# CHE-MING, CHANG

cheming.chang1@gmail.com ◇ GitHub ◇ LinkedIn ◇ Taipei, Taiwan

## EDUCATION

**Ph.D. in Electrical and Computer Engineering, University of Southern California** Aug., 2025 - Present
Research Topic: AI/ML-driven EDA for heterogeneous integration

**B.S. in Electrical Engineering, National Taiwan University** Sept., 2020 - Jun., 2024
Overall GPA: **4.17 / 4.30**
Dean's List Award 2023 Spring
TSMC Semiconductor Advanced IC Design Program

**Highlighted Coursework**: Logic Synthesis and Verification, Physical Design for Nanometer ICs, Integrated Circuit Design, Computer-aided VLSI System Design, Digital Signal Processing in VLSI, Machine Learning

## PUBLICATION

[1] Shao-Yu Lo, **Che-Ming Chang**, Yao-Wen Chang, "Advanced Packaging Warpage Modeling with DeepONet-Based Operator Learning," in *Proc. of ICCAD*, Munich, Germany, October 2025.

[2] **Che-Ming Chang**, Jie-Hong Roland Jiang, Dah-Wei Chiou, Ting Hsu and Guin-Dar Lin, "Quantum Circuit Compilation for Trapped-Ion Processors with the Drive-Through Architecture," in *IEEE Trans. Quantum Eng.*, vol. 6, pp. 1-14, 2025.

[3] Cheng-Yu Chiang, Yi-Hsien Chiang, Chao-Chi Lan, Yang Hsu, **Che-Ming Chang**, Shao-Chi Huang, Sheng-Hua Wang, Yao-Wen Chang, and Hung-Ming Chen, "Mixed-Size Placement Prototyping Based on Reinforcement Learning with Semi-Concurrent Optimization," in *Proc. of ASPDAC*, Tokyo, Japan, January 2025, pp. 893–899.

[4] **Che-Ming Chang**, Prashanth Vijayaraghavan, Charles Mackin, Ashutosh Jadhav, Hsinyu Tsai, Vandana Mukherjee and Ehsan Degan, "Enhancing LLMs for HDL Code Optimization using Domain Knowledge Injection," *Under Review*, 2025.

## RESEARCH EXPERIENCE

**Southern California Computer-Aided Design (SCCAD) Lab** Aug., 2025 - Present
*Graduate Research Assistant, Advised by Prof. Sung-Kyu Lim* *Los Angeles, CA*

LLM-aided Incremental Timing Driven Routing
- Researched in sequence-to-sequence modeling for combinatorial physical design tasks.

**Electronic Design Automation Lab** Aug., 2023 - Jan., 2025
*Research Assistant, Advised by Prof. Yao-Wen Chang* *Taipei, Taiwan*

Mixed-Size Placement Prototyping Based on Reinforcement Learning with Semi-Concurrent Optimization [3]
- Proposed an innovative method to move all macros concurrently and iteratively by a deep Q-network model as an agent in reinforcement learning (RL) in contrast to previous works that place them sequentially.
- Demonstrated 12.3% / 9% additional HPWL reduction on ISPD'05 benchmarks compared to DREAMplace / modern ML-based placer GraphPlanner.

Warpage Modeling for Advanced Packaging with DeepONet-Based Operator Learning [1]
- Developed a convolutional neural network (CNN)-based deep operator network (DeepONet) to predict warpage in advanced packaging designs.
- Achieved 435× speedup over an 8-multiprocess finite element method (FEM) while maintaining an average warpage error of 1.9%.

**IBM Almaden Research Center, AI for EDA group** Jun., 2024 - Sept., 2024
*Research Intern, Advised by Prashanth Vijayaraghavan* *San Jose, CA, USA*

Enhancing LLMs for HDL Code Optimization using Domain Knowledge Injection

- Proposed RTLOpt, the first comprehensive dataset designed to evaluate RTL optimization techniques, including pipelining and clock gating.
- Introduced MASCOT, a framework for HDL code optimization by injecting and retrieving domain-specific knowledge into the code generation process, aided with compiler and simulator feedback.
- Achieved 20% / 10% improvements for larger LLMs /small fine-tuned LLMs compared to zero-shot and ReAct / Chain-of-Thought baseline on HDL optimization tasks.

**Applied Logic and Computation in System Design Lab** Sept., 2022 - June, 2024
*Research Assistant, Advised by Prof. Jie-Hong Roland Jiang* *Taipei, Taiwan*

Quantum Circuit Compilation for Trapped-Ion Processors with Drive-Through Architecture [2]

- Proposed an analytical method to compile a QASM circuit onto a "drive-through" modular architecture by gate partitioning followed by a fidelity-aware placement.
- Achieved 10.8% / 9.6% higher fidelity compared to modern qubit mappers SABRE / t|ket⟩, while ensuring the practicality and scalability of our approach.

## PROJECT EXPERIENCE

**Approximate Logic Synthesis with Multi-Objective Simulated Annealing** Oct., 2023 - Dec., 2023
*Final Project in Logic Synthesis and Verification, GitHub* *Taipei, Taiwan*

- Utilized multi-objective simulated annealing to synthesize approximate circuits within the given error bound based on the synthesis tool ABC.
- Demonstrated substantial area reduction (8-98%) over IWLS benchmarks within 1% error rate and shows quality (5-91%) and runtime improvements over the existing ALS tool ALSRAC.

**QR decomposition engine for 5G MIMO demodulation** Oct., 2023 - Dec., 2023
*Final Project in Computer-aided VLSI System Design* *Taipei, Taiwan*

- Implemented QR decomposition engine with 2.2% soft LLR error rate in Verilog.
- Achieved 206 MHz operating frequency with 37.5 mW power under TSMC 130nm technology with full VLSI design flow (RTL to GDS).

**Real-Time FPGA-based Acoustic Imaging** Nov., 2022 - Jan., 2023
*Final Project in Laboratory of Digital Circuit System, GitHub* *Taipei, Taiwan*

- Implemented a low-complexity delay-and-sum algorithm on Altera DE2-115 FPGA using SystemVerilog to visualize the position of sound sources in real-time via VGA output.
- Demonstrated significant reduction in implementation cost compared to currently high manufacturing costs.

## AWARDS AND HONORS

**Taiwan Chip-based Industrial Innovation Program Research Scholarship**, NSTC Sep, 2024 - Jan. 2025
**Scholarship for Overseas Internship Program**, NTU / Ministry of Education June, 2024 - Aug. 2024
**Research Scholarship**, MediaTek-NTU Research Center Mar., 2024- June, 2024
**Second Prize and Global Nominee**, NASA Space App Challange Oct., 2023
**College Student Research Scholarship**, National Science and Technology Council Aug., 2023 - Feb., 2024

## SKILLS

| | |
|---|---|
| **Languages** | Mandarin (Native), English (Proficient) |
| **Programming Languages** | C/C++, Python, Verilog, SystemVerilog, MATLAB, Javascript, LaTeX |
| **Experience with EDA Tools** | Synopsys VCS / DC / PT, Cadence Innovus, Berkeley ABC, Yosys |

# Layout-Level LLMs for Scalable Physical Design Optimization

Che-Ming Chang
*University of Southern California*
Los Angeles, CA
chemingc@usc.edu

***Abstract*—As technology nodes advance and design scales grow, traditional EDA tools face challenges to deliver high-quality, scalable physical design (PD) optimizations. We propose a layout-level encoder–decoder foundation model that captures both geometric and physical information from placed and globally routed netlists. By defining cell- and net-level tokens and learning embeddings through self-supervised learning, the model enables sequence-to-sequence reasoning directly in the PD domain. In contrast to conventional staged PD flows, our framework can perform cross-stage reasoning and generative optimization across multiple PD tasks. This unified representation allows the model to adapt to downstream applications such as simultaneous gate sizing, buffering and cell relocation, as well as incremental timing-driven global routing—capabilities not attainable under traditional PD flows. We envision this framework to offer a generalizable, scalable paradigm for machine learning-assisted PD optimization.**

## I. Background and Motivation

With increasing complexity of advanced technology nodes and larger design scales, traditional electronic design automation (EDA) tools, while robust, struggles to deliver high-quality optimization at scale. To address these limitations, machine learning (ML)-driven approaches have emerged to accelerate the design closure, with a trend evolving from prediction, optimization and more recently to generation, where model directly propose candidate solutions.

Recently, foundation models such as large language models (LLMs) have gained attention in EDA. Trained on large-scale datasets using self-supervised learning, LLMs are capable of performing a broad range of downstream tasks, providing a new paradigm for optimization at scale. Depending on the underlying architecture, prior works have demonstrated different capabilities of such models.

Encoder-based approaches play a critical role in extracting meaningful representations from the structural and functional properties of circuits [1]. Previous works employed graph neural networks (GNNs) for circuit representation learning problems. These encoders have been successfully applied to tasks such as logic equivalence checking (LEC) and quality-of-result (QoR) prediction [2], [3], [4]. While effective, these works are not suitable for generative tasks, and are confined to synthesis-level abstractions such as and-inverter graphs (AIGs) and gate-level netlist.

Decoder-based approaches, in contrast, leverage text-level reasoning for generative tasks. By employing techniques such as prompt engineering, instruction-code fine-tuning, and retrieval-augmented generation (RAG) [5], these methods have demonstrated success in hardware description language (HDL) generation [6], [7], [8], tool command language (TCL) script generation [9], and design documentation Q&A [10]. However, their applications remain focused on text-based tasks.

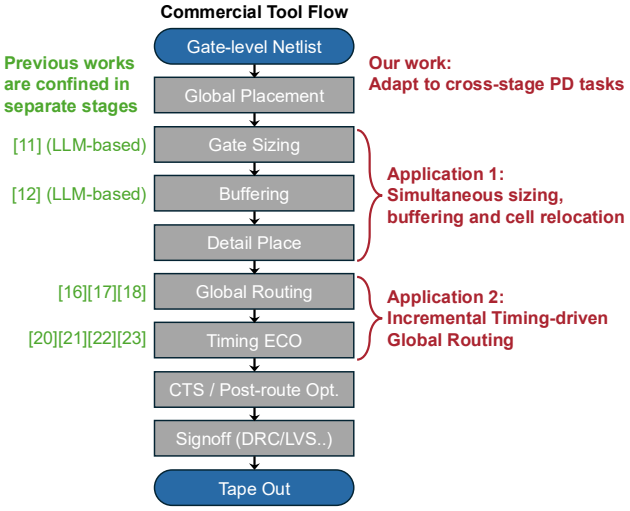


Fig. 1. Staged optimization vs. our layout-foundation model that generalizes to tasks beyond traditional PD stages.

Whether encoder- or decoder-based, the application of LLMs in the **physical design** (PD) stage remains far from trivial and is still less explored. PD problems must satisfy complicated design rule constraints and are inherently combinatorial in nature, which cannot be reasoned through text-based tasks. Only a few recent works have begun to explore layout-level tasks through customized tokenization and sequence-to-sequence learning. For example, LEGO-size [11] casts cells on the timing path as a series of tokens for gate sizing optimization, while BUFFALO [12] serializes buffer tree topology on a single net into a bracketed depth-first-search (DFS) sequence for effective buffer insertion. These pioneering efforts demonstrate the feasibility of layout-level sequence modeling but remain limited to isolated subproblems.

We aim to bridge the existing gap by developing an encoder–decoder based, layout-level foundation model that captures information from placed and globally routed netlists with tokenization at the granularity of routing. As illustrated in Fig. 1, we envision this foundational model can serve as *layout-to-layout generative models*, capable of not only producing candidate solutions but also guiding optimization across PD stages. By defining cell-level and net-level tokens, our framework enables unified layout reasoning that could be applied across traditional PD stage boundaries,

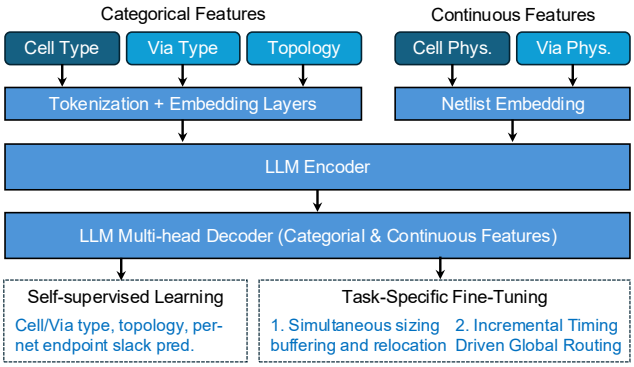


Fig. 2. Overview of the proposed layout-level encoder-decoder foundation model. The framework first performs joint cell-net tokenization for categorical information, and extract design-specific features for netlist-level information. The encoder is trained through self-supervised learning with different prediction tasks. The model is then fine-tuned for downstream PD tasks.

which enlarges the solution space and creates opportunities for further optimizations unattainable under previous staged flows [13].

Furthermore, when coupled with external optimization engines—such as differentiable static timing analysis (STA) [14] or reinforcement learning (RL) reward feedback—these foundation models can move beyond prediction to deliver "better-than-tool" optimization [11], [12]. In this paradigm, LLMs narrow the high-dimensional PD solution space to promising near-optimal regions, where traditional differentiable methods can converge more efficiently. We will demonstrate the capabilities of this model through two downstream applications:

1) **Simultaneous gate sizing, buffering, and incremental cell relocation.** Traditionally, these physical design optimization primitives are handled in separate stages after placement, which is inherently suboptimal. The 2025 CAD Contest [15] and 2026 ISPD Contest highlights the need for integrated framework across these optimization.
2) **Incremental timing-driven global routing.** Classical global routing (GR) methods [16], [17], [18] generate layer assignments and Steiner tree topologies for each net, primarily relying on FLUTE-based Steiner tree construction [19]. These approaches optimize wirelength but lack timing awareness. Timing-driven variants—such as shallow-light trees [20], Elmore delay-minimized Steiner trees [21], and layer assignment techniques [22], [23]—reduce delay but remain limited to subproblems.

By casting these problems into a sequence-to-sequence formulation, our encoder–decoder LLM can generate optimized layout results in a single pass, which is beyond the capability of traditional algorithms.

## II. Research Objectives and Methods

### A. Overview

The goal of this work is to learn a *sequence representation* that captures both geometric and connectivity information of a placed and globally routed *cone* — a sub-circuit with a single primary output. Each cone consists of a set of cells and nets,

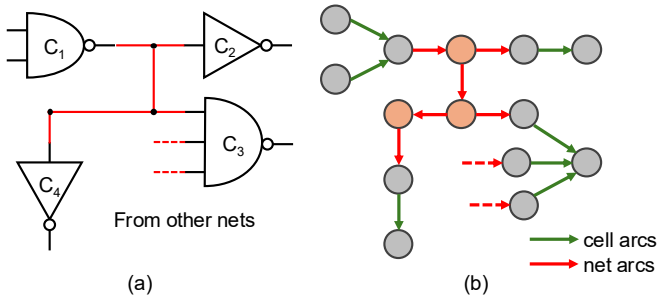


Fig. 3. Example of transforming a placed and routed netlist into a directed acyclic graph (DAG). (a) Example routed netlist, red edges denotes the Steiner tree interconnection. (b) Transformed DAG, where gray nodes denotes cell pin, and orange node denotes the Steiner nodes.

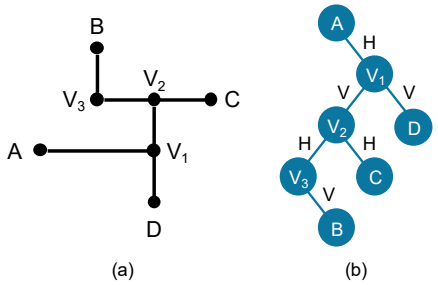


Fig. 4. Example of transforming a Steiner tree into a R-tree. (a) Example Steiner tree. (b) Corresponding R-tree, which can be serialized canonically.

along with corresponding geometric (location, layer), physical (capacitance, gate size, timing) attributes. Unlike previous works that focus on single timing paths [11] or individual nets [12], the cone-level formulation provides a more holistic view of the circuit. It allows the model to reason about timing-critical endpoints where global optimization requires coordinated updates across multiple nets and cells.

Given this representation, the model takes an input sequence describing the current layout and generates an updated sequence representing a globally optimized layout. As illustrated in Figure 2, our framework encodes both categorical and continuous features into a unified embedding space. This hybrid encoding and decoding scheme enables the model to jointly capture the circuit's structural relationships and its detailed geometric and physical attributes. Through self-supervised learning, the model learns transferable layout representations that generalize across designs, which are subsequently fine-tuned for specific downstream PD tasks.

### B. Methodology

*a) Token Representation:* The placed and globally routed netlist can be naturally modeled as a directed acyclic graph (DAG), as shown in Figure 3. Within this structure, we define two types of arcs: *cell-arcs*, representing logic connectivity between input and output pins of cells, and *net-arcs*, representing edges in the rectilinear Steiner tree topology of each routed net, considering directions from its driver to different sinks. Nodes in this graph correspond either to cell pins or to Steiner nodes (vias) along the routing paths.

To enable sequence modeling, we first focus on serializing a single routed net into a canonical sequence. We propose a new data structure, the *R-tree*, which transforms the rectilinear Steiner tree into a canonical form to enable traversal and sequence generation, as illustrated in Figure 4. This representation preserves both geometric adjacency and the directional structure of routing segments, and exhibits elegant properties under rotation or mirroring of the input Steiner tree. For instance, a routing tree shown in Figure 4 can be expressed as $A()(HV_1(VV_2(HV_3()(VB))(HC))(VD))$.

Moreover, backward traversal of the DAG is crucial for incorporating timing-aware information into the representation. For a single routed net $V = \{n_0, n_1, ...n_k\}$ where $n_0$ is the driver with position $p_0$ resistance $r_0$, each sink $n_i$ has position $p_i$, capacitance $c_i$ and negative slack $s_{n_i}$, we can model the Elmore delay of $n_0$ to $n_i$ in a routing tree $T$ as [21]:

$$d_T(n_i) = r_0 C_{n_0} + \sum_{e_v \in \text{path}(n_0, n_i)} r_{e_v} \left( \frac{1}{2} c_{e_v} + C_v \right).$$

where $e_v$ is a segment on the path with endpoint $v$, edge resistance $r_{e_v}$ and edge capacitance $c_{e_v}$ are proportional to edge length $w(e_v)$, multiplied by unit resistance/capacitance of the selected layer. $C_n$ is the total capacitance of subtree rooted at node $n$; i.e., sum of all downstream edge capacitance and sink capacitance, which can be efficiently acquired via backward traversal of the DAG.

We will extend this canonical serialization from single nets to multiple nets. Variants of traversal schemes will be explored for multi-net ordering, to ensure canonical and context-preserving sequence representation of an entire cone.

*b) Netlist-level Embeddings:* While token-level embeddings capture local features of cells and nets, global netlist-level representations are essential for holistic PD optimization. To address this, we extract global embeddings by sampling important subgraphs within each cone, aggregating them through transformer-based encoders capable of modeling long-range dependencies. Unlike traditional graph neural networks (GNNs), which are constrained by fixed $k$-hop message passing, the transformer encoder can utilize self-attention mechanism to capture long-range interactions, thereby learning embeddings that can encode timing-related global information.

*c) Encoder and Decoder Training:* The encoder is designed with shared representation spaces for cell, net, and global embeddings to maintain consistency across different levels of abstraction. Such shared representations ensure that modifications in one domain (e.g., buffering a cell) propagate meaningfully to correlated structures (e.g., the affected nets). The decoder then used the encoded embeddings for various downstream tasks.

During self-supervised learning, we will perform masked token prediction on categorical features (cell/via type, topology) and regression on continuous features (location, delay) to ensure the learned embeddings are physically meaningful, which could be verified via dimensionality reduction techniques such as t-SNE [24]. Additional structural constraints are imposed during decoding to guarantee legal layout sequences — e.g., enforcing connectivity of routing segments, or direction in specific metal layers. Inspired by the high-level concept of *plausibility pruning* in [25], we will incorporate geometric validity checks to discard infeasible routing patterns early in decoding, thereby ensuring legality of generated layouts.

For supervised fine-tuning, the model is trained on pairs of unoptimized–optimized netlists derived from industrial place-and-route (P&R) tools. Power, performance, and area (PPA) feedback are integrated at both global (TNS, WNS) and fine-grained (per-net slack, per-pin capacitance) levels. The decoder output tokens are defined according to the specific downstream task. For instance, if we allow variable cell tokens during output, decoder is fine-tuned for incremental placement optimization with simultaneous gate sizing, buffering and cell relocation; but if we fixed cell tokens to be exactly same as input, then we are focus on incremental timing-driven routing.

The above training will leverage our available GPU resources (8 A100s) to handle the large-scale layout data efficiently. Input data can be obtained by performing early global routing using commercial tools (e.g., Innovus) and exporting the results in design exchange format (.def) files, which can then be parsed to extract the required categorical and continuous features for model inputs.

Finally, the framework interfaces with external tools such as differentiable STA engines [14] or reinforcement learning such as simple preference optimization (SimPO) [26]. Compared with direct preference optimization (DPO) [27], SimPO normalizes rewards by sequence length, making it more suitable for PD tasks where the generated topologies can vary significantly in complexity, ensuring more stable and fair reward signals across different designs.

## III. Expected Impact

The proposed layout-level foundation model represents a significant advancement in ML-driven EDA, with the following expected impacts:

1) **Generalizable foundation for future PD tasks**: The encoder–decoder model, trained with self-supervised learning, produces embeddings transferable across circuits and different PD tasks. This positions the framework as a general-purpose PD foundation model, enabling rapid adaptation to new design rules, technologies, or different downstream tasks.
2) **Scaling combinatorial optimization tasks through LLM**: By representing cells and nets as serialized token sequences, the model could utilize transformer-based attention mechanisms to capture long-range interactions that exceeds traditional GNN limitations. Unlike prior works that focus either on predictive tasks (timing/QoR estimation) or on narrow generative subproblems (e.g., gate sizing, buffer insertion), our model integrates both local and global layout reasoning. We expect LLM to propose holistic, layout-aware optimization solutions that are currently unattainable through staged flows. When coupled with assistance of external tools, we expect to produce better-than-tool layout results.

## REFERENCES


[1] W. Fang, J. Wang, Y. Lu, S. Liu, Y. Wu, Y. Ma, and Z. Xie, "A survey of circuit foundation model: Foundation AI models for VLSI circuit design and EDA," *arXiv preprint arXiv:2504.03711*, 2025.

[2] M. Li, S. Khan, Z. Shi, N. Wang, H. Yu, and Q. Xu, "DeepGate: learning neural representations of logic gates," in *Proc. of DAC*, p. 667–672, 2022.

[3] Z. Shi, H. Pan, S. Khan, M. Li, Y. Liu, J. Huang, H.-L. Zhen, M. Yuan, Z. Chu, and Q. Xu, "DeepGate2: Functionality-aware circuit representation learning," in *Proc. of ICCAD*, pp. 1–9, 2023.

[4] Z. Wang, C. Bai, Z. He, G. Zhang, Q. Xu, T.-Y. Ho, B. Yu, and Y. Huang, "Functionality matters in netlist representation learning," in *Proc. of DAC*, p. 61–66, 2022.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. of NeurIPS*, 2020.

[6] S. Thakur, B. Ahmad, H. Pearce, B. Tan, B. Dolan-Gavitt, R. Karri, and S. Garg, "Verigen: A large language model for verilog code generation," *ACM TODAES*, vol. 29, no. 3, 2024.

[7] S. Liu, W. Fang, Y. Lu, J. Wang, Q. Zhang, H. Zhang, and Z. Xie, "RTLCoder: Fully open-source and efficient LLM-Assisted RTL code generation technique," *IEEE TCAD*, vol. 44, no. 4, pp. 1448–1461, 2025.

[8] Y. Fu, Y. Zhang, Z. Yu, S. Li, Z. Ye, C. Li, C. Wan, and Y. C. Lin, "GPT4AIGChip: Towards next-generation AI accelerator design automation via large language models," in *Proc. of ICCAD*, pp. 1–9, 2023.

[9] H. Wu, Z. He, X. Zhang, X. Yao, S. Zheng, H. Zheng, and B. Yu, "ChatEDA: A large language model powered autonomous agent for EDA," *IEEE TCAD*, vol. 43, no. 10, p. 3184–3197, 2024.

[10] U. Sharma, B.-Y. Wu, S. R. D. Kankipati, V. A. Chhabria, and A. Rovinski, "OpenROAD-assistant: An open-source large language model for physical design tasks," in *Proc. of MLCAD*, pp. 1–7, 2024.

[11] Y.-C. Lu, K. Kunal, G. Pradipta, R. Liang, R. Gandikota, and H. Ren, "LEGO-Size: LLM-Enhanced GPU-Optimized Signoff-Accurate Differentiable VLSI Gate Sizing in Advanced Nodes," in *Proc. of ISPD*, p. 152–162, 2025.

[12] H.-H. Hsiao, Y.-C. Lu, S.-K. Lim, and H. Ren, "BUFFALO: PPA-Configurable, LLM-based Buffer Tree Generation via Group Relative Policy Optimization," in *Proc. of ICCAD*, pp. 1–9, 2025.

[13] Y. Du, Z. Guo, Y. Lin, R. Wang, and R. Huang, "Fusion of global placement and gate sizing with differentiable optimization," in *Proc. of ICCAD*, 2024.

[14] Y.-C. Lu, Z. Guo, K. Kunal, R. Liang, and H. Ren, "INSTA: An ultra-fast, differentiable, statistical static timing analysis engine for industrial physical design applications," in *Proc. of DAC*, pp. 1–7, 2025.

[15] Y.-C. Lu, R. Liang, W.-H. Liu, and H. Ren, "Invited paper: 2025 ICCAD CAD Contest Problem C: Incremental placement optimization beyond detailed placement: Simultaneous gate sizing, buffering, and cell relocation," in *Proc. of ICCAD*, 2025.

[16] S. Liu, P. Liao, R. Zhang, Z. Chen, W. Lv, Y. Lin, and B. Yu, "FastGR: Global routing on CPU-GPU with heterogeneous task graph scheduler," in *Proc. of DATE*, pp. 760–765, 2022.

[17] J. Liu and E. F. Y. Young, "EDGE: Efficient DAG-based global routing engine," in *Proc. of DAC*, p. 1–6, 2023.

[18] W. Li, R. Liang, A. Agnesina, H. Yang, C.-T. Ho, A. Rajaram, and H. Ren, "DGR: Differentiable global router," in *Proc. of DAC*, p. 1–6, 2024.

[19] C. Chu and Y.-C. Wong, "FLUTE: Fast lookup table based rectilinear steiner minimal tree algorithm for VLSI design," *IEEE TCAD*, vol. 27, no. 1, pp. 70–83, 2008.

[20] G. Chen and E. F. Y. Young, "SALT: Provably good routing topology by a novel steiner shallow-light tree algorithm," *IEEE TCAD*, vol. 39, no. 6, pp. 1217–1230, 2020.

[21] H. Wu, X. Li, L. Chen, B. Yu, and W. Zhu, "Delay-driven rectilinear steiner tree construction," *IEEE TCAD*, vol. 44, no. 5, pp. 1928–1941, 2025.

[22] D. Liu, B. Yu, S. Chowdhury, and D. Z. Pan, "TILA-S: Timing-driven incremental layer assignment avoiding slew violations," *IEEE TCAD*, vol. 37, no. 1, pp. 231–244, 2017.

[23] J. Ao, S. Dong, S. Chen, and S. Goto, "Delay-driven layer assignment in global routing under multi-tier interconnect structure," in *Proc. of ISPD*, p. 101–107, 2013.

[24] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *JMLR*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[25] I. Huang, Y. Bao, K. Truong, H. Zhou, C. Schmid, L. Guibas, and A. Fathi, "Fireplace: Geometric refinements of LLM common sense reasoning for 3D object placement," in *Proc. of CVPR*, pp. 13466–13476, 2025.

[26] Y. Meng, M. Xia, and D. Chen, "SimPO: simple preference optimization with a reference-free reward," in *Proc. of NeurIPS*, 2024.

[27] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn, "Direct preference optimization: your language model is secretly a reward model," in *Proc. of NeurIPS*, 2023.