Directed Self-Assembly Based Cut Mask Optimization for Unidirectional Design

Jiaojiao Ou†, Bei Yu†, Jhih-Rong Gao†, Moshe Preil†, Azat Latypov‡, David Z. Pan†
†ECE Department, University of Texas at Austin, Austin, TX USA
‡GLOBALFOUNDRIES Inc, Santa Clara, CA USA
{jiaojiao, bei, jrgao, dpan}@cerc.utexas.edu, {moshe.preil, azat.latypov}@globalfoundries.com

ABSTRACT
Unidirectional design has attracted lots of attention with the scaling down of technology nodes. However, due to the limitation of traditional lithography, printing the randomly distributed dense cuts becomes a big challenge for highly scaled unidirectional layout. Recently directed self-assembly (DSA) has emerged as a promising lithography technique candidate for cut manufacturing because of its ability to form small cylinders inside the guiding templates and the actual pattern size can be greatly reduced. In this paper, we perform a comprehensive study on the DSA cut mask optimization problem. We first formulate it as integer linear programming (ILP) to assign cuts to different guiding templates, targeting at minimum conflicts and line-end extensions. As ILP may not be scalable for very large size problem, we further propose a speed-up method to decompose the problem into smaller ones and solve them separately. We then merge and legalize the solutions without much loss of result quality. The proposed approaches can be easily extended to handle more DSA guiding patterns with complicated shapes. Experimental results show that our methods can significantly reduce the total number of unresolvable patterns and the line-end extensions for the targeted layouts.

Categories and Subject Descriptors
B.2.7 [Hardware, Integrated Circuit]: Design Aids

Keywords
DSA; Cut Mask; Integer Linear Programming

1. INTRODUCTION
As traditional 193i lithography has been pushed to its limit with the scaling down of technology node, industry is looking for the next generation lithography technology, such as multiple patterning lithography, E-beam lithography, extreme ultra violet (EUV), and directed self-assembly (DSA) [1, 2]. Recently, DSA has received a lot of attention due to its low cost and high throughput for manufacturing of dense wires and contact layers [3, 4].

Several works have been done on the investigation of DSA contact layer fabrication and DSA aware contact layer optimization. Yi et al. [5] demonstrated the fabrication of DSA contacts for unidirectional standard cells. Du et al. [6] proposed a DSA aware contact layer optimization method for 1D standard cell design. Du et al. [7] also proposed a DSA aware routing optimization method which targets at reducing the manufacturing cost and throughput of the via layer.

With the dramatically increased design rules of 2D designs for highly scaled technology nodes, industry is also moving towards the unidirectional design due to its large process window and improved manufacturability [8]. However, manufacturing randomly distributed dense cuts for the metal lines of unidirectional design has become a big challenge for traditional pattern lithography. Firstly, for traditional pattern technique, as shown in Figure 1(b), the cuts may cut across adjacent metal lines. Secondly, for some layouts with tip to tip patterns, traditional lithography might cut extra lines off, thus affecting the connectivity of contacts.

Xiao et al. [9] proposed a DSA cut guiding templates redistribution method for unidirectional standard cell design. The DSA templates are determined for all the cuts by scanning the cuts from the four corners of the pre-constructed conflicted graph. Then these templates are legalized in order to remove overlaps. For overlaps which can not be removed, they are marked as conflicts. However, the proposed process in [9] can not guarantee an optimal result. In Template Match step, although it selected the large template to avoid unmarked template, it neglected the relative position of these templates, which introduced a lot of overlap conflicts. Thus, it would take extra efforts and more wire extensions to remove these conflicts.

Xiao et al. [9] proposed a DSA cut guiding templates redistribution method for unidirectional standard cell design. The DSA templates are determined for all the cuts by scanning the cuts from the four corners of the pre-constructed conflicted graph. Then these templates are legalized in order to remove overlaps. For overlaps which can not be removed, they are marked as conflicts. However, the proposed process in [9] can not guarantee an optimal result. In Template Match step, although it selected the large template to avoid unmarked template, it neglected the relative position of these templates, which introduced a lot of overlap conflicts. Thus, it would take extra efforts and more wire extensions to remove these conflicts.

In this paper, we perform a comprehensive investigation on the DSA based end-cutting problem. We first formulate an ILP to search for optimal solution with minimum wire extensions and conflicts for all test cases. Since for large scale ILP problem the runtime would increase exponentially, we further divide a large layout into several smaller pieces and...
2.2 DSA Guiding Template Type

The shape and size of DSA guiding templates can be adjusted to generate different DSA patterns [17]. However, the variation or overlay of the DSA patterns would increase with the growth of DSA guiding template size and complexity.

For some complex guiding templates, they can even generate some unwanted cylinders inside the template [18]. As reported in [10], some regular DSA guiding templates, shown in Figure 2 could have reasonably good variation controllability and manufacturability. In order to ensure the throughput and manufacturability of the DSA cut for metal lines, we would adopt these regular DSA patterns in our implementation.

2.3 Problem Formulation

We define the DSA cut mask optimization problem as follows:

Problem 1 (DSA Cut Mask Optimization) Given a unidirectional layout, we are going to cut the metal lines with a predefined DSA guiding template set by assigning cuts to different DSA guiding templates. The target is to minimize the number of unpatternable cuts and total wire extensions by extending the two ends of the wire to fit them into the DSA guiding template.

3. ALGORITHMS

In this section, we first propose an ILP formulation to find the optimal solution for the problem. Since the ILP formulation is time consuming for very large problem size, we then propose a speed-up method to efficiently solve the problem. Because the guiding patterns are fixed and the metal wires can be extended, it is more difficult than traditional end-cutting problem.

3.1 ILP Formulation

Objective:

Given a unidirectional layout, we are trying to minimize the wire extensions and number of conflicts when applying DSA guiding templates to cut the wires.

\[
\text{Minimize } \sum_{i=1}^{n} (x_{2i} - x_{2i-1} - l_i) + W \sum_{i=1}^{2n} \varepsilon_i
\]

where \(W\) is a constant which denotes the weight of the conflicted cut pattern. If a cut is marked as a conflict, then we should either redesign the layout or adopt other complimentary lithography to remove the conflict, the cost will be quite expensive, thus we define \(W\) to be a very big number to reduce the number of conflicts. \(\varepsilon_i\) is a binary variable indicating the conflicted cut \(i\) if it can not be assigned to any DSA template. \(x_{2i-1}\) and \(x_{2i}\) are continuous variables indicating the left and right side of wire \(i\), they can also be considered as the horizontal positions for cut \(2i - 1\) and \(2i\). \(l_i\) is a constant which indicates the original length of wire \(i\). \(n\) is the number of wires for the layout.

Constraints:

C1: Line-end extension

The line ends of metal wires can only be extended to the left or right of the original design so as not to interrupt the
circuit logical connectivity.

\[
\begin{aligned}
&x_{2i-1} \leq L_i \\
&x_{2i} \geq R_i \\
&x_{2i} - x_{2i-1} \leq l_i + \sigma_i, \quad i = 1, 2, ..., n \\
&x_{2i-1} \geq LL \\
&x_{2i} \leq RR
\end{aligned}
\]

\(\sigma_i\) indicates the wire extension limit for each wire. \(L_i\) and \(R_i\) are constant number indicate the original left and right x-coordinates of wire \(i\). \(LL\) and \(RR\) are the left and right bound of the layout.

**C2:** Constraints for templates

For the two line-ends of a wire, we can consider them as two cuts. For each cut, it can be combined with other cuts to be fitted in one DSA guiding template pattern. There might be several templates candidates for each cut, e.g. cut \(i\) can be printed by a single-hole template, or it can be combined with cut \(j\) and be printed with a two-hole template. Since each cut should be printed by only one template, the constraint is as follows:

\[
\sum_{t_j \in \mathcal{T}_i} t_j = 1, \forall i \in \{1, 2, ..., n\}
\]

where \(\mathcal{T}_i\) is the potential guiding template set for cut \(i\), it includes all the possible DSA guiding templates to print cut \(i\). \(t_j\) is a binary variable indicating one of the potential DSA guiding templates. For \(t_j \in \mathcal{T}_i\), if \(t_j = 1\), cut \(i\) is printed by \(t_j\). The sum of all potential templates for cut \(i\) should be 1. \(m\) indicates the total number of potential DSA guiding templates for cut \(i\).

**C3:** Minimum distance between adjacent templates

Because DSA guiding templates are printed by traditional 193i lithography, adjacent templates should be a minimum distance away from each other. If cut \(i\) and cut \(j\) are in different templates, without any loss of generality, we use \(x_i\) and \(x_j\) to indicate the x-coordinate of the two cuts in adjacent templates.

1. Cuts on the same track
   If two cuts are on the same track, we assume that the initial cut \(i\) is on the right side of cut \(j\), then \(x_i\) is always at least \(\min_z\) larger than \(x_j\). However, if the two cuts are in the same template \(t_k\) or one of the cuts is marked as conflict, then there is no such constraint.

   \[
   x_i - x_j + B \times (\varepsilon_i + \varepsilon_j + \sum t_k) \geq \min_s, \quad t_k \in (T_i \cap T_j)
   \]

   where \(B\) is a very large constant, \(\min_s\) is the minimum template spacing.

2. Cuts on adjacent track
   If cut \(i\) and cut \(j\) are on adjacent tracks, since line end can be extended, cut \(i\) can be on the left or right side of cut \(j\).

   \[
   \begin{aligned}
   &x_i - x_j + B \times (\varepsilon_i + \varepsilon_j + \sum_{t_k \in (T_i \cap T_j)} t_k + d'_i) \geq \min_s \\
   &x_j - x_i + B \times (\varepsilon_i + \varepsilon_j + \sum_{t_k \in (T_i \cap T_j)} t_k + 1 - d'_j) \geq \min_s
   \end{aligned}
   \]

   where \(d'_i\) is a binary variable denoting the relative position of cut \(i\) and cut \(j\). If \(d'_i = 1\), \(x_i\) is on the left side of \(x_j\). If \(d'_i = 0\), \(x_i\) is on the right side of \(x_j\).

**C4:** Constraints inside templates

For different DSA guiding patterns, the positions of the inside cuts are different. For two-hole and three-hole vertical templates, the cuts inside them should be aligned vertically; for four-hole template, the horizontal cuts have a certain distance between them, and the vertical aligned cuts have the same horizontal coordinates. If cut \(i\) and cut \(j\) are in the same template \(t_k\):

\[
\begin{aligned}
&x_i - x_j + B \times (1 - t_k) \geq \theta_k \\
&x_i - x_j - B \times (1 - t_k) \leq \theta_k
\end{aligned}
\]

where \(\theta_k\) indicates the required distance for cut \(i\) and cut \(j\) in template \(t_k\).

### 3.2 Speed-up Method

For large scale size problem, the initial ILP formulation may be quite computational intensive, due to the NP-hardness of ILP problem. For the current biggest benchmark, the number of variables could be more than 90K, and constraints could be more than 70K.

In order to reduce runtime, we can divide the original layouts into several pieces and apply the cut mask optimization algorithm on them separately. After we find the optimal solution for each group, we then merge them together. And there might exist some conflicts on the boundary of each group, therefore we need to remove conflicts after they are merged.

The method consists of three stages: (1) layout division, (2) ILP solution, (3) layout legalization. The ILP formulation is the same as the initial formulation, the layout division and layout legalization are performed as follows. For layout division, since in the unidirectional layout, some cuts are far away from other adjacent cuts, as shown in Figure 3, there is low possibility that these far away cuts will conflict with each other, thus we can group these adjacent cuts together and find the solution for it. However, if the number of variables in one group is smaller than a certain value, the total initialization time of the ILP solver might exceed the program running time that have been saved, therefore, the number of cuts in each group should be larger than a certain threshold value. Because the number of variables for each group is much less than the original problem, the runtime for each group can be greatly reduced.

After all the optimal solution for each group have been found, there might exist some conflicts or overlaps between DSA guiding templates in different layout groups, we then remove these conflicts by merging adjacent cuts into one or group them into a new DSA templates.

### 4. EXPERIMENTAL RESULTS

Our algorithms are implemented in C++ and all the experiments are performed on a Linux workstation with Intel Core i7 3.71GHz CPU and 8GB memory. The state-of-the-art ILP solver, Gurobi [19], is used to solve the ILP formulations.

For fair comparison, the six test benchmarks we used in this work are from the authors of [9]. The benchmarks represent the unidirectional metal layers and they are in different sizes. We adopted the DSA guiding template design rule used in [9]. According to the author, the benchmarks are slightly different.
because the original benchmarks have been overwritten, but the benchmarks are generated by the same method. Since [9] is implemented in MATLAB, the binary is not available to us, we therefore implemented their method to our best effort for comparison.

Table 1 shows the results of comparison in terms of number of conflicts, total wire extensions and runtime for each test benchmark. The second column shows the total number of cuts for each test benchmark. As can be observed from table 1, our ILP formulation and speed-up method have no template conflicts for all the test cases, while on average [9] has 30 remaining conflicts after legalization. Our ILP formulation can achieve about 77% less wire extensions on average, and our speed-up method has about 69% less wire extensions on average than [9]’ method. And the speed-up method is about 2934× faster than ILP on average for all the test benchmarks.

5. CONCLUSION

In this paper, we perform a thorough investigation on the DSA based end-cutting problem for unidirectional layout. Two different approaches have been proposed to assign cuts into different DSA guiding templates. The first ILP approach solves the problem and provides a result with 77% less wire extensions and zero conflicts compared to [9]. The speed-up method solves the problem much more efficiently, with 69% less wire extension and zero conflicts. Both of our methods are generic and can be adaptive to different DSA guiding template shapes, which will be beneficial with the development of DSA manufacturing technique.

6. ACKNOWLEDGMENT

This work is supported in part by National Science Foundation (NSF) and Semiconductor Research Corporation (SRC).

7. REFERENCES