A Framework for Double Patterning-Enabled Design

Rani S. Ghaida, Kanak B. Agarwal*, Sani R. Nassif*, Xin Yuan†, Lars W. Liebmann‡, Puneet Gupta
UCLA, Electrical Engineering Dept.
* IBM Corp., Austin Research Lab
† IBM Corp., Burlington
‡ IBM Corp., Semiconductor Research & Development Center
{rani, puneet}@ee.ucla.edu {kba, nassif, xinyuan, lliebman}@us.ibm.com

***Abstract*—While the next generation of lithography systems is still under development, extending optical lithography using double patterning (DP) is the only solution to continue the scaling of technology. The biggest technical challenge of DP is the presence of mask-assignment conflicts in dense layers. In this paper, we propose a framework for DP conflict removal for standard cells. First, we offer a $O(n)$ algorithm for the mask assignment (up to 223× faster than the ILP-based approach) that guarantees a conflict-free solution for layouts without native conflicts (conflicts that cannot be resolved with stitching). We then formulate the problem of conflict removal as a linear program (LP), which allows extremely fast run-time (under 10 seconds in real time for typical cells). The framework removes DP conflicts and legalizes the layout across all layers simultaneously while minimizing the layout perturbation. For cells from a commercial 22nm library designed without any DP awareness, our method usually removes all DP conflicts without any area increase; for some complex cells, the method still removes all conflicts with a modest 6.7% average increase in area. The method is more general, however, and can also be applied for macro layouts and the interconnect layers in complete designs as we demonstrate in the paper.**

I. Introduction

While the next generation of lithography systems is still under development, extending optical lithography using double patterning (DP) is the only solution to continue the scaling of technology. One of the most favorable DP alternatives is pitch-split DP where layout patterns are formed with two separate exposure and etch (or develop) steps. Hereafter, we will use the term DP to denote pitch-split DP.

For a layout to be DP manufacturable, layout features that violate the minimum spacing of single patterning must be assigned to different masks. DP mask assignment is essentially a two-color labeling problem [1]. The difference from the labeling problem of graph theory is that a layout polygon can be a composite of the layouts of the different masks. The splitting of polygons into multiple parts is known as stitching and the location where the two masks join is called a stitch. Although stitching complicates the labeling problem, it is an efficient and almost-free method to conform many, originally DP-unfriendly, layout patterns to DP (by breaking odd cycles in the conflict graph as in the example of Figure 1). Even with stitching, many patterns cannot be assigned to the two masks without violation of the minimum single-patterning spacing. Such patterns are called native DP conflicts and resolving these conflicts – with certain layout perturbation – is the biggest challenge facing the deployment of DP.

A. Prior Art in Mask Assignment

Prior works in DP mask assignment differ mainly by the way stitches are dealt with. Rule-based stitching where polygons are split at certain fixed locations is proposed in [2, 3]. The drawback of this method is that many stitch locations cannot be found by the rules.

In [4, 5], the layout is segmented into rectangles, stitches that can resolve DP conflicts are determined, and the problem of mask assignment with stitch minimization is formulated as an integer linear program (ILP). Segmentation of the layout into rectangles has many drawbacks. First, it complicates the problem as it forces the consideration a lot of extra stitch locations that should never be used. Consider the example of Figure 2. Rectangle C has spacing violations with both rectangles A and B. As a result, A and B must always be assigned to the same mask to avoid a DP conflict and the stitch location at the joint of A and B is never used. The second drawback

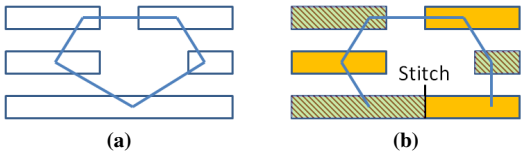


Figure 1: Example of a layout with odd cycle in its conflict graph (a) that was broken by introducing a stitch (b).

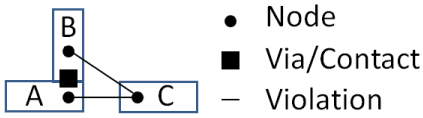


Figure 2: DP mask assignment and conflict removal with segmentation of the layout into rectangles has many drawbacks.

of segmentation is that it forces the method to use a single spacing-rule value. Because rectangles are mapped into nodes, there is no way to distinguish between side-to-side, tip-to-side, and tip-to-tip spacing design rules (DRs) that have different values in modern technologies. Another drawback of the methods of [4, 5] is that ILPs are very time consuming to solve (NP-hard problem [6]). In addition, the method in [5] can only be applied to gridded layouts with a grid size equal to half the pitch, which is not the case for many layers (e.g., M1).

In [7], a method for DP mask assignment with multiple objectives including stitch minimization is proposed. The method is based on min-cut partitioning and the problem is solved in a polynomial time algorithm. The method is still based on the segmentation of the layout into rectangles and cannot handle multiple spacing rules. The work in [8] offers a method to speedup the mask assignment process through graph partitioning.

B. Prior Art in Conflict Removal

Prior art in layout perturbation to resolve DP conflicts [9–11] generally formulates the problem as an ILP (except [11]). Moreover, all previous works segment the layout into rectangles and moves rectangles around to remove DP conflicts.

Working with rectangles has the same drawback discussed earlier and some additional drawbacks. The problem is further complicated because the automated layout perturbation solver (ILP or compaction) needs to maintain the connectivity of rectangles at joints (e.g., L-shape) through additional constraints. Moreover, because the constraints of the solver are defined between rectangles, overlap rules with features from the top and bottom layers cannot be handled correctly. Consider again the example of Figure 2 where an L-shape metal overlaps with a via (or contact) at the corner. If the via movement is blocked, the solver will try to move shapes A and B so that *each* covers the via *completely*. Not only these moves are unnecessary because the via is initially covered, but they can also impact the layout area and the effectiveness of the conflict removal.

In [9], DP requirements are added to the ILP constraints to perform DP-aware layout migration while minimizing area and layout perturbation. In addition to the problems with segmentation, the method leads to unmanageable number of constraints, excessive runtime to solve the ILP, and does not work well when the layout contains DP conflicts initially (i.e. not migrated from a previous generation).

In [10], wire spreading is proposed to remove DP conflicts. All wire-spreading options that reduce DP conflicts are pre-computed and conflicts and wire moves are minimized in the ILP. In addition to the problems common to all prior works that are discussed earlier, wire spreading can reduce the number of conflicts by a modest amount (as the results in [10] show). Many conflicts can be resolved with edge-location and wire-width adjustments but not with wire spreading. Moreover, to avoid creating new DP conflicts, the method only moves segments when their spacing from all neighboring wires after the movement is at least equal to the single-patterning spacing. In many actual cases however, we may be able to move the segment and keep it apart from its neighbors by the spacing rule in the layout[1] (typically half the single-patterning spacing). The method of [10] cannot detect such cases and unnecessarily limits the wire spreading because, otherwise, the entire graph will have to be checked for newly created conflicts for every wire-spreading option.

Rather than solving the problem with an ILP, the work in [11] applies traditional layout compaction – based on minimum-area metric – iteratively as long as DP conflicts are reduced. At each iteration, the process of DP-compliance checking, which includes pattern projection, segmentation, conflict graph generation, and odd cycle detection, is performed initially. DP constraints at odd cycles only are then generated and a trial compaction is performed. The DP-compliance check is repeated and, if the number of odd cycles is reduced, the DP-constraints are permanently committed. In addition to the problems associated with segmentation into rectangles and the large runtime of iterative compaction and performing the DP-compliance check twice at each iteration, the method is not effective in removing DP conflicts and keeps a large number of conflicts unresolved (as reported in [11]). Because DP constraints are generated only at odd cycles, resolving one conflict may create a new conflict in other parts of the layout. As a result, the iterative compaction may stop without removing many DP conflicts that otherwise could have been resolved. In our work, we were able to remove DP conflicts efficiently, effectively, and simultaneously across all layers. This was made possible by essentially defining DP constraints all over the layout in terms of DRs – after an initial coloring that minimizes the number of conflicts – and applying linear programming-based layout compaction once across all layers.

C. Our Approach

In this paper, we propose a framework for DP conflict removal for standard cells and macros. We follow a different approach for the mask assignment than prior works. Essentially, we use DR-dependent projection to determine the features that may cause DP conflicts and their actual, possibly non-rectangular, shapes. We then formulate the problem as a labeling problem that we solve in a $O(n)$ algorithm. In our method, all candidate stitches that can be useful are automatically identified and are reduced by the algorithm. Because we use all candidate stitches, our method guarantees a conflict-free mask-assignment solution when the layout has no native conflicts (conflicts that cannot be resolved with stitching). Using a linear program (LP), DP conflicts are removed and the layout is legalized simultaneously across multiple layers by edge-based layout perturbation. This layout legalization is performed through layout compaction formulated as a *minimum perturbation problem*[2]. The proposed methodology allows the layout designer to design with conventional single-patterning layers and DRs, masking him from the complexity in dealing with double-patterning layers and requirements.

Our proposed methodology for designing DP-compatible layouts is depicted in Figure 3. Using existing non DP-compatible layouts or layouts designed from scratch using conventional rules, we perform an optional step of layout simplification at DP layers for the possible sacrifice of non-crucial parts as described in Section IV. We then carry out DP mask assignment while considering all candidate locations. If the layout contains DP native conflicts, the conflicts are removed and the layout is legalized simultaneously across all layers while minimizing layout perturbation using a LP and maintaining the same area as the original layout. Optionally, in case some native conflicts remain unresolved, the LP-based layout legalization is repeated while allowing an area increase to remove more DP conflicts (all conflicts are removed after this step in most cases).

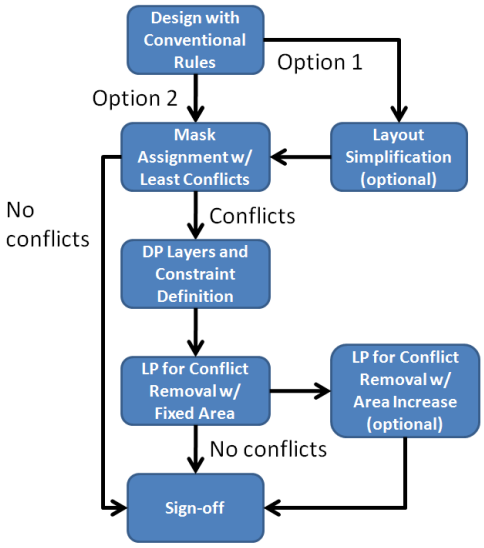


Figure 3: The flow for our proposed method to achieve DP-enabled layout design.

We make the following contributions.

- We offer a framework for DP conflict removal while *minimizing layout perturbation* for standard cells that guarantees the legalization of the layout across all layers simultaneously. We achieve, using our proposed method, conflict-free standard cell layouts that are fully compatible with DP.
- We formulate the problem of conflict removal as a LP, which can be solved in polynomial time [14], as opposed to prior art of conflict removal that formulate the problem as an ILP, which is NP-hard [6].
- We propose a polynomial time $O(n)$ algorithm for DP mask assignment that guarantees a conflict-free solution for layouts without native conflicts by using all candidate stitch locations.
- We handle, during the mask assignment as well as the layout perturbation, complex spacing rules including tip-to-tip and tip-to-side in addition to the minimum spacing.

The remaining paper is organized as follows. Section II describes our mask-assignment approach that handles complex spacing rules and guarantees conflict-free solution for layouts without native conflicts. Section III presents our method for DP conflict removal and layout legalization based on minimum perturbation. Section IV describes a method to improve the effectiveness of the conflict removal. Section V presents the experimental results, while Section VI concludes the paper.

II. DP Mask Assignment

We follow a different approach for the DP layout decomposition than what is presented in the literature. We first find all parts of the layout that have DP spacing violations with neighboring features and, then, we assign these violating parts to the two masks. In this way, candidate locations of stitches are automatically defined and can be easily minimized as we show later in this section. In the end, non-violating parts can be assigned to either masks. If a non-violating part touches features of the same mask, we assign it to that same mask to avoid introducing extra stitches; whereas, if a non-violating part touches features of different masks, we assign it to both masks

[1]When the segment is assigned a different color than its neighbors.

[2]Unlike [11] that uses minimum-area metric for compaction. The advantages of minimum layout perturbation metric over the minimum area metric for layout compaction are discussed in [12, 13].

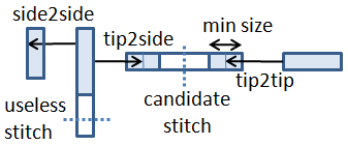


Figure 4: DR-dependent projection to identify violating parts and stitch locations. Violating parts are the blue features and non-violating parts are the clear features.

Table I: Example showing the advantage of handling multiple spacing rules with different values (MS) over using a single minimum spacing (SS) for a local clock buffer controller layout.

# of Transitors	M1 pitch	# of C w/ SS	# of C w/ MS	reduction
460	80nm	198	159	20%

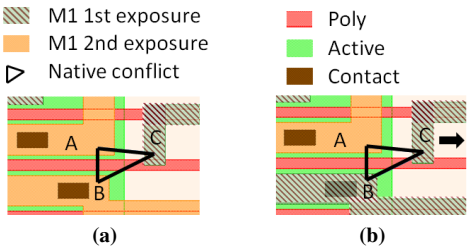


Figure 5: Odd cycle mask assignment can affect the efficiency of conflict removal. In (a), the conflict is on M1 between shapes A and B and can only be fixed if the gates are spaced apart and area is increased; in (b), the conflict is on M1 between shapes B and C and can be fixed by moving C in the direction of the arrow without increasing area.

to maximize the overlap region of the masks. The details of this implementation follows.

A. *Multiple-spacing rules projection*

We start with DR-dependent projection to identify violating parts as illustrated in Figure 4. From each edge in the layout, side or tip, we find features in its neighborhood that violate the corresponding spacing DR. Unlike previous works that can only allow a single spacing rule, we consider three spacing DRs: side-to-side, tip-to-side, and tip-to-tip. Violating parts that are smaller than the minimum feature size allowed on a single mask are grown within polygons of the original layer to meet the minimum requirement. When a single spacing rule is allowed, the largest spacing rule value must be used as the minimum spacing to ensure no DP conflicts are missed. The advantage of considering multiple spacing rules becomes crucial whenever the values of spacing rules differ, which is the case in the latest technologies. Consider a hypothetical process where the minimum spacing and minimum line width on M1 in the layout are 40nm (M1 pitch equal to 80nm). Let us assume that double-patterning is necessary for the M1 layer and that we would like to achieve 2× pitch relaxation with double patterning. In this case, the DP side-to-side spacing should be 120nm because it defines the pitch; let us assume a smaller value of 90nm for the tip-to-tip spacing and an even smaller value of 80nm for the tip-to-side spacing. We run our mask-assignment framework with this set of DRs and run it again on the same layout with a minimum spacing equal to the largest spacing value, i.e. 120nm. We use a layout of a local clock buffer controller that consists of multiple latches and inverters with roughly 460 transistors. The number of DP native conflicts is reduced from 198 in the case of a single spacing rule to 159 when multiple spacing rules are allowed as shown in Table I (i.e. 20% reduction)[3].

B. *Mask-assignment objectives*

The main objective of DP mask assignment is to assign features to the two different masks with the minimum number of conflicts. A secondary objective is to minimize stitches, which may increase the chances of defects due to the DP overlay error between the first and second exposure layers. Because stitches can remove certain conflicts (as illustrated in Figure 1), we consider all possible stitch locations during the mask assignment and get rid of stitches that do not affect the number of conflicts. If a stitch is introduced inside any violating part, then one of the stitch's sides will have to be assigned to the same mask as the neighboring part that created the violation, which leads to a new DP conflict (as in Figure 2). As a result, stitches should be located only in non-violating parts. Since DP conflicts are between violating parts only, stitches are beneficial (i.e. may reduce the number of conflicts) only if placed in non-violating parts that separate two or more violating parts. In other words, a single stitch is sufficient in such non-violating parts and stitches in a non-violating part that connect to a single violating part is useless because we can always assign such non-violating part to the same mask as the connected violating part (see example of Figure 4). In addition, stitches that cannot guarantee the minimum overlap length of the two masks are disregarded (by joining the connected violating parts).

Although an odd cycle will always result in a DP conflict no matter the mask assignment, deciding what features go on the same mask can affect the efficiency of the conflict removal. To see how, consider the example of Figure 5. This layout contains an odd cycle between shapes A, B, and C. In Figure 5(a), the assignment solution leads to a conflict between shapes A and B that can be resolved only if the gates are spaced apart and, consequently, the layout area is increased; whereas, in Figure 5(b), the assignment solution results in a conflict between shapes B and C that can be resolved by moving C to the right without increasing the layout area. To take advantage of this observation, we make violations in the orthogonal orientation of gates (vertical violations for our layouts) more critical than the ones in other orientations (horizontal and diagonal violations for our layouts). Similarly, we make horizontal violations more critical than diagonal violations because the latter typically require less additional separation to fix.

C. *Implementation details*

The assignment of violating parts to the two masks is straightforward and is done in $O(n)$, where n is the number of violations and candidate stitches. The mask assignment is illustrated through the example of Figure 6 and the details of the algorithm are presented in Figure 7. We start by constructing the conflict graph, where violating parts are represented by nodes and violations and stitches are represented by arcs. We represent vertical violations by solid arcs, horizontal violations by dotted arcs, diagonal violations by double-line arcs, and stitches between two shapes by arcs with two-sided arrows. For each connected component (identifying connected components is $O(n)$), we pick a violation-arc with preference to vertical over horizontal and horizontal over diagonal arcs and assign the two connected nodes to different masks. Whenever a new node is assigned, its connected arcs get added to first-in-first-out queues of the different types of violations and stitches to be processed next. A new arc (possibly a stitch-arc) is picked from the different queues with preference to violation-arcs over stitch-arcs and the same preference for the different violation-arcs as before. This process is repeated until all arcs in the component are processed. Each node is assigned only once: when a violation-arc is processed, the two nodes are assigned to different masks and, when a stitch-arc is processed, the two nodes are assigned to the same masks.

We perform an extra step of mask-assignment flipping to further reduce the number of used stitches. Each part of a component that

[3]The rule values are assumed and may be different in an actual process. Although typically tip-to-tip > tip-to-side > side-to-side in processes at the current technology, we assume a side-to-side spacing larger than tip-to-tip and tip-to-side for this example because side-to-side spacing is expected to be the tightest for DP processes (especially when a trim exposure is used to define tips). Yet, whenever the rule values are different and no matter the actual values, handling multiple spacing rules during the mask-assignment and layout legalization leads to a reduced number of conflicts (or at least the same) compared to when a single minimum spacing rule can be handled.

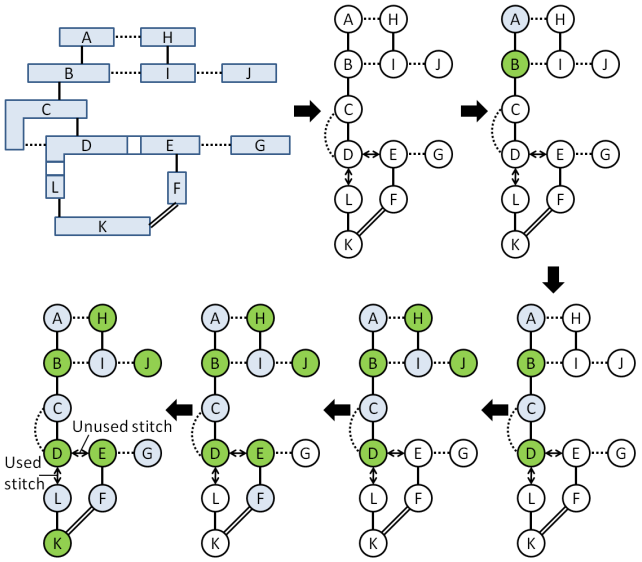


Figure 6: An illustrating example showing each step of the mask assignment process for an isolated region of the layout.

is connected with violation-arcs only (without stitches) is called a sub-component and stitches connects different sub-components. Each sub-component has a flipping score based on which the assignment of its nodes is flipped or kept. When a stitch is processed, we record the connection of the two connected sub-components; if the stitch is used, the flipping score is incremented by one (the score being zero initially); if the stitch is unused, the flipping score is decremented by one. So, by flipping a sub-component with a positive score, the number of stitches is reduced by the amount of the score. We follow a greedy approach and flip sub-components in a decreasing order of scores. When a sub-component is flipped, the sub-component and its neighbors are prevented from future flipping.

D. Stitches vs. conflicts and special cases

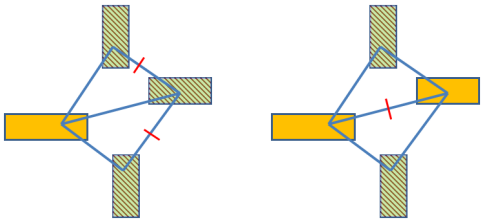
Stitches are manufacturable, DP conflicts are not. When the requirement for the minimum mask overlap is met, stitches are safe to manufacture and their minimization is recommended rather than required. Moreover, stitches may occur in millions in large layouts and reducing the number of stitches by few percents does not have any significant impact on the manufacturing yield. On the other hand, a layout with a single DP conflict can never be manufactured. As a result, our primary objective in this work was to achieve a mask-assignment solution with the least number of DP conflicts. Although our method minimizes the number of stitches, it does not guarantee achieving the minimum number of stitches. Most importantly, because we consider all candidate stitch locations, our method guarantees to reach a solution with DP violations only at the locations of *native conflicts* (i.e. conflicts that cannot be resolved with stitching) and a conflict-free solution for layouts without *native conflicts*. A DP native conflict is defined as an odd cycle in the conflict graph that cannot be resolved with stitching. By performing the mask assignment with a conflict graph that includes all candidate stitches and while ensuring any two nodes with a violation that are not part of an odd cycle are assigned to different masks, our method leads to a solution with zero *non-native conflicts* (i.e. conflicts that are resolvable with stitches).

For some special cases with two or more native-conflict odd cycles share some of their arcs, our method may lead to a solution with non-minimum number of *DP violations* at such native conflicts

[4]Although the loop of line 19 to 22 is theoretically higher than $O(n)$, it takes much less time to execute than the $O(n)$ loop of line 7 to 18 because neighbors of flipped sub-components are skipped and the number of neighbors for a sub-component is less than 3 in most cases and at most 10 in practice.

```
1:  Perform DR-dependent projection.
2:  Identify violating parts.
3:  Identify all useful candidate stitches.
4:  Construct conflict graph with nodes representing violating parts and
    four types of arcs representing vertical violations, horizontal viola-
    tions, diagonal violations, and stitches.
5:  Determine connected components.
6:  Determine connected sub-components (i.e. without stitch connections).
7:  for all Connected components do
8:      Pick any violation-arc with preference to vertical over horizontal
        and horizontal over diagonal and assign its nodes to different masks.
9:      Add arcs connected to the assigned node to FIFO queues for the
        different types of arcs.
10:     Pick an arc (possibly a stitch-arc) from the different queues with
        preference to violation-arcs over stitch-arcs and the same preference
        for the different violation-arcs as above.
11:     Assign the two nodes connected to the arc to different masks if the
        arc is for a violation and to the same mask if the arc is for a stitch.
12:     if Arc is a stitch then
13:         Record the connection of the two sub-components (the two nodes
            connected to the stitch-arc belong to different sub-components).
14:         If the stitch is used (i.e. connected nodes were assigned to
            different masks), increment the flipping-score of the two sub-
            components by one.
15:         If the stitch is unused (both connected nodes assigned to same
            mask), decrement the flipping-score of the two sub-components
            by one.
16:     end if
17:     Repeat 10 and 15 until all arcs in the component are processed.
18: end for
19: for all Sub-component with +ve flipping score sorted by higher score
    do
20:     Skip if already processed or marked not to be processed
21:     Mark as flipped and processed and mark its neighbors not to be
        processed
22: end for
23: for all Nodes do
24:     Flip node if it belongs to a flipped sub-component
25: end for
```

Figure 7: Overview of mask-assignment procedure[4].



(a) **(b)**

Figure 8: Example showing two mask-assignment solutions that our method may give for the same layout (rare case with all diagonal violations) depending on the propagation order of the assignment: (a) with two DP violations and (b) with a single DP violation.

depending on the propagation order of the mask assignment. One such special case is shown in Figure 8). Because we set a propagation preference with purely vertical violations first, diagonal violations second, and purely horizontal violations last, all violations in this four-tip configuration must be diagonal violations for the method to result in the assignment with two DP violations (Figure 8(a)) for some propagation order; otherwise the method will result in the assignment with a single DP violation as in Figure 8(b). Besides the peculiarity of this layout, such four-tip configuration may never occur because contacts/vias are on tracks in actual layouts. Furthermore, the number of DP violations may not reflect the amount of effort needed to remove the violations with layout perturbations and, in this special case, the assignment with two DP violations may be easier to fix than that with a single violation depending on the layout (at the same layer as well as the top and bottom layers).

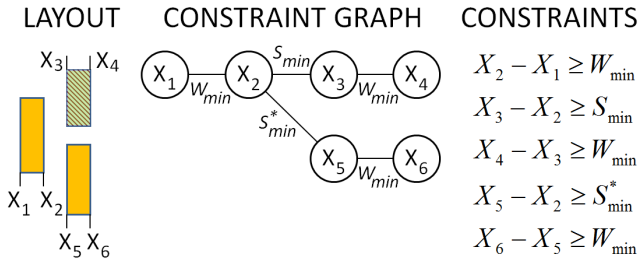


Figure 9: Example of x-direction constraint graph construction and constraint definition for a double-patterned layer. W_{min} is the minimum width rule, S_{min} is the side-to-side spacing rule in the layout, and S^*_{min} is the side-to-side spacing rule for features on the same mask.

III. Conflict Removal while Minimizing Perturbation

After the DP layout decomposition with the minimum number of conflicts is complete, our objective is to make the layout compatible with DP and resolve the conflicts while minimizing layout perturbation. We use the method proposed in [12] for layout legalization with minimum perturbation as the objective. The layout is represented as a constraint graph where nodes correspond to the layout edges and arcs correspond to the DRs that need to be met between any two layout edges. Arcs are assigned weights that correspond to the values of rules as illustrated in Figure 9. Layer-to-layer connectivity is maintained through the DRs between the layers, which are represented in the graph by arcs between nodes of the different layers.

The two mask layouts of any double-patterned layer are defined as stand-alone layers. Spacing DRs between features of the same mask including side-to-side, tip-to-side, and tip-to-tip are mapped into arcs between the nodes of the stand-alone mask layer in the constraint graph. DRs that define the interaction between the two mask layouts (e.g., minimum overlap length) are mapped into arcs between the nodes of the two stand-alone mask layers. For the interactions across different layers in the stack (e.g., M1 and contacts), we define any double-patterned layer as the union of its two mask layouts and map across-layers DRs into arcs between nodes of the union layers[5].

As in layout compaction, the two-dimensional minimum perturbation problem is simplified by solving the one-dimensional problem successively (in x and y directions). The 1D minimum perturbation problem is formulated as a LP as follows.

$$\begin{aligned} Minimize \quad & \sum_i W_i |X_i - X_i^{init}| \\ Subject\ to: \quad & X_j - X_i \geq d_{ij}, \forall A_{ij}, \end{aligned}$$

where X_i and X_i^{init} are the current location and the initial location of node i, W_i is the weight for the perturbation of node i from its initial location. A_{ij} is the arc between nodes i and j, which represents the DR constraint between the two layout elements, and d_{ij} is the weight of arc A_{ij}, which represent the value of the DR.

Figure 9 shows the construction of the constraint graph and the definition of constraints in the x-direction for an example double-patterned layout.

We obtain an equivalent formulation to the original problem with a linear objective function by introducing two new variables L and R for each node i as follows (details in [12]):

$$\begin{aligned} Minimize \quad & \sum_i W_i (R_i - L_i) \\ Subject\ to: \quad & X_j - X_i \geq d_{ij} \quad \forall A_{ij} \\ & L_i \leq X_i, L_i \leq X_i^{init} \quad \forall i \\ & R_i \geq X_i, R_i \geq X_i^{init} \quad \forall i. \end{aligned}$$

This formulation permits the application of the method for practical layouts that use a discrete manufacturing grid for the coordinates.

[5]Rather than using layers of the mask layouts and have the same problem highlighted in Figure 2.

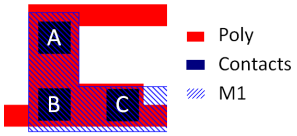


Figure 10: Group of redundant contacts connecting to the poly layer. Contact B has more flexibility of movement than contacts A and C and, thus, we pick B as the required contact and A and C as redundant contacts that may be sacrificed if necessary to resolve conflicts.

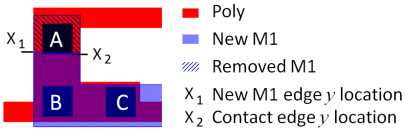


Figure 11: Illustration of M1 simplification for possible sacrifice of contacts.

According to the total unimodularity property [15], when all X_i^{init} and d_{ij} are integers, the solution of the problem consists of integers only. The handling of gridded design rule constraints can be achieved as in [16]. The target on-grid locations are determined and on-grid constraints are relaxed to spacing constraints between the target locations and the cell boundary. After this relaxation, the problem is still formulated and solved as a linear program as detailed in [16].

To handle infeasible constraints, we relax the unsatisfied arc constraints such that all constraints are feasible and a penalty is added in the objective function for the originally infeasible constraint. Section IV gives more details about the handling of infeasible constraints and in-depth details can be found in [12].

Our formulation of the problem maintains all inter and intra layer connectivities, which are represented as constraints in the graph. Internal connectivity of double-patterned layers at stitches is maintained through the minimum overlap length constraint. Moreover, the formulation permits the legalization of the layout for double patterning and the removal of DP conflicts across all layout layers simultaneously.

IV. Sacrificing Non-Crucial Features When Necessary

In actual layouts, we observe that many conflicts on the M1 layer are caused by segments that are added to cover redundant contacts/vias or to maximize the pin-access region. Redundant contacts and vias improve manufacturability, but they are not absolutely required. The same is true for pin segments that are used only to maximize the access region and, consequently, improve the routing efficiency. We take advantage of this observation and, as an option, we allow the *possible* sacrifice of redundancy and extra pin segments to improve the results of the DP conflict removal framework.

A. Sacrifice of redundant contacts/vias

The process of identifying redundant vias is similar to the process of identifying redundant contacts and, for brevity, we only describe the latter process. We start by finding overlap regions of the top layer (M1) and the bottom layer (poly or active). If a single polygon of the overlap region interacts with two or more contacts, these contacts are identified as a group of redundant contacts. Next, we choose one of the contacts from each group to be a required contact/via and add all such required contacts to single contacts to form a new layer of required contacts. The remaining contacts that were not chosen as required contacts are considered redundant. The choice of the required contact among a group is made with preference to the contact with the highest flexibility of movement as illustrated in Figure 10. Contacts that were considered redundant are assigned to a new layer.

If M1 is double patterned, the line-end part of M1 that covers a redundant contact is removed, as shown in Figure 11, and overlapping redundant contacts with M1 is specified as a recommended, but not required, constraint. The LP of the conflict removal method will meet this recommended constraint only when possible without creating a DP conflict or any DR violations. In other words, redundant

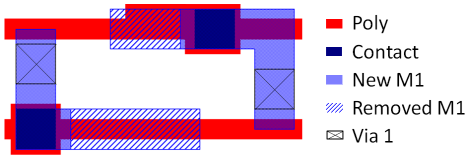


Figure 12: Illustration of M1 simplification for possible sacrifice of pin segments.

contacts will be sacrificed only when necessary to resolve conflicts. To ensure recommended contacts still get a chance to be covered by M1 after the layout is perturbed, we add a required constraint to keep redundant contacts at the same spacing and aligned to the corresponding required contact chosen among the group of redundant contacts.

B. Sacrifice of pin segments

M1 pin segments that do not connect to any other layer in the layout stack are removed for possible sacrifice as shown in Figure 12. To allow the layout perturbation to recover the removed parts when possible without creating violations, the original M1 layer is kept and a recommended constraint is added to the LP problem to minimize the distance between the new M1 edge and the original M1 edge.

The removal of M1 pin segments and M1 parts that cover redundant contacts/vias is performed before the DP mask assignment. This way, because violations are reduced, extra candidate stitches can be identified and taken advantage of to reduce DP conflicts. When the sacrifice is not necessary to resolve conflicts, these extra stitches will be removed by the mask assignment algorithm (by assigning the violating parts of a stitch to the same mask) and the layout perturbation will recover the sacrificed parts as described earlier.

C. Handling recommended constraints during legalization

Recommended constraints are handled in the LP formulation of the conflict removal framework in a similar way as infeasible constraints are handled, i.e. by introducing a new variable to relax the constraint and minimizing this relaxation variable in the objective function. This is illustrated through the example of Figure 11 where M1 covering the redundant contact A is set as a recommended constraint. Here, X_1 is the location of the M1 edge closest to the redundant contact A and X_2 is the location of the edge of the redundant contact closest to the M1 edge with X_1 location. The constraint is then $X_2 - X_1 + r_{12} =$ *contact width rule* $+$ *M1 overlap past contact rule*. r_{12} is included in the objective function so that it is minimized. The minimization of relaxation variables for recommended constraints is given less priority than the minimization of the relaxation variables of infeasible required constraints (by assigning a smaller weight in the objective function). This way, recommended constraints are met only when possible without creating any DP conflicts, DR violation, or area increase.

It is worth noting that the minimization of the relaxation variables can be weighted according to the importance of what is being sacrificed (e.g., pin-access metric such as in [17]).

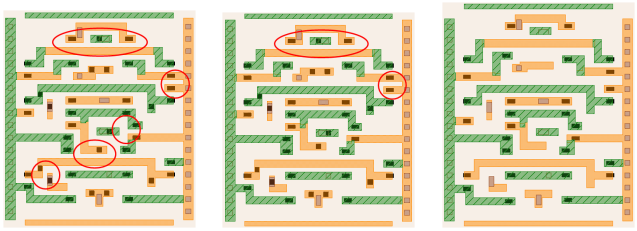
V. Experimental Setup and Results

The DP mask assignment was implemented using Calibre SVRF code [18] and C++ with OpenAccess database. The DP conflict removal with layout perturbation was implemented and integrated into the minimum perturbation based VLSI artwork legalization system [12].

We test our mask assignment approach on the testcases presented in [4] and compare the results with the two mask assignment approaches proposed by the previous work (ILP-based). We use the same layouts for designs ranging from 100K cells to 500K cells as well as the same DP minimum spacing and overlap length as in [4]. Table II shows the results for the mask assignment of the poly layer at the 45nm process node (details about testcases and process can be found in [4]). The results show that our approach is 92× to 233× faster than the pure ILP approach while the number of stitches is

Table II: Results of our DP mask assignment at the poly layer (with DP minimum spacing equal to 90nm) and comparison with previous work of ILP-based mask assignment as well as the conflict cycle detection (CCD) based approach [4]. "Cuts" refers to stitches, "Secs" refers to used stitches, and "min" refers to the minimum overlap length).

		ILP [4]		CCD [4]		Our approach	
Design	**min**	**Cuts**	**Secs**	**Cuts**	**Secs**	**Cuts**	**Secs**
ART-A 45(70%)	8	24290	564.6	25521	378.6	25480	6.1
ART-B 45(70%)	10	72828	2887.4	76550	2316.8	76634	20.5
ART-C 45(70%)	8	121916	8291.2	127935	7895.8	126715	35.5
ART-A 45(90%)	13	25432	612	26629	391	27691	6.3
ART-B 45(90%)	10	76292	2892.2	79836	2355.2	82089	20.5
ART-C 45(90%)	8	126238	8129	132303	8205	135558	37.5



(a) **(b)** **(c)**

Figure 13: Sample results for a cell layout: (a) before DP conflict removal, (b) after conflict removal with fixed area, and (c) after conflict removal with area increase.

larger by a modest 4% to 8.8%. Compared with the conflict cycle detection approach of [4], our method is 62× to 223× faster and lead to almost the same number of stitches (ranging from -1% to +4%).

Our DP conflict removal framework was tested on a commercial 22nm standard-cell and macro layouts. We assume M1 is double patterned and apply the conflict removal method for layouts that have DP conflicts. The M1 minimum spacing in the layout is 40nm and we use a value of 15nm for the minimum overlap length and 80nm for the side-to-side, tip-to-side, and tip-to-tip spacing[6].

In one experiment, we apply our DP conflict removal method to standard cells. The results show that DP conflicts in many cells were completely removed without area increase and any DR violations. For some other cells, few DP conflicts remain unresolvable when the area is fixed. We give two options to deal with such stubborn conflicts. The first option is to keep these conflicts and report their locations so that the layout designer fixes them manually. The second option is to run the conflict removal framework a second run with non-fixed area so that all conflicts are removed. Figure 13 shows an example layout where M1 is double-patterned before and after the layout perturbation to remove conflicts. As Figure 13(b) depicts, the conflict removal method with fixed area is able to remove three out of the four conflicts in the original layout of Figure 13(a). The remaining stubborn conflict is removed when poly and active are allowed to move and area is allowed to increase as shown in Figure 13(c). In this case, the restrictive DR of poly on grid are met by modifying the LP program as described in [16]. A summary of the results is given in Table III. For all cells, the runtime for the entire conflict removal flow (mask assignment plus conflict removal) is less than 10 seconds in real time. In six out of ten cells, all DP conflicts

[6]Because DP was assumed for M1 in the process and to avoid making inadequate assumptions on the differences between the spacing values that we cannot justify, we use the same value for the different rules in the experiments. Nevertheless, the benefits of handling multiple spacing rule values was shown through the example of Section II-A.

Table III: Results of applying our DP conflict removal method with and without area increase to cells from a commercial 22nm library (CA stands for contacts).

	Original		Conflict Removal w/o Area Increase		Conflict Removal w/ Area Increase		
Layout	**Normalized Area**	**Conflicts**	**Conflicts**	**Sacrificed Red. CA**	**Area Overhead**	**Conflicts**	**Sacrificed Red. CA**
LCB + latch 1	1	**1**	**0**	0	-	-	-
latch1	1.6	**1**	**0**	0	-	-	-
oai	1.6	**2**	**0**	0	-	-	-
xor	2.4	**3**	**0**	0	-	-	-
nand4	4.7	**4**	**0**	0	-	-	-
nand3	6.7	**7**	**0**	0	-	-	-
scan latch	2.3	**5**	**2**	0	6.2%	**0**	0
latch2	4.3	**15**	**7**	0	6.6%	**0**	0
latch3	5.3	**3**	**1**	0	5.4%	**0**	0
LCB + latch 2	13.7	**14**	**3**	2	8.3%	**0**	4

Table IV: Results of applying our DP conflict removal method with and without area increase to the layout of a macro (CA stands for contacts).

	Original		Conflict Removal w/o Area Increase		Conflict Removal w/ Area Increase		
Layout	**Normalized Area**	**Conflicts**	**Conflicts**	**Sacrificed Red. CA**	**Area Overhead**	**Conflicts**	**Sacrificed Red. CA**
LCB control. (460 transistors)	50.3	**61**	**24**	2	6.8%	**2**	2

were removed without any area increase or the removal of redundant contacts. In the remaining four cells, few DP conflicts remain after applying our method with fixed area. When we allow the layout area to increase, all conflicts are removed in these cells with an average 6.7% area overhead (at most 8.3% overhead) and with the removal of four redundant contacts in just one of the cells.

In another experiment, we apply our DP conflict removal method for an entire macro, a local clock buffer controller that consists of multiple latches and inverters with roughly 460 transistors. The results are given in Table IV. The method is able to reduce the number of DP conflicts from 61 to 24 without increasing the layout area and down to just two conflicts with an area increase of 6.8% and the removal of two redundant contacts. The runtime of the entire flow for this macro layout is less than one minute in real time (< 2 seconds CPU time).

VI. Conclusions

We proposed a novel framework to enable DP in the design. The mask assignment guarantees a conflict-free solution for layouts without native conflicts and is performed using a $O(n)$ algorithm. The automated DP conflict removal and layout legalization are performed simultaneously across all layout layers while minimizing perturbation using a LP. The method enables designing with conventional DRs and masks the designer from the complexity in dealing with DP layers and requirements. The way we formulate the problem allowed us to achieve high-quality results with extremely fast run-time (under 10 seconds in real time for typical cells). The method targets primarily standard-cell layouts. Complete standard-cell based designs can be formed, as in [19], either by fixing the colors at the cell-boundaries, to ensure that flipping the cell coloring resolves all DP conflicts that may be induced by cell placement, or by using a correct-by-construction approach, where enough spacing is allocated between the colored features of the cell and the cell boundary to prevent placement-induced conflicts. The method is not limited to standard cell-based designs, however, and it can also be applied for full-custom layouts and interconnect layers in complete designs.

VII. Acknowledgments

The authors would like to thank Kevin W. McCullen and Rani R. Narayan from IBM for their help with setting up the minimum perturbation based VLSI artwork legalization system [12].

References

[1] M. Drapeau, V. Wiaux, E. Hendrickx, S. Verhaegen, and T. Machida, "Double patterning design split implementation and validation for the 32nm node," in *Proc. SPIE*, 2007, pp. 652 109–1.

[2] T.-B. Chiou *et al.*, "Development of layout split algorithms and printability evaluation for double patterning technology," in *Proc. SPIE*, 2008, pp. 69 243M–1.

[3] A. Tritchkov *et al.*, "Double-patterning decomposition, design compliance, and verification algorithms at 32nm hp," in *Proc. SPIE*, 2008, pp. p. 71 220S–1.

[4] A. B. Kahng, C.-H. Park, X. Xu, and H. Yao, "Layout decomposition approaches for double patterning lithography," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 6, pp. 939–952, 2010.

[5] K. Yuan, J.-S. Yang, and D. Z. Pan, "Double patterning layout decomposition for simultaneous conflict and stitch minimization," in *Intl. Symp. on Physical Design, ISPD'09*, 2009, pp. 107–114.

[6] M. R. Garey and D. S. Johnson, *A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[7] J.-S. Yang, K. Lu, M. Cho, K. Yuan, and D. Z. Pan, "A new graph theoretic, multi-objective layout decomposition framework for double patterning lithography," in *IEEE Asian and South Pacific Design Automation Conference, ASPDAC'10*, 2010, pp. 637–644.

[8] W.-S. Luk and H. Huang, "Fast and lossless graph division method for layout decomposition using spqr-tree," in *IEEE Intl. Conf. on Computer-Aided Design, ICCAD'10*, 2010, pp. 112–115.

[9] C. Hsu, Y. Chang, and S. R. Nassif, "Simultaneous layout migration and decomposition for double patterning technology," in *IEEE Intl. Conf. on Computer-Aided Design, ICCAD'09*, 2009, pp. 595–600.

[10] K. Yuan and D. Z. Pan, "Wisdom: wire spreading enhanced decomposition of masks in double patterning lithography," in *IEEE Intl. Conf. on Computer-Aided Design, ICCAD'10*, 2009, pp. 32–38.

[11] S.-Y. Chen and Y.-W. Chang, "Native-conflict-aware wire perturbation for double patterning technology," in *IEEE Intl. Conf. on Computer-Aided Design, ICCAD'10*, 2009, pp. 556–561.

[12] F. L. Heng, Z. Chen, and G. Tellez, "A vlsi artwork legalization technique based on a new criteria of minimum layout perturbation," in *ACM/IEEE Intl. Symp. on Physical Design, ISPD'97*, 1997, pp. 116–121.

[13] J. Zhu, F. Fang, and Q. Tang, "Calligrapher: a new layout-migration engine for hard intellectual property libraries," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 9, pp. 1347–1361, 2005.

[14] N. Karmarkar, "A new polynomial time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, 1984.

[15] B. D. Cullity, *Combinatorial optimization: algorithms and complexity*. New York, NY: Prentice-Hall, 1996.

[16] X. Yuan, K. W. McCullen, F.-L. Heng, R. F. Walker, J. Hibbeler, R. J. Allen, and R. R. Narayan, "Technology migration technique for designs with strong ret-driven layout restrictions," in *ACM/IEEE Intl. Symp. on Physical Design, ISPD'05*, 2005, pp. 175–182.

[17] T. Taghavi *et al.*, "New placement prediction and mitigation techniques for local routing congestion," in *IEEE Intl. Conf. on Computer-Aided Design, ICCAD'10*, 2010, pp. 621–624.

[18] Calibre standard verification rule format manual v2009. [Online]. Available: http://www.mentor.com/

[19] L. Liebmann, D. Pietromonaco, and M. Graf, "Decomposition-aware standard cell design flows to enable double-patterning technology," vol. 7974. SPIE, 2011, p. 79740K.