

The Dawn of AI-Native EDA: Opportunities and Challenges of Large Circuit Models

Tsung-Yi Ho¹, Sadaf Khan¹, Jinwei Liu¹, Yu Li¹, Yi Liu¹, Zhengyuan Shi¹, Ziyi Wang¹, Qiang Xu^{1§},
Evangeline F.Y. Young¹, Bei Yu¹, Ziyang Zheng¹, Binwu Zhu¹, Keren Zhu¹

¹The Chinese University of Hong Kong

Yiqi Chen², Ru Huang^{2,3}, Yun Liang², Yibo Lin², Guojie Luo^{2§}, Guangyu Sun², Runsheng Wang², Xinming
Wei², Chenhao Xue², Jun Yang³, Haoyi Zhang², Zuodong Zhang², Yuxiang Zhao², Sunan Zou²

²Peking University ³Southeast University

Lei Chen⁴, Yu Huang⁵, Min Li⁴, Dimitrios Tsaras⁴, Mingxuan Yuan^{4§}, Hui-Ling Zhen⁴

⁴Huawei Noah's Ark Lab ⁵Huawei HiSilicon

Zhufei Chu⁶, Wenji Fang⁷, Xingquan Li⁸, Junchi Yan⁹, Zhiyao Xie⁷, Xuan Zeng¹⁰

⁶Ningbo University ⁷Hong Kong University of Science and Technology

⁸Peng Cheng Laboratory ⁹Shanghai Jiao Tong University ¹⁰Fudan University

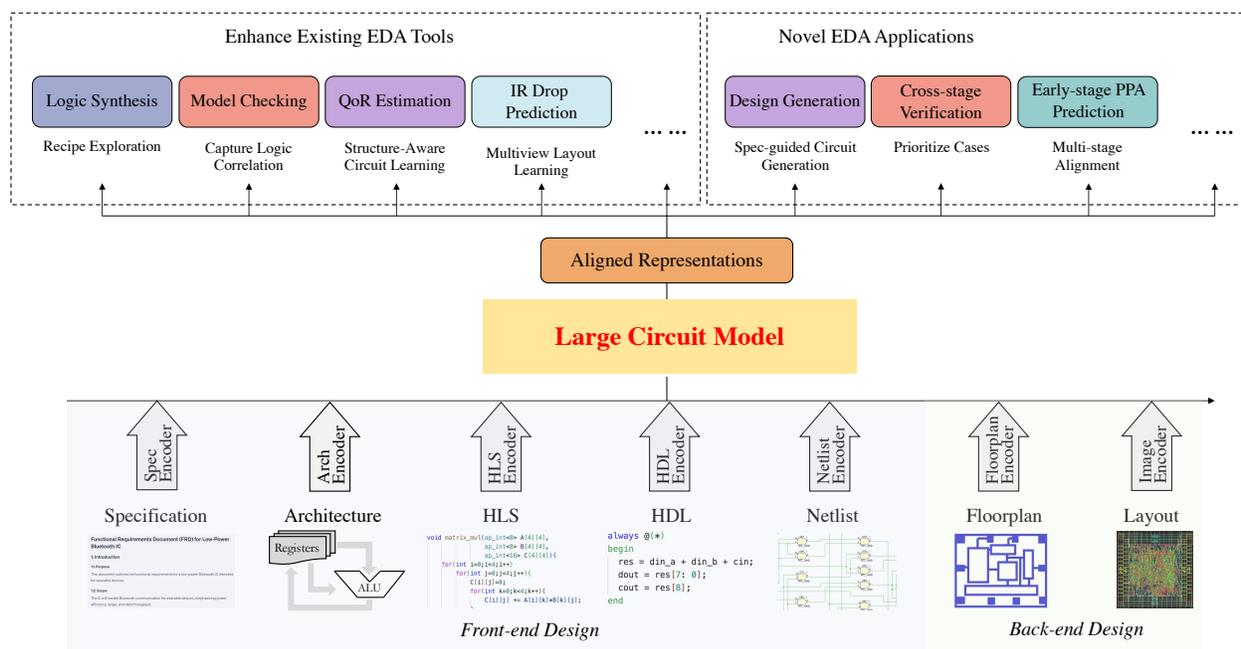


Fig. 1: **Large Circuit Models**: we call upon the creation of dedicated foundation models for circuits, which intricately intertwines computation with structure, unlike other types of data (e.g., texts and images). Specifically, each design stage of the EDA flow is considered a separate modality and requires a specific representation learning strategy to embed the available circuit characteristics. The higher the design level is, the more semantics to represent; The lower the design level is, the more details to represent. Central to the appeal of LCMs is their ability to fuse and align disparate representations throughout the design continuum, creating a unified narrative that spans from high-level functional specifications to detailed physical layouts. This unified approach promises to streamline the EDA process, reduce time-to-market, and improve design PPA.

§ The authors in each institution are ordered alphabetically. **Contact**: qxu@cse.cuhk.edu.hk; gluo@pku.edu.cn; yuan.mingxuan@huawei.com.

Abstract—Within the Electronic Design Automation (EDA) domain, AI-driven solutions have emerged as formidable tools, yet they typically augment rather than redefine existing methodologies. These solutions often repurpose deep learning models from other domains such as vision, text, and graph analytics applying them to circuit design without tailoring to the unique complexities of electronic circuits. Such an “AI4EDA” approach falls short of achieving a holistic design synthesis and understanding, overlooking the intricate interplay of electrical, logical, and physical facets of circuit data. **This paper argues for a paradigm shift from AI4EDA towards AI-native EDA**, integrating AI at the core of the design process. Pivotal to this vision is the development of a multimodal circuit representation learning technique, poised to provide a comprehensive understanding by harmonizing and extracting insights from varied data sources, such as functional specifications, RTL designs, circuit netlists, and physical layouts.

We champion the creation of **large circuit models (LCMs)** that are inherently multimodal, crafted to decode and express the rich semantics and structures of circuit data, thus fostering more resilient, efficient, and inventive design methodologies. Embracing this AI-native philosophy, we foresee a trajectory that transcends the current innovation plateau in EDA, igniting a profound “shift-left” in electronic design methodology. The envisioned advancements herald not just an evolution of existing EDA tools but a revolution, giving rise to novel instruments of design tools that promise to radically enhance design productivity and inaugurate a new epoch where the optimization of circuit performance, power, and area (PPA) is achieved not incrementally, but through leaps that redefine the benchmarks of electronic systems’ capabilities.

Index Terms—AI-native EDA, large circuit models (LCMs), multimodal circuit representation learning, circuit optimization.

1 THE FOUNDATION MODEL PARADIGM

The landscape of artificial intelligence (AI) has been profoundly transformed in recent years by the advent of large foundation models. These models, characterized by their vast scale and general applicability, have demonstrated an uncanny ability to understand, predict, and generate content with a level of sophistication that was previously the exclusive domain of human intelligence.

1.1 The Rise of Foundation Models

Large foundation models represent a significant leap in AI. These models, typically pre-trained on web-scale datasets using self-supervision techniques [1], have been adapted to excel in a wide array of downstream tasks. In the fields of natural language processing (NLP) and computer vision (CV), these models have not only set new benchmarks but have fundamentally redefined the realms of possibility.

In NLP, models like BERT [2] and its derivatives, including RoBERTa [3] and T5 [4], have revolutionized language understanding, especially in contextual interpretation of text, thereby enhancing complex language-based tasks. Concurrently, the decoder-only GPT series [5] has shown remarkable versatility, excelling in diverse tasks from creative writing to code generation and pointing towards the burgeoning potential of artificial general intelligence (AGI). In CV area, self-supervised foundation models [6], [7], [8] have achieved competitive performances in image understanding tasks, rivaling fully supervised approaches.

The recent advent of multimodal foundation models has ushered in a new era of possibilities, integrating diverse data types such as text, images, and audio. A pioneering example is the CLIP model [9], which effectively bridges linguistic and visual data through contrastive learning. This innovation has set the stage for generative models like DALL-E [10] and Stable Diffusion [11], which demonstrate the capability to generate intricate images from textual descriptions, seamlessly blending visual and linguistic understanding. Additionally, the recently introduced promptable CV systems (e.g., SAM [12]) have exhibited exceptional zero-shot generalization in image segmentation, enabling precise object identification and extraction. The emergence of GPT-4V [13] and Gemini [14] further exemplify the evolution of AI, seamlessly navigating and synthesizing multimodal information, thereby opening new avenues for innovation across various fields, from creative content generation to complex problem-solving in engineering and design.

Despite these advancements, the field of circuit design has only begun to scratch the surface of what foundation models can offer. This hesitant engagement contrasts starkly with the transformative potential these models hold for this important field.

1.2 The Unique Challenge of Circuit Data

In the realm of circuit design, a notable phenomenon is the inherent similarity of many new designs to past iterations. Despite these similarities, designers frequently face the challenge of recreating or redesigning circuits from scratch, driven by the subtle yet critical nuances required to meet ambitious performance, power, and area (PPA) objectives. This repetitive process highlights the need for a learning solution that can effectively draw from historical successes and failures.

The emergence of AI for electronic design automation (AI4EDA) solutions [15] marks an attempt to integrate machine learning (ML) techniques into circuit design. These advancements represent significant progress but often only augment, rather than redefine, existing methodologies. Typically, AI4EDA repurposes deep learning models from other domains for EDA tasks such as PPA estimation and optimization, verification, or fault detection. However, within the confines of traditional design frameworks, these models act more as individual analytical tools than as integral components of the design process, often failing to fully address the unique complexities of circuit data.

Specifically, the distinctive nature of circuit data poses unique challenges for machine learning. Unlike text, images, or regular graph data, **circuit design intricately intertwines computation with structure**. Minor structural changes can lead to significant functional impacts, and vice versa. This interdependency renders the task of modeling circuits highly nuanced and complex. Without considering the above, existing AI4EDA solutions frequently fall short in achieving a comprehensive synthesis and understanding of the multifaceted interplay between electrical, logical, and physical aspects of circuit data, which is essential for truly innovative design synthesis.

Recent advancements in AI-native circuit representation learning, such as those presented in [16], [17], have begun to address these unique challenges. The integration of multimodal learning presents a significant opportunity to further enhance their effectiveness. By adopting the principles and capabilities demonstrated by existing foundation models on various types of data, we conceptualize a **paradigm shift from AI4EDA to AI-native EDA**.

Pivotal to this vision is the development of sophisticated **large circuit models (LCMs)**. Envisioned as models adept at integrating and interpreting diverse data types specific to circuit design, LCMs could potentially revolutionize the design, optimization, and verification processes of electronic circuits.

1.3 The Feasibility and Promises of AI-Native LCMs

In the world of semiconductor design, the potential for leveraging large circuit models is not just aspirational; it is rooted in a rich heritage of technological evolution.

Decades of research and development have yielded a vast repository of circuit data. Though proprietary barriers exist, there is enough in the public domain [18], [19], [20] to fuel the development of robust, intelligent models. The industry’s long history provides data that is richly annotated with domain expertise, offering deep insights into the intricacies of circuit design.

Moreover, the landscape of circuit types, though vast, is marked by commonalities that transcend individual designs. Processors, domain accelerators (e.g. digital signal processors (DSPs) and AI accelerators), communication modules, and other core components display a pattern of design module reuse. Examples of these reusable modules include arithmetic units, various decoders, and cryptographic cores. This consistency provides a predictable pattern—akin to an inductive bias—that is conducive to the application of machine learning models.

Advances in neural network architectures, particularly Transformers [21] and graph neural networks (GNNs) [22], are well-suited to capturing the complex, graph-like structure of circuit schematics. They present an opportunity to transform the intricate web of design elements into actionable insights, a feat previously unattainable. The AI advancements from other domains, e.g., CLIP model with multimodal machine learning capabilities [23] and large language models for code generation [24], further underscore the potential for transformative applications in LCMs. These capabilities could be adapted to address the unique challenges in circuit designs of various forms, enabling more nuanced and comprehensive modeling than ever before.

In summary, while the challenges are nontrivial, the development of LCMs is poised on a solid foundation of historical data, pattern prevalence, and cutting-edge computational techniques. The potential for LCMs to revolutionize the field of EDA is not just a theoretical possibility but a tangible goal, driven by the convergence of historical knowledge and modern AI advancements. By processing and interpreting a diverse array of data sources and formats, including schematic diagrams, textual specifications, register-transfer level (RTL) designs, circuit netlists, physical layouts, and performance metrics, LCMs can facilitate a ‘shift-left’ in the design methodology. This proactive AI-native approach enables the early identification of potential performance issues and design bottlenecks, streamlining the testing and redesign processes, and leading to more informed and efficient development cycles.

1.4 Overview of This Perspective Paper

This paper embarks on a comprehensive exploration into the dawn of AI-native EDA, focusing on the development and application of large circuit models that inherently incorporate multimodal data. Spanning nine sections, the paper delves into the historical evolution of EDA, the current state of AI in this field, and the promising future shaped by LCMs.

Section 2 provides a historical overview of EDA, tracing its evolution alongside the semiconductor industry. It emphasizes how the field has navigated challenges of complexity through abstractions, setting a foundation for understanding the significance of LCMs in this evolving landscape. Next, we discuss the current integration of AI in EDA in Section 3, highlighting how deep learning has been utilized to improve EDA processes.

In Section 4, we introduce AI-native LCMs, illustrating their departure from traditional AI4EDA approaches. It delves into how these models encapsulate the intricacies of circuit design, offering a more comprehensive approach to circuit analysis and even creation. Focusing on the development of unimodal circuit representation learning, Section 5 discusses its critical role in building the foundation for multimodal LCMs. It explores the nuances of this approach in achieving a thorough understanding of circuit data. Then, Section 6 navigates the transition to multimodal integration in LCMs. It discusses the development of techniques to align and integrate representations from different design stages, emphasizing the importance of preserving the original design intent.

Section 7 illustrates the potential applications of LCMs through case studies and envisioned scenarios, bridging the gap between theoretical concepts and practical implementations. In Section 8, we explore the application of LCMs in specialized circuit domains, discussing how these models can be adapted to cater to the unique needs of diverse circuit types other than standard digital circuits, including standard cell designs, datapath units, and analog circuits.

Next, we discuss the challenges and opportunities presented by the adoption of LCMs in EDA in Section 9. It highlights issues such as data scarcity and scalability, as well as the potential advancements these challenges can foster. Finally, the paper concludes with a summary of the key insights and a forward-looking perspective in Section 10. It calls for continued collaboration between the AI and EDA communities and suggests future research avenues to further advance the field.

2 HISTORICAL ODYSSEY OF EDA

As we stand on the precipice of this new frontier of AI-native EDA, it is vital to appreciate the historical EDA journey. Understanding the evolution of cutting-edge EDA tools, methodologies, and philosophies will provide invaluable context for the challenges and opportunities that lie ahead.

2.1 Core Objectives and Complexities in EDA

The odyssey of EDA is a chronicle of human ingenuity and technological advancement. It is a story that mirrors the exponential growth of the semiconductor industry, fueled by Moore’s Law, and characterized by the ceaseless push for smaller, faster, and more efficient electronic devices. The journey from simple logic circuits to today’s billion-transistor integrated circuits (ICs) has necessitated a layered hierarchical design methodology with the help of sophisticated EDA toolsets. This hierarchy, marked by stages such as specification, architecture design, high-level algorithm design, RTL design, logic synthesis, and physical design, allows for incremental refinement of the circuit design, each stage adding a layer of detail, ensuring functionality while striving for optimization.

The journey of EDA is not just marked by the sophistication of its tools but also by the fundamental goals that drive its evolution. Two core objectives have consistently shaped the development of EDA solutions:

- **Equivalence and Consistency across Transformations:** Ensuring that each transformation—from behavioral descriptions to gate-level implementation and from logical to physical representation—maintains the original design intent is essential. C-RTL equivalence checking, assertion-based verification (ABV), logic equivalence checking (LEC), sequential equivalence checking (SEC), and various types of simulation tools have been indispensable in this regard, providing designers with the assurance that despite the myriad of transformations a design undergoes, the end result is functionally equivalent to the original specifications. This integrity across various stages, including architecture design, logic synthesis, technology mapping, and place-and-route, is the bedrock upon which reliable electronic design is built.
- **Optimization of PPA and Other Design Factors:** The relentless pursuit of optimizing performance, power, and area is central to EDA. As designs scale and complexities increase, the balance between these three aspects becomes more challenging to achieve. Tools dedicated to PPA optimization employ a variety of techniques, including predictive modeling, heuristic algorithms, and iterative refinement, to squeeze out efficiencies at every level of design. Meanwhile, the traditional PPA triad is no longer the sole focus. With the advent of ultra-deep sub-micron technologies, new concerns have emerged. Circuit reliability has taken center stage, with issues such as electromigration and thermal effects becoming critical. Manufacturability is another growing concern, as variability in fabrication processes can significantly impact yield and performance.

In the fiercely competitive realm of electronic product development, reducing time-to-market (TTM) is paramount. The rapid evolution of consumer electronics, exemplified by the yearly refresh cycles of smartphones and wearables, underscores the urgency to expedite product launches to capture market share and meet consumer expectations. This pressure significantly impacts the EDA process, where the need for TTM can sometimes compromise design thoroughness, leading to potential flaws. For instance, under the gun to release the next generation of microprocessors, teams may bypass exhaustive verification in favor of meeting launch windows, risking the introduction of bugs into the final product. When such issues are not amendable through engineering change orders (ECO) [25], they necessitate a costly and time-consuming redesign, further exacerbating time-to-market pressures. Therefore, this cycle highlights the crucial need for EDA solutions that not only streamline design and verification processes but also ensure design accuracy from the outset.

2.2 EDA for Front-End Design

In the 1980s, the growth of the semiconductor sector was hindered by the manual creation of large schematics, significantly limiting design productivity [26]. The narrative of front-end EDA tools is a testament to the field's evolution from the era of hand-drawn schematics to the sophistication of automated logic synthesis. This evolution has been underpinned by the introduction of hardware description languages (HDLs) like Verilog and VHDL, which have become the bedrock for digital design representation, simulation, and verification.

A typical front-end design flow, also known as logic design, is shown in Fig. 2 a), in which the design specification is transformed into a logic netlist. The front-end design flow begins with a design specification, followed by architecture exploration. Subsequently, HDLs are created to translate the design into a form suitable for implementation, typically at the RTL abstraction level. The introduction of hardware construction languages (like Chisel [27]) and C/C++ high-level synthesis (HLS) adds a new dimension to front-end design and offers more flexibility and efficiency in addressing the complexities of modern front-end design.

After RTLs are created or generated from HLS tools, designers first use static analysis tools such as Lint [28] to identify potential errors and then apply various verification techniques, including logic simulation, emulation, and various formal methods (e.g., model checking). These techniques collectively contribute to validating the functionalities of the RTL design faithfully following the design specification. The verification and testing processes, spanning various transformations and stages, are integral components of design flows, typically consuming between 60% to 70% of the total engineering efforts allocated. This substantial investment underscores their critical role in ensuring the functionality and reliability of circuit designs. Across diverse abstracts of circuit designs, a plethora of verification techniques are employed, reflecting the nuanced requirements and challenges encountered at each stage of development. For example the C-RTL equivalence checking rigorously compares the RTL implementations against the C-based specification models. This verification method, as evidenced by studies such as [29], [30], is frequently applied, particularly in the context of data-path intensive designs, as highlighted by Hector from Synopsis [31]. Given that circuit designs within this abstract primarily encapsulate hardware behavior while abstracting concrete physical details, theorem provers and SMT solvers emerge as pivotal tools for enhancing verification efficacy [32], [33].

Next, the RTL implementation undergoes the next stage in the design flow, wherein logic synthesis tools have revolutionized the way HDL code is transformed into gate-level representations. Logic synthesis typically involves three main steps: elaboration, logic optimization, and technology mapping. The primary objective of logic synthesis is to transform RTL codes into a gate-level netlist that meets specific design constraints while optimizing for power efficiency, maximizing performance, and minimizing the required silicon area, all within an acceptable timeframe. An indispensable aspect of logic synthesis involves conducting logic and sequential equivalence checks between optimized netlists and their initial counterparts, as underscored by studies such as [34], [35], [36]. Furthermore, custom equivalence checking techniques have been tailored to cater to specific circuit design requirements, such as those pertaining to clock-gating [37].

The collective progression of these front-end design and verification tools has not only streamlined the design process but also expanded the realm of what is possible in digital circuit design. As we navigate increasingly complex design landscapes, these tools have become indispensable in the relentless pursuit of innovation and optimization in digital systems.

2.3 EDA for Back-End Design

For modern chip design, the back-end design flow, also referred to as layout design, is depicted in Fig. 2 b), transitioning from a gate-level or generic technology (GTech) netlist to a finalized layout [38].

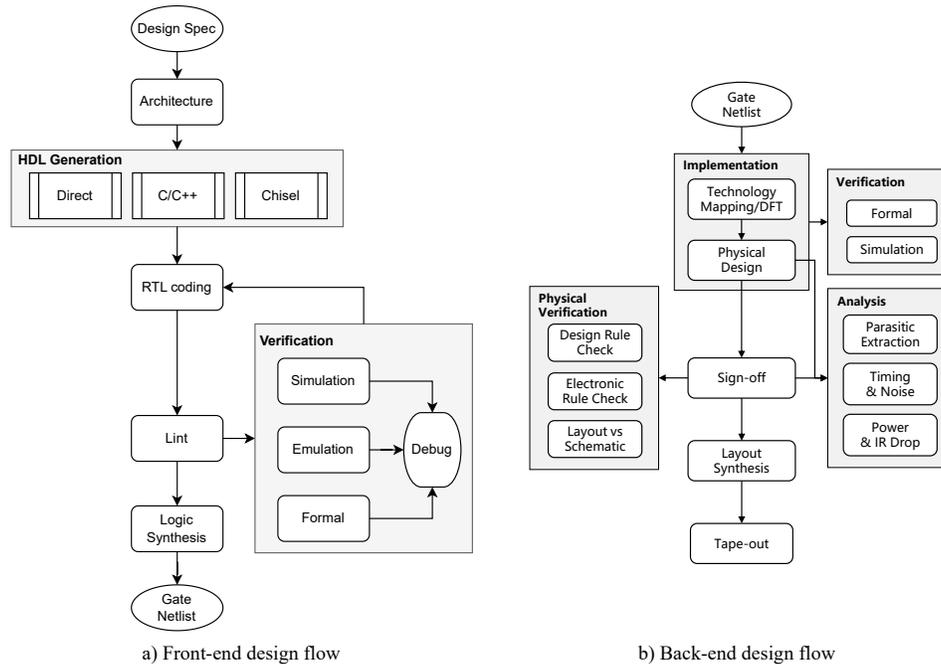


Fig. 2: Typical front-end and back-end design flows.

This intricate process initiates with technology mapping, where a process library is applied to adapt the synthesized gate-level netlist to a specified technology library, with a keen focus on optimizing PPA constraints. To enhance testability for mass production, testability features such as scan chains, built-in self-test (BIST) circuits, and boundary scan are incorporated into the design. The subsequent phase, physical design, is tasked with establishing the chip's physical layout, entailing floorplanning, power delivery network (PDN) design, placement, clock tree synthesis (CTS), and routing.

- **Floorplanning:** Floorplanning establishes the chip's physical layout by optimizing the placement of major blocks to minimize interconnect lengths and ensure efficient silicon area utilization. It involves strategic arrangement considering timing, power, and thermal constraints to set a foundation for the design.
- **Power Delivery Network (PDN) Design:** PDN design ensures stable power supply across the chip, aiming to minimize voltage drop and maintain power integrity. The design of power and ground networks is crucial for delivering power efficiently, with considerations for IR drop, current density, and electromigration.
- **Placement:** Placement optimizes the arrangement of standard cells or IP blocks within the floorplan to enhance performance, power, and area. It strategically positions components to reduce wire length, congestion, and considers timing and thermal impacts, employing algorithms to find an optimal configuration.
- **Clock Tree Synthesis (CTS):** CTS distributes the clock signal to synchronize the circuit's operations with minimal skew and jitter. Designing a balanced clock distribution network ensures reliable and synchronized performance across the chip.
- **Routing:** Routing connects the components based on the established placement and netlist, aiming to complete

interconnections without design rule violations or signal integrity issues. It optimizes for shortest paths, minimizes crosstalk and delay, and manages layer assignment and congestion.

As chip designs escalate in complexity, the functionalities of back-end EDA tools extend beyond mere layout creation and routing, embracing a multi-faceted optimization challenge. For example, thermal analysis tools empower designers to forecast and address thermal hotspots, guaranteeing the chip's dependable performance across diverse environmental scenarios. Also, various design for yield (DfY) strategies are required to maximize the manufacturing yield by identifying and mitigating potential yield detractors, performing layout adjustments to address process variations, defect probabilities, and other manufacturing imperfections. Advanced DfY tools and methodologies analyze critical areas, apply lithography-friendly design principles, and optimize the layout to enhance robustness against variations in the fabrication process, ensuring higher yields and reliability of the final product [39].

Physical verification stands as a critical final step in the back-end design phase, ensuring that the chip layout adheres to all necessary specifications and standards before proceeding to manufacturing. This process involves an array of checks, including design rule checking (DRC), electrical rule checking (ERC), and layout versus schematic (LVS) verification. DRC is essential for validating the layout against a set of predefined rules to ensure manufacturability, focusing on physical dimensions and spacing between circuit elements to prevent fabrication errors. ERC goes a step further by examining the electrical integrity of the design, identifying issues such as signal integrity, power distribution problems, and ensuring the circuit meets its functional requirements. Lastly, LVS verification confirms that the layout accurately reflects the original schematic design, guaranteeing that the physical representation matches the intended circuit behavior. Together, these verification steps identify and rectify potential layout issues, safeguarding the correctness of the final chip.

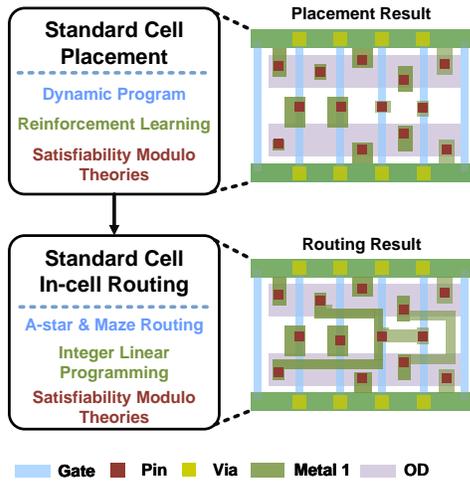


Fig. 3: A typical standard cell design flow.

In summary, the back-end EDA tools have fundamentally transformed the landscape of chip design, empowering designers to craft complex integrated circuits that house billions of transistors operating in unison on a single chip. As semiconductor technology progresses, the significance of EDA tools in the back-end design phase is poised to grow, continuing to fuel innovation and enhance efficiency in chip design research and engineering practices.

2.4 EDA for Specialized Circuits

Beyond EDA tools for regular digital circuit designs, the field has witnessed a notable specialization in toolsets designed to meet the unique requirements of standard cells, datapath units, and analog circuits. This evolution underscores the maturation of EDA, providing designers with tailored solutions to optimize these fundamental components efficiently. Specialized EDA tools have become indispensable in addressing the nuanced challenges presented by each component type, enhancing the precision and performance of chip designs.

2.4.1 EDA for Standard Cells

Standard cells, the building blocks of digital ICs, follow predefined structures that align with a library’s specifications, enabling their reuse across diverse designs. The focus of EDA tools in standard cell design is primarily on automating the layout generation process, encompassing crucial steps like placement and in-cell routing, as shown in Fig. 3.

The placement process is dedicated to determining the optimal transistor locations within a cell to maximize space utilization while maintaining functionality and performance integrity. The common solution algorithms for the placement includes dynamic program, reinforcement learning, and satisfiability modulo theories. Innovations in placement strategies, as highlighted in [40], [41], have introduced methods to expedite this intricate procedure while ensuring routability and design efficiency. In contrast, in-cell routing tackles the intricate task of establishing connections within the cell, a process complicated by the rigorous area constraints of standard cells. The in-cell routing are usually solved by A-star, interger linear programming, and satisfiability modulo theories. This stage demands specialized routing solutions, distinct from those applied to broader digital circuits, to navigate the tight confines of cell layouts. Contributions from [42], [43] have

provided targeted approaches to in-cell routing, addressing the unique challenges of standard cell design.

2.4.2 EDA for Datapath Circuits

The evolution of datapath circuits, from individual components such as adders, multipliers, multiply-accumulate (MAC) units to the entire datapath, is a testament to the continuous advancements in EDA technologies. Over the years, EDA tools have evolved to address the increasing complexity and performance demands of these critical components.

Adders: Adders serve as the cornerstone of arithmetic operations in digital circuits. The design of adders, from simple ripple-carry to more advanced carry-lookahead and prefix adders, has significantly benefited from EDA tools. These tools employ optimization algorithms to reduce latency, conserve area, and minimize power consumption, crucial for enhancing the overall performance of digital systems. The capability of EDA tools to simulate various adder configurations allows designers to select the most suitable architecture for specific applications, balancing speed with resource utilization.

Specifically, prefix-tree adders, recognized for their efficiency in parallel carry computation, have seen significant development and optimization through EDA solutions. Early designs focused on basic parallel prefix adders like the Kogge-Stone and Brent-Kung adders, which provided a foundation for understanding the balance between speed and area [44]. Recent advancements have introduced more sophisticated designs such as the Sparse Kogge-Stone and Spanning Tree adders, optimizing for both power efficiency and silicon area [45]. Datapath compilers have become instrumental in navigating the trade-offs between different prefix-tree configurations, employing algorithmic and heuristic methods to select the optimal structure for a given application scenario.

Multipliers: Multipliers are pivotal in performing fast arithmetic computations, crucial for applications ranging from general computing to specialized tasks in signal processing and machine learning. EDA technologies have facilitated the design of high-performance multipliers by exploring innovative architectures like Booth encoding and Wallace tree multiplication.

The Wallace Tree technique involves grouping the partial products generated from the multiplication process and then summing these groups in stages, which reduces the overall height of the addition tree and, consequently, the propagation delay. This architecture is particularly favored in digital signal processing (DSP) and graphics processing units (GPUs) where rapid mathematical computations are critical. Over the years, enhancements to the Wallace Tree architecture have aimed at optimizing its layout to minimize area and power consumption while maximizing speed, reflecting the ongoing advancements in EDA tools to meet the evolving demands of semiconductor technology.

MAC Units: The design of MAC units, essential for digital signal processing and deep learning applications, has similarly benefited from the innovations in EDA tools. The integration of optimized adder designs with efficient multipliers within MAC units is critical for achieving high throughput and low latency. EDA tools now utilize analytical models and simulation-based methods to explore various MAC unit architectures, including fused multiply-add (FMA) configurations that perform multiplication and addition in a single operation and pipelined designs, to meet specific performance goals [46].

Floating-Point Units (FPUs): Floating-point units are essential for executing arithmetic operations on floating-point numbers, a necessity in applications requiring a wide dynamic range, such as scientific computing, graphics, and machine learning algorithms.

The evolution of FPUs under the guidance of EDA tools highlights the industry’s commitment to addressing the precision, performance, and power efficiency challenges inherent in floating-point operations. Techniques such as pipelining and parallel processing have been integral in enhancing the throughput of FPUs, allowing for simultaneous execution of multiple floating-point operations. Advances in EDA methodologies have facilitated the exploration of novel FPU designs, such as the adoption of FMA units, as in MAC unit designs.

Datapath circuits: Beyond individual components, the design of entire datapath circuits, which comprise a combination of adders, multipliers, MAC units, and other logic elements, represents a complex challenge addressed by EDA tools.

These tools adopt a comprehensive strategy for refining datapath circuits, ensuring seamless integration and peak efficiency among components. As depicted in Fig. 4, the design journey initiates with pinpointing a target application and its corresponding architectural design, thereby defining a broad and intricate design space. This space might include, for example, CPU tasks like SPEC2017 benchmarks or GPU tasks such as matrix multiplication, targeting either CPU or GPU architectures accordingly. The design space diverges into two principal domains: the application space, outlining application-specific parameters like dataflow patterns or neural network mapping strategies, and the architecture space, detailing the structural and resource parameters, such as CPU issue width or the quantity of MACs in a neural processing unit (NPU).

The intersection of parameters from these domains establishes a “design point”, which, upon post-compilation application mapping, is subjected to thorough evaluation and validation via cutting-edge EDA tools. This rigorous process iteratively explores and assesses new design points until the exploration goals are achieved.

The progression to new design points is typically steered by optimization algorithms, which have advanced significantly. These optimizations fall into two categories: black box optimization, which proceeds without presuppositions about the design space, often utilizing Bayesian Optimization (BO) for its efficacy in exploring datapath circuit design spaces, and other black box methods like simulated annealing (SA), genetic algorithms (GA), and hill-climbing techniques. Conversely, optimizations that incorporate domain knowledge demand an in-depth understanding of the architecture, aiming for enhancements through precise, targeted adjustments to the datapath. Techniques such as bottleneck analysis have shown to outperform conventional black box approaches by focusing on specific areas for improvement within the datapath architecture.

2.4.3 EDA for Analog Circuits

The design process for analog and mixed-signal ICs significantly differs from digital design, showcasing the unique challenges and complexities of analog circuits.

Fig. 5 illustrates a typical analog IC design flow, starting with a detailed set of circuit specifications covering area, power, and performance requirements. The front-end design phase is crucial, establishing the pre-layout circuit netlist that defines the circuit’s functionalities via meticulous topology design and device sizing. This phase sets the foundation for the circuit’s operational features and optimization criteria.

Moving to the back-end, attention turns to physical layout implementation. Analog physical design, including placement and routing stages similar to digital methods, requires a detailed approach due to analog circuits’ sensitivity to parasitics. This phase also incorporates considerations for parasitic effects, component matching, and other layout-dependent factors essential for preserving the circuit’s integrity and performance [47].

A major challenge in analog design is performance optimization, marked by its nonlinearity and the lack of clear functional expressions. Despite these obstacles, the EDA community has significantly advanced the automation of analog IC design over the years. These efforts have covered various areas, such as topology selection or exploration [48], analog sizing [49], analog placement-and-route [50].

In summary, the journey of specialized circuit designs encapsulates a dynamic interplay of art and science. As technologies advance and design requirements become more stringent, the role of EDA tools in facilitating efficient, accurate, and innovative design solutions continues to be of paramount importance.

3 AI FOR EDA: STATE-OF-THE-ART

The prowess of deep learning, particularly its capability to discern patterns from historical design data, offers promising enhancements to EDA processes. This modern thrust is propelled by an ambition to harness the extensive repository of design knowledge accumulated across decades to drive superior and more efficient design methodologies.

3.1 Supervised Learning in EDA

The utilization of supervised learning in EDA represents a significant stride towards integrating AI into the optimization and estimation of design objectives. This subsection categorizes various supervised AI4EDA solutions based on their application stage within the standard design flow, highlighting seminal works in each category for a focused overview. For those seeking an exhaustive review, references such as [15], [51] offer comprehensive surveys on the subject.

3.1.1 Pre-RTL ML Methods

At the architecture level, supervised ML methods diverge into two primary categories: ML for rapid system modeling and ML as a design methodology.

- **ML for Fast System Modeling:** This approach employs ML to quickly estimate performance and power metrics of circuits and systems. Notable examples include the work by Joseph et al. [52] and Ithemal [53], which apply linear and recurrent neural network (RNN) models for CPU performance modeling, respectively. McPAT-Calib [54] enhances CPU power modeling by integrating ML models with the analytical tool McPAT for calibration. PANDA [55] advances this approach by reducing training data requirements and eliminating dependency on McPAT for power modeling. Boom-Explorer [56] automates design space exploration for the RISC-V BOOM microarchitecture. Beyond CPUs, XAPP [57] predicts GPU performance by analyzing dynamic and static properties of single-thread CPU code, while Wu et al. [58] model GPU power by examining kernel scaling behaviors. SVR-NoC [59] focuses

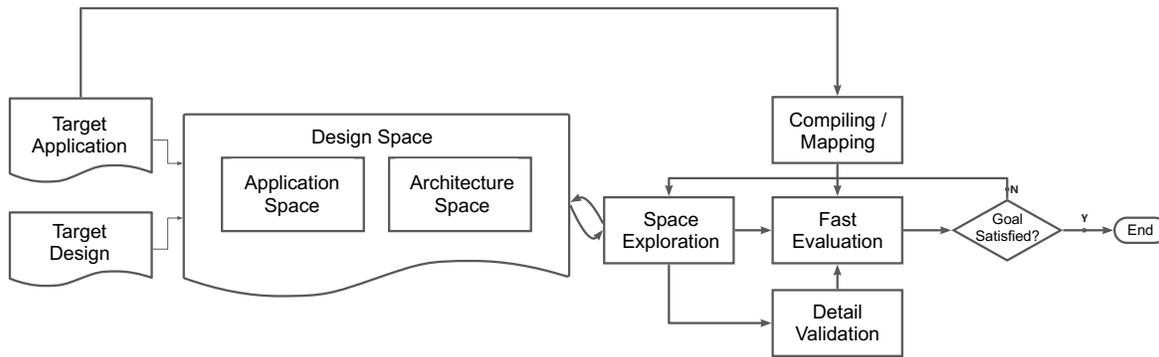


Fig. 4: A typical datapath circuits design flow.

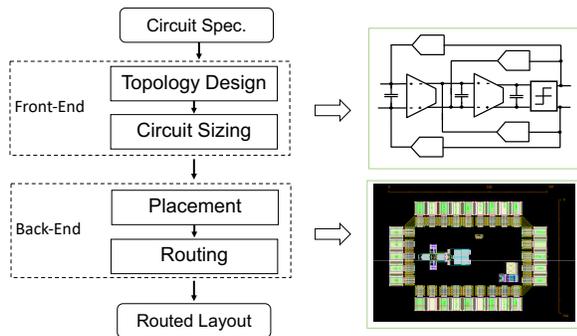


Fig. 5: A typical analog IC design flow.

on predicting latency and waiting times in mesh-based network-on-chips (NoCs).

- ML as a Design Method:** In microarchitecture design, ML techniques facilitate innovative solutions. Shi et al. [60] employ an LSTM model to derive insights from historical program counters for cache replacement using an SVM-based predictor. Pythia [61] reimagines prefetching as a reinforcement learning challenge, while Hermes [62] leverages ML to predict off-chip load request outcomes. Additional applications include task allocation [62], power management [63], and resource management for CPU [64] and AI accelerators [65].

In high-level synthesis, the application of ML models for rapidly estimating design metrics has become increasingly prevalent. For instance, Dai et al. [66] focus on timing and resource usage, Pyramid [67] estimates throughput, Ustun et al. [68] look at operation delay, Zhao et al. [69] consider routing congestion, and Lin et al. [70] dedicate their efforts to power consumption analysis. These studies underscore the versatility of ML in covering a broad spectrum of design metrics, highlighting its capacity to provide comprehensive insights early in the design process.

Moreover, ML’s role extends to facilitating design space exploration (DSE) in HLS, exemplified by the work of Ustun et al. [68], Liu et al. [71], and Meng et al. [72], who implement active learning strategies to navigate the DSE, using predictive ML models as stand-ins for actual synthesis processes. This approach allows for a more efficient evaluation of design alternatives without the need for exhaustive synthesis runs. Additionally, contributions by Kim et al. [73], Mahapatra et al. [74], and Wang et al. [75] demonstrate the integration of ML with traditional optimization algorithms,

enhancing their efficacy in navigating complex design spaces. Sun et al. [76] introduced a novel approach using correlated multivariate Gaussian process models to capture the intricate interdependencies among multiple objectives across various design fidelities. Yu et al. [77] proposed the IT-DSE framework, leveraging a surrogate model pre-trained on historical design data to refine the search process, illustrating how accumulated design knowledge can be effectively reused to optimize new projects.

In the realm of tensor computations, HASCO [78] employs ML for DSE. This methodology optimizes both software programs and hardware accelerators, showcasing ML’s capacity to bridge the gap between software and hardware domains to achieve optimized system performance.

3.1.2 RTL-Stage ML Methods

At the RTL stage, innovative ML solutions have emerged to predict the PPA without conducting logic synthesis. Initial attempts, such as SNS by Xu et al. [79] and the work by Sengupta et al. [80], employ a methodology where the RTL code is converted into an abstract syntax tree (AST) format, from which features are extracted to forecast the design’s PPA. Subsequent advancements, including SNS-v2 [81] and MasterRTL [82], claim enhanced accuracy compared to earlier efforts, showcasing the rapid progress in ML applications for RTL analysis. Additionally, there has been a focused effort on applying ML for precise timing or logic estimation [83], [84].

Power modeling at the RTL stage has also attracted lots of attention. There are two primary categories: design-time power estimation and runtime on-chip power modeling. For design-time estimation, PRIMAL [85] stands out for offering per-cycle power evaluations tailored to each target design, alongside other notable ML-based approaches [86], [87]. For runtime power modeling, DEEP [88] introduces an efficient on-chip model that incorporates low-overhead hardware design, utilizing ML to identify power-correlated RTL signals, or ‘power proxies’. This method, along with other ML-based on-chip power modeling solutions like [89], [90], demonstrates the potential of ML in creating dynamic power models that adapt to real-time conditions. Moreover, APOLLO [91] presents a versatile solution applicable to both design-time and runtime scenarios. Simmani [92] and the early power modeling work [93] focus on fast power emulation on FPGA and other platforms, highlighting the broader applicability of ML methods in facilitating efficient power analysis during the design phase.

In the realm of RTL testing and verification, Bayesian networks, as explored by Fine et al. [94], offer a probabilistic model-based

approach for coverage-based test generation, underscoring the potential of ML in optimizing test planning. Design2Vec [95] advances this further by learning semantic abstractions of RTL designs, facilitating functionality prediction and efficient test generation that notably shortens verification cycles. Katz et al.'s [96] decision tree-based method for learning microarchitectural behaviors exemplifies ML's utility in enhancing test stimuli quality.

3.1.3 Netlist-Stage ML Methods

Within the netlist stage, supervised learning methods have been leveraged to address a spectrum of challenges including logic synthesis, quality-of-results (QoR) prediction, verification support, and security concerns.

ML-based models have been particularly effective in assessing synthesis quality and influencing the optimization process. For instance, LSOracle [97] utilizes ML to determine the most appropriate optimizers for various logic networks, thereby enhancing the overall synthesis outcomes. Yu et al. [98] propose to classify and select among multiple random synthesis flows by their quality, subsequently focusing on the most efficacious ones. Their further research [99] extends to evaluating expected delay and area outcomes for synthesis flow candidates, offering a data-driven approach to guide synthesis decisions.

More recent advancements, such as AlphaSyn [100] integrate Monte Carlo tree search with tailored learning strategies for area reduction, showcasing the potential of combining ML with heuristic search techniques for synthesis optimization. Additionally, SLAP [101] targets the enhancement of design timing by identifying and utilizing candidate cuts that lead to improved synthesis results during technology mapping. Their subsequent work [102] further demonstrates the ability of ML models to pinpoint post-routing timing critical paths, focusing technology mapping efforts on these areas to minimize delays. DeepGate2 [17] develops a pre-trained model that predicts the behavioral correlation of logic gates in netlists and prioritizes SAT-sweeping process to accelerate *frac* optimization operation.

Following logic synthesis, innovative machine learning solutions are being developed to foresee the post-physical design quality of previously unknown circuit netlists. Tools like Net² [103] pave the way by predicting wirelength and timing information, effectively capturing the implications of placement on the netlist. GRANNITE [104] advances this further by facilitating the propagation of RTL toggle rate down to the gate-level netlist, aiming for rapid and accurate average power estimation. Similarly, GRAPSE [105] evaluates average power based on unoptimized and unmapped netlists, showcasing improvements in both speed and precision of power estimation. Recently, DeepSeq [106] learns a generic sequential netlist representation that accurately embeds the switching activity behavior and predicts the dynamic power estimation.

Moreover, ML methods have shown exceptional prowess in deriving high-level abstractions from bit-blasted netlists, unlocking new potentials across various domains within EDA. These high-level abstractions are instrumental in enhancing functional verification, logic minimization, datapath synthesis, and the detection of malicious logic within circuits. For instance, tools like ReIGNN [107] and GNN-RE [108] utilize ML for reverse engineering tasks, such as identifying state registers and deciphering the functionality of subcircuits. Additionally, ABGNN [109] leverages graph neural networks to delineate the boundaries of arithmetic blocks in flattened gate-level netlists, while GAMORA [110]

employs GNNs to infer high-level functional blocks from gate-level data. The success of these methodologies is largely attributed to the capacity of GNNs to discern intricate structural patterns and relationships within netlists, underscoring the transformative impact of ML in enhancing the efficiency and intelligence of EDA processes.

3.1.4 Layout-Stage ML Methods

The layout stage presents a crucial phase where ML methods have been increasingly applied to predict or optimize various design metrics such as wirelength, routability, timing, and IR-drop.

ML for Placement Stage Enhancements The placement stage, which determines the optimal locations of macros and standard cells in the layout, is pivotal for achieving desired design metrics. Early applications of ML aimed to augment traditional placement strategies. PADE [111] incorporates support vector machines (SVM) and neural networks for datapath extraction and evaluation, facilitating datapath-aware placement strategies. DREAMPlace, developed by Lin et al. [112], conceptualizes the placement challenge as akin to training a neural network, thus accelerating the global placement process by harnessing GPU computing capabilities. Building on DREAMPlace, Agnesina et al. [113] apply multi-objective Bayesian optimization for macro placement design space exploration, demonstrating the potential of ML in enhancing macro-placement outcomes.

ML also assists in predicting design metrics in the later routing phase, benefiting both iterative refinement and early-stage optimization. Many studies have explored early-stage routability prediction. RouteNet [114] uses a CNN to forecast the post-routing design rule violations (DRVs), thus avoiding difficult-to-route placements. Another study [115] guides macro placement based on predicted routability. Chang et al. [116] introduce a neural architecture search (NAS) for the autonomous development of routability prediction models, eliminating the need for manually designed machine learning models. Pan et al. [117] propose a federated learning-based approach for routability evaluation, addressing data privacy concerns. To achieve better routability prediction performance, Zheng et al. propose a multimodal neural network Lay-Net [118], which aggregates both layout and netlist information. The ultimate purpose of routability prediction is to assist routability optimization. Liu et al. [119] incorporate a fully convolutional network (FCN)-based routability prediction model into the DREAMPlace framework, using it as a penalty factor to explicitly optimize for routability. PROS [120] introduces a routing congestion predictor as a plug-in for commercial placers, effectively adjusting cost parameters to mitigate congestion issues. Moreover, Zheng et al. [121] develop LACO, a look-ahead mechanism designed to address the distribution shift problem in congestion modeling.

Timing is another important metric for placement. The field of pre-routing timing prediction at the placement stage has witnessed a range of modeling approaches leveraging various features and machine learning techniques. Studies like those by Barboza et al. [122] and He et al. [123] have implemented tree-based methods, incorporating careful manual feature extraction. TF-Predictor [124] employs Transformers to treat timing paths as sequences, while Guo et al. [125] have devised a customized GNN inspired by static timing analysis mechanisms. Additionally, recent work by Wang et al. [126] addresses the re-structuring of netlists due to timing optimization, integrating graph data from netlists with layout image information through multimodal fusion. Moreover, Liang et

al. [127] focus on cross-talk prediction, exploring various machine learning models for this purpose. To reduce turn-around time at the pre-routing stage, Liu et al. [128] propose a concurrent learning-assisted early-stage timing optimization framework called TSteiner, which guides the refinement of Steiner points based on gradients obtained from a GNN-driven timing evaluator.

ML for Sign-Off Enhancements: During the routing and sign-off stages, the precision of sign-off timing, especially using the path-based static timing analysis (PBA), becomes crucial. However, the PBA process is time-consuming, leading to the application of machine learning models for predicting path-based timing based on quicker graph-based analysis (GBA) results. The pioneering work by Kahng et al. [129] was instrumental in predicting PBA from GBA using carefully engineered features and a tree-based model. Subsequent studies, such as [124], [130], have delved into various machine learning models, including transformers and GNN, to enhance the accuracy of GBA-PBA predictions.

Additionally, IR-drop analysis is a critical component in the sign-off stage. Several studies have investigated rapid IR-drop estimation using machine learning, focusing on either static or dynamic analysis to cater to different requirements. For instance, works like IncPIRD [131] and XGBIR [132] concentrate on static IR-drop analysis. In contrast, studies such as [133] target dynamic IR-drop analysis.

ML for Manufacturability Enhancements: In the field of design for manufacturing (DFM), leveraging ML has become pivotal for bolstering the reliability of lithography and manufacturing processes, with layout patterns often analyzed as images. Studies like GAN-SRAF [134], GAN-OPC [135], Develset [136], and L2O-ILT [137] use various ML methods to improve mask synthesis printability. Other works, such as those by Watanabe et al. [138], Ye et al. [139], Lin et al. [140] and Chen et al. [141], focus on lithography modeling to simulate printed patterns from mask clips. For identifying layout patterns prone to printing failures like shorts or opens, ML-enhanced lithography hotspot detection is explored in various studies. For example, Yang et al. [142] propose to extract layout features with discrete cosine transform and utilize a CNN architecture for hotspot detection. The performance is further improved with the proposed bias learning algorithm because of the imbalanced dataset. Inspired by the object detection problem in computer vision, Chen et al. [143] propose to detect multiple hotspots within large layouts simultaneously. In [144], the binarized neural network is utilized to speed up the hotspot detection flow. New network architecture is designed based on residual networks to achieve higher detection accuracy and performance. Additionally, ML further contributes to yield estimation and analysis, as seen in works like Ciccazzo et al. [145], Nakata et al. [146], and Alawieh et al. [147].

3.1.5 Cross-Stage ML Methods

In addition to stage-specific applications, ML4EDA has significantly impacted the broader task of design flow tuning, garnering substantial interest.

Kwon et al. [148] introduce a novel approach that blends tensor decomposition with regression analysis to recommend parameters for both logic synthesis and physical design stages, demonstrating ML's capability to streamline design parameterization. FIST [149] utilizes a clustering strategy to automate the adjustment of flow parameters, aiming for enhanced design quality. Furthermore, PTPT [150] presents a multi-objective Bayesian optimization frame-

work equipped with a multi-task Gaussian model, significantly improving the design flow tuning process's efficiency.

Verification, a critical component throughout the design process, has also seen the integration of ML to validate circuit design correctness. Cho et al. [151] propose an efficient lithography-aware router, which moves lithography verification to the routing stage, effectively enhancing the quality of the printed layout.

3.2 Reinforcement Learning in EDA

Reinforcement learning (RL) in EDA has emerged as a powerful method for navigating the expansive solution spaces inherent in logic synthesis and physical design, often uncovering innovative solutions that surpass traditional, intuition-based approaches. Innovations like Synopsys.ai [152] underscore this trend, showcasing AI-driven methodologies that enhance PPA metrics across the design spectrum.

In logic synthesis, Liu et al.'s PIMap framework [153] exemplifies the application of RL by optimizing LUT-based FPGAs through graph partitioning and iterative synthesis operation selection, leveraging parallelization for efficiency gains. FlowTune, introduced by Yu et al. [154], employs a multi-stage multi-armed bandit (MAB) strategy to constrain the search space and streamline the synthesis process. Pei et al.'s AlphaSyn [100], utilizing a domain-specific Monte Carlo tree search (MCTS), and Zhu et al.'s approach [155], framing logic synthesis as a Markov decision process (MDP) with a graph convolutional network (GCN), both illustrate the capacity of RL to thoroughly explore synthesis strategies. DRiLLS by Hosny et al. [156] and subsequent works like those by Peruvemba et al. [157] further extend this exploration, introducing constraints and optimization targets into the RL models to fine-tune synthesis outcomes. RL has also been applied to logic optimization challenges. For instance, Haaswijk et al. [158] and Timoneda et al. [159] leverage policy gradient methods and GCNs to optimize majority-inverter graphs (MIGs), showcasing RL's adaptability to various logic structures.

In physical design, the application of RL ranges from automating chip floorplanning, as demonstrated by Mirhoseini et al. [160], to minimizing area and wirelength in floorplanning processes like GoodFloorplan [161]. Agnesina et al.'s [162] use of RL to tune physical design flows for improved PPA metrics and RL-Sizer by Lu et al. [163] for gate sizing highlight RL's potential to refine physical design processes, including timing optimization [164] and mask optimization in the RL-OPC process [165]. For clock tree synthesis, research efforts are directed toward predicting the quality of the clock network and enhancing timing optimization by leveraging clock skew. GAN-CTS [166] employs a conditional generative adversarial network (GAN) combined with reinforcement learning for predicting and optimizing CTS outcomes.

3.3 Leveraging Large Language Models in EDA

The integration of generative AI, particularly large language models (LLMs), into IC designs is emerging as a transformative trend. By utilizing proprietary datasets, IC design companies can develop AI assistants to enhance and expedite the design process. These tools, capable of providing in-depth insights, automate and refine traditionally manual tasks like design conceptualization and verification. Consequently, a growing body of research explores the application of LLMs in EDA, tackling a broad spectrum of tasks including RTL code generation, task planning, script generation, and bug fixing. While still in the early stages, these

studies underscore the profound potential of LLMs to improve the efficiency and efficacy of EDA tools.

This section delves into the use of LLMs for RTL code generation—a key area of focus. It categorizes the research into LLM-aided RTL design generation and verification. Additionally, we explore LLM applications in generating EDA scripts and high-level architecture design.

3.3.1 RTL Generation through LLMs

The advent of large language models has ushered in a new era for RTL code generation, offering solutions that have the potential to redefine traditional approaches.

Early explorations in this domain primarily focused on evaluating models against simple design tasks, hindered by the absence of standardized benchmarks. This challenge has been recently addressed with the introduction of comprehensive benchmarks like RTL-LLM [167] and VerilogEval [168], facilitating a more robust comparison of LLM capabilities across complex design tasks. RTL-LLM stands out by providing an open-source benchmark with thirty detailed design tasks, accompanied by ground-truth RTL code for functionality verification. It emphasizes three core objectives: syntax correctness, functional accuracy, and design quality, showcasing a significant leap in performance through innovative prompt engineering techniques like self-planning. Similarly, VerilogEval expands the evaluation framework by gathering Verilog code from diverse sources to construct over 100 test cases. Its approach of collecting additional RTL code for model training demonstrates comparable performance with advanced models like GPT-3.5, yet its training data and model remain unreleased to the public.

Commercial LLMs are utilized for RTL generation, with initial attempts applying GPT-2 for code completion showing promising results [169]. Subsequent developments have introduced tools like ChipGPT [170] and AutoChip [171], which leverage GPT-3.5 to refine code generation through prompt engineering and feedback loops, further reducing the need for human intervention. Chip-Chat’s [172] achievement in designing a microprocessor with GPT-4 underscores LLMs’ potential to autonomously generate hardware description languages.

Recently, the shift towards fine-tuning open-source LLMs presents a viable alternative for customized model development, addressing privacy concerns in VLSI design. Projects like Chip-NeMo [173], RTL-Coder [174], and BetterV [175] have demonstrated significant advancements, employing domain adaptation techniques and automated training dataset generation to enhance LLM efficiency and performance for RTL code generation.

3.3.2 Enhancing Verification with LLMs

The application of LLMs extends beyond RTL code generation to the verification processes. These models assist in both functional correctness and security analysis, showcasing their versatility and depth in enhancing design validation.

Functional Verification through LLMs: LLMs have made significant strides in functional verification by translating natural language specifications into SystemVerilog assertions (SVAs). This process ensures that RTL implementations adhere to their intended specifications. Notably, [176], [177] leverage human-written specification sentences alongside RTL designs to generate precise SVAs. AssertLLM [178] takes a proactive approach by generating assertions directly from comprehensive specification documents, even before the RTL design phase. This method is complemented by a benchmark set that pairs natural language

specifications with golden RTL implementations, offering a robust framework for evaluating assertion generation. Furthermore, LLMs have achieved success in solving the Boolean Satisfiability (SAT) problem [179], which can be applied to verify arithmetic circuits.

Security Verification Leveraging LLMs: Security validation, critical in identifying and mitigating common vulnerability enumerations (CVEs), has also benefited from LLM integration. Ahmad et al. [180] demonstrate the capacity of LLMs to repair hardware security bugs, provided the bug’s location is known. Further research includes leveraging ChatGPT to recommend secure RTL code [181] and employing LLMs in hardware security assertion generation [182]. The latter develops an evaluation framework and benchmark suite that encompasses real-world hardware designs, illustrating LLMs’ potential to contribute significantly to security validation efforts.

3.3.3 EDA Script Generation and Architecture Design

The versatility of LLM-based solutions in EDA also extends to embrace tasks like EDA script generation and high-level architectural design.

EDA Script Generation: ChatEDA [183] introduces an LLM-based agent designed to facilitate EDA tool control using natural language, offering an alternative to traditional TCL scripts. This agent supports a range of operations from RTL code to the graphic data system version II (GDSII), encompassing automated task planning, script generation, and task execution, making EDA tools more accessible and efficient.

Architectural Design: GPT4AIGChip [184] leverages LLMs to generate C code for AI accelerator high-level synthesis. Similarly, Yan et al. [185] examine the use of LLMs in optimizing compute-in-memory (CiM) DNN accelerators, showcasing the model’s potential in enhancing computational efficiency. Further extending the scope, Liang et al. [186] delve into quantum architecture design, exploring the frontiers of quantum computing. SpecLLM [187] contributes to this growing body of work by providing a dataset of architecture specifications at various abstraction levels, investigating LLMs’ capabilities in both generating and reviewing these specifications.

3.4 AI for Specialized Circuits

The advent of AI4EDA also presents a unique opportunity to redefine the design and optimization of specialized circuits, including standard cells, datapath components, and analog circuits.

3.4.1 AI for Standard Cells

The application of AI in standard cell design, particularly in placement and routing, presents a unique set of challenges due to their high density and strict routability requirements. An AI-assisted approach, utilizing reinforcement learning, has been shown to improve placement sequences and routability, offering better wire length performance [188]. Additionally, RL methods have been used to address DRC violations post-routing [189], simplifying the routing process and enabling the use of A-star or maze routing for optimal solutions. Machine learning techniques have also facilitated the adaptation of DRC rules, easing the migration of standard cell layouts across technology nodes [190]. A notable area for AI application is in the evaluation of standard cell layouts, where machine learning models can rapidly assess performance without the need for detailed simulations.

3.4.2 AI for Datapath Circuits

Machine learning-based methods are emerging as a powerful tool for optimizing the design of datapath circuits, enabling enhanced efficiency and performance. By leveraging the distinct functionalities and structures of datapath circuits, AI can facilitate a more effective design optimization process.

Roy et al. [191] employ machine learning to predict the Pareto frontier for adders within the physical design domain. It exemplifies how machine learning can be leveraged for design space exploration, providing insights into optimal design configurations. Utilizing an integrated framework that combines variational graph autoencoders with graph neural processes, [192] develops a novel approach for automatic feature learning of prefix adder structures. This method facilitates sequential optimization, enabling the exploration of Pareto-optimal structures alongside quality metrics. Another study [193] employs multi-perception neural networks to analyze and learn from existing designs and performance data of adders and multipliers. This approach not only achieves high prediction accuracy but also outpaces traditional optimization methods in speed. Moreover, the RL-MUL framework [194] introduces a novel RL strategy for enhancing multiplier designs. By adopting matrix and tensor representations for the compressor tree and leveraging CNN as the agent, this method allows for dynamic adjustments to the multiplier structure, showcasing the adaptability of AI in complex design optimization.

3.4.3 AI for Analog Circuits

AI's integration into analog IC design automation marks a pivotal advancement, enhancing both the efficiency and effectiveness of algorithms. This integration capitalizes on graph and image data representations, mirroring circuit topologies and layouts [195], to address the challenges inherent to analog design—namely, slow performance evaluation and high search complexity.

AI in Analog Topology Generation: The integration of AI into the generation of analog topologies is revolutionizing the field by speeding up evaluation processes, honing in on more efficient search spaces, and improving optimization techniques. Among the diverse approaches, variational graph autoencoders (VGAEs) have been employed for circuit topologies as showcased by Lu et al. [196], while RL-based methods have been applied to power converters, as demonstrated by Fan et al. [197]. More broadly, Zhao et al. have utilized RL alongside predefined libraries to address a wider array of problems [198]. Poddar et al. have introduced a data-driven strategy for selecting topologies and sizing devices, employing a variational autoencoder (VAE) to synthesize data and thereby reduce simulation expenses [199]. To tackle the complexities of large circuit design, hierarchical methods are being investigated. Lu et al. have put forward a bi-level Bayesian optimization technique for $\Delta - \Sigma$ modulators [200], while Fayazi et al. and Hakhamaneshi have delved into intermediate topology representations and GNN models for voltage node prediction, respectively [201] [202]. These developments suggest that AI holds significant promise in streamlining the generation of complex topologies, including those of larger circuits comprising multiple sub-circuits.

AI in Analog Sizing: AI is playing a pivotal role in advancing optimization within the realm of analog sizing, notably through the use of ML as surrogate models and RL for direct optimization efforts. ML models, particularly feed-forward neural networks, have been adeptly trained to closely approximate circuit performance metrics. These models, when operated in inference mode, enable

the prediction of new, unseen design points, thereby enhancing the efficiency of the search process [203]. On another front, RL, especially via the GCL-RL algorithm, marries RL techniques with graph neural networks to adeptly optimize analog sizing across varying technological domains. This synergy leverages GNNs' robust capability to encapsulate circuit topologies within the optimization framework [204]. Such methodologies, along with other RL-centric approaches, aim squarely at the intricate balance between global exploration and local exploitation, a balance that is essential for achieving sample efficiency in analog sizing tasks. Innovative strategies, including the use of Voronoi trees for the decomposition of the design space and Monte Carlo tree search (MCTS) for honing in on local search areas, highlight the complex tactics employed to navigate the vast, high-dimensional optimization landscapes with greater efficiency [205]. The field's progress and the diverse methodologies employed are thoroughly reviewed in a dedicated book chapter, offering a deep dive into the significant advancements and techniques in ML applications for analog sizing [206].

AI in Analog Layout Automation: The application of AI in analog layout automation significantly enhances processes such as constraint extraction, placement, and routing, as extensively reviewed in [207].

For constraint extraction in analog layouts, graph-based methodologies are pivotal for identifying symmetry in netlists. These methods encompass graph similarity analysis, edit distance computation, and unsupervised learning for device matching, alongside convolutional graph neural networks for the prediction of layout constraints [208]. A detailed survey on these techniques is provided in [209].

ML's role in analog layout extends to automating the imitation of expert designs, modeling circuit performance, and optimizing the layout process. GeniusRoute [210] leverages variational autoencoders for making routing predictions that mimic human expertise, impacting various aspects of layout design including well generation [211], placement strategies [212], and cell generation processes [213]. CNNs and GNNs are utilized for predicting the performance of designs, thereby optimizing placement and minimizing the dependency on extensive simulations [214], [215]. The significant impact of ML on performance-driven placement and optimization in analog layout is thoroughly examined in [216].

Finally, addressing the pre-layout and post-layout simulation gap in analog IC design is vital. ML predicts post-layout parasitics directly from schematics to enhance simulation accuracy and speed up design. For example, ParaGraph [217] employs GNNs for accurate parasitic predictions, using ensemble models for specific value ranges. Early performance assertions using CNNs [218] and layout-aware optimization with BagNet [219], utilizing deep neural networks and evolutionary algorithms, streamline the design process. TAG combines text, self-attention networks, and GNNs for a comprehensive circuit representation, aiding in various predictions [195].

4 LARGE CIRCUIT MODELS: A NEW HORIZON

As discussed in the previous section, AI4EDA solutions have shown remarkable potential, yielding promising outcomes across a spectrum of tasks. However, these solutions predominantly exhibit a task-specific orientation, which, while effective in narrow applications, often limits their scalability and adaptability to the broad spectrum of design challenges.

Venturing into the domain of large circuit models (refer to Fig. 1) marks a bold departure from the previous AI4EDA solutions, moving towards a more integrated and AI-native design process. The term ‘large’ in LCMs signifies both the substantial model size and the vast array of circuit data collected from various EDA stages for circuit pre-training. Such a foundational model concept promises a unified framework that transcends task-oriented limitations, ensuring that LCMs are robust, versatile, and capable of handling the diverse tasks of modern circuit design with limited fine-tuning.

4.1 Motivation

The realm of AI4EDA, despite its advancements, faces inherent limitations by primarily repurposing machine learning models from disparate domains to tackle EDA challenges. This approach necessitates the development of distinct models for each specific EDA task. While these models have demonstrated efficacy on benchmark datasets, their ability to generalize to novel designs remains a subject of concern. The unique blend of computation and structure inherent to circuit data requires a nuanced understanding that transcends the capabilities of generic AI solutions. For instance, adapting LLMs for RTL generation without a deep comprehension of circuit design nuances often falls short of achieving optimal PPA results.

The emergence of large foundational models, such as BERT [2], GPT [5], and MAE [8]), has redefined AI’s landscape, offering a bifurcated approach of extensive pre-training on diverse data followed by targeted fine-tuning for specific tasks. This methodology has been instrumental in achieving breakthroughs across various data types, heralding a new era of AI applications. The introduction of multimodal foundation models like GPT-4V [13] and Gemini [14] further exemplifies this trend, facilitating previously unimaginable applications by harmonizing disparate types of data.

Drawing inspiration from these developments, we propose a paradigm shift towards AI-native EDA through the adoption of large circuit models. LCMs, with their focus on learning comprehensive circuit representations, are designed to encapsulate the intricate details and unique characteristics of circuits at every design stage. Echoing the CLIP model’s success in bridging text and vision, LCMs aim to forge a similar convergence within EDA, weaving together high-level functional specifications with the minutiae of physical layouts. This holistic approach not only promises to refine the EDA workflow but also aims to significantly reduce time-to-market and enhance the overall design quality such as PPA and circuit reliability.

By championing LCMs, we stand on the cusp of revolutionizing EDA, transcending task-specific limitations, and embracing a future where AI-native solutions drive innovation, efficiency, and excellence in circuit design.

4.2 Overview of LCMs

The EDA workflow, extending from initial specification to the detailed final layout, encompasses a variety of circuit design formats, each demanding distinct encoders within the LCMs. These encoders, designed to handle specific modalities – specification, architecture design, high-level algorithms, RTL design, circuit netlists, and physical layouts – are the core components of LCMs. To effectively leverage the diverse data inherent to each design modality, LCMs must be pre-trained with a focus on general yet comprehensive design knowledge. This involves not just a

superficial understanding but a deep encoding of the nuances present in each modality. For instance, in the circuit netlist modality, the encoded representations must encapsulate both the functional intent and the physical structure of the circuits. This depth of understanding facilitates a more accurate and cohesive foundation for subsequent design tasks. Please refer to Section 5 for details.

The next step in harnessing the power of LCMs involves the fusion and alignment of these unimodal representations to form a cohesive multimodal representation [23]. This process is critical in bridging the gaps between disparate stages of the design process, employing advanced techniques such as shared representation spaces, cross-modal pre-training, and innovative fusing strategies. These methodologies aim to synthesize the information captured in individual modalities into a unified, actionable framework that can guide the design process from conception to completion.

Since the specifications, RTL codes, netlists and layout designs are representative formats in front-end and back-end flows, the perspective paper outlines three primary alignment challenges:

- **Spec-HLS-RTL Representation Alignment:** Utilizing the transformative self-attention mechanism inherent to Transformers, this approach seeks to harmonize the representations of architecture design, high-level C/C++ prototypes, and RTL designs. This unified space enables the coexistence and interaction among these modalities, facilitating a seamless transition across design stages.
- **RTL-Netlist Representation Alignment:** Inspired by the groundbreaking CLIP model, this challenge leverages contrastive learning and mask-and-prediction training strategies. The goal is to map the embeddings of RTL designs and circuit netlists into a shared latent space, ensuring a coherent progression from logical design to physical implementation.
- **Netlist-Layout Representation Alignment:** The final alignment challenge focuses on the crucial step of ensuring that the physical layout accurately mirrors the detailed design captured in the netlist. This alignment is vital for the physical realization of the design, embodying the transition from theoretical models to tangible, manufacturable circuits.

By confronting these alignment challenges head-on, LCMs promise to revolutionize the EDA workflow, enabling novel applications and methodologies that were previously unattainable. This detailed exploration (please refer to Section 6) sets the stage for a comprehensive discussion on multimodal alignment techniques, further elaborated in subsequent sections, heralding a new era of AI-native circuit design.

4.3 Opportunities and Potentials

By accumulating knowledge learned from diverse circuit types and applying cross-stage learning on various design modalities, the potentials of LCMs extend across various aspects of design and verification:

- **Enhanced Verification:** LCMs promise to revolutionize verification by harnessing a deep, cross-stage understanding of circuit designs. This enables more streamlined verification processes, significantly reducing iterations and enhancing the detection of design flaws early in the design cycle.

- **Early and Precise PPA Estimation:** The comprehensive insights LCMs offer into design data empower them to provide early and accurate PPA predictions. This capability ensures that critical design decisions are informed and strategic from the outset, aligning with optimal design objectives.
- **Streamlined Optimization:** By pinpointing the true bottlenecks affecting PPA, LCMs can facilitate targeted optimizations. This not only accelerates the design optimization process but also ensures that improvements are effectively implemented across different design stages, enhancing overall design quality.
- **Innovative Design Space Exploration:** The intelligence imbued within LCMs opens the door to expansive design space exploration. Designers are equipped to discover novel architectures that ingeniously balance PPA trade-offs, fostering creativity and innovation in circuit design.
- **Generative Design Solutions:** Perhaps the most revolutionary aspect of LCMs is their potential to underpin generative models capable of autonomously crafting efficient and innovative circuits. This could drastically reduce the time-to-market for new chip designs, offering a competitive edge in the rapidly evolving semiconductor industry.

In essence, LCMs represent not just a technological advancement but a paradigm shift in how circuit design and verification are approached. The full realization of LCMs' potential, however, hinges on the development of sophisticated AI-native techniques for circuit representation learning, challenging the EDA community to explore and harness these untapped capabilities.

5 UNIMODAL CIRCUIT REPRESENTATION LEARNING

The journey toward an AI-native EDA paradigm embarks with the essential development of robust unimodal circuit representation learning. These foundational representations are the building blocks for the envisioned multimodal LCMs. This section delves into the nuances of unimodal circuit representation learning, underscoring its indispensable role in establishing a comprehensive and nuanced foundation for sophisticated LCMs. The insights garnered here are paramount for achieving a holistic comprehension of circuit data, which is crucial for the realization of advanced LCMs.

5.1 Representation Learning for Front-End Design

Circuit design commences with the specification and architecture design phase, where the high-level functional intents are formulated. At this juncture, techniques derived from natural language processing are invaluable, transforming specifications into structured, machine-interpretable representations.

As we descend the design hierarchy, representation learning must adeptly adapt to the increasing granularity of detail. At the SystemC and RTL stages, the representation's focus shifts to encompassing the logical and behavioral intricacies of the circuit. In this domain, machine learning paradigms such as LLMs for code, graph neural networks, and hybrid models become instrumental, skillfully capturing the complex logic structures and their interrelations.

5.1.1 Representation Learning for Architecture Design

The performance and power consumption of architectures exhibit an intrinsic dependence on specific application contexts. In pursuit of optimizing the trade-off among PPA for targeted applications, architectural designers traditionally employ detailed simulation tools complemented by extensive domain-specific expertise. This conventional methodology, while comprehensive, tends to be both time-intensive and prone to human errors. The advent of LCM presents a novel paradigm, facilitating rapid exploration of architectural design spaces by leveraging insights into the nuanced interactions between application workloads and architectural configurations. Thus, it is imperative for LCM to encapsulate application workload representations adaptable to various architectural designs.

Several endeavors have been undertaken in tasks related to architectural design. For instance, NPS [220] utilizes a specialized GNN called AssemblyNet for workload representation learning, leveraging both the application's code structure and its runtime states. Trained with a data prefetch task, AssemblyNet identifies the characteristics of typical program slices and minimizes the inaccuracy of sampling-based simulation. Perfvec [221] proposes to learn independent program and architecture representation for generalizable performance modeling. Supervised with an instruction incremental latency prediction task, the yielded model demonstrates applicability on performance modeling across different microarchitectures. On the other hand, several studies have explored the representation of architecture in depth. For instance, GRL-DSE [222] leverages graph representation learning to establish a compact and continuous embedding space for microarchitecture. This approach, utilizing self-supervised learning, enhances the efficiency of identifying optimal microarchitecture parameters. Meanwhile, daBO [223] presents an architecture representation for accelerators enriched with domain-specific knowledge. It involves the manual identification of critical factors that significantly influence the architecture's PPA, and seeks the optimal parameter combinations within this newly defined representation space.

However, existing studies still face challenges in workload and architecture characterization:

- Many models struggle to account for performance-critical factors like branch mispredictions and cache misses, which relate to broader historical states and resist capture through static execution snapshots. An effective LCM must grasp these long-term and complex relationships to accurately represent application workloads.
- The intricate relationship between application workloads and power consumption has been underexplored. An ideal LCM would not only integrate power-related factors tailored to varied application workloads, such as flip rates and dynamic voltage fluctuations, but also encapsulate the complex interplay between power consumption and performance, ensuring a cohesive modeling of both aspects.
- Current methods primarily concentrate on direct analysis of source code or simulation traces, which overlooks the incorporation of substantial domain knowledge accumulated by experienced architecture designers over the years. A LCM should aim to blend these disparate strands of knowledge, facilitating an enhanced representation learning in terms of accuracy and interpretability.

At architectural exploration stage, the focus should be on developing representations that accurately mirror the multidimensional nature of hardware design, capturing not just the static features

but also the dynamic interactions within the system. To achieve this, we should employ advanced ML techniques that can process and integrate information from various data sources, including code structure, runtime behavior, and architectural parameters. This process involves constructing multi-layered embeddings that reflect the hierarchical nature of hardware systems, from individual components to the entire architecture. These representations should be learned through a combination of supervised and unsupervised learning tasks, designed to highlight different aspects of the hardware’s performance and operational characteristics. By doing so, LCMs can provide a rich, nuanced understanding of the design space, guiding designers towards solutions that optimize performance, power, and area in concert.

5.1.2 Representation Learning for HLS/RTL

HLS and RTL represent two pivotal stages in the digital circuit design process. HLS provides a higher-level abstraction, utilizing high-level programming languages such as C, C++, or SystemC to articulate the functionality and behavior of the hardware system. Conversely, RTL offers a more granular view, detailing the data flow between registers and the operations on that data in Verilog or VHDL. Transitioning from HLS to RTL, designers typically employ HLS tools to synthesize the higher-level representation into its detailed RTL counterpart.

To incorporate deep learning in understanding and optimizing HLS/RTL representations, we can explore two innovative methodologies. One method interprets code as a series of tokens, analogous to words in natural language, making it possible to apply NLP techniques to HLS/RTL codes. A particularly effective strategy in this domain is masked language modeling (MLM), where certain tokens are obscured during model training, prompting a Transformer-based encoder (such as BERT) to infer the missing tokens. This self-supervised learning approach yields representations rich in the semantic essence of the hardware design, capturing the functional nuances at both the HLS and RTL levels. Another method may represent HLS/RTL designs as control data flow graphs (CDFGs) offers a graphical perspective, mapping out the control and data dependencies within the design. Here, advanced GNNs come into play, learning from the complex web of interactions and dependencies depicted in the CDFGs. This method allows for the extraction of comprehensive representations that embody the intricate structure and operational logic of the design, providing a solid foundation for subsequent optimization and synthesis tasks.

The former token view is more aligned with the high-level specifications and contains more syntax information. With the application of language models that excel in capturing global relationships, we can get representations that encompass the overall behavior and functionality of the design. Besides, the learned representations will benefit the generalizability and scalability of the attention-based models. On the other hand, the graph view is more aligned with the lower-level gate-level representations and contains more structural and semantic information. Compared to language models, GNNs focus more on extracting local information.

To enhance the effectiveness of the learned representations, we may consider combining these two views by employing multi-view learning techniques. There are different strategies for integrating these views. The simplest approach involves concatenating the representations obtained from each view and passing them through a multi-layer perceptron (MLP). This allows for the fusion of information from both views, leveraging their individual strengths.

Alternatively, a more sophisticated approach is cross-modal prediction, which facilitates deeper interaction between the two views. Through cross-modal prediction, the model is trained to predict one view based on the other view, encouraging the exploration of shared information and dependencies between the representations. By employing multi-view learning techniques, we can maximize the potential of the learned representations and create a more unified and enriched representation of HLS/RTL.

The learned HLS/RTL representations would offer a wide range of applications for various downstream tasks. For instance, they can be leveraged to predict PPA directly from the HLS/RTL, enabling efficient estimation of these crucial design metrics. Additionally, the learned representations can be employed for formal verification to verify the correctness and functional behavior of the design.

5.1.3 Representation Learning for Circuit Netlist

At the netlist level, the design serves as a pivotal junction bridging the front-end design phase with the subsequent back-end processes. Integrating machine learning into logic synthesis, physical design, or verification necessitates a nuanced understanding of the netlist’s graph topology alongside gate functionality. This dual focus ensures the netlist encapsulates both the high-level behaviors critical in front-end designs and the intricate structures that profoundly influence PPA in back-end designs.

Initiatives like the DeepGate Family [16], [17] stand at the forefront of crafting generalized gate-level representations. The first version [16] targets circuits in the and-inverter graph (AIG) format and innovatively employs random simulation outcomes to pre-train circuit netlists, with logic-1 probabilities as labels encapsulating crucial functional and structural insights. This pre-training strategy equips DeepGate to capture the core attributes of gate-level circuit designs, allowing for subsequent fine-tuning across a range of front-end applications, such as logic verification [224] and design for testability [225].

DeepGate2 [17] advances this approach by disentangling functional and structural representations within a netlist, learning distinct embeddings for each through specialized labels. Functional embeddings leverage pairwise truth table similarities for supervision, aligning netlists of similar functionalities in close proximity within the functional embedding space. This alignment aids in discerning behavioral similarities and discrepancies. Concurrently, structural embeddings predict pairwise reconvergence, mirroring topological nuances and the complex interconnectivity among logic cells in netlists. Beyond the DeepGate Family, FGNN [226] introduces a novel contrastive learning task focused on differentiating functionally equivalent from inequivalent circuits, enriching the dataset through strategic perturbations to generate logically equivalent circuit variants.

After technology mapping, netlists are transformed into a form optimized for the target technology, presenting new challenges and opportunities for representation learning. This stage is critical, as it directly influences the final PPA outcomes of the design. While we can still formulate the post-mapping netlist as a directed graph and utilize a GNN-based model similar to DeepGate to learn general representations, the complexity of post-mapping netlists, characterized by their technology-specific primitives and configurations, necessitates sophisticated representation learning techniques that can accurately capture the nuances of these transformations.

While the primary focus of logic synthesis has been on optimizing combinational logic, the sequential behavior of circuits

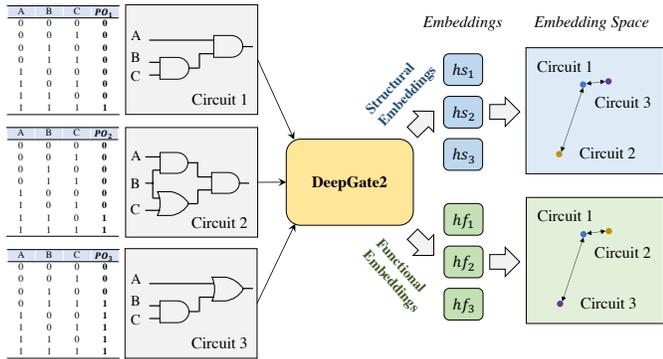


Fig. 6: DeepGate2: structural and functional disentangled netlist representation learning.

is also a critical facet to represent. DeepSeq [106] expands upon the DeepGate technique by elucidating the temporal correlations within sequential netlists. This advancement is facilitated by leveraging both transition and logic-1 probabilities for supervision across each logic gate and memory element, where transition probabilities unveil insights into the circuit’s state transition behaviors and logic-1 probabilities illuminate functional and topological characteristics. Such a nuanced approach allows DeepSeq to adeptly encode the complex dynamics and behaviors of sequential circuits, proving instrumental for downstream applications such as netlist-level power estimation and reliability analysis.

5.2 Representation Learning for Back-End Design

Advancing to the physical design stage, representation learning confronts the geometric and spatial intricacies of the circuit layout. Here, convolutional neural networks (CNNs) and vision Transformers (ViT) are particularly adept at capturing the spatial relationships and critical topology in this phase. The objective is to distill the physical design’s essence into a representation that not only mirrors the layout’s complexities but also yields actionable insights for further optimization and refinement.

The meticulous development of unimodal representations across each design stage knits a rich tapestry of circuit knowledge. Existing studies have explored unimodal learning for prediction of various factors such as routability, IR drop, and lithography hotspots [114], [135], [227]. Although back-end design consists of many design stages with different levels of geometric abstraction, as shown in Fig. 2 b), existing studies mostly focus on individual stages. There are a few key problems yet to be solved before making back-end representation learning practical in real design applications. For easier understanding, we interpret the layout representation learning task by comparing with computer vision tasks on images.

Modern layouts consist of rectilinear shapes with a layer property to represent placement and routing information. These shapes need to follow design rules like minimum width, spacing, area, and so on. Detailed of shapes matter. A layout representation encoder needs to capture the detailed changes in layouts. Besides, each shape in a layout is located at a layer. A layer is like an RGB channel of image, so a straightforward way is to encode shapes at each layer into a channel. However, modern layouts often have more than 20 layers, including metal and via layers, which goes far beyond the typical cases of images.

Unlike images in computer vision which can be resized without losing major information, the dimensions of layouts change with design scales, e.g., 256×256 , 1024×1024 , 4096×4096 , and beyond. Simply resizing a layout like images can lose a lot of information, because individual pixels from layouts of different design scales can correspond to the same geometric resolution defined by manufacturing technologies. A layout representation encoder needs to handle various layout dimensions in a universal way for training on different designs.

A layout of a chip design contains both geometric and topological (i.e., interconnect) information, its representation needs to align with its circuit graph as well. For instance, if two geometric shapes of adjacent layers (e.g., a metal layer and a via layer) are located at the same positions, they are regarded as connected. A layout representation encoder should be able to identify such topological correlation between shapes. Meanwhile, back-end design has many stages, the geometric information in a layout evolves from abstract to concrete, with more and more details, representations at each stage should align with each other as well.

These problems raise challenges in learning general representations for back-end design and also call for the multidimensional alignment that is emblematic of LCMs, which will be detailed in the subsequent section.

6 HARMONIZING REPRESENTATIONS: A MULTIMODAL SYMPHONY

In the realm of circuit design, moving away from unimodal representation learning towards a multimodal integration approach offers a fertile ground for innovation. This strategy seeks to merge the distinct representations from each design phase into a cohesive and unified narrative, ensuring a seamless transition across the design stages. Such integration not only maintains a consistent flow of information but also enriches the design process with enhanced coherence.

6.1 Implementing Multimodal Circuit Alignment

Central to the concept of multimodal circuit learning is the understanding that all design stages, although distinct in form, share a common functional objective. By applying sophisticated feature extraction and alignment techniques, it becomes possible to overcome the semantic disconnects that typically arise in representation learning. This ensures that the original design intent is not only preserved but also accentuated throughout the entire design lifecycle. The adoption of machine learning models, particularly those leveraging scalable self-attention mechanisms and joint embedding spaces, promises to lead the charge towards a more integrated and holistic approach to circuit design.

A potential solution to achieve this alignment involves the use of masked modeling across different modalities. This technique, inspired by successful applications [228] in natural language processing, involves selectively hiding parts of the input data across modalities and then training the model to predict these masked portions. By applying this method across circuit design representations—ranging from natural language specifications, high-level algorithms, and RTL implementations to detailed physical layouts can learn a joint representation that captures the essence of the design process at various abstraction levels. This joint representation is crucial for the model to understand the transition from high-level specifications to detailed implementations, enabling

it to navigate the complexities of circuit design with greater precision and efficiency.

However, addressing the variability in how a high-level design can be mapped to multiple lower-level implementations, each with PPA characteristics, poses a significant challenge. To tackle this, models need to be equipped with the ability to recognize and evaluate the trade-offs associated with different design choices. Integrating reinforcement learning techniques with the multimodal learning framework can provide a solution. By setting the optimization of PPA metrics as the reward function, the model can learn to navigate the space of possible implementations, identifying solutions that best meet the specified criteria. Furthermore, incorporating attention mechanisms can enhance the model’s ability to focus on relevant features across modalities, thereby improving its capacity to predict implementations that not only meet functional requirements but also optimize for PPA objectives. Through these methods, the implementation of multimodal alignment in circuit design can become not just a theoretical concept but a practical tool for advancing the field.

Considering the vast differences between design modalities, aligning them in a single step is a formidable challenge. To address this, we propose a phased approach to multimodal alignment, partitioning the process into three distinct phases: “Spec-HLS-RTL Representation Alignment,” “RTL-Netlist Representation Alignment,” and “Netlist-Layout Representation Alignment.” Since the RTL design contains high-level semantics and netlist is more suitable for aligning with the following backend designs, this strategy employs these two designs as intermediaries, facilitating a more manageable and focused alignment process. By breaking down the alignment into these stages, we can concentrate on specific transitions within the design flow, allowing for a more tailored application of machine learning techniques to each phase. This phased approach not only makes the task of alignment more feasible but also ensures that each stage of the design process is optimally aligned, leading to more coherent and efficient design outcomes. Through careful implementation of this strategy, we aim to bridge the gap between the various design modalities, ultimately fostering a more integrated and seamless circuit design environment.

6.2 Spec-HLS-RTL Representation Alignment

The transition from conceptual specifications to RTL implementation involves a complex journey through natural language specifications, architectural exploration, high-level languages such as SystemC, and hardware description languages like Verilog and VHDL. Leveraging LCMs within a multimodal framework would significantly refine this transformation across different stages, boosting the quality, efficiency, and pace of the design process. LCMs orchestrate a unified representation space that ensures the harmonious integration of front-end design elements across various formats. This unified approach not only streamlines the capture of intricate relationships among circuit components but also accelerates design generation, enhances optimization efforts, and streamlines verification, embodying a leap forward in circuit design methodology.

One of the paramount applications of aligning representations at this stage is the potential substantial improvement in RTL generation. As discussed earlier, existing RTL generation techniques merely fine-tune large language models on HDL code, a process that lacks circuit-specific understanding. With

the aligned representations, we could devise a more sophisticated tokenization strategy for HDL code, paving the way for a deeper understanding and representation of hardware design intricacies. This method transcends the capabilities of existing approaches by generating RTL code that is not only syntactically accurate but also semantically rich, closely aligned with the initial specifications and high-level design intentions. Such advancements promise to elevate the precision and applicability of automatically generated RTL, ensuring designs are both optimized and verifiable from the outset.

Furthermore, the C2RTL verification process benefits significantly from the aligned representation facilitated by LCMs, addressing a pivotal challenge in the transformation from high-level specifications to RTL. This verification phase necessitates a thorough comparison of functional behaviors across natural language specifications, high-level programming languages like C/C++, and RTL implementations. Traditionally, within the EDA framework, this comparison has been both labor-intensive and prone to errors, largely due to the disconnect between the abstract, functional descriptions at the high level and the detailed, hardware-specific implementations at the low level. Bridging this gap between high-level and low-level circuit representations has been a long-standing challenge for the EDA community.

The adoption of LCMs with multimodal alignment into this process introduces a transformative approach to C2RTL verification. By harmonizing the representations of the circuit’s functionality across different stages, these models significantly streamline the verification process. LCMs can identify and resolve discrepancies by meticulously comparing the generated RTL representation against its high-level counterparts. This capability is enhanced by the transformer technology, renowned for its ability to attend selectively to various parts of the input based on their relevance. Such focused attention allows the models to concentrate on areas where discrepancies between the intended functionality and its RTL implementation are most pronounced, offering precise insights and resolutions to designers. This method not only reduces the time and effort traditionally associated with C2RTL verification but also increases the accuracy and reliability of the verification process, marking a significant advancement in ensuring circuit design integrity and performance [182], [229].

6.3 RTL-Netlist Representation Alignment

The RTL-Netlist Representation Alignment stage is crucial for bridging the gap between RTL, AIG netlists, and post-mapping netlists. This alignment paves the way for numerous applications, significantly impacting early PPA estimation, design optimization, and verification processes.

One of the primary benefits of RTL-Netlist alignment is the enhancement of early PPA estimation. By aligning representations from the RTL design phase through to the netlist level, designers can gain insights into the potential power, performance, and area characteristics of their designs much earlier in the development cycle. This early insight allows for more informed decision-making, enabling adjustments to the design that can lead to optimal PPA outcomes. Such proactive adjustments can significantly reduce the need for time-consuming and costly revisions at later stages, streamlining the design process and accelerating time-to-market.

Beyond early PPA estimation, RTL-Netlist alignment also opens the door to more sophisticated design optimization strategies. By having a clear view of how RTL designs translate into netlist implementations, designers can identify and address inefficiencies

at a much deeper level. This insight enables the application of targeted optimizations that can improve the overall quality and efficiency of the design. Moreover, leveraging machine learning models trained on aligned data sets allows for the automation of some optimization tasks, further enhancing the design efficiency and effectiveness.

Finally, the alignment between RTL and netlist representations significantly benefits the verification process. With a comprehensive understanding of how design intentions are manifested in the netlist, verification teams can develop more accurate and efficient testing strategies. This alignment ensures that the verification process is not only faster but also more thorough, reducing the likelihood of errors slipping through to later stages. The ability to detect and address potential issues early on, based on a deep understanding of the aligned representations, is invaluable in maintaining design integrity and reliability.

6.4 Netlist-Layout Representation Alignment

The aspiration to align the circuit netlist with its physical layout is not merely an ambition but a transformative step in EDA. In traditional EDA workflows, the netlist, which represents the logical abstraction of a circuit, and the physical layout, which represents the concrete geometries of the circuit, have been treated as separate entities. However, the increasing complexity of modern integrated circuits has highlighted the need for a tighter integration between the logical and physical domains.

By aligning the netlist with the physical layout, designers can gain a deeper understanding of the relationship between the logical function and physical form of a circuit. This alignment enables a unified perspective of the design, where the logical and physical aspects are considered together, rather than in isolation. It allows designers to analyze and optimize the design from a holistic standpoint, taking into account the impact of physical constraints on logical functionality, and vice versa. Another key benefit of achieving this alignment is the ability to revolutionize the verification process. Traditionally, verification has been performed separately for the logical and physical domains, leading to potential mismatches and design errors. With a multimodal approach that considers both domains simultaneously, designers can detect and resolve issues that arise due to the interaction between the logical and physical aspects of the design. This comprehensive view of the design across stages ensures that the final product meets the desired specifications and performs as expected.

Furthermore, the integration of logical and physical information opens up new possibilities for design optimization. By presenting an integrated picture of the entire design space, designers can explore a wider range of possibilities and make more informed decisions. This comprehensive perspective allows designers to identify and address potential bottlenecks or issues early in the design process, leading to improved quality and efficiency. A specific example of the significance of integrating netlist-layout information is in pre-routing timing prediction. Pre-routing timing prediction aims to accurately evaluate potential sign-off timing violations in the early stages of the design process, reducing design cycles and avoiding costly iterations. Traditionally, pre-routing timing evaluation methods, such as static timing analysis, have primarily focused on netlist information, which represents the interconnections between cells in a design. However, these methods often overlook the crucial role that layout information plays in timing prediction. As most timing optimization techniques

require space to insert or resize gates, the circuit layout that reflects spatial information has a large impact on sign-off timing performance. Neglecting layout information can lead to inaccurate timing predictions and sub-optimal design decisions. Through netlist-layout representation alignment, LCM can provide more accurate estimates of sign-off timing performance. This enables designers to identify and address timing issues early in the design process, reducing the likelihood of sign-off violations and the need for time-consuming iterations.

In summary, the evolution towards a multimodal symphony in circuit design represents not just a technical advancement, but a reimagining of how design processes can be optimized for efficiency, innovation, and coherence. The potential for such an approach to revolutionize the field lies in its ability to harmonize disparate data types and design stages into a single, unified framework, paving the way for breakthroughs in design methodology and implementation.

7 PIONEERING LCM APPLICATIONS

While extensive empirical data are yet to be available, the potential applications of LCMs can be vividly illustrated through hypothetical scenarios and conceptual frameworks. The narrative examples presented in this section serve to bridge the gap between abstract concepts and tangible applications, offering a glimpse into the transformative impact LCMs could have on the EDA field.

7.1 Circuit Learning for SAT

The Boolean Satisfiability (SAT) problem identifies if there exists at least one assignment that makes a given Boolean formula to be *True*. SAT problem acts as a fundamental problem in many areas, especially in the EDA fields, such as logic equivalence checking, model checking, and testing. Over the past few decades, the SAT community has advocated adopting the conjunctive normal form (CNF) as the *de facto* standard format for problem instances and developed numerous advanced CNF-based SAT solvers [231], [232]. However, the efficacy of CNF-based solvers recently encounter bottlenecks in solving hard SAT problems, prompting past research to explore circuit-based solvers or strategies as a potential breakthrough. In this section, we aim to demonstrate the impact of the large circuit model on SAT solving.

First, the circuit netlist serves as a natural representation of SAT problems within the field of EDA and also can be efficiently derived from various combinatorial optimization problems. Inspired by an early endeavor [230], a circuit-based universally efficient reformulation mechanism could significantly reduce the complexity before solving these problems. The LCMs, especially the uni-modal netlist encoders, are capable of capturing the structural features across various netlist distributions. Exploiting this knowledge allows for the exploration of a global transformation flow based on reinforcement learning, ultimately minimizing the overall complexity of the solving process.

Table 1 shows our preliminary results when applying the netlist encoder to accelerate SAT solving for industrial logic equivalence checking cases I1-I5. In the Baseline setting, the instances are solved directly using the Kissat solver [232]. We denote the RL agent runtime, transformation time, and solving time as \mathcal{T}_{agent} , \mathcal{T}_{trans} and \mathcal{T}_{solve} , respectively, measured in *seconds*. The overall runtime, which sums up all three components, is denoted as \mathcal{T}_{all} in *seconds*. Additionally, we list the number of variables (# Vars) and

TABLE 1: Solving time comparison between Ours and [230] on LEC cases.

Case	Baseline			[230]					Ours								
	# Vars	# Clas	T_{solve}	# Vars	# Clauses	T_{trans}	T_{solve}	T_{all}	Red.	# Vars	# Clas	T_{agent}	T_{trans}	T_{solve}	T_{all}	Red.	Red.*
I1	42,069	105,711	322.46	5,616	54,529	5.31	51.49	56.80	82.39%	3,160	31,281	9.27	5.62	4.43	19.26	94.03%	66.08%
I2	44,949	112,954	708.97	6,052	60,573	5.61	147.85	153.46	78.35%	4,112	41,873	9.81	6.12	4.41	20.81	97.07%	86.44%
I3	42,038	105,629	531.94	5,612	54,825	5.21	109.89	115.10	78.36%	3,849	37,329	8.37	5.61	2.91	17.56	96.70%	84.74%
I4	37,275	93,678	289.89	5,038	49,805	4.61	90.05	94.66	67.35%	3,478	34,013	7.32	5.11	2.50	15.01	94.82%	84.14%
I5	30,087	75,537	172.79	4,006	38,069	3.91	38.77	42.67	75.30%	2,311	22,473	4.78	4.31	1.10	10.50	93.92%	75.39%
Avg.			405.21					92.54	77.16%						16.63	95.90%	82.03%

clauses (# Clas), the reduction in T_{all} compared to Baseline (**Red.**) and compared to [230] (**Red.***). The solving time is reduced by 96.14% and 82.03% on average, respectively.

Second, gate-level embeddings proficiently encapsulate the logical correlations among gates within a circuit netlist, ensuring that gates sharing functional similarities are closely aligned within the embedding space. This alignment allows for a precise representation of logical connections between variables in the SAT formulation. By integrating these gate-level embeddings, we can highlight and utilize the discerned correlations to expedite the SAT-solving process. This is achieved by embedding these correlations as additional constraints in the initial SAT problem instances, thereby enhancing the solver’s efficiency.

Third, traditional heuristic strategies (e.g., branching heuristics) predominantly depend on the correlation between variables in CNF representations, which cannot preserve the circuit’s topological structure. Recent advancements, such as [17], showcase the effectiveness of a unimodal netlist encoder in capturing the intricate gate-level logic correlations within circuit netlists.

Building upon the above, the LCMs excel in identifying gate-level functional relationships within circuit netlists based on the unimodal netlist encoders. By harnessing the power of LCMs, new and efficient circuit-based SAT-solving strategies can be developed, ultimately improving the overall performance and effectiveness of heuristic designs.

7.2 LCM for Logic Synthesis

Logic synthesis stands at the crossroads of multiple representations and sophisticated algorithms, such as truth tables, sum-of-products, binary decision diagrams (BDDs), and directed acyclic graphs (DAGs), with none asserting complete dominance. This diversity underscores a fundamental challenge: selecting and optimizing the most effective representation for a logic function. Herein lies the transformative potential of LCM. By learning and internally representing the same logic function across diverse formats, LCMs exhibit unparalleled adaptability. Their deep understanding of intricate relationships and optimization pathways within logic synthesis allows for a flexible approach to representing complex logic functions. This adaptability becomes instrumental in handling multifaceted inputs and expressions of logic, showcasing LCMs’ capability to revolutionize the representation and optimization of logic functions in a way previously unattainable.

As we venture into the realm of nanometer-scale technologies, the significance of technology-independent optimizations becomes increasingly pronounced, focusing on metrics like the number of literals and logic depth in DAGs for area and delay evaluation. Marrying these optimization strategies with the physical realities of the technology landscape introduces a new layer of complexity. LCMs are poised to address this challenge head-on by predicting physical

characteristics such as timing, area, and power more accurately. Integrating physical awareness, LCMs offer a groundbreaking tool in logic optimization, enabling designers to base their decisions on a nuanced understanding of circuit behavior. This foresight not only refines optimization strategies but also facilitates superior PPA trade-offs, marking a leap forward in logic synthesis.

The crux of technology mapping, especially in FPGA and ASIC design, lies in navigating the constraints of heterogeneous logic blocks, interconnect resources, and optimal cell selection while balancing the PPA trade-offs. Addressing structural bias during technology mapping demands meticulous algorithmic strategies. LCMs herald a new era in technology mapping by leveraging their ability to learn from diverse logic representations and adapt mapping strategies to the nuanced requirements of the input data and technological constraints. Their versatility in overcoming structural biases through contextually aware mappings, coupled with the iterative feedback loop and physical information integration, offers tailored insights for refining mapping strategies. LCMs’ scalability further underscores their effectiveness in managing complex circuit designs, presenting a compelling case for overcoming longstanding challenges in technology mapping.

In essence, the conceptual application of LCMs in logic synthesis promises a shift towards more efficient, accurate, and adaptable design processes, positioning them as the cornerstone for the next generation of circuit design methodologies.

7.3 LCM for Equivalence Checking

Equivalence checking stands as a critical verification step in digital circuit design, ensuring that functionality is preserved through synthesis or manual modifications. Traditional methods, while reliable, struggle with scalability in the face of increasingly dense designs and the complex optimizations required to meet PPA goals. Here, LCMs emerge as a transformative solution, offering a paradigm shift towards interactive equivalence checking that enhances the efficiency and effectiveness of the process.

LCMs have the unique potential to revolutionize this domain by enabling an end-to-end interactive equivalence checking process. This approach is particularly beneficial for ECO optimizations and custom design styles, where the goal extends beyond functional equivalence to include high-quality design modifications. Leveraging their deep understanding of circuit semantics, LCMs can offer insightful recommendations for design adjustments and patches during the interactive ECO phase. Drawing from extensive training on diverse circuit data, LCMs can identify underlying patterns and rules of successful designs, suggesting targeted modifications to resolve detected discrepancies. These suggestions are not only based on historical success but are also ranked according to their anticipated impact on PPA, empowering designers with informed choices that align with their specific objectives.

Furthermore, the iterative nature of LCMs means that these recommendations can be refined based on designer feedback, creating an efficient feedback loop that streamlines the equivalence checking and modification process. This iterative engagement not only accelerates the identification of viable design solutions but also enhances the overall quality of the final design.

In addition to transforming equivalence checking into an interactive dialogue, LCMs hold promise for augmenting existing equivalence checking systems. Traditional algorithms have exploited the empirical distribution of circuit designs, wherein current practices include: 1) partitioning and selecting fine-grained proof strategies [233], 2) adapting various encodings from a problem instance to a canonical solver instance [234], and 3) employing design-specific equivalence checking strategies (e.g., for multipliers [235]). These solutions remain limited by the need for hand-crafted heuristics and specialized strategies. For instance, LCMs, with their ability to automatically understand design intent and manage the distribution of design data, can act as a neural backbone for these systems. They can manage various heuristics in formal solvers or function as a neural scheduler for task distribution, significantly enhancing the performance and efficiency of equivalence checking processes.

This dual approach—transforming equivalence checking into an interactive process and augmenting existing systems—highlights the pioneering potential of LCMs. By leveraging the power of LCMs, designers can navigate the complexities of modern circuit verification with greater ease and precision, promising to elevate the verification process to new heights of efficiency and effectiveness.

7.4 LCM for Physical Design

Physical design is the stage that converts the logical representations of a circuit into the physical representations. In this stage, a physical layout is generated by partitioning, floorplanning, placement, and routing. This process requires solving many NP-hard combinatorial optimization problems and is extremely complex and time-consuming. As the scale of an electronic design keeps increasing and the feature size keeps shrinking, traditional approaches to physical design face serious challenges. LCMs, on the other hand, could provide new perspectives on processing the physical representations of an electronic design and even new methodologies in dealing with these tricky combinatorial optimization problems.

A trained LCM could offer guidance to placement and routing for wirelength, routability, and timing optimization. Consider the placement optimization process, where traditional methods have leveraged unimodal information for guidance, from gradient prediction [236] to routing congestion forecasting [114]. Common practices involve transforming layout features into image-like data for machine learning model predictions, often employing vision-based models like CNNs and Vision Transformers. Yet, this approach may overlook crucial interconnect information, given the challenges vision-based methods face in preserving topological details alongside spatial relationships. Recent explorations into multimodal representations for physical design, however, illuminate a promising path forward. Studies like LHNN [237] introduce dual GNNs to capture both topological (circuit interconnections) and spatial relationships, merging these insights in latent space. Similarly, Lay-Net [118] proposes substituting the GNN with CNN for spatial analysis, capitalizing on the superior spatial awareness of vision-based methods. Despite these advancements, LCMs have the capacity to move beyond merely integrating topological

and spatial relationships. By aligning with additional modalities, designers gain the unprecedented ability to pinpoint layout hotspots at earlier design stages and implement preemptive countermeasures. This proactive approach facilitated by LCMs allows for the early identification of potential issues related to timing, power, and thermal management, enabling adjustments before they escalate into more significant challenges.

To sum up, LCMs could learn the underlying characteristics of a physical representation and reveal new directions for design and optimization. More excitingly, they have the potential to serve as the foundation of new learning-based heuristics and revolutionize the traditional way of physical design, eliminating the burden of constantly designing new algorithms.

8 TAILORING LCMs FOR SPECIALIZED CIRCUITS

Exploring specialized circuit domains reveals a diverse array of unique designs that extend beyond the standard digital circuits typically encountered in EDA workflows. Standard cell designs, datapath circuits, memory macros, and analog circuits possess distinct characteristics that necessitate custom approaches. The expansion of LCMs into these specialized arenas heralds a promising enhancement for design efficiency and optimization.

8.1 Large Circuit Models for Standard Cells

Standard cells form the fundamental building blocks of digital designs, comprising basic logic gates and complex combinational functions. Their design is critical for the overall performance and power efficiency of the chip. LCMs in this domain could leverage generative models to propose new cell architectures that optimize for a variety of constraints, including power, performance, area, and even novel objectives like robustness to process variations. Furthermore, these models could predict the impact of cell design changes on the higher levels of the design hierarchy, enabling a holistic approach to optimization.

For the front-end design of standard cells, LCM can be employed for library pruning and cell characterization. The requirements for standard cell libraries differ between high-performance circuit design and low-power circuit design [238], [239]. Historically, designers have often relied on experience and extensive simulations to select a subset for a new cell library. LCM can leverage existing selection experiences to better choose suitable cells for the specific design scenario. Additionally, it can leverage generative models to continuously explore new topological architectures, subsequently refining the generation process based on SPICE simulation results as feedback for continual improvement. Characterization is the most time-consuming step in standard cell design, requiring extensive SPICE simulations to generate liberty libraries. However, the significance varies across different PVT corners and standard cells. Therefore, accuracy-aware supervised learning can enhance the overall precision of libraries while reducing runtime by prioritizing the importance of different corners and cells [240], [241].

8.2 Large Circuit Models for Datapath Circuits

Datapath circuits, essential for performing arithmetic and logical operations within microarchitectures, stand at the core of performance-critical computing. These components notably benefit from bit-level optimization, necessitating a detailed focus on timing and power constraints.

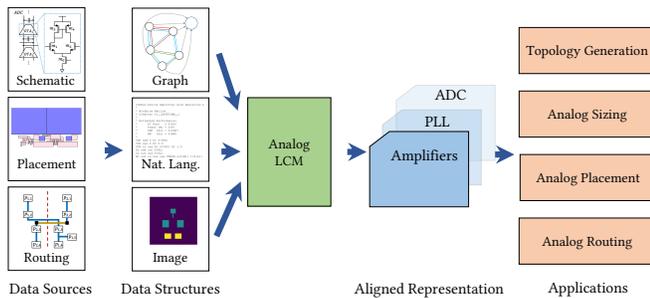


Fig. 7: Overview of large circuit models for analog EDA.

In datapath design, the optimization of a myriad of parameters is crucial for balancing PPA effectively. Acting as surrogate models during design space exploration, LCMs can significantly streamline the optimization process. LCMs specifically tailored for datapath circuits offer a promising approach by employing specialized architectures adept at understanding the complexities of arithmetic operations. This enables them to enhance logical efficiency while optimizing the physical layout. Through training on diverse datasets, encompassing both synthetic and real-world datapath designs, LCMs pave the way for exploring innovative datapath configurations that extend beyond traditional design methodologies.

Specifically, LCMs bring a new dimension to datapath design evaluation, offering a comprehensive and accurate analysis that transcends traditional limitations. Traditional evaluation methods often rely on a constrained set of benchmarks, limiting the scope of assessment. In contrast, LCMs draw upon a broad and profound understanding of target applications, facilitating a more extensive evaluation of datapath designs. This broad perspective enables a “shift-left” in the evaluation process, providing early and insightful assessments that encompass not only architectural considerations but also subsequent stages like placement and routing. Such a shift enhances the overall efficiency and effectiveness of the evaluation process.

Moreover, LCMs’ deep domain knowledge in both the logical functions and physical implementations of datapath circuits allows them to automatically pinpoint optimization bottlenecks. This capability not only accelerates the design optimization cycle but also furnishes designers with critical insights for further enhancements. By integrating LCMs into the datapath design process, engineers can achieve a level of optimization and efficiency previously unattainable, heralding a new era in the evolution of datapath circuits and their implementation in modern microarchitectures.

In summary, LCMs’ detailed grasp of datapath complexities allows them to offer strategic recommendations that go beyond parameter adjustments, influencing the architectural framework of the circuit’s RTL design. The ultimate goal is to utilize LCMs for the automated generation of circuit datapaths, tailored to specific Process Design Kits (PDKs) and targeted software applications, thereby revolutionizing the design process.

8.3 Large Circuit Models for Analog Circuits

Analog EDA shares similarities and differences with digital EDA. Like digital workflows, analog EDA encompasses front-end netlist design and back-end layout design. Analog LCMs also demand holistic solutions that span different design flow stages. Conversely, analog circuits exhibit distinct data structures and performance

evaluations compared to their digital counterparts, which are primarily logic-driven. In analog circuits, device-level topology and physical implementation are crucial. The sub-structure of transistors, capacitors, and resistors determines circuit functionality, making the detailed graph structures (such as network motifs) and device parameters essential for capture by analog LCMs. Moreover, analog circuits involve various types and evaluations, with different performance evaluations requiring specific circuit implementations.

Analog circuit design is an art that melds intricate knowledge of device physics with the subtleties of the intended application. LCMs for analog circuits must capture this depth of knowledge, translating it into models that can navigate the analog design space with its continuous variables and stringent performance metrics. These models could predict analog behavior from device-level up to system-level specifications, assist in layout generation, and automate the tedious tuning process of analog parameters. By doing so, LCMs could drastically reduce the design time and enhance the performance of analog circuits, which remain a bottleneck in mixed-signal chip design.

TAG [195] represents an early effort to develop a circuit representation model for analog EDA. It introduces a netlist embedding mechanism and a “pretrain-then-finetune” strategy to apply embedding vectors across various applications. However, it lacks a unified, aligned representation across all design stages, with its effectiveness constrained by the initial pretraining target, layout distance. Its potential applications are somewhat limited compared to a comprehensive LCM for analog circuits.

Fig. 7 presents our vision for future analog large circuit models. These models are inherently multimodal, capable of processing various data structures from different design stages. Text and graph structures can represent a netlist, while images may be used for layout designs. An Analog LCM converts these inputs into vectors, mapping the designs to a unified embedding space. The generated circuit embedding vectors can then support various downstream tasks across different circuit types (such as amplifiers, PLLs, and ADCs), catering to a range of applications from topology design to routing.

9 CHALLENGES AND OPPORTUNITIES: THE DUAL EDGES

Embarking on the quest for LCMs unveils a realm filled with both challenges and opportunities. The journey is strewn with hurdles like data scarcity, scalability issues, and interoperability with existing EDA tools, yet each challenge surmounted paves the way for uncharted opportunities.

9.1 Data Issues

Data scarcity stands out as a critical hurdle, given the dependency of LCMs on extensive, high-quality datasets for training. The realm of circuit design, particularly at the granularity required for effective LCM operation, suffers from a lack of publicly available data, posing risks of overfitting and undermining the models’ generalization capabilities. Tackling this issue head-on, we introduce three possible solutions.

First, innovative data augmentation techniques emerge as a key solution. For instance, equivalent circuits can be generated through systematic circuit transformations, effectively expanding the dataset without the need for additional real-world data. This approach not only enhances the diversity of training material but

also deepens the model’s understanding of circuit variability and design principles.

Second, on the synthetic data front, leveraging LLMs to generate realistic RTL code presents an exciting opportunity. This strategy involves using LLMs’ advanced generative capabilities to create new RTL designs, which can then serve both as additional training data and as benchmarks for further refining the RTL generative models themselves. This creates a self-reinforcing loop where LCMs continually improve through iterative training on both real and synthetically generated data. Such a mechanism not only addresses the issue of data scarcity but also contributes to the evolution of more sophisticated and capable generative models, marking a significant leap towards fully realizing the transformative potential of LCMs in the EDA landscape.

Third, the development of community-driven platforms for data sharing and collaboration could significantly alleviate the scarcity issue. By fostering an ecosystem where academia and industry share circuit data and design challenges, the field can collectively advance the state of LCM research, ensuring a diverse and comprehensive dataset that mirrors the multifaceted nature of electronic design automation.

In essence, while data scarcity presents a formidable challenge, it also opens the door to a range of inventive strategies that not only address the immediate issue but also enrich the EDA domain. Through collaborative efforts, technological advancements, and a commitment to innovation, the potential of LCMs in revolutionizing circuit design remains within reach.

9.2 Scalability and Interoperability

Scalability emerges as a pivotal challenge in the realm of LCMs, especially as we delve into complex, vast-scale circuit designs that define the next generation of electronic devices. The quest for scalability is not just about accommodating larger designs but also about enhancing computational efficiency and sophistication in model architecture. This involves pioneering hierarchical modeling techniques that can intuitively decompose complex designs into manageable submodules, algorithmic optimizations that streamline model training and inference, and the implementation of parallel processing strategies to distribute computational workload effectively. Each of these advancements contributes to a robust foundation, equipping LCMs to tackle increasingly ambitious design projects while maintaining precision and efficiency.

Moreover, as LCMs grow in complexity and capability, ensuring their interoperability with the existing mosaic of EDA tools becomes paramount. The modern circuit design ecosystem is a tapestry of specialized design flows, tools, scripts, libraries, and technologies, each contributing to various stages of the design process. Bridging the gap between the innovative potential of LCMs and the established practices of current EDA workflows necessitates a concerted effort for deeper collaboration between the AI research community and EDA professionals. Such collaboration aims to weave AI-driven methodologies seamlessly into the fabric of EDA, enhancing tool compatibility, data exchange protocols, and user interfaces. This symbiotic relationship stands to not only streamline the integration of LCMs into existing design pipelines but also to catalyze the mutual evolution of both AI technologies and EDA tools and methodologies, heralding a new era of design automation that is both more intelligent and intuitive.

9.3 New Opportunities

Beyond merely enhancing existing EDA tools, LCMs present the exciting prospect of birthing entirely new categories of EDA tools, ones that could fundamentally alter how design, verification, and optimization are approached.

One of the most promising opportunities presented by LCMs is the ability to conduct early-stage, precise PPA estimation. Traditionally, accurate PPA metrics could only be determined after substantial design progress, often at the post-synthesis or post-layout stages. LCMs, however, can predict these critical metrics much earlier in the design process, leveraging aligned representations among modalities. This capability allows for more informed decision-making at the outset of a project, guiding design choices in alignment with PPA objectives and significantly accelerating the optimization cycle. Early-stage PPA estimation not only enhances design efficiency but also enables a more agile response to evolving design requirements and constraints.

LCMs also enable a paradigm shift towards cross-stage verification, a holistic approach that transcends the conventional, compartmentalized verification processes. Traditional EDA methodologies often treat verification as a stage-specific task, siloed within the design flow. However, LCMs, with their comprehensive understanding of circuit knowledge across various stages, facilitate a unified verification framework. This cross-stage verification can detect inconsistencies and errors early in the design process, reducing the iterative cycles typically required to rectify such issues. By leveraging the predictive power of LCMs, designers can ensure coherence and fidelity from the initial specifications to the final physical layouts, significantly streamlining the verification process.

Moreover, LCMs unlock the potential for generative design, particularly for well-structured circuits such as datapath units. Datapath units, with their regular structures and predictable performance metrics, are ideal candidates for LCM-driven generative design approaches. LCMs can generate optimal circuit configurations that meet specified criteria, exploring a vast design space that might be infeasible for human designers to cover comprehensively. This generative capability can lead to innovative circuit designs that optimize PPA metrics, potentially discovering novel architectural solutions that traditional design methodologies might overlook. Furthermore, generative design facilitated by LCMs can automate aspects of the design process for these structured circuits, reducing manual effort and enabling a focus on higher-level design challenges.

Finally, the synergy between large language models and LCMs presents a particularly promising area of exploration. LLMs, with their advanced natural language processing capabilities, can serve as intuitive, conversational interfaces for designers, translating high-level design specifications into actionable insights and suggestions; While the LCMs, with their deep understanding of circuitry and design principles, can analyze and optimize the granular details of the netlist, ensuring that the final design aligns with the desired performance, power, and area constraints. This collaborative interaction between LLMs and LCMs allows for a seamless transition from abstract design concepts to concrete, optimized circuit representations. Bridging the gap between high-level design intent and detailed technical execution, this synergy enables a more holistic and integrated approach to circuit design.

In summary, the development of LCMs is fraught with challenges, yet each obstacle surmounted brings the EDA community one step closer to realizing the full potential of these innovative

models. The promise of LCMs to significantly streamline the design process, elevate design quality, and accelerate the development of cutting-edge electronic systems highlights the critical importance of addressing these challenges.

10 CONCLUSION

As we navigate the evolving landscape of AI-driven EDA, the potential of large circuit models emerges as a beacon of innovation, promising to redefine the paradigms of circuit design and analysis.

Specifically, we advocate for a paradigm shift from task-oriented AI4EDA methodologies to more integrated, AI-native foundation models. LCMs stand at the crossroads of this transition, offering a holistic representation that encapsulates the multifaceted aspects of circuit design—from logical structuring to physical realization. The promise of LCMs lies in their ability to harness deep learning for capturing the intricate dependencies and characteristics of large-scale circuit netlists, thereby facilitating more efficient, accurate, and innovative design strategies.

Looking ahead, the journey toward fully realizing the potential of LCMs is laden with a vast array of research problems waiting to be addressed. From the refinement of representation learning techniques to accommodate the unique circuit characteristics at various design stages, to the development of scalable, effective alignment models capable of interpreting and optimizing complex netlists, the field is ripe for exploration.

In conclusion, the dawn of AI-native EDA heralded by LCMs presents a transformative vision for the future of circuit design and analysis. By embracing this new frontier, we stand to unlock unprecedented levels of efficiency, creativity, and precision in the creation of the next generation of electronic devices.

REFERENCES

- [1] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, “On the opportunities and risks of foundation models,” *arXiv preprint arXiv:2108.07258*, 2021.
- [2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds., 2019, pp. 4171–4186.
- [3] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A robustly optimized BERT pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019.
- [4] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.
- [5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 1877–1901, 2020.
- [6] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International Conference on Machine Learning (ICML)*, 2020, pp. 1597–1607.
- [7] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9729–9738.
- [8] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 16 000–16 009.
- [9] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, “Learning transferable visual models from natural language supervision,” in *International Conference on Machine Learning (ICML)*, 2021, pp. 8748–8763.
- [10] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, “Zero-shot text-to-image generation,” in *International Conference on Machine Learning (ICML)*, 2021, pp. 8821–8831.
- [11] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 10 684–10 695.
- [12] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, “Segment anything,” *arXiv preprint arXiv:2304.02643*, 2023.
- [13] Z. Yang, L. Li, K. Lin, J. Wang, C.-C. Lin, Z. Liu, and L. Wang, “The dawn of LLMs: Preliminary explorations with GPT-4V(ision),” *arXiv preprint arXiv:2309.17421*, vol. 9, no. 1, 2023.
- [14] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth *et al.*, “Gemini: A family of highly capable multimodal models,” *arXiv preprint arXiv:2312.11805*, 2023.
- [15] M. Rapp, H. Amrouch, Y. Lin, B. Yu, D. Z. Pan, M. Wolf, and J. Henkel, “MLCAD: A survey of research in machine learning for CAD keynote paper,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 10, pp. 3162–3181, 2022.
- [16] M. Li, S. Khan, Z. Shi, N. Wang, Y. Huang, and Q. Xu, “DeepGate: Learning neural representations of logic gates,” in *ACM/IEEE Design Automation Conference*, 2022, pp. 667–672.
- [17] 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 *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–9.
- [18] “OpenCores.” [Online]. Available: <http://opencores.org/>
- [19] “XiangShan RISC-V processor.” [Online]. Available: <https://github.com/OpenXiangShan/XiangShan>
- [20] K. Asanović, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz, S. Karandikar, B. Keller, D. Kim, J. Koenig, Y. Lee, E. Love, M. Maas, A. Magyar, H. Mao, M. Moreto, A. Ou, D. A. Patterson, B. Richards, C. Schmidt, S. Twigg, H. Vo, and A. Waterman, “The rocket chip generator,” EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17, Apr 2016.
- [21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
- [22] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, “Graph neural networks: A review of methods and applications,” *AI open*, vol. 1, pp. 57–81, 2020.
- [23] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, “Multimodal machine learning: A survey and taxonomy,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.
- [24] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le *et al.*, “Program synthesis with large language models,” *arXiv preprint arXiv:2108.07732*, 2021.
- [25] C.-C. Lin, K.-C. Chen, S.-C. Chang, M. Marek-Sadowska, and K.-T. Cheng, “Logic synthesis for engineering change,” in *ACM/IEEE Design Automation Conference*, 1995, pp. 647–652.
- [26] G. De Micheli, “Chip challenge,” *IEEE Solid-State Circuits Magazine*, vol. 2, no. 4, pp. 22–26, 2010.
- [27] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Avizienis, J. Wawrzyniec, and K. Asanović, “Chisel: Constructing Hardware in a Scala Embedded Language,” in *ACM/IEEE Design Automation Conference (DAC)*, 2012, pp. 1216–1225.
- [28] S. C. Johnson, *Lint, a C program checker*. Bell Telephone Laboratories Murray Hill, 1977.
- [29] C. I. C. Marquez, M. Strum, and W. J. Chau, “Formal equivalence checking between high-level and RTL hardware designs,” in *2013 14th Latin American Test Workshop-LATW*. IEEE, 2013, pp. 1–6.
- [30] R. Mukherjee, M. Purandare, R. Polig, and D. Kroening, “Formal techniques for effective co-verification of hardware/software co-designs,” in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, pp. 1–6.
- [31] Synopsys. (2024) Vc formal datapath validation. [Online]. Available: <https://www.synopsys.com/verification/static-and-formal-verification/vc-formal/vc-formal-datapath-validation.html>
- [32] A. Koelbl, R. Jacoby, H. Jain, and C. Pixley, “Solver technology for system-level to RTL equivalence checking,” in *2009 Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2009, pp. 196–201.
- [33] B.-Y. Huang, H. Zhang, P. Subramanyan, Y. Vazel, A. Gupta, and S. Malik, “Instruction-level abstraction (ILA) a uniform specification for system-on-chip (SoC) verification,” *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 1, pp. 1–24, 2018.

- [34] A. Mishchenko, S. Chatterjee, R. Brayton, and N. Een, "Improvements to combinational equivalence checking," in *Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design*, 2006, pp. 836–843.
- [35] J. Baumgartner, H. Mony, V. Paruthi, R. Kanzelman, and G. Janssen, "Scalable sequential equivalence checking across arbitrary design transformations," in *2006 International Conference on Computer Design*. IEEE, 2006, pp. 259–266.
- [36] Z. Chen, X. Zhang, Y. Qian, Q. Xu, and S. Cai, "Integrating exact simulation into sweeping for datapath combinational equivalence checking," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023, pp. 1–9.
- [37] Y.-Y. Dai, K.-Y. Khoo, and R. K. Brayton, "Sequential equivalence checking of clock-gated circuits," in *Proceedings of the 52nd Annual Design Automation Conference*, 2015, pp. 1–6.
- [38] C. J. Alpert, D. P. Mehta, and S. S. Sapatnekar, *Handbook of algorithms for physical design automation*. CRC press, 2008.
- [39] X. Li, Z. Huang, S. Tao, Z. Huang, C. Zhuang, H. Wang, Y. Li, Y. Qiu, G. Luo, H. Li, H. Shen, M. Chen, D. Bu, W. Zhu, Y. Cai, X. Xiong, Y. Jiang, Y. Heng, P. Zhang, B. Yu, B. Xie, and Y. Bao, "iEDA: An open-source infrastructure of EDA," in *Asia and South Pacific Design Automation Conference (ASPDAC)*. IEEE, 2024.
- [40] Y.-L. Li, S.-T. Lin, S. Nishizawa, H.-Y. Su, M.-J. Fong, O. Chen, and H. Onodera, "NCTUcell: A DDA-aware cell library generator for FinFET structure with implicitly adjustable grid map," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.
- [41] C.-K. Cheng, C.-T. Ho, D. Lee, and D. Park, "A routability-driven complimentary-FET (CFET) standard cell synthesis framework using SMT," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020, pp. 1–8.
- [42] D. Park, D. Lee, I. Kang, S. Gao, B. Lin, and C.-K. Cheng, "SP&R: Simultaneous placement and routing framework for standard cell synthesis in sub-7nm," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2020, pp. 345–350.
- [43] S. Choi, J. Jung, A. B. Kahng, M. Kim, C.-H. Park, B. Pramanik, and D. Yoon, "PROBE3.0: A systematic framework for design-technology pathfinding with improved design enablement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023. [Online]. Available: <https://doi.org/10.1109/TCAD.2023.3334591>
- [44] A. Beaumont-Smith and C.-C. Lim, "Parallel prefix adder design," in *Proceedings 15th IEEE Symposium on Computer Arithmetic. ARITH-15 2001*. IEEE, 2001, pp. 218–225.
- [45] S. Rakesh and K. V. Grace, "A comprehensive review on the vlsi design performance of different parallel prefix adders," *Materials Today: Proceedings*, vol. 11, pp. 1001–1009, 2019.
- [46] N. P. Jouppe, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.
- [47] H. Chen, M. Liu, X. Tang, K. Zhu, N. Sun, and D. Z. Pan, "Challenges and opportunities toward fully automated analog layout design," *Journal of Semiconductors*, vol. 41, no. 20070021, p. 111407, 2020.
- [48] Z. Zhao and L. Zhang, "An automated topology synthesis framework for analog integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 39, no. 12, pp. 4325–4337, 2020.
- [49] W. Lyu, P. Xue, F. Yang, C. Yan, Z. Hong, X. Zeng, and D. Zhou, "An efficient Bayesian optimization approach for automated optimization of analog circuits," *IEEE Transactions on Circuits and Systems I*, vol. 65, no. 6, pp. 1954–1967, 2018.
- [50] K. Zhu, H. Chen, M. Liu, X. Tang, N. Sun, and D. Z. Pan, "Effective analog/mixed-signal circuit placement considering system signal flow," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.
- [51] H. Ren and J. Hu, *Machine Learning Applications in Electronic Design Automation*. Springer, 2022.
- [52] P. Joseph, K. Vaswani, and M. J. Thazhuthaveetil, "Construction and use of linear regression models for processor performance analysis," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2006.
- [53] C. Mendis, A. Renda, S. Amarasinghe, and M. Carbin, "Ithema: Accurate, portable and fast basic block throughput estimation using deep neural networks," in *International Conference on Machine Learning (ICML)*, 2019.
- [54] J. Zhai, C. Bai, B. Zhu, Y. Cai, Q. Zhou, and B. Yu, "McPAT-Calib: A microarchitecture power modeling framework for modern CPUs," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021, pp. 1–9.
- [55] Q. Zhang, S. Li, G. Zhou, J. Pan, C.-C. Chang, Y. Chen, and Z. Xie, "PANDA: Architecture-level power evaluation by unifying analytical and machine learning solutions," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2023, pp. 01–09.
- [56] C. Bai, Q. Sun, J. Zhai, Y. Ma, B. Yu, and M. D. Wong, "BOOM-Explorer: RISC-V BOOM microarchitecture design space exploration framework," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2021.
- [57] N. Ardalani, C. Lestourgeon, K. Sankaralingam, and X. Zhu, "Cross-architecture performance prediction (XAPP) using CPU code to predict GPU performance," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2015.
- [58] G. Wu, J. L. Greathouse, A. Lyashevsky, N. Jayasena, and D. Chiou, "GPGPU performance and power estimation using machine learning," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2015.
- [59] Z. Qian, D.-C. Juan, P. Bogdan, C.-Y. Tsui, D. Marculescu, and R. Marculescu, "SVR-NoC: A performance analysis tool for network-on-chips using learning-based support vector regression model," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2013, pp. 354–357.
- [60] Z. Shi, X. Huang, A. Jain, and C. Lin, "Applying deep learning to the cache replacement problem," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2019, pp. 413–425.
- [61] R. Bera, K. Kanellopoulos, A. Nori, T. Shahroodi, S. Subramoney, and O. Mutlu, "Pythia: A customizable hardware prefetching framework using online reinforcement learning," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2021.
- [62] S. Lu, R. Tessier, and W. Burleson, "Reinforcement learning for thermal-aware many-core task allocation," in *Great Lakes Symposium on VLSI*, 2015.
- [63] N. AbouGhazaleh, A. Ferreira, C. Rusu, R. Xu, F. Liberato, B. Childers, D. Mosse, and R. Melhem, "Integrated CPU and L2 cache voltage scaling using machine learning," in *ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES)*, 2007.
- [64] C. Dubach, T. M. Jones, E. V. Bonilla, and M. F. O'Boyle, "A predictive model for dynamic microarchitectural adaptivity control," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2010, pp. 485–496.
- [65] S.-C. Kao, G. Jeong, and T. Krishna, "ConfuciusX: Autonomous Hardware Resource Assignment for DNN Accelerators using Reinforcement Learning," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2020, pp. 622–636.
- [66] S. Dai, Y. Zhou, H. Zhang, E. Ustun, E. F. Young, and Z. Zhang, "Fast and accurate estimation of quality of results in high-level synthesis with machine learning," in *Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2018.
- [67] H. M. Makrani, F. Farahmand, H. Sayadi, S. Bondi, S. M. P. Dinakarrao, H. Homayoun, and S. Rafatirad, "Pyramid: Machine learning framework to estimate the optimal timing and resource usage of a high-level synthesis design," in *International Conference on Field-Programmable Logic and Applications (FPL)*, 2019.
- [68] E. Ustun, C. Deng, D. Pal, Z. Li, and Z. Zhang, "Accurate operation delay prediction for FPGA HLS using graph neural networks," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020, pp. 1–9.
- [69] J. Zhao, T. Liang, S. Sinha, and W. Zhang, "Machine learning based routing congestion prediction in fpga high-level synthesis," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 1130–1135.
- [70] Z. Lin, Z. Yuan, J. Zhao, W. Zhang, H. Wang, and Y. Tian, "PowerGear: Early-stage power estimation in FPGA HLS via heterogeneous edge-centric GNNs," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2022, pp. 1341–1346.
- [71] H.-Y. Liu and L. P. Carloni, "On learning-based methods for design-space exploration with high-level synthesis," in *Design automation conference (DAC)*, 2013.
- [72] P. Meng, A. Althoff, Q. Gautier, and R. Kastner, "Adaptive threshold non-Pareto elimination: Re-thinking machine learning for system level design space exploration on FPGAs," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2016, pp. 918–923.
- [73] R. G. Kim, J. R. Doppa, and P. P. Pande, "Machine learning for design space exploration and optimization of manycore systems," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–6.

- [74] A. Mahapatra and B. C. Schafer, "Machine-learning based simulated annealer method for high level synthesis design space exploration," in *Electronic System Level Synthesis Conference (ESLsyn)*, 2014, pp. 1–6.
- [75] Z. Wang and B. C. Schafer, "Machine learning to set meta-heuristic specific parameters for high-level synthesis design space exploration," in *ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.
- [76] Q. Sun, T. Chen, S. Liu, J. Chen, H. Yu, and B. Yu, "Correlated multi-objective multi-fidelity optimization for HLS directives design," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 27, no. 4, pp. 1–27, 2022.
- [77] Z. Yu, C. Bail, S. Hu, R. Chen, T. He, M. Yuan, B. Yu, and M. Wong, "IT-DSE: Invariance risk minimized transfer microarchitecture design space exploration," in *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–9.
- [78] Q. Xiao, S. Zheng, B. Wu, P. Xu, X. Qian, and Y. Liang, "HASCO: Towards agile hardware and software co-design for tensor computation," in *IEEE/ACM International Symposium on Computer Architecture (ISCA)*, 2021, pp. 1055–1068.
- [79] C. Xu, C. Kjellqvist, and L. W. Wills, "SNS's not a synthesizer: a deep-learning-based synthesis predictor," in *International Symposium on Computer Architecture (ISCA)*, 2022.
- [80] P. Sengupta, A. Tyagi, Y. Chen, and J. Hu, "How good is your Verilog RTL code? a quick answer from machine learning," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, 2022.
- [81] C. Xu, P. Sharma, T. Wang, and L. W. Wills, "Fast, robust and transferable prediction for hardware logic synthesis," in *IEEE/ACM International Symposium on Microarchitecture*, 2023, pp. 167–179.
- [82] W. Fang, Y. Lu, S. Liu, Q. Zhang, C. Xu, L. W. Wills, H. Zhang, and Z. Xie, "MasterRTL: A pre-synthesis PPA estimation framework for any RTL design," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2023.
- [83] D. S. Lopera and W. Ecker, "Applying GNNs to timing estimation at RTL," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022.
- [84] N. Wu, J. Lee, Y. Xie, and C. Hao, "Lostin: Logic optimization via spatio-temporal information with hybrid graph models," in *International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2022.
- [85] Y. Zhou, H. Ren, Y. Zhang, B. Keller, B. Khailany, and Z. Zhang, "PRIMAL: Power inference using machine learning," in *Design Automation Conference (DAC)*, 2019.
- [86] D. Lee, L. K. John, and A. Gerstlauer, "Dynamic power and performance back-annotation for fast and accurate functional hardware simulation," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2015.
- [87] A. K. A. Kumar and A. Gerstlauer, "Learning-based CPU power modeling," in *ACM/IEEE Workshop on Machine Learning for CAD (MLCAD)*, 2019.
- [88] Z. Xie, S. Li, M. Ma, C.-C. Chang, J. Pan, Y. Chen, and J. Hu, "DEEP: Developing extremely efficient runtime on-chip power meters," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022.
- [89] D. Zoni, L. Cremona, and W. Fornaciari, "PowerProbe: Run-time power modeling through automatic RTL instrumentation," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2018.
- [90] D. J. Pagliari, V. Peluso, Y. Chen, A. Calimera, E. Macii, and M. Poncino, "All-digital embedded meters for on-line power estimation," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2018.
- [91] Z. Xie, X. Xu, M. Walker, J. Knebel, K. Palaniswamy, N. Hebert, J. Hu, H. Yang, Y. Chen, and S. Das, "APOLLO: An automated power modeling framework for runtime power introspection in high-volume commercial microprocessors," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2021.
- [92] D. Kim, J. Zhao, J. Bachrach, and K. Asanović, "Simmani: Runtime power modeling for arbitrary RTL with automatic signal selection," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2019.
- [93] J. Yang, L. Ma, K. Zhao, Y. Cai, and T.-F. Ngai, "Early stage real-time SoC power estimation using RTL instrumentation," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2015.
- [94] S. Fine and A. Ziv, "Coverage directed test generation for functional verification using bayesian networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2003.
- [95] S. Vasudevan, W. J. Jiang, D. Bieber, R. Singh, C. R. Ho, C. Sutton *et al.*, "Learning semantic representations to verify hardware designs," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, pp. 23 491–23 504, 2021.
- [96] Y. Katz, M. Rimon, A. Ziv, and G. Shaked, "Learning microarchitectural behaviors to improve stimuli generation quality," in *ACM/IEEE Design Automation Conference (DAC)*, 2011.
- [97] W. L. Neto, M. Austin, S. Temple, L. Amaru, X. Tang, and P.-E. Gaillardon, "Lsoracle: A logic synthesis framework driven by artificial intelligence," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2019, pp. 1–6.
- [98] C. Yu, H. Xiao, and G. De Micheli, "Developing synthesis flows without human knowledge," in *ACM/IEEE Design Automation Conference (DAC)*, 2018.
- [99] C. Yu and W. Zhou, "Decision making in synthesis cross technologies using LSTMs and transfer learning," in *ACM/IEEE Workshop on Machine Learning for CAD (MLCAD)*, 2020, pp. 55–60.
- [100] Z. Pei, F. Liu, Z. He, G. Chen, H. Zheng, K. Zhu, and B. Yu, "AlphaSyn: Logic synthesis optimization with efficient monte carlo tree search," in *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023, pp. 1–9.
- [101] W. L. Neto, M. T. Moreira, Y. Li, L. Amaru, C. Yu, and P.-E. Gaillardon, "SLAP: A supervised learning approach for priority cuts technology mapping," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 859–864.
- [102] W. L. Neto, M. T. Moreira, L. Amaru, C. Yu, and P.-E. Gaillardon, "Read your circuit: leveraging word embedding to guide logic optimization," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2021, pp. 530–535.
- [103] Z. Xie, R. Liang, X. Xu, J. Hu, C.-C. Chang, J. Pan, and Y. Chen, "Preplacement net length and timing estimation by customized graph neural network," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 11, pp. 4667–4680, 2022.
- [104] Y. Zhang, H. Ren, and B. Khailany, "GRANNITE: Graph neural network inference for transferable power estimation," in *Design Automation Conference (DAC)*, 2020.
- [105] M. Rakesh, P. Das, A. Terkar, and A. Acharyya, "GRASPE: Accurate post-synthesis power estimation from RTL using graph representation learning," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2023, pp. 1–5.
- [106] S. Khan, Z. Shi, M. Li, and Q. Xu, "DeepSeq: Deep sequential circuit learning," *arXiv preprint arXiv:2302.13608*, 2023.
- [107] S. D. Chowdhury, K. Yang, and P. Nuzzo, "ReIGNN: State register identification using graph neural networks for circuit reverse engineering," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021, pp. 1–9.
- [108] L. Alrahis, A. Sengupta, J. Knechtel, S. Patnaik, H. Saleh, B. Mohammad, M. Al-Qutayri, and O. Sinanoglu, "GNN-RE: Graph neural networks for reverse engineering of gate-level netlists," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 8, pp. 2435–2448, 2021.
- [109] Z. He, Z. Wang, C. Bail, H. Yang, and B. Yu, "Graph learning-based arithmetic block identification," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–8.
- [110] N. Wu, Y. Li, C. Hao, S. Dai, C. Yu, and Y. Xie, "Gamora: Graph learning based symbolic reasoning for large-scale Boolean networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2022.
- [111] S. Ward, D. Ding, and D. Z. Pan, "PADE: A high-performance placer with automatic datapath extraction and evaluation through high dimensional data learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2012, pp. 756–761.
- [112] Y. Lin, S. Dhar, W. Li, H. Ren, B. Khailany, and D. Z. Pan, "DREAM-Place: Deep learning toolkit-enabled GPU acceleration for modern VLSI placement," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.
- [113] A. Agnesina, P. Rajvanshi, T. Yang, G. Pradipta, A. Jiao, B. Keller, B. Khailany, and H. Ren, "AutoDMP: Automated DREAMPlace-based macro placement," in *ACM International Symposium on Physical Design (ISPD)*, 2023.
- [114] Z. Xie, Y.-H. Huang, G.-Q. Fang, H. Ren, S.-Y. Fang, Y. Chen, and J. Hu, "RouteNet: Routability prediction for mixed-size designs using convolutional neural network," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018.
- [115] Y.-H. Huang, Z. Xie, G.-Q. Fang, T.-C. Yu, H. Ren, S.-Y. Fang, Y. Chen, and J. Hu, "Routability-driven macro placement with embedded CNN-based prediction model," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2019.
- [116] C.-C. Chang, J. Pan, T. Zhang, Z. Xie, J. Hu, W. Qi, C. Lin, R. Liang, J. Mitra, E. Fallon, and Y. Chen, "Automatic routability predictor development using neural architecture search," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021.

- [117] J. Pan, C.-C. Chang, Z. Xie, A. Li, M. Tang, T. Zhang, J. Hu, and Y. Chen, "Towards collaborative intelligence: Routability estimation based on decentralized private data," in *ACM/IEEE Design Automation Conference (DAC)*, 2022.
- [118] S. Zheng, L. Zou, P. Xu, S. Liu, B. Yu, and M. Wong, "Lay-Net: Grafting netlist knowledge on layout-based congestion prediction," in *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–9.
- [119] S. Liu, Q. Sun, P. Liao, Y. Lin, and B. Yu, "Global placement with deep learning-enabled explicit routability optimization," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2021, pp. 1821–1824.
- [120] J. Chen, J. Kuang, G. Zhao, D. J.-H. Huang, and E. F. Young, "PROS: A plug-in for routability optimization applied in the state-of-the-art commercial EDA tool using deep learning," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.
- [121] S. Zheng, L. Zou, S. Liu, Y. Lin, B. Yu, and M. Wong, "Mitigating distribution shift for congestion optimization in global placement," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [122] E. C. Barboza, N. Shukla, Y. Chen, and J. Hu, "Machine learning-based pre-routing timing prediction with reduced pessimism," in *ACM/IEEE Design Automation Conference (DAC)*, 2019.
- [123] X. He, Z. Fu, Y. Wang, C. Liu, and Y. Guo, "Accurate timing prediction at placement stage with look-ahead RC network," in *ACM/IEEE Design Automation Conference (DAC)*, 2022, pp. 1213–1218.
- [124] P. Cao, G. He, and T. Yang, "TF-Predictor: Transformer-based pre-routing path delay prediction framework," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, no. 99, pp. 1–1, 2022.
- [125] Z. Guo, M. Liu, J. Gu, S. Zhang, D. Z. Pan, and Y. Lin, "A timing engine inspired graph neural network model for pre-routing slack prediction," in *ACM/IEEE Design Automation Conference (DAC)*, 2022, pp. 1207–1212.
- [126] Z. Wang, S. Liu, Y. Pu, S. Chen, T.-Y. Ho, and B. Yu, "Restructure-tolerant timing prediction via multimodal fusion," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [127] R. Liang, Z. Xie, J. Jung, V. Chauha, Y. Chen, J. Hu, H. Xiang, and G.-J. Nam, "Routing-free crosstalk prediction," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.
- [128] S. Liu, Z. Wang, F. Liu, Y. Lin, B. Yu, and M. Wong, "Concurrent sign-off timing optimization via deep steiner points refinement," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [129] A. B. Kahng, U. Mallappa, and L. Saul, "Using machine learning to predict path-based slack from graph-based timing analysis," in *IEEE International Conference on Computer Design (ICCD)*, 2018, pp. 603–612.
- [130] Y. Ye, T. Chen, Y. Gao, H. Yan, B. Yu, and L. Shi, "Graph-learning-driven path-based timing analysis results predictor from graph-based timing analysis," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2023, pp. 547–552.
- [131] C.-T. Ho and A. B. Kahng, "IncPIRD: Fast learning-based prediction of incremental IR drop," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019.
- [132] C.-H. Pao, A.-Y. Su, and Y.-M. Lee, "XGBIR: An XGBoost-based IR drop predictor for power delivery network," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2020, pp. 1307–1310.
- [133] Y.-C. Fang, H.-Y. Lin, M.-Y. Sui, C.-M. Li, and E. J.-W. Fang, "Machine-learning-based dynamic IR drop prediction for ECO," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–7.
- [134] M. B. Alawieh, Y. Lin, Z. Zhang, M. Li, Q. Huang, and D. Z. Pan, "GAN-SRAF: subresolution assist feature generation using generative adversarial networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 40, no. 2, pp. 373–385, 2020.
- [135] H. Yang, S. Li, Z. Deng, Y. Ma, B. Yu, and E. F. Young, "GAN-OPC: Mask optimization with lithography-guided generative adversarial nets," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 39, no. 10, pp. 2822–2834, 2020.
- [136] G. Chen, Z. Yu, H. Liu, Y. Ma, and B. Yu, "DevelSet: Deep neural level set for instant mask optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 42, no. 12, pp. 5020–5033, 2023.
- [137] B. Zhu, S. Zheng, Z. Yu, G. Chen, Y. Ma, F. Yang, B. Yu, and M. D. Wong, "L2O-ILT: Learning to optimize inverse lithography techniques," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2023.
- [138] Y. Watanabe, T. Kimura, T. Matsunawa, and S. Nojima, "Accurate lithography simulation model based on convolutional neural networks," in *Optical Microlithography XXX*, vol. 10147. SPIE, 2017, pp. 137–145.
- [139] W. Ye, M. B. Alawieh, Y. Lin, and D. Z. Pan, "LithoGAN: End-to-end lithography modeling with generative adversarial networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2019.
- [140] Y. Lin, M. Li, Y. Watanabe, T. Kimura, T. Matsunawa, S. Nojima, and D. Z. Pan, "Data efficient lithography modeling with transfer learning and active data selection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 38, no. 10, pp. 1900–1913, 2018.
- [141] G. Chen, Z. Pei, H. Yang, Y. Ma, B. Yu, and M. Wong, "Physics-informed optical kernel regression using complex-valued neural fields," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [142] H. Yang, L. Luo, J. Su, C. Lin, and B. Yu, "Imbalance aware lithography hotspot detection: a deep learning approach," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 16, no. 3, pp. 033 504–033 504, 2017.
- [143] J. Chen, Y. Lin, Y. Guo, M. Zhang, M. B. Alawieh, and D. Z. Pan, "Lithography hotspot detection using a double inception module architecture," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 18, no. 1, pp. 013 507–013 507, 2019.
- [144] Y. Jiang, F. Yang, B. Yu, D. Zhou, and X. Zeng, "Efficient layout hotspot detection via binarized residual neural network ensemble," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 40, no. 7, pp. 1476–1488, 2020.
- [145] A. Ciccazzo, G. Di Pillo, and V. Latorre, "A SVM surrogate model-based method for parametric yield optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 35, no. 7, pp. 1224–1228, 2015.
- [146] K. Nakata, R. Orihara, Y. Mizuoka, and K. Takagi, "A comprehensive big-data-based monitoring system for yield enhancement in semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing (TSM)*, vol. 30, no. 4, pp. 339–344, 2017.
- [147] M. B. Alawieh, D. Boning, and D. Z. Pan, "Wafer map defect patterns classification using deep selective learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.
- [148] J. Kwon, M. M. Ziegler, and L. P. Carloni, "A learning-based recommender system for autotuning design flows of industrial high-performance processors," in *ACM/IEEE Design Automation Conference (DAC)*, 2019.
- [149] Z. Xie, G.-Q. Fang, Y.-H. Huang, H. Ren, Y. Zhang, B. Khailany, S.-Y. Fang, J. Hu, Y. Chen, and E. C. Barboza, "FIST: A feature-importance sampling and tree-based method for automatic design flow parameter tuning," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2020.
- [150] H. Geng, T. Chen, Y. Ma, B. Zhu, and B. Yu, "PTPT: physical design tool parameter tuning via multi-objective bayesian optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 42, no. 1, pp. 178–189, 2022.
- [151] M. Cho, K. Yuan, Y. Ban, and D. Z. Pan, "Eliad: Efficient lithography aware detailed routing algorithm with compact and macro post-opc printability prediction," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 7, pp. 1006–1016, 2009.
- [152] Synopsys, "Synopsys. ai unveiled as industry's first full-stack, ai-driven eda suite for chipmakers," 2023. [Online]. Available: <https://news.synopsys.com/2023-03-29-Synopsys-ai-Unveiled-as-Industrys-First-Full-Stack,-AI-Driven-EDA-Suite-for-Chipmakers>
- [153] G. Liu and Z. Zhang, "PIMap: A flexible framework for improving LUT-based technology mapping via parallelized iterative optimization," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 11, no. 4, pp. 1–23, 2019.
- [154] C. Yu, "FlowTune: Practical multi-armed bandits in Boolean optimization," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020, pp. 1–9.
- [155] K. Zhu, M. Liu, H. Chen, Z. Zhao, and D. Z. Pan, "Exploring logic optimizations with reinforcement learning and graph convolutional network," in *ACM/IEEE Workshop on Machine Learning for CAD (MLCAD)*, 2020, pp. 145–150.
- [156] A. Hosny, S. Hashemi, M. Shalan, and S. Reda, "DRiLLS: Deep reinforcement learning for logic synthesis," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2020, pp. 581–586.
- [157] Y. V. Peruvemba, S. Rai, K. Ahuja, and A. Kumar, "RL-guided runtime-constrained heuristic exploration for logic synthesis," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2021, pp. 1–9.

- [158] W. Haaswijk, E. Collins, B. Seguin, M. Soeken, F. Kaplan, S. Süsstrunk, and G. De Micheli, "Deep learning for logic optimization algorithms," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–4.
- [159] X. Timoneda and L. Cavigelli, "Late breaking results: Reinforcement learning for scalable logic optimization with graph neural networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1378–1379.
- [160] A. Mirhoseini, A. Goldie, M. Yazgan, J. W. Jiang, E. Songhori, S. Wang, Y.-J. Lee, E. Johnson, O. Pathak, A. Nazi *et al.*, "A graph placement methodology for fast chip design," *Nature*, vol. 594, pp. 207–212, 2021.
- [161] Q. Xu, H. Geng, S. Chen, B. Yuan, C. Zhuo, Y. Kang, and X. Wen, "Good-Floorplan: Graph convolutional network and reinforcement learning-based floorplanning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 10, pp. 3492–3502, 2021.
- [162] A. Agnesina, K. Chang, and S. K. Lim, "VLSI placement parameter optimization using deep reinforcement learning," in *IEEE/ACM International Conference on Computer-Aided Design*, 2020, pp. 1–9.
- [163] Y.-C. Lu, S. Nath, V. Khandelwal, and S. K. Lim, "RL-Sizer: VLSI gate sizing for timing optimization using deep reinforcement learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 733–738.
- [164] Y.-C. Lu, W.-T. Chan, D. Guo, S. Kundu, V. Khandelwal, and S. K. Lim, "RL-CCD: Concurrent clock and data optimization using attention-based self-supervised reinforcement learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [165] X. Liang, Y. Ouyang, H. Yang, B. Yu, and Y. Ma, "RL-OPC: Mask optimization with deep reinforcement learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 43, no. 1, pp. 340–351, 2024.
- [166] Y.-C. Lu, J. Lee, A. Agnesina, K. Samadi, and S. K. Lim, "GAN-CTS: A generative adversarial framework for clock tree prediction and optimization," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019.
- [167] Y. Lu, S. Liu, Q. Zhang, and Z. Xie, "RTL-LLM: An open-source benchmark for design RTL generation with large language model," *arXiv preprint arXiv:2308.05345*, 2023.
- [168] M. Liu, N. Pinckney, B. Khailany, and H. Ren, "VerilogEval: evaluating large language models for Verilog code generation," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2023.
- [169] X. Liang, "Hardware descriptions code completion based on a pre-training model," in *IEEE Conference on Telecommunications, Optics and Computer Science (TOCS)*, 2021, pp. 228–232.
- [170] K. Chang, Y. Wang, H. Ren, M. Wang, S. Liang, Y. Han, H. Li, and X. Li, "ChipGPT: How far are we from natural language hardware design," *arXiv preprint arXiv:2305.14019*, 2023.
- [171] S. Thakur, J. Blocklove, H. Pearce, B. Tan, S. Garg, and R. Karri, "AutoChip: Automating HDL generation using LLM feedback," *arXiv preprint arXiv:2311.04887*, 2023.
- [172] J. Blocklove, S. Garg, R. Karri, and H. Pearce, "Chip-Chat: Challenges and opportunities in conversational hardware design," in *ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD)*, Sep. 2023.
- [173] M. Liu, T.-D. Ene, R. Kirby, C. Cheng, N. Pinckney, R. Liang, J. Alben, H. Anand, S. Banerjee, I. Bayraktaroglu, B. Bhaskaran, B. Catanzaro, A. Chaudhuri, S. Clay, B. Dally, L. Dang, P. Deshpande, S. Dhodhi, S. Halepete, E. Hill, J. Hu, S. Jain, B. Khailany, G. Kokai, K. Kunal, X. Li, C. Lind, H. Liu, S. Oberman, S. Omar, S. Pratty, J. Raiman, A. Sarkar, Z. Shao, H. Sun, P. P. Suthar, V. Tej, W. Turner, K. Xu, and H. Ren, "ChipNeMo: Domain-adapted LLMs for chip design," *arXiv preprint arXiv:2311.00176*, 2023.
- [174] S. Liu, W. Fang, Y. Lu, Q. Zhang, H. Zhang, and Z. Xie, "RTL-Coder: Outperforming GPT-3.5 in design RTL generation with our open-source dataset and lightweight solution," *arXiv preprint arXiv:2312.08617*, 2023.
- [175] Z. Pei, H.-L. Zhen, M. Yuan, Y. Huang, and B. Yu, "BetterV: Controlled Verilog Generation with Discriminative Guidance," *arXiv preprint arXiv:2402.03375*, 2024.
- [176] M. Orenes-Vera, M. Martonosi, and D. Wentzlaff, "Using LLMs to facilitate formal verification of RTL," *arXiv preprint arXiv:2309.09437*, 2023.
- [177] C. Sun, C. Hahn, and C. Trippel, "Towards improving verification productivity with circuit-aware translation of natural language to systemverilog assertions," in *First International Workshop on Deep Learning-aided Verification (DAV)*, 2023.
- [178] W. Fang, M. Li, M. Li, Z. Yan, S. Liu, H. Zhang, and Z. Xie, "AssertLLM: Generating and evaluating hardware verification assertions from design specifications via multi-LLMs," *arXiv preprint arXiv:2402.00386*, 2024.
- [179] Y. Zhang, H.-L. Zhen, Z. Pei, Y. Lian, L. Yin, M. Yuan, and B. Yu, "Sola: Solver-layer adaption of llm for better logic reasoning," *arXiv preprint arXiv:2402.11903*, 2024.
- [180] B. Ahmad, S. Thakur, B. Tan, R. Karri, and H. Pearce, "Fixing hardware security bugs with large language models," *arXiv preprint arXiv:2302.01215*, 2023.
- [181] M. Nair, R. Sadhukhan, and D. Mukhopadhyay, "Generating secure hardware using ChatGPT resistant to CWEs," *Cryptology ePrint Archive*, Paper 2023/212, 2023. [Online]. Available: <https://eprint.iacr.org/2023/212>
- [182] R. Kande, H. Pearce, B. Tan, B. Dolan-Gavitt, S. Thakur, R. Karri, and J. Rajendran, "LLM-assisted generation of hardware assertions," 2023.
- [183] Z. He, H. Wu, X. Zhang, X. Yao, S. Zheng, H. Zheng, and B. Yu, "ChatEDA: A large language model powered autonomous agent for EDA," 2023.
- [184] 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 *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–9.
- [185] Z. Yan, Y. Qin, X. S. Hu, and Y. Shi, "On the viability of using LLMs for SW/HW co-design: An example in designing CiM DNN accelerators," *arXiv preprint arXiv:2306.06923*, 2023.
- [186] Z. Liang, J. Cheng, R. Yang, H. Ren, Z. Song, D. Wu, X. Qian, T. Li, and Y. Shi, "Unleashing the potential of LLMs for quantum computing: A study in quantum architecture design," *arXiv preprint arXiv:2307.08191*, 2023.
- [187] M. Li, W. Fang, Q. Zhang, and Z. Xie, "SpecLLM: Exploring generation and review of VLSI design specification with large language model," *arXiv preprint arXiv:2401.13266*, 2024.
- [188] H. Ren and M. Fojtik, "Invited- NVCell: Standard cell layout in advanced technology nodes with reinforcement learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1291–1294.
- [189] —, "Standard cell routing with reinforcement learning and genetic algorithm in advanced technology nodes," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2021, pp. 684–689.
- [190] A. C.-W. Liang, C. H.-P. Wen, and H.-M. Huang, "A general and automatic cell layout generation framework with implicit learning on design rules," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 30, no. 9, pp. 1341–1354, 2022.
- [191] S. Roy, Y. Ma, J. Miao, and B. Yu, "A learning bridge from architectural synthesis to physical design for exploring power efficient high-performance adders," in *IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, Jul. 2017, pp. 1–6.
- [192] H. Geng, Y. Ma, Q. Xu, J. Miao, S. Roy, and B. Yu, "High-speed adder design space exploration via graph neural processes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 8, pp. 2657–2670, Aug. 2022.
- [193] J. Cheng, Y. Xiao, Y. Shao, G. Dong, S. Lyu, and W. Yu, "Machine-learning-driven architectural selection of adders and multipliers in logic synthesis," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 28, no. 2, pp. 20:1–20:16, Mar. 2023.
- [194] D. Zuo, Y. Ouyang, and Y. Ma, "RL-MUL: Multiplier design optimization with deep reinforcement learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [195] K. Zhu, H. Chen, W. J. Turner, G. F. Kokai, P.-H. Wei, D. Z. Pan, and H. Ren, "TAG: Learning circuit spatial embedding from layouts," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2022.
- [196] J. Lu, L. Lei, F. Yang, L. Shang, and X. Zeng, "Topology optimization of operational amplifier in continuous space via graph embedding," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2022, p. 142–147.
- [197] S. Fan, N. Cao, S. Zhang, J. Li, X. Guo, and X. Zhang, "From specification to topology: Automatic power converter design via reinforcement learning," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021.
- [198] Z. Zhao, J. Luo, J. Liu, and L. Zhang, "Signal-division-aware analog circuit topology synthesis aided by transfer learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 42, no. 11, pp. 3481–3490, 2023.
- [199] S. Poddar, A. Budak, L. Zhao, C.-H. Hsu, S. Maji, K. Zhu, Y. Jia, and D. Z. Pan, "A data-driven analog circuit synthesizer with automatic topology selection and sizing," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2024.
- [200] J. Lu, Y. Li, F. Yang, L. Shang, and X. Zeng, "High-level topology synthesis method for Δ - Σ modulators via bi-level bayesian optimization," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 12, pp. 4389–4393, 2023.

- [201] M. Fayazi, M. T. Taba, E. Afshari, and R. Dreslinski, "Angel: Fully-automated analog circuit generator using a neural network assisted semi-supervised learning approach," *IEEE Transactions on Circuits and Systems I*, vol. 70, no. 11, pp. 4516–4529, 2023.
- [202] K. Hakhamaneshi, M. Nassar, M. Phielipp, P. Abbeel, and V. Stojanovic, "Pretraining graph neural networks for few-shot analog circuit modeling and design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 42, no. 7, pp. 2163–2173, 2023.
- [203] A. Budak, M. Gandara, W. Shi, D. Pan, N. Sun, and B. Liu, "An efficient analog circuit sizing method based on machine learning assisted global optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 5, pp. 1209–1221, 2022.
- [204] H. Wang, K. Wang, J. Yang, L. Shen, N. Sun, H.-S. Lee, and S. Han, "GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2020.
- [205] A. Zhao, X. Wang, Z. Lin, Z. Bi, X. Li, C. Yan, F. Yang, L. Shang, D. Zhou, and X. Zeng, "cVTS: A constrained Voronoi tree search method for high dimensional analog circuit synthesis," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [206] A. F. Budak, S. Zhang, M. Liu, W. Shi, K. Zhu, and D. Z. Pan, *Machine Learning for Analog Circuit Sizing*. Cham: Springer International Publishing, 2022, pp. 307–335.
- [207] S. M. Burns, H. Chen, T. Dhar, R. Harjani, J. Hu, N. Karmokar, K. Kunal, Y. Li, Y. Lin, M. Liu, M. Madhusudan, P. Mukherjee, D. Z. Pan, J. Poojary, S. Ramprasath, S. S. Sapatnekar, A. K. Sharma, W. Xu, S. Yaldiz, and K. Zhu, *Machine Learning for Analog Layout*. Cham: Springer International Publishing, 2022, pp. 505–544.
- [208] K. Kunal, P. Poojary, T. Dhar, M. Madhusudan, R. Harjani, and S. Sapatnekar, "A general approach for identifying hierarchical symmetry constraints for analog circuit layout," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.
- [209] K. Zhu, H. Chen, M. Liu, and D. Z. Pan, "Automating analog constraint extraction: From heuristics to learning: (invited paper)," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2022, pp. 108–113.
- [210] K. Zhu, M. Liu, Y. Lin, B. Xu, S. Li, X. Tang, N. Sun, and D. Z. Pan, "GeniusRoute: A new analog routing paradigm using generative neural network guidance," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019.
- [211] B. Xu, Y. Lin, X. Tang, S. Li, L. Shen, N. Sun, and D. Z. Pan, "WellGAN: Generative-adversarial-network-guided well generation for analog/mixed-signal circuit layout," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.
- [212] A. Gusmão, N. Horta, N. Lourenço, and R. Martins, "Late breaking results: Attention in Graph2Seq neural networks towards push-button analog IC placement," in *ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1360–1361.
- [213] P.-C. Wang, M. P.-H. Lin, C.-N. J. Liu, and H.-M. Chen, "Layout synthesis of analog primitive cells with variational autoencoder," in *International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2023.
- [214] M. Liu, K. Zhu, J. Gu, L. Shen, X. Tang, N. Sun, and D. Z. Pan, "Towards decrypting the art of analog layout: Placement quality prediction via transfer learning," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2020, pp. 496–501.
- [215] Y. Lin, Y. Li, D. Fang, M. Madhusudan, S. S. Sapatnekar, R. Harjani, and J. Hu, "Are analytical techniques worthwhile for analog IC placement?" in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2022, pp. 154–159.
- [216] P. Xu, J. Li, T.-Y. Ho, B. Yu, and K. Zhu, "Performance-driven analog layout automation: Current status and future directions," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2024.
- [217] H. Ren, G. F. Kokai, W. J. Turner, and T.-S. Ku, "ParaGraph: Layout parasitics and device parameter prediction using graph neural networks," in *ACM/IEEE Design Automation Conference (DAC)*, 2020.
- [218] Q. Zhang, S. Su, J. Liu, and M. S.-W. Chen, "CEPA: CNN-based early performance assertion scheme for analog and mixed-signal circuit simulation," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2020.
- [219] K. Hakhamaneshi, N. Werblun, P. Abbeel, and V. Stojanović, "BagNet: Berkeley analog generator with layout optimizer boosted with deep neural networks," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019.
- [220] Y. Fang, Z. Liu, Y. Lu, J. Liu, J. Li, Y. Jin, J. Chen, Y. Chen, H. Zheng, and Y. Xie, "NPS: A framework for accurate program sampling using graph neural network," *arXiv preprint arXiv:2304.08880*, 2023.
- [221] L. Li, T. Flynn, and A. Hoisie, "Learning independent program and architecture representations for generalizable performance modeling," *arXiv preprint arXiv:2310.16792*, 2023.
- [222] X. Yi, J. Lu, X. Xiong, D. Xu, L. Shang, and F. Yang, "Graph representation learning for microarchitecture design space exploration," in *ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [223] C. Sakhuja, Z. Shi, and C. Lin, "Leveraging domain information for the efficient automated design of deep learning accelerators," in *IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 2023, pp. 287–301.
- [224] M. Li, Z. Shi, Q. Lai, S. Khan, S. Cai, and Q. Xu, "On EDA-driven learning for SAT solving," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.
- [225] Z. Shi, M. Li, S. Khan, L. Wang, N. Wang, Y. Huang, and Q. Xu, "DeepTPI: Test point insertion with deep reinforcement learning," in *IEEE International Test Conference (ITC)*, 2022, pp. 194–203.
- [226] 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 *ACM/IEEE Design Automation Conference*, 2022, pp. 61–66.
- [227] Z. Xie, H. Ren, B. Khailany, Y. Sheng, S. Santosh, J. Hu, and Y. Chen, "PowerNet: Transferable dynamic IR drop estimation via maximum convolutional neural network," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2020.
- [228] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang *et al.*, "CodeBERT: A pre-trained model for programming and natural languages," *arXiv preprint arXiv:2002.08155*, 2020.
- [229] M. Orenes-Vera, M. Martonosi, and D. Wentzlaff, "From RTL to SVA: LLM-assisted generation of formal verification testbenches," *arXiv preprint arXiv:2309.09437*, 2023.
- [230] N. Eén, A. Mishchenko, and N. Sörensson, "Applying logic synthesis for speeding up SAT," in *Theory and Applications of Satisfiability Testing*. Springer, 2007, pp. 272–286.
- [231] N. Sorensson and N. Een, "MiniSAT v1.13 - a SAT solver with conflict-clause minimization," *SAT*, vol. 2005, no. 53, pp. 1–2, 2005.
- [232] A. Fleury and M. Heisinger, "CADICAL, KISSAT, PARACOBA, PLINGELING and TREENGELING entering the SAT competition 2020," *SAT Competition*, vol. 2020, p. 50, 2020.
- [233] Cadence, "Conformal Smart LEC," 2022. [Online]. Available: https://www.cadence.com/en_US/home/resources/datasheets/conformal-smart-lec-ds.html
- [234] S. Zou, J. Zhang, B. Shi, and G. Luo, "BESWAC: Boosting exact synthesis via wiser SAT solver call," in *IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE)*, 2024.
- [235] Z. Chen, X. Zhang, Y. Qian, Q. Xu, and S. Cai, "Integrating exact simulation into sweeping for datapath combinational equivalence checking," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2023, pp. 1–9.
- [236] L. Liu, B. Fu, M. D. F. Wong, and E. F. Y. Young, "Xplace: An Extremely Fast and Extensible Global Placement Framework," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022.
- [237] B. Wang, G. Shen, D. Li, J. Hao, W. Liu, Y. Huang, H. Wu, Y. Lin, G. Chen, and P. A. Heng, "LHNN: Lattice hypergraph neural network for VLSI congestion prediction," in *ACM/IEEE Design Automation Conference (DAC)*, San Francisco, CA, July 2022.
- [238] Y. Pu, C. Shi, G. Samson, D. Park, K. Easton, R. Beraha, A. Newham, M. Lin, V. Rangan, K. Chatha, D. Butterfield, and R. Attar, "A 9-mm² ultra-low-power highly integrated 28-nm CMOS SoC for Internet of Things," *IEEE Journal Solid-State Circuits*, vol. 53, no. 3, pp. 936–948, 2018.
- [239] S. Jain, S. Khare, S. Yada, V. Ambili, P. Salihundam, S. Ramani, S. Muthukumar, M. Srinivasan, A. Kumar, S. K. Gb, R. Ramanarayanan, V. Erraguntla, J. Howard, S. Vangal, S. Dighe, G. Ruhl, P. Aseron, H. Wilson, N. Borkar, V. De, and S. Borkar, "A 280mV-to-1.2V wide-operating-range IA-32 processor in 32nm CMOS," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2012, pp. 66–68.
- [240] F. Klemme and H. Amrouch, "Efficient learning strategies for machine learning-based characterization of aging-aware cell libraries," *IEEE Transactions on Circuits and Systems I*, vol. 69, no. 12, pp. 5233–5246, 2022.
- [241] Mentor, a Siemens Business, *Solido Characterization Suite*, 2023. [Online]. Available: <https://eda.sw.siemens.com/en-US/ic/solido/>