blocked via locations, the reroute could still create an FVP if multiple vias are inserted within a 3×3 subregion. In this case, the HC of all the edges incident to each via in the newly created FVP are increased as shown in line 15. Hence, the vias in FVPs grow more expensive to use and FVPs can be potentially eliminated through R&R iterations.

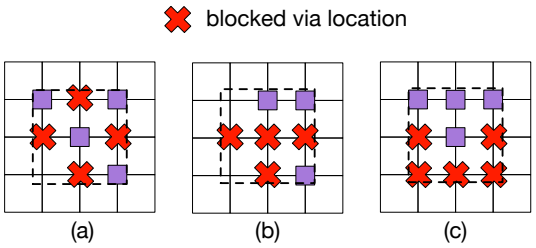

Fig. 10. Examples of how vias are blocked in via layer TPL violation removal based R&R.

D. 3-colorability check of decomposition graph

The target of via layer TPL violation removal based R&R is eliminate all the FVPs on via layers. However, even if all FVPs are successfully eliminated, there is a small chance that the decomposition graph is still not 3-colorable. Figure 11 gives two examples of via patterns which do not contain FVP but their TPL decomposition graph is not 3-colorable. The via pattern contains a via in the center and the rest of vias are on the periphery. We refer to such via patterns as *wheel via patterns*, since their structure is like a wheel. Thus, after via layer TPL violation removal based R&R, a decomposition graph is constructed for via layers. Then, we perform a fast 3-colorability check of decomposition graph. A greedy based Welsh-Powell algorithm [35] is applied for the check. If it is 3-colorable, our SADP-aware detailed routing exits. If not, R&R is called to fix the coloring conflict. We note that this case did not happen in our experiments in Section 4. This demonstrates that our FVP-based heuristic is good enough to remove all TPL violations on via layers in practice.

E. TPL-aware double via insertion

Our SADP-aware detailed routing is optimized for DVI, and via layers are TPL decomposable. In post-routing stage, DVI is performed and a large number of redundant vias will be inserted on via layers. The inserted redundant vias together with original vias in the routing solution will potentially cause TPL violations. Figure 12 shows an example of post-routing DVI for two adjacent single vias v_1 and v_2 on a via layer. Each single via has three feasible DVICs. In Figure 12(b), two redundant vias are inserted at a_1 and a_2 for the two single vias. The four-via pattern is not 3-colorable in TPL

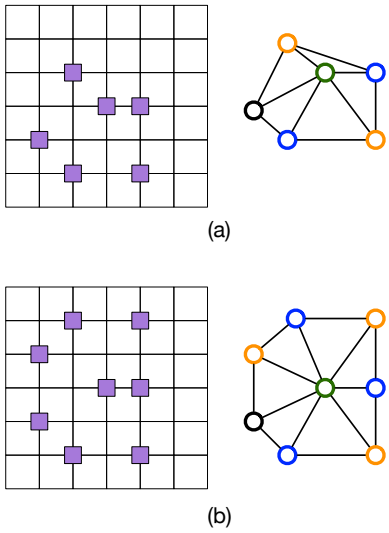

Fig. 11. (a) A "wheel via pattern" containing 5 vias, and its uncolorable TPL decomposition graph. (b) A "wheel via pattern" containing 7 vias, and its uncolorable TPL decomposition graph.

layout decomposition, and a TPL violation occurs. Thus, it is necessary to consider TPL decomposability of via layers even in the post-routing DVI. Figure 12(c) is another method of DVI with TPL awareness. Each of the single vias is inserted with a redundant via, and the four-via pattern is 3-colorable. We refer to post-routing DVI with consideration of via layer TPL decomposability as the TPL-aware DVI problem. The formal problem statement is as follows.

Given a SADP-aware detailed routing solution with TPL decomposable via layers, and a set of design rules, we perform double via insertion without modifying routing solution. The objective is to maximize the total number of inserted redundant vias. The constraint are that via layers are still TPL decomposable and metal layers are still SADP decomposable after double via insertion.

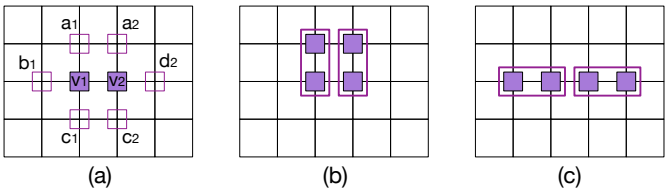

Fig. 12. Post-routing DVI. (a) Two adjacent single vias. (b) A TPL violation occurs after DVI. (c) TPL-aware DVI.

The post-routing DVI problem can be reduced into the maximum independent set problem, which is NP-complete [11]. To further consider via layer TPL decomposability and metal layer SADP decomposability in DVI makes the TPL-aware DVI a more challenging problem. To accurately evaluate the effectiveness of our DVI consideration in SADP-aware detailed routing, an integer linear program (ILP) is formulated for the TPL-aware DVI problem. By solving the ILP optimally, we can fairly compare the dead via count in our SADP-aware detailed routing with and without DVI consideration. The ILP formulation is presented as follows.

For each via_i, three binary variables oV_i, gV_i, and bV_i indicate its TPL color (orange, green, or blue) in 3-coloring. $oV_i = 1$, $gV_i = 0$, and $bV_i = 0$ if via_i's TPL color is orange. $oV_i = 0$, $gV_i = 1$, and $bV_i = 0$ if via_i's TPL color is green. $oV_i = 0$, $gV_i = 0$, and $bV_i = 1$ if via_i's TPL color is blue. Another binary variable uV_i is introduced in case it is uncolorable. $uV_i = 1$ if via_i is uncolorable in 3-coloring. For each feasible $DVIC_j$ of via_i, a binary variable D_{ij} indicates whether a redundant via is inserted at $DVIC_j$ for via_i. $D_{ij} = 1$ if a redundant via is inserted. Similarly, three binary variables oD_{ij}, gD_{ij}, bD_{ij} are introduced to indicate the redundant via's TPL color in 3-coloring. Both B and B' are extremely big constants. The mathematic formulation is shown as follows.

Objective:

$$maximize \quad \sum_{i=1}^{n}\sum_{j=1}^{m} D_{ij} - B \times \sum_{i=1}^{n} uV_i$$

where m is the number of feasible DVICs of via_i, and n is total number of single vias in the routing solution.

Constraints:

C1: For each via_i,

$$\sum_{j=1}^{m} D_{ij} \leq 1$$

C2: If $DVIC_{ij}$ and $DVIC_{i'j'}$ are in conflict,

$$D_{ij} + D_{i'j'} \leq 1$$

C3: For each via_i,

$$oV_i + gV_i + bV_i + uV_i = 1$$

C4: For each $DVIC_j$ of via_i,

$$oD_{ij} + gD_{ij} + bD_{ij} - B' \times (D_{ij} - 1) \geq 1$$
$$oD_{ij} + gD_{ij} + bD_{ij} + B' \times (D_{ij} - 1) \leq 1$$

C5: If via_i and $via_{i'}$ are within same-color via pitch,

$$oV_i + oV_{i'} \leq 1$$
$$gV_i + gV_{i'} \leq 1$$
$$bV_i + bV_{i'} \leq 1$$

C6: If via_i and $DVIC_{j'}$ of $via_{i'}$ are within same-color via pitch,

$$oV_i + oD_{i'j'} + B' \times (D_{i'j'} - 1) \leq 1$$
$$gV_i + gD_{i'j'} + B' \times (D_{i'j'} - 1) \leq 1$$
$$bV_i + bD_{i'j'} + B' \times (D_{i'j'} - 1) \leq 1$$

C7: If $DVIC_j$ of via_i and $DVIC_{j'}$ of $via_{i'}$ are within same-color via pitch,

$$oD_{ij} + oD_{i'j'} + B' \times (D_{ij} + D_{i'j'} - 2) \leq 1$$
$$gD_{ij} + gD_{i'j'} + B' \times (D_{ij} + D_{i'j'} - 2) \leq 1$$
$$bD_{ij} + bD_{i'j'} + B' \times (D_{ij} + D_{i'j'} - 2) \leq 1$$

C8: For each $DVIC_j$ of each via_i,

$$D_{ij}, oD_{ij}, gD_{ij}, bD_{ij} \in \{0,1\}$$
$$oV_i, gV_i, bV_i, uV_i \in \{0,1\}$$

As mentioned above, the major purpose of ILP formulation is to fairly evaluate the DVI consideration in our SADP-aware detailed routing. Solving the ILP may be time consuming when the problem size is big. This solution to the post-routing TPL-aware DVI problem is not realistic in practice. Alternatively, we propose a fast heuristic for TPL-aware DVI problem, which is shown in Algorithm 3. Note that the time complexity of the algorithm is $O(nlogn)$ where n is number of feasible DVICs.

```
TPL pre-coloring on existing vias;
initialize a priority queue (PQ);
for each feasible DVIC_j of via_i do
    setDP(DVIC_j);
    PQ.push(DVIC_j);
end
while !PQ.empty() do
    DVIC = PQ.top();
    if !DVIC.isValid() then
        PQ.pop();
    else
        if DVIC.DP!=computeDP(DVIC) then
            setDP(DVIC);
            PQ.pop();
            PQ.push(DVIC);
        else
            insert a redundant via at DVIC;
            PQ.pop();
        end
    end
end
TPL coloring on inserted redundant vias;
for each uncolorable inserted DVIC do
    Un-insert the redundant via;
end
```

Algorithm 3: Fast heuristic for TPL-aware DVI

Two critical issues need to be tackled in the design of TPL-aware DVI heuristic. The first issue is how to choose a feasible DVIC to insert a redundant via, i.e., DVI ordering. For each feasible $DVIC_j$ of each single via_i, a *DVI penalty* (DP) is defined to determine the priority to insert a redundant via at it in the DVI process. The bigger DP of a feasible DVIC, the lower priority to insert a redundant via at the DVIC.

$$\begin{aligned} DP_{DVIC_j} = {} & \delta \times \#\ of\ feasible\ DVICs\ of\ via_i \\ & + \lambda \times \#\ of\ conflicting\ DVICs\ with\ DVIC_j \\ & + \mu \times \#\ of\ killed\ DVICs\ by\ DVIC_j \end{aligned}$$

As shown in above equation, the DP computation consists of three parts. δ, λ, and μ are three parameters to control the weight of three parts. The first part is the number of feasible DVICs of via_i. It is more likely for single vias with fewer feasible DVICs to become dead vias. Hence, we would like to insert redundant vias with higher priority for those single vias. The second part is the number of conflicting DVICs with $DVIC_j$. To insert a redundant via at a DVIC with more conflicting DVICs will leave less chance of DVI for other single vias. Thus, we charge a bigger penalty on a feasible DVIC with more conflicting DVICs. The last part is how many feasible DVICs will be killed if a redundant via is inserted at $DVIC_j$. We say a feasible DVIC is killed if the redundant via at it form an FVP with existing vias. As shown in lines 2-5 of Algorithm 3, we compute DP for each feasible $DVIC_j$ of each via_i, and push it into PQ. The top element in PQ is the feasible DVIC with smallest DP. Note that the DP of each feasible $DVIC_j$ in the PQ is constantly updated in the DVI process, which is shown in lines 11-14 of Algorithm 3.

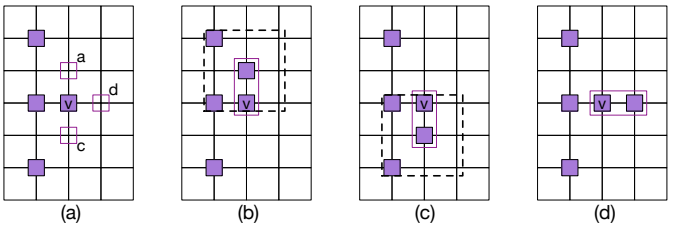

Fig. 13. TPL-aware DVI. (a) Four single via, and via v has three feasible DVICs. (b) An FVP occurs when a redundant via is inserted at a. (b) An FVP occurs when a redundant via is inserted at c. (c) No FVP occurs when a redundant via is inserted at d.

The second issue is how to ensure via layer TPL decomposability during DVI. We again apply the idea of FVP to maintain the constraint. Specifically, we do not allow inserting a redundant via at a feasible DVIC if it creates an FVP. Figure 13 shows an example of DVI for v among four single vias. The single via v has three feasible DVCs, namely a, c, and d. If a redundant via is inserted at a, an FVP is created as shown in Figure 13(b). Similarly, an FVP is created if a redundant via is inserted at c in Figure 13(c). The only valid choice is DVI at d which does not create any FVPs. Thus, every time before we insert a redundant via at a feasible DVIC, a validity check is perform as shown in line 8 of Algorithm 3. This check can be easily done in $O(1)$. Specifically, for each feasible $DVIC_j$ of single via_i, three conditions are checked. It is valid if and only if three conditions are all false. The three conditions are list as follows.

- a redundant via is already inserted at one of $DVIC_j$'s conflicting DVICs.
- a redundant via is already inserted at a feasible DVIC of via_i, which is not $DVIC_j$.
- an FVP is created if a redundant via is inserted at $DVIC_j$.

IV. Experimental results

We implemented our proposed algorithm in C++ programming language. We run all the experiments on a machine with a 2.4 GHz Intel Core i5 CPU and 8 GB memory. Gurobi 6.5 is called to solve the ILPs. Benchmarks from [18] are used to generate experimental results. Each circuit contains three routing layers metal 1, metal 2, and metal 3. Metal 1 is not

allowed for routing, and the preferred routing direction for metal 2 and metal 3 are horizontal and vertical, respectively. The detailed benchmarks statistics are listed in Table I. In the first subsection, we demonstrate the consideration of DVI and via layer TPL decomposability in both SIM and SID types SADP-aware detailed routing. In the second subsection, we compare the performance of ILP and heuristic solutions to TPL-aware DVI problem. Note that all parameters are kept the same for all the experiments in this section. The values of all the parameters are listed in Table II.

TABLE I
STATISTICS OF BENCHMARKS FROM [18]

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 II
PARAMETER VALUES IN THE EXPERIMENTS

	Cost assignment scheme				TPL-aware DVI		
parameter	α	AMC	β	γ	δ	λ	μ
value	8	1	4	4	1	1	1

A. SADP-aware detailed routing considering DVI and via layer TPL

In this subsection, we demonstrate our consideration of DVI and via layer TPL decomposability in both SIM and SID types SADP-aware detailed routing. For each type of SADP process, we run four sets of experiments, including SADP-aware detailed routing, SADP-aware detailed routing considering DVI, SADP-aware detailed routing considering via layer TPL decomposability, and SADP-aware detailed routing consider both DVI and via layer TPL decomposability. For a fair and accurate comparison, the post-routing TPL-aware DVI problem is solved by the ILP approach. The Table III and Table IV shows the experimental results of considering DVI and via layer TPL decomposability in SIM and SID types SADP-aware detailed routing, respectively. "CPU" is the detailed routing runtime, "#DV" denotes dead via count and "#UV" denotes the number of uncolorable vias in via layer TPL layout decomposition. Both "#DV" and "#UV" are reported from post-routing TPL-aware DVI ILP solution. Note that the routability for all benchmarks in four sets of experiments is 100%, thus we do not list it due to the limited table width.

As shown in the first two sets of experiments in Table III, without considering via TPL manufacturability, there are numerous uncolorable vias in TPL-aware DVI ILP solution. It indicates that TPL violations exist on via layers in SADP-aware detailed routing solution. Compared with the baseline, SADP-aware detailed routing considering DVI can reduce dead via count by 32%. With the consideration of via layer TPL decomposability in SADP-aware detailed routing, no uncolorable via is reported. Thus, via layers are TPL decomposable after post-routing DVI. Note that the consideration of via layer TPL decomposability indirectly helps to reduce dead via count since vias are more spread out. Hence, more space is left for DVI in post-routing stage. Finally, the SADP-aware detailed routing considering both DVI and via layer TPL decomposability can simultaneously reduce dead via count by 62% and ensure the via layers are TPL decomposable. The overheads are only 3% increase on wirelength and via count, respectively. With more routing constraints, runtime increases by 48% due to more R&R iterations.

As shown in Table IV, the consideration of DVI in SADP-aware detailed routing can reduce dead via count by 33%. Furthermore, with the consideration of via layer TPL decomposability, via layers are ensured to be TPL decomposable after post-routing DVI. Finally, the SID type SADP-aware detailed routing considering both DVI and via layer TPL decomposability can simultaneously reduce dead via count by 60% and ensure the via layers are TPL decomposable. The overheads of wirelength and via count are 3% increase, respectively. In addition, total detailed routing runtime is increased by 50%.

In addition to the above two sets of experiments, we also compare the SIM-type SADP-aware detailed routing considering both DVI and via layer TPL decomposability with [36], which is the previously published conference paper of this work. In this paper, we enlarge the parameters used in our cost assignment scheme to emphasize DVI consideration. As shown in Table V, we have further 33% dead via reduction compared with [36] with only 1% increase in both wirelength and via count. Meanwhile, the runtime keeps almost the same.

B. TPL-aware DVI

In this subsection, we compare the performance of our proposed ILP and heuristic solutions to TPL-aware DVI problem. We generate routing solutions by both SIM and SID types SADP-aware detailed routing with consideration of DVI and via layer TPL decomposability. Based on each routing solution, the post-routing TPL-aware DVI is solved by both ILP and heuristic approaches. For heuristic approach, we do 3-coloring of via layer patterns after DVI by Welsh-Powell algorithm [35]. Table VI and Table VII show experimental results of TPL-aware DVI for SIM and SID type SADP-aware detailed routing, respectively. In Table VI, compared with ILP approach, our proposed heuristic has more than $600\times$ speedup. Meanwhile, no uncolorable via is reported. It further demonstrates that via layer TPL decomposability can be ensured in DVI by prohibiting FVPs. Finally, our heuristic only has about 8% more dead vias than that of ILP approach. In Table VII, compared with ILP approach, our proposed heuristic has almost $500\times$ speedup, no uncolorable via, and about 10% more dead vias.

V. CONCLUSION

In this paper, we consider DVI and via layer manufacturability by TPL in both SIM and SID types SADP-aware detailed routing. A cost assignment scheme is integrated into our SADP-aware detailed routing framework to consider DVI and via layer TPL decomposability. In addition, a via layer TPL violation based rip-up and reroute is applied to ensure

TABLE III

	SIM type SADP-aware routing					Consider DVI					Consider via layer TPL					Consider DVI & via layer TPL				
CKT	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV
ecc	35423	4969	15.6	291	24	35512	4982	18.8	163	23	35782	4965	19.7	132	0	35837	5027	22.9	105	0
efc	45856	7707	30.2	880	104	46085	7842	33.1	611	64	47059	7819	41.0	440	0	47217	7965	41.2	357	0
ctl	56902	9132	31.5	663	63	57244	9228	35.6	412	36	57591	9172	37.5	305	0	57908	9289	46.2	240	0
alu	56986	10053	38.5	1227	113	57633	10312	36.6	858	51	58724	10252	50.8	600	0	59103	10486	51.4	481	0
div	120267	20153	86.1	2302	272	121489	20553	103.3	1775	139	123295	20377	150.6	1133	0	124024	20800	157.3	883	0
top	379114	70185	261.1	9068	317	382784	71489	282.1	5932	161	393030	71459	377.5	4199	0	394752	72878	367.2	3461	0
Ave.	115758.0	20366.5	77.20	2405.1	148.8	116791.1	20734.3	84.96	1625.1	79.0	119246.8	20674.0	112.90	1134.8	0	119806.8	21074.1	114.41	921.1	0
Nor.	1.00	1.00	1.00	1.00	1.00	1.01	1.02	1.10	0.68	0.53	1.03	1.02	1.46	0.47		1.03	1.03	1.48	0.38	

TABLE IV

	SID type SADP-aware routing					Consider DVI					Consider via layer TPL					Consider DVI & via layer TPL				
CKT	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV
ecc	35290	4918	14.9	247	22	35445	4939	17.7	146	14	35515	4908	17.7	133	0	35692	4960	18.7	104	0
efc	45561	7650	31.1	843	83	45877	7765	29.2	590	46	46709	7773	42.6	450	0	47030	7943	51.7	383	0
ctl	56994	9048	35.7	626	61	56954	9138	36.8	445	28	57361	9108	44.7	312	0	57467	9201	46.3	269	0
alu	56725	9919	37.5	1133	94	57322	10144	32.9	874	62	58186	10116	50.2	594	0	58407	10324	52.8	503	0
div	119808	19948	101.9	2289	260	120855	20221	111.6	1646	143	122686	20275	151.3	1133	0	123129	20531	144.7	898	0
top	377388	69477	222.3	8901	1203	381251	70908	260.4	5693	785	389897	70954	373.9	4266	0	391793	72176	351.9	3512	0
Ave.	115227.6	20160.0	73.92	2339.6	287.1	116284.0	20519.1	81.46	1564.0	179.6	118392.33	20522.33	113.43	1246.2	0	118919.6	20855.8	111.05	944.8	0
Nor.	1.00	1.00	1.00	1.00	1.00	1.01	1.02	1.10	0.67	0.63	1.03	1.02	1.53	0.49		1.03	1.03	1.50	0.40	

TABLE V
SADP-AWARE DETAILED ROUTING WITH DVI AND VIA LAYER TPL DECOMPOSABILITY CONSIDERATION

	[36]					Consider DVI & via layer TPL				
CKT	WL	#Vias	CPU(s)	#DV	#UV	WL	#Vias	CPU(s)	#DV	#UV
ecc	35724	4966	19.6	146	0	35837	5027	22.9	105	0
efc	46604	7809	45.5	477	0	47217	7965	41.2	357	0
ctl	57642	9209	43.1	336	0	57908	9289	46.2	240	0
alu	58289	10249	50.2	655	0	59103	10486	51.4	481	0
div	122810	20399	147.3	1325	0	124024	20800	157.3	883	0
top	389998	71588	386.1	5271	0	394752	72878	367.2	3461	0
Ave.	118511	20703	115.3	1368	0	119806	21074	114.4	921	0
Nor.	1.00	1.00	1.00	1.00		1.01	1.01	0.99	0.67	

TABLE VI
TPL-AWARE DVI FOR SIM TYPE SADP-AWARE DETAILED ROUTING

	ILP			Heuristic		
	#DV	#UV	CPU (s)	#DV	#UV	CPU (s)
ecc	105	0	1505.0	113	0	1.0
efc	357	0	1332.7	389	0	1.4
ctl	240	0	3275.0	257	0	1.9
alu	481	0	2868.3	533	0	1.8
div	883	0	1505.0	979	0	3.9
top	3461	0	5075.0	3740	0	13.1
Ave.	921.1	0	2593.52	1001.8	0	3.89
Nor.	0.92		667.57	1.00		1.00

TABLE VII
TPL-AWARE DVI FOR SID TYPE SADP-AWARE DETAILED ROUTING

	ILP			Heuristic		
	#DV	#UV	CPU (s)	#DV	#UV	CPU (s)
ecc	104	0	20.4	111	0	0.9
efc	383	0	323.4	418	0	1.3
ctl	269	0	175.1	294	0	1.6
alu	503	0	1292.0	559	0	1.7
div	898	0	2930.1	1008	0	4.7
top	3512	0	6509.9	3917	0	12.2
Ave.	944.8	0	1875.21	1051.1	0	3.77
Nor.	0.90		497.40	1.00		1.00

via layers are TPL decomposable. In the post-routing stage, we tackle the TPL-aware DVI problem, and propose both ILP and high-performance heuristic solutions. The experimental results demonstrate that the consideration of DVI and via layer TPL manufacturability is effective and efficient with minimal overheads.

ACKNOWLEDGEMENT

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

REFERENCES

[1] Christopher Cork, Jean-Christophe Madre, and Levi Barnes, "Comparison of triple-patterning decomposition algorithm using aperiodic tiling patterns," in *Proc. of SPIE*, May 2008, p. 702839.

[2] Bei Yu, Kun Yuan, Boyang Zhang, Duo Ding, and David Z. Pan, "Layout decomposition for triple patterning lithography," in *Proc. of ICCAD*, November 2011.

[3] Shao-Yun Fang, Yao-Wen Chang, and Wei-Yu Chen, "An novel layout decomposition algorithm for triple patterning lithography," in *Proc. of DAC*, June 2012.

[4] Kuang Jian and Evangeline F. Y. Young, "An efficient layout decomposition approach for triple patterning lithography," in *Proc. of DAC*, June 2013.

[5] 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 SPIE*, April 2011, p. 79740L.

[6] Hongbo Zhang, Yueline Du, Martin D. F. Wong, and Rasit Topaloglu, "Self-aligned double patterning decomposition for overlay minimization and hot spot detection," in *Proc. of DAC*, June 2011.

[7] Yongchan Ban, Kevin Lucas, and David Z. Pan, "Flexible 2D layout decomposition framework for spacer-type double patterning lithography," in *Proc. of DAC*, June 2011.

[8] 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 ISPD*, March 2012.

[9] Shao-Yun Fang, Yi-Shu Tai, and Yao-Wen Chang, "Layout decomposition for spacer-is-metal (SIM) self-aligned double patterning," in *Proc. of ADP-DAC*, Jan 2015.

[10] Lars Liebmann, Vassilios Gerousis, Paul Gutwin, Mike Zhang, Geng Han, and Brian Cline, "Demonstrating production quality multiple exposure patterning aware routing for the 10nm node," in *Proc. of SPIE*, March 2014.

[11] Kuang-Yao Lee and Ting-Chi Wang, "Post-routing redundant via insertion for yield/reliability improvement," in *Proc. of ASPDAC*, January 2006.
[12] Puneet Gupta and Evanthia Papadopoulou, "Yield analysis and optimization," in *The Handbook of Algorithms for VLSI Physical Design Automation*. CRC Press, June 2010.
[13] Jhih-Rong Gao and David Z. Pan, "Flexible self-aligned double patterning aware detailed routing with prescribed layout planning," in *Proc. of ISPD*, March 2012.
[14] 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 DAC*, June 2013.
[15] 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 DAC*, June 2014.
[16] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z. Pan, "Self-aligned double patterning aware pin access and standard cell layout co-optimization," in *Proc. of ISPD*, March 2014.
[17] Yixiao Ding, Chris Chu, and Wai-Kei Mak, "Detailed routing for spacer-is-metal type self-aligned double/quadruple patterning lithography," in *Proc. of DAC*, June 2015.
[18] 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 DAC*, June 2015.
[19] 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," *ACM Transactions on Design Automation of Electronic Systems*, vol. 21, 2016.
[20] Yixiao Ding, Chris Chu, and Wai-Kei Mak, "Self-aligned double patterning lithography aware detailed routing with color pre-assignment," *IEEE Trans. Computer-Aided Design Integrated Circuits Systems*, 2016.
[21] Qiang Ma, Hongbo Zhang, and Martin D.F. Wong, "Triple patterning aware routing and its comparison with double patterning aware routing in 14nm technology," in *Proc. of DAC*, June 2012.
[22] Yen-Hung Lin, Bei Yu, David Z. Pan, and Yih-Lang Li, "Triad: A triple patterning lithography aware detailed router," in *Proc. of ICCAD*, November 2012.
[23] Po-Ya Hsu and Yao-Wen Chang, "Non-stitch triple patterning-aware routing based on conflict graph pre-coloring," in *Proc. of ASP-DAC*, January 2015.
[24] Kuang-Yao Lee, Cheng-Kok Koh, Ting-Chi Wang, and Kai-Yuan Chao, "Optimal post-routing redundant via insertion," in *Proc. of ISPD*, April 2008.
[25] Kuang-Yao Lee, Shing-Tung Lin, and Ting-Chi Wang, "Redundant via insertion with wire bending," in *Proc. of ISPD*, March 2009.
[26] Cheok-Kei Lei, Po-Yi Chiang, and Yu-Min Lee, "Post-routing redundant via insertion with wire spreading capability," in *Proc. of ASPDAC*, Janunary 2009.
[27] Kuang-Yao Lee, Shing-Tung Lin, and Ting-Chi Wang, "Post-routing redundant via insertion and line end extension with via density consideration," in *Proc. of ISPD*, March 2009.
[28] Shing-Tung Lin, Kuang-Yao Lee, Ting-Chi Wang, Cheng-Kok Koh, and Kai-Yuan Chao, "Simultaneous redundant via insertion and line end extension for yield optimization," in *Proc. of ASPDAC*, Janunary 2011.
[29] Gang Xu, Li-Da Huang, David Z. Pan, and Martin D. F. Wong, "Redundant-via enhanced maze routing for yield improvement," in *Proc. of ASPDAC*, Janunary 2005.
[30] Huang-Yu Chen, Mei-Fang Chiang, Yao-Wen Chang, Lumdo Chen, and Brian Han, "Full-chip routing considering double-via insertion," in *Proc. of DAC*, July 2006.
[31] Kun Yuan, Katrina Lu, and David. Z. Pan, "Double patterning lithography friendly detailed routing with redundant via consideration," in *Proc. of DAC*, July 2009.
[32] Chih-Ta Lin, Yen-Hung Lin, Guan-Chan Su, and Yih-Lang Li, "Dead via minimization by simultaneous routing and redundant via insertion," in *Proc. of ASPDAC*, Janunary 2010.
[33] Yixiao Ding, Chris Chu, and Wai-Kei Mak, "Throughput optimization for SADP and e-beam based manufacturing of 1D layout," in *Proc. of DAC*, June 2014.
[34] Koichi Nakayama, Chikaaki Kodama, Toshiya Kotani, Shigeki Nojima, Shoji Mimotogi, and Shinji Miyamoto, "Self-aligned double and quadruple patterning layout principle," in *Proc. of SPIE*, March 2012, p. 916678.
[35] D. J. A. Welsh and M. B. Powell, "An upper bound for the chromatic number of a graph and its application to timetabling problems," *The Computer Journal*, vol. 10, pp. 85–86, 1967.
[36] Yixiao Ding, Chris Chu, and Wai-Kei Mak, "Self-aligned double patterning-aware detailed routing with double via insertion and via manufacturability consideration," in *Proc. of DAC*, June 2016.