Automated Cross-PDK Standard Cell Migration with Graph Isomorphism-Based Topology Equivalence Verification

Jaewoo Oh **Heeyoung Ahn** **Seongwook Ha**

Abstract

Migrating verified design assets across process nodes is increasingly relevant in standard cell library development. However, design rules between heterogeneous nodes do not scale uniformly: minimum poly width, metal spacing, and well enclosure margins each follow distinct trajectories. This non-uniform scaling can inadvertently alter connectivity during migration. Conventional LVS is designed for within-PDK layout–schematic checking and does not directly support cross-PDK netlist-to-netlist comparison due to model naming and extraction semantic differences.

We present a framework for automated standard cell migration from SkyWater 130 nm to FreePDK45, with topology equivalence verification based on graph isomorphism. Our approach models circuit netlists as device-net bipartite graphs and applies the VF2 algorithm to determine structural equivalence. Unlike LVS, VF2 operates on an abstract connectivity graph, enabling PDK-independent comparison under explicit device-type normalization. It also produces a vertex-level bijection mapping each source device to its migrated counterpart, facilitating systematic debugging. To accommodate the physical source/drain (S/D) symmetry of MOSFETs in the target PDKs, we define an extended isomorphism with relaxed edge-matching conditions permitting S/D interchange. This assumption is specific to symmetric MOSFET models and requires validation for asymmetric devices.

The framework comprises KLayout-based netlist extraction, NetworkX-based VF2 checking, and gdstk-based layout generation. We evaluate 42 standard cells spanning 10 gate types; 32 cells (76.2%) pass DRC, LVS, and topology verification. The remaining 10 cells fail due to *contact uniqueness violation*: a structural limitation where a single shared diffusion contact must connect to two distinct nets, which is physically unrealizable in linear layouts. We provide a detailed structural analysis and discuss mitigation strategies including non-linear placement and diffusion break insertion.

1 Introduction

1.1 Background and Motivation

Standard cell libraries encode process-specific physical constraints into DRC-clean, LVS-correct layouts. Developing such libraries requires substantial effort in circuit design, layout construction, characterization, and multi-corner verification SkyWater [2020], FreePDK45. When porting designs to a different process node, the associated library must be reconstructed to comply with new design rules.

A naïve migration approach is optical shrink, uniformly scaling all coordinates by a constant factor. In practice, design rules between heterogeneous nodes exhibit non-uniform scaling. Comparing SkyWater 130 nm and FreePDK45, minimum poly width shrinks from 130 nm to 50 nm ($\sim 2.6\times$), while

Preprint. Under review.

contact dimensions, metal pitch, and well margins follow different ratios. Some rules introduce constraints absent in the source process. Consequently, optically shrunk layouts incur numerous DRC violations, and correcting these violations risks altering circuit connectivity.

1.2 Problem Statement

Migration correctness requires two conditions. First, the layout must comply with the target PDK's design rules (verifiable via DRC). Second, the migrated layout must preserve the source circuit's topology—the same transistors connected in the same configuration.

DRC addresses only geometric constraints and is agnostic to connectivity. LVS verifies connectivity but operates within a single-PDK context, comparing a layout-extracted netlist against a reference schematic expressed in the same PDK's device models. Cross-PDK comparison requires matching netlists from *different* PDKs, where device model names (e.g., `sky130_fd_pr__nfet_01v8` vs. `NMOS_VTL`) and extraction semantics differ. Applying conventional LVS to this task would require non-trivial normalization.

We formalize the problem as follows. Given source layout L_{src} in PDK_A and migrated layout L_{dst} in PDK_B, determine whether the extracted netlists N_{src} and N_{dst} are topologically equivalent. Equivalence requires identical transistor count and type composition (NMOS/PMOS), with terminal connections mapping to the same logical nets, subject to S/D interchange permissibility for symmetric MOSFET models.

1.3 Scope and Contributions

We target combinational logic cell migration from SkyWater SKY130 to FreePDK45. Sequential elements, analog circuits, and performance optimization are outside scope. Layout generation employs cell-type-specific templates; we do not claim PDK-agnostic layout automation. Our contributions center on topology-equivalence verification and failure characterization, while layout generation is intentionally template-scoped.

This paper makes the following contributions:

- A cross-PDK topology verification methodology based on VF2 graph isomorphism, operating on PDK-independent graph abstractions under device-type normalization and producing explicit device-level mappings.
- An extended isomorphism definition with relaxed edge-matching conditions accommodating MOSFET S/D symmetry, with clearly stated applicability conditions.
- An end-to-end migration pipeline integrating KLayout extraction, NetworkX-based VF2 verification, and gdstk layout generation.
- A systematic analysis of *contact uniqueness violation* as a structural failure mode in linear layouts, with formal characterization of the topological conditions under which it arises.

2 Related Work

2.1 Layout Migration Techniques

Prior work on layout migration falls into rule-based and optimization-based categories.

Rule-based methods apply predefined transformation rules to produce target-compliant layouts. Heng et al. Heng et al. [1997] proposed a legalization technique minimizing perturbation while resolving DRC violations through constrained coordinate adjustment. While effective for intra-PDK legalization, extending rule-based methods to cross-PDK migration requires defining transformation rules that account for non-linear design rule relationships.

Optimization-based methods formulate layout generation as constraint satisfaction or mathematical programming, encoding design rules as constraints and area or wirelength as objectives. These methods can provide DRC compliance guarantees but require separate topology verification.

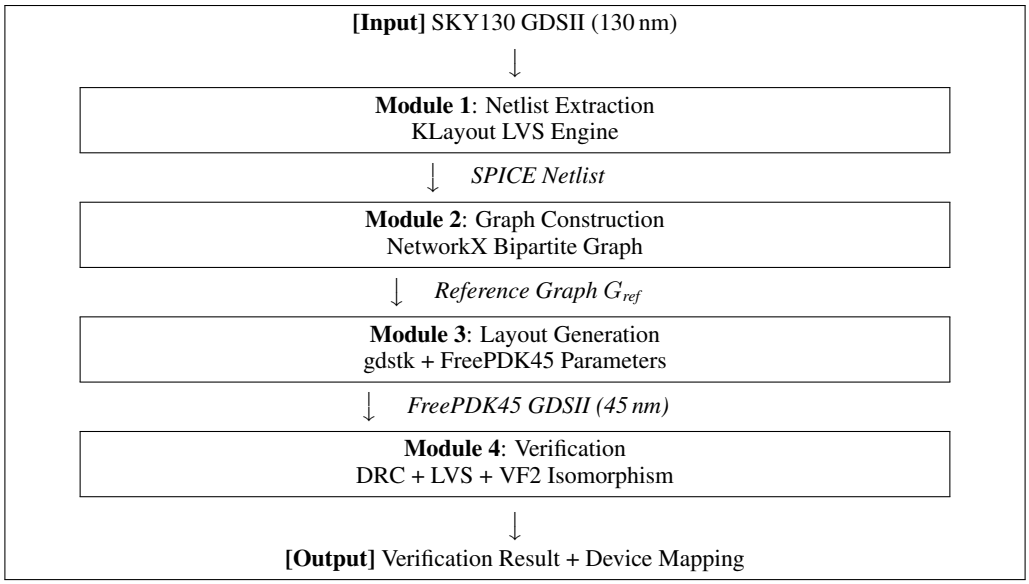


Figure 1: System architecture. DRC validates geometry; LVS verifies intra-PDK consistency; VF2 verifies cross-PDK topology equivalence.

2.2 Circuit Equivalence Verification

LVS is the standard methodology for verifying layout–schematic consistency. Commercial tools (Calibre, Assura, KLayout) assume the layout and schematic share the same PDK context. Cross-PDK application introduces three challenges: incompatible device model names, differing net naming conventions, and PDK-specific extraction procedures. Additionally, LVS provides binary PASS/-FAIL verdicts without cross-PDK diagnostic information.

2.3 Graph Isomorphism and VF2

Graph isomorphism determines whether two graphs are structurally identical. The problem is in NP but not known to be NP-complete or in P. VF2 Cordella et al. [2004] is a state-space search algorithm for subgraph isomorphism that extends partial mappings while applying feasibility pruning. It is implemented in NetworkX Hagberg et al. [2008] and widely used in molecular structure comparison and pattern recognition.

We adopt VF2 for three reasons. First, it operates on abstract graph representations; once netlists are converted to device-net bipartite graphs with normalized type attributes, PDK-specific naming is eliminated. Second, when isomorphism exists, VF2 produces an explicit bijection between vertex sets, providing device-level correspondence useful for debugging. Third, NetworkX's `GraphMatcher` accepts user-defined compatibility functions, enabling domain-specific matching conditions such as S/D interchange.

3 Proposed Methodology

3.1 System Architecture

The framework consists of four modules (Figure 1), each with well-defined interfaces, implemented in Python.

Module 1 invokes KLayout KLayout LVS in batch mode with the SKY130 rule deck to extract SPICE netlists.

Module 2 parses netlists and constructs circuit graphs. PDK-specific model names are normalized to binary NMOS/PMOS classification; net names become abstract identifiers. The resulting graph is a PDK-independent topological representation.

Table 1: Tools and libraries.

Module	Tool	Version	Purpose
Extraction	KLayout	0.28+	LVS-based SPICE extraction
Graph	NetworkX	3.0+	Graph construction & VF2
Layout	gdstk	0.9+	GDSII generation
Verification	KLayout	0.28+	DRC/LVS checking
Pipeline	Python	3.10+	Orchestration

Module 3 applies cell-type templates with FreePDK45 parameters to generate layouts using gdstk gdstk. Template-based generation is a deliberate scoping decision; the focus is verification methodology and failure analysis.

Module 4 performs three independent checks: DRC for geometric compliance, LVS for intra-PDK consistency, and VF2 for cross-PDK topology equivalence. A cell passes only if all three succeed.

Table 1 summarizes the tools used.

3.2 Netlist Extraction and Parsing

KLayout's LVS engine is invoked via command-line interface. For SKY130, we use the rule deck from `efabless/klayout-sky130`; for FreePDK45, the `lvs_freepdk45.lylvs` deck from NCSU.

Extracted netlists follow standard SPICE format:

$$\texttt{M}\langle\text{name}\rangle \quad \langle D\rangle \quad \langle G\rangle \quad \langle S\rangle \quad \langle B\rangle \quad \langle\text{model}\rangle \tag{1}$$

The parser extracts device name, terminal nets, and model name using regular expressions. Device type is classified by substring matching: `nfet`/`nmos` $\rightarrow$ NMOS; `pfet`/`pmos` $\rightarrow$ PMOS. This normalization eliminates PDK-specific model name differences.

3.3 Circuit Graph Modeling

Parsed netlists are modeled as bipartite graphs.

[Circuit Graph] Given device set $\mathcal{D} = \{d_1, \ldots, d_n\}$, the circuit graph $G = (V, E)$ is defined as:

Vertex set:

$$V_D = \{\texttt{DEV}_i : d_i \in \mathcal{D}\}, \quad V_N = \{n : n \in \textstyle\bigcup_{d\in\mathcal{D}}\{d.D, d.G, d.S\}\} \tag{2}$$

$$V = V_D \cup V_N \tag{3}$$

Edge set:

$$E = \{(\texttt{DEV}_i, n, \tau) : d_i \in \mathcal{D},\ \tau \in \{D, G, S\},\ n = d_i.\tau\} \tag{4}$$

Each edge carries terminal type τ. Device vertices have attribute `device_type` $\in \{\texttt{NMOS}, \texttt{PMOS}\}$.

This representation strips PDK-specific information, leaving only device types and connectivity for direct cross-PDK comparison.

3.4 MOSFET Source/Drain Symmetry

In MOSFET fabrication, source and drain regions are formed through identical implantation steps and are physically symmetric about the channel. For **symmetric device models**, interchanging S/D terminals does not alter electrical characteristics.

We state the applicability conditions explicitly. The SKY130 models (`sky130_fd_pr__nfet_01v8`, `pfet_01v8`) and FreePDK45 baseline models are symmetric devices where S/D interchange preserves electrical equivalence. This assumption does not hold universally: devices with asymmetric lightly doped drain profiles, strong layout-dependent effects, or explicitly asymmetric parameters

may exhibit different behavior upon terminal interchange. The extended isomorphism below applies only to symmetric models; for other models, S/D symmetry should be validated through SPICE simulation. Such validation was not performed in this work.

For symmetric models, the following configurations are equivalent:

$$\text{Config. A:} \quad D = Y,\ G = A,\ S = \text{VDD} \tag{5}$$

$$\text{Config. B:} \quad D = \text{VDD},\ G = A,\ S = Y \tag{6}$$

3.5 Extended Graph Isomorphism

To accommodate S/D symmetry, we relax the standard isomorphism definition.

[Extended Graph Isomorphism] Two circuit graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are **extended-isomorphic** if there exists a bijection $f : V_1 \to V_2$ satisfying:

C1 (Vertex type preservation): $\forall\, v \in V_1$: $\texttt{type}(v) = \texttt{type}(f(v))$.

C2 (Device type preservation): $\forall\, v \in V_D$: $\texttt{dtype}(v) = \texttt{dtype}(f(v))$.

C3 (Connectivity with S/D relaxation): $\forall\, (u, v, \tau) \in E_1$, $\exists\, (f(u), f(v), \tau') \in E_2$ such that $\tau = \tau'$ or $\{\tau, \tau'\} = \{D, S\}$.

Gate connections must match exactly; source and drain connections may interchange. We implement this through custom compatibility functions:

$$\phi_V(v_1, v_2) = \begin{cases} \top, & \texttt{type}(v_1) = \texttt{type}(v_2) = \texttt{net} \\ \top, & \texttt{type}(v_1) = \texttt{type}(v_2) = \texttt{device} \wedge \\ & \texttt{dtype}(v_1) = \texttt{dtype}(v_2) \\ \bot, & \text{otherwise} \end{cases} \tag{7}$$

$$\phi_E(e_1, e_2) = \begin{cases} \top, & \tau(e_1) = \tau(e_2) \\ \top, & \{\tau(e_1), \tau(e_2)\} = \{D, S\} \\ \bot, & \text{otherwise} \end{cases} \tag{8}$$

For asymmetric models, ϕ_E reduces to exact matching ($\tau(e_1) = \tau(e_2)$ only), yielding standard isomorphism. The framework architecture remains unchanged; only compatibility function configuration differs.

3.6 Topology Verification Algorithm

Algorithm 1 summarizes the verification procedure. A two-phase approach avoids invoking VF2 on trivially non-matching pairs: quick screening checks transistor count and type composition before full isomorphism checking.

VF2's worst-case complexity is $O(n! \cdot n)$, but circuit graphs in our experiments contain 8–16 vertices with regular bipartite structure. Execution time was below 1 ms for all cells.

The device mapping returned on PASS provides traceability for debugging and validation.

3.7 Layout Generation

The layout module applies cell-type templates with FreePDK45 parameters to produce GDSII layouts. Template-based generation is adopted because the focus is verification methodology and failure analysis; general-purpose layout generation is an orthogonal effort.

3.7.1 Cell Width Computation

Cell width W is determined by contact count n_c, poly gate count n_p, and DRC parameters:

$$W = 2m + n_c \cdot c_s + n_p \cdot p_w + 2 \cdot e_{ca} + (n_c + n_p - 1) \cdot s_{cp} \tag{9}$$

Table 2 lists parameter values.

Algorithm 1 Extended-isomorphism-based topology verification.

Require: Reference netlist N_{ref}, migrated netlist N_{mig}
Ensure: Verdict $v \in \{\text{PASS}, \text{FAIL}\}$, device mapping M
// Phase 1: Quick Screening
$D_{\text{ref}} \leftarrow \text{ParseNetlist}(N_{\text{ref}})$
$D_{\text{mig}} \leftarrow \text{ParseNetlist}(N_{\text{mig}})$
if $|D_{\text{ref}}^{\text{NMOS}}| \neq |D_{\text{mig}}^{\text{NMOS}}|$ **or** $|D_{\text{ref}}^{\text{PMOS}}| \neq |D_{\text{mig}}^{\text{PMOS}}|$ **then**
  return FAIL, $\emptyset$
end if
// Phase 2: VF2 Isomorphism
$G_{\text{ref}} \leftarrow \text{BuildGraph}(D_{\text{ref}})$
$G_{\text{mig}} \leftarrow \text{BuildGraph}(D_{\text{mig}})$
GM $\leftarrow$ GraphMatcher$(G_{\text{ref}}, G_{\text{mig}}, \phi_V, \phi_E)$
if GM.is_isomorphic() **then**
  return PASS, GM.mapping
else
  return FAIL, $\emptyset$
end if

Table 2: FreePDK45 parameters for cell width computation.

Symbol	Description	Value (nm)
m	Cell boundary margin	50
c_s	Contact size	65
p_w	Poly width	50
e_{ca}	Contact-to-active enclosure	5
s_{cp}	Contact-to-poly spacing	90

3.7.2 Grid Snapping

All coordinates snap to the manufacturing grid ($g = 2.5\,\text{nm}$):

$$\texttt{snap}(x) = \left\lfloor \frac{x}{g} + 0.5 \right\rfloor \cdot g \tag{10}$$

3.7.3 Layer Stack

Table 3 summarizes FreePDK45 layers.

4 Experiments

4.1 Benchmark

We evaluate the framework on 42 standard cells from SkyWater SKY130 `sky130_fd_sc_hd` (Table 4). All available drive-strength variants are included for each type.

The benchmark is limited to combinational gates due to template-based generation constraints. The verification methodology itself is agnostic to cell complexity.

4.2 Verification Criteria

A cell passes migration if it satisfies all three checks:

DRC: FreePDK45 design rule compliance via KLayout DRC; zero violations required. Table 5 lists key rules.

LVS: Intra-PDK netlist–schematic consistency within FreePDK45.

VF2: Cross-PDK topology equivalence between SKY130-extracted and FreePDK45-extracted netlists.

Table 3: FreePDK45 layout layers.

Number	Layer	Purpose
1	Active	NMOS/PMOS diffusion regions
2	PWELL	NMOS well region
3	NWELL	PMOS well region
4	NPLUS	N+ implant (NMOS S/D)
5	PPLUS	P+ implant (PMOS S/D)
9	Poly	Gate electrode
10	Contact	Metal-to-diffusion/poly via
11	Metal1	First metal routing layer

Table 4: Benchmark standard cells.

Type	Function	PMOS	NMOS	Variants
INV	$Y = \overline{A}$	1	1	7
BUF	$Y = A$	2	2	7
NAND2	$Y = \overline{A \cdot B}$	2	2	4
NOR2	$Y = \overline{A + B}$	2	2	4
AND2	$Y = A \cdot B$	3	3	4
NAND3	$Y = \overline{ABC}$	3	3	3
NOR3	$Y = \overline{A + B + C}$	3	3	3
AND3	$Y = ABC$	4	4	3
OR2	$Y = A + B$	3	3	4
OR3	$Y = A + B + C$	4	4	3
Total				**42**

The three checks are complementary: DRC verifies geometry, LVS verifies intra-PDK connectivity, VF2 verifies cross-PDK equivalence.

4.3 Results

Table 6 summarizes per-type results. Of 42 cells, 32 (76.2%) pass all verification stages.

The sharp type-level dichotomy—seven types at 100%, three at 0%—suggests the failure mode is primarily topology-driven rather than instance-specific.

4.4 Visual Comparison

Figures 2 and 3 show source and migrated layouts for INV_16 and BUF cells.

4.5 Device Mapping Output

VF2 produces an explicit bijection on PASS. Table 7 shows an example for INV_1.

This mapping enables device-level traceability across PDK boundaries.

5 Failure Analysis

5.1 Compound Gate Structure

The 10 failing cells share a common structure: compound gates formed by cascading a basic gate with an output inverter.

$$\text{AND3} = \text{NAND3} + \text{INV} \tag{11}$$

$$\text{OR2} = \text{NOR2} + \text{INV} \tag{12}$$

$$\text{OR3} = \text{NOR3} + \text{INV} \tag{13}$$

Table 5: Key FreePDK45 design rules.

Rule	Description	Value (nm)
POLY.1	Min. poly width	50
POLY.2	Min. poly spacing	75
POLY.3	Poly extension over active	55
ACTIVE.1	Min. active width	90
ACTIVE.2	Min. active spacing	80
CONTACT.1	Contact size	65×65
CONTACT.2	Contact spacing	75
CONTACT.6	Contact-to-poly spacing	90
METAL1.1	Min. Metal1 width	65
METAL1.2	Min. Metal1 spacing	65
METAL1.3	Metal1 contact enclosure	35
WELL.4	Min. well width	200

Table 6: Migration results by cell type.

Type	Total	Pass	Fail	Rate
INV	7	7	0	100.0%
BUF	7	7	0	100.0%
NAND2	4	4	0	100.0%
NOR2	4	4	0	100.0%
AND2	4	4	0	100.0%
NAND3	3	3	0	100.0%
NOR3	3	3	0	100.0%
AND3	3	0	3	0.0%
OR2	4	0	4	0.0%
OR3	3	0	3	0.0%
Total	**42**	**32**	**10**	**76.2%**

5.2 Contact Uniqueness Violation

In linear layouts, all transistors occupy a single diffusion row, and adjacent pairs share source/drain regions. This minimizes area but introduces a constraint: each shared contact connects to exactly one net.

We illustrate using the AND3 PMOS array:

`[src]-[poly_A]-[mid1]-[poly_B]-[mid2]-[poly_C]-[mid3]-[poly_$3]-[drn]`

The netlist specifies:

$$\text{M\$1 (NAND P1):} \quad D = \$3,\ G = A,\ S = \text{VDD} \tag{14}$$

$$\text{M\$2 (NAND P2):} \quad D = \text{VDD},\ G = B,\ S = \$3 \tag{15}$$

$$\text{M\$3 (NAND P3):} \quad D = \text{VDD},\ G = C,\ S = \$3 \tag{16}$$

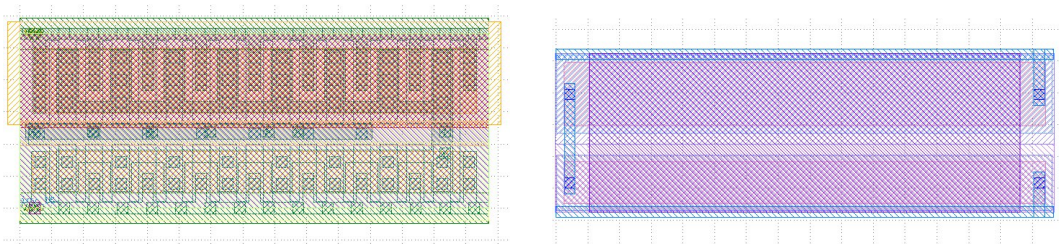
$$\text{M\$4 (INV P):} \quad D = \text{VDD},\ G = \$3,\ S = X \tag{17}$$

In NAND3, the PMOS devices connect in parallel with \$3 as output. In linear layout, `mid3` is shared between M\$3 and M\$4. From M\$3's perspective, `mid3` must connect to \$3. From M\$4's perspective, `mid3` must connect to VDD. A single contact cannot serve both nets—a physical impossibility we term *contact uniqueness violation*.

5.3 Structural Conditions

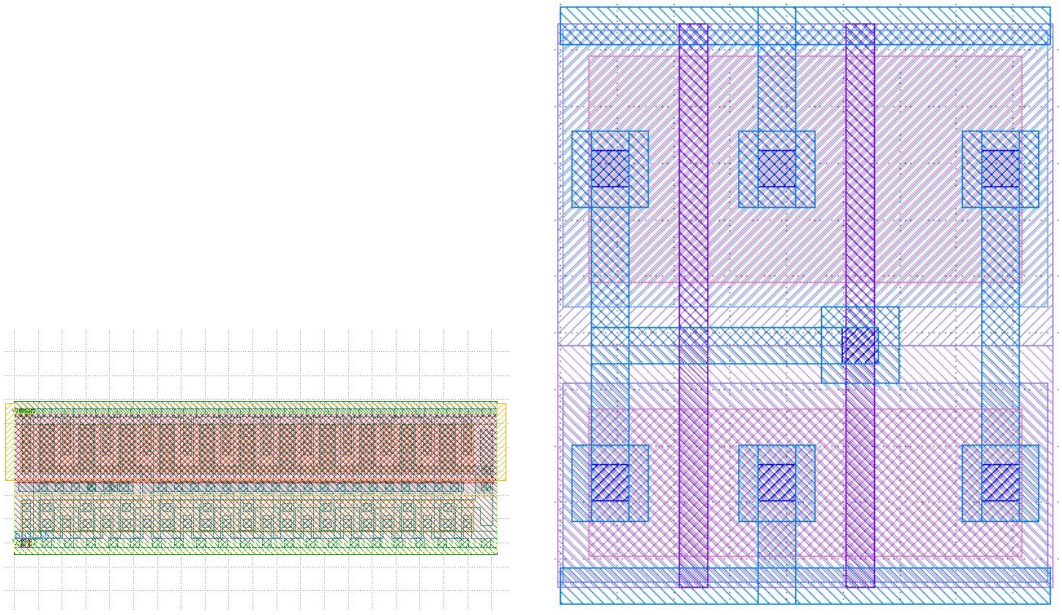
We characterize when violation occurs by contrasting passing and failing types.

INV, NAND2, NOR2, NAND3, NOR3 are single basic gates without cascades; no inter-block internal nodes exist.



(a) Source: SKY130 (b) Migrated: FreePDK45

Figure 2: INV_16 layout comparison. The migrated layout passes DRC, LVS, and VF2 verification.



(a) Source: SKY130 (b) Migrated: FreePDK45

Figure 3: BUF layout comparison. Topology equivalence confirmed by VF2.

BUF cascades two inverters, each with one PMOS/NMOS. The single shared region connects to exactly one net (first inverter output).

AND2 (NAND2+INV) has two PMOS in NAND2. The cascade boundary contact connects only to the NAND2 output; no conflict arises.

Violation occurs when: (i) the basic gate contains **three or more parallel transistors**, and (ii) the internal output net and the cascade inverter's supply connection compete for the same shared contact.

6 Discussion

6.1 Complementary Roles of VF2 and LVS

VF2-based verification complements rather than replaces LVS by addressing cross-PDK netlist-to-netlist comparison.

LVS verifies: does this FreePDK45 layout match its schematic? VF2 verifies: does this FreePDK45 layout's topology match the source SKY130 topology?

Neither subsumes the other. A layout could pass LVS but fail VF2 if the reference schematic was incorrectly generated. Conversely, LVS could fail due to parasitic insertion while VF2 passes, since VF2 compares only active device topology.

Table 7: VF2 device mapping for INV_1.

Source (SKY130)	Migrated (FreePDK45)	Type
M$1 (D=VPWR, G=A, S=Y)	M$1 (D=VDD, G=A, S=Y)	PMOS
M$2 (D=VGND, G=A, S=Y)	M$2 (D=GND, G=A, S=Y)	NMOS

The explicit bijection from VF2 provides diagnostic value beyond binary PASS/FAIL. LVS mismatch reports assume single-PDK context; VF2 mappings identify cross-PDK device correspondences directly.

6.2 Template-Based Generation Limitations

Template-based layout generation has two limitations. First, new cell types require manual template definition, non-trivial for complex cells (flip-flops, AOI/OAI gates). Second, drive-strength variants may require structural differences difficult to generalize within a single template.

Future work could explore constraint-based placement or topology-preserving scaling that maintains relative placement while adjusting coordinates for target PDK rules.

6.3 S/D Symmetry Scope

The extended isomorphism applies only to symmetric MOSFET models. Rigorous validation would require SPICE simulation comparing S/D-swapped configurations across representative inputs. This was not performed in the present work.

The framework accommodates asymmetric devices by restricting ϕ_E to exact matching, reducing to standard isomorphism without structural changes.

6.4 Implications

The VF2-based methodology applies to migration results from any source, including commercial EDA tools or optimization-based generators, provided device-type mapping and netlist normalization are available. The contact uniqueness violation analysis provides criteria for selecting appropriate layout strategies (linear, non-linear, diffusion-broken) based on cell topology.

7 Conclusion

We presented a framework for automated cross-PDK standard cell migration with topology equivalence verification based on VF2 graph isomorphism. Circuit netlists are modeled as device-net bipartite graphs with extended isomorphism accommodating MOSFET S/D symmetry under stated applicability conditions. Unlike LVS, VF2 enables PDK-independent comparison under device-type normalization and provides explicit device-level mappings.

Evaluation on 42 SKY130-to-FreePDK45 migrations demonstrates 76.2% success, with failures concentrated in three compound gate types. We identify contact uniqueness violation as the root cause and characterize the topological conditions under which it arises.

Limitations: The benchmark is restricted to 42 combinational cells; layout generation is template-based; S/D symmetry was not validated via SPICE simulation.

Future work: (i) Implement diffusion break to resolve contact uniqueness violations; (ii) extend benchmark to flip-flops and AOI/OAI gates; (iii) validate S/D symmetry via SPICE; (iv) evaluate parasitic extraction and timing characteristics of migrated layouts.

References

L. P. Cordella, P. Foggia, C. Sansone, and M. Vento. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(10):1367–1372, 2004.

FreePDK45 Process Design Kit. NC State University. https://www.eda.ncsu.edu/wiki/FreePDK45:Contents

SkyWater SKY130 Open Source PDK. Google and SkyWater Technology, 2020. https://github.com/google/skywater-pdk

F.-L. Heng, Z. Chen, and G. E. Tellez. A VLSI artwork legalization technique based on a new criterion of minimum layout perturbation. In *Proc. IEEE/ACM Int. Symp. Physical Design*, pp. 116–121, 1997.

A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proc. 7th Python in Science Conf.*, pp. 11–15, 2008.

L. H. Heitzmann. gdstk: GDSII/OASIS library for IC layout. https://github.com/heitzmann/gdstk

M. Köfferlein. KLayout: High-performance layout viewer and editor. https://www.klayout.de