# An Extended Study of Gear-Ratio-Aware Standard Cell Layout Generation for DTCO Exploration

Chung-Kuan Cheng, *Fellow, IEEE,* Andrew B. Kahng, *Fellow, IEEE,* Bill Lin, *Senior Member, IEEE,* Yucheng Wang, *Graduate Student Member, IEEE,* and Dooseok Yoon, *Graduate Student Member, IEEE*

***Abstract*—Advanced nodes decouple contacted poly pitch (CPP) and lower-metal pitch to improve routability. We present CPCell, an efficient standard-cell layout generation framework to support arbitrary gear ratio (GR) and offset parameters through a fine-grained layered grid graph and constraint-programming-based placement–routing co-optimization. Layout quality is improved via Middle-of-Line routing, M0 pin enablement, pin accessibility constraints and a weighted multi-objective formulation that jointly optimizes cell layouts. To scale to netlists with up to 48 transistors, we incorporate acceleration techniques including transistor clustering, identical transistor partitioning, routing lower bound tightening and early termination strategies. Comprehensive cell-level and block-level studies are conducted to evaluate GR and offset choices, quantify the benefits of the proposed objectives and assess their impact on power, performance, area and IR-drop outcomes.**

***Index Terms*—Standard-cell layout, design-technology co-optimization, constraint programming, pin accessibility.**

## I. Introduction

Early nodes assumed uniform scaling between contacted poly pitch (CPP) and metal pitch. Advanced nodes decouple CPP and lower-metal pitch to reduce routing congestion on lower metals [1]. As metal pitch scales faster than CPP [2], the CPP-to-metal-1 (M1) pitch ratio, called *gear ratio* (GR), becomes a key design-technology co-optimization (DTCO) parameter [3]. GR affects cell-level routability and block-level implementation quality. Reducing the M1 pitch increases the number of M1 tracks and reduces routing congestion. An *offset* specifies the distance between the leftmost poly and M1 columns. Shifting the M1 grid aligns local and global routing grids, enabling placement legalization under complex GR settings. Figure 1 compares M1 routing resources within a 7-CPP cell: 1:1 GR [4] provides six usable columns (a), 3:2 GR provides 10 (b), and 5:3 GR provides 11 without offset or up to 12 with offset (c). Non-1:1 GRs can misalign the poly and M1 grids [5]; offset variants restore alignment, preventing block-level legalization failures and enabling accurate technology evaluations.

Prior work studies gear ratio and offset, mainly using post-processed layouts or backside power delivery. Jeong et al. [6] propose a non-1:1 gear-ratio-aware DTCO framework that improves pin accessibility using post-placement pin relocation and post-routing stretching. Their method reduces routing violations without increasing wirelength, timing or power.

This work was supported in part by Logic Pathfinding Lab, Samsung Semiconductor Inc., in part by NSF under Grant CCF-2110419, in part by DARPA under Grant HR0011-18-2-0032, and in part by the C-DEN Center. *(Corresponding author: Yucheng Wang.)*

The authors are with the University of California San Diego, La Jolla, CA 92093 USA (e-mail: yuw132@ucsd.edu).

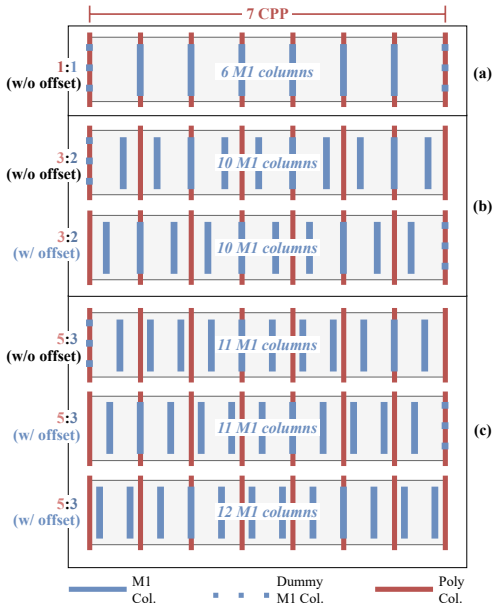



Fig. 1. M1 routing resources in a 7-CPP cell for (a) 1:1 GR, (b) 3:2 GR and its offset variant, and (c) 5:3 GR and its two offset variants. Dummy columns at both cell boundaries prevent overlap with adjacent cells; all other columns are available for routing.

However, because it introduces gear ratio effects as post-processing, it does not model non-1:1 GR effects during cell generation. Cheng et al. [7] integrate non-1:1 gear ratios into SMT-based standard cell generation with backside power delivery, but they evaluate only area. Comprehensive studies in sub-7 nm technology options [8] that jointly evaluate GR and offset choices on both cell-level and block-level metrics are still limited. This paper extends prior work [5], SMTCell, with critical DTCO features summarized in Table I. CPCell inherits the relative-layered-grid-graph placement-and-routing formulation from SMTCell [5] [9]. We introduce CPCell with the following key extensions.

(1) *CPCell Framework:* CPCell is a standard-cell layout generation tool and a continuation of the prior SP&R framework [4] and various other prior works [10] [11] [12] with an emphasis on customization and block-level power, performance and area (PPA). It is compatible with CP-SAT solver from OR-Tools [13], which employs an integer- and SAT-based encoding compatible with previous SMT formulations in Z3 [14] [15]. The workflow of CPCell is depicted in Figure 2. Table I summarizes the design-technology co-optimization (DTCO) capabilities of representative standard-cell layout generation frameworks.

Compared to prior SMT-, MILP- and heuristic-based approaches, CPCell is the only framework that supports arbitrary gear ratios and offset variants under PROBE3.0 Process Design Kit (PDK) [8] when solved to exact certification. Under the early-termination (Section V-D), this guarantee applies to the primary objective, cell width.

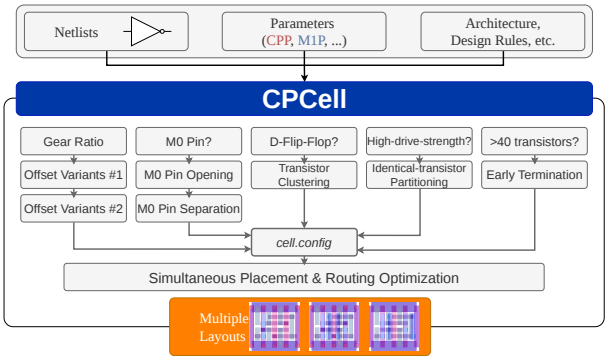



Fig. 2. CPCell workflow. Inputs are the cell netlist, design rules and GR. Per-cell configurations specify offset variants, M0 pins and acceleration settings. Enumerating all offsets produces multiple layout candidates.

(2) *Gear Ratio & Offset:* Flexible metal pitch and offset parameters modulate routing resources and cell- and block-level metrics (Sections VI and VII). Prior work [5] explores 3:2 and 5:3 GR without offset variants, causing legalization issues that limit PPA. Under 3:2 GR, a 3-CPP cell aligns with the global grids (Figure 3(a)), whereas a 2-CPP cell can violate the routing grid (b). An M1 offset restores alignment (c). CPCell enumerates offset options for each netlist to cover all legal grid alignments and generates the corresponding layouts.

(3) *Layout enhancement:* Prior work relies primarily on M1 for vertical routing [5] [8]. This reliance creates routing blockages and reduces block-level routability. We add Middle-of-Line (MOL) routing for intra-cell connectivity [16] [17] and enable M0 pins to improve block-level access. Figure 4 compares DFFHQN_X1 layouts under a 3:2 GR. CPCell reduces cell width by four CPP and enables M0 pins without adding M1 routing.

(4) *Acceleration Techniques:* GR and offset increase routing complexity [5]; for example, prior D flip-flop flows require user-given partitioning. Our complementary techniques exploit netlist structure and CP-SAT behavior: *transistor clustering* identifies structural regularities; *identical transistor partitioning* (ITP) removes ordering redundancy; *routing lower bound tightening* (RLBT) accelerates optimality certification; and *early termination with a relative optimality gap* trades optimality for scalability. Together, they reduce runtime by up to 88.96% and scale to 48-transistor netlists.

Section II introduces the proposed graph-based framework for constraint formulation. Section III presents the objective functions. Section IV introduces M0 pin accessibility constraints. Section V describes the acceleration techniques used to reduce solver runtime. Section VI evaluates the impact of gear ratio and offset at cell-level using various layout metrics. Section VII assesses block-level implications in terms of PPA and IR-drop. Our code, cell layout, Liberty, and evaluation scripts are open source and available on GitHub [18].

## II. Constraint Formulation

Recall that *gear ratio* (GR) is the ratio between (contacted) poly and M1 pitches. Common GR values include integer or half-integer ratios, such as 1:1 or 3:2.

Given any GR parameter, our tool dynamically formulates constraints based on a graph structure. In terms of constraint formulation, we incorporate largely the same set of constraints as SP&R [4]. The inherited fine-grained relative layered grid graph captures the gear-ratio details required for constraint formulation. Our framework treats pitches as variables to construct a relative layered grid graph, representing a potential cell layout.

In our graph structure, we consider a four-layer design[1]: placement layer $L^1$ (vertical), M0 layer $L^2$ (horizontal), M1 layer $L^3$ (vertical), and M2 layer $L^4$ (horizontal). Since our formulation relies on vertex-based constraints, accurately capturing the varying distances between vertices is crucial for satisfying complex design rules [19]. Table II summarizes the symbols and notation used in the formulation.

### A. Relative Layered Grid Graph Construction

Before graph construction, we determine the total cell width and layer-specific column and row sets. The horizontal boundary $w_{\text{total}}$ depends on the allowable diffusion break count ($c_{db}$), contacted poly pitch ($mp^1$), and accumulated PMOS ($w_p$) and NMOS ($w_n$) widths:

$$w_{\text{total}} = \max(w_p, w_n) + c_{db} \cdot mp^1, w_p \in \mathbb{Z}^+, w_n \in \mathbb{Z}^+. \quad (1)$$

Let $i$ be the index of a layer. We discretize the layout area into a grid of valid coordinate points. These points correspond to the intersections of physical routing tracks. Let $C^i$ and $R^i$ denote the sets of valid x-coordinates (columns) and y-coordinates (rows), respectively, for each layer $L^i$. Our targeted cell layout is unidirectional, meaning that each layer has tracks running in a direction orthogonal to that of its neighbor layer(s), with $mp^i$ as the pitch between adjacent tracks. Therefore, a horizontal layer shares columns with its adjacent vertical layer(s); a vertical layer shares rows with its adjacent horizontal layer(s). Since gear ratio is only defined between vertical layers, horizontal layers must have the same pitch (i.e., $mp^2 = mp^4$) and thus all layers must share the same set of rows,

$$R^1 = R^2 = R^3 = R^4 = \{\, h \cdot mp^i \mid h \in \mathbb{Z}^+,\ h \le t \,\}, \quad i = 2 \text{ or } 4. \quad (2)$$

Creating the column sets $C^i$ must allow for $mp^1 \neq mp^3$ and must consider offsets $\delta$. Figure 5 illustrates how the $mp^1$ and $mp^3$ values can result in an irregular pattern of columns on $L^2$. The column set $C^2$ must contain all columns from both $C^1$ and $C^3$ to represent the densely placed routing tracks, with overlapped columns being merged to avoid redundancy. We achieve this using Algorithm 1, which iteratively creates

[1]While this work utilizes a four-layer stack consistent with the target PDK, our proposed framework generalizes to arbitrary layer counts, subject to increased computational complexity.

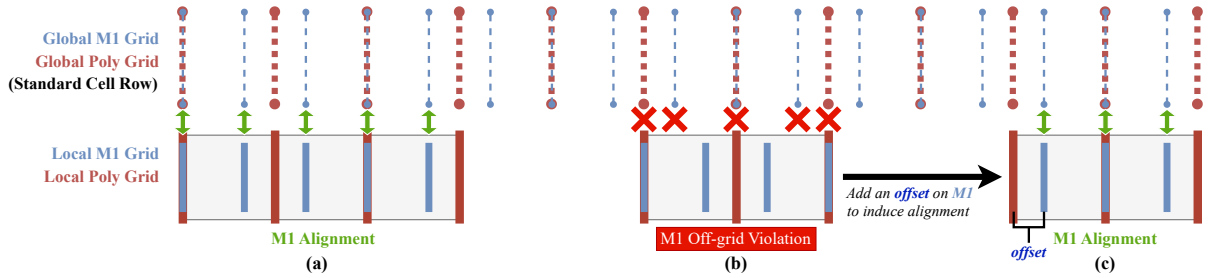



Fig. 3. Placement under 3:2 GR. (a) A 3-CPP cell aligns with the global grids. (b) A 2-CPP cell is misaligned and requires either relocation, which burdens routing, or an M1 offset. (c) An offset aligns the 2-CPP cell at the same placement site as (b).

TABLE I
SUMMARY OF AVAILABLE DTCO FEATURES IN STANDARD-CELL LAYOUT GENERATION TOOLS.

| Framework | SP&R [4] | PROBE3.0 [8] | Csyn-fp [20] | NVCell [21] [22] | SO3 [23] | Au-Medal [17] | SMTCell-GR [5] | Our Work (CPCell) |
|---|---|---|---|---|---|---|---|---|
| **Arbitrary Gear Ratio** | No | No | No | No | No | No | **Yes** | **Yes** |
| **Arbitrary Offset** | No | No | No | No | No | No | No | **Yes** |
| **Method** | SMT | SMT | Tree Search, DP, SMT, MILP | SA, RL, GA | MILP | SMT | SMT | CP |
| **Optimality Condition** | Δ (Prioritized) | Δ (Prioritized) | Δ (Sequential) | Δ (Sequential) | **Yes** | Δ (Routing) | Δ (Prioritized) | **Yes** |
| **MOL Routing** | No | No | **Yes** | Unspecified | **Yes** | **Yes** | No | **Yes** |
| **Parameterizable Gate Cut** | No | No | No | No | No | No | No | **Yes** |
| **Acceleration Technique** | • Symmetry breaking<br>• Datapath-aware partitioning | • Symmetry breaking<br>• Datapath-aware partitioning | • Diffusion-based clustering<br>• Redundant branch pruning<br>• Fast cost estimation | • ML-based clustering | • Symmetry breaking<br>• Maximum occupying track<br>• Misaligned gate specification<br>• Structure-driven partitioning | Δ (Routing) | • Symmetry breaking<br>• Datapath-aware partitioning | • Symmetry breaking<br>• Transistor clustering<br>• ITP<br>• RLBT<br>• Early termination |
| **Technology** | PROBE | PROBE | ASAP7 [16] | Unspecified | PROBE | ASAP7 [16] | PROBE | PROBE |

SMT: Satisfiability modulo theories; DP: Dynamic Programming; SA: Simulated Annealing; RL: Reinforcement Learning; GA: Genetic Algorithm; MILP: Mixed Integer Linear Programming; CP: Constraint Programming; ITP: Identical Transistor Partitioning; RLBT: Routing Lower Bound Tightening. Key additions are highlighted in **Bold**.

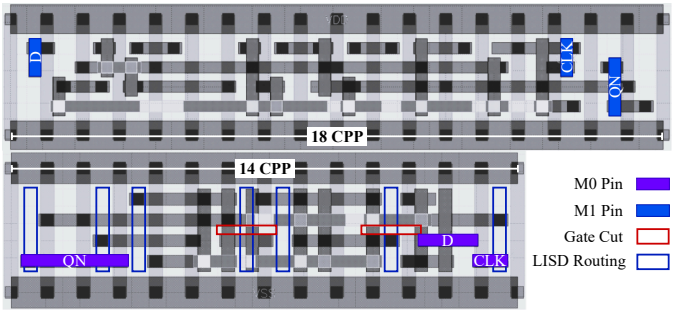



Fig. 4. Comparison of DFFHQN_X1 layouts generated by PROBE3.0 [8] (top) and CPCell (bottom) under 3:2 GR. CPCell achieves a 4-CPP reduction with gate cuts and enables M0 pins without M1 routing.

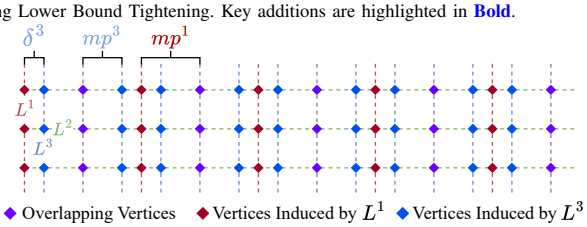



Fig. 5. Top-down view of the irregular pattern of columns on $L^2$ induced by different pitch values on $L^1$ and $L^3$.

columns for vertical layers first (Lines 1–7) and then merges them on horizontal layers without repetition (Lines 8–10), ensuring uniqueness for each column in a horizontal layer.

After creating $\mathcal{C}$ and $\mathcal{R}$, we construct a *relative layered grid graph* $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are a set of disjoint vertex sets and a set of disjoint edge sets, respectively:

$$\mathcal{V} = \{V^i | i \in \{1, ..., 4\}, \tag{3}$$

$$V^i = \{v | v = (i; r; c), r \in R^i, c \in C^i\}\} \tag{4}$$

$$\mathcal{E} = \{E^i | i \in \{1, ..., 4\}, \tag{5}$$

$$E^i = \{e | e = (v_1, v_2),\ v_1 \in V^i, v_2 \in V^j, j \in \{i-1, i\}\}\}. \tag{6}$$

Each vertex $v$ is a triplet $(i; r; c)$ that contains layer $i$, row $r$ and column $c$. Each edge $e$ is a pair of vertices whose locations are adjacent (i.e., nearest neighbors) created along the direction of this layer. To generate $\mathcal{V}$ and $\mathcal{E}$, we use Algorithm 2: for horizontal layers, edges are created along each row (Lines 8–10); for vertical layers, edges are created along each column (Lines 11–13). These edges facilitate intra-layer routing. Edges between layers, called vias, are created if and only if the two vertices are on adjacent layers and have the same $row$ and

TABLE II
SUMMARY OF SYMBOLS AND DESCRIPTIONS.

| Symbol | Description |
|---|---|
| $G = (\mathcal{V}, \mathcal{E})$ | The relative layered grid graph consisting of disjoint vertex sets $\mathcal{V}$ and edge sets $\mathcal{E}$. |
| $v = (i; r; c)$ | A vertex triplet representing a specific coordinate defined by layer $i$, row $r$ and column $c$. |
| $L^i$ | The $i^{th}$ layer of the design. |
| $w_{total}, w_p, w_n, c_{db}$ | The total cell width, PMOS transistor width, NMOS transistor width and the allowable diffusion break count, respectively. |
| $mp^i$ | The pitch value between adjacent tracks on layer $L^i$. |
| $\delta^i$ | The offset value from the left edge for layers $L^i$. |
| $C^i$ | The column set for layer $L^i$. |
| $R^i$ | The row set for layer $L^i$. |
| $scx_c$ | The super cut node on column $c$ used to abstract the shape and flow of cut rows. |
| $g_{(v,\{L,R,F,B\})}$ | Geometric variables (GVs) assigned to vertex $v$ indicating the End-of-Line of a metal segment in direction left, right, front and back. |
| $P$ | The set of external pins. |
| $\mathcal{N}$ | The set of nets excluding power and ground nets. |
| $s_n$ | The source vertex of net $n$. |
| $T_n$ | The set of terminal vertices of net $n$. |
| $t_{n,k}$ | The $k^{th}$ terminal (sink) vertex of net $n$. |
| $f_{n,k}(a)$ | A binary flow variable indicating whether the routing path from $s_n$ to terminal $t_{n,k}$ uses arc $a_{(u,v)}$. |
| $\mathcal{A}$ | The set of directed arcs obtained by orienting each undirected edge in $\mathcal{E}$ in both directions. |
| $\mathcal{A}^+(v)$, $\mathcal{A}^-(v)$ | The set of outgoing arcs from vertex $v$ and the set of incoming arcs to vertex $v$, respectively. |
| $a_{(u,v)}$ | A directed routing arc from vertex $u$ to vertex $v$. |

**Algorithm 1:** Column Set Creation

```
for i ∈ {1, 3} do
    c ← δ^i
    C^i ← ∅
    while c ≤ w_total do
        C^i ← C^i ∪ {c}
        c ← c + mp^i
    C ← C ∪ C^i
for i ∈ {2, 4} do
    C^i ← C^{i−1} ∪ C^{i+1}        // Implicitly, C^5 = ∅
    C ← C ∪ C^i
return C
```

*column* (Lines 14–17).

### B. Routing

We model routing as a *multi-commodity flow* problem [24] [25] on the relative layered grid graph (Figure 6). CPCell's directed encoding empirically scales better than prior undirected models [5] [12]. Each edge $\{u, v\} \in E$ becomes two directed arcs $a_{(u,v)}, a_{(v,u)} \in \mathcal{A}$; flow is defined exclusively on these arcs to control direction and capacity.

**Middle-of-Line (MOL) Routing.** MOL routing [16] uses the Local Interconnect Source-Drain (LISD) and gate layers to connect adjacent terminals without consuming (global) metal routing resources. When terminals align in a column, they can be "merged" through the shared active region. This allows routing to treat the column as a single sink, rather than requiring distinct flows for every terminal.

Let $\mathcal{N}$ denote the set of signal nets. For each net $n \in \mathcal{N}$, let $s_n$ be the source and $T_n = \{t_{n,0}, \ldots, t_{n,K_n-1}\}$ be the set of sink terminals. We model global routing using binary flow

**Algorithm 2:** Relative Layered Grid Graph Construction

```
for i ∈ {1...4} do
    V^i ← ∅, E^i ← ∅
    // In the order of increasing rows and columns
    for r ∈ R^i do
        for c ∈ C^i do
            v ← (i; r; c)
            V^i ← V^i ∪ {v}
            if layer i is horizontal then
                Get the nearest left vertex v_a in V^i
                E^i ← E^i ∪ {{v, v_a}}
            if layer i is vertical then
                Get the nearest lower vertex v_b in V^i
                E^i ← E^i ∪ {{v, v_b}}
            // if via can be constructed below
            v_c ← (i − 1; r; c)
            if v_c ∈ V^{i−1} then
                E^i ← E^i ∪ {{v, v_c}}
    V ← V ∪ V^i, E ← E ∪ E^i
return V, E
```

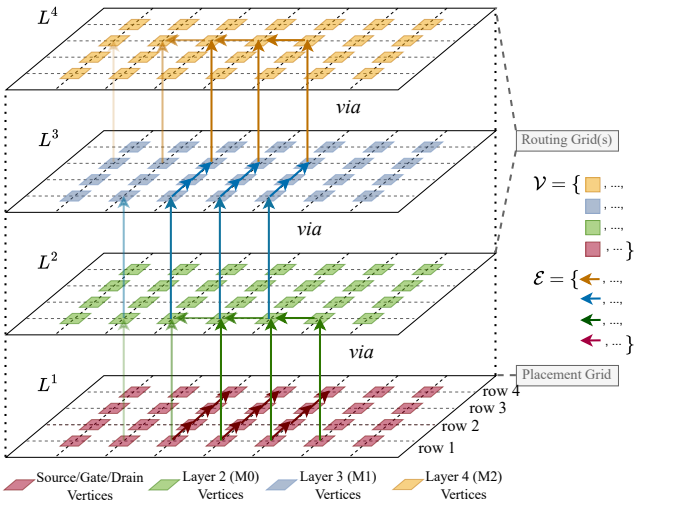



Fig. 6. Routing source/drain and gate terminals to frontside IO through $L^2$–$L^4$. Super cut nodes $scx$ control middle-row flow on each placement-grid column. Intra-layer connections follow layer orientation; vias connect layers at access points (colored squares).

variables:

$$f_{n,k}(a) \in \{0, 1\}, \quad \forall a \in \mathcal{A}, \tag{7}$$

indicating whether the path from $s_n$ to terminal $t_{n,k}$ traverses arc $a$. To capture the interaction between global routing flow and local MOL connectivity, we introduce *super cut node*, $scx_c \in \{0, 1\}$, which determines the physical continuity of the placement grid ($L^1$) at column $c$. $scx_c = 1$ represents a physical break—either as a gate cut at even columns or a LISD cut at odd columns—which enforces electrical isolation and requires terminals to be routed individually by metal layers. Conversely, $scx_c = 0$ implies a continuous active region where terminals are electrically shorted. The local connectivity allows the router to bypass flow requirements for individual terminals, provided the column itself is connected. $scx_c$ is defined as an abstract cut value associated with column $c$.

For gate cuts, the *cut width* is specified in contacted poly pitch (CPP) units [26]. We enforce it by grouping super cut nodes across adjacent even columns (Figure 7); a two-CPP

cut requires paired activation of $scx_c$ on neighboring even columns. Cut width affects cell width: Figure 7(a), with a 2-CPP cut, uses one fewer CPP than Figure 7(b), with a 1-CPP cut. Unless specified otherwise, we use 2-CPP cuts to reduce the area of cross-coupled structures in D flip-flops, latches and multiplexers.

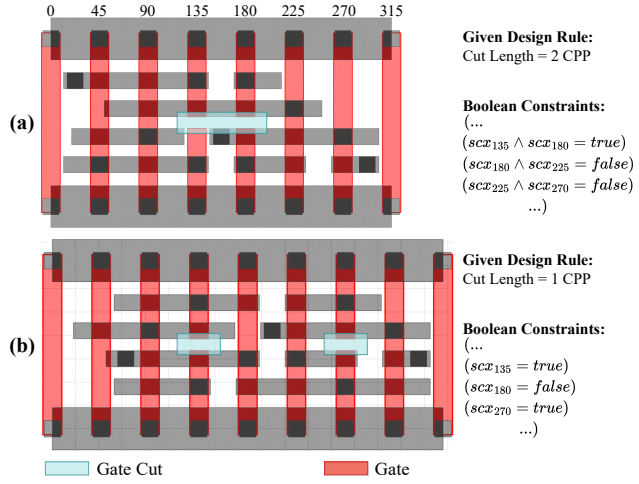



Fig. 7. MUX2_X1 with (a) 2-CPP and (b) 1-CPP cut widths. Super cut nodes $scx$ abstract cut shapes: adjacent columns activate together in (a), while nodes activate individually in (b).

**Flow Conservation Constraints.** For each terminal $t_{n,k}$ of net $n$ that is not merged by MOL routing, the flow must satisfy the following conservation laws:

$$\sum_{a \in \mathcal{A}^+(v)} f_{n,k}(a) - \sum_{a \in \mathcal{A}^-(v)} f_{n,k}(a) = \begin{cases} 1 & \text{if } v = s_n \\ -1 & \text{if } v = t_{n,k} \\ 0 & \text{otherwise} \end{cases}. \quad (8)$$

### C. Layer-based Fine-grained Design Rule Checking

CPCell enforces the complete set of conditional design rules specified by the target PDK [8], including Minimum Area Rule (MAR), Via Separation, End-Of-Line spacing (EOL), Step Height Rule (SHR) and Parallel Run Length (PRL). We parameterize these rules per layer and encode them using geometric variables to ensure full compliance.

Geometric variables are introduced in [4] to indicate the End-Of-Line of a metal segment. We define geometric variables

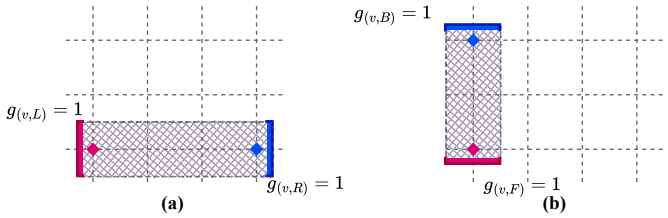



Fig. 8. Vertex geometric variables mark metal endpoints in four directions: (a) left/right and (b) front/back.

(GVs) in our relative layered grid graph by assigning a set of Boolean variables to each vertex $v$: $g_{(v,L)}, g_{(v,R)}, g_{(v,F)}, g_{(v,B)}$, representing the four directions—left, right, front and back.[2] On horizontal layers, a left-end GV $g_{(v,L)}$ must be paired with exactly one right-end GV $g_{(u,R)}$ to create a horizontal metal segment (see Figure 8(a)). Similarly, on vertical layers, a front-end GV $g_{(v,F)}$ must be paired with one back-end GV $g_{(u,B)}$ to create a vertical metal segment (see Figure 8(b)).

The non-uniformity in a relative layered grid graph requires the design rule parameters to be defined in terms of Euclidean distance. Prior methods [4] [12] [27] define these design rule parameters in terms of "number of vertices", which is valid only in a uniform grid where all edge lengths are equal.

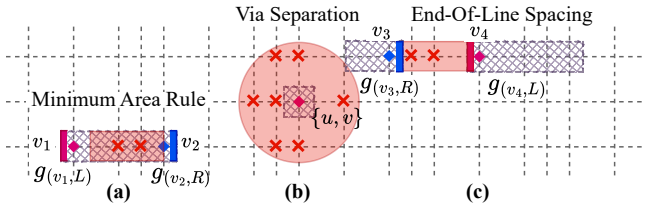



Fig. 9. Design rule checking on $L^2$ with distance input. The red-shaded region represents the prohibited area. (a) Minimum Area Rule (MAR) uses geometric variables and prohibits the current metal segments from ending too closely together. (b) Via Separation uses edge variables and prohibits any adjacent via within the Euclidean distance. (c) End-Of-Line spacing (EOL) uses geometric variables and prohibits any adjacent metal from ending too close to the current segment.

We input design rule parameters based on distance and track the total distance traveled along each path until the desired distance is attained. Design rule parameters can also be defined for each layer, as different layers may have varying metal pitches. Figure 9 demonstrates three of the design rules: MAR, Via Separation and EOL. Figure 9(a) shows the usage of geometric variables to encode the minimum area of a metal segment. Once $g_{(v_2,R)}$ is instantiated, the leftward $g_{(v_1,L)}$ within the user-defined length is prohibited (indicated by the red cross sign). Similarly, Figure 9(c) shows the usage of geometric variables to encode the end-of-line spacing of a metal segment. Once $g_{(v_4,L)}$ is instantiated, the leftward $g_{(v_3,R)}$ within the user-defined length is prohibited (indicated by the red cross sign). Via Separation, on the other hand, does not use any geometric variables as illustrated in Figure 9(b). Each via is represented by an undirected edge $\{u, v\}$, where $u = (i-1; r; c)$ and $v = (i; r; c)$. Once a via edge $\{u, v\}$ is instantiated, all vias within the user-defined radius are prohibited (indicated by the red cross sign). We encode these three rules using the Boolean cardinality constraint at-most-1 from CP-SAT. For other design rules, we examine all routing edges and their neighboring vertices to prohibit access within the specified distance.

## III. Multi-Objective Optimization

We formulate cell layout generation as a weighted multi-objective optimization problem solved using CP-SAT [13]. Prior methods [4] [5] apply lexicographic optimization, which minimizes objectives sequentially and may sacrifice optimality beyond the first few objectives. In contrast, we guarantee global optimality for the weighted-sum objective when solved to exact certification. While cell width and wirelength are sufficient to optimize primary cell-level metrics, *auxiliary objectives* play a critical role in steering the solver toward the optimal solution more efficiently. The complete set of objectives is defined as follows.

(1) $CW$: Cell width measured in the number of contacted poly

[2]Front corresponds to the bottom endpoint (lower row index) and Back corresponds to the top endpoint (higher row index).

pitch (#CPP).
(2) $SGD$: Total #source/gate/drain shared.
(3) $WL$: Weighted wirelength.
(4) $DBX$: Diffusion break placement, measured by the sum of placement column indices of all diffusion breaks.
(5) $M2$: Top metal (M2) utilization, defined as the count of M2 tracks containing at least one active routing segment.

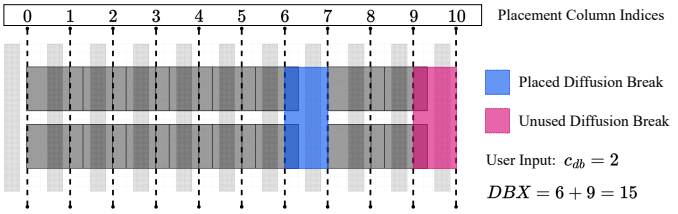



Fig. 10. Illustration of diffusion break placement and the $DBX$ objective. When the maximum allowable diffusion breaks ($c_{db}$) exceed the layout requirements, the objective forces surplus breaks (pink) to the far right edge, minimizing the active cell width.

Together, the final objective function is:

$$\begin{aligned} \min \quad & \lambda_0 \cdot CW + \lambda_1 \cdot WL - \lambda_2 \cdot SGD \\ & - \lambda_3 \cdot DBX + \lambda_4 \cdot M2. \end{aligned} \tag{9}$$

Unless stated otherwise, we use fixed integer weights in all experiments: $\lambda_0 = 1000$ and $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$. We choose $\lambda_0$ to make $CW$ the primary optimization target. $SGD$, $DBX$ and $M2$ are auxiliary objectives that bias the search towards favorable structural patterns—such as increased MOL routing, rightward diffusion break placement and reduced M2 usage—thereby accelerating convergence without altering the optimal $CW$ and $WL$. The runtime impact of each auxiliary objective is detailed in Section VI-B via an ablation study.

While the roles of $SGD$ and $M2$ are relatively intuitive, the purpose of $DBX$ is less obvious. Figure 10 shows an example of diffusion break placement and the $DBX$ objective value. The $DBX$ objective encourages diffusion breaks to be placed toward the right boundary of the layout canvas. Importantly, the user-defined parameter $c_{db}$ serves only as an *upper bound* on the available diffusion breaks; CPCell automatically determines the actual number required for a layout. To ensure satisfiability, $c_{db}$ is often set higher than the minimum necessary count. In these cases, the $DBX$ objective naturally pushes any surplus (unused) diffusion breaks to the far right edge, preventing them from fragmenting the active region or increasing the effective cell width. Consequently, maximizing the rightward placement of diffusion breaks implicitly minimizes the occupied layout area, reinforcing the primary cell width objective.

## IV. M0 Pin Enhancement

With enabled MOL routing, many nets no longer require M1 for internal routing, allowing M1 pins to be replaced by M0 pins and thereby increasing the number of access points available to the block-level router (see Figure 11). Within the given cell width, an M1 pin is accessible through four M2 routing tracks, an M0 pin can overlap up to seven M1 routing tracks, and it further benefits from additional routing resources enabled by non-1:1 gear ratios at the M1 layer. We follow the same set of constraints as SP&R [4] to guarantee pin accessibility at M1. However, M0 pin accessibility is limited by routing blockages at both the M0 and M1 layers: pin-to-pin and pin-to-metal spacing rules limit extensibility at M0, while internal M1 metals may fully block routing tracks above. In CPCell, these effects are captured through two hard constraints–pin separation (PS) and minimum pin opening (MPO)–which together guarantee robust pin accessibility.

### A. Pin Separation (PS)

Let $\mathcal{P}$ denote the set of external pins. With M0 pin enablement, each pin $p \in \mathcal{P}$ is assigned at least one access vertex on the M0 layer. We denote the representative access vertex of pin $p$ by

$$v_p = (2; r_p; c_p) \in V^2. \tag{10}$$

If the number of available M0 rows is at least $|\mathcal{P}|$, CPCell enforces PS as a hard constraint by requiring each pin to occupy a distinct M0 row:

$$r_{p_i} \neq r_{p_j}, \quad \forall\, p_i \neq p_j,\ p_i, p_j \in \mathcal{P}. \tag{11}$$

PS isolates pin access points across rows, complementing minimum pin opening (MPO), which expands access regions along M0.

### B. Minimum Pin Opening (MPO)

While PS enforces spatial isolation among pins, each external pin must also expose a sufficient number of access points to adjacent M1 routing tracks to ensure block-level routability. Minimum Pin Opening (MPO) enforces a hard lower bound on the horizontal extent of each M0 pin, measured by the number of accessible M1 routing tracks it overlaps.

For pin $p \in \mathcal{P}$, let $\mathcal{C}_p \subseteq C^2$ be the unoccupied M1 tracks overlapping the pin. CPCell requires the hard constraint:

$$|\mathcal{C}_p| \geq \Theta, \quad \forall\, p \in \mathcal{P}, \tag{12}$$

where $\Theta$ is the user-specified minimum pin opening. Each external pin thus exposes multiple access points despite internal M1 routing blockages. Together, PS and MPO provide spatial isolation and sufficient pin exposure for downstream routing under congestion.

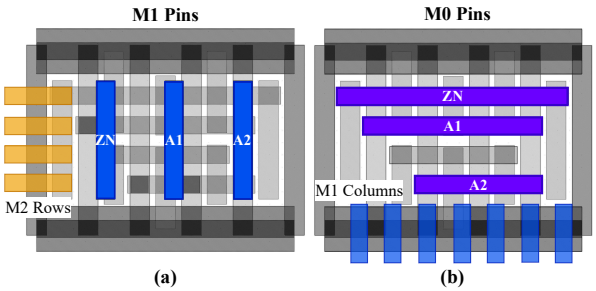



Fig. 11. NAND2_X2 pin access under 3:2 GR: (a) M1 pins accessed by M2 rows (golden yellow); (b) M0 pins accessed by M1 tracks (dark blue).

## V. Acceleration Techniques

Gear ratios and offsets increase routing resources and search complexity [5]. SP&R partitioning requires expert knowledge of optimal datapath structures [28]; SMTCell-GR pre-partitioning reuses solutions across related netlists [5]; and SO3 [23] uses structural partitioning, misaligned gate specification and maximum occupying track. Their reliance on manual

**Algorithm 3:** Graph Construction and Clustering

**Input:** Netlist, cluster bounds $[k_{\min}, k_{\max}]$
**Output:** set of transistor clusters $Clst$

```
1  M ← BuildCircuitGraph(Netlist);
2  M ← M \ {VDD, VSS}
3  foreach net node n ∈ M such that deg(n) ≤ 2 do
4      {tr_i, tr_j} ← Neighbors(M, n);
5      AddEdge(M, tr_i, tr_j);
6      M ← M \ {n};
7  for net node n ∈ M do
8      for neighbor tr ∈ Neighbors(M, n) do
9          w(n, tr) ← deg(n)
10 𝕏 ← KamadaKawaiLayout(M, w)
11 Clst, noise ← HDBSCAN(M, 𝕏, k_min, k_max);
12 return Clst;
```

intervention or prior design knowledge limits application to unseen netlists. We introduce four complementary techniques: *transistor clustering* groups strongly coupled devices to reduce placement granularity; *identical transistor partitioning* (ITP) removes equivalent permutations through canonical ordering; *routing lower bound tightening* (RLBT) accelerates optimality certification; and *early termination* uses a relative gap to trade optimality for runtime on large netlists.

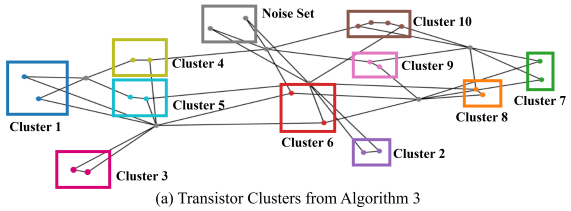
### A. Transistor Clustering

We cluster transistors whose diffusion should be shared and impose structural constraints. Our approach is designed as a lightweight add-on to constraint solving. Figure 12 shows a one-to-one correspondence between the clusters from our method and the clusters in the optimal ground-truth layout of DFFHQN_X1. By utilizing these clusters to constrain the search space, the solver operates at the cluster-level rather than on individual transistors. Our clustering procedures are detailed in Algorithm 3 and summarized below:

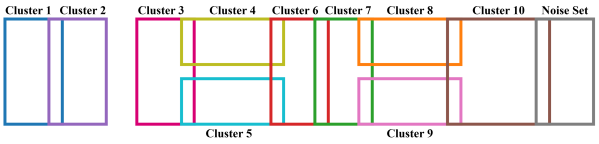
*Lines 1–8:* We build a graph $M$ with both transistors and nets represented as vertices. Connections are established exclusively between transistor vertices and net vertices: a link exists between a transistor $tr$ and a net $n$ if the net is incident to the transistor's source, drain, or gate. We remove power and ground net nodes. We then remove any net node with degree $\leq 2$, and reconnect the incident transistors directly. This representation directly correlates with the potential candidate clusters for diffusion sharing and placement proximity.

*Lines 9–12:* We assign weights to fan-out edges from each net node proportional to the net degree and compute a two-dimensional embedding using the Kamada-Kawai spring model [29]. The model optimizes a spring-based energy function to preserve graph distances in Euclidean space, producing a layout in which strongly connected transistors are placed in close proximity.

*Line 13–14:* We apply Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [30] [31] to the embedded nodes, using the pairwise Euclidean distances in the embedding space as the clustering metric. HDBSCAN is chosen because it identifies density-based clusters without requiring a predefined cluster count, and explicitly models a noise set. Transistors classified as *noise* are isolated or weakly connected in the embedding space and are treated as free-floating to preserve satisfiability. HDBSCAN also accepts minimum and maximum cluster sizes, denoted as $k_{min}$ and $k_{max}$. $k_{min}$ is always set to two as it is the minimum effective cluster size. $k_{max}$ can range from two to six, depending on the netlist. When $k_{max} = 6$, the search space is aggressively pruned, but the solver may produce suboptimal solutions or even become unsatisfiable. Setting $k_{max} = 2$ is safer but produces limited speedup. The effect of these parameters is studied in Section VI-D. HDBSCAN returns a noise set of transistors and a set of density-based clusters $\{Clst_1, Clst_2, \ldots\}$, where each cluster $Clst_k$ captures a group of transistors with strongly coupled connectivity. The noise set is discarded as it may over-constrain the layout and result in an unsatisfiable search space. Because clustering restricts the encoded feasible set, it does not always guarantee optimality with respect to the original formulation.




(a) Transistor Clusters from Algorithm 3




(b) DFFHQN_X1 Optimal Layout Solution (with Cluster Overlays from Algorithm 3)

Fig. 12. Algorithm 3 reveals transistor clusters found in the optimal layout solution for DFFHQN_X1. (a) Identified 10 transistor clusters and one noise set via Algorithm 3. (b) The optimal layout solution generated without cluster-based constraints. The colored overlays show the correspondence between the clusters identified by Algorithm 3 and the clusters in the optimal placement.

Once transistor clusters are identified, we enforce diffusion continuity within each cluster. Let $Clst_k$ be a cluster comprising a set of transistors $\mathcal{T}_k$. For each transistor $tr \in \mathcal{T}_k$, let $x_{tr}$ denote its left coordinate and $w_{tr}$ its diffusion width. We enforce contiguous placement by constraining the total span of the cluster to equal the sum of the individual transistor widths:

$$\left(\max_{tr\in\mathcal{T}_k}(x_{tr}+w_{tr})\right) - \min_{tr\in\mathcal{T}_k} x_{tr} = \sum_{tr\in\mathcal{T}_k} w_{tr}. \tag{13}$$

The term $\max(x_{tr} + w_{tr}) - \min(x_{tr})$ represents the width of the bounding box of $Clst_k$. Enforcing equality with the sum of widths eliminates spacing gaps and overlaps, ensuring a single uninterrupted diffusion chain. Each PMOS-only and NMOS-only cluster constraint is enforced by Equation 13. For a cluster of size $|\mathcal{T}_k| = 2$ containing one PMOS and one NMOS transistor, we additionally require gate sharing ($scx_c = 0$). For clusters of size $|\mathcal{T}_k| > 2$, containing both PMOS and NMOS devices, the total width $\sum w_{tr}$ is determined by the transistor type with the maximum aggregate diffusion width; transistors of the complementary type are constrained to lie within this spatial bound.

### B. Identical Transistor Partitioning (ITP)

High-drive-strength cells (e.g., X2, X4 and X8 variants) contain groups of electrically identical transistors to achieve the

target drive strength. Without additional partitioning, the solver must explore every permutation of these identical transistors, even though many such permutations yield identical layouts. CPCell partitions identical transistors into a canonical ordering. Given a set $S$ of transistors with identical gate, source, drain and type attributes, we enforce ordering constraints of the form:

$$x_{tr_1} \leq x_{tr_2} \leq \cdots \leq x_{tr_{|S|}}, \tag{14}$$

where $x_{tr}$ denotes the placement coordinate. This symmetry breaking explores only one layout per permutation class.

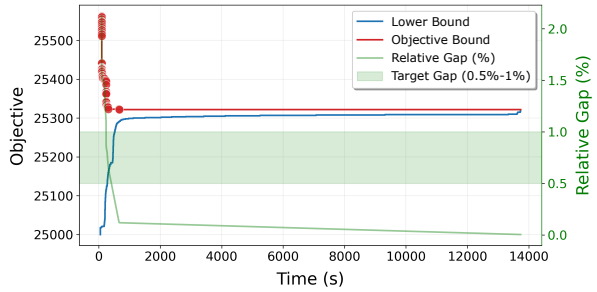



Fig. 13. Search progress of the CP-SAT solver for DFFHQN_X1 with cell-width ($CW$) and wirelength ($WL$) objectives. The red line graph denotes the objective, where each marker corresponds to the discovery of a new feasible layout, while the blue line graph represents the lower bound.

### C. Routing Lower Bound Tightening (RLBT)

Half-perimeter wirelength (HPWL) [32] from intermediate feasible placements provides a routing lower bound for faster optimality verification. CP-SAT combines branch-and-bound, linear programming (LP) relaxations and Boolean satisfiability (SAT) reasoning [33]. In Figure 13, red denotes the best feasible objective and blue the relaxation-derived lower bound. Branching, constraint propagation and relaxation refinement tighten the bounds; convergence certifies optimality.

Empirically, we observe that CP-SAT often identifies an optimal or near-optimal solution early in the search, whereas a substantial portion of the remaining runtime is devoted to tightening the lower bound to certify optimality. If the lower bound can be explicitly derived early on, we can reduce the runtime for such a certification. At block-level, HPWL is commonly used as a proxy for placement quality. At cell-level, we can use HPWL to estimate the wirelength of each net from the intermediate feasible placement solution, which serves as a routing lower bound. Once CP-SAT converges on a placement, this estimation immediately strengthens the lower bound, thereby accelerating optimality verification.

### D. Early Termination & Relative Gap

To reduce solver runtime for large problem instances, we adopt an early-termination criterion based on the *relative gap* between the current objective bound and the lower bound. In our experiments, the relative gap is set to $0.5\%$ or $1\%$ for large netlists and is defined as

$$\text{Relative Gap} = \frac{|\text{ObjectiveBound} - \text{LowerBound}|}{|\text{ObjectiveBound}|}. \tag{15}$$

From Figure 13, we observe that after the initial rapid objective improvement before 672 s, no better solution was found; instead, the solver devotes $13,062$ s exclusively to tightening the lower bound to formally certify optimality. This behavior motivates the use of the relative gap for early termination. When the relative gap is used, we only guarantee optimality for the primary objective, cell width.

TABLE III
PROCESS TECHNOLOGY PARAMETERS FROM PROBE3.0 [8].

| **Parameter** | **Value** |
|---|---|
| Contacted poly pitch ($CPP$) | 45 nm |
| M0 pitch ($M0P$) | 24 nm |
| M1 pitch ($M1P$) | 45 nm, 30 nm, 27 nm |
| M2 pitch ($M2P$) | 24 nm |
| Fin count | 2 |
| Cell height | 144 nm (4 routing tracks) |
| BEOL stack | 14 layers (M0–M13) |
| Design rules | Min. EOL=21 nm, MAR=595 nm$^2$ |
| Supply volt. / corner / temperature | 0.7 V, $TT$, 25°C |

## VI. Cell-level Experiments on Gear-ratio-enabled Cell Layouts

All cells are generated under the predictive 3-nm-class technology defined by the PROBE3.0 framework [8]; Table III summarizes the process-technology parameters. Cell-level experiments were conducted on an Intel Core i9-12900K CPU with 64 GB RAM using CP-SAT solver (v9.14) [13] with six parallel search workers. All experiments target the 2-Fin 4-Routing Track (2F4T) configuration, whose horizontal routing is more constrained than in the previously studied 3-Fin 5-Routing Track (3F5T) configuration [5], better exposing gear-ratio effects. All layouts use the single-diffusion-break setting, reducing transistor spacing and cell width. We analyze representative cells across gear ratios and offsets, ablate objective functions and weights using DFFHQN_X1, and evaluate acceleration techniques on several D flip-flops and a high-drive-strength buffer.

### A. Cell Metrics

We present a sensitivity study on gear ratios in Table IV and an averaged full library comparison with previous gear-ratio-enabled work in Table V. In Table IV, we report results for 22 high-drive-strength and functionally diverse cells out of the 40 cells [5] designed under 2F4T configuration. We evaluate three gear ratio (GR) settings–45:45 (1:1), 45:30 (3:2) and 45:27 (5:3)–along with their respective offset counterparts. For a given GR parameter, the number of valid offset choices is determined by the alignment between the poly grid (pitch $mp^1$) and the M1 grid (pitch $mp^3$). Specifically, admissible offsets are restricted to integer multiples of the greatest common divisor (gcd), or $\gcd(mp^3, mp^1)$ within one M1 period. 1:1 GR admits no valid offset, whereas 3:2 GR has one valid offset of 15 nm (3:2$^\dagger$), and 5:3 GR has two valid offsets of 9 nm and 18 nm (5:3$^\dagger$ and 5:3$^{\dagger\dagger}$).

Across all settings, 1:1 GR consistently achieves the shortest CP-SAT runtime due to its smaller routing search space. However, its limited routing resources result in the largest average cell width. Across the 22 reported cells, non-1:1 GR configurations reduce average cell width from 7.82 CPP to 7.68–7.73 CPP. The two 5:3 GR offset configurations have the lowest average width, including a one-CPP reduction for MUX2_X1. The mean per-cell normalized wirelength is slightly

TABLE IV

CELL METRICS AND RUNTIMES FOR 2F4T GEAR-RATIO-ENABLED CELLS: 1:1 GR HAS NO OFFSET VARIANT; 3:2 GR HAS ONE 15 NM OFFSET (3:2†); AND 5:3 HAS TWO OFFSETS OF 9 NM AND 18 NM (5:3† AND 5:3††), RESPECTIVELY. WIRELENGTH IS NORMALIZED TO THE 1:1 GR BASELINE.

| Cell Name | #FET | #Net | Cell Width (CPP) | | | | | | Normalized Wirelength | | | | | | #M2 | | | | | | CP-SAT Runtime (s) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† |
| AND2_X2 | 8 | 7 | 5 | 5 | 5 | 5 | 5 | 5 | 1.00 | 0.88 | 0.90 | 0.88 | 0.88 | 0.90 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 5 | 4 | 5 | 4 | 4 |
| AND3_X2 | 10 | 9 | 6 | 6 | 6 | 6 | 6 | 6 | 1.00 | 0.95 | 0.98 | 0.95 | 0.98 | 0.99 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 7 | 6 | 7 | 7 | 8 |
| AOI21_X2 | 12 | 8 | 8 | 7 | 7 | 7 | 7 | 7 | 1.00 | 1.53 | 1.64 | 1.36 | 1.37 | 1.40 | 0 | 2 | 2 | 1 | 1 | 1 | 15 | 22 | 32 | 27 | 26 | 42 |
| AOI22_X2 | 16 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 1.00 | 1.00 | 1.00 | 1.00 | 1.10 | 1.00 | 0 | 0 | 0 | 0 | 1 | 0 | 53 | 533 | 210 | 278 | 136 | 252 |
| BUF_X4 | 10 | 5 | 7 | 7 | 7 | 7 | 7 | 7 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 18 | 14 | 18 | 29 | 20 |
| BUF_X8 | 20 | 5 | 13 | 13 | 13 | 13 | 13 | 13 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 210 | 712 | 613 | 1105 | 2931 | 991 |
| DFFHQN_X1 | 24 | 17 | 14 | 14 | 14 | 14 | 14 | 14 | 1.00 | 1.00 | 1.02 | 1.00 | 1.00 | 1.01 | 2 | 2 | 2 | 2 | 2 | 2 | 121 | 254 | 300 | 352 | 429 | 414 |
| INV_X4 | 8 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 1 | 3 | 3 |
| INV_X8 | 16 | 4 | 9 | 9 | 9 | 9 | 9 | 9 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 3 | 3 | 3 |
| LHQ_X1 | 8 | 13 | 10 | 10 | 10 | 10 | 10 | 10 | 1.00 | 1.00 | 1.02 | 1.00 | 0.98 | 1.00 | 1 | 1 | 1 | 1 | 1 | 1 | 28 | 141 | 256 | 111 | 353 | 250 |
| MUX2_X1 | 12 | 12 | 8 | 8 | 8 | 8 | 7 | 7 | 1.00 | 0.93 | 1.01 | 0.93 | 0.89 | 0.94 | 1 | 1 | 1 | 1 | 1 | 1 | 9 | 9 | 13 | 26 | 9 | 9 |
| NAND2_X2 | 8 | 6 | 5 | 5 | 5 | 5 | 5 | 5 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 4 | 5 | 6 | 6 | 6 |
| NAND3_X2 | 12 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1 | 1 | 1 | 1 | 1 | 1 | 12 | 24 | 28 | 45 | 51 | 50 |
| NAND4_X2 | 16 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1 | 1 | 1 | 1 | 1 | 1 | 15 | 53 | 47 | 115 | 71 | 66 |
| NOR2_X2 | 12 | 8 | 5 | 5 | 5 | 5 | 5 | 5 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 5 | 5 | 6 | 6 | 6 |
| NOR3_X2 | 12 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1 | 1 | 1 | 1 | 1 | 1 | 11 | 23 | 35 | 33 | 56 | 41 |
| NOR4_X2 | 16 | 9 | 9 | 9 | 9 | 9 | 9 | 9 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1 | 1 | 1 | 1 | 1 | 1 | 19 | 61 | 58 | 122 | 97 | 123 |
| OAI21_X2 | 12 | 8 | 8 | 7 | 7 | 7 | 7 | 7 | 1.00 | 1.53 | 1.64 | 1.36 | 1.37 | 1.40 | 0 | 2 | 2 | 1 | 1 | 1 | 15 | 36 | 17 | 29 | 45 | 21 |
| OAI22_X2 | 16 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 1.00 | 1.00 | 1.00 | 1.00 | 1.10 | 1.00 | 0 | 0 | 0 | 0 | 1 | 0 | 48 | 456 | 167 | 297 | 73 | 266 |
| OR2_X2 | 8 | 7 | 5 | 5 | 5 | 5 | 5 | 5 | 1.00 | 0.85 | 0.86 | 0.84 | 0.83 | 0.87 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 5 | 4 | 5 | 5 | 4 |
| OR3_X2 | 10 | 9 | 6 | 6 | 6 | 6 | 6 | 6 | 1.00 | 0.96 | 0.98 | 0.95 | 0.98 | 0.99 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 9 | 7 | 11 | 8 | 8 |
| XOR2_X1 | 10 | 9 | 6 | 6 | 6 | 6 | 6 | 6 | 1.00 | 0.99 | 1.00 | 0.98 | 0.98 | 0.99 | 1 | 1 | 1 | 1 | 1 | 1 | 4 | 6 | 6 | 9 | 7 | 6 |
| **Average** | | | 7.82 | 7.73 | 7.73 | 7.73 | 7.68 | 7.68 | 1.00 | 1.03 | 1.05 | 1.01 | 1.02 | 1.02 | 0.41 | 0.59 | 0.59 | 0.50 | 0.59 | 0.50 | 26.86 | 108.41 | 83.23 | 118.68 | 197.95 | 117.86 |

TABLE V

COMPARISON OF AVERAGED CELL METRICS AND TOTAL RUNTIMES ACROSS PROBE3.0, SMTCELL-GR, AND OUR WORK FOR 2F4T GEAR-RATIO (GR) ENABLED CELLS. THE 1:1 GR HAS NO OFFSET VARIANT; THE 3:2 GR INCLUDES A 15 NM OFFSET (3:2†); AND THE 5:3 GR INCLUDES 9 NM (5:3†) AND 18 NM (5:3††) OFFSETS. WIRELENGTH IS NORMALIZED TO THE 1:1 GR BASELINE. "—": UNAVAILABLE CONFIGURATION FOR THE TOOL.

| Tool | Average Cell Width (CPP) | | | | | | Average Normalized Wirelength | | | | | | Average #M2 | | | | | | Total Runtime (s) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† | 1:1 | 3:2 | 3:2† | 5:3 | 5:3† | 5:3†† |
| PROBE3.0 [8] | — | 6.60 | — | — | — | — | — | 1.16 | — | — | — | — | — | 0.35 | — | — | — | — | — | 20518.00 | — | — | — | — |
| SMTCell-GR [5] | 6.83 | 6.50 | — | 6.40 | — | — | 1.64 | 1.31 | — | 1.40 | — | — | 0.35 | 0.38 | — | 0.38 | — | — | 7136.49 | 12946.25 | — | 11481.80 | — | — |
| Our work | 6.43 | 6.08 | 6.56 | 6.08 | 6.56 | 6.56 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.23 | 0.35 | 0.75 | 0.23 | 0.81 | 0.63 | 753.30 | 2477.40 | 2584.32 | 2821.10 | 4952.41 | 3762.09 |

higher than the 1:1 baseline, and the effect of GR and offset vary by cell. The 1:1 GR configuration has the lowest average runtime in this experiment.

In Table V, we compare our framework against PROBE3.0 [8] and SMTCell-GR [5] across the full set of 40 cells. Our approach achieves smaller average cell widths and normalized wirelengths, comparable or lower average M2 usage, and up to a $9.47\times$ runtime speedup over the available baselines, while additionally supporting offset gear-ratio configurations.

### B. Ablation Study on Objectives

We ablate auxiliary objectives on DFFHQN_X1 to evaluate their individual and combined effects on solver runtime. We perform comparison studies of five objective combinations to isolate the individual and combined effects of auxiliary objectives on solver convergence behavior:

(a) $\lambda_0 \cdot CW + \lambda_1 \cdot WL$
(b) $\lambda_0 \cdot CW + \lambda_1 \cdot WL - \lambda_2 \cdot SGD$
(c) $\lambda_0 \cdot CW + \lambda_1 \cdot WL - \lambda_3 \cdot DBX$
(d) $\lambda_0 \cdot CW + \lambda_1 \cdot WL + \lambda_4 \cdot M2$
(e) $\lambda_0 \cdot CW + \lambda_1 \cdot WL - \lambda_2 \cdot SGD - \lambda_3 \cdot DBX + \lambda_4 \cdot M2$

Study (a) considers only cell width and wirelength minimization and serves as the baseline configuration. Studies (b), (c) and (d) individually activate source/gate/drain sharing, diffusion break columns and M2 usage, respectively, to isolate their effects on solver convergence. Study (e) enables all auxiliary objectives to evaluate their combined impact on accelerating solver convergence.

Figure 14 compares CP-SAT convergence for DFFHQN_X1 under different objective configurations. All configurations converge to the same final cell width and wirelength, confirming that auxiliary objectives do not affect layout optimality. However, convergence speed differs significantly. Enabling individual acceleration objectives substantially reduces runtime relative to the baseline: Study (b) achieves a 96.7% speedup, Study (c) achieves a 90.6% speedup and Study (d) achieves a 95.7% speedup. Finally, combining all auxiliary objectives in Study (e) yields the fastest convergence overall, achieving a 97.6% speedup. These results demonstrate that carefully designed auxiliary objectives can improve solver efficiency without degrading layout quality.

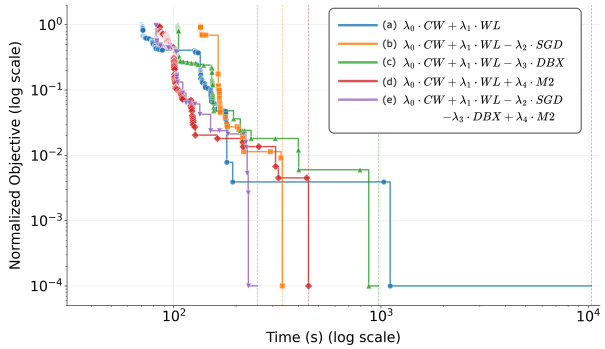



Fig. 14. Search progress of the CP-SAT solver for DFFHQN_X1 under different objective sets, shown on a logarithmic time scale with normalized objective values. The vertical dashed line marks the end of the search.

### C. Effect of M0 Pin Accessibility Constraints

We model M0 pin accessibility using the count of unblocked on-grid access points, approximated by the number of M1 routing tracks that overlap each extended M0 pin.

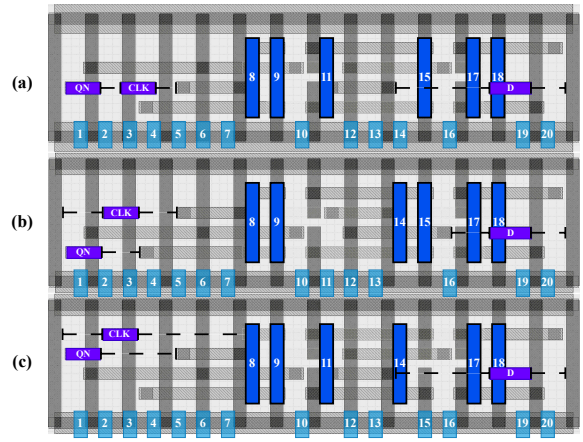



Fig. 15. Comparison of pin accessibility for DFFHQN_X1: (a) baseline, (b) PS enabled and (c) PS and MPO ($\Theta$ = 4) enabled. M0 pins are shown in purple with dashed extensible regions; black-outlined M1 tracks indicate routing blockages and light-blue segments denote candidate access locations.

Prior works [34] [35] show that alignment to routing tracks and the resulting number of on-grid access points strongly correlate with pin accessibility, so the overlap count serves as a simple proxy. Figure 15 compares the DFFHQN_X1 layouts under progressively enabled pin accessibility constraints. We measure M0 pin accessibility by the number of unblocked M1 routing tracks that geometrically overlap each extended M0 pin. Table VI details the specific track counts and indices for each configuration. Figure 15(a) shows the baseline where pins share a single row, limiting track access. Figure 15(b) shows the effect of PS, which distributes pins across three rows and reduces interference. Figure 15(c) shows the highest accessibility achievable by combining PS with MPO ($\Theta$ = 4). M0 pins extend to a minimum of four M1 access points, with the CLK pin reaching seven points.

TABLE VI
NUMBER AND INDICES OF M1 ROUTING TRACKS OVERLAPPING EACH EXTENDED M0 PIN FOR DFFHQN_X1 UNDER PS AND MPO ($\Theta$).

| Index | PS | MPO ($\Theta$) | Pin | #Access | M1 Track Indices |
|---|---|---|---|---|---|
| (a) | – | – | QN | 2 | 1, 2 |
| | | | CLK | 2 | 3, 4 |
| | | | D | 3 | 16, 19, 20 |
| (b) | ✓ | – | QN | 3 | 1, 2, 3 |
| | | | CLK | 4 | 1, 2, 3, 4 |
| | | | D | 2 | 19, 20 |
| (c) | ✓ | 4 | QN | 4 | 1, 2, 3, 4 |
| | | | CLK | 7 | 1, 2, 3, 4, 5, 6, 7 |
| | | | D | 4 | 15, 16, 19, 20 |

## D. Effect of Acceleration Techniques

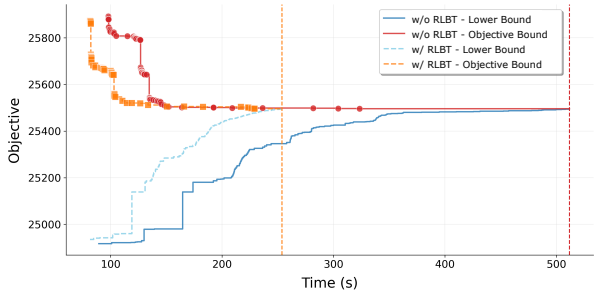



Fig. 16. CP-SAT search progress for DFFHQN_X1 with and without routing lower bound tightening, showing objective and lower bounds over time. The vertical dashed line marks search termination.

In this section, we evaluate the four acceleration techniques introduced in Section V. RLBT is always enabled in CPCell, as it directly reduces CP-SAT optimality certification overhead. As shown in Figure 16, a tight routing lower bound substantially accelerates search convergence, reducing solver runtime from 512 s to 254 s (50.4% speedup).

Table VII details the runtime impact of acceleration techniques on cells sourced from PROBE3.0 [8] (DFFHQN_X1, DFFRNQ_X1), ASAP7 [16] (DFFHQN_X3/X4, BUF_X16), and SO3 [23] (2BDFFHQN_X1) under 3:2 GR setting. A maximum runtime of 150000 s is imposed, after which the best feasible solution is reported. Transistor clustering is deterministic given a fixed seed (in our studies, we use seed = 42). We default the parameters to $k_{\min} = 2, k_{\max} = 4$. If $k_{\max} = 4$ reports infeasible, we manually lower the threshold by 1 until feasibility is met. Transistor clustering consistently improves runtime while preserving optimal width, wirelength, and M2 usage. Specifically, DFFHQN_X1 achieves a 36.6% speedup with clustering ($k_{\max} = 2$) and a 58.3% speedup (106 s) with clustering ($k_{\max} = 4$). Similarly, DFFRNQ_X1 achieves a 42.1% speedup with clustering ($k_{\max} = 2$); however, aggressive clustering ($k_{\max} = 4$) renders the run unsatisfiable. For high-drive-strength cells, combining clustering with ITP yields the highest gains. DFFHQN_X3 achieves a massive 88.9% speedup (9211 s to 1017 s) using ITP and clustering ($k_{\max} = 2$). DFFHQN_X4 achieves a 47.5% speedup using ITP and clustering ($k_{\max} = 3$), though larger cluster sizes again lead to unsatisfiability. For large-scale designs, early termination via relative gap is necessary to avoid timeouts. A 0.5% gap enables BUF_X16 to converge in 533 s with a reported cell width of 25 CPP. A 1.0% gap enables 2BDFFHQN_X1 to converge in 15773 s with a reported cell width of 25 CPP. In summary, acceleration techniques significantly reduce search complexity but require careful tuning of cluster size and gap tolerance to maintain satisfiability.

TABLE VII
CELL METRICS AND RUNTIME FOR 3:2 GR CELLS. CLST.: MAXIMUM CLUSTER SIZE $k_{\max}$. ITP: IDENTICAL TRANSISTOR PARTITIONING ENABLEMENT. GAP: RELATIVE GAP.

| Cell Name | #FET | #Net | Acceleration | | | Width (CPP) | WL (nm) | #M2 | RT (s) |
|---|---|---|---|---|---|---|---|---|---|
| | | | Clst. | ITP | Gap | | | | |
| DFFHQN_X1 | 24 | 17 | — | — | — | 14 | 2541.0 | 2 | 254 |
| | | | 2 | — | — | 14 | 2541.0 | 2 | 161 |
| | | | 4 | — | — | 14 | 2541.0 | 2 | **106** |
| DFFRNQ_X1 | 28 | 18 | — | — | — | 17 | 4166.0 | 4 | 11251 |
| | | | 2 | — | — | 17 | 4166.0 | 4 | **6512** |
| | | | 4 | — | — | — | — | — | *UNSAT* |
| DFFHQN_X3 | 32 | 17 | — | — | — | 16 | 2826.5 | 2 | 9211 |
| | | | 2 | ✓ | — | 16 | 2826.5 | 2 | **1017** |
| | | | 3 | ✓ | — | — | — | — | *UNSAT* |
| DFFHQN_X4 | 34 | 17 | — | — | — | 19 | 2966.5 | 2 | 133523 |
| | | | 3 | ✓ | — | 19 | 2966.5 | 2 | **70156** |
| | | | 4 | ✓ | — | — | — | — | *UNSAT* |
| 2BDFFHQN_X1 | 44 | 29 | — | — | — | 25 | 5440.5 | 2 | *Timeout* |
| | | | 4 | — | 0.5% | 25 | 5060.5 | 2 | 146785 |
| | | | 4 | — | 1.0% | 25 | 6042.0 | 2 | **15773** |
| BUF_X16 | 48 | 5 | — | — | — | 25 | 2876.0 | 0 | *Timeout* |
| | | | — | ✓ | — | 25 | 2017.5 | 0 | *Timeout* |
| | | | — | ✓ | 0.5% | 25 | 2876.0 | 0 | **533** |

## E. Scalability Comparison with Other Tools for PROBE3.0

Table VIII compares layout generation tools for D flip-flop designs using 3:2 GR against the best runtime results from Table VII. All tools were tested on the same machine with a 150000 s timeout, using their respective acceleration techniques.

TABLE VIII
CELL METRICS AND RUNTIME (RT) FOR DIFFERENT LAYOUT GENERATION TOOLS UNDER 3:2 GR AND 2F4T CONFIGURATION FOR D FLIP-FLOPS.

| Cell Name | #FET | #Net | Tool | Width (CPP) | WL (nm) | #M2 | RT (s) |
|---|---|---|---|---|---|---|---|
| DFFHQN_X1 | 24 | 17 | PROBE3.0 [8] | 18 | 3803.0 | 2 | 17010 |
| | | | SMTCell-GR [5] | 18 | 4504.5 | 2 | 9075 |
| | | | SO3 [23] | 14 | 2577.0 | 2 | 262 |
| | | | Our work | 14 | 2541.0 | 2 | 106 |
| DFFRNQ_X1 | 28 | 18 | PROBE3.0 [8] | 20 | 5550.0 | 5 | 4576 |
| | | | SMTCell-GR [5] | — | — | — | *Timeout* |
| | | | SO3 [23] | 17 | 4197.0 | 4 | *Timeout* |
| | | | Our work | 17 | 4166.0 | 4 | 6512 |
| 2BDFFHQN_X1 | 44 | 29 | PROBE3.0 [8] | — | — | — | *Timeout* |
| | | | SMTCell-GR [5] | — | — | — | *Timeout* |
| | | | SO3 [23] | — | — | — | *Timeout* |
| | | | Our work | 25 | 6042.0 | 2 | 15773 |

Specifically, PROBE3.0 and SMTCell-GR use partitioned netlists from prior work [4] and the Z3 solver [15] with multi-threading. SO3 [23] uses Gurobi [36] with 12 workers (double our CP-SAT configuration) and restricts topology optimization, gate alignments, diffusion breaks, and occupying track to reduce runtime. Given variations in rule support and encoding across tools, width serves as the primary metric, with wirelength and M2 track count provided for reference. CPCell requires no prior solution data, allowing seamless application to unseen netlists. It consistently achieves the smallest (or tied) cell width (CPP) often with lower runtime.

## VII. BLOCK-LEVEL EVALUATIONS ON GEAR-RATIO-ENABLED CELL LAYOUTS

In this section, we explore example impacts of gear ratio at block-level. First, we confirm whether using offset cell layouts can in practice mitigate the placement legalization bottleneck. Second, we perform a comparative analysis of GR choices in terms of block-level PPA. Third, we evaluate IR-drop to show power integrity trade-offs introduced by GR alternatives.

These examples show how our framework can illuminate such phenomena as the following:

(1) Offset cell layouts can improve the block PPA while also reducing runtime.
(2) Although a smaller M1 pitch generally benefits block-level routing, it does not always lead to better PPA because of the trade-off between routing resources and wire capacitances.
(3) A denser power distribution network (PDN) enabled by a GR with a smaller M1 pitch does not necessarily lead to smaller IR-drop compared to the sparser PDN of a GR with a larger M1 pitch.

Standard cell LVS and parasitic extraction are implemented with Cadence Pegasus v21.3 [37] and Quantus v21.1 [38], and characterization is performed with Cadence Liberate v23.1 [39]. Cell layout signoff is implemented with KLayout [40]. Synthesis and place-and-route (P&R) are implemented with Synopsys Design Compiler R-2020.09 [41] and IC Compiler II R-2020.09 [42], respectively. IR-drop is measured with Cadence Voltus v21.1 [43]. P&R and IR-drop are executed using 16 threads on a server equipped with an Intel Xeon Gold 6148 CPU (2.40 GHz, 80 threads) and 384 GB of memory. Our scripts are available at [18].

### A. Gear Ratio & Offset Enablement

Prior work [5] identifies block-level legalization failure and proposes offset cells as a remedy. We evaluate 1:1, 3:2, and 5:3 GR libraries. For 3:2 GR, 15 nm-offset cells (3:2$^{\dagger}$) are grouped with 0-offset cells in an equivalent cell group (ECG).[3] For 5:3 GR, 9 nm- and 18 nm-offset cells (5:3$^{\dagger}$ and 5:3$^{\dagger\dagger}$) are similarly grouped with 0-offset cells.

Table IX shows data for JPEG [44] across target clock periods (TCPs) from 100 ps to 400 ps. Performance is represented by the effective clock period (CLKP): TCP minus worst negative slack. For each nominal TCP, we run the flow at TCP, TCP $\pm 0.1$ ps and TCP $\pm 0.2$ ps over a range of utilizations. At each utilization, runs that fail to route or have #DRV $> 300$ are discarded. We select the highest evaluated utilization at which at least one of the five runs remains, and report the median of each metric over all remaining runs. The medians are computed per metric and do not necessarily correspond to a single run, so the comparisons in Table IX show marginal trends across GR options. All evaluated runs are listed in our repository [18]. Each TCP is evaluated independently, and failure at one TCP is not assumed to imply failure at any other TCP. We compute the effective clock period as $\mathrm{TCP} - \mathrm{WNS}$, where WNS is the worst negative slack. Figure 17 shows the PPA comparison plot for 3:2 and 5:3 GR. For both gear ratios, adding offset variants reduces area with similar power and performance, except at the 100 ps TCP.

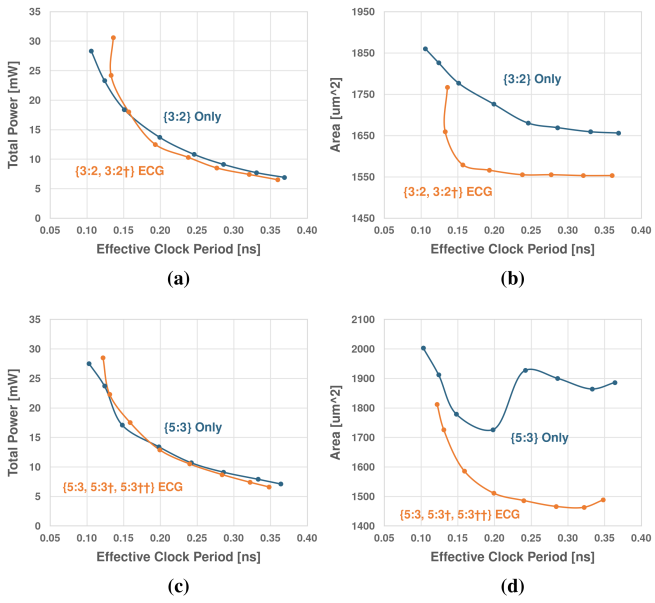



Fig. 17. Offset-cell PPA: (a) 3:2 GR total power; (b) 3:2 GR area; (c) 5:3 GR total power; (d) 5:3 GR area, each vs. effective CLKP.

Figure 18 shows P&R runtime comparison results. The runtime data confirm that ECGs yield shorter runtimes than 0-offset sets in most cases, possibly suggesting that finding legalizable physical variants takes less time than finding legalizable placement locations.

[3]Equivalent Cell Groups contain functionally identical physical variants, e.g., with different pin locations or colors. Variant-aware legalization swaps a misaligned cell for an aligned variant instead of moving it.

TABLE IX
COMPARISON OF BLOCK-LEVEL METRICS ACROSS TARGET CLOCK PERIODS

| | | | Target Clock Period (ns) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 0.10 | 0.12 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 |
| 1:1 GR | | #Insts | 60255 | 47718 | 44714 | 43774 | 43215 | 42965 | 42850 | 42700 |
| | | Wirelength ($\mu$m) | 92560 | 87659 | 82759 | 73234 | 69993 | 70875 | 68578 | 69116 |
| | | Total Power (mW) | 29.7 | 24.6 | 18.2 | 13.0 | 10.3 | 8.9 | 7.6 | 6.7 |
| | | Worst Negative Slack (ns) | -0.021 | -0.013 | 0.000 | 0.009 | 0.014 | 0.022 | 0.028 | 0.034 |
| | | Effective CLKP (ns) | 0.121 | 0.133 | 0.150 | 0.191 | 0.236 | 0.278 | 0.322 | 0.366 |
| | | Area ($\mu m^2$) | 2199 | 2017 | 1935 | 1886 | 1878 | 1836 | 1838 | 1836 |
| 3:2 GR | 0-offset Only | #Insts | 57889 | 54403 | 52951 | 47918 | 46307 | 45750 | 45319 | 45376 |
| | | Wirelength ($\mu$m) | 74528 | 71986 | 71602 | 72681 | 65940 | 64729 | 63707 | 64154 |
| | | Total Power (mW) | 28.3 | 23.3 | 18.4 | 13.7 | 10.8 | 9.1 | 7.7 | 6.9 |
| | | Worst Negative Slack (ns) | -0.006 | -0.004 | -0.001 | 0.001 | 0.004 | 0.014 | 0.019 | 0.031 |
| | | Effective CLKP (ns) | 0.106 | 0.124 | 0.151 | 0.199 | 0.246 | 0.286 | 0.331 | 0.369 |
| | | Area ($\mu m^2$) | 1860 | 1826 | 1777 | 1726 | 1680 | 1669 | 1659 | 1656 |
| | Mixed-offset | #Insts | 57155 | 44195 | 42117 | 41543 | 41207 | 41103 | 41106 | 41087 |
| | | Wirelength ($\mu$m) | 76773 | 79973 | 66652 | 59613 | 56396 | 55492 | 57627 | 55264 |
| | | Total Power (mW) | 30.6 | 24.2 | 18.0 | 12.5 | 10.3 | 8.5 | 7.4 | 6.5 |
| | | Worst Negative Slack (ns) | -0.036 | -0.013 | -0.007 | 0.007 | 0.012 | 0.023 | 0.029 | 0.040 |
| | | Effective CLKP (ns) | 0.136 | 0.133 | 0.157 | 0.193 | 0.238 | 0.277 | 0.321 | 0.360 |
| | | Area ($\mu m^2$) | 1767 | 1659 | 1579 | 1566 | 1555 | 1555 | 1553 | 1553 |
| 5:3 GR | 0-offset Only | #Insts | 59183 | 57455 | 52756 | 47062 | 54266 | 53125 | 52780 | 52663 |
| | | Wirelength ($\mu$m) | 77686 | 80350 | 68864 | 73755 | 72539 | 70100 | 70454 | 68210 |
| | | Total Power (mW) | 27.5 | 23.7 | 17.1 | 13.4 | 10.7 | 9.1 | 7.9 | 7.1 |
| | | Worst Negative Slack (ns) | -0.003 | -0.004 | 0.002 | 0.003 | 0.008 | 0.014 | 0.017 | 0.036 |
| | | Effective CLKP (ns) | 0.103 | 0.124 | 0.148 | 0.198 | 0.242 | 0.286 | 0.333 | 0.364 |
| | | Area ($\mu m^2$) | 2003 | 1912 | 1779 | 1726 | 1927 | 1900 | 1864 | 1886 |
| | Mixed-offset | #Insts | 54104 | 50761 | 41131 | 39900 | 39594 | 38890 | 38803 | 38731 |
| | | Wirelength ($\mu$m) | 72021 | 68720 | 71780 | 66803 | 64239 | 57964 | 55328 | 55335 |
| | | Total Power (mW) | 28.5 | 22.3 | 17.5 | 12.9 | 10.5 | 8.7 | 7.4 | 6.6 |
| | | Worst Negative Slack (ns) | -0.022 | -0.011 | -0.009 | 0.001 | 0.010 | 0.016 | 0.028 | 0.052 |
| | | Effective CLKP (ns) | 0.122 | 0.131 | 0.159 | 0.199 | 0.240 | 0.284 | 0.322 | 0.348 |
| | | Area ($\mu m^2$) | 1812 | 1726 | 1586 | 1511 | 1486 | 1466 | 1463 | 1488 |

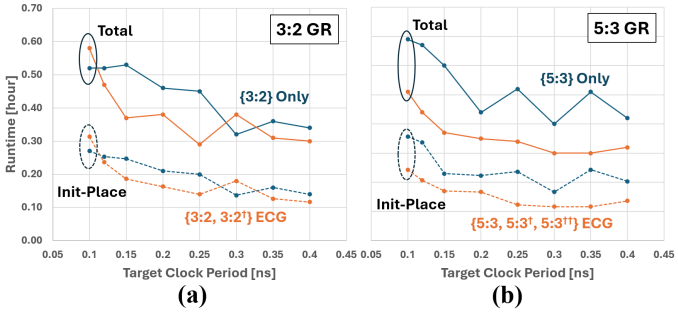



Fig. 18. Block-level P&R runtime comparison. (a) 3:2 GR; (b) 5:3 GR.

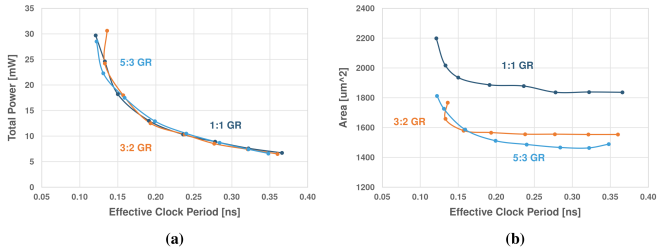



Fig. 19. PPA comparison for identifying the optimal GR in the DTCO context. (a) total power vs. effective clock period; (b) area vs. effective clock period.

## B. Choice of Optimal GR in DTCO Context

Smaller M1 pitch reduces routing congestion but increases wire capacitance and power, making GR selection a PPA trade-off for DTCO. Figure 19 compares GRs[4] and shows area benefits for 3:2 GR with power and performance comparable to 1:1 and 5:3 GRs, except at the 100 ps TCP.

(1) Figure 19(a) indicates that power consumption is comparable across all GRs over the entire frequency range.
(2) Figure 19(b) indicates that the 3:2 and 5:3 GRs provide significant area reductions compared to the 1:1 GR across the entire frequency range.
(3) Figure 19(b) reveals a crossover in area efficiency: the 5:3 GR requires the least area at lower frequencies, whereas the 3:2 GR becomes more area-efficient in the high-frequency regime.

These studies suggest that selection of the best GR option may depend on the target design parameters.

[4]ECGs are used for 3:2 and 5:3 GRs.

## C. Impact of Gear Ratio on IR-drop

Gear ratio is used not only for PPA benefit but also for IR-drop reduction. With a smaller M1 pitch, additional routing resources on M1 can be used to enable a finer PDN mesh to mitigate IR-drop. Figure 20(a) shows the setting of power stripe distance for each GR to 18 times the M1 pitch, which effectively places the worst IR-drop at approximately 10% of VDD at a clock period of 100 ps: 810 nm, 540 nm and 486 nm for the 1:1, 3:2 and 5:3 GR, respectively. Figure 20 also shows that the 3:2 GR has the worst IR-drop (77.0 mV), while the 1:1 GR has the lowest IR-drop (60.2 mV). The 3:2 and 5:3 GRs show a higher IR-drop than the 1:1 GR despite their denser PDNs, because their smaller areas induce higher current densities. From such IR-drop evaluation studies, a technology developer could, e.g., decide to apply the 5:3 GR to gain IR-drop benefits, accepting the higher cost required for denser M1 patterning. This highlights the necessity of holistic block-level evaluation in DTCO decision-making.

## D. Ablation Study on DTCO Features

We quantify the block-level contribution of two DTCO features: the M0 pin accessibility constraints of Section IV and

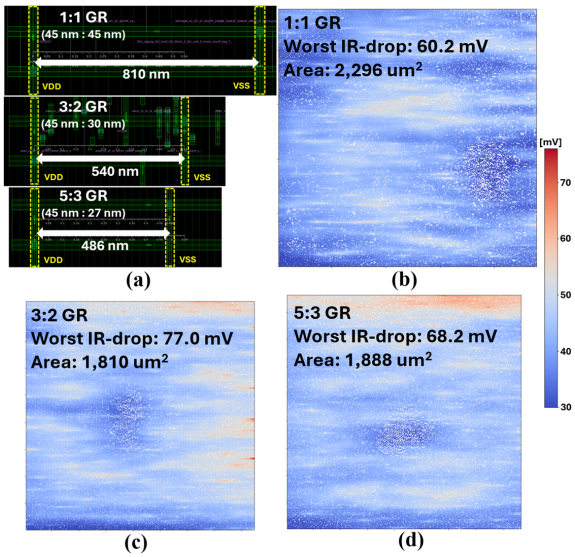



Fig. 20. GR-dependent static IR-drop with M1-pitch-scaled power stripe spacing: (a) stripe spacing; IR-drop maps for (b) 1:1, (c) 3:2, and (d) 5:3 GR.

TABLE X
PIN-ACCESSIBILITY METRICS FOR (a) PA_A, (b) PA_B, AND (c) PA_C (FIGURE 15), AT 0.90 UTILIZATION AND TCP = 0.10 ns. VALUES ARE ROUNDED TO TWO DECIMAL PLACES.

| Metric | Pin-Accessibility Configuration | | |
|---|---|---|---|
| | (a) | (b) | (c) |
| Avg. Cell Width (CPP) | 6.43 | 6.43 | 6.45 |
| Wirelength ($\mu$m) | 94964 | 94503 | 96273 |
| Total Power (mW) | 28.7 | 30.0 | 30.1 |
| Worst Negative Slack (ns) | -0.005 | -0.003 | -0.010 |
| Effective CLKP (ns) | 0.105 | 0.103 | 0.110 |
| Area ($\mu$m$^2$) | 2282 | 2282 | 2320 |
| #DRV | 277 | 11 | 0 |

parameterizable gate cuts. Both studies use JPEG [44] at 0.90 utilization and a target clock period of 0.1 ns.

We first assess downstream routability across configurations (a), (b) and (c) of Figure 15. Configuration (a) applies no pin access constraint, (b) applies pin separation (PS), and (c) applies both PS and minimum pin opening (MPO). Configurations (a) and (b) yield identical average cell width, while (c) widens the congested cell XOR2_X1 by one CPP to secure additional pin opening. Table X reports block-level wirelength, power, timing, area and post-route design rule violation count (#DRV). PS reduces the reported post-route DRV count from 277 to 11 without changing block area. With PS and MPO, the reported count is zero. Relative to the unconstrained library, the combined configuration increases block area by 1.7% and reported total power by 4.9%. However, improved routability is observed in the evaluated JPEG configuration.

We quantify the contribution of enabling gate cuts. Table XI compares configuration (c) with and without the 2-CPP gate cut. The gate cut reduces average cell width by 3.0% (6.65 to 6.45 CPP), with the reduction concentrated in cells whose transistor netlists contain crossings such as MUX2_X1, LHQ_X1 and DFFHQN_X1. At the block level this yields 6.5% area reduction (2482 to 2320 $\mu$m$^2$) and 3.3% wirelength reduction, at a cost of 1.3% power increase and an added 1 ps of WNS.

TABLE XI
CELL-LEVEL AND BLOCK-LEVEL METRICS WITH OR WITHOUT GATE CUT. BLOCK-LEVEL RESULTS USE 0.90 UTILIZATION AND TCP = 0.10 ns.

| Metric | Gate Cut Setting | |
|---|---|---|
| | w/o Gate Cut | w/ Gate Cut (2-CPP) |
| Avg. Cell Width (CPP) | 6.65 | 6.45 |
| Wirelength ($\mu$m) | 99538 | 96273 |
| Total Power (mW) | 29.7 | 30.1 |
| Worst Negative Slack (ns) | -0.009 | -0.010 |
| Effective CLKP (ns) | 0.109 | 0.110 |
| Area ($\mu$m$^2$) | 2482 | 2320 |

## VIII. CONCLUSION

CPCell extends gear-ratio studies with a standard-cell layout framework supporting arbitrary gear ratios and offset variants for DTCO under PROBE3.0. Its acceleration techniques reduce CP-SAT runtime by up to 88.96%, enabling generation for netlists with up to 48 transistors. Cell-level experiments show reduced average width for 3:2 and 5:3 GRs, with cell-dependent wirelength changes. Block-level results show area and wirelength benefits, with PPA and IR-drop trade-offs that depend on target conditions. CPCell connects cell optimization to block implementation for systematic technology and architecture evaluation.

## ACKNOWLEDGMENT

The authors acknowledge the use of ChatGPT for improving the English phrasing and grammatical quality of the manuscript.

## REFERENCES

[1] L. W. Liebmann, V. Gerousis, P. Gutwin, X. Zhu, and J. Petykiewicz, "Exploiting regularity: breakthroughs in sub-7nm place-and-route," in *Proc. SPIE 10148, Design-Process-Technology Co-Optimization for Manufacturability XI*, 2017, Art. no. 101480E.

[2] IEEE International Roadmap for Devices and Systems (IRDS), "Executive summary 2022," IEEE, Piscataway, NJ, USA, 2022, doi: 10.60627/C13Z-V363.

[3] C. Chidambaram, A. B. Kahng, M. Kim, G. Nallapati, S. C. Song, and M. Woo, "A novel framework for DTCO: Fast and automatic routability assessment with machine learning for sub-3nm technology options," in *Proc. Symp. VLSI Technol.*, 2021, pp. 1–2.

[4] D. Lee et al., "SP&R: SMT-based simultaneous place-and-route for standard cell synthesis of advanced nodes," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 10, pp. 2142–2155, Oct. 2021.

[5] C.-K. Cheng, A. B. Kahng, B. Lin, Y. Wang, and D. Yoon, "Gear-ratio-aware standard cell layout framework for DTCO exploration," in *Proc. ACM Int. Workshop Syst.-Level Interconnect Pathfinding (SLIP)*, 2023, Art. no. 2, pp. 1–10, doi: 10.1145/3632409.3640475.

[6] J. Jeong, S. Chung, K. Jo, and T. Kim, "Synthesis and utilization of standard cells amenable to gear ratio of gate-metal pitches for improving pin accessibility," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2023, pp. 1–6.

[7] C.-K. Cheng, B. Kang, B. Lin, and Y. Wang, "Invited: Scaling standard cell layout using track height compression and design technology co-optimization," in *Proc. ACM Int. Symp. Phys. Design (ISPD)*, 2025, pp. 11–19.

[8] S. Choi et al., "PROBE3.0: A systematic framework for design-technology pathfinding with improved design enablement," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 43, no. 4, pp. 1218–1231, Apr. 2024.

[9] UC San Diego VLSI Lab, *SMTCellUCSD: Cell Layout Generation for DTCO/STCO Exploration Toolkit*, ver. 1.5 (preview), GitHub, commit 8d292bb, May 23, 2025. [Online]. Available: https://github.com/ckcheng ucsd/SMTCellUCSD

[10] C.-K. Cheng, C.-T. Ho, D. Lee, B. Lin, and D. Park, "Complementary-FET (CFET) standard cell synthesis framework for design and system technology co-optimization using SMT," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 6, pp. 1178–1191, Jun. 2021.

[11] C.-K. Cheng, C.-T. Ho, D. Lee, and B. Lin, "Monolithic 3D semiconductor footprint scaling exploration based on VFET standard cell layout

methodology, design flow, and EDA platform," *IEEE Access*, vol. 10, pp. 65971–65981, 2022.
[12] I. Kang, D. Park, C. Han, and C.-K. Cheng, "Fast and precise routability analysis with conditional design rules," in *Proc. 20th Syst. Level Interconnect Predict. Workshop (SLIP)*, 2018, pp. 4:1–4:8.
[13] L. Perron and V. Furnon, *OR-Tools*, ver. 9.14, Google, Jun. 19, 2025. [Online]. Available: https://github.com/google/or-tools/releases/tag/v9.14
[14] L. de Moura and N. Bjørner, "Z3: An efficient SMT solver," in *Proc. 14th Int. Conf. Tools Algorithms Construction Anal. Syst. (TACAS)*, ser. Lecture Notes in Computer Science, vol. 4963, 2008, pp. 337–340.
[15] C. W. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli, "Satisfiability modulo theories," in *Handbook of Satisfiability*, 2nd ed., A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, Eds., ser. Frontiers in Artificial Intelligence and Applications, vol. 336. Amsterdam, The Netherlands: IOS Press, 2021, ch. 33, pp. 1267–1329.
[16] V. Vashishtha, M. Vangala, and L. T. Clark, "ASAP7 predictive design kit development and cell design technology co-optimization: Invited paper," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2017, pp. 992–998.
[17] A. B. Kahng, S. Kang, S. Kim, J. Lee, and D. Yoon, "Au-MEDAL: Adaptable grid router with metal edge detection and layer integration," in *Proc. 31st Asia South Pacific Design Autom. Conf. (ASP-DAC)*, 2026, pp. 1294–1300.
[18] ABKGroup, *CPCellUCSD: CP-SAT-Based Standard Cell Layout Generator*, GitHub, commit 28eb91f, Sep. 6, 2026. [Online]. Available: https://github.com/ABKGroup/CPCellUCSD
[19] X. Xu, B. Cline, G. Yeric, B. Yu, and D. Z. Pan, "Self-aligned double patterning aware pin access and standard cell layout co-optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 5, pp. 699–712, May 2015.
[20] K. Baek and T. Kim, "CSyn-fp: Standard cell synthesis of advanced nodes with simultaneous transistor folding and placement," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 43, no. 2, pp. 627–640, Feb. 2024.
[21] H. Ren and M. Fojtik, "Invited—NVCell: Standard cell layout in advanced technology nodes with reinforcement learning," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, 2021, pp. 1291–1294.
[22] C.-T. Ho et al., "NVCell 2: Routability-driven standard cell layout in advanced nodes with lattice graph routability model," in *Proc. ACM Int. Symp. Phys. Design (ISPD)*, 2023, pp. 44–52.
[23] C.-K. Cheng, A. B. Kahng, B. Kang, S. Kang, J. Lee, and B. Lin, "SO3-Cell: Standard cell layout automation framework for simultaneous optimization of topology, placement, and routing," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2025, pp. 1–9.
[24] K. Han, A. B. Kahng, and H. Lee, "Evaluation of BEOL design rule impacts using an optimal ILP-based detailed router," in *Proc. 52nd ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, 2015, Art. no. 68, pp. 1–6.
[25] T. C. Hu, "Multi-commodity network flows," *Oper. Res.*, vol. 11, no. 3, pp. 344–360, May/Jun. 1963.
[26] K. Han, A. B. Kahng, H. Lee, and L. Wang, "ILP-based co-optimization of cut mask layout, dummy fill, and timing for sub-14nm BEOL technology," in *Proc. SPIE Photomask Technol.*, vol. 9635, 2015, Art. no. 96350E.
[27] D. Park, I. Kang, Y. Kim, S. Gao, B. Lin, and C.-K. Cheng, "ROAD: Routability analysis and diagnosis framework based on SAT techniques," in *Proc. ACM Int. Symp. Phys. Design (ISPD)*, 2019, pp. 65–72.
[28] D. Lee, "Logical reasoning techniques for VLSI applications," Ph.D. dissertation, Univ. California, San Diego, La Jolla, CA, USA, 2022.
[29] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Inf. Process. Lett.*, vol. 31, no. 1, pp. 7–15, Apr. 1989.
[30] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Proc. 17th Pacific-Asia Conf. Knowl. Discovery Data Mining (PAKDD)*, 2013, pp. 160–172.
[31] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[32] C. Sechen and A. Sangiovanni-Vincentelli, "The TimberWolf placement and routing package," *IEEE J. Solid-State Circuits*, vol. 20, no. 2, pp. 510–522, Apr. 1985.
[33] L. Perron, F. Didier, and S. Gay, "The CP-SAT-LP solver (invited talk)," in *Proc. 29th Int. Conf. Principles Practice Constraint Program. (CP)*, ser. Leibniz International Proceedings in Informatics, vol. 280, 2023, pp. 3:1–3:2.
[34] A. B. Kahng, J. Kuang, W.-H. Liu, and B. Xu, "In-route pin access-driven placement refinement for improved detailed routing convergence," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 3, pp. 784–788, Mar. 2022.
[35] F.-K. Sun, H. Chen, C.-Y. Chen, C.-H. Hsu, and Y.-W. Chang, "A multithreaded initial detailed routing algorithm considering global routing guides," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2018, Art. no. 82, pp. 1–7.
[36] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2025. [Online]. Available: https://www.gurobi.com
[37] Cadence Design Systems, Inc., *Pegasus Verification System*. [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/silicon-signoff/pegasus-verification-system.html
[38] Cadence Design Systems, Inc., *Quantus Extraction Solution*. [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/silicon-signoff/quantus-extraction-solution.html
[39] Cadence Design Systems, Inc., *Liberate Characterization Solution*. [Online]. Available: https://www.cadence.com/en_US/home/tools/custom-ic-analog-rf-design/library-characterization/liberate-trio-characterization-suite.html
[40] M. Köfferlein, *KLayout Layout Viewer and Editor*. [Online]. Available: https://www.klayout.de
[41] Synopsys, Inc., *Design Compiler*. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/design-compiler.html
[42] Synopsys, Inc., *IC Compiler II*. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/physical-implementation/ic-compiler.html
[43] Cadence Design Systems, Inc., *Voltus IC Power Integrity Solution*. [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/silicon-signoff/voltus-ic-power-integrity-solution.html
[44] A. Tekyildiz, C. Gutierrez, and M. Krepa, *JPEG Encoder (mkjpeg)*, OpenCores, updated Sep. 23, 2020. [Online]. Available: https://opencores.org/projects/mkjpeg
[45] R. Usselmann, *AES IP Core*, OpenCores, updated Jun. 10, 2018. [Online]. Available: https://opencores.org/projects/aes_core
[46] OpenHW Group, *CORE-V CV32E40P User Manual*. [Online]. Available: https://docs.openhwfoundation.org/projects/cv32e40p-user-manual/



**Chung-Kuan Cheng** is Distinguished Professor of CSE and Adjunct Professor of ECE at the University of California, San Diego. His research interests include machine learning and design automation for microelectronic circuits. He received the Ph.D. degree in Electrical Engineering and Computer Sciences from the University of California, Berkeley.



**Andrew B. Kahng** is Distinguished Professor of CSE and ECE at the University of California, San Diego. His interests include IC physical design, the design-manufacturing interface, combinatorial optimization, and AI/ML for EDA and IC design. He received the Ph.D. degree in Computer Science from the University of California, San Diego.



**Bill Lin** received BS, MS and PhD degrees in EECS from UC Berkeley. He is a Professor of Electrical and Computer Engineering at UC San Diego, affiliated with the Center for Wireless Communications and the Center for Networked Systems, CNS. He is also currently the Associate Dean for Research for the Jacobs School of Engineering.



**Yucheng Wang** is a Ph.D. student in the Computer Science Department of the University of California at San Diego. He received his bachelor's degree in Computer Science from Purdue University, West Lafayette, in 2021. His research interests include standard-cell layout automation, design technology co-optimization strategies and graph visualization.



**Dooseok Yoon** received the M.S. degree in electrical and computer engineering from the University of California at San Diego, La Jolla, CA, USA, in 2024. He is currently pursuing the Ph.D. degree at the University of California at San Diego, La Jolla. His research interests include VLSI physical design and design-technology co-optimization.