ILP-Based Co-Optimization of Cut Mask Layout, Dummy Fill and Timing for Sub-14nm BEOL Technology

Kwangsoo Han[†], Andrew B. Kahng[†‡], Hyein Lee[†] and, Lutong Wang[†]

[†]ECE and [‡]CSE Depts., University of California at San Diego, La Jolla, CA USA 92093
{kwhan, abk, hyeinlee, luw002}@ucsd.edu

ABSTRACT

Self-aligned multiple patterning (SAMP), due to its low overlay error, has emerged as the leading option for 1D gridded back-end-of-line (BEOL) in sub-14*nm* nodes. To form actual routing patterns from a uniform "sea of wires", a cut mask is needed for line-end cutting or realization of space between routing segments. Constraints on cut shapes and minimum cut spacing result in end-of-line (EOL) extensions and non-functional (i.e. dummy fill) patterns; the resulting capacitance and timing changes must be consistent with signoff performance analyses and their impacts should be minimized.

In this work, we address the co-optimization of cut mask layout, dummy fill, and design timing for sub-14*nm* BEOL design. Our central contribution is an optimizer based on integer linear programming (ILP) to minimize the timing impact due to EOL extensions, considering (i) minimum cut spacing arising in sub-14*nm* nodes; (ii) cut assignment to different cut masks (color assignment); and (iii) the eligibility to merge two unit-size cuts into a bigger cut. We also propose a heuristic approach to remove dummy fills after the ILP-based optimization by extending the usage of cut masks. Our heuristic can improve critical path performance under minimum metal density and mask density constraints.

In our experiments, we study the impact of number of cut masks, minimum cut spacing and metal density under various constraints. Our studies of optimized cut mask solutions in these varying contexts give new insight into the tradeoff of performance and cost that is afforded by cut mask patterning technology options.

1. INTRODUCTION

Self-aligned multiple patterning (SAMP), due to its low overlay error, has emerged as the leading option for the 1D gridded BEOL on "1×" or "Mx" layers in sub-14*nm* nodes. Figure 1(a) shows a part of a target layout, which is finalized as of post-routing design. In the first step of fabrication, we first generate uniform "sea of wires" as shown in Figure 1(b). In the next step, to form actual routing patterns from the "sea of wires", we make cuts on the wire segments by using cut masks as shown in Figure 1(c). Figure 1(d) shows the final layout. In addition to the target layout, the final layout includes end-of-line (EOL) extensions, which are attached to the routing segments of the target layout, and dummy fills, which are floating.

There are several ways to print cuts, such as 193*i* immersion lithography and electron-beam (e-beam) technology. E-beam is costly due to the intrinsically low throughput of "writing" as opposed to "printing". Conventional 193*i* patterning remains a viable alternative. However, multiple 193*i* cut masks typically must be used to increase granularity. In other words, for a set of cut shapes to be printed by the same cut mask, the spacing between any two cuts must be at least the minimum cut spacing*. To print cut shapes with closer spacing requires more cut masks (colors). Further, complex cut shapes (e.g., non-rectangular shapes) may cause pattern fidelity loss and risk of yield loss. Thus, cuts must be assigned to different cut masks (color assignment) and nicely distributed with simple cut shapes, i.e., rectangular shapes. These constraints on cut mask shapes and colorability result in EOL extensions beyond what is originally seen in the layout tool; the resulting capacitance and timing changes must be consistent with signoff performance analyses. Furthermore, cut mask shapes determine the amount of non-functional (i.e., dummy fill) patterns that remain from the original "sea of wires"; this must be consistent with area density bounds and timing constraints.

In this work, we address the co-optimization of cut mask layout, dummy fill, and design timing for sub-14*nm* BEOL design. Our central contribution is an optimizer based on integer linear programming (ILP) that minimizes the timing impact due to EOL extensions, with consideration of (i) minimum cut spacing arising in sub-14*nm* nodes; (ii) cut

*According to 2015 ITRS reports,[6] the lithography "cliff" for 2D shapes (e.g., cut) is approximately 110*nm*.

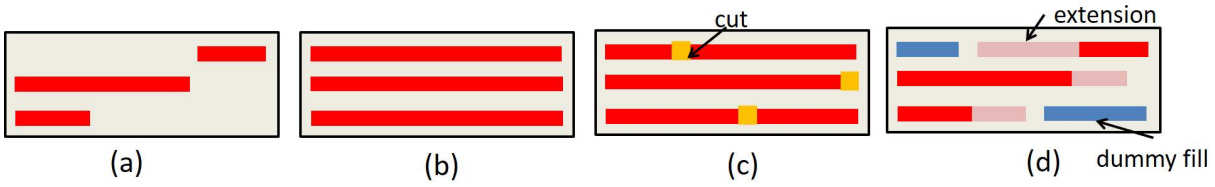

Figure 1: SAMP process overview: (a) target layout; (b) 1D wires; (c) 1D wires with cuts; and (d) final layout.

assignment to different cut masks (color assignment); and (iii) the eligibility to merge multiple unit-size cuts into a larger cut.

We minimize timing impacts by assigning a timing slack-dependent weight to each wire segment. To enable our optimization to apply at full-chip scale, a partitioning-based method is used to achieve linear scaling of runtime with layout area. Finally, beyond finding optimal locations of line-end cuts, we develop a heuristic to remove dummy fills in an effort to improve timing performance of critical paths, subject to minimum metal density and cut mask density constraints. Our contributions are summarized as follows.

- To our knowledge, ours is the first work that considers timing, metal density, and cut mask density simultaneously.
- We formulate an ILP-based optimization of unit-size cut locations together with cut mask assignment (color assignment). Our ILP-based optimizer minimizes the timing impacts (i.e., slack degradation on critical timing paths) due to EOL extensions of wire segments.
- We develop a post-ILP optimization flow that further optimizes timing by enlarging and/or inserting cuts to remove dummy fills around timing-critical segments, while satisfying prescribed minimum metal density constraints and considering cut mask density uniformity.
- Our experiments across different numbers of cut masks, minimum metal densities and minimum cut spacings give insight into a significant performance-cost tradeoff that can be afforded by cut mask patterning technology options.

The remainder of this paper is organized as follows. Section 2 gives a review of related works. Section 3 describes our cut mask optimization approach, which consists of (i) ILP-based optimization of unit-size cut locations, (ii) a scalable partitioning-based method to handle larger designs, and (iii) a heuristic to remove dummy fills according to various cut mask layout rules. Section 4 presents our experimental results, and Section 5 concludes the paper.

2. RELATED WORKS

In this section, we first introduce previous works to support the use of SAMP and 193*i* line-end cuts towards sub-14*nm* nodes. We then list a couple of related works for cut mask optimization.

Using 193*i* and line-end cuts towards sub-14*nm* nodes. Owa et al.[8] investigate the possibility of extending 193*i* patterning to sub-10*nm* nodes. They provide experimental data for SAMP and Litho-Etch (LE) cuts down to the 5*nm* node. Notably, they apply *self-aligned quadruple patterning* (SAQP), a type of SAMP process, with 11.8*nm* half-pitch to support the use of unidirectional patterning with multiple LE cuts at the 7*nm* node. A cost model of SAMP is evaluated, assuming that the cost (and, number of repetitions) of litho-etch processes is simply proportional to transistor density. This assumption ensures printability but is pessimistic in that it increases the node-to-node per-transistor cost scaling factor from 0.7× to 0.86×, making it a less cost-effective option. Gillijns et al.[4] study 193*i* patterning for N10 and N7 BEOL[†], contrasting the use of cut masks against the removal of all excess metal fill shapes. They show that when moving to the N7 node, a line-end cut option affords better process window with fewer cut masks, at the cost of increased wire

[†]N10 (resp. N7) is foundry nomenclature for "10nm node" (resp. "7nm node"), just as foundry nomenclature for the first foundry FinFET node (with minimum metal pitch = 64*nm*) was N14 (Samsung, GLOBALFOUNDRIES) or N16 (TSMC).

length, capacitance and power. These two works provide motivating context for our present study. In our present work, we focus on achievable tradeoffs between IC performance and cut mask cost on $1\times$ layers. In particular, we demonstrate an effective timing optimization that simultaneously keeps mask cost down by using fewer cut masks.

Shortest path-based approach. Zhang et al.[9] use a shortest path-based method to improve the printability of cuts. The authors categorize cuts into two groups based on their printability. One type is *regular*, which is a cut adjacent to a routing segment. The other type is *critical*, which is a cut adjacent to the line-end of a routing segment. The authors investigate tradeoff between performance and printability. However, their model is not timing-aware, and it does not consider the usage of multiple cut masks. Also, since regular cuts may be printed without printability issues, there may be a guardband to be optimized.

Integer linear programming-based approaches. Du et al.[3] propose a hybrid optimization of cut masks with e-beam by using integer linear programming (ILP). In their work, they generate minimum spacing rules within the same and across tracks according to a lithography simulation. They propose an ILP model to handle these constraints. The objective is to minimize the usage of throughput-constrained e-beam technology. Compared to Zhang et al.,[9] Du et al.[3] use more realistic design rules derived from lithography simulation. However, their solver takes up to a day to obtain an optimal solution for larger designs. Ding et al.[2] improve Du et al.'s ILP formulation to reduce solver runtime; their updated ILP formulation has fewer binary variables and introduces an extension limit for each wire segment which can reduce ILP solver runtime. However, rather than performing a design-specific timing optimization, Ding et al. simply minimize the sum of EOL extensions without consideration of possible tradeoffs involving timing-critical wire segments. Furthermore, their ILP formulation does not support cut assignments to multiple cut masks.

3. OUR APPROACH

In this section, Subsection 3.1 describes our ILP-based optimization of cut locations, and cut assignments to different cut masks, to minimize the impact of end of line extensions on timing. Subsection 3.2 then proposes a timing- and density-aware post-ILP optimization to minimize the impact of dummy fills considering metal and mask densities. Subsection 3.3 explains our overall flow.

3.1. ILP-based Cut Mask Optimization

For a given 1D routed layout, we seek cut locations and assignments of cuts to different cut masks so as to minimize the impact of line end extensions on critical-path timing. We assume a horizontal routing layer in the following discussion. For any horizontal wire segment w, there are exactly two unit-size (i.e., minimum horizontal half-pitch $\times$ minimum vertical half-pitch size) cuts at each right and left end-of-line. Given minimum cut spacing min_s, any two cuts that are located within min_s of each other cannot be printed with a single cut mask. To address this printability problem, we can relocate one or both of the cuts so that their separation becomes larger than min_s, or so that they are merged and form a larger cut. Another solution is to assign the cuts to different cut masks. If we use a unique color to represent each cut mask, then assigning each cut to a cut mask is equivalent to assigning a color to each cut (we refer to this as *color assignment*).

In our formulation, cuts are located on grid points, with each grid point corresponding to the intersection of perpendicular tracks of adjacent metal layers. We define the *relocation range* for each cut as the set of grid points to which the cut can be relocated. Relocation ranges may be subject to maximum EOL extension limits for each wire segment, and cannot overlap with any existing routing segments. There is an obvious tradeoff between timing and cost: relocation of cuts leads to EOL extensions which affect the timing of paths going through the extended wire segment. On the other hand, use of additional cut masks, while helping to control line-end extensions, adds to process cost and, potentially, process variability as well.

We now describe our ILP formulation for the cut mask optimization problem. The variables used in our formulation are summarized in Table 1.

Minimize: $\sum_{w \in W} a_w \cdot ((x_{cr_w} - x_{cl_w}) - (r_w - l_w))$

The objective is to minimize the weighted sum of EOL extensions. The weight a_w for each wire segment w is assigned based on the timing slack of the net.

Table 1: Description of notations

Term	Meaning
a_w	weighting factor for wire segment w
l_w	x-coordinate of the left boundary of wire segment w
r_w	x-coordinate of the right boundary of wire segment w
cl_w	the index of the left cut of wire segment w
cr_w	the index of the right cut of wire segment w
x_i	x-coordinate of cut c_i
n_i^k	0-1 variable indicating that cut c_i is assigned to cut mask k
$m_{i,j}$	0-1 variable indicating that cut c_i and cut c_j form a bigger cut
$d_{i,j}$	0-1 variable indicating that cut c_i is on the left of cut c_j
G	a large positive constant

Constraints:

(i) Constraints for cut mask assignment.

$$\sum_{k=1}^{|K|} n_i^k = 1 \qquad \forall\, c_i \in C \tag{1}$$

Constraint (1) forces each cut to be assigned to exactly one of $|K|$ cut masks.

(ii) Constraints for cut pairs on the same track.

Two cuts c_i and c_j form a cut pair in set S_1 if (i) they are on the same track and (ii) the possible cut locations of c_i and c_j are within the minimum cut spacing of each other, considering their *relocation ranges*. The relocation range for c_i is the maximal contiguous set of grid points where c_i can be located. To achieve a legal solution for a cut pair, the two cuts should be (i) kept at least the minimum cut spacing apart from each other, as shown in Figure 2(a); (ii) merged into one cut, as shown in Figure 2(b); or (iii) assigned to different cut masks, as shown in Figure 2(c). For a cut pair c_i, c_j, a valid merging requires two cuts to be overlapped or abutted. Without loss of generality, we assume $x_i > x_j$.

$$x_i - x_j \geq 0 \qquad \forall (c_i, c_j) \in S_1 \tag{2}$$

$$x_i - x_j + G \times m_{i,j} + G \times (2 - n_i^k - n_j^k) \geq min_s \qquad \forall (c_i, c_j) \in S_1 \tag{3}$$

$$x_i - x_j - G \times (1 - m_{i,j}) + G \times (n_i^k - n_j^k) \leq min_w \qquad \forall (c_i, c_j) \in S_1 \tag{4}$$

$$x_i - x_j - G \times (1 - m_{i,j}) - G \times (n_i^k - n_j^k) \leq min_w \qquad \forall (c_i, c_j) \in S_1 \tag{5}$$

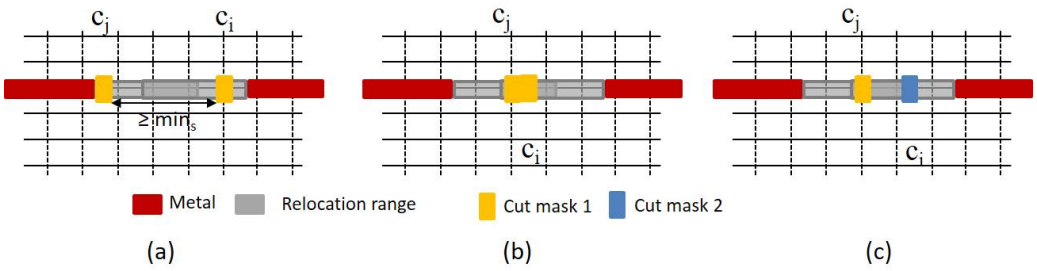

Figure 2: Cut pair c_i, c_j on the same track: (a) separating by minimum cut spacing; (b) merging to one cut; and (c) assigning to different cut masks.

Given two neighboring wire segments on the same track, if cut c_j is the right-end cut of the left wire segment and cut c_i is the left-end cut of the right wire segment, Constraint (2) keeps their relative cut locations in order, as shown

in Figure 2(a). The variable G is a large positive constant, and $m_{i,j}$ is a 0-1 variable indicating whether the two cuts are merged into a larger cut. When $m_{i,j} = 0$, Constraint (3) ensures that two cuts are either separated by at least the minimum cut spacing (see Figure 2(a)) or assigned to different cut masks (see Figure 2(c)). If the two cuts are merged, Constraints (4) and (5) ensure that they are assigned to the same cut mask, as shown in Figure 2(b).

(iii) Constraints for cut pairs on different tracks.

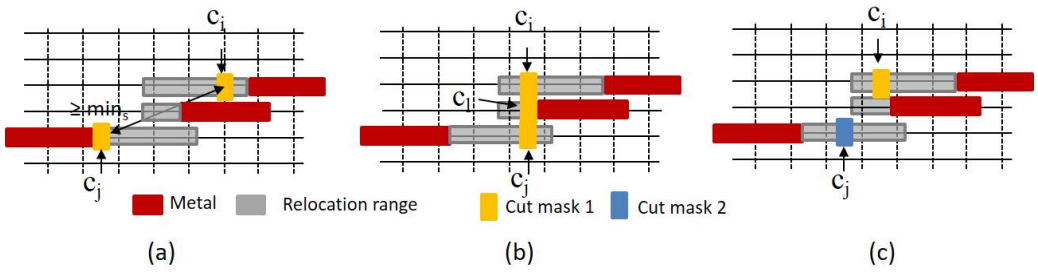

Figure 3: Cut pair c_i, c_j on two different tracks: (a) separating by minimum cut spacing; (b) merging by vertical alignment; and (c) assigning to different cut masks.

Two cuts c_i and c_j form a cut pair in set S_2 if they are on different tracks and their possible cut locations (i.e., relocation ranges) are within the minimum cut spacing. To legalize a given cut pair, the two cuts should be (i) kept at least the minimum cut spacing apart as shown in Figure 3(a); (ii) vertically aligned into a larger cut as shown in Figure 3(b); or (iii) assigned to different cut masks as shown in Figure 3(c).

$$x_i - x_j + G \times (d_{i,j} + m_{i,j} + (2 - n_i^k - n_j^k)) \geq min_s \qquad \forall (c_i, c_j) \in S_2 \tag{6}$$

$$x_j - x_i + G \times ((1 - d_{i,j}) + m_{i,j} + (2 - n_i^k - n_j^k)) \geq min_s \qquad \forall (c_i, c_j) \in S_2 \tag{7}$$

$$x_i - x_j + G \times (1 - m_{i,j}) + G \times (2 - n_i^k - n_j^k) \geq 0 \qquad \forall (c_i, c_j) \in S_2 \tag{8}$$

$$x_i - x_j - G \times (1 - m_{i,j}) - G \times (2 - n_i^k - n_j^k) \leq 0 \qquad \forall (c_i, c_j) \in S_2 \tag{9}$$

Indicator $d_{i,j}$ is a 0-1 variable indicating whether cut c_i is on the left side of cut c_j. Specifically, $d_{i,j} = 1$ indicates cut c_i is to the left of cut c_j. Indicator $m_{i,j}$ is a 0-1 variable indicating whether the two cuts c_i and c_j are vertically aligned and merged into a larger cut. Since we do not know the cut location in advance, for two vertically overlapped relocation ranges, either cut may be on the left side of the other. Similar to Constraint (3), when $m_{i,j} = 0$, Constraints (6) and (7) force the two cuts to be separated by at least the minimum cut spacing or assigned to different cut masks. Again, G is a large positive constant. If $m_{i,j} = 1$, Constraints (8) and (9) align the cuts c_i and c_j when they are assigned to the same cut mask. The vertical alignment requires that all aligned cuts share the same x-coordinate on contiguous tracks. Special consideration must be taken for the vertical alignment of cuts on multiple (i.e., ≥ 3) tracks. For two cuts c_i and c_j on two non-adjacent tracks, and a cut c_l on the track between the tracks of c_i and c_j, Constraints (10) – (13) ensure the vertical alignment between c_i and c_l if they are on the same cut mask. We enforce similar constraints between c_j and c_l. Figure 3(b) shows the result when three cuts are vertically aligned. Note that we do not allow vertical alignment if there is no available intersection with relocation ranges on intervening tracks.

$$x_l - x_i + G \times (1 - m_i^j) - G \times (n_i^k - n_l^k) \geq 0 \tag{10}$$

$$x_l - x_i + G \times (1 - m_i^j) + G \times (n_i^k - n_l^k) \geq 0 \tag{11}$$

$$x_l - x_i - G \times (1 - m_i^j) - G \times (n_i^k - n_l^k) \leq 0 \tag{12}$$

$$x_l - x_i - G \times (1 - m_i^j) + G \times (n_i^k - n_l^k) \leq 0 \tag{13}$$

Determining weights for EOL extensions of routing segments. For our optimization to be timing-aware, we must capture the timing impact of EOL extensions. A small amount of EOL extensions on the most timing-critical path may degrade the timing and result in an increase of the design's clock period (thus, reducing the maximum clock frequency of the design). On the other hand, even a large amount of EOL extensions on a non-critical path may not cause any degradation of the clock period. We model the timing criticality of possible EOL extensions by assigning weights that are derived from timing slacks computed by *static timing analysis*, e.g., using the *Synopsys PrimeTime* tool. Timing slack of a given net is used to determine the criticality of every wire segment of that net. For each net, we first obtain the timing slack of the most critical path passing through the net. Since the timing slack is defined per timing path, we distribute the timing slack among nets of the path based on *stage delays* (a *stage* consists of a logic gate or primary input of the design, along with its driven net). For example, given a timing path of two stages with a path timing slack of $+50ps$, if the first and second stage delays are $200ps$ and $300ps$, respectively, we assign $+20ps$ (e.g., $50 \times 200/(200+300)$) and $+30ps$ of timing slack to the nets of the first and second stages, respectively.

We then classify all wire segments into two groups, based on the calculated net slack values. The first group includes all wire segments of clock nets and of nets that have negative net slacks. All other wire segments are included in the second group. We assign a higher weight to segments in the first group.[‡] By minimizing the weighted sum of EOL extensions, our optimization will avoid EOL extensions on wire segments with higher weights (i.e., on timing-critical nets).

Analysis of the number of variables and constraints. Given a set of cuts C for wire segments W, and $|K|$ cut masks, we obtain sets of cut pairs S_1, S_2. The number of variables and constraints are as follows.

- The number of variables m is $|S_1| + |S_2|$.
- The number of variables d is $|S_2|$.
- The number of variables n is $|C| \cdot |K|$.
- The number of variables x is $|C|$.
- The number of Constraints (1) is $|C|$.
- The number of Constraints (2) – (5) is $|S_1|$.
- The number of Constraints (6) – (9) is $|S_2|$.
- The number of Constraints (10) – (13) is $F \cdot |S_2|$, where F is a constant.

3.2. Timing- and Density-Aware Post-ILP Optimization

We now explain our timing- and density-aware post-ILP optimization flow, which starts from the ILP solution achieved as described in the preceding subsection. Given layer t with all cuts assigned to cut masks, we iteratively consider regions above and below given routing segments – so as to remove dummy fills by enlarging or inserting cuts using all available cut masks – until the total metal density of the layer t reaches the target minimum metal density constraint. To maintain awareness of timing, we process all routing segments in the ascending order of their net slack, as discussed above. Our flow also attempts to maintain mask density uniformity across all cut masks.

Algorithm 1 describes the detail of our post-ILP optimization flow. Our flow optimizes layer by layer from the output solution of ILP-based optimization. The inputs are the output layer t from ILP-based optimization, target minimum metal density ρ_{min}, set of cut masks (colors) K and minimum cut spacing min_s. The output is the optimized layer t_{opt} with dummy fills. Lines 1-2 calculate the current metal density d_m of layer t and mask density P_k for cut mask k of layer t. Lines 3-4 collect all routing segments W in layer t and sort all routing segments $w \in W$ in the ascending order of their net slack. We then set the Δ, which is used to determine the target region to apply cuts, as one (Line 5). Lines 6-16 iteratively add cuts on all available cut masks until $\rho_m \leq \rho_{min}$. For each routing segment w (Line 7), we check the upper (left) and lower (right) horizontal (vertical) tracks $track_{cur}$ (Line 8), which are exactly Δ tracks apart from the horizontal (vertical) track of the routing segment w.

[‡]We set a weight of $w = 2$ for segments in the first group, and weight $w = 1$ for segments in the second group.

Algorithm 1 Timing- and density-aware post-ILP optimization

Procedure *postILP_opt*()
Input: Layer t after ILP optimization, target minimum metal density ρ_{min}, set of cut masks (colors) K, minimum cut spacing min_s
Output: Layer t_{opt} with dummy fills

1: $\rho_m \leftarrow$ metal density of layer t;
2: $P_k \leftarrow$ mask density of cut mask $k \in K$ in layer t;
3: $W \leftarrow$ set of routing segments in layer t;
4: Sort all $w \in W$ in ascending order of their net slack;
5: $\Delta \leftarrow 1$;
6: **while** $\rho_m \geq \rho_{min}$ **do**
7: **for all** $w \in W$ **do**
8: **for all** $track_{cur} \in \{track_r + \Delta, ..., track_r - \Delta\}$ **do**
9: $v \leftarrow defineTargetRegion(w, track_{cur})$;
10: $Q \leftarrow enumCandidateCuts(v, t, min_s)$;
11: $t \leftarrow selectCuts(t, Q, P_k)$;
12: $updateDensity(\rho_m, P_k, t)$;
13: **end for**
14: **end for**
15: $\Delta \leftarrow \Delta + 1$;
16: **end while**
17: $t_{opt} \leftarrow dummyFillInsertion(t)$;
18: Return t_{opt};

The function $defineTargetRegion(w, track)$ defines a target region v to be cut as shown in Figure 4(a). For this target region v, the function *enumCandidateCuts*() then enumerates all possible sets of candidate cuts on each cut mask k. Minimum cut spacing min_s (Line 10) is considered in this step. We only allow rectangular shapes on cut masks for better fidelity of metals. To avoid forming non-rectangular shapes, we consider existing cuts on the neighboring tracks of the target region v. Figure 4(b) shows an example of all candidate cuts on each cut mask for the target region. Neighboring regions of the target region v are checked so that rectangular shapes are always preserved when enlarging or inserting cuts.

After obtaining the set of candidate cuts Q on each cut mask k, to account for the mask density uniformity, we first select the set on a cut mask with the least mask density. We then pick the set that has the minimum mask density among the remaining cut masks, and cover the target region which is not covered by previous cuts. Figure 4(c) shows an example solution with the assumption of $\rho_3 \leq \rho_2 \leq \rho_1$. Finally, based on the optimized cut mask solution, we obtain the actual layout pattern with EOL extensions and dummy fills for layer t (Line 17).

3.3. Overall Flow

Our overall flow is shown in Figure 5. In the flow, we perform two steps: (i) ILP-based cut mask optimization and (ii) timing/density-aware post-ILP optimization. To achieve a scalable optimization for full-chip layouts, we use a partitioning-based, distributable optimization strategy. Namely, to overcome the poor scalability of ILP, we split the layout into many *clips* and run the ILP-based optimization for each clip in parallel. (A *clip* is simply a rectangular piece of the chip layout.) The typical clip size is $3\mu m$ by $3\mu m$.[§] Our second step of post-ILP optimization to improve critical path performance removes dummy fills with consideration of metal density and mask density constraints. This step is achieved using an efficient heuristic, with no need for distributed implementation.

Handling cuts at boundaries with multiple iterations. One drawback of partitioning-based optimization is that clips can interfere with each other so that their solutions may not be compatible with each other when stitched together within the entire chip. To avoid such situations, we perform several iterations of the optimization so that all cuts are processed

[§]We use a foundry N28 BEOL stack with 2.5× scaled N7 library cells. Therefore, the clip size in N7 will be $1.2\mu m$ by $1.2\mu m$.

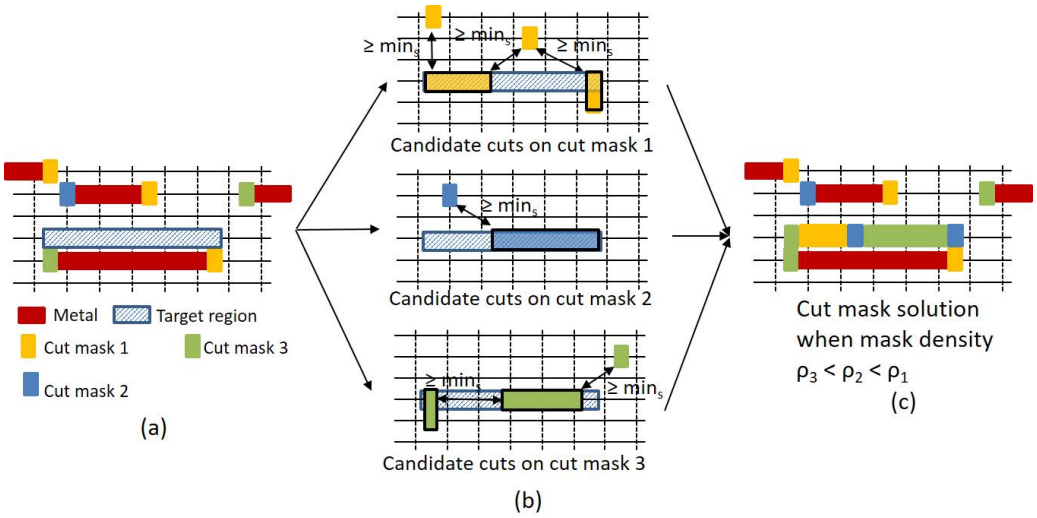

Figure 4: Our post-ILP optimization: (a) defining a target region for a timing critical segment; (b) enumerating candidate cuts on each cut mask covering the target region, considering minimum cut spacing min_s; and (c) applying cuts based on mask density.

without conflict. Figure 6 shows how we partition the layout in each of three iterations that comprise our partitioning-based optimization.

In the first iteration, we optimize cuts within each clip, without considering boundaries between clips, as shown in Figure 6(a). In the second iteration, we optimize and solve conflicts near the horizontal boundaries between vertically adjacent clips of Figure 6(a), as shown in Figure 6(b). In this second iteration, we adjust the height of a clip to be four times the minimum cut spacing. In each clip, we only optimize for regions within minimum cut spacing of horizontal boundaries and keep the solutions obtained in the first step for the other regions in the clip. In this way, we can solve all conflicts on horizontal boundaries. Similarly, in the third iteration, we optimize clips that cover vertical boundaries between pairs of horizontally adjacent clips in Figure 6(a). The width of clips in this iteration is determined similarly to the height of clips in the second iteration; see Figure 6(c). After completing the three iterations, we will have covered all cuts without inducing conflict between the clip solutions.

4. EXPERIMENTAL SETUP AND RESULTS

In this section, we present our experimental setup and results. We experiment on the impact of number of cut masks, minimum metal density, minimum cut spacing and EOL extensions. These experiments show the effectiveness of our optimization and tradeoffs between performance and cost.

4.1. Experimental Setup

The program is written in C++ with *OpenAccess 2.6*[13] API to support DEF/LEF[12] and handle routing segments. We use IBM CPLEX[11] as the ILP solver. Parallel optimization is enabled by OpenMP[15] API. We perform experiments with 40 threads on a 2.6*GHz* Intel Xeon E5-2690 dual-CPU server. Reported runtimes are "wall clock" time between start and termination of each given experiment.

We evaluate our approach using an encryption core (AES) and a media processing core (JPEG) from OpenCores,[14] as well as an ARM Cortex M0 design without memories. We synthesize the designs with *Synopsys Design Compiler H-2013.03-SP3*[16] from RTL netlists. We then perform placement and routing with *Cadence Encounter Digital Implementation System v14.1*,[10] using an abstracted N7 library from a leading IP provider.

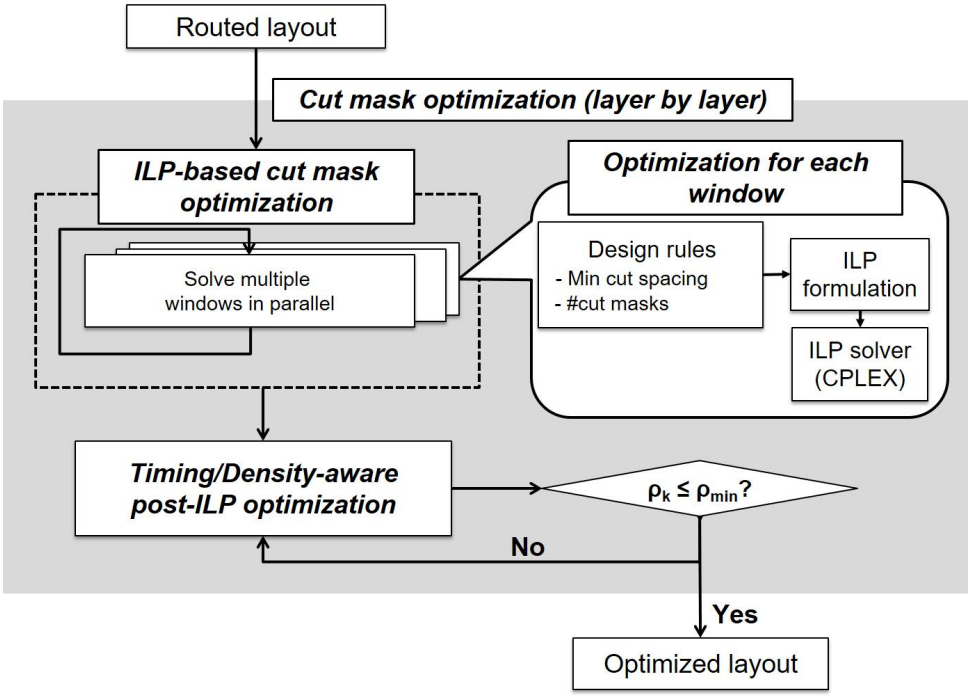

Figure 5: Overall flow of cut mask optimization.

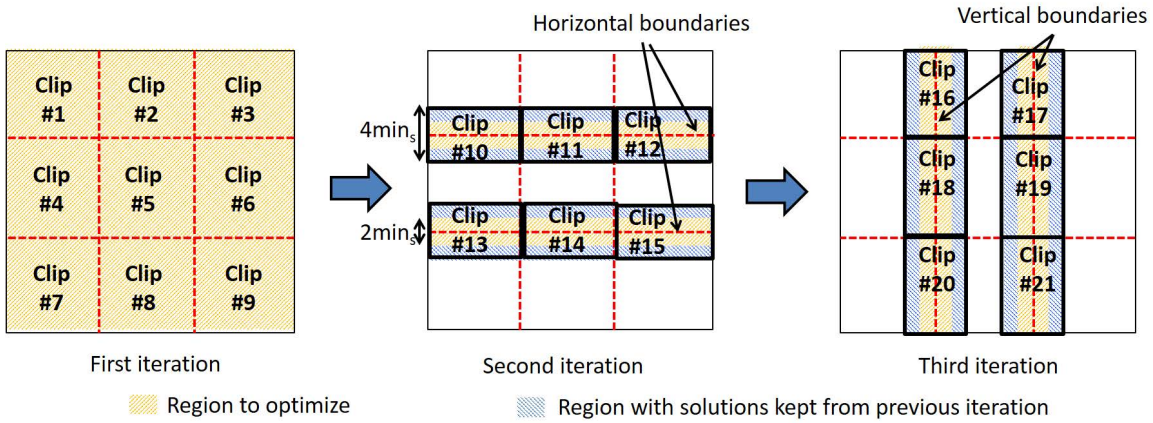

Figure 6: Partition for each iteration to handle cuts at boundaries.

Table 2: Testcases.

Node	Design	#cells	#nets	Area (μm)	Util. (%)	#stages on critical path	#segments				
							M2	M3	M4	M5	M6
N7	ARM Cortex M0	8994	9048	8272	81	49	33311	21359	10606	6306	2595
	AES	13340	13602	9807	86	6	46034	29552	16935	10453	4939
	JPEG	54215	49124	54238	84	28	168672	101771	37497	17763	4106
N5	ARM Cortex M0	8386	8440	7778	76	57	31881	20934	10534	6194	2547
	AES	11650	11912	8596	81	11	42819	28176	16223	10480	4960
	JPEG	57396	50368	57419	76	52	177943	108715	38220	17871	3429

Since our N7 technology is missing detailed BEOL stack information which is necessary for design enablement, we scale up the N7 library cells' dimensions to use an N28 BEOL stack, following the methodology described in Han et al.[5¶]. The methodology described by Chan et al.[1] is used to derive the missing resistance (R) and capacitance (C) information for N7 BEOL from original N28 wire RC values. Here, R (C) is defined as per unit-length resistance (capacitance) in a specific foundry node. We scale the N28 wire R by 13× to derive the N7 wire R, accounting for the rapid increase of resistivity in advanced nodes[‖]. N28 wire C is scaled by 0.4× to derive the N7 wire C considering geometric scaling. We also project to N5 foundry technology by scaling wire R and C further (e.g., 22× and 0.28× for R and C from N28 BEOL, respectively) and derating standard cells' delay based on 2015 ITRS models.[6] The delay and transition time of standard cells are scaled by 0.75× according to the ratio of I/CV parameters of N7 and N5. The gate capacitances of standard cells are scaled by 0.86×. Table 2 summarizes key parameters of our testcases.

In all our experiments, we derive minimum cut spacings for N7 and N5 foundry nodes according to the 2015 ITRS Lithography Chapter.[6] Based on the ITRS discussion, the 2D lithography pitch cliff is approximately 110*nm*, which corresponds to cut pitch. M2 pitches of foundry N7 and N5 nodes are 36*nm* and 24*nm*, respectively. Since our enablement uses a foundry N28 BEOL stack, we use multiples of M2 pitch as minimum cut spacing, which are derived from metal and cut pitch numbers. We use four M2 pitches and five M2 pitches as the minimum cut spacing values in the N7 and N5 nodes, respectively. Minimum cut spacing is checked based on center-to-center Euclidean distance between cuts. Figure 7 illustrates forbidden locations caused by an existing cut on the cut mask.

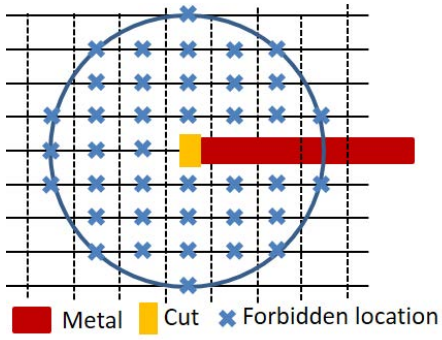

Figure 7: Minimum cut spacing and forbidden locations for the same color cuts.

Our BEOL stack consists of six layers (i.e., M1 – M6) of 1× minimum M2 pitch and two layers (i.e., M7 – M8) of 2× minimum M2 pitch. We assume that SAMP will only be applied to the six layers with 1× minimum M2 pitch. Therefore, we vary the number of cut masks, minimum cut spacing and target minimum metal density only on the layers M2 – M6 to study the impact of each parameter. We report worst negative slack (WNS) and total wire capacitance using *Cadence Encounter Digital Implementation System v14.1* for each of the N7 and N5 implementations (Note that smaller (e.g., more negative) values of WNS are worse because "slack" corresponds to "timing safety".). As a calibration, the typical FO4 buffer delays in N7 and N5 are 23*ps* and 18*ps*, respectively. Our analysis is performed with coupling capacitance and signal integrity (i.e., crosstalk-induced delay impact analysis) options in the timing analysis.

4.2. Experimental Results

We now report our experimental results, including the impact of number of cut masks, minimum metal density, minimum cut spacing and EOL extensions. Our experiments demonstrate the effectiveness of our optimization as well as substantial available tradeoffs between performance and cost. Figure 8 visualizes a fragment of layout of the M2 layer with optimized EOL extensions and dummy fills, for the N7 Cortex M0 testcase, using four cut masks.

[¶]Of course, the foundry N28 BEOL stack that we use may not embody new layout ground rules that govern detailed routing in N7 and N5.

[‖]$13 \approx (2.4^4) \times 0.4$. The resistance (R) increases by 2.4× per node for four nodes starting from N28 BEOL, with a geometric scaling factor of 0.4 from N28 BEOL to N7 BEOL.

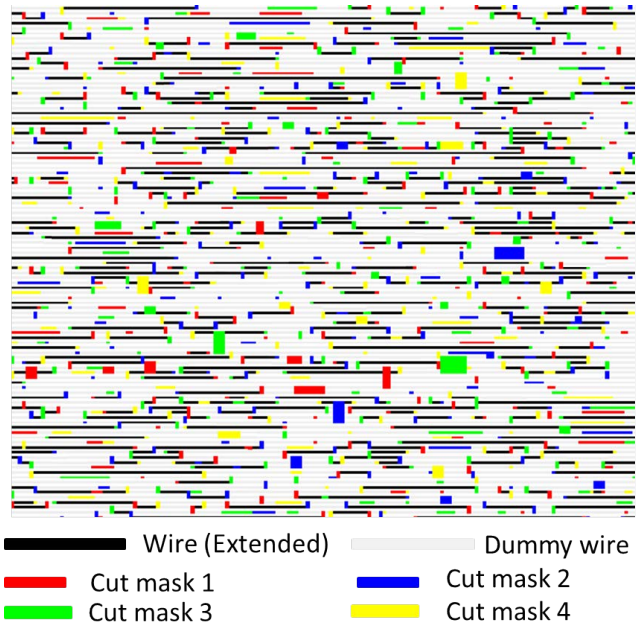

Figure 8: An example layout of M2 with EOL extensions and dummy fills for Cortex M0.

Impact of number of cut masks. Table 3 shows results with various number of cut masks for each layer (i.e., options C1 – C12). We use minimum cut spacing as four M2 pitches with N7 technology and target minimum metal density as 40%. We assume that in the SAMP process, the width and spacing of metal are both equal to the half-pitch, so that maximum track occupancy gives 50% metal density. For each option, we report the WNS, total wire capacitance, sum of EOL extensions, the number (percentage) of infeasible clips and runtime. An infeasible clip means that the ILP solver cannot find a feasible solution for the ILP instance corresponding to the clip for the given numbers of cut masks and minimum cut spacing. Regardless of the testcase, infeasible clips exist for all options C1 to C5, implying that option C6 is the set of minimum numbers of cut masks that ensures solution feasibility in all three of our testcases. Further, by comparing options C6, C11 and C12, we observe that using two more than the minimum number of cut masks in each of the layers has little effect on timing. We also note that the Cortex M0 testcase has a larger WNS variation than the AES testcase among different options, even though AES always has larger EOL extensions than Cortex M0. This is because Cortex M0 has more stages on its critical timing path than AES, and so the cumulative timing impact (over the entire critical path) seen in Cortex M0 is larger. Runtimes for our optimization are larger for options with numbers of cut masks similar to those in option C6 (the set of minimum numbers of cut masks). Also, among the three testcases, runtime increases roughly linearly for each option according to the number of segments (see Table 2).

Impact of minimum metal density. Table 4 shows the results with various minimum metal density constraints (i.e., 40%, 42.5%, 45%). We set minimum cut spacing as four M2 pitches and use option C6 with the minimum number of cut masks for a feasible solution. The WNS improvement is up to $14ps$ by decreasing the target metal density from 45% to 40% among three testcases. We also observe that runtime does not change among different target metal densities, which means that runtime of our post-ILP optimization is negligible compared to that of the ILP-based cut mask optimization step.

Impact of minimum cut spacing. Table 5 shows results with different minimum cut spacings for N7 and N5. Minimum metal density of 40% is enforced. For each design in each node, we first find the option that has the minimum number of cut masks per each layer. We then investigate the impacts on timing, total wire capacitance and total EOL extensions when we add one or two more masks for each layer. When we compare the results for N7 and N5, we observe that N5 is more sensitive to the number of cut masks. For example, AES for N7 shows $1ps$ difference in WNS between the options (3,2,2,2,2) and (5,4,4,4,4), but AES for N5 shows $11ps$ difference. This is because wire delay is more dominant than gate delay in N5 compared to N7. Also, going from N7 to N5, the increase of per unit-length wire resistance is greater than the decrease in per unit-length wire capacitance.

Table 3: Results for different numbers of cut masks, per layer (node = N7, spacing = 4, density = 40%).

Design	Option	#cut masks (M2,M3,M4,M5,M6)	WNS (ns)	Cap. (fF)	EOL Ext. (μm)	#infeasible clips (%)	Time (s)
Cortex M0	C1	2,1,1,1,1	-0.099	9379	9864	1199 (20.7)	1291
	C2	3,1,1,1,1	-0.09	9124	6908	1119 (19.4)	2084
	C3	3,2,1,1,1	-0.083	8920	5167	420 (7.3)	2322
	C4	3,2,2,1,1	-0.074	8747	3414	108 (1.9)	2306
	C5	3,2,2,2,1	-0.068	8667	2637	28 (0.5)	2302
	C6	3,2,2,2,2	-0.062	8601	2416	0 (0.0)	2299
	C7	4,2,2,2,2	-0.058	8527	1382	0 (0.0)	2281
	C8	4,3,2,2,2	-0.056	8496	610	0 (0.0)	2255
	C9	4,3,3,2,2	-0.049	8500	413	0 (0.0)	2258
	C10	4,3,3,3,2	-0.049	8499	365	0 (0.0)	2260
	C11	4,3,3,3,3	-0.048	8477	358	0 (0.0)	2261
	C12	10,10,10,10,10	-0.047	8412	0	0 (0.0)	171
AES	C1	2,1,1,1,1	-0.033	12396	14370	1807 (31.2)	2106
	C2	3,1,1,1,1	-0.037	11991	10038	1667 (28.8)	3279
	C3	3,2,1,1,1	-0.031	11828	8848	834 (14.4)	3898
	C4	3,2,2,1,1	-0.017	11598	6829	288 (5.0)	3925
	C5	3,2,2,2,1	-0.015	11473	5436	83 (1.4)	3921
	C6	3,2,2,2,2	-0.015	11391	4866	0 (0.0)	3918
	C7	4,2,2,2,2	-0.018	11226	2826	0 (0.0)	3755
	C8	4,3,2,2,2	-0.016	11154	1496	0 (0.0)	3798
	C9	4,3,3,2,2	-0.014	11146	1004	0 (0.0)	3854
	C10	4,3,3,3,2	-0.014	11137	868	0 (0.0)	3855
	C11	4,3,3,3,3	-0.014	11117	850	0 (0.0)	3855
	C12	10,10,10,10,10	-0.013	11148	0	0 (0.0)	344
JPEG	C1	2,1,1,1,1	-0.04	37225	42178	4152 (12.8)	7344
	C2	3,1,1,1,1	-0.024	36023	28280	3858 (11.9)	10574
	C3	3,2,1,1,1	-0.022	34913	15052	926 (2.9)	11010
	C4	3,2,2,1,1	0.007	34360	9657	151 (0.5)	11036
	C5	3,2,2,2,1	0.015	34166	8243	22 (0.1)	11052
	C6	3,2,2,2,2	0.018	33992	8008	0 (0.0)	11058
	C7	4,2,2,2,2	0.021	33758	3940	0 (0.0)	10618
	C8	4,3,2,2,2	0.022	33680	1479	0 (0.0)	10686
	C9	4,3,3,2,2	0.026	33649	1127	0 (0.0)	10698
	C10	4,3,3,3,2	0.029	33650	1062	0 (0.0)	10699
	C11	4,3,3,3,3	0.03	33557	1059	0 (0.0)	10703
	C12	10,10,10,10,10	0.033	33504	0	0 (0.0)	2792

Table 4: Results for different metal density targets with minimum (3,2,2,2,2) numbers of colors per layer (node = N7).

Design	Density (%)	WNS (ns)	Cap. (fF)	EOL Ext. (μm)	#infeasible clips (%)	Time (s)
Cortex M0	40	-0.062	8601	2416	0 (0.0)	2299
	42.5	-0.068	8741	2416	0 (0.0)	2299
	45	-0.076	8801	2459	0 (0.0)	2299
AES	40	-0.015	11391	4866	0 (0.0)	3918
	42.5	-0.017	11560	4838	0 (0.0)	3918
	45	-0.026	11749	4854	0 (0.0)	3918
JPEG	40	0.018	33992	8008	0 (0.0)	11058
	42.5	0.013	34648	8105	0 (0.0)	11058
	45	0.009	35277	8025	0 (0.0)	11058

Impact of EOL extensions. Our next experiment performs ILP-based cut mask optimization by itself, without any post-ILP optimization, to highlight the impact of EOL extensions. We compare the *best* unit-size cut solution against the (near-) *worst* unit-size cut solution. To find the (near-) worst solution, we maximize the weighted sum of extensions, instead of minimizing this weighted sum. The maximum length of each relocation range is restricted to 30 M2 pitches according to our clip size. Therefore, the (near-) worst solution from our ILP-based cut mask optimization may not be the worst solution over the solution space since a wire segment cannot be extended beyond its clip boundaries. However, this (near-) worst solution is bad enough to demonstrate a strong impact of EOL extensions. We conduct experiments in both N7 and N5 nodes. For N7, we use four M2 pitches for minimum cut spacing, option (3,2,2,2,2) and 40% for target metal density. To isolate the timing impact of EOL extensions from dummy fills, the best (BEST) and the worst (WORST) solutions have only EOL extensions without dummy fills. We compare BEST and WORST with the original target layout (ORIG)

Table 5: Results for different cut spacing rules (density = 40%).

Design	Node	Min cut spacing (M2 pitches)	#cut masks (M2,M3,M4,M5,M6)	WNS (ns)	Cap. (fF)	EOL Ext. (μm)	#infeasible clips (%)	Time (s)
Cortex M0	N7	4	3,2,2,2,2	-0.062	8601	2416	0 (0.0)	2299
			4,3,3,3,3	-0.048	8477	358	0 (0.0)	2261
			5,4,4,4,4	-0.048	8479	37	0 (0.0)	1281
	N5	5	4,3,3,2,2	-0.054	6376	2129	0 (0.0)	3922
			5,4,4,3,3	-0.043	6280	539	0 (0.0)	3459
			6,5,5,4,4	-0.037	6250	96	0 (0.0)	1900
AES	N7	4	3,2,2,2,2	-0.015	11391	4866	0 (0.0)	3918
			4,3,3,3,3	-0.014	11117	850	0 (0.0)	3855
			5,4,4,4,4	-0.014	11110	84	0 (0.0)	2265
	N5	5	5,3,3,2,2	-0.079	8401	2792	0 (0.0)	5685
			6,4,4,3,3	-0.077	8242	504	0 (0.0)	4116
			7,5,5,4,4	-0.068	8222	17	0 (0.0)	1809
JPEG	N7	4	3,2,2,2,2	0.018	33992	8008	0 (0.0)	11058
			4,3,3,3,3	0.026	33649	1127	0 (0.0)	10698
			5,4,4,4,4	0.031	33526	93	0 (0.0)	5711
	N5	5	4,3,3,2,2	0.035	26396	5414	0 (0.0)	16048
			5,4,4,3,3	0.045	26156	1111	0 (0.0)	14390
			6,5,5,4,4	0.055	26092	193	0 (0.0)	8487

to see the pure impact of EOL extensions. We also add a comparison to our final layout (BEST + POST-ILP), which accounts for the impact of both EOL extensions and dummy fills. Table 6 shows the results for WORST, BEST, ORIG, and BEST + POST-ILP. For N5, we use five M2 pitches for minimum cut spacing, and the option with minimum number of cut masks for each design as determined in previous experiments (see Table 5). The target minimum metal density is set to be 40%. Table 7 shows the results for N5.

Table 6: Comparison of BEST vs. WORST unit-size cut solutions (node = N7, spacing = 4, density = 40%).

Design	Status	#cut masks (M2,M3,M4,M5,M6)	WNS (ns)	Cap. (fF)	EOL Ext. (μm)	#infeasible clips (%)	Time (s)
Cortex M0	ORIG	3,2,2,2,2	-0.006	7155	N/A	N/A	N/A
	BEST		-0.013	7286	2416	0 (0.0)	2251
	WORST		-0.192	12637	68130	0 (0.0)	228
	BEST + POST-ILP		-0.062	8601	2416	0 (0.0)	2251
AES	ORIG	3,2,2,2,2	0.018	9642	N/A	N/A	N/A
	BEST		0.017	9950	4866	0 (0.0)	4118
	WORST		-0.058	16663	91618	0 (0.0)	298
	BEST + POST-ILP		-0.015	11391	4866	0 (0.0)	4118
JPEG	ORIG	3,2,2,2,2	0.079	26557	N/A	N/A	N/A
	BEST		0.076	26986	8008	0 (0.0)	10758
	WORST		-0.149	51354	319140	0 (0.0)	3126
	BEST + POST-ILP		0.018	33992	8008	0 (0.0)	10758

By comparing BEST and WORST timing to ORIG timing, we observe that among three testcases, WORST EOL extensions can degrade WNS by up to $228ps$. For all three designs, the average gap between BEST and WORST timing in N5 is $90ps$ larger than in N7. Compared to ORIG, our BEST + POST-ILP optimization achieves an average timing degradation of only $50ps$ and $67ps$ for N7 and N5, respectively, including the impact of dummy fills. When we compare ORIG, BEST and BEST + POST-ILP solutions, it is apparent that most of the WNS degradation is caused by dummy fills.

5. CONCLUSION

In this work, we have studied the co-optimization of cut mask layout, dummy fill, and design timing for sub-$14nm$ BEOL design. We propose an ILP-based cut mask optimizer and a heuristic for post-ILP optimization. Our cut mask optimization flow for varying contexts (e.g., number of cut masks, target minimum metal density, minimum cut spacing, EOL extensions) indicate that there can be significant potential tradeoffs of performance and cost. Our ongoing work addresses such topics as: (i) improved timing-aware weight assignment in ILP; (ii) implementation of an ECO routing flow for infeasible routing clips to reduce the mask cost; (iii) comprehension of the difference between coupling to floating dummy fills and coupling to EOL extensions; and (iv) co-optimization of detailed routing and cut mask solutions.

Table 7: Comparison of BEST vs. WORST unit-size cut solutions (node = N5, spacing = 5, density = 40%).

Design	Status	#cut masks (M2,M3,M4,M5,M6)	WNS (ns)	Cap. (fF)	EOL Ext. (μm)	#infeasible clips (%)	Time (s)
Cortex M0	ORIG	4,3,3,2,2	-0.001	5801	N/A	N/A	N/A
	BEST		-0.008	5905	2129	0 (0.0)	3940
	WORST		-0.383	11288	67966	0 (0.0)	254
	BEST + POST-ILP		-0.054	6376	2129	0 (0.0)	3940
AES	ORIG	5,3,3,2,2	0.006	8073	N/A	N/A	N/A
	BEST		0.0056	8192	2792	0 (0.0)	5710
	WORST		-0.125	14942	89445	0 (0.0)	306
	BEST + POST-ILP		-0.079	8401	2792	0 (0.0)	5710
JPEG	ORIG	4,3,3,2,2	0.098	26557	N/A	N/A	N/A
	BEST		0.094	26986	5414	0 (0.0)	16048
	WORST		-0.101	51354	342031	0 (0.0)	3859
	BEST + POST-ILP		0.035	33992	5414	0 (0.0)	16048

REFERENCES

1. T.-B. Chan, A. B. Kahng and J. Li, "Toward Quantifying the IC Design Value of Interconnect Technology Improvements", *Proc. SLIP*, 2013.
2. Y. Ding, C. Chu and W. K. Mak, "Throughput Optimization for SADP and E-beam Based Manufacturing of 1D Layout", *Proc. DAC*, 2014, pp. 1-6.
3. Y. Du, H. Zhang, M. D. F. Wong and K.-Y. Chao, "Hybrid Lithography Optimization with E-beam and Immersion Processes for 16nm 1D Gridded Design", *Proc. ASPDAC*, 2012, pp. 707-712.
4. W. Gillijns, S. M. Y. Sherazi, D. Trivkovic, B. Chava, B. Vandewalle, V. Gerousis, P. Raghavan, J. Ryckaert, K. Mercha, D. Verkest, G. McIntyre and K. Ronse, "Impact of a SADP Flow on the Design and Process for N10/N7 Metal Layers", *SPIE Advanced Lithography*, 2015, pp. 942709-942709.
5. K. Han, A. B. Kahng and H. Lee, "Evaluation of BEOL Design Rule Impacts Using an Optimal ILP-Based Detailed Router", *Proc. DAC*, 2015.
6. *International Technology Roadmap for Semiconductors*. `http://www.itrs2.net/`
7. D. K. Lam, E. D. Liu, M. C. Smayling and T. Prescop, "E-beam to Complement Optical Lithography for 1D Layouts", *SPIE Advanced Lithography*, 2011, pp. 797011-797011.
8. S. Owa, S. Wakamoto, M. Murayama, H. Yaegashi and K. Oyama, "Immersion Lithography Extension to Sub-10nm Nodes with Multiple Patterning", *SPIE Advanced Lithography*, 2014, pp. 90520O-90520O.
9. H. Zhang, Y. Du, M. D. F. Wong and K.-Y. Chao, "Lithography-Aware Layout Modification Considering Performance Impact", *Proc. ISQED*, 2011, pp. 1-5.
10. Cadence Encounter Digital Implementation System User Guide. `http://www.cadence.com`
11. IBM ILOG CPLEX. `www.ilog.com/products/cplex/`
12. LEF DEF reference 5.7. `http://www.si2.org/openeda.si2.org/projects/lefdef`
13. Si2 OpenAccess. `http://www.si2.org/?page=69`
14. OpenCores: Open Source IP-Cores. `http://www.opencores.org`
15. OpenMP Architecture Review Board, "OpenMP Application Program Interface, Version 3.1".
16. Synopsys Design Compiler User Guide. `http://www.synopsys.com`
17. Synopsys PrimeTime User Guide. `http://www.synopsys.com`