Timing Yield-Aware Color Reassignment and Detailed Placement Perturbation for Double Patterning Lithography

Mohit Gupta[†], Kwangok Jeong[†], and Andrew B. Kahng[†‡]
[†]CSE and [‡]ECE Departments, UC San Diego, La Jolla, CA
mohit@vlsicad.ucsd.edu, kjeong@vlsicad.ucsd.edu, abk@cs.ucsd.edu

ABSTRACT

Double patterning lithography (DPL) is a likely resolution enhancement technique for IC production in $32nm$ and below technology nodes. However, DPL gives rise to two independent, uncorrelated distributions of linewidth on a chip, resulting in a 'bimodal' linewidth distribution and an increase in performance variation. [13] suggested that new physical design mechanisms could reduce harmful covariance terms that contribute to this performance variation.

In this paper, we propose new bimodal-aware timing analysis and optimization methods to improve timing yield of standard-cell based designs that are manufactured using DPL. Our first contribution is a DPL-aware approach to timing modeling, based on detailed analysis of cell layouts. Our second contribution is an ILP-based maximization of 'alternate' mask coloring of instances in timing-critical paths, to minimize harmful covariance and performance variation. Third, we propose a dynamic programming-based detailed placement algorithm that solves mask coloring conflicts and can be used to ensure "double patterning correctness" after placement or even after detailed routing, while minimizing the displacement of timing-critical cells with manageable ECO impact.

With a 45nm library and open-source design testcases, our timing-aware recoloring and placement optimizations together achieve up to $232ps$ (resp. $36.22ns$) reduction in worst (resp. total) negative slack, and 78% (resp. 65%) reduction in worst (resp. total) negative slack variation.

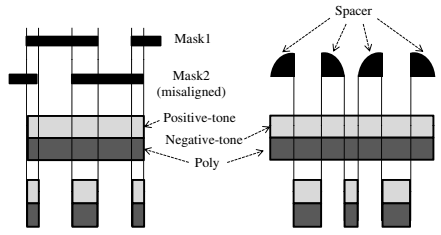
1. INTRODUCTION

Double patterning lithography (DPL) has been gaining attention as a relatively high-throughput and low-cost lithography technique for deep submicron technologies. Next-generation patterning techniques, such as $157nm$, extreme ultraviolet (EUV), nanoimprint, e-beam direct write, etc., still face high cost, difficulty of materials and processes, or prohibitively low throughputs. Hence, double patterning with traditional lithography tools using $193nm$ is regarded as a highly promising technique for $32nm$ and $22nm$ technologies.

DPL effectively doubles the achievable pattern density by using multiple exposure and etch steps or by using an additional spacer formation followed by multiple etch steps per layer. Double exposure [1], double patterning [2] [7] and sacrificial spacer double patterning [11] are three major types of double patterning in the International Technology Roadmap for Semiconductors (ITRS) [18], and with variants to enhance printability and reduce variability [3] [12].

To use double patterning, patterns in the same layer need to be decomposed into two groups, which are implemented independently [5] [15]. The pattern decomposition problem for double patterning is similar to a 2-colorability problem. Layout patterns are converted to a graph, in which a vertex represents an individual pattern and an edge represents a coloring conflict between vertices that are separated by less than the given technology resolution distance; colors are then assigned to vertices while minimizing the coloring conflicts. Routing methods considering double patterning coloring decomposition are discussed in [4].

Even with a layout decomposition and coloring solution that guarantees printability of a given design, other process variations, such as overlay or CD error, hamper easy adoption of double patterning. Overlay error can cause catastrophic open faults when a polygon pattern is split and its pieces are assigned to different masks. Extensions for overlay error-tolerance are required at the interface between split patterns. Using a stitching method [16], incomplete connectivity problems can be avoided, but overlay-induced width and space variation increases on-chip variation. Notably, overlay error in double exposure/patterning with negative-tone resist for trench-first BEOL process, or in spacer double patterning with positive-tone resist, results in large critical dimension (CD) errors [18]. For the most critical front-end-of-line (FEOL) layer, i.e., the poly layer, double exposure/patterning with positive-tone resist and spacer double patterning with negative-tone resist cause large CD error as shown in Figure 1.



(a) Double exposure/double patterning with positive-tone photoresist (b) Spacer double patterning with negative-tone photoresist

Figure 1: Abbreviated representation of DPL processes. CD variations are caused by overlay error in (a) and by spacer thickness variation in (b).

A number of works address the impact of overlay error in double patterning. For BEOL, various sources of overlay error are discussed and an analytical capacitance model that comprehends overlay error is proposed in [17] [8]. However, overlay error in BEOL may not be a critical limiting factor for double patterning since the impact of overlay is comparable with that of other traditional variation sources [14]. However, in FEOL, CD variation is the most critical issue, hence we expect double patterning solutions that result in large CD variation due to overlay will not be deployable for the poly layer.

Excluding double patterning solutions causing large CD variation from overlay error, there still exist CD variations between the two poly lines that are printed in different steps of double patterning, due to the interference between successive exposures, non-uniform topography, imperfect etch biasing, etc. Dusa et al. [6] observe ∼8% mean CD difference between two poly groups in a double patterning process, with the two CD populations being uncorrelated to each other.

ICCAD'09, November 2–5, 2009, San Jose, California, USA.

The work of [13] points out new '*bimodal*' performance analysis and design optimization challenges arising from the two CD populations in double patterning. The existence of the two independent CD populations within a die takes away the presumptions of spatial correlation that have always been used to reduce pessimism in timing closure; this causes unpredictable clock skew and timing slack problems. The authors of [13] analyze delay variation of a timing path consisting of cell instances that can have differing delay variations due to the bimodal CD distribution. The delay variation of the timing path is calculated as the sum of variances of the delay value of each cell and the covariances between the delay values of different cells. Since the covariance terms are small between the uncorrelated cells, the delay variation for a path composed of uncorrelated different types of cells is smaller than that of a path composed of only correlated cells, as shown in Figure 3. To exploit such an implication, a 'self-compensation'-like technique (cf. [9]) – in the sense of deliberate balancing of cell coloring in timing paths – has been suggested to maximize the use of uncorrelated cells in a timing path. Separately, using the same type of coloring for all cells in a given clock network can help reduce the clock skew variation, which exists even if the CD mean difference between the two populations is $0nm$.

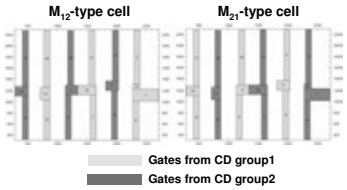


Figure 2: Two different colorings for a NOR3 cell [13].

Intentional timing-aware coloring manipulation can cause additional coloring conflicts between adjacent cells. This can be solved by recoloring of conflicting cells or shifting cells with the help of whitespaces [13] so as to not cause coloring conflicts. A problem is that placement perturbation at the post-route stage or at near the signoff stage can always introduce new timing violations due to the changes in parasitic RC from ECO routing, unless the placement and routing changes are very localized. In [10], assist-feature correctness (AF-Corr) is achieved by dynamic programming (DP) that minimizes displacement of cells while ensuring space for correct assist-feature insertion.

In this paper, we exploit the property of the 'self-compensation' along with a post-placement mask coloring and layout correction flow, to reduce the timing variation in double patterning. The main contributions of our work are summarized as follows.

- We investigate the mechanism of the delay variation due to bimodal CD distribution, and develop a new metric – *coloring sequence cost* – to capture timing variation in double patterning.
- We present a timing-aware optimal color assignment technique – *alternate color assignment* – using integer linear programming (ILP).
- We implement a dynamic programming-based detailed placement to ensure the printability of double patterning – *double patterning lithography correctness (DPL-Corr)* – utilizing whitespace management as in AF-Corr [10], after optimizing coloring using the alternate coloring assignment.
- For our testcases in $45nm$ technology, we achieve up to $232ps$ (resp. $36.22ns$) reduction in worst (resp. total) negative slack reduction, and delay variation reduction from $380ps$ (resp. $64ns$) to $84ps$ (resp. $22ns$) in terms of the worst (resp. total) negative slack.

The remainder of this paper is organized as follows. In Section 2, we present an optimal alternate color assignment method for the timing-critical paths. In Section 3, we describe our two DP algorithms for coloring conflict removal. Section 3.1 gives a basic DP algorithm **SHIFT** and Section 3.2 proposes an enhanced DP algorithm **SHIFT+RECOLOR**. Section 4 discusses the experimental flow and results. We evaluate both alternate color assignment and DPL-Corr techniques separately and together. Finally, Section 5 gives conclusions.

2. ALTERNATE COLORING OF TIMING CRITICAL PATHS

2.1 Preliminaries

Pattern decomposition in double patterning can be performed at *design-level* or at *cell-level*. We use the term 'coloring' when referring to the pattern decomposition and assignment of patterns to masks. In this paper, we deal with the coloring of the poly layer, which has the greatest impact on the design timing.

Design-level coloring is 'flat': the color of each pattern is decided considering colors of the surrounding patterns, so that instances of a master cell can be colored in a number of ways according to the different surrounding colors and distances from them. In addition, for each colored instance of a given master cell, there exist two different timing models, since we cannot presume whether poly lines in one color will have larger CD or smaller CD than poly lines in the other color. For accurate timing analysis, each differently-colored master cell must be characterized using SPICE simulations, and each cell instance must refer to the timing model defined for each colored version and each CD value. Hence the number of timing models for a given master cell will be the number of different colorings multiplied by two CD values. This explosion of library complexity increases the complexity of sizing optimization, which is essentially a problem of selection from within a predefined cell library.

Cell-level coloring, on the other hand, is 'hierarchical': there is an optimal pattern decomposition for a given master cell, and each master can have only two timing models (i.e., only two differently-colored possible instantiations) corresponding to the two possible (extreme) CD values. We assume cell-based coloring in the rest of this paper to have realistic complexity. Figure 2, reproduced from [13], shows the two different cell-based coloring implementations of a NOR3 gate.

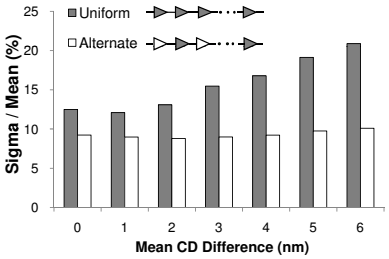


Figure 3: Relative delay variation σ/μ (%) over all process corners.

A timing path that consists of only one type of cells, e.g., M_{12} in Figure 2, and the other timing path that consists of only the other type of cells, e.g., M_{21} in Figure 2, can have very different timing delay due to the bimodal CD distribution, since the delay difference between M_{12} and M_{21} of each stage is accumulated. In the corner-based timing signoff methodology, we take the worst delay among the two cases, which increases the signoff difficulty. Figure 3 shows that the delay variation of a path comprised of differently colored cells ("in balance") is smaller than the delay variation of a path comprised of only one type of cell. This is due to the smaller covariance terms from the uncorrelated delay variation between differently colored cells.

Given the above assumptions and implications, in the following subsections we propose an optimal color assignment of cells that leads to less pessimism as well as smaller timing variation.

2.2 A New Metric: Coloring Sequence Cost

We begin by quantifying "balance" in the coloring of timing paths. The impact of DPL's bimodal CD distribution on cell delay varies according to the number of poly lines in a cell, the topology of the circuit, the assigned color for each poly gate, and the specific transistors that are activated during signal transitions. A path consisting of only buffers (each buffer comprising two cascaded inverters) experiences small impact from the bimodal CD distribution, since the CD change of the first inverter can be compensated by that of the second inverter. On the other hand, a path consisting of only one-stage inverters can have two different worst-case delay values when all inverters are assigned the same color: the inverters will have either all positive CD changes or all negative CD changes, so that there is

no compensation. For cells that are more complex than an inverter or buffer, the impact on delay of bimodal CD distribution is complicated. Table 1 shows SPICE simulation results on the NAND2 shown in Figure 4(a), according to transitive inputs (A1 and A2), switching direction (rise and fall), and bimodal CD variation. Comparing the second and fifth rows, we observe that the CD of A2 has negligible impact on the rise delay due to the transition of A1 ($rise_{A1}$). Similarly, the CD of A1 does not affect rise delay due to the transition of A2 ($rise_{A2}$). We can conclude that rise delay of the NAND2 depends only on the CD of transitive PMOS.

However, both fall delays - due to the transition of A1 ($fall_{A1}$) or of A2 ($fall_{A2}$) - are affected by both CD values of A1 and A2. For series transistors MN1 and MN2, coloring of MN2 affects cell delay triggered by MN1, and vice versa. We observe that fall delay values in the fifth and sixth rows are in between the values in the second and third rows, which are the slowest and fastest delays when both MN1 and MN2 have smaller or larger CD. This suggests that average CD change of MN1 and MN2 can be used to represent fall delay change.[1]

Table 1: Delay changes due to the CD changes of the transitive input gates of NAND2 with input poly lines corresponding to two input pins A1 and A2.

A1 CD (nm)	A2 CD (nm)	$fall_{A1}$ (sec)	$rise_{A1}$ (sec)	$fall_{A2}$ (sec)	$rise_{A2}$ (sec)
51	51	4.979e-11	9.730e-11	5.465e-11	1.131e-10
49	49	4.823e-11	8.825e-11	5.148e-11	1.021e-10
50	50	4.830e-11	9.290e-11	5.308e-11	1.076e-10
51	49	4.905e-11	9.726e-11	5.232e-11	1.022e-10
49	51	4.889e-11	8.828e-11	5.379e-11	1.130e-10

Coloring sequence cost (CSC) for a timing arc. To account for the different impact of bimodal CD changes on the cell delay, we define a *coloring sequence cost* (CSC) that scores how poly lines are colored alternately from input to output, i.e., we use CSC as a quantitative measure of the coloring balance of timing paths. The smaller the CSC, the more color alternation is in the signal propagation path in a cell, which implies smaller delay variation due to the bimodal CD distribution. For a single NMOS or PMOS device, we assign CSC value of either 1 or -1 according to the color of the transistor. CSC for a network of transistors is calculated as follows.

- Parallel transistors: 1 or -1 of the transitive input poly
- Series transistors: average CSC of all transistors
- Fingered transistors: average CSC of all transistors
- Cascaded transistors: CSC of each stage is added up

Based on the above rules, examples of the CSC calculation for BUF, NAND2, and AND2 cells are shown below. We calculate CSC for each timing arc for each coloring version of a master cell. For example, $CSC_{C_{12},rise_{A1}}$ denotes the coloring sequence cost for rise delay due to a transitive input A of a coloring version C_{12}. We use '1' and '-1' for the CSC of the transistors formed by black and white poly lines in Figure 4, respectively.

• NAND2: There are two poly lines and one logic stage as shown in Figure 4(c).

- $CSC_{C_{12},rise_{A1}}$ (= MP1: on) = 1
- $CSC_{C_{12},fall_{A1}}$ (= MN1, MN2: on) = (1+(-1))/2 = 0
- $CSC_{C_{12},rise_{A2}}$ (= MP2: on) = -1
- $CSC_{C_{12},fall_{A2}}$ (= MN1, MN2: on) = (1+(-1))/2 = 0
- $CSC_{C_{21},rise_{A1}}$ (= MP1: on) = -1
- $CSC_{C_{21},fall_{A1}}$ (= MN1, MN2: on) = (-1+1)/2 = 0
- $CSC_{C_{21},rise_{A2}}$ (= MP2: on) = 1
- $CSC_{C_{21},fall_{A2}}$ (= MN1, MN2: on) = (-1+1)/2 = 0

[1]Using the average of MN1 and MN2 may not be accurate, since delay impacts of MN1 and MN2 are different due to different charge-sharing effects for MN1 and MN2. We can extend the methods presented here to use accurate delay values via cell characterization with different CD combinations.

• BUF: There are two poly lines which are cascaded (INV followed by INV) as shown in Figure 4(b).

- $CSC_{C_{12},rise_A}$ (= MN1: on, MP2: on) = 1 + (-1) = 0
- $CSC_{C_{12},fall_A}$ (= MP1: on, MN1: on) = 1 + (-1) = 0
- $CSC_{C_{21},rise_A}$ (= MN1: on, MP2: on) = -1 + 1 = 0
- $CSC_{C_{21},fall_A}$ (= MP1: on, MN1: on) = -1 + 1 = 0

• AND2: AND2 consists of a NAND2 and an INV as shown in Figure 4(c). CSCs of NAND2 and INV are added. We show only example CSC calculations for the rise and fall delay of the C_{12} cell by A1.

- $CSC_{C_{12},rise_{A1}}$ (= MN1, MN2: on + MP3: on) = (1 + (-1)) / 2 + (-1) = -1
- $CSC_{C_{12},fall_{A1}}$ (= MP1: on + MN3: on) = 1 + 1 = 2

In our experimental testbed, we analyze the schematic and layout of all cell masters used in our testcases, and calculate the CSC value for each timing arc of each coloring version.

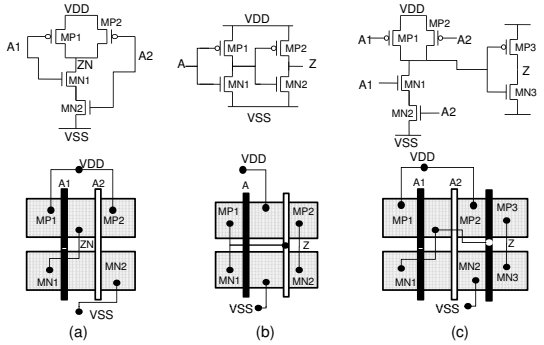


Figure 4: Schematic and layout of (a) NAND2, (b) BUF, and (c) AND2 cells.

Coloring sequence cost for a path ($CSCP$). Given the CSC values of all timing arcs, we define the coloring sequence cost of a path ($CSCP$) as a weighted sum of the CSC values of its timing arcs. Since the impact of CSC is relative to each timing arc delay, the weight is given by the delay value D_l of the timing arc l.

$$CSCP_i = \sum_{l \in i} CSC_l \cdot D_l$$

To verify the correlation between $CSCP$ and the actual delay variation, we extract a timing path from a design which has 22 stages of logic cells. We assign a color to each cell in the path randomly, then calculate $CSCP$ for each path coloring and measure the path delay using two bimodal-aware timing libraries, G1L-G2S and G1S-G2L.[2]

Figure 5 shows the correlation between the calculated $CSCP$ and the delay difference for 1,300 random colorings of the timing path. We observe that $CSCP$ and timing variation have a strong positive correlation, i.e., correlation coefficient of 0.902, and rank correlation of 0.900.

2.3 Optimal Color Assignment Problem

Due to the high correlation between $CSCP$ and delay variation, we seek to minimize delay variation by minimizing $CSCP$ of critical timing paths.

Optimal timing path coloring problem:
• Given P a set of timing-critical paths.
• Assign coloring of each cell in the timing paths so as to minimize $Max_{i \in P} \mid CSCP_i \mid$, where $CSCP_i$ is the coloring sequence cost of path i.

[2]Suppose each group of poly lines in a cell has CD value either CD1 or CD2, according to the color of the poly. G1L-G2S (G1S-G2L) represents the case that CD1 (CD2) is larger than CD2 (CD1). Section 4.1 discusses the bimodal-aware timing libraries in more detail.

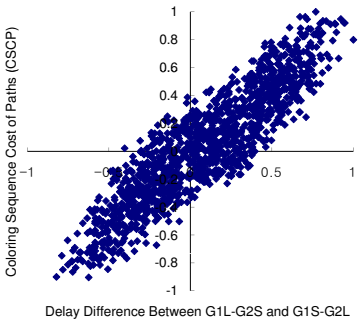


Figure 5: Correlation between $CSCP$ and the delay difference between the two bimodal-aware timing libraries G1L-G2S and G1S-G2L, for 1,300 different colorings of the timing path.

For the top-k timing-critical paths in a design, this can be formulated as an integer linear program (ILP) using an indicator variable M for the maximum magnitude of any $CSCP$ and binary variables x_j and y_j to capture the color of a cell j.

Minimization of maximum $CSCP$:

- **Objective:**
  Minimize M
- **Subject to:**

$$\begin{aligned} M &\geq CSCP_i, && 1 \leq i \leq k \\ M &\geq -CSCP_i, && 1 \leq i \leq k \\ CSCP_i &= && \textstyle\sum_{l \in i} CSC_{C_{12},l}(j) \cdot x_j + CSC_{C_{21},l}(j) \cdot y_j \\ x_j + y_j &= 1 \\ x_j &\in \{0,1\} \\ y_j &\in \{0,1\} \end{aligned}$$

where P is a set of k timing-critical paths, and a path i is a set of timing arcs. $CSC_{C_{12},l}(j)$ and $CSC_{C_{21},l}(j)$ are the two different CSC values of a timing arc l of a cell j with respect to the two coloring versions of the cell, C_{12} and C_{21}, respectively.

An ILP solver (*ILOG CPLEX v10.110* [22]) is used to solve this problem, and returns the optimal color values for cells in the top-k timing paths. As shown in Table 4, ILP runtimes are reasonable and scale well – e.g., 2.13 *sec* for 500 critical timing paths.

3. COLORING CONFLICT REMOVAL: DPL-CORR

The double patterning process dictates a technology-specific *minimum resolution spacing* to be maintained between poly lines of adjacent cells in a standard-cell row. For correct printability, the closest poly lines belonging to adjacent cells should either have different colors, such that they can be printed in separate exposures, or be separated by the minimum resolution spacing Res_{min} of the single exposure system for the given process technology. Our timing optimization by alternate color assignment within timing-critical paths, presented in Section 2, can introduce coloring conflicts between neighbors in a row for which colors have already been determined.

Given a cell coloring solution that maximizes the alternate coloring of timing-critical paths, we solve resulting coloring conflicts by detailed placement perturbation within available whitespace using a dynamic programming (DP) formulation [10]. Our **SHIFT** approach is aware of timing-critical cells and seeks to minimize perturbation of these cells to preserve timing goals of the design.

We note that placement perturbation, as a post-processing step after cell coloring, may not converge to a complete conflict removal in designs with high utilization of layout area. To counter this, we introduce **SHIFT+RECOLOR**, a recoloring approach, which recolors the cells during whitespace optimization to remove additional coloring conflicts. We make sure that this recoloring introduces minimum change to the colors of the cells in the alternate color assignment solution obtained in Section 2. We describe these dynamic programming formulations in the rest of this section.

3.1 Dynamic Programming Formulation for DPL-Corr (SHIFT)

We use the following notation.

- L_j^{PS} (R_j^{PS}) respectively denote the space between the leftmost poly (rightmost poly) and the cell outline of a cell j.
- L_j^{PC} (R_j^{PC}) respectively represent the color of the leftmost poly (rightmost poly) for a cell j.
- x_j denotes the left x-coordinate of a cell j.
- w_j represents the width of a cell j.
- δ_j denotes the displacement of a cell j from its original left x-coordinate.
- The sites in a standard-cell row are indexed from left to right. A cell occupies multiple placement sites in a standard-cell row. s_j denotes the leftmost placement site index for cell j.

Figure 6 illustrates the above notations for two adjacent cells a and $a-1$ in a standard-cell row. We consider only the boundary poly lines in the cells (i.e., those with external x-coordinates), as the internal poly lines in a cell are assumed to have been colored alternately and therefore do not have a bearing on the neighboring cells.

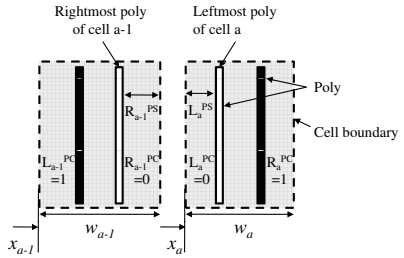


Figure 6: Variables used in the DP problem formulation.

Cells can be shifted in multiples of the placement site width, which is the finest positional granularity in the standard-cell row structure. For a given cell a, we formulate the minimum perturbation placement problem for removing coloring conflicts as follows.

- **Objective:**

$$\text{Minimize} \sum \mid \delta_i \mid,$$

- **Subject to:**

$$x_a + \delta_a - x_{a-1} - \delta_{a-1} - w_{a-1} + L_a^{PS} + R_{a-1}^{PS} \geq Res_{min}$$

when L_a^{PC} and R_a^{PC} are equal

We solve this problem via a DP recurrence. The cost function for placing a cell a at placement site b is as follows.

- **SHIFT:**

$$Cost(a,b) = \lambda_a \mid s_a - b \mid + Min_{i=x_{a-1}-SRCH}^{x_{a-1}+SRCH}\{Cost(a-1,i) + HCost(a,b,a-1,i)\}$$

$$Cost(1,b) = \lambda_1 \mid s_1 - b \mid$$

λ_a ($= e^{-\alpha slack_a}$) defines the weight of cell a according to its timing criticality using its slack value $slack_a$. This weight determines the relative importance of preserving the initial placement as opposed to displacing the cell to placement site b. The value of α is chosen such that it allows positive-slack cells to move while restricting the movement of timing-critical cells. $SRCH$ is the range over which the cell may be displaced. As noted above, the displacement is made in multiples of site width, and so the runtime for our algorithm is contained using this $SRCH$ parameter. High-utilization designs may require a large range of displacements to utilize the whitespace available in selective pockets but with a penalty of increase in runtime to attempt a search over larger $SRCH$.

$HCost$ denotes the cost of displacing cell a to site b, relative to a's immediate left neighbor in the row and depending on the distance between the corresponding boundary poly lines and

their colors. The method for computing the $HCost$ is shown in Figure 7. For each of the displacements made for cell a, we compute the cost of displacing the cell a by computing the cost incurred in displacing the cell by virtue of its criticality. The other cost incurred is the minimum of the summation of cost of displacing its left neighbor (over the set of displacements in the search range, whose cardinality is $(2\times SRCH+1)$) and the corresponding $HCost$. We take care of the flipped orientation of cells in the calculation of $HCost$. For instance, if the cell a is placed in flip-south (FS) or flip-north (FN) orientation - that is, if a is mirrored about the y-axis - then L_a^{PS} corresponds to R_a^{PS} and vice-versa. Similarly, R_a^{PC} is used in cost calculation instead of L_a^{PC}.

HCost(a,b,a-1,i) of Cell a

Input:
Displacement of cell a corresponding to site index b : δ_b
Displacement of cell $a-1$ corresponding to site index i : δ_i
x-left coordinate and width of cell a : x_a and w_a
x-left coordinate and width of cell $a-1$: x_{a-1} and w_{a-1}

Output:
Value of $HCost$

Algorithm:
01.**Case** $a=1$: $HCost(1,b)=0$
02.**Case** $a>1$
/* Calculate the spacing between cell a and $a-1$ according to new x-coordinates */
03. $spacing = x_a+\delta_b-x_{a-1}-\delta_i-w_{a-1}+L_a^{PS}+R_{a-1}^{PS}$
04.**If** $((spacing-Res_{min}<0)$ && $(L_a^{PC}==R_{a-1}^{PC}))$
$HCost(a,b,a-1,i)=\infty$
05.**else**
$HCost(a,b,a-1,i)=0$

Figure 7: $HCost$ algorithm for coloring conflict removal

3.2 DP with Recoloring (SHIFT+RECOLOR)

To maintain the timing goals of the design, the timing-critical cells need to be locked in their position while performing detailed placement perturbation. This can lead to very little or no reduction in the number of coloring conflicts, if these timing-critical cells block the movement of non-timing critical cells. The lack of whitespace available for perturbation in very high-utilization designs may also lead to non-convergence of the above-stated DP approach. The only alternative available at our disposal to remove coloring conflicts is the recoloring of the cells which are not fixed by the alternate coloring algorithm in Section 2. Therefore, we need to include the recoloring of the cells into our DP recurrence and assign a cost for recoloring the cell in our DP formulation. Removal of coloring conflicts must be achieved (even with slight degradation in timing) since it is needed to ensure printability of the layout patterns. Therefore, we also allow recoloring of fixed-color cells, but they are given high weight compared to other cells (as done for timing-critical cells in placement perturbation). We include the color of the cell as a new dimension for DP for this approach. Without loss of generality, we assume that cell a has original color C_{12} and is recolored to C_{21}. The cost of placing cell a at placement site b when recoloring of cells is allowed is as follows.

• **SHIFT+RECOLOR:**

$$
\begin{aligned}
Cost(a,b,C_{12}) &= \lambda_a \mid s_a-b \mid + Min[\\
&\quad \{Min_{i=x_{a-1}-SRCH}^{x_{a-1}+SRCH}\{Cost(a-1,i,C_{12}) \\
&\qquad + HCost(a,b,C_{12},a-1,i,C_{12})\}\}, \\
&\quad \{Min_{i=x_{a-1}-SRCH}^{x_{a-1}+SRCH}\{Cost(a-1,i,C_{21}) \\
&\qquad + HCost(a,b,C_{12},a-1,i,C_{21})\}\}] \\
Cost(a,b,C_{21}) &= \lambda_a \mid s_a-b \mid + \lambda_c^a + Min[\\
&\quad \{Min_{i=x_{a-1}-SRCH}^{x_{a-1}+SRCH}\{Cost(a-1,i,C_{12}) \\
&\qquad + HCost(a,b,C_{21},a-1,i,C_{12})\}\}, \\
&\quad \{Min_{i=x_{a-1}-SRCH}^{x_{a-1}+SRCH}\{Cost(a-1,i,C_{21}) \\
&\qquad + HCost(a,b,C_{21},a-1,i,C_{21})\}\}] \\
Cost(1,b,C_{12}) &= \lambda_1 \mid s_1-b \mid \\
Cost(1,b,C_{21}) &= \lambda_1 \mid s_1-b \mid + \lambda_c^1
\end{aligned}
$$

$Cost(a,b,C_{12})$ denotes the minimum cost of placing the cell a with original color C_{12} at placement site b when the color of cell $a-1$ can be either C_{12} or C_{21}. Similarly, $Cost(a,b,C_{21})$ denotes the minimum cost of placing the recolored cell a with color C_{21} at placement site b when the color of cell $a-1$ can be either C_{12} or C_{21}. The recoloring weight of cell a is defined as λ_c^a $(=e^{-\beta slack_a})$ where β can take different values to assign different recoloring weights to the cells. Fixed-color cells have high recoloring weight for high values of β because they also happen to be timing-critical cells with negative slack values. This DP formulation can achieve conflict-free design by combining placement perturbation and recoloring.

4. EXPERIMENTS

4.1 Experimental Setup

Library preparation. Each cell has two poly groups; one group ($G1$) is generated by the first patterning of the double patterning process, and the other group ($G2$) is generated by the second patterning. Each group, $G1$ or $G2$, has worst (MAX) and best (MIN) CD values. We use MAX_{G1} and MIN_{G1} to denote the $\pm 3\sigma$ CD values of the group $G1$, respectively, and MAX_{G2} and MIN_{G2} for the group $G2$. Each group has CD variation that is independent of the other's, so there are four corner combinations for a single master cell, i.e., (MAX_{G1}, MAX_{G2}), (MAX_{G1}, MIN_{G2}), (MIN_{G1}, MAX_{G2}) and (MIN_{G1}, MIN_{G2}).

Among the four combinations, (MAX_{G1}, MAX_{G2}) dominates the worst-case delay [13]. This worst-case is split into two different scenarios, $MAX_{G1} \geq MAX_{G2}$ and $MAX_{G1} \leq MAX_{G2}$, since we cannot presume which poly group will have larger or smaller gate length than the other. We use $G1L-G2S$ and $G1S-G2L$ as the names of bimodal-aware timing libraries corresponding to the scenarios $MAX_{G1} \geq MAX_{G2}$ and $MAX_{G1} \leq MAX_{G2}$, respectively. Each scenario can further be split based on the mean CD difference values between the two groups.

We choose the most commonly used 36 standard cells from the Nangate $45nm$ Open Cell Library [20]. We create two coloring versions for each standard cell, e.g., $NAND2\_C_{12}$ and $NAND2\_C_{21}$ for a $NAND2$ cell. We characterize delay of all coloring versions of cells using the Predictive Technology Model (PTM) [19], with respect to the two scenarios $G1L-G2S$ and $G1S-G2L$ and the three cases of CD mean difference equal to 2, 4 and $6nm$. Table 2 summarizes timing libraries that we generate for bimodal-aware timing analysis. Unimodal CD values corresponding to the bimodal CD values follow the calculation used in [6]. With the unimodal timing models, we require one timing signoff, while with the bimodal timing models, we need two timing signoffs, one for each of the $G1L-G2S$ and $G1S-G2L$ cases.

Table 2: Bimodal-aware timing libraries.

Mean CD Diff (→)	$2nm$		$4nm$		$6nm$	
	CD (nm) of each CD group $G1$ and $G2$					
Corner Name (↓)	G1	G2	G1	G2	G1	G2
Unimodal	53.51		56.32		59.22	
$G1L-G2S$	51	49	52	48	53	47
$G1S-G2L$	49	51	48	52	47	53

We define the minimum spacing between same-colored poly lines (Res_{min}) to be $330nm$, which is calculated by subtracting the poly width ($50nm$) from twice the defined poly pitch ($2\times 190nm$) in the Nangate $45nm$ library. Separately, all 72 (36 × 2) standard cells are analyzed and CSC values are recorded.

Testcase preparation. We implement two open-source cores, AES and $JPEG$, obtained as RTL from the open-source site *opencores.org* [21]. We synthesize the cores using *Cadence RTL Compiler v5.2* [23] with the original (non-bimodal) timing library which does not have coloring information and which assumes worst-case CD values of $50nm$ for all transistors. We use *Cadence SOC Encounter v7.2* [24] to place and route with two different placement utilizations (70% and 90%), to compare the difficulty of coloring conflict removal.

We then assign color C_{12} or C_{21} to all cell instances by replacing the original master cell names, e.g., NAND2 with one of its colored master cell names, $NAND2\_C_{12}$ or $NAND2\_C_{21}$. For this initial coloring assignment, the only objective is to not create coloring conflicts which is the only constraint for the traditional DPL pattern decomposition. We first assign a color to the leftmost cell in each cell row and assign a color of the next cell so as to not create a coloring conflict with the first cell, then iterate this simple assignment method to the end of the cell row. For each initially colored design, we extract RC parasitics from *SOC Encounter* and then perform timing analysis with *Synopsys PrimeTime vB-2008.12-SP2* [25].

Table 3 summarizes design and timing information of our testcases when the $2nm$ CD mean difference library is used. WNS and TNS respectively represent the worst negative slack of the design and the total negative slack (which is the sum of all negative slacks) over all the end points of timing paths. WNS can be regarded as the feasibility of timing closure at the given clock cycle time, and TNS can be regarded as the required effort to fix all timing violations of the design. Both WNS and TNS are reported using unimodal library in Rows 7 and 8, and using bimodal-aware timing libraries in Rows 9 and 10. Timing with the original (single CD distribution) timing library is met at the given clock cycle times. However, due to the bimodal CD distribution, timing of the double patterning-applied designs is significantly degraded. It must be noted that the unimodal timing analysis is more pessimistic than the bimodal-aware timing analysis. We also observe that the use of the bimodal-aware timing library can by itself directly improve timing significantly, due to the 'intrinsic' alternate coloring within a cell, as we observe in the fall delay of a NAND2 in which CD variation of MN1 is compensated by opposite CD variation of MN2 (refer to Figure 4).

Table 3: Testcase information.

	$AES70$	$AES90$	$JPEG70$	$JPEG90$
#Instances	26026	20784	92898	92619
#-C_{12} instances	21350	13388	77807	60136
#-C_{21} instances	4676	7396	15091	32483
Area (um^2)	44848	29765	175742	137571
Util. (%)	69.10	91.15	68.85	88.31
WNS (ns) w/ Uni.	-0.728	-0.789	-1.113	-1.113
TNS (ns) w/ Uni.	-161.0	-145.3	-671.1	-587.1
WNS (ns) w/ Bi.	-0.117	-0.157	-0.191	-0.147
TNS (ns) w/ Bi.	-5.9	-2.7	-8.170	-7.756

4.2 Experimental Flow

Figure 8 shows our design optimization framework for double patterning. Major steps are shown on the left-hand side, and output data on the right-hand side. Solid arrows show the design flow and dashed arrows show the data flow.

Step 1: Initial design. For the initial testcase preparation, we use a traditional timing-driven design implementation flow. The design starts with RTL netlists and timing constraints, and is synthesized, placed and routed with the original (traditional) worst-case timing library of single CD distribution.

Step 2: Initial coloring. The framework performs initial coloring for double patterning, in which no coloring conflicts are allowed. The output is design exchange format (initial_colored.def).

Step 3: Timing analysis. Based on the coloring information and bimodal-aware timing libraries, a static timer analyzes timing, and generates an ILP problem instance for the top-k critical timing paths.

Step 4: Optimal coloring. The ILP solver finds the optimal alternate coloring solution for the selected timing paths, and at the same time, a pre-defined coloring is assigned to all clock buffers. All coloring solutions for the cells in the timing-critical paths and clock paths (keep_color.list), timing slack values for all cell instances (slack.list), and the updated design (opt_colored.def) are generated.

Step 5: Conflict removal. DPL-Corr solves the coloring conflicts created during Step 4, subject to the coloring constraints (keep_color.list) and timing constraints (slack.list). Partially disconnected nets due to the placement perturbation in DPL-Corr are ECO-routed, and a final design (opt.def) that does not have coloring conflicts is generated.

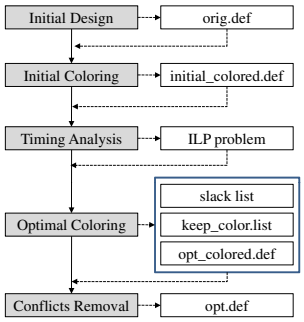


Figure 8: Design framework for bimodal-aware timing optimization.

The orig.def is not suitable for double patterning. After Step 2, initial_colored.def is amenable to double patterning, since this design does not have coloring conflicts. We use this initial_colored.def as the reference double patterning-applied design for our comparison. The opt_colored.def is used to show the pure effect of our optimal coloring method, since the placement locations of cells and the routing are not disturbed from the reference design. We finally compare the timing quality of the final design (opt.def) with the reference design (initial_colored.def). We note that our optimization framework does not increase the given die area at all, since it perturbs the placement using existing whitespace between placed cells.

4.3 Experimental Results

Our first experiment verifies the quality of our alternate color assignment method, varying the number of timing-critical paths taken into account. We apply the alternate color assignment to the timing-critical paths and uniform color assignment to the clock paths, but do not apply DPL-Corr that may introduce other timing uncertainty from placement perturbation and ECO-routing.[3]

We compare the $CSCP$ and TNS values of the initially colored design (initial_colored.def) with those of an optimally colored design (opt_colored.def). Table 4 shows $CSCP$ and TNS reduction from the alternate color assignment on different top-k critical paths of $AES70$. Runtime is listed in Row 2. Average $|CSCP|$ of the top-k critical paths (Avg. $CSCP$) of the initially colored design ("Initial Coloring") and of the alternately colored design ("Alternate Coloring") is given in Rows 4 and 8, respectively. The total negative slack[4] of the "Initial Coloring" and "Alternate Coloring" are compared in Rows 5-7 and 9-11, respectively, for CD mean difference of $2nm$, $4nm$ and $6nm$. We observe that as the number of timing paths increases, the average $CSCP$ of the optimized design decreases somewhat, and the total negative slack improves.

Our second experiment verifies the quality of our alternate color assignment method on different testcases. Table 5 summarizes the WNS and TNS reduction from the alternate color assignment for different testcases. For this experiment, we consider top-400 critical paths for $AES70$ and $AES90$, and top-2000 critical paths for $JPEG70$ and $JPEG90$. We observe that WNS is improved by more than $100ps$ and TNS is improved from -8.2ns to -1.5ns in $JPEG70$, even with $2nm$ mean CD difference libraries.[5]

Our third experiment measures the performance of the proposed DPL-Corr in Section 3. Table 6 shows the results of our

[3]The uniform color assignment of the clock paths are not so effective for our testcases, since the number of clock stages from the clock source to registers is two and three for $AES70-90$ and $JPEG70-90$ testcases, respectively. However, the uniform color assignment of the clock network may have a significant impact when a large number of stages is used for clock networks, as discussed in [13].

[4]We perform timing analysis with the two worst-corner bimodal-aware timing libraries $G1L-G2S$ and $G1S-G2L$. Between the two timing results, the worse one is reported as the timing of the design.

[5]Percentage improvements can be affected by the number of stages of timing-critical paths.

Table 4: Average $CSCP$ of the top-k critical paths and TNS (ns) reduction via the alternate color assignment. TNS is measured at three mean CD difference cases.

#Critical timing paths (k)		100	200	300	400	500
Runtime (s)		0.22	0.37	0.58	1.97	2.13
Initial Coloring	Avg. $CSCP$	1.93	1.93	1.94	1.92	1.90
	TNS @$2nm$	-5.88	-5.88	-5.88	-5.88	-5.88
	TNS @$4nm$	-15.30	-15.30	-15.30	-15.30	-15.30
	TNS @$6nm$	-26.03	-26.03	-26.03	-26.03	-26.03
Alternate Coloring	Avg. $CSCP$	1.88	1.86	1.89	1.86	1.85
	TNS @$2nm$	-4.64	-4.17	-3.66	-3.34	-3.15
	TNS @$4nm$	-13.14	-12.34	-11.29	-10.45	-9.72
	TNS @$6nm$	-23.11	-22.00	-20.88	-19.84	-19.08

Table 5: WNS (ns) and TNS (ns) comparison before and after alternate coloring for different testcases. (Pure impact of alternate coloring without including placement perturbation)

CD diff.			AES70	AES90	$JPEG$70	$JPEG$90
$2nm$	Init.	WNS	-0.117	-0.157	-0.191	-0.147
	Color	TNS	-5.88	-2.72	-8.17	-7.76
	Alt.	WNS	-0.110	-0.115	-0.090	-0.139
	Color	TNS	-3.34	-1.70	-1.48	-4.03
$4nm$	Init.	WNS	-0.226	-0.224	-0.354	-0.250
	Color	TNS	-15.30	-8.72	-26.56	-20.89
	Alt.	WNS	-0.159	-0.166	-0.145	-0.200
	Color	TNS	-10.46	-3.25	-3.85	-10.40
$6nm$	Init.	WNS	-0.324	-0.314	-0.527	-0.423
	Color	TNS	-26.03	-18.12	-64.64	-47.01
	Alt.	WNS	-0.238	-0.275	-0.267	-0.286
	Color	TNS	-19.84	-14.36	-22.40	-19.24

DP-based coloring conflict removal algorithms on different testcases. The table shows the performance statistics for DP implementations on testcases with "Initial Coloring" and "Alternate Coloring" respectively. The experiments are performed for different values of α to highlight the performance of the algorithm as the movement of timing-critical cells is restricted progressively from unrestricted movement to no movement.

The performance data are reported for both DP variants, i.e., with and without recoloring. The value of $SRCH$ is taken as 50 and the value of β for determining recoloring weight is 30. We report the number of conflicts after placement optimization in Columns 5 and 9, the sum of displacements for timing-critical cells (SODTCC) in Columns 6 and 10, and the sum of displacements of non-timing critical cells (SODNTCC) in Columns 7 and 11 with respect to **SHIFT** and **SHIFT+RECOLOR**. The number of recolorings of fixed-color cells (FCCD) is in Column 12. The sum of displacements of timing-critical (non-critical) cells, SODTCC (SODNTCC), decreases (increases) with increase in the value of α as the movement of timing-critical cells is restricted and movement of non-critical cells is needed to compensate for that restriction to achieve the same results. Runtime in Columns 8 and 13 shows linear scalability of our algorithms with respect to instance count. The instance counts of JPEG testcases are 4∼5 times larger than those of AES testcases, and the runtimes for JPEG testcases are 3∼4 times larger than those for AES.

The first DPL-Corr algorithm **SHIFT** is able to resolve all the coloring conflicts for testcases with a given initial coloring at 70% placement utilization, when α values are 0 and 20. For high-utilization testcases (AES90 and $JPEG$90), the algorithm is able to remove around 33% and 39%, respectively, of conflicts without any recoloring. Understandably, the algorithm performance suffers drastically when α is set to ∞, since this means that all timing-critical cells are locked in their positions. The number of conflicts can be reduced by increasing the $SRCH$ range for displacements since the whitespace is often not distributed evenly over the entire standard-cell row. This approach can reduce the number of conflicts but the runtime can increase substantially. It should be noted that whitespace management alone cannot guarantee complete conflict removal in high-utilization designs because the algorithm is restricted by the lack of whitespace needed for complete conflict removal.

The goal of complete conflict removal can be realized fully with our second DPL-Corr algorithm, **SHIFT+RECOLOR**. We apply this DP algorithm to the alternate coloring results since this is the case of interest from a flow and methodology standpoint. The number of conflicts reduces to 0 for all values of α, albeit with a slight penalty incurred in recoloring the fixed color cells obtained from alternate coloring results. We reiterate the importance of complete conflict removal over maximizing alternate coloring in timing-critical paths, since the former is a requirement for printability of the layout patterns in DPL regime. The sum of displacements of timing-critical cells (SODTCC) and the sum of displacements of non-timing critical cells (SODNTCC) are both reduced considerably with this approach, implying less effort in subsequent ECO-routing. We also observe that there is a slight degradation in timing when DP-based recoloring is applied; this is attributable to the fact that the colors of some fixed cells are changed. The runtime is of the order of several minutes, since the number of conflicts is smaller. When timing-critical cells are locked in their initial placement positions ($\alpha = \infty$), we must resort to DP-based recoloring to guarantee a conflict-free design. In general, we observe that DP with recoloring can handle all of our testcases with ease and can guarantee complete conflict removal with practical runtime.

Finally, we compare timing slack and timing slack variation before and after coloring optimization, including the effect of the placement perturbation and ECO-routing due to the conflict removal. In the case that there exists enough whitespace in the design, our first DPL-Corr algorithm **SHIFT** can solve coloring conflicts completely without significant degradation, for reasonable ranges of α values. Table 7 shows timing of the initial coloring and timing after coloring optimization with DPL-Corr **SHIFT**, for the $JPEG$70 testcase. We use timing criticality weight α=20. 'Worst WNS' and 'Worst TNS' represent the worse timing slack between the two corner libraries, and 'WNS diff.' and 'TNS diff.' represent the slack difference between the two corner libraries.

Due to the placement perturbation and ECO routing, the worst negative slack is degraded by a maximum of 38ps from the results of the alternate color assignment only. However, we still observe 87ps, 171ps and 232ps of WNS reduction, and 4.74ns, 16.11ns and 36.22ns of TNS reduction for 2nm, 4nm and 6nm mean CD difference in bimodal CD distribution, respectively, for $JPEG$70. In addition, the maximum variation of worst (total) negative slack between the two corner libraries is reduced from 380ps (63.55ns) to 84ps (22.03ns) for 6nm mean CD difference libraries. This implies more robust timing of the design with respect to CD distribution changes of $G1$ and $G2$.

Table 7: WNS and TNS comparison before and after applying alternate coloring and DPL-Corr SHIFT, and the timing difference between two worst bimodal-aware libraries on $JPEG$70 testcases with respect to the mean CD difference.

Stage	Timing	Mean CD diff.		
		$2nm$	$4nm$	$6nm$
Initial Coloring	WNS (ps) w/ $G1L-G2S$	-191	-354	-527
	WNS (ps) w/ $G1S-G2L$	-51	-105	-147
	Worst WNS (ps)	-191	-354	-527
	WNS diff. (ps)	140	249	380
	TNS (ns) w/ $G1L-G2S$	-8.17	-26.56	-64.64
	TNS (ns) w/ $G1S-G2L$	-0.30	-0.66	-1.09
	Worst TNS (ns)	-8.17	-26.56	-64.64
	TNS diff. (ns)	7.87	25.90	63.55
Alternate Coloring + DPL-Corr (**SHIFT**)	WNS (ps) w/ $G1L-G2S$	-85	-183	-295
	WNS (ps) w/ $G1S-G2L$	-104	-171	-211
	Worst WNS (ps)	-104	-183	-295
	WNS diff. (ps)	19	12	84
	TNS (ns) w/ $G1L-G2S$	-3.43	-10.45	-28.42
	TNS (ps) w/ $G1S-G2L$	-1.32	-3.74	-6.39
	Worst TNS (ns)	-3.43	-10.45	-28.42
	TNS diff. (ns)	2.11	6.71	22.03

In the case that the design is very congested, our second DPL-Corr algorithm **SHIFT+RECOLOR** can completely solve the coloring conflicts without significant timing degradation. Table 8 summarizes timing slack changes due to alternate color assignment and DPL-Corr, including ECO-routing and recoloring. Again, the worse WNS and TNS values are chosen from the results of the two bimodal-aware timing analyses. We report the highest utilization cases for each design.

In summary, we effectively reduce the timing slack as well as timing slack variation in double patterning-applied designs using the alternate color assignment technique, and we completely solve coloring conflicts, thus minimizing timing slack degradation, using our two DPL-Corr techniques.

Table 6: DP-based coloring conflict removal. SODTCC and SODNTCC denote sum of displacements (μm) of timing-critical cells and of non-timing critical cells, respectively. FCCD represents the number of recolored cells during DPL-Corr.

	Testcase	#Conf.	α	SHIFT				SHIFT+RECOLOR				
				#Conf.	SODTCC	SODNTCC	Runtime (s)	#Conf.	SODTCC	SODNTCC	FCCD	Runtime (s)
Random Coloring	$AES70$	4027	0	0	880.08	2835.94	98.59	0	63.46	8.36	72	271.62
			20	0	872.86	3368.51	98.72	0	47.50	38.00	85	270.74
			∞	4000	0	23.37	97.30	0	0	1829.89	147	273.82
	$AES90$	6574	0	4394	995.22	8961.73	52.22	0	12.35	13.68	27	205.76
			20	4394	971.09	9638.51	52.05	0	11.59	42.18	29	209.42
			∞	6516	0	88.16	53.76	0	0	4824.67	32	218.02
	$JPEG70$	11198	0	0	1049.18	7034.18	205.19	0	158.08	40.28	36	578.82
			20	0	715.16	8980.73	204.79	0	143.83	254.98	34	578.87
			∞	10794	0	250.99	235.76	0	0	5034.05	73	649.00
	$JPEG90$	27296	0	16668	4785.34	49532.05	347.23	0	64.98	71.25	152	607.88
			20	16668	4660.13	53842.2	347.73	0	54.72	213.75	158	607.86
			∞	27296	0	0	348.04	0	0	22005.04	199	678.69
Alternate Coloring	$AES70$	368	0	0	140.03	158.84	99.83	0	38.19	6.08	81	164.31
			20	0	143.07	209.19	99.47	0	27.17	21.28	96	164.26
			∞	353	0	7.22	97.96	0	0	241.68	144	190.32
	$AES90$	139	0	29	94.62	442.13	50.02	0	1.90	1.14	36	144.93
			20	29	96.33	501.22	50.56	0	1.52	13.49	47	144.96
			∞	118	0	43.7	52.03	0	0	156.75	32	153.97
	$JPEG70$	219	0	0	72.96	121.79	354.98	0	158.84	60.80	34	578.97
			20	0	54.91	184.11	354.83	0	143.26	140.98	35	579.38
			∞	135	0	37.05	352.24	0	0	380.19	60	650.48
	$JPEG90$	712	0	0	346.18	1726.53	210.36	0	30.21	9.5	161	607.71
			20	0	349.22	2048.20	210.14	0	20.33	55.48	182	607.91
			∞	626	0	164.35	234.17	0	0	616.93	171	676.26

Table 8: WNS and TNS comparison before and after applying alternate coloring and DPL-Corr SHIFT +RECOLOR on $AES90$.

Stage	Timing	Mean CD diff.		
		$2nm$	$4nm$	$6nm$
Initial Coloring	WNS (ns)	-0.157	-0.224	-0.314
	TNS (ns)	-2.72	-8.72	-18.12
Alternate Coloring Only	WNS (ns)	-0.115	-0.166	-0.275
	TNS (ns)	-1.70	-3.25	-14.36
DPL-Corr (**SHIFT+RECOLOR**)	WNS (ns)	-0.128	-0.178	-0.243
	TNS (ns)	-1.78	-3.72	-12.21

5. CONCLUSIONS AND ONGOING WORK

Double patterning is an inevitable lithography solution and is being adopted for $32nm$ and below technologies. However, due to the two CD populations within a die, on-chip timing variability increases substantially beyond the variability that occurs with traditional single-exposure lithography.

To mitigate the timing variability in double patterning, we have proposed a new metric that quantifies the delay variation of timing paths, and implemented an optimal cell-based timing-aware color assignment technique for double patterning that reduces both timing delay as well as timing variation. To address the increased coloring conflicts due to this intentional timing-aware coloring, we have also proposed a dynamic programming-based detailed placement algorithm that minimizes coloring conflicts by perturbing placement and exploiting whitespace in the given placement.

With our new methodology, we effectively reduce the timing delay as well as timing variation for DPL-patterned designs. We achieve maximum $232ps$ ($36ns$) reduction in the worst (total) negative slack and 78% (65%) reduction in the worst (total) negative slack variation in double patterning-applied designs.

Our next goals are (i) to seek more accurate metrics for the timing path balancing in order to further enhance the timing quality, (ii) to explore different objectives for the placement perturbation, (iii) to investigate a tradeoff of the costs between recoloring and displacement on timing quality, and finally, (iv) to develop a simultaneous timing-aware coloring and conflict removal methodology as a golden timing and placement optimizer for double patterning lithography.

6. REFERENCES

[1] A. M. Biswas et al., "Extension of 193 nm Dry Lithography to 45-nm Half-Pitch Node: Double Exposure and Double Processing Technique", *Proc. SPIE Photomask Technology*, Vol. 6349, pp. 63491P-1 - 63491P-9.

[2] G. Capetti et al., "Sub k1 = 0.25 Lithography with Double Patterning Technique for 45nm Technology Node Flash Memory Devices at 193nm", *Proc. SPIE Optical Microlithography*, vol. 6520, pp. 65202K-1 - 65202K-12, 2007

[3] K. Chen, W. Huang, W. Li and P. Varanasi, "Resist Freezing Process for Double-Exposure Lithography", *Proc. SPIE Advances in Resist Materials and Processing Technology XXV*, Vol. 6923, 2008, pp. 69230G-1–69230G-10.

[4] M. Cho, Y. Ban and D. Z. Pan, "Double Patterning Technology Friendly Detailed Routing", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp.506-511.

[5] M. Drapeau et al., "Double Patterning Design Split Implementation and Validation for the 32nm Node", *Proc. SPIE Design for Manufacturability through Design-Process Integration*, Vol. 6521, 2007.

[6] M. Dusa et al., "Pitch Doubling Through Dual-Patterning Lithography: Challenges in Integration and Litho Budgets", *Proc. SPIE Conference on Optical Microlithography*, 2007, pp. 65200G-1 - 65200G-10.

[7] J. Finders, M. Dusa and S. Hsu, "Double Patterning Lithography: The Bridge Between Low k1 ArF and EUV", *Microlithography World*, Feb. 2008.

[8] R. S. Ghaida and P. Gupta, "Design-Overlay Interactions in Metal Double Patterning", *Proc. SPIE Design for Manufacturability through Design-Process Integration III*, Vol. 7275, 2009, pp. 14.1-10.

[9] P. Gupta, A. B. Kahng, Y. Kim and D. Sylvester, "Self-Compensating Design for Focus Variation", *Proc. Design Automation Conf.*, June 2005, pp. 365-368.

[10] P. Gupta, A. B. Kahng and C.-H. Park, "Detailed Placement for Improved Depth of Focus and CD Control", *Proc. Asia and South Pacific Design Automation Conf.*, Jan. 2005, pp. 343-348.

[11] B. Hwang et al., "Development of 38nm Bit-Lines using Copper Damascene Process for 64-Giga bits NAND Flash", *Proc. Advanced Semiconductor Manufacturing Conference*, 2008, pp. 49-51.

[12] M. Hori et al., "Sub-40-nm Half-Pitch Double Patterning with Resist Freezing Process", *Proc. SPIE Advances in Resist Materials and Processing Technology XXV*, Vol. 6923, 2008, pp. 69230H-1–69230H-8.

[13] K. Jeong and A. B. Kahng, "Timing Analysis and Optimization Implications of Bimodal CD Distribution in Double Patterning Lithography", *Proc. Asia and South Pacific Design Automation Conf.*, 2009, pp. 486-491.

[14] K. Jeong, A. B. Kahng, and R. O. Topaloglu, "Is Overlay Error More Important Than Interconnect Variations in Double Patterning?", *Proc. ACM International Workshop on System Level Interconnect Prediction*, 2009, to appear.

[15] A. B. Kahng, C.-H. Park, X. Xu and H. Yao, "Layout Decomposition for Double Patterning Lithography", *Proc. ACM/IEEE Intl. Conf. on Computer-Aided Design*, 2008, pp. 465-472.

[16] K. Yuan, J.-S. Yang, D. Z. Pan, "Double Patterning Layout Decomposition for Simultaneous Conflict and Stitch Minimization", *Proc. International Symposium on Physical Design*, 2009, pp. 107-114.

[17] J.-S. Yang and D. Z. Pan, "Overlay Aware Interconnect and Timing Variation Modeling for Double Patterning Technology", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp. 488-493.

[18] *International Technology Roadmap for Semiconductors*, 2007 Edition, http://public.itrs.net/.

[19] *Predictive Technology Model*, http://www.eas.asu.edu/∼ptm.

[20] *NANGATE*, http://www.nangate.com/

[21] *OPENCORES.ORG*, http://www.opencores.org/.

[22] *ILOG CPLEX*, http://www.ilog.com/products/cplex/.

[23] *Cadence RTL Compiler User's Manual.* http://www.cadence.com/.

[24] *Cadence SOC Encounter User's Manual.* http://www.cadence.com/.

[25] *Synopsys PrimeTime User's Manual.* http://www.synopsys.com/.