Routability-Aware Placement for Advanced FinFET Mixed-Signal Circuits using Satisfiability Modulo Theories

Hao Chen[1], Walker J. Turner[2], David Z. Pan[1], and Haoxing Ren[2]
[1]ECE Department, The University of Texas at Austin, Austin, TX, USA
[2]NVIDIA Corporation, USA
haoc@utexas.edu, wturner@nvidia.com, dpan@ece.utexas.edu, haoxingr@nvidia.com

***Abstract*—Due to the increasingly complex design rules and geometric layout constraints within advanced FinFET nodes, automated placement of full-custom analog/mixed-signal (AMS) designs has become increasingly challenging. Compared with traditional planar nodes, AMS circuit layout is dramatically different for FinFET technologies due to strict design rules and grid-based restrictions for both placement and routing. This limits previous analog placement approaches in effectively handling all of the new constraints while adhering to the new layout style. Additionally, limited work has demonstrated effective routability modeling, which is crucial for successful routing. This paper presents a robust analog placement framework using satisfiability modulo theories (SMT) for efficient constraint handling and routability modeling. Experimental results based on industrial designs show the effectiveness of the proposed framework in optimizing placement metrics while satisfying the specified constraints.**

I. Introduction

As processor technology continues to evolve into the deep-nanometer era, full-custom analog/mixed-signal (AMS) integrated circuits (ICs) must be designed using fin-shaped field-effect transistors (FinFETs) to meet performance requirements. While these transistors enjoy reduced leakage currents and superior control of the electric-field within the device channel, the three-dimensional fin-shape structure imposes even more challenges to the already intricate AMS layout process.

Restricted by manufacturability issues, circuit layout in advanced FinFET nodes (e.g., N7, N5) enforce a row-based placement style for fin alignment [1]. As a result, AMS layout implementations rely heavily on *layout primitives*, the basic building block of an AMS circuit formed by a group of low-level devices (e.g., transistors, resistors) [2]. The use of primitive cells brings several benefits to the layout procedure. By forcing related primitives to have standard heights, the fin alignment problem, as well as numerous complicated design rules in the front-end-of-line (FEOL), can be handled directly. Moreover, specific hard-to-automate layout patterns (e.g., interdigitation) can be directly implemented by layout experts to ensure performance requirements for each primitive. In practice, *cell-based primitives* are adopted for better coverage of a wide range of AMS circuit classes and a higher degree of design freedom. Figure 1(a) shows partial layout patterns of an optimized primitive cell consisting of ten transistors.

To reduce design complexity without significantly sacrificing solution quality and flexibility, advanced FinFET AMS circuits adopt a *region-based layout methodology*, where primitive cells with similar properties (e.g., PMOS, NMOS, standard cells) are grouped and placed in the same *region*. As shown in Figure 1(b), primitives in the same region share the same cell height to meet the row-based placement requirement. Region-based layouts allow easy base-layer design rule checking (DRC) and enable making reusable stand-alone intellectual properties (IPs) as each

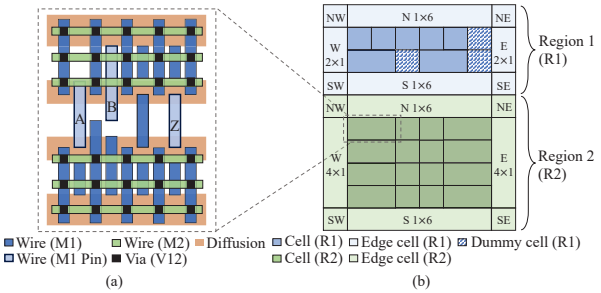


Fig. 1: Example of the region-based layout style with primitives. (a) Layout patterns of a primitive cell with 10 transistors. (b) A 2-region layout design.

region is self-contained. In real-world designs, there could be more than five regions in a building block.

As placement determines the overall wiring topology in the later routing stage, various geometrical constraints should be carefully considered to guarantee circuit robustness and the performance specification. Typical constraints include symmetry, proximity, and fixed-boundary constraints [3]. For region-based layouts, additional constraints exhibited in Figure 2 are introduced, including

- *Hierarchical Symmetry Constraints* that place a set of primitive cells symmetrically with respect to multiple joint axes simultaneously. Violating hierarchical symmetry constraints can result in degraded power-supply rejection ratio, mismatched differential signals, and higher offset voltages [4].
- *Array Constraints* that force an adjacent placement of some cells in the same region with optional layout patterns such as interdigitation, common-centroid, and central-symmetrical. One can further extend array constraints to a hierarchical structure containing an array of arrays.
- *Cluster Constraints* that place a set of primitive cells as close as possible. Note that a cluster constraint can apply to objects from different regions.
- *Extension Constraints* that reserve extra spaces around the target object (e.g., primitive, array, region) to be filled with some specified dummy cells. Extension constraints ensure spacing between adjacent cells to reduce electromigration concerns and extension of diffusion to minimize layout-dependent effects.

Routability awareness is also a crucial factor. As signal interconnections are increasingly complicated in advanced FinFET nodes, placement algorithms without routability consideration could lead to severe routing congestion, making them unroutable. Limited work has demonstrated an effective way to simultaneously consider all the aforementioned constraints and

978-3-9819263-6-1/DATE22/©2022 EDAA

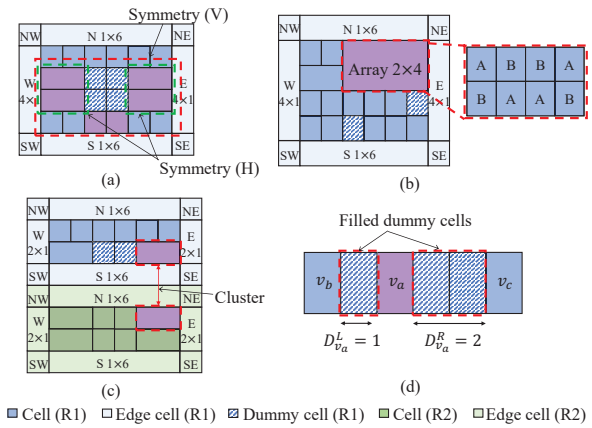


Fig. 2: Illutstrations of analog placement constraints: (a) Hierarchical symmetry constraint, (b) Array constraint, (c) Cluster constraint, and (d) Extension constraint.

the routability issue. Furthermore, directly extending previous methods for the region-based layout style would incur runtime overhead and solution quality degradation.

In this paper, we proposed a novel AMS placement framework targeting advanced FinFET designs. Translating the placement problem into satisfiability modulo theories (SMT) formulas, the proposed AMS placement framework is capable of handling the above geometrical constraints and routability considerations simultaneously. Highlights of this work are summarized as follows:

- We propose an SMT-based placement framework targeting a new region-based layout for advanced FinFET AMS designs with comprehensive constraint handling.
- We develop a window-based pin density checking SMT formulation for placement routability consideration. Without the checking scheme, the placement solutions may be unroutable due to heavy local routing congestion.
- We leverage incremental SMT solving techniques for efficient wirelength optimization, enabling the application of our framework to large-scale designs.
- Experimental results on advanced industrial designs demonstrate the effectiveness of the proposed placement framework in generating high-quality solutions while satisfying all the specified constraints.

II. Related Work

We classify previous AMS placement techniques into two paradigms: 1) stochastic methods and 2) analytical approaches.

Stochastic methods leverage probabilistic techniques (e.g., simulated annealing) to search solutions on topological representations (e.g., sequence pairs, B*-trees). These algorithms have been shown to be effective in handling various geometric constraints, including symmetry [5], common-centroid [6], regularity [7], proximity [8], pre-placed [9], fixed-boundary [10], and mimimum/maximum separation [11]. Performance-related constraints such as monotonic current flows [12] have also been studied. Despite the aforementioned previous work for a wide range of constraint handling, the extensibility of new constraints and the adaptation for region-based layouts are restricted as the manufacturing process evolves.

Analytical approaches optimize placement objectives directly using mathematical formulations. [13] proposes a hierarchical placement framework using mixed-integer linear programming (MILP). However, it suffers from poor scalability for large-scale circuits. In [14], a nonlinear programming (NLP) method is presented to achieve better scalability while considering complex design aspects such as system signal flows. Still, the constraint relaxation and the extra legalization step restrict its potential to handle various constraints simultaneously. [15] encodes the analog placement problem into SMT formulations. The flexible syntax of SMT enables efficient problem modeling and various constraint support. However, it fails to manage routability consideration and wirelength optimization. The use of integer theory may also lead to runtime overhead for system-level designs (e.g., ADC, PLL).

Most of the previous approaches target traditional planar layouts, and limited work has presented an end-to-end placement framework for FinFET circuits. In [16], a meshed tree representation is proposed to handle FinFET-induced constraints (e.g., fin alignment, mask conflict, mask density balance) together with traditional geometric constraints. Nevertheless, the lack of ability to consider routability and additional constraints introduced by the new region-based layout methodology make it unsuitable for designs in more advanced FinFET nodes.

III. Preliminaries

A. Satisfiability Modulo Theories (SMT)

A first-order theory $\mathcal{T}$ comprises a signature $\Sigma_\mathcal{T}$ and axioms $A_\mathcal{T}$, where $\Sigma_\mathcal{T}$ contains a set of constants, functions, and predicate symbols; $A_\mathcal{T}$ is a set of first-order logic (FOL) sentences over $\Sigma_\mathcal{T}$ providing the meaning of symbols. Specifically, the axioms $A_\mathcal{T}$ ensures some legal interpretations in general FOL are not legal under $\mathcal{T}$. A structure $\mathcal{G}$ is a $\mathcal{T}$-model if $\mathcal{G}$ entails A for every $A \in A_\mathcal{T}$. Consider a first-order formula F, if there exists a $\mathcal{T}$-model $\mathcal{G}$ and variable assignment σ such that $\langle \mathcal{G}, \sigma \rangle$ entails F, then F is *satisfiable modulo* $\mathcal{T}$.

The SMT problem determines whether a FOL formulation $F_\mathcal{T}$ is satisfiable (SAT) or unsatisfiable (UNSAT) under some background theories. A model and variable assignment $\langle \mathcal{G}, \sigma \rangle$ of $F_\mathcal{T}$ is returned if $F_\mathcal{T}$ is SAT. Combined theories are also supported for a richer modeling language. Frequently implemented theories in modern SMT solvers include the empty theory (i.e., uninterpreted symbols with equality), arithmetic on integers/reals, bit-vectors (BVs), and the array theory. An example SMT formula can be written as

$$(a + 2 \leq b) \wedge (f(a) = f(b) \vee q(a) \neq \neg q(b)), \qquad (1)$$

where it includes atomic formulas over a language of equality, integer arithmetic, and uninterpreted symbols with Boolean combinations.

In this work, we formulate the placement problem with SMT formulas under the BV theory with Boolean combinations. Fully transferable to propositional logic, pure BV formulas can achieve magnitudes of solving time speedup compared with integers theories. Compared with propositional SAT and MILP formulations, SMT formulas supports richer modeling expressions (e.g., if-then-else, either-or, Boolean cardinality), thus amenable for efficient, versatile formulations for placement instances.

B. Problem Formulation

The placement problem for region-based FinFET AMS layouts is formulated as follows:

Problem 1 (Region-based FinFET AMS Placement): Given a circuit netlist T, a placement grid pattern, a set of primitive cells $V = \{v_i | 1 \leq i \leq |V|\}$, a set of regions $R = \{r_i | 1 \leq i \leq |R|\}$, and a set of AMS placement constraints (listed in Section I), generate an overlap-free placement solution such that each cell $v_i \in V$ locates in the associated region, the specified constraints are satisfied, the routability is considered, and the total placement cost is minimized.

IV. Algorithms

The overall flow of our AMS placement framework is shown in Figure 3, which consists of three main phases: 1) power analysis, which specifies the netlist-dependent power abutment constraints, 2) SMT-based placement, which determines the cells' coordinates with optimized wirelength/routability while satisfying the constraints, and 3) post-processing, which inserts edge/dummy cells for each region and sets the final placement result from the final SMT model and variable assignment.

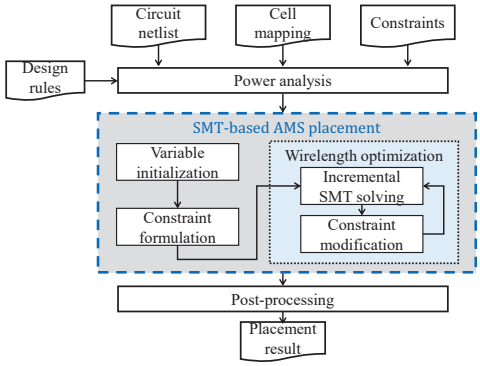


Fig. 3: Computation flow of our AMS placement framework.

A. Power Analysis

Before initializing the SMT formulations, we perform power analysis to capture the power abutment constraints. Due to the standard cell-like placement of region-based layout of primitive cells, cells from different power groups should be placed in disjointed rows; otherwise, a short violation between different power nets would occur, as shown in Figure 4.

B. SMT-Based AMS Placement

To simultaneously handle all the constraints while preserving the flexibility of considering new constraints for unseen designs, we adopt SMT formulations to model the AMS placement problem. Additionally, an incremental solving procedure is built upon the SMT placement kernel to optimize the placement objectives efficiently. Table I summarizes the notations used in this paper.

TABLE I: Notations used in this paper.

Symbol	Description
$V/N/R$	The set of all cells/nets/regions.
$v_i/n_i/r_i$	The i^{th} cell/net/region in $V/N/R$.
x_{v_i}, y_{v_i}	BV vars for the scaled bottom-left coordinate of cell v_i.
w_{v_i}, h_{v_i}	BV constants for scaled width and height of cell v_i.
x_{r_i}, y_{r_i}	BV vars for the scaled bottom-left coordinate of region r_i.
w_{r_i}, h_{r_i}	BV vars for scaled width and height of region r_i.
$x^l_{n_i}, y^l_{n_i}$	BV vars for the bottom-left corner of n_i's bounding box.
$x^h_{n_i}, y^h_{n_i}$	BV vars for the top-right corner of n_i's bounding box.
$f_R(v_i)$	Mapping function $f_R : V \rightarrow R$ that returns the region associated with cell $v_i \in V$.

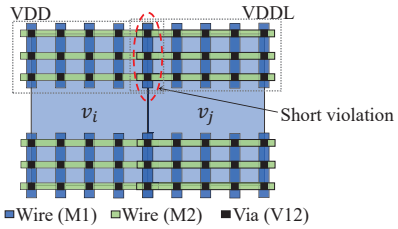


Fig. 4: Example of a power abutment constraint violation. The overlapping power pins of cells v_i and v_j are associated with nets VDD and VDDL, respectively.

1) SMT Variable Initialization: Given a user-specified utilization ratio γ^{ur} and aspect ratio γ^{ar}, we calculate the expected width W and height H of the design as

$$W = \gamma^{ar}\widehat{A},\ H = \frac{\widehat{A}}{\gamma^{ar}}, \tag{2}$$

where $\widehat{A} = \frac{A}{\gamma^{ur}}$ is the estimated total area, and $A = \sum_{v_i \in V} area(v_i)$ is the summation of the area of each cell $v_i \in V$.

Letting the greatest common divider (GCD) of all cell widths and heights be $\bar{w}$ and $\bar{h}$, we scale down the entire design by dividing variables/constants in Table I related to the x-axis, y-axis by $\bar{w}$ and $\bar{h}$, respectively, and define

$$L_x = \log_2(\tilde{W}) + 1,\ L_y = \log_2(\tilde{H}) + 1, \tag{3}$$

where L_x (resp. L_y) represents the size for BV variables associated with the x (resp. y) coordinate, $\tilde{W} = \frac{W}{\bar{w}}$ denotes the scaled design width, and $\tilde{H} = \frac{H}{\bar{h}}$ is the scaled design height. By scaling down the BV variables, every feasible SMT solution for the placement formulation directly guarantees a row-based implementation demanded by region-based layouts and makes sure that all the leftover spaces can be filled by dedicated dummy cells with width $\bar{w}$ and height $\bar{h}$. Significant runtime speedup is also achieved due to the refined solution search space.

C. SMT Constraint Formulation

a) Region Constraints: For each region r_i, region constraints determine its dimension, ensure non-overlapping between r_i and every other region $r_j \in R \setminus \{r_i\}$, and restrict the cells $v_i \in V$ with $f_R(v_i) = r_i$ within the boundaries of r_i. To decide the dimension of a region r_i, we generate a set of scaled width/height candidate pairs S_{r_i} such that for each pair $(w', h') \in S_{r_i}$, we have

$$(w'-1)h' < \widehat{A}_{r_i},\ w'(h'-1) < \widehat{A}_{r_i},\ \text{and}\ w'h' \geq \widehat{A}_{r_i}, \tag{4}$$

where $\widehat{A}_{r_i} = \frac{A_{r_i}}{\gamma^{ur}_{r_i}}$ denotes the approximated area of r_i, A_{r_i} is the area summation of cells in r_i, and $\gamma^{ur}_{r_i}$ is the user-defined utilization ratio for r_i. We formulate the constraints for region dimension as:

$$\bigvee_{(w',h') \in S_{r_i}} (w_{r_i} = w' \wedge h_{r_i} = h'). \tag{5}$$

For non-overlapping of regions, we define the SMT formulas as:

$$\begin{aligned} &\forall r_i, r_j \in R, i \neq j. \\ &\Big(ule(x_{r_i} + w_{r_i} + D_x, x_{r_j}) \vee ule(x_{r_j} + w_{r_j} + D_x, x_{r_i}) \vee \\ &\quad ule(y_{r_i} + h_{r_i} + D_y, y_{r_j}) \vee ule(y_{r_j} + h_{r_j} + D_y, y_{r_i}) \Big), \end{aligned} \tag{6}$$

where $ule(\cdot,\cdot)$ represents the "unsigned less than or equal to operator ($\leq$)" for BVs, and D_x (resp. D_y) is the reserved space for left/right (resp. bottom/top) region edge cells. To limit the cells inside the boundaries of their associated regions, we have

$$\forall v_i \in V.\Big(ule(x_r, x_{v_i}) \wedge ule(x_{v_i} + w_{v_i}, x_r + w_r) \wedge ule(y_r, y_{v_i}) \wedge ule(y_{v_i} + h_{v_i}, y_r + h_r)\Big), \tag{7}$$

where $r = f_R(v_i)$ is the associated region of v_i. The non-overlapping between cells can be formulated similarly as in (6) by setting the reserved space to 0.

b) Hierarchical Symmetry Constraints: For a symmetry group C_{sym}, we formulate the y-axis symmetry constraints as:

$$\begin{cases} 2x_{v_i} + w_{v_i} = 2x_{sym}, & v_i \text{ is self-symmetric,} \\ x_{v_i} + w_{v_i} + x_{v_i'} = 2x_{sym}, & \text{otherwise,} \end{cases} \tag{8}$$

where x_{sym} is an auxiliary BV variable representing the symmetry axis, and $v_i' \in C_{sym}$ is the cell to be placed symmetrically to v_i. The symmetry constraints for other symmetry axes can be defined similarly. Note that with a hierarchical symmetry constraint, a cell can be placed symmetrically with respect to multiple symmetry axes simultaneously.

c) Array Constraints: We model the array constraints as:

$$\begin{aligned} &\Big(x^l_{arr} = \min_{v_i \in C_{arr}} x_{v_i}\Big) \wedge \Big(x^h_{arr} = \max_{v_i \in C_{arr}} x_{v_i} + w_{v_i}\Big) \wedge \\ &\Big(y^l_{arr} = \min_{v_i \in C_{arr}} y_{v_i}\Big) \wedge \Big(y^h_{arr} = \max_{v_i \in C_{arr}} y_{v_i} + h_{v_i}\Big) \wedge \\ &\Big((x^h_{arr} - x^l_{arr})(y^h_{arr} - y^l_{arr}) = |C_{arr}|\Big), \end{aligned} \tag{9}$$

where x^l_{arr}, x^h_{arr}, y^l_{arr}, y^h_{arr} are the left, right, bottom, and top array boundaries, and C_{arr} contains the set of cells inside the array. Additional constraints can be formulated to realize specific array patterns. For example, the widely adopted common-centroid pattern can be modeled as:

$$\Big(\sum_{v_a \in C^A_{arr}} x_{v_a} = \sum_{v_b \in C^B_{arr}} x_{v_b}\Big) \wedge \Big(\sum_{v_a \in C^A_{arr}} y_{v_a} = \sum_{v_b \in C^B_{arr}} y_{v_b}\Big), \tag{10}$$

where $C^A_{arr}, C^B_{arr} \subset C_{arr}$ are two disjoint cell groups in the array.

d) Cluster Constraints: For cluster C_{clus}, we add a virtual net connecting the cells in C_{clus} and define four new auxiliary variables x^l_{clus}, x^h_{clus}, y^l_{clus}, and y^h_{clus} similarly as in (9) for the net bounding box. Then, by assigning a higher weight constant to the virtual net, cells within a cluster will be placed as close as possible after wirelength optimization.

e) Extension Constraints: For extension constraints on regions, cells, and arrays, we adjust their non-overlapping constraints to reserve extra spaces from their boundaries. For instance, the adjusted non-overlapping constraints on cell v_a in Figure 2(d) can be modeled as:

$$\begin{aligned} &\forall v_i \in V,\ i \neq a, f_R(v_i) = f_R(v_j). \\ &\Big(ule(x_{v_i} + w_{v_i} + D^L_{v_a}, x_{v_a}) \vee ule(x_{v_a} + w_{v_a} + D^R_{v_a}, x_{v_i}) \vee \\ &ule(y_{v_i} + h_{v_i}, y_{v_a}) \vee ule(y_{v_j} + h_{v_j}, y_{v_a})\Big), \end{aligned} \tag{11}$$

where $D^L_{v_a}$ and $D^R_{v_a}$ denote the reserved space for v_a from the left and right boundaries, respectively. The extension constraints for regions and arrays can be formulated similarly as in (11).

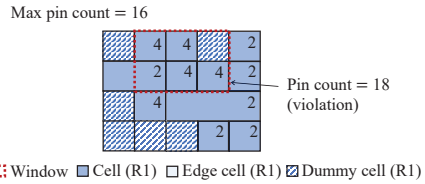


Fig. 5: Example of a pin density constraint violation, where the total pin count inside the window exceeds the threshold.

f) Power Abutment Constraints: For a set of power groups P, we introduce $|P| - 1$ auxiliary variables, denoted as $y^1_{pow}, y^2_{pow}, \ldots, y^{|P|-1}_{pow}$ such that $y^1_{pow} < y^2_{pow} < \cdots < y^{|P|-1}_{pow}$. Then, for cells in the i^{th} power group P_i, we have

$$\forall v_i \in P_i.\Big(y^{i-1}_{pow} \leq y_{v_i} \wedge y_{v_i} + h_{v_i} \leq y^i_{pow}\Big). \tag{12}$$

Note that the lower (resp. upper) bound of the first (resp. last) power group is bounded by the region boundary constraints.

g) Pin Density Constraints: Compared with digital circuits, AMS designs have fewer nets, and thus the routability issue is dominated by local pin density. Therefore, we propose a window-based pin density checking formulation to prevent localized routing congestion. Let $\tilde{W}$ and $\tilde{H}$ be the scaled width and height of a design, β_x and β_y be the scaled width and height of a check window, we construct a set of check windows M with $|M| = (\tilde{W} - \beta_x + 1)(\tilde{H} - \beta_y + 1)$ covering the entire design floorplan. For each check window $m_j \in M$, we add a set of auxiliary Boolean indicators B such that $|B| = |V|$, where $b_{i,j} \in B$ equals 1 if cell v_i overlaps with m_j. Let the bottom-left coordinate of m_j be (x_{m_j}, y_{m_j}). The indicators $b_{i,j}$ satisfy the following formulation:

$$\begin{aligned} \forall v_i \in V.\Big(&\big(ugt(x_{m_j} + \beta_x, x_{v_i}) \wedge ugt(x_{v_i} + w_{v_i}, x_{m_j}) \wedge \\ &ugt(y_{m_j} + \beta_y, y_{v_i}) \wedge ugt(y_{v_i} + y_{v_i}, y_{m_j})\big) \rightarrow b_{i,j}\Big), \end{aligned} \tag{13}$$

where $ugt(\cdot,\cdot)$ denote the "unsigned greater than operator ($>$)" for BVs. To further restrict the total pin count inside every window by a threshold λ_{th}, we apply pseudo-Boolean constraints

$$\forall m_j \in M.\Big(\sum_{i=1}^{|B|} |P(v_i)| \cdot b_{i,j} \leq \lambda_{th}\Big), \tag{14}$$

where $P(v_i)$ represents the set of signal pins inside v_i. Figure 5 illustrates the concept of window-based pin density checking. Combining (13) and (14), our formulation improves the overall routability by eliminating local pin density hotspots.

D. Wirelength Optimization

Given an SMT placement instance $F_{\mathcal{T}}$ specifying the constraints described above, we optimize the total wirelength by performing incremental SMT solving on $F_{\mathcal{T}}$. Algorithm 1 sketches the incremental solving procedure. For each net $n_i \in N$, we attach four auxiliary variables $x^l_{n_i}$, $x^h_{n_i}$, $y^l_{n_i}$, and $y^h_{n_i}$ signifying the net bounding box. In Lines 1-3, we initialize an SMT expression Φ for the weighted total wirelength with the weight η_{n_i} of each net. Virtual nets added by cluster constraints are also included. Then, incremental solving on $F_{\mathcal{T}}$ continues until reaching the maximum iteration K_{iter} or the ending criteria (i.e., UNSAT, timeout) is met, as shown in Lines 4-10. If F_T is SAT, we record the resulting solution and add a stricter wirelength

Algorithm 1 *IncrementalSMTSolving*($F_{\mathcal{T}}$)

Input: The SMT placement formulation $F_{\mathcal{T}} = (\Sigma_{\mathcal{T}}, A_{\mathcal{T}})$.
Output: A model and variable assignment $\langle \mathcal{G}, \sigma \rangle$ with wirelength Φ.

1: $\Phi := 0$;
2: **for each** net $n_i \in N$ **do** ▷ Init. the total wirelength expression
3: $\quad \Phi := \Phi + \eta_{n_i}(x^h_{n_i} - x^l_{n_i} + y^h_{n_i} - y^l_{n_i})$;
4: **for** $i = 1$ to K_{iter} **do**
5: $\quad$ **if** *Solve*($F_{\mathcal{T}}$) = SAT **then**
6: $\quad\quad \langle \mathcal{G}, \sigma \rangle := \textit{Model}(F_{\mathcal{T}})$;
7: $\quad\quad \Phi' := \textit{Wirelength}(\langle \mathcal{G}, \sigma \rangle)$;
8: $\quad\quad$ Add wirelength optimization constraint $(\Phi < \zeta\Phi')$ to $F_{\mathcal{T}}$;
9: $\quad\quad$ Add assumptions to freeze some region/cell variables in $\Sigma_{\mathcal{T}}$;
10: $\quad$ **else break**;
return $(\langle \mathcal{G}, \sigma \rangle, \Phi)$;

constraint $(\Phi < \zeta\Phi')$ to F_T, where $\zeta \in (0, 1]$ is a constant whose value decreases as the iteration increases, and Φ' denote the total wirelength based on the current solution $\langle \mathcal{G}, \sigma \rangle$. By adding assumptions to some regions and cell variables (Line 9), we can achieve significant runtime speedup due to the refined solution space. In practice, we maintain two sorted queues in descending order for regions and cells to select the assumed variables, denoted as Q_r and Q_v, respectively. The priority functions PR_{r_i} for region r_i and PR_{v_i} for cell v_i are defined as:

$$PR_{r_i} = A_{r_i}, \; PR_{v_i} = \delta_1 |P(v_i)| + \delta_2 \sum_{n_i \in N(v_i)} deg(n_i), \quad (15)$$

where $P(v_i)$ denotes the set of pins on v_i, $N(v_i)$ is the set of nets connected to v_i, $deg(n_i)$ is the degree of n_i, and δ_1, δ_2 are weighted constants. In our implementation, δ_1 and δ_2 are set to 10 and 1, respectively.

V. Experimental Results

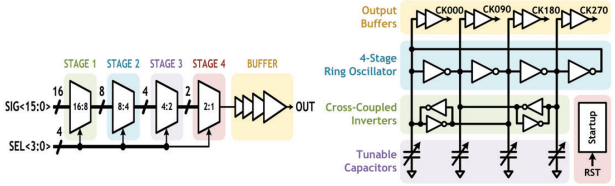
The proposed AMS placement framework is implemented in C++ with the Z3 SMT solver [17]. All experiments are conducted on a Linux workstation with an Intel Xeon 2.7 GHz CPU with 128GB memory. We conduct experiments on two industrial designs, including a multiplexing buffer (BUF) and a voltage-controlled oscillator (VCO). Figure 6 shows the circuit schematics. All benchmark circuits are designed by experienced AMS circuit designers under TSMC 5nm process, and the placement constraints are annotated by layout experts. After generating the placement layouts, the routing is completed by an analog router [18] with slight manual adjustment if needed. Then, we analyze the routed layouts by comparing detailed post-layout extractions and simulations with the optimized manual layout. Table II lists the benchmark statistics.

TABLE II: Statistics of the circuit benchmarks.

Benchmark	#Regions	#Cells	#Nets	Tech
BUF	1	42	66	5nm FinFET
VCO	2	110	71	5nm FinFET

A. 16-to-1 Multiplexing Buffer

The multiplexing buffer is a full-custom design that selects between 16 signals using 4-bit control for performance and status monitoring. The circuit includes an output buffer for driving large parasitic and device loads, as shown in Figure 6(a). Hierarchical symmetry constraints should be applied for lower delay and parasitic capacitance variability between stages. To demonstrate



(a) (b)

Fig. 6: Circuit schematics. (a) 16-to-1 multiplexing buffer. (b) Four-stage voltage-controlled oscillator.

TABLE III: Comparisons of total area, half-perimeter wirelength (HPWL), routed wirelength (RWL), routed via count (VIA), and runtime of the BUF.

	Manual	w/o Cstr.	w/ Cstr.
Area (μm^2)	56.64 (1.49)	38.09 (1.00)	38.09 (1.00)
HPWL (μm)	N/A	95.07 (1.35)	70.22 (1.00)
RWL (μm)	N/A	134.33 (1.62)	82.90 (1.00)
VIA	N/A	326 (1.08)	300 (1.00)
Runtime (s)	N/A	798.54 (6.87)	116.18 (1.00)

the importance of the proposed constraints, we generate layouts with and without the hierarchical symmetry constraints. Critical constraints such as pin density constraints are applied to both layouts to ensure routablility. Note that for both layouts, the optimization loop (Algorithm 1) terminates after five iterations.

Table III shows the comparisons of the basic placement metrics. Capable of generating numerous solutions while satisfying the constraints, our framework can potentially produce superior placement results. The manual layout results in a 49% larger total area. Comparing the layouts with and without hierarchical symmetry constraints, we observe that the layout without proper constraints results in 35% longer HPWL, 62% longer RWL, 8% more vias, and 6.87× longer runtime, authenticating that with accurate design constraints specified, the proposed SMT-based placer has significant runtime speedup and quality improvement.

Table IV shows the detailed analyses of the insertion delays and rise/fall times, including the results of all internal stages. For the internal stages, the automated layout with proper constraints achieves an 8% reduction in insertion delay with lower delay variability compared to the manual layout. It also obtains lower average rise/fall times with significantly smaller variability. For the delays of all input-to-output paths, the automated layout achieves 6% average insertion delay reduction and considerably smaller delay variability. The notable performance degradation of the layout generated without the additional constraints demonstrates the significance of accurate constraint specification.

B. Four-Stage Voltage-Controlled Oscillator

The four-stage VCO is designed to generate complimentary 7.7GHz in-phase and quadrature-phase clocks at a nominal 750mV supply. As shown in Figure 6(b), it includes start-up circuitry and 3-bit thermometer-encoded control of digitally tunable capacitors for frequency trimming. We apply hierarchical symmetry, array, cluster, and extension constraints on top of the necessary constraints. To highlight the efficacy of the proposed design constraints, we generate another layout without applying the crucial constraints above. The optimization loop terminates after four iterations.

Table V details the results on placement metrics. With superb constraint solving ability, our SMT-based AMS placer achieves a more optimized circuit size where the area of the manual

TABLE IV: Comparison of the post-layout insertion delay and rise/fall times of the BUF.

Stage		Insertion Delay (ps)			Rise/Fall Time (ps)		
		Manual	w/o Cstr.	w/ Cstr.	Manual	w/o Cstr.	w/ Cstr.
1	Avg	12.3	10.3	9.5	12.1 / 10.9	11.0 / 10.01	10.0 / 9.1
	SD	0.88	0.83	0.07	0.55 / 0.55	1.31 / 1.16	0.06 / 0.06
2	Avg	12.0	11.9	10.5	12.2 / 12.5	11.9 / 11.9	10.4 / 10.6
	SD	0.91	0.51	0.17	0.77 / 0.59	0.76 / 0.71	0.17 / 0.17
3	Avg	12.4	12.3	11.8	13.4 / 11.6	11.9 / 10.9	12.0 / 10.9
	SD	0.50	0.35	0.04	0.98 / 0.41	0.26 / 0.21	0.02 / 0.04
4	Avg	9.4	11.0	10.1	10.2 / 10.4	10.2 / 10.2	9.2 / 9.2
	SD	1.01	0.01	0.15	-	-	-
OUT	Avg	35.8	35.8	35.2	5.8 / 5.8	6.1 / 6.2	6.1 / 6.2
	SD	-	-	-	-	-	-
Total	Avg	82.0	81.4	77.2	-	-	-
	SD	0.92	1.28	0.18	-	-	-

TABLE V: Comparison of total area, half-perimeter wirelength (HPWL), routed wirelength (RWL), routed via count (VIA), and runtime of the VCO.

	Manual	w/o Cstr.	w/ Cstr.
Area (μm^2)	68.89 (1.23)	56.14 (1.00)	56.14 (1.00)
HPWL (μm)	N/A	231.82 (1.57)	147.90 (1.00)
RWL (μm)	N/A	292.32 (1.88)	155.45 (1.00)
VIA	N/A	576 (1.60)	361 (1.00)
Runtime (s)	N/A	205.90 (1.87)	110.26 (1.00)

layout is 23% larger. Comparing the two layouts generated by our framework, we can find that without specifying proper constraints it produces 57% longer HPWL, 88% longer RWL, 60% more vias, and 87% longer runtime. Since more constraints are specified for a refined solution space, the runtime of the VCO is faster even though it is a more complicated design with more regions, cells, and nets than the BUF circuit. In practice, complicated AMS systems require sophisticated constraints, thus suited to our framework.

Table VI summarizes the post-layout simulation results. One can observe that the layout generated by our framework consistently outperforms the manual design in terms of power consumption and operating frequency under various supply voltages (from 650mV to 900mV). As shown in Table VI, the manual layout results in 2% higher power consumption and 2% slower oscillation frequency on average compared with ours. Comparing the two automated layouts, we observe that the layout without correct constraints results in 12% degradation on the oscillation frequency. To further investigate the layout behavior at ultra-high speed, we analyze the oscillation frequency for various capacitor trim codes. Figure 7 plots the performance comparisons with different capacitor trim codes. As shown in Figure 7, with various supply voltages, the automated layout with precise constraints consistently achieves higher oscillation frequency for all capacitor trim codes.

VI. Conclusion

This work has presented an end-to-end AMS placement framework for advanced FinFET technology. SMT formulations have been shown to handle various placement constraints simultaneously for region-based AMS layouts. A window-based pin density checking method has been proposed to eliminate local congestion for better routability. An incremental SMT solving scheme has been proposed to optimize the placement objectives. Experimental results have demonstrated the efficiency and effectiveness of the proposed framework in producing high-quality placement solutions.

TABLE VI: Comparison of the post-layout power consumption and oscillation frequency of the VCO.

Supply (mV)	Power (μW)			Frequency (GHz)		
	Manual	w/o Cstr.	w/ Cstr.	Manual	w/o Cstr.	w/ Cstr.
650	304.4	302.2	300.2	3.02	2.76	3.08
700	398.8	395.1	392.7	3.28	2.97	3.34
750	507.5	501.2	499.6	3.49	3.15	3.55
800	632.4	622.2	621.6	3.67	3.28	3.73
850	774.6	759.7	758.5	3.83	3.39	3.88
900	936.0	912.6	914.4	3.96	3.48	4.00
Norm.	1.02	1.00	1.00	0.98	0.88	1.00

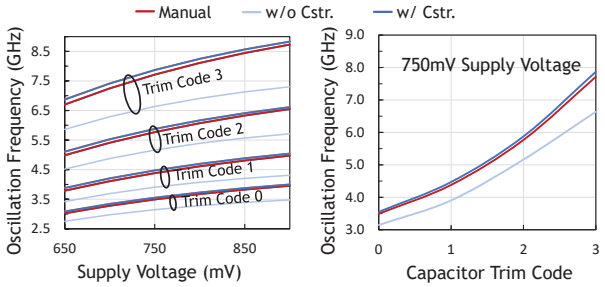


Fig. 7: Oscillation frequency v.s. Supply voltage under various capacitor trim codes for the VCO.

Acknowledgement

This work is supported in part by NVIDIA Corporation, the DARPA IDEA program, and the NSF under Grant No. 1704758.

References

[1] K. Kunal *et al.*, "ALIGN: Open-source analog layout automation from the ground up," in *Proc. DAC*, 2019, pp. 1–4.

[2] M. Madhusudan *et al.*, "Analog layout generation using optimized primitives," in *Proc. DATE*, 2021, pp. 1234–1239.

[3] A. Hastings and R. A. Hastings, *The Art of Analog Layout*, 1st ed. USA: Prentice Hall, 2001.

[4] H. Chen *et al.*, "Universal symmetry constraint extraction for analog and mixed-signal circuits with graph neural networks," in *Proc. DAC*, 2021, pp. 1243–1248.

[5] F. Balasa and K. Lampaert, "Symmetry within the sequence-pair representation in the context of placement for analog design," *IEEE TCAD*, vol. 19, no. 7, pp. 721–731, 2000.

[6] L. Xiao *et al.*, "Practical placement and routing techniques for analog circuit designs," in *Proc. ICCAD*, 2010, pp. 675–679.

[7] P.-Y. Chou *et al.*, "Heterogeneous B*-Trees for analog placement with symmetry and regularity considerations," in *Proc. ICCAD*, 2011, pp. 512–516.

[8] P.-H. Lin and S.-C. Lin, "Analog placement based on hierarchical module clustering," in *Proc. DAC*, 2008, pp. 50–55.

[9] H.-F. Tsao *et al.*, "A corner stitching compliant B*-Tree representation and its applications to analog placement," in *Proc. ICCAD*, 2011, pp. 507–511.

[10] Q. Ma *et al.*, "Simultaneous handling of symmetry, common centroid, and general placement constraints," *IEEE TCAD*, vol. 30, no. 1, pp. 85–95, 2011.

[11] I.-P. Wu *et al.*, "QB-trees: Towards an optimal topological representation and its applications to analog layout designs," in *Proc. DAC*, 2016, pp. 1–6.

[12] A. Patyal *et al.*, "Analog placement with current flow and symmetry constraints using pcp-sp," in *Proc. DAC*, 2018, pp. 1–6.

[13] B. Xu *et al.*, "Hierarchical and analytical placement techniques for high-performance analog circuits," in *Proc. ISPD*, 2017, pp. 55–62.

[14] K. Zhu *et al.*, "Effective analog/mixed-signal circuit placement considering system signal flow," in *Proc. ICCAD*, 2020, pp. 1–8.

[15] S. M. Saif *et al.*, "Pareto front analog layout placement using satisfiability modulo theories," in *Proc. DATE*, 2016, pp. 1411–1416.

[16] Y.-S. Lu *et al.*, "WB-Trees: A meshed tree representation for finfet analog layout designs," in *Proc. DAC*, 2018, pp. 1–6.

[17] L. De Moura and N. Bjørner, "Z3: An efficient smt solver," in *Proc. TACAS*, 2008, pp. 337–340.

[18] H. Chen *et al.*, "Toward silicon-proven detailed routing for analog and mixed-signal circuits," in *Proc. ICCAD*, 2020, pp. 1–8.