Triple Patterning Aware Detailed Placement Toward Zero Cross-Row Middle-of-Line Conflict

Yibo Lin[1], Bei Yu[1,2], Biying Xu[1], and David Z. Pan[1]

[1]ECE Department, University of Texas at Austin, Austin, TX, USA
[2]CSE Department, Chinese University of Hong Kong, NT, Hong Kong
{yibolin, bei, biying, dpan}@cerc.utexas.edu

***Abstract*—Triple patterning lithography (TPL) is one of the most promising lithography technology in sub-14nm technology nodes, especially for complicated low metal layer manufacturing. To overcome the intra-cell routability problem and improve the cell regularity, recently middle-of-line (MOL) layers are employed in standard cell design. However, MOL layers may introduce a large amount of cross-row TPL conflicts for row based design. Motivated by this challenge, in this paper we propose the first TPL aware detailed placement toward zero cross-row MOL conflict. In standard cell pre-coloring, boolean based look-up table is proposed to reduce solution space. In detailed placement stage, two powerful techniques, i.e., local reordered single row refinement (LRSR) and min-cost flow based conflict removal, are proposed to provide zero TPL conflict solution. The experimental results demonstrate the effectiveness of our proposed methodologies.**

I. Introduction

With the scaling of the feature size in sub-14nm technology nodes, semiconductor industry is challenged by the manufacturability with conventional 193nm wavelength immersion (193i) lithography. Although under intensive research and development, the next generation lithography techniques, such as extreme ultra-violet lithography, directed self-assembly, electron beam lithography, and nanoimprint lithography, are postponed due to yield and throughput issues [1], [2]. Multiple patterning lithography (MPL), which reuses conventional 193i lithography, currently has been heavily utilized in industry [3]. For instance, double patterning lithography (DPL) has been introduced in 14nm technology node [4]. In emerging sub-14nm technology nodes, LELELE type triple patterning lithography (TPL) is a very promising option for complicated low metal layer manufacturing [5].

As a fundamental and critical problem, layout decomposition for TPL or even general MPL has been studied extensively in the literature [6]–[11]. Due to the NP-hardness [12], most layout decomposers apply heuristic methods to search for near-optimal solutions, thus there may be a large amount of conflicts left in decomposed layout. To overcome this limitation, several studies integrate patterning constraints in early design stages. For example, how to introduce TPL friendly design in detailed routing has been discussed in [13]–[16]. Besides, there are several studies proposing different methodologies to TPL aware detailed placement [17]–[22]. For ordered single row problem, Yu et al. [17], [18] proposed a unified graph model to cell placement and coloring assignment. Kuang et al. [19] and Chien et al. [20] recently further improved the detailed placement solutions in [17]. A special case of TPL aware single row placement was considered in [21], [22], where each type of standard cell has only one final coloring solution.

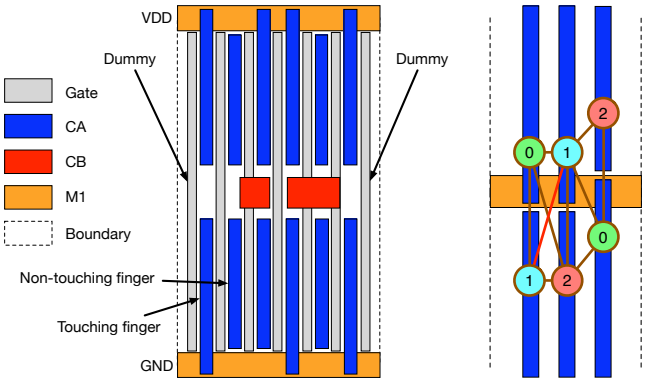

Fig. 1. An example of MOL based standard cell structure. (a) MOL consists of CA and CB layers. (b) If placement is not carefully designed, CA layer may involve lots of TPL cross-row conflicts.

Since the $20nm$ technology node and beyond, to overcome the intra-cell routability problem and improve the cell regularity, a Tungsten-based middle of line (MOL) structure is employed for standard cell design [23]. MOL structure is made up of two different local interconnection layers, namely CA and CB (also called IM1 and IM2 [24]). An example of MOL enabled standard cell structure is shown in Figure 1(a), where CA and CB layer features are labelled as blue and red, respectively. We can see from Figure 1(b), if placement is not carefully designed, on CA layer there are several native TPL conflicts cross different standard cell rows. Therefore, for advanced technology nodes where MOL layers are employed, standard cell coloring and placement should take the cross-row conflicts into account. However, all existing TPL aware placement works assume there is no conflict between cells in different rows.

In this paper, motivated by the particular structures of MOL layers, we propose a comprehensive study to TPL aware detailed placement to overcome cross-row TPL conflicts. To the best of our knowledge, this work is the first TPL aware detailed placement targeting at cross-row conflict removal. The contributions of our paper can be highlighted as follows:

- We carry out a comprehensive study on standard cell level coloring strategy and boolean based look-up table (LUT)

978-1-4673-8388-2/15/$31.00 ©2015 IEEE

construction for MOL structures.

- We propose single row placement techniques that allow local reordering for conflict and stitch minimization.
- We develop a concurrent approach for multi-row conflict removal.
- Experimental results demonstrate the effectiveness of our proposed framework.

The rest of this paper is organized as follows: Section II shows the definitions of related concepts and our overall flow. Section III and Section IV propose the cell level decomposition and TPL aware detailed placement, respectively. Section V gives the experimental results, followed by conclusion in Section VI.

II. Preliminaries and Overall Flow

A. MOL Structure

As shown in Figure 1(a), MOL layers typically include CA and CB [24]. In our work, we focus on CA layer colorability since this layer is more likely to cause TPL conflicts across different rows. In a row based layout structure, standard cells are aligned to placement rows with identical height. Horizontal power and ground (PG) rails are shared by neighboring rows. Thus neighboring rows have to be aligned in a back-to-back manner; i.e., a row orientated to N must have neighbors orientated to FS, shown as Figure 2(b), vice versa. CA features can touch the PG rails. For simplicity, we define a *touching finger* as a CA feature that touches the PG rails, and a *non-touching finger* as a CA feature that does not hit any PG rail. All fingers are aligned to specific grids since they are aligned between features of gate layer. Due to the existence of touching fingers, it is very likely to introduce conflicts between cross-row CA features. Figure 1(b) shows an example of coloring failure caused by cross-row conflicts if only three masks are available. We also observed that there is no conflict between MOL fingers within a standard cell.

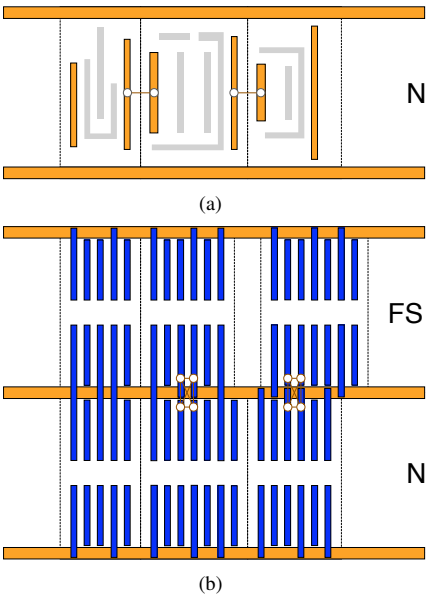

Fig. 2. Example of conflicts on (a) intra-row Metal-1 layer, (b) cross-row MOL layer.

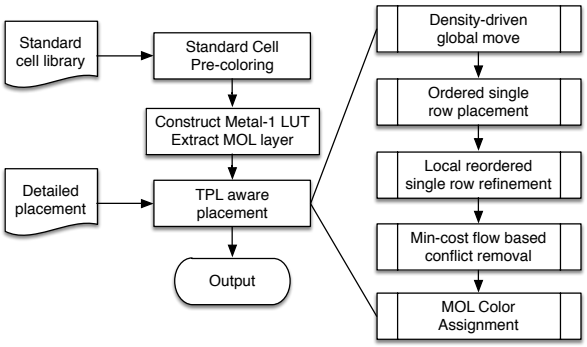

Fig. 3. Overall flow of the methodologies for TPL detailed placement.

B. Colorability Analysis for Advanced Standard Cell

In advanced standard cell design, the standard cell colorability is different from conventional one (e.g., 45nm technology node). On one hand, due to the employment of MOL layers for local connection near PG rails, there is no cross-row conflict on Metal-1 layers, as shown in Figure 2(a), and the Metal-1 layer conflict can only happen between two abutting cells in the same row. On the other hand, if cells are not carefully placed, there would be a large amount of cross-row conflicts on MOL layers. Figure 2(b) shows such conflicts.

C. Overview of Proposed Flow

The overall flow of our methodologies is shown in Figure 3. There are two main phases, standard cell phase and detailed placement phase. In the standard cell phase, given standard cell library as the input, we perform pre-coloring for metal layers in standard cells, generate LUT for Metal-1 layer and extract MOL features for next phase. In the second phase, TPL aware detailed placement is performed to optimize wirelength, assign coloring solution and minimize conflicts. This phase consists of different placement approaches with TPL constraint consideration and post placement color assignment for MOL layer. The density-driven global move is very similar to [25], so we skip it for brevity. The output of the framework is decomposed layouts with optimized placement solution and color assignment for both Metal-1 and MOL layer.

III. Standard Cell Decomposition

In triple patterning lithography, features of a certain layer are decomposed into three masks. Two features that are too close to each other cannot be assigned to the same mask; otherwise, it will introduce a conflict. For standard cells, we consider Metal-1 layer and MOL layer for triple patterning layout decomposition.

A. Standard Cell Pre-Coloring

For Metal-1 layer, the cell pre-coloring problem in a row based layout structure is very similar to [17]–[19]. Although the solution space for a whole standard cell can be very large, there is no need to enumerate all possible coloring solutions for

placement. Due to the fact that conflicts come from boundary features of cells in the TPL detailed placement problem, only boundary conditions need to be considered. In our Metal-1 layer decomposition flow, we apply the backtracking algorithm from [18] and enumerate all color combinations for boundary features. Features not belong to the boundaries are assigned with any color combination that carries out a legal coloring solution. Due to limited number of boundary features, it is applicable to store the coloring solutions as an input for detailed placement stage.

For MOL layer pre-coloring, inspired by the Metal-1 layer, we only consider coloring solutions of features that are close to cell top or cell bottom. As suggested in [26], we forbid stitching for CA layer. At first glance, the pre-coloring problems of Metal-1 layer and MOL layer are very similar. However, the difference lies in the problem size. In advanced technology node, for Metal-1 layer, there are usually fewer than five boundary features for left or right side of a standard cell, while the number of fingers for MOL layer is much larger. Usually large cells have even more fingers; e.g. a D flip-flop has more than twenty fingers at top boundary.

B. BLUT Construction

According to previous analysis, we generate pre-coloring solutions of Metal-1 layer for each cell and pre-compute the minimum colorable distance for every cell pair with different coloring solutions. The distances are stored in an LUT.

For MOL layer, the LUT is discrepant to that of Metal-1. Here is an example to explain the difference. Suppose we have cell C_i at bottom and cell C_j is on top of cell C_i. Conflicts may occur at discrete values of the position pairs. For simplicity, let the horizontal position of cell C_i be p_i. Possible situation is that conflicts occur if cell C_j is placed to position $p_i + 1$, $p_i + 2$, $p_i + 5$, $p_i + 6$, while position $p_i + 3$ and $p_i + 4$ are safe for cell C_j. Therefore, the LUT for MOL should contain a set of conflict ranges instead of a single required distance like Metal-1.

Nevertheless, as mentioned in Section II-A, such kind of approach may suffer from large problem sizes. Figure 4 gives an example of solution spaces of four particular cells. The maximum amount of candidate coloring solutions for Metal-1 comes from XOR2_X2, which is 12, but its MOL layer has more than 100000 coloring solutions. It is no longer feasible to store all the solutions in the LUT. One way is to select partial solutions, but it will increase the difficulty to resolve conflicts for fewer coloring options.

Due to the regularity of MOL features, we can solve the problem more efficiently. All possible topological patterns of four abutting fingers can be represented by the patterns listed in Figure 5 with proper flipping and rotation. Most of these patterns are TPL friendly except pattern F which results in a 4-clique structure (K4). It is impossible to resolve a K4 with only three colors. As the topological patterns depend on cross-row cell positions, it has to be avoided during placement stage. If we can produce a K4-free placement solution for MOL layer, it should not be difficult to find a legal coloring solution by post placement color assignment due to the regularity. Hence,

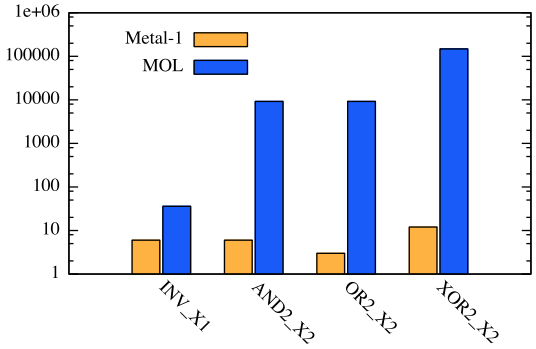

Fig. 4. Example of LUT sizes for Metal-1 and MOL layer.

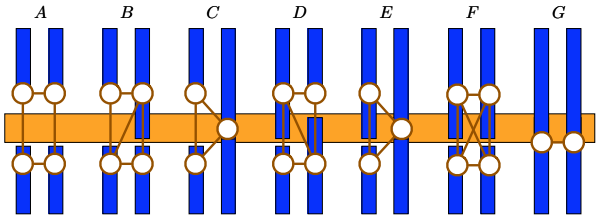

Fig. 5. Possible patterns of four abutting MOL fingers.

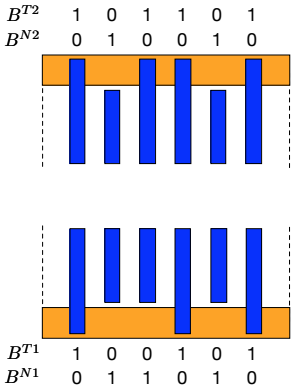

Fig. 6. Example of boolean representation for MOL fingers.

there is no need to explore the coloring solutions for MOL layer before placement.

The checking for K4 can be dynamically performed in placement instead of keeping an LUT. To further simplify the problem, we classify MOL fingers according to the vertical distances to cell boundary. Without loss of generality, we assume all touching fingers have the same distance to cell boundary, so do non-touching fingers. A boolean based LUT (BLUT) is applied to represent the existence of different finger types. For each standard cell, four bitsets are required to store the fingers, shown as Figure 6. Bottom fingers are represented by B^{T1} and B^{N1} which denote touching fingers and non-touching fingers, respectively. If no finger exists at a certain grid j, both $B^{T1}(j)$ and $B^{N1}(j)$ are set to 0. Top fingers are represented by B^{T2} and B^{N2}. The bottom and top concept here are determined when a cell is orientated to N. Similar to cells, each row needs four bitsets R^{T1}, R^{N1}, R^{T2} and R^{N2}.

The checking for K4 between two neighboring row i and row $i + 1$ is summarized in Alg. 1. As the orientation of a placement row is fixed, we assume row i is orientated to N

and row $i+1$ is orientated to FS. So bitset R_i^{T2} and R_i^{N2} of row i interact with bitset R_{i+1}^{T2} and R_{i+1}^{N2} of row $i+1$. The basic idea is to perform bitwise *and* operations for bitsets and check consecutive 1s. Same approach can be applied to check K4 between a cell and its neighboring rows. Considering that the size of bitsets for a cell is smaller than that for a row, we can truncate the row bitsets by *shifting* before performing the checking.

Algorithm 1 Row K4 Checking

Require: Bitsets R_i^{T2}, R_i^{N2}, R_{i+1}^{T2}, R_{i+1}^{N2}
Ensure: Whether there exists K4 between row i and row $i+1$
1: $A_1 = R_i^{T2} \& R_{i+1}^{N2}$;
2: $A_2 = R_i^{N2} \& R_{i+1}^{T2}$;
3: $n =$ length of A_1;
4: **for** $j = 1$ *to* n **do**
5: **if** $A_1(j-1)A_1(j)$ or $A_2(j-1)A_2(j)$ **then**
6: Return true;
7: **end if**
8: **end for**
9: Return false;

IV. TPL Aware Detailed Placement

In this section, we present our scheme of TPL aware detailed placement to remove conflicts and minimize wirelength. To maintain the properties of input placement solutions such as wirelength and routability, the maximum displacement constraint is introduced to limit the amount of movement for each cell in manhattan distance.

A. Single Row Placement Problem

Previous studies have shown that single row placement has impressive performance in simultaneous color assignment, conflict removal, and wirelength minimization for Metal-1 layer [17], [19], [21], [22]. In our scheme, the cost of single row placement problem comes from both horizontally and vertically abutting cells. Hence the information of neighboring rows should also be considered.

Problem 1. *(Single Row Placement Problem) Given a row of standard cells and its neighboring rows, determine the position, orientation, and coloring for cells to minimize cost in Eqn.* (1).

$$cost = \Delta WL + \alpha \cdot N_{ST} + \beta \cdot N_{CF} \tag{1}$$

where N_{ST} denotes the number of stitches and N_{CF} stands for the number of conflicts. Both Metal-1 conflicts and MOL K4 structures are generalized to N_{CF}. Parameter α is set to 1 and β is set to a very large value. The cells in the neighboring rows are fixed when optimizing current row. An example for our single row placement problem is shown in Figure 7. Blue rectangles stand for MOL fingers and orange rectangles represent boundary Metal-1 features. Note that orientations of cells should correspond to their rows; i.e. if the row orientation is N, then cell orientation must be N or FN; if the row orientation is FS, then cell orientation must be S or FS. Although conflicts can be resolved by inserting whitespaces between abutting cells, horizontal flipping is also very effective due to the asymmetry of boundary features. For example, by flipping cell 1, both its vertical and horizontal conflicts are removed. Flipping is even more important when whitespaces are very limited in highly congested regions.

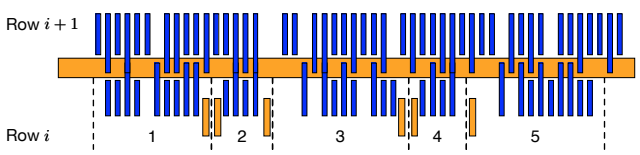

Fig. 7. A simplified example of single row placement problem.

1) Ordered Single Row Placement (OSR): Single row placement problem with arbitrary order is well known NP-hard. However, optimal solutions can be found in polynomial time if the order of the cells is fixed [27]–[30]. Previous studies on TPL aware placement have already explored different algorithms for the application of intra-row conflicts [17]–[19], [21]. Here we adopt the linear dynamic programming algorithm in [18] as it is more compatible to maximum displacement constraints. The cost function for the algorithm is extended to check cross-row K4 which will be counted into conflicts.

2) Local Reordered Single Row Refinement (LRSR): With only OSR, the performance is limited in terms of conflict removal owing to the high demands for whitespaces. If local reordering is available, larger solution space will contribute to fewer conflicts and stitches. In spite of the NP-hardness of the arbitrary order single row placement, polynomial time algorithm is available for the problem with local reordering.

Inspired by [31], we construct a graph model to handle horizontal flipping, local reordering, local shifting, and Metal-1 color assignment simultaneously. The difference of our method is that 3 vertices are introduced for each cell during local reordering to reduce number of edges. For simplicity, we explain the graph model for different techniques separately and then show the unified model. In the LRSR problem, the input has been optimized by OSR for wirelength, the main target is to further refine conflicts and stitches with small wirelength degradation. Therefore, displacement cost is applied instead of wirelength. The cost function can be re-written as follows,

$$cost = D + \alpha \cdot N_{ST} + \beta \cdot N_{CF} \tag{2}$$

where D denotes the total movement of cells.

Consider the placement row shown in Figure 7, whose the orientation is N. As only horizontal flipping is allowed within a row, a cell can be orientated to N or FN. Then for each cell, two vertices are introduced to represent its orientation, as Figure 8 shows. For instance, vertex N_1 represents that cell 1 has an orientation of N, while vertex FN_1 denotes it is flipped to FN. If a flipping solution for two abutting cells results in Metal-1 conflicts, the edge is assigned to a very large cost, such as the edge between N_1 and N_2. If a flipping solution of one cell leads to cross-row K4, all its input edges are assigned to a large cost like the edge connects s to N_1. These edges are marked in red. The problem of computing the best flipping solution is equavalent to finding the shortest path

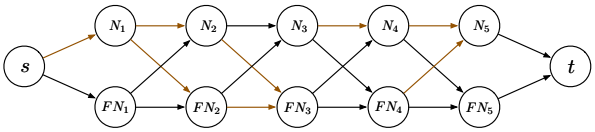

Fig. 8. Example of cell flipping graph.

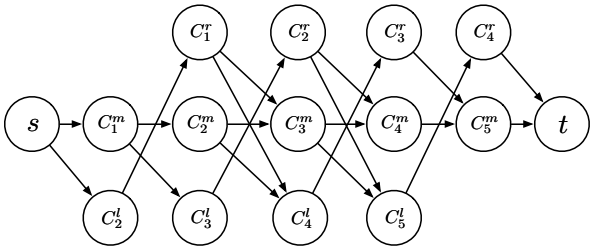

Fig. 9. Example of local cell reordering graph.

from s to t in the graph. In Figure 8, we can find a legal path $s \to FN_1 \to N_2 \to N_3 \to FN_4 \to FN_5 \to t$.

The local reordering technique enables local swap of neighboring cells within a row. It can be stated by the following definition.

Definition 1 (P-reordering)**.** *Given a row of standard cells, select p consecutive cells to switch positions between them, such that the cost defined in Eqn.*(2) *is minimized.*

Figure 9 gives an example of the reordering graph when p is equal to 2. Each cell i corresponds to three vertices C_i^l, C_i^m and C_i^r, as it can swap with its preceding cell or its succeeding cell. In Figure 9, the edge from C_1^m to C_2^m stands for the condition that cell 1 and cell 2 are kepted in original order. The edge from C_2^l to C_1^r represents the swap of cell 1 and cell 2. In this problem, if cell i has swapped with cell $i+1$, it is not allowed to further swap with cell $i+2$.

We can summarize the graph construction of 2-reordering problem for a row with N cells as follows.

1) For each cell i ($2 \leq i \leq N$), vertex C_i^l has an output edge to vertex C_{i-1}^r.
2) For each cell i ($1 \leq i \leq N-1$), vertex C_i^m has an output edge to vertex C_{i+1}^m.
3) For each cell i ($1 \leq i \leq N-2$), vertex C_i^m has an output edge to vertex C_{i+2}^l.
4) For each cell i ($1 \leq i \leq N-2$), vertex C_i^r has an output edge to vertex C_{i+2}^m.
5) For each cell i ($1 \leq i \leq N-3$), vertex C_i^r has an output edge to vertex C_{i+3}^l.

As additional source vertex s and target vertex t are introduced, extra edges $s \to C_1^m$, $s \to C_2^l$, $C_{N-1}^r \to t$ and $C_N^m \to t$ are inserted.

For further flexibility, small movement at the original position and swapped positions for a cell is allowed. Let p_i^m be the original position of cell i, p_i^l denote the position if cell i is swapped with cell $i-1$, and p_i^r denote the position swapped with cell $i+1$. Additional vertices are inserted to enable positions such as $[p_i^l - d, p_i^l + d]$, $[p_i^m - d, p_i^m + d]$ and $[p_i^r - d, p_i^r + d]$, where d is the maximum shifting value for this step. At the same time, the graph model can also determine the coloring solutions for Metal-1 layer.

The full algorithm uses a unified graph model that combines all the techniques mentioned above. In the unified graph model, each vertex has four attributes. Swapping attribute contains the swapping status (C_i^l, C_i^m or C_i^r) of current cell. Shifting attribute d_i represents the shifting amount from current or swapped position. Color attribute k_i represents the Metal-1 coloring solution and flipping attribute f_i represents the flipping status of a cell. The interconnection of the unified graph can be derived from previous cell flipping graph and local reordering graph. Edge cost is determined by all the attributes of two vertices. Unified graph model is used in the implementation to solve the LRSR problem optimally.

Let N be the total number of cells, d be the maximum shifting value for LRSR and K be the maximum number of coloring solutions. Since three types of vertices are introduced for local reordering and there are $2d+1$ candidate positions in local shifting for each vertex, the total number of vertices in the graph is $3N \cdot (2d+1) \cdot K \cdot 2$. As the graph model turns out to be a directed acyclic graph, the shortest path problem can be solved with $O(d^2K^2N)$ time complexity. In the experiment, we set d to 1 placement site, so the time complexity is $O(K^2N)$.

The local cell reordering algorithm can be extended from 2-reordering to p-reordering ($p \geq 3$), though the number of vertices in the graph will increase significantly. In the case of $p = 3$, each cell needs 8 vertices to represent all the swapping conditions. Then the total number of vertices for the whole graph will increase to $8N \cdot (2d+1) \cdot K \cdot 2$.

B. Min-Cost Flow Based Conflict Removal

While it is true that single row placement can remove most conflicts, it tends to fail in very dense regions. Movement between multiple rows is necessary to resolve all the conflicts. Conventional approaches such as global move are able to iteratively move cells out of congested regions. But due to the greedy nature, the solution may not be good from a global view. When cells compete for whitespaces, greedy based method may process the less "urgent" cells before the most "urgent" one. For example, cell i and cell j share one candidate whitespace; cell i has another candidate whitespace, while cell j does not. The desired procedure is to place cell j to its only whitespace and move cell i to the remained candidate position. It is very difficult for greedy approaches to handle such kind of situations.

Problem 2. *(Conflict Removal Problem) Given a set of cells with conflicts, move these cells to eliminate conflicts with minimum degradation of the solution quality from previous stage.*

This step aims at eliminating the Metal-1 conflicts and MOL K4 during post single row placement stage. We need to move the conflict cells to whitespaces with minimum conflicts and displacement. To solve this problem, a min-cost flow based algorithm is proposed to add concurrent characteristics by assigning conflict cells to whitespaces simultaneously.

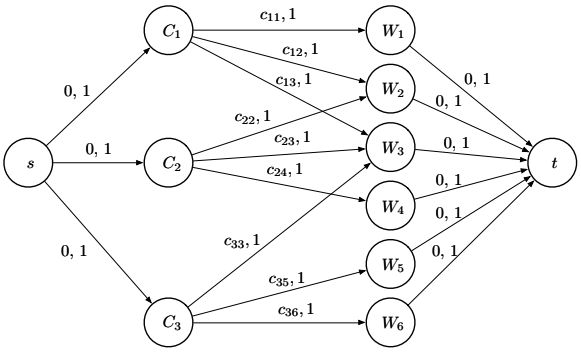

Fig. 10. Example of min-cost flow based whitespace assignment.

Figure 10 shows an example of the constructed graph for min-cost flow algorithm. Vertex C_i indicates cells and W_j denotes whitespaces. Edges are marked with (cost, capacity) pairs. The basic procedure of this algorithm is summarized as Alg. 2.

Algorithm 2 Min-cost Flow Based Conflict Removal

Require: A set of cells $C_1, C_2, ..., C_n$ and a set of whitespaces $W_1, W_2, ..., W_m$

Ensure: Assign cells to whitespaces with minimum costs

1: Detect conflicts and select candidate cells;
2: Collect whitespaces near candidate cells;
3: Construct graph G and candidate solution map S:
4: Add source vertex s and target vertex t to G;
5: Add edge $s \rightarrow C_i$ and edge $W_j \rightarrow t$ with cost 0 and capacity 1 for all i and j;
6: Add edge $C_i \rightarrow W_j$ iff C_i can be placed to W_j legally:
7: Calculate the best cost b with Eqn.(2) when C_i is placed to W_j by enumerating all combinations of flipping and coloring solutions;
8: Assign b to the edge cost and 1 to edge capacity;
9: Add the corresponding solution of b to S;
10: Solve min-cost flow for G;
11: Assign cell C_i to whitespace W_j if the edge $C_i \rightarrow W_j$ has non-zero flow and apply corresponding flipping and coloring solutions stored in S.

When selecting candidate cells, we detect *conflict chains* and group related cells; A *conflict chain* is a set of cells that form a connected component if we connect them with conflict edges. e.g. if there is conflict between cell 1 and cell 2, and also conflict between cell 2 and cell 3, then these cells are grouped together, called a conflict chain. For a conflict chain with n cells, only smallest $n-1$ cells will be selected for movement, because it is easier for small cells to fit into whitespaces. During whitespace collection, we guarantee each whitespace pair has a distance larger than coloring distance to avoid potential conflicts after cell assignment.

C. MOL Layer Color Assignment

After TPL aware placement, we obtain a placement solution without Metal-1 conflicts or MOL K4. We still need to assign color to MOL layer. Due to the regularity of MOL features, we can decompose the MOL layer in a row-by-row approach. Layout decomposition for row based layout structure has been well studied by [8], [11]. The region we need to perform coloring algorithm is not the placement row, but the interaction region between two neighboring rows, shown in the dashed region of Figure 11. According to the analysis for different types of patterns in Figure 5, we scan the MOL fingers from left to right, construct a pattern with previous finger pair (left) and current finger pair, and perform pattern matching to find the coloring solution. For instance, if a pattern matches pattern B in Figure 5, the bottom right finger should be assigned with the same color as the top left finger; then the top right finger should be assigned to last available color. The MOL finger color assignment can be solved in linear time without conflicts as long as no K4 exists.

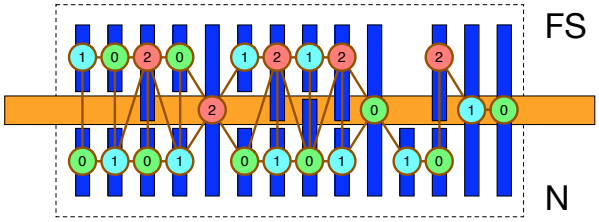

Fig. 11. Example of MOL coloring solutions.

TABLE I
BENCHMARK STATISTICS

Benchmark	cell#	net#	d_{max}	$K_{Metal-1}$
alu-70/80/85/90	2125	2524	5	12
byp-70/80/85/90	9116	10075	5	18
div-70/80/85/90	6050	6390	5	18
ecc-70/80/85/90	1286	1488	5	18
efc-70/80/85/90	1956	1964	5	48
ctl-70/80/85/90	2350	2578	5	48
top-70/80/85/90	21251	21608	5	48

V. EXPERIMENTAL RESULTS

Our algorithms were implemented in C++ and tested on an eight-core 3.40 GHz Linux server with 32 GB RAM. The min-cost flow problem is solved by the successive shortest path algorithm in Boost library [32]. We use Nangate 15nm library [33] as our initial standard cell library. Due to different standard cell libraries, we cannot directly use the benchmark from [18]. Therefore, we synthesize new placement solutions for OpenSPARC T1 designs using the Nangate 15nm library and Cadence Encounter [34]. For each design, we choose four different utilization rates, 0.7, 0.8, 0.85, and 0.9. Usually, the higher utilization rate, the harder to find legal placement and coloring solutions. Table I lists all the statistics of different test cases. Columns "cell#" and "net#" are the total number of cells and the total number of nets in each test case, respectively. Column "d_{max}" is the maximum displacement constraint for each placement. Column "$K_{Metal-1}$" is the maximum number of cell pre-coloring solutions among all standard cell types for Metal-1 layer. The value of $K_{Metal-1}$ is related to the look-up table size.

During standard cell pre-coloring, the coloring distance for both Metal-1 and MOL layers is set to 80. For Metal-1 layer

the upper bound of coloring solutions is set to 50, while for MOL layer the upper bound is set to 10. During standard cell placement, since Nangate 15nm insert dummy poly on the cell boundaries, there is no Metal-1 conflict in a single row. Although such design can effectively provide legal single row coloring solution, the inserted dummy pitch would cause additional area penalty, as shown in Figure 1(a). To better compare the performances of different placement techniques, we increase the coloring distance on Metal-1 for neighboring cells to 110.

Table II analyzes the performances of the proposed BLUT construction (see Section III-B) and different detailed placement algorithms (see Section IV). Here we compare three different design methodologies, as listed in three columns. Column "**LUT+OSR**" is based on conventional look-up table construction and ordered single row placement. Here we implement the linear dynamic programming algorithm in [18] to search for optimal single row placement solution. Column "**BLUT+OSR**" is extended from "LUT+OSR" that instead of conventional LUT construction, the BLUT introduced in Section III-B is applied here. Column "**BLUT+All**" employs BLUT and all the TPL aware detailed placement proposed in Section IV. For each design methodology, we list four different metrics: "ΔWL", "CF", "ST", and "Time(s)". "ΔWL" measures the total wirelength change after optimization. "CF" gives the total conflict number on Metal-1 and MOL layer. "ST" gives the stitch number only on Metal-1 layer as stitching is not allowed on MOL layer. "Time(s)" lists the process runtime in seconds.

We first compare methodologies "LUT+OSR" and "BLUT+OSR", and both of them utilize the ordered single row placement in [18]. The difference is that the former one applies the conventional look-up table, while the latter one uses the BLUT as in Section III-B. We can see that, on average "LUT+OSR" introduces more than 5000 conflicts, while "BLUT+OSR" only reports less than 10 conflicts. The reason is that under MOL based structure, conventional look-up table may contain too many pre-coloring combinations. To control the LUT size, some pre-coloring solutions are pre-deleted in cell library. Therefore, LUT based method is not able to fix most of the conflicts. Compared with conventional LUT, the proposed BLUT enables larger solution space, and thus is more powerful to fix cross-row conflicts.

We further compare "BLUT+OSR" and "BLUT+All", where "BLUT+All" integrates all the detailed placement techniques developed in this paper. From Table II we can see that both of them can improve wirelength by around 1%. Compared with "BLUT+OSR", through some powerful techniques, such as local reordered single row refinement (LRSR) and min-cost flow based conflict removal, "BLUT+All" can resolve all the TPL conflicts. It shall be noted that the runtime penalty of "BLUT+All" is not significant, i.e., "BLUT+All" only introduces less than 11% of runtime overhead against "BLUT+OSR". The reason is that for each single row, OSR based method is applied first. Only if there exist remaining conflicts, more expensive LRSR and min-cost flow based methods are carried out.

VI. Conclusion

Motivated by a large amount of cross-row TPL conflicts from middle-of-line (MOL) layers, in this paper we have proposed a comprehensive study to TPL aware detailed placement toward zero cross-row conflict. Several effective techniques, such as BLUT construction, local reordered single row refinement (LRSR) and min-cost flow based conflict removal, are proposed. Our framework is verified through a set of placement test cases using 15nm standard cell library. With further scaling of transistor feature size and MOL layers, the cross-row conflicts would be an emerging problem for TPL friendly design. We believe this paper will stimulate more future research on TPL aware standard cell design and physical design.

Acknowledgment

This work is supported in part by NSF and SRC. The authors would like to thank Dr. Yong-Chan Ban at LG Electronics for helpful comments.

References

[1] L. Liebmann, A. Chu, and P. Gutwin, "The daunting complexity of scaling to 7nm without EUV: Pushing DTCO to the extreme," in *Proceedings of SPIE*, vol. 9427, 2015.

[2] B. J. Lin, "Optical lithography with and without NGL for single-digit nanometer nodes," in *Proceedings of SPIE*, vol. 9426, 2015.

[3] D. Z. Pan, B. Yu, and J.-R. Gao, "Design for manufacturing with emerging nanolithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 32, no. 10, pp. 1453–1472, 2013.

[4] L. Liebmann, V. Gerousis, P. Gutwin, M. Zhang, G. Han, and B. Cline, "Demonstrating production quality multiple exposure patterning aware routing for the 10nm node," in *Proceedings of SPIE*, 2014.

[5] K. Lucas, C. Cork, B. Yu, G. Luk-Pat, B. Painter, and D. Z. Pan, "Implications of triple patterning for 14 nm node design and patterning," in *Proceedings of SPIE*, vol. 8327, 2012.

[6] B. Yu, K. Yuan, B. Zhang, D. Ding, and D. Z. Pan, "Layout decomposition for triple patterning lithography," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2011, pp. 1–8.

[7] S.-Y. Fang, Y.-W. Chang, and W.-Y. Chen, "A novel layout decomposition algorithm for triple patterning lithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 33, no. 3, pp. 397–408, March 2014.

[8] H. Tian, H. Zhang, Q. Ma, Z. Xiao, and M. D. F. Wong, "A polynomial time triple patterning algorithm for cell based row-structure layout," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012, pp. 57–64.

[9] J. Kuang and E. F. Young, "An efficient layout decomposition approach for triple patterning lithography," in *ACM/IEEE Design Automation Conference (DAC)*, 2013, pp. 69:1–69:6.

[10] Y. Zhang, W.-S. Luk, H. Zhou, C. Yan, and X. Zeng, "Layout decomposition with pairwise coloring for multiple patterning lithography," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013, pp. 170–177.

[11] H.-A. Chien, S.-Y. Han, Y.-H. Chen, and T.-C. Wang, "A cell-based row-structure layout decomposer for triple patterning lithography," in *ACM International Symposium on Physical Design (ISPD)*, 2015, pp. 67–74.

[12] B. Yu, K. Yuan, D. Ding, and D. Z. Pan, "Layout decomposition for triple patterning lithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 3, pp. 433–446, March 2015.

[13] Q. Ma, H. Zhang, and M. D. F. Wong, "Triple patterning aware routing and its comparison with double patterning aware routing in 14nm technology," in *ACM/IEEE Design Automation Conference (DAC)*, 2012, pp. 591–596.

[14] Y.-H. Lin, B. Yu, D. Z. Pan, and Y.-L. Li, "TRIAD: A triple patterning lithography aware detailed router," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012, pp. 123–129.

TABLE II
PERFORMANCE EVALUATION OF LUT CONSTRUCTION AND DETAILED PLACEMENT ALGORITHMS

Benchmark	LUT+OSR				BLUT+OSR				BLUT+All			
	ΔWL	CF	ST	Time(s)	ΔWL	CF	ST	Time(s)	ΔWL	CF	ST	Time(s)
alu-70	-1.53%	1778	75	1007.65	-3.71%	2	72	3.68	-3.66%	0	72	3.96
alu-80	-1.46%	2277	83	895.27	-2.91%	11	86	3.44	-2.36%	0	87	4.48
alu-85	-0.86%	2516	85	828.55	-2.19%	20	83	3.19	-1.65%	0	89	4.32
alu-90	0.13%	2823	87	767.20	-0.69%	30	90	3.10	-0.09%	0	89	4.08
byp-70	-0.78%	6335	806	4453.00	-2.09%	0	791	16.41	-2.09%	0	791	16.62
byp-80	-0.39%	8485	852	4058.07	-1.34%	2	853	15.50	-1.32%	0	854	16.00
byp-85	-0.39%	9736	931	3971.18	-1.39%	10	918	15.45	-1.26%	0	918	16.96
byp-90	-0.14%	10965	1010	3676.96	-0.96%	29	992	14.61	-0.79%	0	1000	18.12
div-70	-1.89%	4857	589	2807.06	-3.47%	7	583	10.40	-3.37%	0	583	11.56
div-80	-1.00%	6237	620	2570.82	-2.52%	12	605	9.98	-2.35%	0	604	11.25
div-85	-1.06%	6879	635	2474.66	-2.34%	24	631	9.85	-1.95%	0	631	12.09
div-90	-0.40%	7638	686	2353.77	-1.66%	22	676	9.38	-1.46%	0	673	11.33
ecc-70	-2.51%	1078	253	626.37	-4.45%	0	248	2.46	-4.45%	0	248	2.46
ecc-80	-1.39%	1382	270	565.63	-3.37%	1	262	2.27	-3.27%	0	262	2.48
ecc-85	-1.35%	1508	278	509.79	-2.82%	2	272	2.09	-2.22%	0	272	2.33
ecc-90	-0.72%	1835	294	384.69	-1.33%	23	295	1.79	-0.27%	0	303	2.25
efc-70	-3.96%	1641	235	1048.91	-6.51%	0	239	3.86	-6.51%	0	239	3.84
efc-80	-3.16%	2119	253	944.11	-4.98%	0	246	3.82	-4.98%	0	246	3.84
efc-85	-1.57%	2349	258	840.85	-3.47%	3	253	3.63	-3.32%	0	253	3.92
efc-90	0.09%	2534	261	832.88	-1.51%	23	264	3.68	-0.79%	0	267	4.64
ctl-70	-3.22%	2158	437	1243.38	-4.57%	1	439	5.04	-4.50%	0	439	5.08
ctl-80	-2.15%	2535	444	1139.64	-3.52%	2	447	4.82	-3.42%	0	447	5.11
ctl-85	-1.83%	2870	452	1046.36	-2.83%	10	458	4.60	-2.67%	0	458	5.32
ctl-90	-0.87%	3196	474	978.42	-1.68%	13	475	4.40	-1.34%	0	478	5.52
top-70	-0.94%	14435	1157	12402.91	-2.26%	8	1126	38.29	-2.23%	0	1125	41.17
top-80	-0.49%	19295	1201	11269.44	-1.63%	6	1181	36.40	-1.58%	0	1185	39.08
top-85	-0.17%	21870	1212	11057.75	-1.33%	21	1213	35.53	-1.08%	0	1213	40.13
top-90	0.20%	24634	1257	10382.27	-0.83%	23	1270	34.40	-0.73%	0	1272	40.86
avg.	-1.21%	**6284**	542	3040.63	-2.58%	**10**	538	10.79	-2.35%	**0**	539	12.10
ratio	-0.51	-	1.01	251.29	-1.08	-	1.00	0.89	-1.00	-	1.00	1.00

[15] P.-Y. Hsu and Y.-W. Chang, "Non-stitch triple patterning-aware routing based on conflict graph pre-coloring," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2015, pp. 390–395.

[16] Z. Liu, C. Liu, and E. F. Young, "An effective triple patterning aware grid-based detailed routing approach," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2015, pp. 1641–1646.

[17] B. Yu, X. Xu, J.-R. Gao, and D. Z. Pan, "Methodology for standard cell compliance and detailed placement for triple patterning lithography," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013, pp. 349–356.

[18] B. Yu, X. Xu, J.-R. Gao, Y. Lin, Z. Li, C. Alpert, and D. Z. Pan, "Methodology for standard cell compliance and detailed placement for triple patterning lithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 5, pp. 726–739, May 2015.

[19] J. Kuang, W.-K. Chow, and E. F. Y. Young, "Triple patterning lithography aware optimization for standard cell based design," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 108–115.

[20] H.-A. Chien, Y.-H. Chen, S.-Y. Han, H.-Y. Lai, and T.-C. Wang, "On refining row-based detailed placement for triple patterning lithography," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 34, no. 5, pp. 778–793, 2015.

[21] H. Tian, Y. Du, H. Zhang, Z. Xiao, and M. D. F. Wong, "Triple patterning aware detailed placement with constrained pattern assignment," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 116–123.

[22] T. Lin and C. Chu, "TPL-aware displacement-driven detailed placement refinement with coloring constraints," in *ACM International Symposium on Physical Design (ISPD)*, 2015, pp. 75–80.

[23] T. Kauerauf, A. Branka, G. Sorrentino, P. Roussel, S. Demuynck, K. Croes, K. Mercha, J. Bommels, Z. Tokei, and G. Groeseneken, "Reliability of MOL local interconnects," in *IEEE International Reliability Physics Symposium (IRPS)*, 2013, pp. 2F–5.

[24] M. Rashed, N. Jain, J. Kim, M. Tarabbia, I. Rahim, S. Ahmed, J. Kim, I. Lin, S. Chan, H. Yoshida, S. Beasor, L. Yuan, J. Kye, J. Chee, A. Mittal, D. Doman, S. Johnson, U. Schroeder, N. Cave, T. Tang, J. Stephen, R. Augur, S. Kengeri, and S. Venkatesan, "Innovations in special constructs for standard cell libraries in sub 28nm technologies," in *IEEE International Electron Devices Meeting (IEDM)*, 2013, pp. 9.7.1–9.7.4.

[25] S. Popovych, H.-H. Lai, C.-M. Wang, Y.-L. Li, W.-H. Liu, and T.-C. Wang, "Density-aware detailed placement with instant legalization," in *ACM/IEEE Design Automation Conference (DAC)*, 2014, pp. 122:1–122:6.

[26] X. Xu, B. Cline, G. Yeric, B. Yu, and D. Z. Pan, "A systematic framework for evaluating cell level middle-of-line (MOL) robustness for multiple patterning," in *Proceedings of SPIE*, vol. 9427, 2015.

[27] A. B. Kahng, S. Reda, and Q. Wang, "Architecture and details of a high quality, large-scale analytical placer," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2005, pp. 891–898.

[28] J. Vygen, "Algorithms for detailed placement of standard cells," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 1998, pp. 321–324.

[29] A. B. Kahng, P. Tucker, and A. Zelikovsky, "Optimization of linear placements for wirelength minimization with free sites," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 1999, pp. 241–244.

[30] U. Brenner and J. Vygen, "Faster optimal single-row placement with fixed ordering," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2000, pp. 117–121.

[31] Y. Du and M. D. F. Wong, "Optimization of standard cell based detailed placement for 16 nm FinFET process," in *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, 2014, pp. 357:1–357:6.

[32] "Boost C++ Library," http://www.boost.org.

[33] "NanGate FreePDK15 Open Cell Library," http://www.nangate.com/?page_id=2328.

[34] "Cadence SOC Encounter," http://www.cadence.com.