Analog/Mixed-Signal Layout Optimization Using Optimal Well Taps

Ramprasath S
Univ. of Minnesota, USA

Meghna Madhusudan
Univ. of Minnesota, USA

Arvind K. Sharma
Univ. of Minnesota, USA

Jitesh Poojary
Univ. of Minnesota, USA

Soner Yaldiz
Intel Labs, USA

Ramesh Harjani
Univ. of Minnesota, USA

Steven M. Burns
Intel Labs, USA

Sachin S. Sapatnekar
Univ. of Minnesota, USA

ABSTRACT

Well island generation and well tap placement pose an important challenge in automated analog/mixed-signal (AMS) layout. Well taps prevent latchup within a radius of influence in a well island, and must cover all devices. Automated AMS layout flows typically perform well island generation and tap insertion as a postprocessing step after placement. However, this step is intrusive and potentially alters the placement, resulting in increased area, wire length, and performance degradation. This work develops a graph-based optimization that integrates well island generation, well tap insertion, and placement. Its efficacy is demonstrated within a stochastic placement engine. Experimental results show that this approach generates better area, wire length and performance metrics than traditional methods, at the cost of a marginal runtime degradation.

CCS CONCEPTS

• **Hardware → Physical design (EDA)**; **Analog and mixed-signal circuit optimization**; *Placement*.

KEYWORDS

Well island generation, Well tap sharing, Analog circuits, Mixed-signal circuits, AMS layout automation, Circle graph, Jordan curve

ACM Reference Format:
Ramprasath S, Meghna Madhusudan, Arvind K. Sharma, Jitesh Poojary, Soner Yaldiz, Ramesh Harjani, Steven M. Burns, and Sachin S. Sapatnekar. 2022. Analog/Mixed-Signal Layout Optimization Using Optimal Well Taps. In *Proceedings of the 2022 International Symposium on Physical Design (ISPD '22), March 27–30, 2022, Virtual Event, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3505170.3506728

This work is supported in part by the DARPA IDEA program, as part of the ALIGN project, under SPAWAR Contract N660011824048.

ISPD '22, March 27–30, 2022, Virtual Event, Canada

ACM ISBN 978-1-4503-9210-5/22/03…$15.00
https://doi.org/10.1145/3505170.3506728

1 INTRODUCTION

In a layout, MOSFET devices of different channel types (P, N, Deep N, or Deep P channel) are associated with different *well layers* that constitute the bulk regions of the devices. Continuous regions of the same well layer can span multiple devices, and are known as *well islands*. Each well island must be connected to a corresponding power supply, through a *well tap* cell that is typically provided by a foundry as part of the process design kit (PDK). These connections can prevent latchup [5, 16] within a well-layer-specific radius of influence (denoted $\mathcal{R}_w$ for well layer w) specified in the PDK.

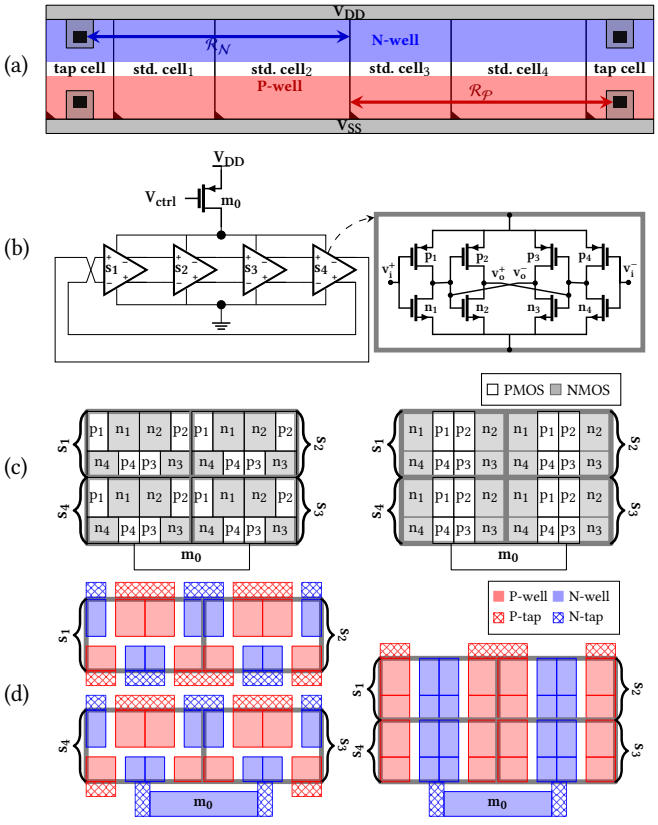


Figure 1: (a) Digital layout with regular well islands and taps; (b) An inverter-coupled VCO; (c) Two post-placement layouts of the VCO; (d) The corresponding layouts of the VCO after well definition and tap placement.

In digital circuits, standard cell rows (Figure 1(a)) have systematically demarcated well islands, with well tap cells placed at regular intervals, connected to power pins [18]. In contrast, analog/mixed signal (AMS) designs face stringent performance constraints that require low parasitics between critical transistors. This forces transistors of different well types to be proximate in the layout, making well definition and tap placement problems much more complex. In a typical automated AMS layout generation flow, in the post-placement stage, the well regions of devices are first merged to create well islands, e.g., using geometric operations [3] or generative adversarial networks (GANs) [17]. The placement is then altered to meet design rule constraints on spacing between well regions; next, well taps are added. Each of these incurs area overheads.

AMS placement choices can profoundly influence these well-related overheads. Consider a differential ring oscillator based voltage controlled oscillator (VCO) (Figure 1(b)). Two layouts for the VCO are shown in Figure 1(c) at the post-placement stage, where four large rectangles show the current-starved differential inverter stages s_1 through s_4 of the VCO, each consisting of four PMOS and four NMOS devices. After well regions are defined, the layouts must insert adequate spacing between wells and must place well taps: the corresponding layouts after this stage for the VCO are shown in Figure 1(d). The left configuration, which has complex well structures, incurs significant well separation requirements and well tap overheads. The area cost of the layout at left is 35% higher than the optimal layout at the right, resulting in larger parasitics and degraded performance. In [13], the problem is partly alleviated by generating well islands during placement, but (a) it is limited to generating only rectangular well islands and (b) it ignores the overheads of well tap insertion.

This paper optimizes the *true* cost of well island generation and tap insertion *during* placement and optimally positions well taps using a novel graph-based algorithm. Section 2 covers preliminaries, graph-based constructs, and the placement optimization framework. Next, Section 3 describes the algorithms for well island generation and for finding the optimal number of well taps. Next, Section 4 demonstrates the improvements in placement quality achieved by our algorithm, and finally, Section 5 concludes the paper.

2 WELL TAP GRAPH FORMULATION

2.1 Preliminaries

We use the telescopic operational transconductance amplifier (OTA) in Figure 2(a) as a running example in this section. Our approach is applied to an automatic layout generator that uses library-based annotation [4, 11] to identify subcircuits known as primitives (using the terminology of [4]): the identified primitives of the OTA are labeled in Figure 2(a). For each primitive, a cell generator constructs candidate DRC-clean layouts with various aspect ratios.

For each generated primitive layout, the generator is modified to provide variants: *tapped*, with well taps, and *tapless*, without any well tap. A well tap may be vertical or horizontal or both, and different devices in the same circuit may have different tap orientations and have multiple taps. Figure 2(b) and (c) illustrates two variants for the differential pair (M_3, M_4) in Figure 2(a). Each tapped cell has an *active* and *tap* region, as shown in the figure.

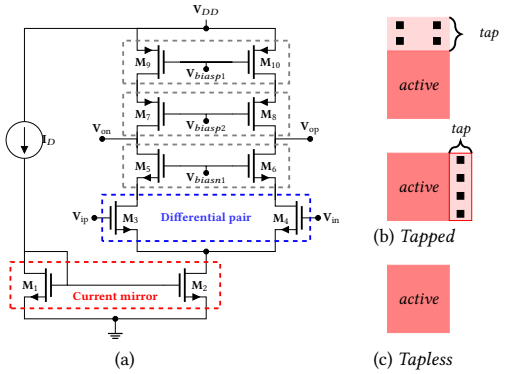


Figure 2: (a) Telescopic OTA without a bias circuit; (b) and (c) NMOS differential pair (M_3,M_4) layouts.

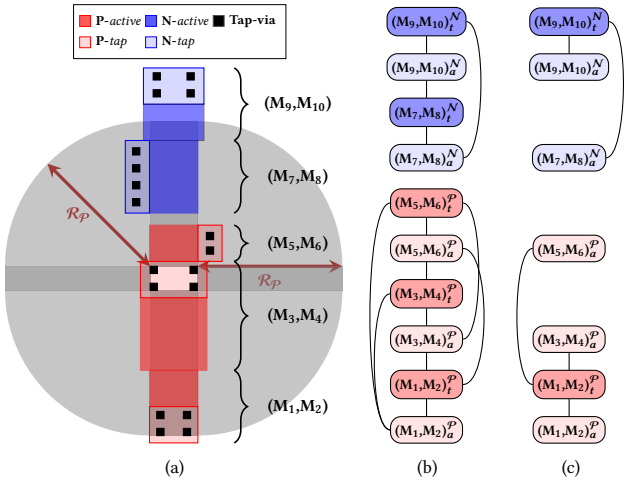


Figure 3: (a) Layout of telescopic OTA in Figure 2(a). (b) Its equivalent well tap graph. (c) The optimal tap vertices.

A well tap for well layer w has a radius of influence, $\mathcal{R}_w$, defined in the PDK. The bulk impedance between the tap and any active device within this distance lies within the foundry-characterized limit for overcoming latchup. The locus of regions of equal substrate resistance from a point is a Euclidean circle. Typically, $w \in \{\mathcal{N}, \mathcal{P}, \text{Deep } \mathcal{N}, \text{Deep } \mathcal{P}\}$, representing an N-well, P-well, Deep N-well and Deep P-well, respectively; each has a different $\mathcal{R}_w$. To ensure that the depletion region of different device types do not overlap during operation, well islands of different well types need to be separated in the layout by a minimum spacing specified in the PDK.

2.2 Graph Construction

We devise a formulation to generate a layout with optimal well taps and well islands. The input is a placement P, specified using coordinates of all the cells and their orientation, where all cells are *tapped*. Our formulation detects and replaces redundant *tapped* cells in P with their *tapless* version, and then compacts the layout.

For the core problem of detecting redundant taps, we construct a undirected well tap graph $G(P) = (V, E)$. The vertex set $V = A \cup T$, where the vertices in A and T represent the set of *active* or *tap* cells, respectively. An *active* vertex C_a is connected to a *tap* vertex D_t by

an undirected edge if (a) C_a lies completely within distance $\mathcal{R}_w$ of D_t, and (b) C_a and D_t belong to the same well island. The graph G is bipartite with A and T forming the two parts.

Figure 3(a) shows a candidate layout of the telescopic OTA, where each cell X has both *active* (X_t^w) and *tap* (X_a^w) regions, represented by darker and lighter red/blue rectangles, respectively. The continuous P-well region comprising the active and tap regions of (M_1,M_2), (M_3,M_4) and (M_5,M_6) forms a P-well island, while the remaining devices form an N-well island. The vertical space between the cells (M_5,M_6) and (M_7,M_8) is required to honor the spacing requirement between the N-well and P-well islands.

The halo region around the tap for (M_3,M_4), shown by the gray region, completely envelopes the active regions of cells (M_1,M_2), (M_3,M_4) and (M_5,M_6); therefore, it is a valid tap for these cells. Although the halo overlaps the PMOS devices (M_7,M_8), it is not a valid tap for these as they lie in a different well. Figure 3(b) shows the corresponding well tap graph G with the edges between active and tap vertices. Here, $(M_3,M_4)_t^{\mathcal{P}}$ has edges with each of $(M_1,M_2)_a^{\mathcal{P}}$, $(M_3,M_4)_a^{\mathcal{P}}$ and $(M_5,M_6)_a^{\mathcal{P}}$, the NMOS devices that lie within its halo. The number of strongly connected components of the graph G correspond to the number of wells: here, the two components correspond to the N-well vertices (shown in shades of blue) and P-well vertices (shown in shades of red).

2.3 Stochastic placement engine

Mainstream analog placers are based on stochastic placement engines such as simulated annealing [1, 3, 7, 9, 12], where at a succession of temperatures, a set of random perturbations are performed; cost-reducing moves are accepted and cost-increasing moves are conditionally accepted, with a probability that decreases with temperature. We will augment this procedure to perform well island generation and well tap selection for each candidate placement.

Our formulation is based on a small integer linear program (ILP), which practically does not significantly increase the total runtime of layout synthesis. We explore the space of block arrangement corresponding to sequence pairs [12], and each perturbation alters the current sequence pair as in [9]. At each step, the optimizer evaluates the objective function and ensures the satisfaction of constraints. We will modify the optimization procedure to incorporate well island definition and well tap insertion, so that the cost function can be evaluated incrementally after each perturbation.

3 DETAILS OF THE ALGORITHM

We now describe the key steps of the algorithm. For any candidate placement P evaluated by the stochastic placement engine, we first define a set of well islands, mapping the problem to that of planar routing. Next, we determine the optimal choice of well taps for P based on an ILP formulation and determine the cost function for the layout for the placement engine. At the end of the stochastic optimization, we describe how the ILP solution is translated into a set of well islands for the optimal placement. We also describe the extension of the approach to a hierarchical placement methodology.

3.1 Well Island Definition

Based on the graph defined above, we first identify groups of active and tap cells that can share a *well island*. Each such well island must have at least one tap cell. Therefore, for each tap cell, we first identify the candidate active devices that can share its well island. A set of active cells can belong to the same well island as a tap cell if they are either placed adjacent to the tap cell, or can be connected to the tap cell in the layout with a rectilinear well layer shape. This connection is analogous to routing a net, whose pins are the active and tap cells, on a single layer (corresponding to the substrate). The active and tap cells of different well types thus become obstacles while routing this net. This has similarities to the problem of multipin single-layer routing, which is an NP-complete problem [8], of N-well and P-well pins. However, this problem is more complex, because it must also partition the set of N-well and P-well pins to optimize the overheads of well formation, as illustrated in Figure 1(d).

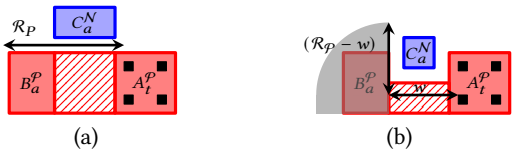


Figure 4: (a) Unobstructed and (b) partially obstructed straight-line routes between two cells of same well type.

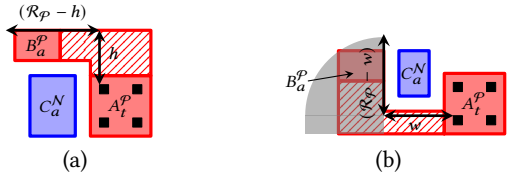


Figure 5: L-shaped route between two vertices of same well layer: (a) no line-of-sight (b) fully obstructed line-of-sight.

To make the problem tractable, we solve a simpler problem of connecting two-pin connections between tap cells and active cells of the same well layer. We consider two types of routes:

- *Straight-line routes:* Two cells (pins) of the same well type can be connected using a rectangle-shaped route if they are in either vertical or horizontal line-of-sight and they are unobstructed or partially obstructed by a different well layer obstacle. Figure 4(a) and (b) illustrate the unobstructed and partially obstructed scenarios with cells A_t and B_a of well type $\mathcal{N}$ and a cell C_a of of well type $\mathcal{P}$ acting as an obstacle. The hatched rectangle shape in each of the figures represents the straight-line route.
- *L-shaped routes:* If two cells (pins) have no vertical or horizontal line-of-sight, or if the line-of-sight is fully obstructed by an obstacle, then the vertices could be connected using L-shaped routes. Figure 5(a) and (b) illustrate L-shaped routes for the no-line-of-sight and fully-obstructed-line-of-sight cases, respectively.

The straight-line and L-shape routes are found using straightforward geometric operations, based on the coordinates of the cells to be connected by the route, and those of intermediate obstacles. Details are omitted due to space limitations.

Two routes (well islands) of same well type can overlap to create a larger island. The well separation makes it *illegal* for two well islands of different well types to be within a minimum well-separation of d_w. Two well islands that have a separation less than d_w represent an illegal *short* between the corresponding two nets. A legal island configuration is a collection of islands that have no illegal shorts between any two well islands. To obey design rules,

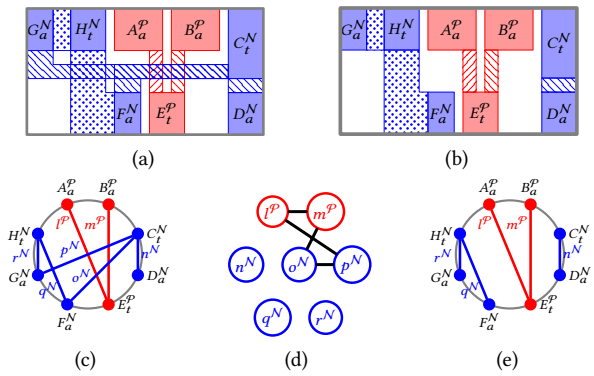


Figure 6: (a) Layout with overlapping well islands; (b) Optimal non-overlapping well islands; (c) Jordan curve corresponding to layout in (a); (d) Circle graph corresponding to the curve in (c); (e) Optimal nonoverlapping edges.

the pins of the nets and the routes for dissimilar well types must be spaced apart by d_w. Therefore, once the straight-line/L-shaped connections are identified, we process the placement P for a candidate sequence pair by adding the well separation constraint into the compaction procedure (embedded within the sequence pair computation [9, 12]) to ensure legality. This allows us to guarantee, in later steps, when we choose a subset of tap cells and routes, that the final placement remains legal after compacting the layout to remove white space corresponding to unselected tap cells/routes.

As an example of illegal island formation, consider the configuration in Figure 6(a) with eight pins, $\{A_a^{\mathcal{P}}, B_a^{\mathcal{P}}, C_t^{\mathcal{N}}, D_a^{\mathcal{N}}, E_t^{\mathcal{P}}, F_a^{\mathcal{N}}, G_a^{\mathcal{N}}, H_t^{\mathcal{N}}\}$, three of which are tap cells. The pins are connected by a set of nets connected corresponding to the straight-line and L-shaped routes between active cells and tap cells $\{l^{\mathcal{P}} = (A_a^{\mathcal{P}}, E_t^{\mathcal{P}})$, $m^{\mathcal{P}} = (B_a^{\mathcal{P}}, E_t^{\mathcal{P}})$, $n^{\mathcal{N}} = (D_a^{\mathcal{N}}, C_t^{\mathcal{N}})$, $o^{\mathcal{N}} = (F_a^{\mathcal{N}}, C_t^{\mathcal{N}})$, $p^{\mathcal{N}} = (G_a^{\mathcal{N}}, C_t^{\mathcal{N}})$, $q^{\mathcal{N}} = (F_a^{\mathcal{N}}, H_t^{\mathcal{N}})\}$, $r^{\mathcal{N}} = (G_a^{\mathcal{N}}, H_t^{\mathcal{N}})\}$. The figure shows that this configuration is illegal due to shorts between the net pairs $(l^{\mathcal{P}}, o^{\mathcal{N}})$, $(l^{\mathcal{P}}, p^{\mathcal{N}})$, $(m^{\mathcal{P}}, o^{\mathcal{N}})$ and $(m^{\mathcal{P}}, p^{\mathcal{N}})$. Note that the overlapping net pair $(p^{\mathcal{N}}, q^{\mathcal{N}})$ is not a short as both nets are of the same well type.

To minimize the number of well islands while ensuring legality, a maximal number of nets must be routed without shorts. This is the maximal planar topological routing problem for two-pin nets [8, 10], which can be solved optimally with the following three steps:
Step 1: *Jordan curve representation:* A Jordan curve is a closed non-self-intersecting curve in a plane that divides the plane into two regions: the *interior* and the *exterior*. In our formulation, the Jordan curve is the smallest bounding box enclosing all pins under consideration. The interior (exterior) of the bounding box corresponds to the interior (exterior) of the Jordan curve. All the pins that intersect the bounding box are placed as pins on the curve either in a clockwise or counter-clockwise fashion. Any pin that does not intersect the bounding box can be placed on the curve using linear-time transformations shown in [10].

For brevity, we demonstrate the technique using just the vertices that intersect the bounding box. Figure 6(c) shows a Jordan curve (circle) equivalent of the layout in Figure 6(a). The Jordan curves represents the perimeter of the smallest bounding box enclosing the vertices, and the pins on the perimeter are placed clockwise. The chords $l^{\mathcal{P}}, m^{\mathcal{P}}, n^{\mathcal{N}}, o^{\mathcal{N}}, p^{\mathcal{N}}, q^{\mathcal{N}}$, and $r^{\mathcal{N}}$ represent the nets with straight-line and L-shaped routes. When two chords of different well types intersect in the circle graph, we have a short.
Step 2: *Circle graph construction:* A circle graph $CG = (V_c, E_c)$ is constructed for the Jordan curve, where V_c is the set of all the chords in the Jordan curve, and the edges in E_c represent the illegal shorts. Hence, two vertices in V_c are connected by an edge if the corresponding chords are of different well types and they intersect in the Jordan curve. Figure 6(d) shows the circle graph corresponding to the Jordan curve in Figure 6(c). The chords in Figure 6(d) correspond to the nets. The circle graph has edges between nodes that correspond to the intersections of chords (nets) $l^{\mathcal{P}}$ and $m^{\mathcal{P}}$ with chords $o^{\mathcal{N}}$ and $p^{\mathcal{N}}$, which are shorted in Figure 6(c).[1]
Step 3: *Supowit's algorithm* [14] finds the maximum independent set (MIS) of circle graph CG, where MIS is the largest subset of V_c such that no two vertices in the subset are connected by an edge. The MIS of CG represents the maximum number of nets without shorts. Supowit's algorithm uses the fact that the MIS of a subgraph remains the same or increases by one element on addition of a new vertex to a subgraph. This fact is used to construct a dynamic programming approach that exhaustively searches through all possible subgraphs in $O(|V_c|^2)$ time, where $|V_c|$ is the cardinality of V_c. Figure 6(e) shows the maximum number of non-overlapping edges found using Supowit's algorithm on the circle graph in Figure 6(d).

After these three steps, we identify the minimal number of well islands: cells that are connected by a chord belong to the same island. In this example, the islands are $\{F_a^{\mathcal{N}}, G_t^{\mathcal{N}}, H_a^{\mathcal{N}}\}$, $\{A_a^{\mathcal{P}}, B_a^{\mathcal{P}}, E_t^{\mathcal{P}}\}$, and $\{C_t^{\mathcal{N}}, D_a^{\mathcal{N}}\}$. Figure 6(b) shows the layout without shorts corresponding to the chords shown in Figure 6(e).

3.2 Well Tap Optimization

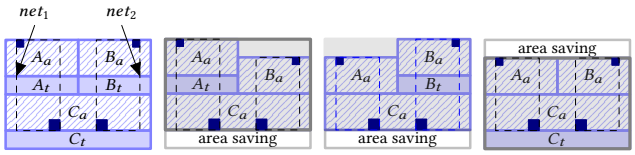
3.2.1 Preliminary Problem Formulation. Given the well tap graph $G(P)$ for a placement P, a tap vertex is said to **cover** an active vertex if there exists an edge between the two vertices. The task of well tap optimization is to find an *optimal* set S that covers all the active nodes and retain them. One simplistic definition of the optimal S is the set with least cardinality that covers all the active vertices. If a well tap node $t \in S' = T \setminus S$, it implies that all the active nodes that are covered by t are covered by one or more vertices in S. This indicates that the well taps corresponding to the vertices in S' are redundant and can be removed from the layout, i.e., the associated tapped cell is made tapless. Since S has the smallest cardinality, removing well taps in S' would superficially appear to yield an optimal layout. However, this optimization involves further subtleties, and as we will show below, removing all elements in S' does not necessarily provide an optimal layout.

3.2.2 Area and HPWL. To recognize the sources of suboptimality arising from using S with least cardinality, we first need to understand the cost function used in analog layout synthesis. Typical analog placers [1, 7, 9, 12, 15] generate an optimal layout by minimizing the following cost function F for placement P:

$$F(P) = \lambda_1 \cdot f_1(A(P)) + \lambda_2 \cdot f_2(L(P)) \quad (1)$$

where $A(P)$, the area of P, is the area of the smallest bounding box of all cells; $L(P)$ is the sum of the half-perimeter wire-lengths

[1] Note that this circle graph is **different from the well tap graph** defined in Section 2.2 and is used to find minimum number of legal well islands.



(a) All taps (b) $S = \{A_t\}$ (c) $S = \{B_t\}$ (d) $S = \{C_t\}$

Figure 7: Removal of well taps vs area/HPWL saving; Shaded region represents the bounding box used for area calculation; Dashed lines represent bounding boxes of nets net_1 and net_2 used for HPWL calculation.

(HPWLs) of all the nets in the design; λ_i are the weights that indicate the relative importance of area and wire-length; and f_i are the normalization functions that map $A(P)$ and $L(P)$ to the range [0,1].

To show that the set S does not necessarily translate to maximal area and HPWL savings, which are the primary metrics used in (1), we consider the layout in Figure 7 with three blocks, A, B, and C. The figure shows the bounding boxes of net_1 connecting A and C and net_2 connecting B and C. Figure 7(a) shows the a layout where all blocks have well taps. If the radius $\mathcal{R}_w$ dictates that only one of these blocks is required to retain the well tap, then S could either be $\{A_t\}$, $\{B_t\}$ or $\{C_t\}$. Figures 7(b), (c), and (d) show the compacted layout obtained on retaining each of candidate sets S. The area $A(P)$ of the layout is the same in Figures 7 (b), (c), and (d), but the HPWL improvement depends on the choice of S. It can be seen that the improvement in HPWL of net net_1 and net_2 for $S = \{C_t\}$ (Figure 7(d)) is better than the other two cases. When A_t or B_t is retained, the HPWL of one net reduces while that of the other is unaltered, but when C_t is retained, the HPWL of both net_1 and net_2 is reduced. Thus, the impact of removal of well tap nodes on the area and HPWL must be considered to find an optimal S.

Figures 7(b)–(d) also illustrate the factors to consider while calculating the area impact of tap node removal. The *void strip* created by removing C_t from Figure 7(a) corresponds to the area saved in Figures 7(b) and (c); the void strip created when A_t and B_t are removed together represents the area saved in Figure 7(d). Thus, we save area if the removed taps align to create a void strip.

HPWL saving is possible when a tap node that is to be removed intersects the bounding box of a net and slices the box into two rectangles. For example, A_t intersects the bounding box of net_1 in Figure 7(a) and bisects it into two rectangles, one each lying in A_a and C_a. The removal of A_t thus leads to reduction of HPWL of net_1, as seen in Figure 7(b). Such an analysis is used to estimate the area and HPWL impact of removing well tap nodes.

3.2.3 Symmetry. A primary constraint used in analog circuit layouts is symmetry between blocks whose performance needs to be matched. *Self-symmetry* specifies line(s) of symmetry for a block such that its subhierarchies must be symmetric about the line(s). *Symmetry pairs* define pairs of blocks (typically of the same size) that must be placed symmetrically about a line of symmetry. A symmetry group is a collection of self symmetry and symmetry pair constraints that share a common axis of symmetry.

Figure 8(a) shows an example layout satisfying a symmetry group with a vertical axis of symmetry, with a symmetry pair $\{A, B\}$ and self-symmetry for C. Upon removal of the tap B_t of cell B, A and B are no longer symmetrical as seen in Figure 8(b). The asymmetry

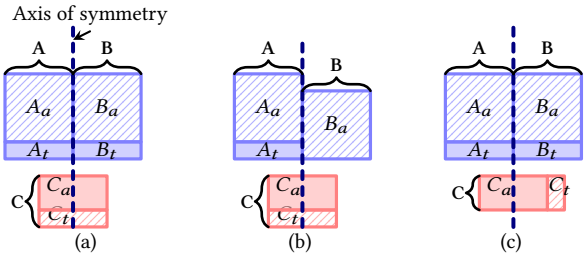


Figure 8: Symmetry constraint honoring layout in (a); symmetry violation in (b) upon tap removal; self-symmetry constraint violation in (c).

(a) Well tap at corner (b) Well tap at center

Figure 9: Centered well taps in (b) are preferred over (a).

arises from removal of tap from one of the vertices in a symmetry pair. This implies that the two blocks belonging to a symmetry pair need to simultaneously reject or retain the well taps in the layout.

For a self symmetric block, the tap vertex is forced to be symmetric in the input placement by forcing the placer to use the variant of the block with a horizontal (vertical) tap for a vertical (horizontal) axis of symmetry. This ensures that that the scenario in Figure 8(c) does not occur. Thus, tap assignment must simply ensure that symmetry pair constraints are honored by retaining/removing the corresponding well taps simultaneously.

3.2.4 Centering. AMS designers prefer placing the well taps approximately equidistant from all the devices covered by the tap so that the largest bulk impedance to any of the devices is minimized, making the tap maximally effective for all cells. Figure 9(a) and (b) show two different valid well tap locations that satisfy the radius constraint: between these two variants, Figure 9(b) is preferred since the largest distance from the well tap to any block is lower.

3.2.5 Refined Formulation. From the observations in Sections 3.2.2–3.2.4, we develop a new formulation for the optimal set S of well tap vertices that incorporates area and HPWL saving, symmetry constraints and tap centering. We use an integer linear program that maximizes a weighted sum objective to identify S. The objective function of this ILP optimizes the area savings, HPWL savings, and tap centering, while the constraints enforce symmetry and ensure that all transistors are covered by a well tap.

If $x_i \in \{0, 1\}$ is the indicator variable corresponding to the presence of a tap node $i \in T$ in S, the objective and constraints are:
(1) Well tap coverage: Every active vertex in the graph G must be adjacent to at least one vertex in S to ensure that it has a well tap within distance $\mathcal{R}_w$. For each edge $e_{i,j} \in E$ between an active cell i and tap cell j, this is formulated as the ILP constraint:

$$\sum_{\forall (e_{i,j}) \in E} x_i \geq 1 \tag{2}$$

(2) Area and HPWL cost: The ILP formulation begins with a configuration where all cells have well taps and then determines the set of tap vertices to be removed. We can save area by only removing vertices that align in the layout horizontally or vertically, such that their removal creates a void strip. Let Q be the *set of sets* of all tap

vertices whose well taps align, such that they create a void strip when removed: in Figure 7(a), $Q = \{\{A_t, B_t\}, \{C_t\}\}$. Let h_k and w_k represent the height and width of a void strip formed by removal of $q_k \in Q$. If two different well type islands, one each on top and bottom (left and right) of a horizontal (vertical) void strip have a separation $sep < h_k + d_w$ ($sep < w_k + d_w$), then the removal of the void strip entirely would violate the well-separation requirement. To honor the well-separation and save the area, instead of removing the entire horizontal or vertical void strip a slice can be removed from the strip, whose dimensions (w_{s_k}, h_{s_k}) are given by:

$$\{w_{s_k}, h_{s_k}\} = \begin{cases} \{w_k, \min(h_k, h_k + d_w - sep)\} & \text{horizontal void} \\ \{\min(w_k, w_k + d_w - sep)\, h_k\}, & \text{vertical void} \end{cases} \quad (3)$$

The area saved by removing all tap vertices in set q_k, $\Delta Area_k = w_{s_k} \times h_{s_k}$. We use a single variable x_{q_k} to represent all tap cells in set q_k, so that they are all retained or removed together.

If the HPWL bounding boxes of N_i nets are sliced by tap node $i \in q_k$, the HPWL savings, ΔL_k, by removing all the taps in q_k are:

$$\Delta L_k = \left(\sum_{i \in q_k} N_i\right) \times \begin{cases} w_{s_k}, & \text{vertical void} \\ h_{s_k}, & \text{horizontal void} \end{cases} \quad (4)$$

The total area and HPWL saving upon removing all the vertices in $q_k \in Q$ can be expressed as:

$$x_{q_k} \times \left(\lambda_1 \Delta Area_k + \lambda_2 \Delta L_k\right) \quad (5)$$

where λ_1 (λ_2) is the importance of reducing area (HPWL), as in (1). (3) Symmetry: If i and j are two tap vertices are two cells in a symmetry pair, then both tap nodes must be simultaneously retained or rejected by using the same ILP variable x_i for both tap cells. (4) Centering: The *cover set* C_i of tap vertex i is the set of all active nodes that are adjacent to i in G. The range of the cover set, r_i, is the the maximum Euclidean distance between i and its *cover set*. We augment the cost function to minimize r_i to incentivize the retention of tap vertices that are near the center of each cover set.

Combining all of the above, we formulate the following ILP:

$$\arg\max_{x_i} \sum_{q_k \in Q} x_{q_k} \left(\lambda_1 \Delta Area_k + \lambda_2 \Delta L_k\right) - \sum_{i \in T} \lambda_3 x_i r_i \quad (6)$$

$$\text{such that} \sum_{\forall (e_{i,j}) \in E} x_i \geq 1, \quad \begin{vmatrix} \forall j \in A \\ x_i \in \{0,1\},\ \forall i \in T \end{vmatrix}$$

$$S = \{i | x_i = 1, i \in T\} \quad (7)$$

where λ_3 indicates the relative importance of tap centering. We obtain the set of retained well tap nodes from (7).

3.2.6 Computational Complexity. Our core ILP formulation, without pruning variables (e.g., using symmetry or void strip considerations) can be mapped to the *minimum weight dominating set* (MWDS) problem on a bipartite graph, an NP-complete problem [6].

Specifically, a set of vertices $D \subseteq V$ is a dominating set of G if every vertex in V is either in D or adjacent to one or more vertices in D. If every vertex $i \in V$ is associated with a weight w_i, then the minimum weight dominating set (MWDS) is a dominating set with smallest sum of weights. For example, for the layout of the telescopic OTA in Figure 3(a), there are no symmetry or void strip constraints, S becomes a MWDS of G with the constraint that S is a subset of the set of tap cells, T, instead of the entire vertex set, V. The weight of vertex $i \in T$, w_i for this MWDS problem is:

$$w_i = (\lambda_1 \Delta Area_i + \lambda_2 \Delta L_i - \lambda_3 r_i) \quad (8)$$

Since MWDS is NP-complete, the complexity of its best-known optimal solution is exponential in $|V|$. We require $S \subseteq T$ instead of $S \subseteq V$, but this merely reduces the complexity to be exponential in $|T|$ instead of $|V|$. In practice, several factors ensure that the $|T|$ is not large, making the problem tractable: (1) void strip alignment constraints and symmetry pair constraints reduce $|T|$; (2) the use of hierarchy (Section 3.4) reduces the problem size at each hierarchical level. Finally, another mitigating factor is the structure of the problem: the dense connections in the well tap graph are localized within the Euclidean radius R_w of tap cells. This yields a sparse block-diagonal ILP constraint matrix. Typical ILP solution methods (e.g., branch-and-cut) solve an LP relaxation of the problem, and benefit from the sparsity and structure of this matrix.

(a) Before (b) After

Figure 10: Routes being preserved after removal of redundant tap vertices; Optimal well taps $S = \{A_t, C_t\}$.

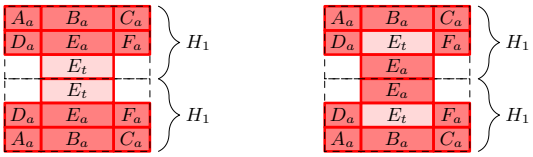
3.3 Well Island and Well Tap Layout

The goal of solving the ILP (6) is to identify well islands and well tap locations, so that the cost function for candidate placement P in the stochastic placement engine corresponds to a legal solution. At this stage, we arrive at a placement that can retain only the tap vertices in S, form legal well islands. We shift the locations of blocks in the placement to recover the area associated with unused tap cells, ensuring that well spacing requirements are met. If a tap cell of height h is deleted, its area is reclaimed by the layout by moving blocks above it downwards by distance h. A similar operation is used to move blocks to the left when a horizontal tap cell is deleted. The area and HPWL of the layout is estimated after this operation.

This analysis is sufficient to determine the cost function for the placer during an intermediate iteration, and it is not necessary to translate this solution to a layout until the very end, as a post-processing step after the optimal sequence pair configuration is selected. At this stage, the tap and active vertices must be connected to ensure all the required well islands are generated. The edges between vertices were guaranteed to be routable during the construction of G as described in Section 3.1. By construction, the reduction of redundant taps, which creates void strips that traverse the block, will ensure that the the updated layout remains routable. This is illustrated in Figure 10(a) with three tapped cells A–C, where the optimal set $S = \{A_t, C_t\}$ and B_t can be removed. Figure 10(b) shows the updated layout after removing B_t. In both figures, the hatched L-shaped region connecting C_t and B_a is routable, and the route is shortened after removing the redundant tap cell.

3.4 Hierarchical Placement

AMS designs are typically placed hierarchically, because design blocks tend to have a small number of components that can be



(a) Well tap at center (b) Well tap at boundary

Figure 11: Suboptimality in hierarchical placement; H_2 hierarchy has two instances of H_1 one above another.

logically grouped. Hierarchical placement can be visualized as a tree like structure, where each node represents a level of hierarchy and the children of the node are the blocks to be placed at that hierarchy level. The VCO in (Figure 1(b)), has blocks s_1–s_4 at the first level of hierarchy, and inverter stages at the next level.

In fact, the use of hierarchy in placement is beneficial for the ILP formulation in (6), since a smaller problem is solved at every hierarchy. In practice, we impose a time limit for the solution of the ILP to avoid long runtimes, and the ILP solution may not be optimal: reducing the problem size increases the likelihood of obtaining an optimal layout at each level. Placement at any hierarchical level is unaware of the global structure, and the locally optimum solution at some level of hierarchy may not be globally optimum for the overall layout. The well tap formulation inherits this problem for hierarchical placement: well taps that are optimal for the current hierarchy level may be suboptimal for the global layout.

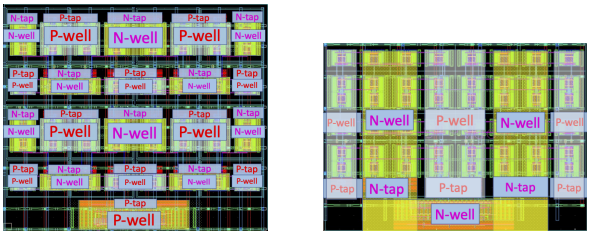
We illustrate two scenarios in Figure 11 using a layout that has two levels of hierarchy: H_1 and H_2, where H_1 has six cells (A-F) and H_2 has two instances of H_1 that are placed one above another, with mirror symmetry about the horizontal axis that separates them. In this case, the tap nodes E_t in Figure 11(a) can be combined to retain a single well that covers all the devices in H_2, resulting in overall savings in the area and potentially HPWL. A second scenario is shown in Figure 11(b), where no such union is possible. Thus, any tap nodes that are close to the boundary in a hierarchy may possibly combine with other tap node or can potentially cover active nodes in another hierarchy. Using this intuition, we add a distance term to the cost function in (6) that incentivizes the retention of wells that are closer to the boundary of a hierarchical block. If d_i is the Euclidean distance between node i and the boundary, the new ILP cost function, using another importance factor λ_4, is:

$$\max_{x_i} \sum_{q_k \in Q} \left(x_{q_k}\left(\lambda_1 \Delta Area_k + \lambda_2 \Delta L_i\right)\right) - \sum_{i \in T} x_i(\lambda_3 r_i - \lambda_4 d_i) \quad (9)$$

4 RESULTS

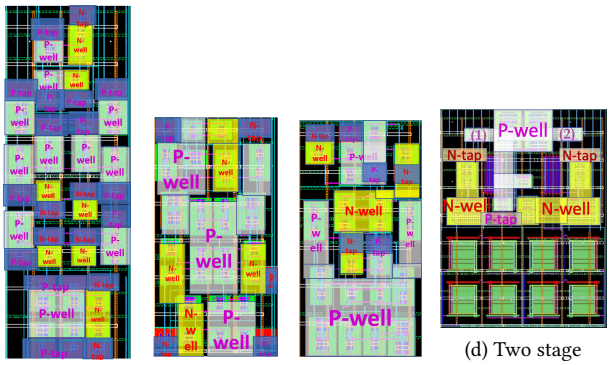
The stochastic placement algorithm, including well island generation and tap sharing optimization, are implemented in C++ and compiled using GCC 8.2.0. The ILP solver lp_solve [2] is used to find the optimal number of taps using the formulation in (7). A time limit of 1ms is set for the solver to arrive at an optimal solution. The layouts are generated using a commercial 12nm PDK on a Linux server with Intel Xeon(R) 2.20GHz Silver 4114 processors with 160GB memory. The tools Cadence Spectre, Calibre nmLVS and Calibre xACT are used for circuit simulation, layout vs schematic checking and parasitic extraction of the layouts respectively.

Comparisons We compare our approach, which generates well islands and optimal well-taps *during placement*, against:



(a) Approach (A) (b) Proposed approach

Figure 12: Comparison of layouts of differential ring oscillator based VCO generated using various approaches.



(a) Approach (A) (b) Approach (B) (c) Proposed (d) Two stage differential OTA

Figure 13: (a)–(c) Comparator layouts from various approaches. (d) Non-rectangular islands built by our approach.

Approach (A), which generates layouts using a layout generator based on [4], with built in well taps for each cell in the layout. These layouts honor $\mathcal{R}_w$ constraints by treating each cell as an island with its own tap and do not need a separate well island generation step. Approach (B), which uses the placer [4] to generate layouts without any well taps, and then, *for that specific placement,* manually generates optimal well islands (minimum number of islands with optimal HPWL and area) and inserts well taps. This is the best achievable result from an approach (e.g., WellGAN [17]) that generates well islands and insert well taps *after* optimal well-oblivious placement.

From Table 1, the area and HPWL for our method are much better than Approach (A), and sometimes noticeably superior to the manual Approach (B). Post-layout performance metrics (Table 2) from our method are generally superior to Approaches (A) and (B). For all tested layouts, the ILP solver is able to find an optimal solution in every iteration of the placement within the 1ms limit.

Figure 12 compares the layout of the differential ring oscillator based VCO in Figure1(b), generated using Approach (A) and our proposed approach. Approach (B) generates a similar layout as Approach (A), and it is not separately shown here. We achieve 23% lower area and 11% lower HPWL than Approaches (A) and (B). This translates to a 49% improvement in the maximum frequency of the VCO. The maximum frequency depends on the parasitics between the transistors, which are reduced in our approach. The placement from Approach (A) and (B) results in a suboptimal layout (as in the figure at left in Figure 1(c),(d)) since it does not consider the impact of well islands, their separation and tap location during placement. Both layouts have optimal power routing, and it is the improved signal routing that is the cause of performance enhancement.

Table 1: Comparisons of Area and HPWL against Approaches (A) and (B); ΔArea, ΔHPWL are the area and HPWL savings, respectively; ΔT_{Placer} and ΔT_{Total} are the runtime overhead in the placer and in the overall layout generation flow, respectively.

Circuit	Approach (A)				Approach (B)		Proposed				ΔArea		ΔHPWL		ΔT_{placer}	ΔT_{total}
	Area (μm^2)	HPWL (μm)	T_{placer} (s)	T_{total} (s)	Area (μm^2)	HPWL (μm)	Area (μm^2)	HPWL (μm)	T_{placer} (s)	T_{total} (s)	vs. (A)	vs. (B)	vs. (A)	vs. (B)	vs. (A), (B)	vs. (A), (B)
Eight-stage VCO	195.08	104.42	44	65	160.25	94.70	160.25	94.70	57	72	17.9%,	0.0%	9.3%,	0.0%	29%	11%
Differential ring oscillator based VCO	393.45	167.95	90	141	393.45	167.95	301.00	149.35	137	173	23.5%	23.5%	11.1%	11.1%	52%	23%
Comparator	171.91	103.17	21	52	121.62	83.76	116.21	74.18	22	43	32.4%	4.4%	28.1%	11.0%	4%	-17%
High speed comparator	224.89	101.58	74	120	184.73	93.90	184.73	93.90	99	121	17.9%	0.0%	7.6%	0.0%	34%	1%
Five transistor high frequency OTA	79.23	58.46	11	18	71.56	52.26	71.56	52.26	15	21	28.3%	0.0%	10.6%	0.0%	38%	20%
Cascode current mirror OTA	171.86	85.35	27	40	145.78	81.69	141.89	77.94	35	46	17.4%	2.7%	8.7%	4.6%	28%	17%
Folded cascode OTA	59.61	42.38	57	90	49.67	37.58	49.67	37.58	87	109	20.0%	0.0%	11.3%	0.0%	52%	21%
Two stage differential OTA	284.33	134.20	35	51	255.24	121.43	255.24	123.98	46	62	10.2%	0.0%	7.6%	-2.1%	30%	21%
LDO error amplifier	169.25	154.89	31	40	145.15	150.18	145.15	150.18	42	51	14.2%	0.0%	3.0%	0.0%	35%	29%

Figures 13(a)–(c) compare our layout for a comparator against Approaches (A) and (B). We save 32% area and 28% HPWL over Approach (A). Relative to the manual Approach (B), we achieve slightly lower area and an 11% HPWL improvement, primarily in power routing. This HPWL reduction reduces routing congestion and leads to a small performance improvements.

The layout for a two stage differential OTA in Figure 13(d) illustrates more complex well structures built using our approach. This layout has straight-lined well "routes" (left N-well island), L-shaped "routes" (right N-well island; P-well island). Symmetry constraints are maintained, e.g., when P-taps of symmetric devices (1), (2) are removed and their wells are merged with the larger P-well island.

Table 1 compares the improvement in area and HPWL achieved using our proposed approach. The results are demonstrated for various classes of analog and mixed signal designs: VCO, comparator, OTA and error amplifier. The ΔArea and ΔHPWL column show the difference in area and HPWL with respect to the approaches (A) and (B). The area and HPWL savings using the proposed approach are apparent from the columns ΔArea and ΔHPWL of this table. The ILP formulation takes each candidate placement and incurs a runtime overhead in solving it to find the optimal well taps. The table shows both the absolute placer runtime, T_{Placer}, and the percentage difference, ΔT_{Placer}, between the placer runtime of our approach and Approach (A). The placer runtimes for Approaches (A) and (B) are similar. While our method involves a small increase in the runtime, the end result is a legal placement with well island formation, and well taps that obey the $\mathcal{R}_w$ tap constraints. The gain in area and HPWL over the baseline justify the increased runtime.

When we compare the total runtime, T_{total}, of the physical design flow, including routing (shown in Table 1), the percentage change in runtime, ΔT_{total} with respect to Approach (A) is more modest. In fact, the removal of the redundant well taps aids in reducing the obstacles for the routing and can actually improve the runtime for routing: for cases like comparator, this causes a net *reduction* in the total runtime. The overall runtime of Approach (B) is significantly larger (hours vs. minutes) due to the manual effort involved.

5 CONCLUSION

A graph-based automated well island generation and well tap insertion algorithm for AMS circuits is proposed. This algorithm estimates the true cost of a layout during placement to achieve an optimal solution. Comparisons between the proposed algorithm and existing approaches using post-layout simulation of various classes of AMS circuits demonstrate the efficacy of this approach.

Table 2: Performance comparison of the layouts generated with and without using the well islands and well taps

Circuit	Performance metric	Approach (A)	Approach (B)	Proposed approach
High-speed comparator	Evaluation delay (ps)	64	60	60
	Precharge delay (ps)	48	47	47
Comparator	Evaluation delay (ps)	192	153	151
	Precharge delay (ps)	149	119	116
Five-transistor high-frequency OTA	DC gain (dB)	22.13	22.13	22.13
	3-dB freq. (MHz)	335	343	343
	UGF (GHz)	4.08	4.18	4.18
Differential ring oscillator based VCO	Max. freq. (GHz)	3.16	3.16	4.71
	Min. freq. (MHz)	311.90	311.90	327.13
	Voltage range (V)	[0,0.5]	[0,0.5]	[0,0.5]
Folded cascode OTA	DC gain (dB)	13.40	13.75	14.80
	3-dB freq(GHz)	2.39	2.68	2.64
	UGF(GHz)	7.60	7.77	8.05
Two-stage differential OTA	DC gain (dB)	44.4	45.0	44.7
	3-dB freq(MHz)	4.04	3.52	4.41
	UGF(MHz)	685	810	798

REFERENCES

[1] Balasa, F., Maruvada, S., and Krishnamoorthy, K. Using red-black interval trees in device-level analog placement with symmetry constraints. In *Proc. ASP-DAC* (2003).

[2] Berkelaar, M., Eikland, K., and Notebaert, P. lp_solve 5.5, Open source (mixed-integer) linear programming system. Version 5.1.0.0 dated 1 May 2004.

[3] Cohn, J., Garrod, D., Rutenbar, R., and Carley, L. KOAN/ANAGRAM II: New tools for device-level analog placement and routing. *JSSC 26*, 3 (1991).

[4] Dhar, T., Kunal, K., Li, Y., Madhusudan, M., Poojary, J., Sharma, A. K., Xu, W., Burns, S. M., Harjani, R., Hu, J., et al. ALIGN: A system for automating analog layout. *IEEE Des. Test 38*, 2 (2020).

[5] Farbiz, F., and Rosenbaum, E. Modeling and understanding of external latchup in CMOS technologies Part I: modeling latchup trigger current. *IEEE T. Device Mater. Rel. 11*, 3 (2011).

[6] Fomin, F. V., Grandoni, F., Pyatkin, A. V., and Stepanov, A. A. Bounding the number of minimal dominating sets: a measure and conquer approach. In *Proc. ISAAC* (2005), p. 573–582.

[7] Koda, S., Kodama, C., and Fujiyoshi, K. Linear programming-based cell placement with symmetry constraints for analog IC layout. *TCAD 26*, 4 (March 2007).

[8] Lim, A., Thanvantri, V., and Sahni, S. Planar topological routing. *TCAD 16*, 6 (1997), 651–656.

[9] Ma, Q., Xiao, L., Tam, Y.-C., and Young, E. F. Y. Simultaneous handling of symmetry, common centroid, and general placement constraints. *TCAD 30*, 1 (2011).

[10] Marek-Sadowska, M., and Tarng, T. T.-K. Single-layer routing for VLSI: analysis and algorithms. *TCAD 2*, 4 (1983), 246–259.

[11] Massier, T., Graeb, H., and Schlichtmann, U. The sizing rules method for CMOS and bipolar analog integrated circuit synthesis. *TCAD 27* (December 2008).

[12] Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y. Rectangle-packing-based module placement. In *Proc. ICCAD* (1995).

[13] Nakatake, S., Kawakita, M., Ito, T., Kojima, M., Kojima, M., Izumi, K., and Habasaki, T. Regularity-oriented analog placement with diffusion sharing and well island generation. In *Proc. ASP-DAC* (2010).

[14] Supowit, K. Finding a maximum planar subset of a set of nets in a channel. *TCAD 6*, 1 (1987), 93–94.

[15] Tam, Y.-C., Young, E. F., and Chu, C. Analog placement with symmetry and other placement constraints. In *Proc. ICCAD* (2006).

[16] Troutman, R. R. *Latchup in CMOS technology: the problem and its cure.* Kluwer Academic Publishers, Norwell, MA, 1986.

[17] Xu, B., Lin, Y., Tang, X., Li, S., Shen, L., Sun, N., and Pan, D. Z. WellGAN: Generative-adversarial-network-guided well generation for analog/mixed-signal circuit layout. In *Proc. DAC* (2019).

[18] Yang, Y., He, J., and Manohar, R. Dali: A gridded cell placement flow. In *Proc. ICCAD* (2020).