SCIENCE CHINA
Information Sciences

October 2024, Vol. 67, Iss. 10, 200402:1–200402:42
https://doi.org/10.1007/s11432-024-4155-7

• POSITION PAPER •
Special Topic: AI Chips and Systems for Large Language Models

# Large circuit models: opportunities and challenges†

Lei CHEN[5], Yiqi CHEN[7], Zhufei CHU[6], Wenji FANG[3], Tsung-Yi HO[1], Ru HUANG[7,11], Yu HUANG[4], Sadaf KHAN[1], Min LI[5], Xingquan LI[9], Yu LI[1], Yun LIANG[7], Jinwei LIU[1], Yi LIU[1], Yibo LIN[7], Guojie LUO[8*], Hongyang PAN[2], Zhengyuan SHI[1], Guangyu SUN[7], Dimitrios TSARAS[5], Runsheng WANG[7], Ziyi WANG[1], Xinming WEI[8], Zhiyao XIE[3], Qiang XU[1*], Chenhao XUE[7], Junchi YAN[10], Jun YANG[11], Bei YU[1], Mingxuan YUAN[5*], Evangeline F.Y. YOUNG[1], Xuan ZENG[2], Haoyi ZHANG[7], Zuodong ZHANG[7], Yuxiang ZHAO[7], Hui-Ling ZHEN[5], Ziyang ZHENG[1], Binwu ZHU[1], Keren ZHU[1] & Sunan ZOU[8]

[1]*Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong* 999077*, China;*
[2]*School of Microelectronics, State Key Laboratory of Integrated Chips and System, Fudan University, Shanghai* 200433*, China;*
[3]*Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong* 999077*, China;*
[4]*Huawei HiSilicon, Shenzhen* 518129*, China;*
[5]*Huawei Noah's Ark Lab, Hong Kong* 999077*, China;*
[6]*Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo* 315211*, China;*
[7]*School of Integrated Circuits, Peking University, Beijing* 100871*, China;*
[8]*School of Computer Science, Peking University, Beijing* 100871*, China;*
[9]*Peng Cheng Laboratory, Shenzhen* 518052*, China;*
[10]*School of Artificial Intelligence, Shanghai Jiao Tong University, Shanghai* 200240*, China;*
[11]*School of Integrated Circuits, Southeast University, Nanjing* 210096*, China*

Received 22 April 2024/Revised 19 July 2024/Accepted 15 September 2024/Published online 25 September 2024

**Abstract** Within the electronic design automation (EDA) domain, artificial intelligence (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 study argues for a paradigm shift from AI4EDA towards AI-rooted EDA from the ground up, 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, register-transfer level (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-rooted 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.

**Keywords** AI-rooted EDA, large circuit models (LCMs), multimodal circuit representation learning, circuit optimization

* Corresponding author (email: gluo@pku.edu.cn, qxu@cse.cuhk.edu.hk, yuan.mingxuan@huawei.com)
† The institutions and authors are ordered alphabetically.

 

# 1 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 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 the CV area, self-supervised foundation models [6–8] 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 exemplifies 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 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 and optimization. Specifically, AI4EDA involves applying or adapting existing ML algorithms and AI methodologies to improve specific tasks within the current electronic design automation (EDA) framework. 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.

However, 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-rooted 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-rooted EDA from the ground up. Here, AI-rooted EDA refers to the development of new ML techniques and methodologies for EDA that are fundamentally based on AI principles from the ground up. It involves developing representation learning solutions for circuit data from scratch and creating new EDA solutions that inherently rely on AI techniques for their core functionality.

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 Feasibility and promises of AI-rooted 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][1)] 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 [20] and graph neural networks (GNNs) [21], 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 [22] and large language models for code generation [23], 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-rooted 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-rooted EDA, focusing on the development and application of large circuit models that inherently incorporate multimodal data. Spanning 9 sections, the sudy 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.

1) OpenCores. OpenCores. http://opencores.org/.

In Section 4, we introduce AI-rooted 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 study 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-rooted 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 submicron 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.

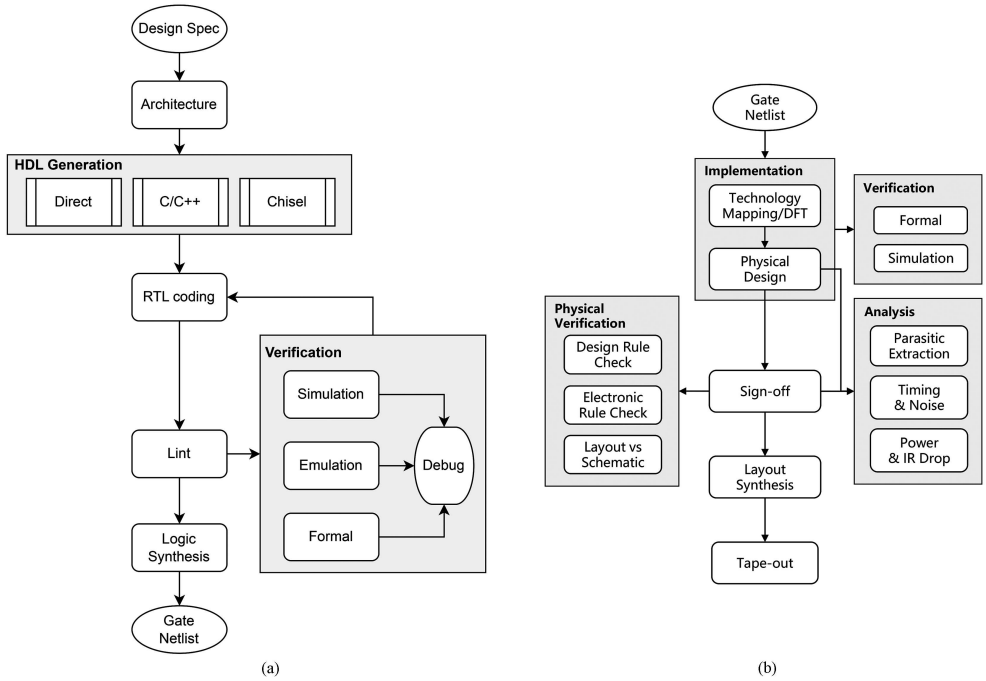


**Figure 1** Typical (a) front-end and (b) back-end design flows.

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) [24], 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 [25]. 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 Figure 1(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 [26]) 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 [27] 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 compo-

nents 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 [28, 29], is frequently applied, particularly in the context of data-path intensive designs, as highlighted by [30]. Given that circuit designs within this abstract primarily encapsulate hardware behavior while abstracting concrete physical details, theorem provers and satisfiability modulo theories (SMT) solvers emerge as pivotal tools for enhancing verification efficacy [31, 32].

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 [33–35]. Furthermore, custom equivalence checking techniques have been tailored to cater to specific circuit design requirements, such as those pertaining to clock-gating [36].

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 very-large-scale integration (VLSI) designs, the back-end design flow, also referred to as layout design, is depicted in Figure 1(b), transitioning from a gate-level or generic technology (GTech) netlist to a finalized layout [37].

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 scans 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. We list a few representative techniques for each task in the following.

• **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. Ref. [38] applied the simulated annealing optimization technique to VLSI floorplanning. The authors demonstrate how simulated annealing can effectively explore the solution space to find optimal or near-optimal floorplans, significantly influencing subsequent floorplanning methodologies. Ref. [39] introduced B*-Trees for representing non-slicing floorplans. This representation allows for more flexible and efficient manipulation of floorplan topologies, leading to better optimization of area and wirelength in VLSI designs.

• **PDN design.** PDN design ensures a 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. Ref. [40] presented a methodology for designing power distribution systems in modern CMOS technology. It focuses on the selection and placement of decoupling capacitors to manage power integrity and reduce noise, offering practical guidelines and strategies for effective PDN design. Ref. [41] comprehensively discussed the principles and strategies for designing power distribution networks in VLSI circuits. It highlights the importance of hierarchical design, decoupling strategies, and the integration of power grids with signal routing to ensure robust and efficient power delivery.

• **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. Ref. [42] presented enhancements to the simulated annealing algorithm for row-based placement, incorporating techniques to improve convergence speed and solution quality. It significantly influenced the development of placement tools by demonstrating the effectiveness of simulated annealing in handling the placement problem. Ref. [43] formulated the placement problem as a quadratic programming task. It achieves high-quality placements with efficient computational performance by iteratively solving quadratic programs and slicing the problem into smaller sections. Ref. [44] introduced a fast and efficient analytical placement method that combines cell shifting, iterative local refinement, and a hybrid net model. This approach provides a balance between placement quality and runtime, making it suitable for large-scale designs. Ref. [45] was an analytical placer that addresses the challenges of large-scale mixed-size designs, including preplaced blocks and density constraints. By combining quadratic placement with discrete optimization, NTUPlace3 achieves excellent wire length and timing performance. A comprehensive survey can be found [46].

• **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. Ref. [47] introduced a zero-skew clock routing algorithm that aims to minimize wirelength while achieving zero skew. The approach uses a recursive geometric method to balance the clock tree, significantly reducing skew and improving timing reliability in VLSI designs. Ref. [48] resented an efficient buffer sizing algorithm aimed at reducing clock skew in the presence of process variations in VLSI designs. By quantitatively estimating the skew distribution through Monte-Carlo SPICE simulations and analyzing the impact of process variations on wire and buffer delays, the algorithm strategically adjusts the number and size of buffers on critical paths. A comprehensive survey can be found [37].

• **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. Ref. [49] introduced CUGR, a detailed-routability-driven 3D global routing algorithm that utilizes a probabilistic resource model to optimize routing quality and efficiency. The proposed approach incorporates two key techniques: 3D pattern routing, which combines pattern routing and layer assignment to optimize wire length and routability, and multi-level 3D maze routing, which uses a coarsened grid graph to efficiently find routable regions and detailed paths. Ref. [50] resented a negotiation-based global routing algorithm that focuses on achieving timing closure in complex VLSI designs. The algorithm iteratively adjusts routing paths and resources to meet timing constraints, ensuring reliable performance in the final design. Ref. [51] was a comprehensive tool for field-programmable gate array (FPGA) research that integrates packing, placement, and routing. The study demonstrates its effectiveness in optimizing FPGA designs, making it a widely used tool in the field. A comprehensive survey can be found for ASIC routing [52] and FPGA routing [53].

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 [54].

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 and 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.

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.

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 include dynamic program, reinforcement learning, and satisfiability modulo theories. Innovations in placement strategies, as highlighted in [55, 56], 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 is usually solved by A-star, integer 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 [57, 58] 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, and 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 adder designs, such as Sklansky, Kogge-Stone, and Brent-Kung adders [59], adopt predefined rules to generate adders with arbitrary bit-widths and optimized performance and area, laying a foundation for efficient design practices. 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 [60]. In addition to adder designs created by human experts, datapath compilers have become instrumental in navigating the vast prefix-tree design space and balance the trade-offs between different configurations, employing algorithmic [61] and heuristic methods [62, 63] to select the optimal structure for a given application scenario. The synthesized adders exhibit improved quality by effectively meeting various constraints such as delay, area, and fanout.

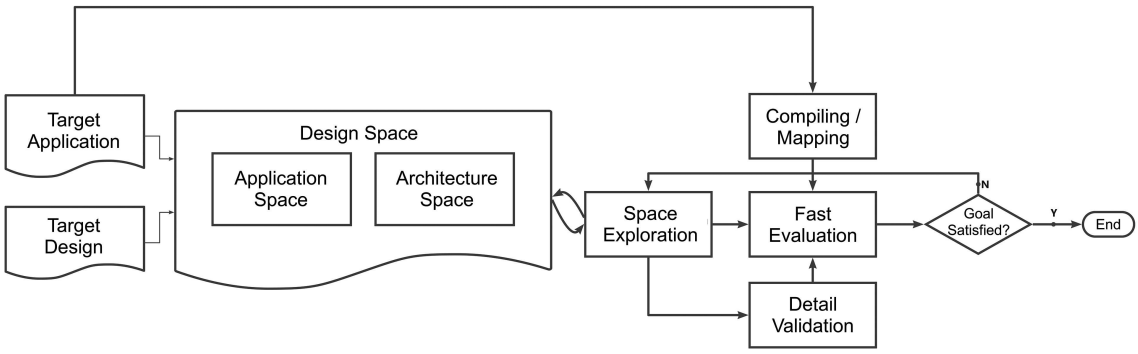


**Figure 2** Typical datapath circuits design flow.

**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 and compressing these groups in parallel, which is particularly favored in digital signal processing (DSP) and graphics processing units (GPUs) where rapid mathematical computations are critical. Similar to the development of prefix adders, early designs of Wallace trees relied on predefined rules to determine the grouping of partial products [64,65]. Recent advancements in efficient Wallace tree synthesis involve heuristic optimization algorithms, such as simulated annealing and integer linear programming [66], to assign compressors to different partial product groups, achieving reduced area and optimized delay. These advancements in automatic Wallace tree synthesis continuously enhance high-performance multiplier architectures 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. By leveraging existing IP libraries [67] for design selection, or jointly utilizing automatic adder/multiplier synthesis tools, EDA tools could integrate optimized adder designs with efficient multipliers within MAC units to achieve high throughput and low latency.

**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 fused multiply accumulate (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 Figure 2, the design journey initiates with pinpointing a target application and its corresponding architectural design, thereby defining a broad and intricate design space. The application scope includes general computing [68] and specialized functions, such as high-performance computing [69], cryptography [70], and digital signal processing [71], targeting either traditional CPU designs or domain-specific accelerators. The design space diverges into two principal domains: the application space, outlining application-specific parameters like dataflow patterns or neural network mapping strategies [72], and the architecture space, detailing the structural and resource parameters, such as CPU pipeline width [73] or the quantity of MACs in a neural processing unit (NPU) [74].

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

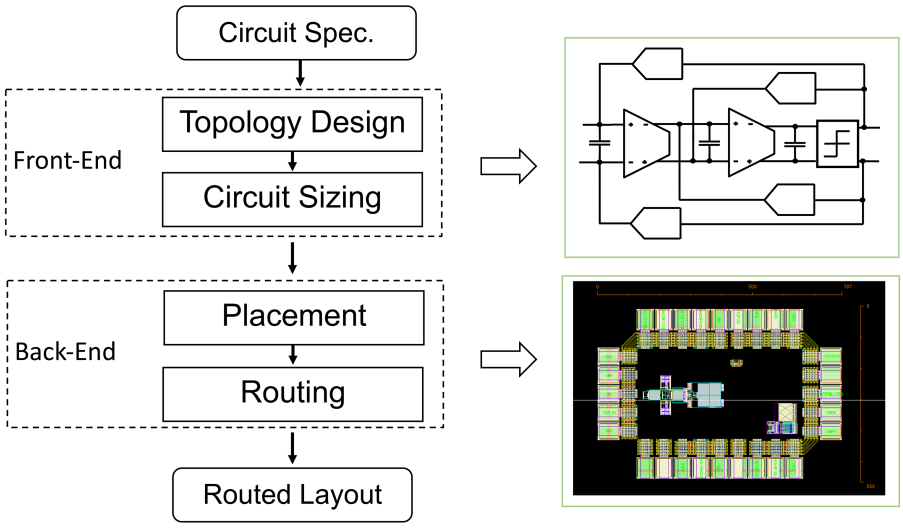


**Figure 3** (Color online) Typical analog IC design flow.

EDA tools. This rigorous process of iteratively exploring and evaluating new design points to achieve the targeted objectives of power, performance, and area (PPA) is commonly referred to as design space exploration (DSE).

The progression to new design points is typically steered by optimization algorithms, which have advanced significantly. These optimizations fall into two categories, either by leveraging black-box optimization or incorporating domain knowledge. The black-box optimization formulates the DSE process as a general searching problem and proceeds without presuppositions about the design space, often utilizing simulated annealing (SA) [75], genetic algorithms (GA) [76], and Bayesian optimization (BO) [73, 77]. 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 [78, 79] often exhibit superior exploration efficiency by focusing on specific areas for improvement within the datapath architecture. Despite these advancements, traditional EDA methods for optimizing datapath design face significant challenges in integrating generic heuristic algorithms with highly customized domain-specific knowledge, which impedes the enhancement of design space exploration (DSE) efficiency.

### 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.

Figure 3 illustrates a typical analog IC design flow, starting with a detailed set of circuit specifications covering area, power, and performance requirements. Analog EDA is typically divided into the front-end and the back-end. The front-end includes topology and device sizing, while the back-end focuses on the physical design. A survey for analog CAD can be found in [80].

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. Analog device sizing recently is an active research field. Analog sizing decides the parameters for the devices, such as widths for transistors, in analog IC. A survey for the recent analog sizing research can be found in [81].

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. A survey for analog layout automation can be found in [82].

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 [83], analog sizing [84], analog placement-and-route [85].

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, as discussed in previous surveys [86–91]. 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 studies in each category for a focused overview. For those seeking an exhaustive review, references such as [15, 92] offered 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. [93] and Ithemal [94], which apply linear and recurrent neural network (RNN) models for CPU performance modeling, respectively. McPAT-Calib [95] enhances CPU power modeling by integrating ML models with the analytical tool McPAT for calibration. PANDA [96] advances this approach by reducing training data requirements and eliminating dependency on McPAT for power modeling. BOOM-Explorer [73] automates design space exploration for the RISC-V BOOM microarchitecture. Beyond CPUs, XAPP [97] predicts GPU performance by analyzing dynamic and static properties of single-thread CPU code, while Wu et al. [98] modeled GPU power by examining kernel scaling behaviors. SVR-NoC [99] focuses 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. [100] employed an LSTM model to derive insights from historical program counters for cache replacement using an SVM-based predictor. Pythia [101] reimagines prefetching as a reinforcement learning challenge, while Hermes [102] leverages ML to predict off-chip load request outcomes. Additional applications include task allocation [102], power management [103], and resource management for CPU [104] and AI accelerators [105].

In high-level synthesis, the application of ML models for rapidly estimating design metrics has become increasingly prevalent. For instance, Dai et al. [106] focused on timing and resource usage, Pyramid [107] estimates throughput, Ustun et al. [108] looked at operation delay, Zhao et al. [109] considered routing congestion, and Lin et al. [110] dedicated 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. [108], Liu et al. [111], and Meng et al. [112], 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. [113], Mahapatra et al. [114], and Wang et al. [115] demonstrate the integration of ML with traditional optimization algorithms, enhancing their efficacy in navigating complex design spaces. Sun et al. [116] 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. [117] 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 [118] 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. [119] and the work by Sengupta et al. [120], 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 [121] and MasterRTL [122], 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 [123, 124].

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 [125] stands out for offering per-cycle power evaluations tailored to each target design, alongside other notable ML-based approaches [126, 127]. For runtime power modeling, DEEP [128] 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 [129, 130], demonstrates the potential of ML in creating dynamic power models that adapt to real-time conditions. Moreover, APOLLO [131] presents a versatile solution applicable to both design-time and runtime scenarios. Simmani [132] and the early power modeling work [133] 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. [134], offer a probabilistic model-based approach for coverage-based test generation, underscoring the potential of ML in optimizing test planning. Design2Vec [135] 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 [136] 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 optimization [137], quality-of-results (QoR) prediction [138], and more [139,140].

• **ML for logic optimization.** ML-based models have shown significant effectiveness in evaluating synthesis quality and influencing the optimization process. For instance, LSOracle [141] utilizes ML to select the most suitable optimizers for different logic networks, thereby improving overall synthesis outcomes. Additionally, SLAP [142] targets design timing improvement by identifying and utilizing candidate cuts that enhance synthesis results during technology mapping. Their further work [143] demonstrates the ability of ML models to pinpoint post-routing timing critical paths, directing technology mapping efforts to minimize delays. Yu et al. [144] proposed classifying and selecting from multiple random synthesis flows based on their quality, focusing on the most effective ones. Their subsequent research [145] extends to predicting expected delay and area outcomes for synthesis flow candidates, providing a data-driven approach to guide synthesis decisions. Recent advancements, such as AlphaSyn [146], integrate Monte Carlo tree search with tailored learning strategies for area reduction, highlighting the potential of combining ML with heuristic search techniques for synthesis optimization. EasySO [147] proposes a hybrid discrete-continuous action space to jointly optimize the operation sequence and corresponding arguments.

• **ML for QoR prediction.** Following logic synthesis, innovative ML solutions can foresee the post-physical design quality of previously unknown circuit netlists. Tools like Net$^2$ [148] pave the way by predicting wirelength and timing information, effectively capturing the implications of placement on the netlist. PreRoutGNN [149] utilizes global circuit pre-training, local delay learning attentional cell modeling to realize pre-routing timing prediction. GRANNITE [150] advances this further by facilitating the propagation of the RTL toggle rate down to the gate-level netlist, aiming for rapid and accurate

average power estimation. Similarly, GRAPSE [151] evaluates average power based on unoptimized and unmapped netlists, showcasing improvements in both speed and precision of power estimation. Recently, DeepSeq [152] 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 [153] and GNN-RE [154] utilize ML for reverse engineering tasks, such as identifying state registers and deciphering the functionality of subcircuits. Additionally, ABGNN [155] leverages graph neural networks to delineate the boundaries of arithmetic blocks in flattened gate-level netlists, while Gamora [156] 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 [89, 157–160].

• **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 the desired design metrics. Early applications of ML aimed to augment traditional placement strategies. PADE [161] incorporates support vector machines (SVM) and neural networks for datapath extraction and evaluation, facilitating datapath-aware placement strategies. DREAMPlace, developed by Lin et al. [162], 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. [163] applied 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 [164] uses a CNN to forecast the post-routing design rule violations (DRVs), thus avoiding difficult-to-route placements. Another study [165] guides macro placement based on predicted routability. Chang et al. [166] introduced 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. [167] proposed a federated learning-based approach for routability evaluation, addressing data privacy concerns. To achieve better routability prediction performance, Zheng et al. [168] proposed a multimodal neural network Lay-Net, which aggregates both layout and netlist information. The ultimate purpose of routability prediction is to assist routability optimization. Liu et al. [169] incorporated 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 [170] introduces a routing congestion predictor as a plug-in for commercial placers, effectively adjusting cost parameters to mitigate congestion issues. Moreover, Zheng et al. [171] developed 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. [172] and He et al. [173] have implemented tree-based methods, incorporating careful manual feature extraction. TF-Predictor [174] employs Transformers to treat timing paths as sequences, while Guo et al. [175] have devised a customized GNN inspired by static timing analysis mechanisms. Additionally, recent work by Wang et al. [176] addressed the restructuring of netlists due to timing optimization, integrating graph data from netlists with layout image information through multimodal fusion. Moreover, Liang et al. [177] focused 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. [178] proposed 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. [179] was instrumental in predicting PBA from GBA using carefully engineered features and a tree-based model. Subsequent studies, such as [174, 180], 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, studies like IncPIRD [181] and XGBIR [182] concentrate on static IR-drop analysis. In contrast, studies such as [183] 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 [184], GAN-OPC [185], DevelSet [186], and L2O-ILT [187] use various ML methods to improve mask synthesis printability. Other studies, such as those by Watanabe et al. [188], Ye et al. [189], Lin et al. [190], and Chen et al. [191], 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. [192] proposed to extract layout features with discrete cosine transform and utilized 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. [193] proposed to detect multiple hotspots within large layouts simultaneously. In [194], 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 studies like Ciccazzo et al. [195], Nakata et al. [196], and Alawieh et al. [197].

#### 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. [198] introduced 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 [199] utilizes a clustering strategy to automate the adjustment of flow parameters, aiming for enhanced design quality. Furthermore, PTPT [200] presents a multi-objective Bayesian optimization framework 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. [201] proposed 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 [202] underscore this trend, showcasing AI-driven methodologies that enhance PPA metrics across the design spectrum.

In logic synthesis, Liu et al.'s PIMap framework [203] 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 [204], employs a multi-stage multi-armed bandit (MAB) strategy to constrain the search space and streamline the synthesis process. Pei et al.'s AlphaSyn [146], utilizing a domain-specific Monte Carlo tree search (MCTS), and Zhu et al.'s approach [205], 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. [206] and subsequent studies like those by Peruvemba et al. [207] 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. [208]

and Timoneda et al. [209] leveraged 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. [210], to minimizing area and wirelength in floorplanning processes like GoodFloorplan [211]. At the stage of placement and routing, DeepPlace [212] and PRNet [213] incorporate RL to bridge the placement with the subsequent routing task and HubRouter [214] devises a framework that integrates a deep generative model with RL to expedite the resolution of routing problems. Agnesina et al.'s [215] used of RL to tune physical design flows for improved PPA metrics and RL-Sizer by Lu et al. [216] for gate sizing highlight RL's potential to refine physical design processes, including timing optimization [217] and mask optimization in the RL-OPC process [218]. 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 [219] 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 RTLLM [220] and VerilogEval [221], facilitating a more robust comparison of LLM capabilities across complex design tasks. RTLLM 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 [222]. Subsequent developments have introduced tools like ChipGPT [223] and AutoChip [224], 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 [225] 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 ChipNeMo [226], RTL-Coder [227], and BetterV [228] 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, Refs. [229, 230] leveraged human-written specification sentences alongside RTL designs to generate precise SVAs. AssertLLM [231] 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 [232], which can be applied to verify arithmetic circuits.

**Security verification leveraging LLMs.** Security validation, critical in identifying and mitigating common vulnerability enumerations (CWEs), has also benefited from LLM integration. Ahmad et al. [233] demonstrated 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 [234] and employing LLMs in hardware security assertion generation [235]. 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 [236] 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 [237] leverages LLMs to generate C code for AI accelerator high-level synthesis. Similarly, Yan et al. [238] examined 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. [239] delved into quantum architecture design, exploring the frontiers of quantum computing. SpecLLM [240] 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 [241]. Additionally, RL methods have been used to address DRC violations post-routing [242], 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 [243]. 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. [244] employed 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, Ref. [245] developed 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 [246] 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 [247] 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 [248], to address the challenges inherent to analog design-namely, slow performance evaluation, and high search complexity.

**AI for 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. [249], while RL-based methods have been applied to power converters, as demonstrated by Fan et al. [250]. More broadly, Zhao et al. [251] have utilized RL alongside predefined libraries to address a wider array of problems. Poddar et al. [252] 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. To tackle the complexities of large circuit design, hierarchical methods are being investigated. Lu et al. [253] have put forward a bi-level Bayesian optimization technique for $\Delta$-$\Sigma$ modulators, while Fayazi et al. [254] and Hakhamaneshi et al. [255] have delved into intermediate topology representations and GNN models for voltage node prediction, respectively. 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 for 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 [256]. 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 [257]. 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 [258]. 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 [81].

**AI for 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 [259].

For constraint extraction in analog layouts, graph-based methodologies are pivotal for identifying

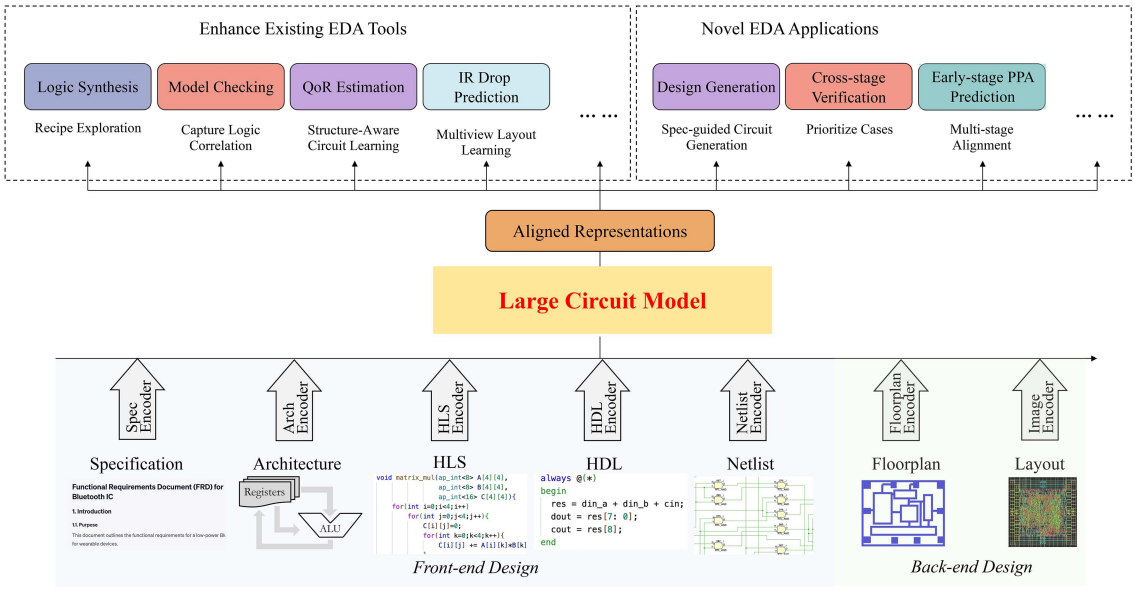


**Figure 4** (Color online) Large circuit models. We call upon the creation of dedicated foundation models for circuits, which intricately intertwine computation with structure, unlike other types of data (e.g., texts and images). Specifically, each design stage of the EDA flow is considered a separate 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.

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 [260]. A detailed survey on these techniques is provided in [261].

ML's role in analog layout extends to automating the imitation of expert designs, modeling circuit performance, and optimizing the layout process. GeniusRoute [262] leverages variational autoencoders for making routing predictions that mimic human expertise, impacting various aspects of layout design including well generation [263], placement strategies [264], and cell generation processes [265]. CNNs and GNNs are utilized for predicting the performance of designs, thereby optimizing placement and minimizing the dependency on extensive simulations [266, 267]. The significant impact of ML on performance-driven placement and optimization in analog layout is thoroughly examined in [268].

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 [269] employs GNNs for accurate parasitic predictions, using ensemble models for specific value ranges. Early performance assertions using CNNs [270] and layout-aware optimization with BagNet [271], 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 [248].

## 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 Figure 4) marks a bold departure from the previous AI4EDA solutions, moving towards a more integrated and AI-rooted 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-rooted 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-rooted 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 [22]. 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-rooted 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 empowers 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-rooted 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-rooted 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 [272] 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 [273] 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 [274] 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 [275] 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. An LCM should aim to blend these disparate strands of knowledge, facilitating an enhanced representation learning in terms of accuracy and interpretability.

At the 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 toward 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) offer 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, 276] 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 [277] and design for testability [278].

In Figure 5, 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 [279] 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.

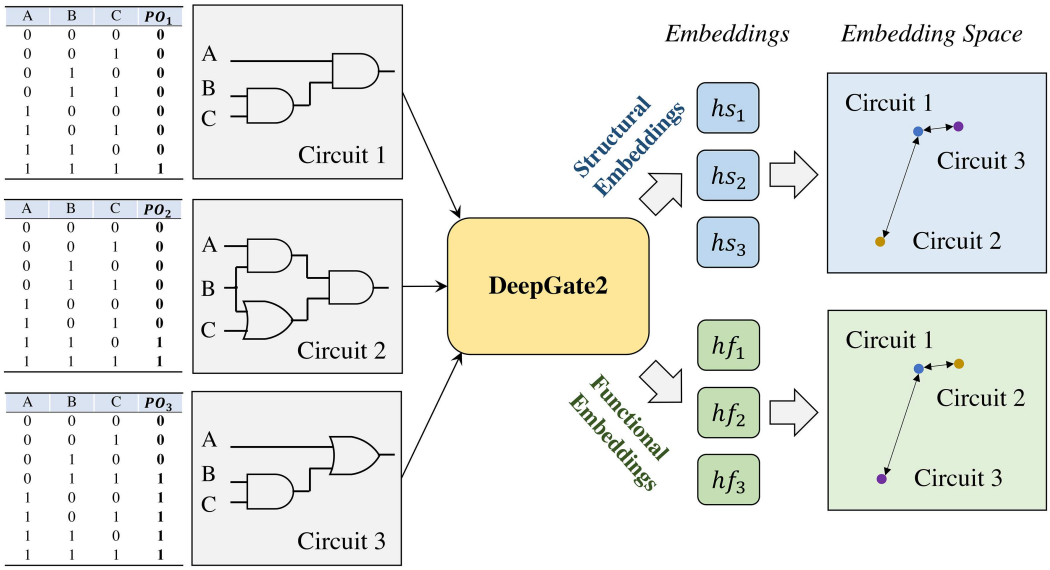


**Figure 5** (Color online) DeepGate2: structural and functional disentangled netlist representation learning.

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 is also a critical facet to represent. DeepSeq [152] 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 the prediction of various factors such as routability, IR drop, and lithography hotspots [164, 185, 280]. Although back-end design consists of many design stages with different levels of geometric abstraction, as shown in Figure 1(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., $256\times256$, $1024\times1024$, $4096\times4096$, 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 toward 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 [281] 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 [235, 282].

### 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 im-

pacting 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.

**Table 1** Solving time comparison between ours and [285] on LEC cases

| Case | Baseline | | | [285] | | | | | | Ours | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | #Vars | #Clas | $\mathcal{T}_{\text{solve}}$ | #Vars | #Clauses | $\mathcal{T}_{\text{trans}}$ | $\mathcal{T}_{\text{solve}}$ | $\mathcal{T}_{\text{all}}$ | Red. (%) | #Vars | #Clas | $\mathcal{T}_{\text{agent}}$ | $\mathcal{T}_{\text{trans}}$ | $\mathcal{T}_{\text{solve}}$ | $\mathcal{T}_{\text{all}}$ | Red. (%) | Red.* (%) |
| I1 | 42069 | 105711 | 322.46 | 5616 | 54529 | 5.31 | 51.49 | 56.80 | 82.39 | 3160 | 31281 | 9.27 | 5.62 | 4.43 | 19.26 | 94.03 | 66.08 |
| I2 | 44949 | 112954 | 708.97 | 6052 | 60573 | 5.61 | 147.85 | 153.46 | 78.35 | 4112 | 41873 | 9.81 | 6.12 | 4.41 | 20.81 | 97.07 | 86.44 |
| I3 | 42038 | 105629 | 531.94 | 5612 | 54825 | 5.21 | 109.89 | 115.10 | 78.36 | 3849 | 37329 | 8.37 | 5.61 | 2.91 | 17.56 | 96.70 | 84.74 |
| I4 | 37275 | 93678 | 289.89 | 5038 | 49805 | 4.61 | 90.05 | 94.66 | 67.35 | 3478 | 34013 | 7.32 | 5.11 | 2.50 | 15.01 | 94.82 | 84.14 |
| I5 | 30087 | 75537 | 172.79 | 4006 | 38069 | 3.91 | 38.77 | 42.67 | 75.30 | 2311 | 22473 | 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** |

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 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 [283,284]. However, the efficacy of CNF-based solvers recently encountered 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 [285], 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, wherein the average values are displayed in bold. In the Baseline setting, the instances are solved directly using the Kissat solver [284]. Let $\mathcal{T}_{\text{agent}}$, $\mathcal{T}_{\text{trans}}$, and $\mathcal{T}_{\text{solve}}$ denote the RL agent runtime, transformation time, and solving time, respectively, measured in seconds. The overall runtime, which sums up all three components, is denoted as $\mathcal{T}_{\text{all}}$ in seconds. Additionally, we list the number of variables (#Vars) and clauses (#Clas), the reduction in $\mathcal{T}_{\text{all}}$ compared to Baseline (Red.) and compared to [285] (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.

**Table 2** Comparing the number of SAT calls and the runtime of the SAT sweepers

| Circuit | Statistic | | | SAT calls | | | Total runtime (s) | | |
|---|---|---|---|---|---|---|---|---|---|
| | PI/PO | #Lev | #And | [290] | Ours | Red. (%) | [290] | Ours | Red. (%) |
| C1 | 128/128 | 4372 | 57247 | 13826 | 570 | 95.88 | 7.46 | 3.15 | 57.77 |
| C2 | 24/25 | 225 | 5416 | 100 | 74 | 26.00 | 5.43 | 3.11 | 42.73 |
| C3 | 22/1 | 29 | 703 | 4 | 1 | 75.00 | 19.07 | 9.10 | 52.28 |
| C4 | 114/1 | 91 | 19354 | 20 | 12 | 40.00 | 6.49 | 4.04 | 37.75 |
| C5 | 126/1 | 83 | 20971 | 6 | 4 | 33.33 | 0.48 | 0.21 | 56.25 |
| C6 | 96/1 | 79 | 14389 | 10 | 5 | 50.00 | 0.30 | 0.15 | 50.00 |
| Avg. | – | – | – | – | – | **53.37** | – | – | **49.46** |

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 various logic representations and sophisticated algorithms. Each representation, such as truth tables [286], sum-of-products [287], binary decision diagrams [288], and directed acyclic graphs (DAGs) [289], is associated with complex algorithms, 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. In this subsection, we aim to demonstrate the impact of the LCMs on logic synthesis, that is, technology-independent logic optimization and technology mapping.

First, as we venture into the realm of nanometer-scale technologies, the importance of technology-independent optimization becomes increasingly apparent. The focus is on metrics such as literals and logic depth used in DAGs for area and delay evaluation. Combining these optimization strategies with the physical realities of the technology landscape introduces new complexities. LCMs are poised to tackle this challenge head-on by more accurately predicting physical characteristics such as timing, area, and power. By integrating physical awareness, LCMs provide a groundbreaking tool for logic optimization, enabling designers to make decisions based on a nuanced understanding of circuit behavior. This foresight not only refines optimization strategies but also promotes superior PPA trade-offs, marking a leap forward in logic synthesis. Recent advancements, such as [17], demonstrate the effectiveness of LCM in capturing complex functional correlations within circuit netlists. We combine LCMs into SAT sweeper to guide equivalence class selection. To be specific, the updated manager sorts all candidate equivalence classes by computing the cosine similarity of their embeddings.

Table 2 presents our preliminary results from applying the netlist encoder to accelerate SAT-sweeping for industrial logic equivalence checking cases C1–C6, wherein the values are displayed in bold. The baseline is one of the most efficient and scalable SAT sweepers publicly available at this time [290]. Let PI/PO, #Lev, and #And denote the number of primary inputs and primary outputs, logic levels, and internal AND-nodes in the original AIG, respectively. The number of satisfiable SAT calls performed by the solver in each engine is defined as SAT calls. Additionally, total runtime compares the runtime, and Red. indicates the runtime reduction from baseline to ours. The sweeper demonstrates a significant reduction of 53.37% in SAT calls and 49.46% in runtime.

Second, a key aspect of technology mapping, especially in FPGA and ASIC design, lies in balancing PPA trade-offs while addressing the constraints of heterogeneous logic blocks, interconnect resources, and optimal cell selection. Tackling structural bias during the technology mapping process requires meticulous algorithmic strategies. LCMs can overcome structural bias through context-aware mapping versatility, which is enhanced by iterative feedback loops and the integration of physical information. Traditional algorithms for physically aware technology mapping necessitate time-consuming online placement to

obtain netlist location information [291]. In contrast, LCMs can learn delay information from gate-level netlists and calculate delay differences between netlists. This approach provides more constraints for physically aware technology mapping and significantly reduces the runtime of the process. The scalability of LCMs further underscores their effectiveness in managing complex circuit designs, presenting a compelling solution to longstanding challenges in technology mapping.

Essentially, the conceptual application of LCMs in logic synthesis promises a shift towards more efficient, accurate, and adaptable design processes, positioning it as a cornerstone of next-generation 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[2)], (2) adapting various encodings from a problem instance to a canonical solver instance [292], and (3) employing design-specific equivalence checking strategies (e.g., for multipliers [35]). 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. The information available within the placement stage includes the circuit's functionality,

2) Cadence. Conformal Smart LEC, 2022.

the topological netlist connections, and the specific locations and dimensions of the standard cells. This information can be represented in multiple modalities, such as networks, point clouds, and images. These modalities can be modeled using distinct data structures, or they can be integrated with other modalities to complement the information. Consider the placement optimization process, where traditional methods have leveraged unimodal information for guidance, from gradient prediction [293] to routing congestion forecasting [164]. 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 [294] introduce dual GNNs to capture both topological (circuit interconnections) and spatial relationships, merging these insights in latent space. Similarly, Lay-Net [168] 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 multimodality features to perceive both 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.

Furthermore, LCMs should be empowered with the capability of strong circuit representation. In the cutting-edge field of natural language processing within artificial intelligence, basic language models can perform shallow reasoning on text, such as identifying text categories. Advanced large language models, however, can understand the intent behind user instructions and perform various tasks, including text generation, contextual reasoning, and natural language question answering. These capabilities derive from vast amounts of natural language data, powerful models (transformers), and advanced training methods (self-supervised learning and reinforcement learning from human feedback). Following this trend, brand-new large circuit model architecture along with more data training can result in significant improvements in capabilities. In other words, LCMs can achieve a deep understanding of circuit structure and apprehend the implementation and optimization processes of physical design methodologies. This strong representation of layout 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 [295, 296]. 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 [297][3)].

### 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.

In datapath design, exploring the vast and complex design space 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.

Inspired by the successes of deep generative models in the fields of computer vision and natural language processing, LCMs leverage datapath unit generation as a non-trivial self-supervision task to learn robust structural representations. Specifically, the encoders of LCMs project the datapath units into a continuous latent space, whereas the decoders of LCMs learn to reconstruct the original datapath units from the latent embedding. In doing so, LCMs not only foster a profound structural comprehension of datapath unit designs but also function as 'neural design libraries', enabling the sampling of distinct datapath units to assemble a comprehensive and optimized datapath design. Unlike traditional IP libraries constrained to utilizing previously seen designs, LCMs possess the capacity to extrapolate the robust structural latent representation, thereby enabling the generation of novel, previously unseen designs.

Furthermore, the profound domain expertise of LCMs in both the internal structure and physical implementations of datapath circuits empowers them to predict design quality at later stages of the design process. In this capacity, LCMs act as 'neural design evaluators', facilitating an extensive 'shift-left' in the evaluation process. It allows for early and insightful assessments that encompass not only architectural considerations but also critical subsequent phases such as placement and routing, significantly boosting the efficiency and effectiveness of the design evaluation process.

Functioning both as neural design libraries and neural design evaluators, LCMs essentially position themselves as 'neural design optimizers'. By assessing design quality at physical design stages and identifying optimization bottlenecks, LCMs use the evaluation results to conditionally sample from the latent space, enabling the identification of optimal designs tailored to current design scenarios. The integration of LCMs into the datapath design process allows engineers to achieve levels of optimization and efficiency previously unattainable, heralding a new era in the evolution of datapath circuits and their implementations in modern microarchitectures.

In summary, LCMs' detailed grasp of datapath complexities allows them to offer strategic recommendations that go beyond design selection and 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

3) Mentor, a Siemens Business. Solido Characterization Suite, 2023.

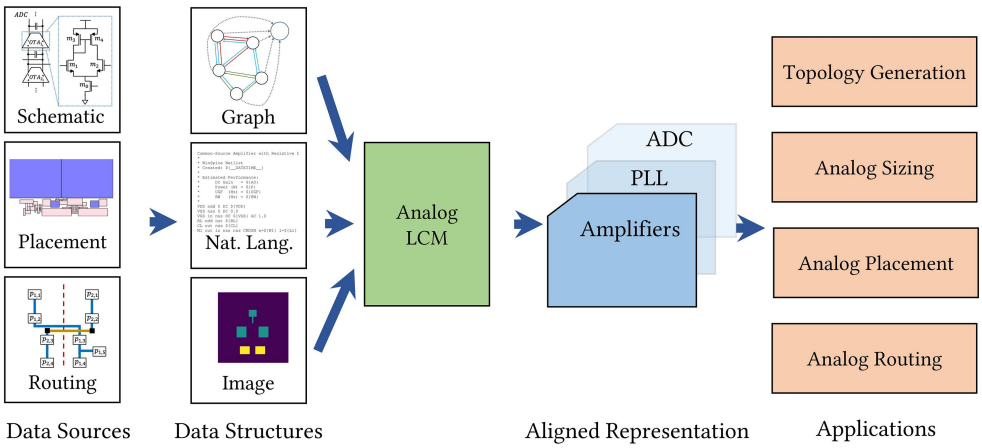


**Figure 6** (Color online) Overview of large circuit models for analog EDA.

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 [248] 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.

Figure 6 presents our vision for future analog large circuit models. These models are inherently multimodal, and 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 circuit augmentation, effectively expanding the dataset without the need for additional real-world data. Different logic synthesis commands can be employed to create various circuit netlists, and the parameters of physical design tools can be adjusted to generate more layouts.

Similar approaches are used in OpenABC-D [298] and CircuitNet [299], which start with only a few RTL designs and produce millions of netlists or layouts. Additionally, hardware design languages can be represented as e-graphs and transformed into different yet functionally equivalent designs using the egg framework [300]. This approach not only enhances the diversity of training samples 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 [227]. 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. For example, DeepGate3 [276] introduces a window-shifting method to enable the fine-tuning of the model on large AIGs. It first cuts a circuit into areas with 512 maximum number of gates, then encodes the large AIG area by area to get the final embedding. Advancements like the above technique contribute 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-rooted 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-rooted 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.

**Acknowledgements** This work was supported in part by Hong Kong S.A.R. General Research Fund (Grant No. 14212422) and Research Matching (Grant No. CSE-7-2022).

**References**


1 Bommasani R, Hudson D A, Adeli E, et al. On the opportunities and risks of foundation models. 2021. ArXiv:2108.07258
2 Devlin J, Chang M W, Lee K, et al. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2019. 4171–4186
3 Liu Y, Ott M, Goyal N, et al. RoBERTa: a robustly optimized BERT pretraining approach. 2019. ArXiv:1907.11692
4 Raffel C, Shazeer N, Roberts A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res, 2020, 21: 5485–5551
5 Brown T, Mann B, Ryder N, et al. Language models are few-shot learners. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), 2020. 33: 1877–1901
6 Chen T, Kornblith S, Norouzi M, et al. A simple framework for contrastive learning of visual representations. In: Proceedings of International Conference on Machine Learning (ICML), 2020. 1597–1607
7 He K, Fan H, Wu Y, et al. Momentum contrast for unsupervised visual representation learning. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020. 9729–9738
8 He K, Chen X, Xie S, et al. Masked autoencoders are scalable vision learners. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 16000–16009
9 Radford A, Kim J W, Hallacy C, et al. Learning transferable visual models from natural language supervision. In: Proceedings of International Conference on Machine Learning (ICML), 2021. 8748–8763
10 Ramesh A, Pavlov M, Goh G, et al. Zero-shot text-to-image generation. In: Proceedings of International Conference on Machine Learning (ICML), 2021. 8821–8831
11 Rombach R, Blattmann A, Lorenz D, et al. High-resolution image synthesis with latent diffusion models. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 10684–10695
12 Kirillov A, Mintun E, Ravi N, et al. Segment anything. 2023. ArXiv:2304.02643
13 Yang Z, Li L, Lin K, et al. The dawn of LMMs: preliminary explorations with GPT-4V(ision). 2023. ArXiv:2309.17421
14 Team G, Anil R, Borgeaud S, et al. Gemini: a family of highly capable multimodal models. 2023. ArXiv:2312.11805
15 Rapp M, Amrouch H, Lin Y, et al. MLCAD: a survey of research in machine learning for CAD keynote paper. IEEE Trans Comput-Aided Des Integr Circ Syst, 2022, 41: 3162–3181
16 Li M, Khan S, Shi Z, et al. DeepGate: learning neural representations of logic gates. In: Proceedings of ACM/IEEE Design Automation Conference, 2022. 667–672
17 Shi Z, Pan H, Khan S, et al. DeepGate2: functionality-aware circuit representation learning. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9
18 Xu Y, Yu Z, Tang D, et al. Towards developing high performance RISC-V processors using agile methodology. In: Proceedings of the 55th IEEE/ACM International Symposium on Microarchitecture (MICRO), 2022. 1178–1199
19 Asanovic K, Avizienis R, Bachrach J, et al. The Rocket Chip Generator. Technical Report UCB/EECS-2016-17, EECS Department, University of California, Berkeley, 2016
20 Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), 2017
21 Zhou J, Cui G, Hu S, et al. Graph neural networks: a review of methods and applications. AI Open, 2020, 1: 57–81
22 Baltrusaitis T, Ahuja C, Morency L P. Multimodal machine learning: a survey and taxonomy. IEEE Trans Pattern Anal Mach Intell, 2019, 41: 423–443
23 Austin J, Odena A, Nye M, et al. Program synthesis with large language models. 2021. ArXiv:2108.07732
24 Lin C C, Chen K C, Chang S C, et al. Logic synthesis for engineering change. In: Proceedings of ACM/IEEE Design Automation Conference, 1995. 647–652
25 Micheli G. Chip challenge. IEEE Solid-State Circ Mag, 2010, 2: 22–26
26 Bachrach J, Vo H, Richards B, et al. Chisel: constructing hardware in a Scala embedded language. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2012. 1216–1225
27 Johnson S C. Lint, a C Program Checker. Murray Hill: Bell Laboratories, 1977
28 Marquez C I C, Strum M, Chau W J. Formal equivalence checking between high-level and RTL hardware designs. In: Proceedings of the 14th Latin American Test Workshop-LATW, 2013. 1–6
29 Mukherjee R, Purandare M, Polig R, et al. Formal techniques for effective co-verification of hardware/software co-designs. In: Proceedings of the 54th Annual Design Automation Conference, 2017. 1–6
30 Synopsys. VC formal datapath validation. 2024. https://www.synopsys.com/verification/static-and-formal-verification/vc-formal/vc-formal-datapath-validation.html
31 Koelbl A, Jacoby R, Jain H, et al. Solver technology for system-level to RTL equivalence checking. In: Proceedings of Design, Automation & Test in Europe Conference & Exhibition, 2009. 196–201
32 Huang B Y, Zhang H, Subramanyan P, et al. Instruction-level abstraction (ILA) a uniform specification for system-on-chip (SoC) verification. ACM Trans Des Autom Electron Syst, 2018, 24: 1–24
33 Mishchenko A, Chatterjee S, Brayton R, et al. Improvements to combinational equivalence checking. In: Proceedings of the IEEE/ACM International Conference on Computer-aided Design, 2006. 836–843
34 Baumgartner J, Mony H, Paruthi V, et al. Scalable sequential equivalence checking across arbitrary design transformations. In: Proceedings of International Conference on Computer Design, 2006. 259–266
35 Chen Z, Zhang X, Qian Y, et al. Integrating exact simulation into sweeping for datapath combinational equivalence checking. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9
36 Dai Y Y, Khoo K Y, Brayton R K. Sequential equivalence checking of clock-gated circuits. In: Proceedings of the 52nd Annual Design Automation Conference, 2015. 1–6
37 Alpert C J, Mehta D P, Sapatnekar S S. Handbook of Algorithms for Physical Design Automation. Boca Raton: CRC Press, 2008
38 Wong D, Leong H W, Liu H. Simulated Annealing for VLSI Design. Berlin: Springer Science & Business Media, 2012
39 Chang Y C, Chang Y W, Wu G M, et al. B*-trees: a new representation for non-slicing floorplans. In: Proceedings of the 37th Annual Design Automation Conference, 2000. 458–463
40 Smith L D, Anderson R E, Forehand D W, et al. Power distribution system design methodology and capacitor selection for modern CMOS technology. IEEE Trans Adv Packag, 1999, 22: 284–291
41 Zhu Q K. Power Distribution Network Design for VLSI. Hoboken: John Wiley & Sons, 2004
42 Sehen C. An improved simulated annealing algorithm for row-based placement. In: Proceedings of IEEE International

Conference on Computer-Aided Design, 1987. 478–481
43 Kleinhans J M, Sigl G, Johannes F M, et al. GORDIAN: VLSI placement by quadratic programming and slicing optimization. IEEE Trans Comput-Aided Des Integr Circ Syst, 1991, 10: 356–365
44 Viswanathan N, Chu C C N. FastPlace: efficient analytical placement using cell shifting, iterative local refinement and a hybrid net model. In: Proceedings of the International Symposium on Physical Design, 2004. 26–33
45 Chen T C, Jiang Z W, Hsu T C, et al. NTUplace3: an analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints. IEEE Trans Comput-Aided Des Integr Circ Syst, 2008, 27: 1228–1240
46 Qiu Y, Xing Y, Zheng X, et al. Progress of placement optimization for accelerating VLSI physical design. Electronics, 2023, 12: 337
47 Chao T H, Hsu Y C, Ho J M, et al. Zero skew clock routing with minimum wirelength. IEEE Trans Circ Syst II, 1992, 39: 799–814
48 Deng C, Cai Y, Zhou Q, et al. An efficient buffer sizing algorithm for clock trees considering process variations. In: Proceedings of the 6th Asia Symposium on Quality Electronic Design (ASQED), 2015. 108–113
49 Liu J, Pui C W, Wang F, et al. CUGR: detailed-routability-driven 3D global routing with probabilistic resource model. In: Proceedings of the 57th ACM/IEEE Design Automation Conference (DAC), 2020. 1–6
50 Liu W H, Kao W C, Li Y L, et al. NCTU-GR 2.0: multithreaded collision-aware global routing with bounded-length maze routing. IEEE Trans Comput-Aided Des Integr Circ Syst, 2013, 32: 709–722
51 Betz V, Rose J. VPR: a new packing, placement and routing tool for FPGA research. In: Proceedings of International Workshop on Field Programmable Logic and Applications, 1997. 213–222
52 Hu J, Sapatnekar S S. A survey on multi-net global routing for integrated circuits. Integration, 2001, 31: 1–49
53 Stojilović M. Parallel FPGA routing: survey and challenges. In: Proceedings of the 27th International Conference on Field Programmable Logic and Applications (FPL), 2017. 1–8
54 Li X, Huang Z, Tao S, et al. iEDA: an open-source infracstructure of EDA. In: Proceedings of Asia and South Pacific Design Automation Conference (ASPDAC), 2024
55 Li Y L, Lin S T, Nishizawa S, et al. NCTUcell: a DDA-aware cell library generator for FinFET structure with implicitly adjustable grid map. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2019. 1–6
56 Cheng C K, Ho C T, Lee D, et al. A routability-driven complimentary-FET (CFET) standard cell synthesis framework using SMT. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020. 1–8
57 Park D, Lee D, Kang I, et al. SP&R: simultaneous placement and routing framework for standard cell synthesis in sub-7nm. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2020. 345–350
58 Choi S, Jung J, Kahng A B, et al. PROBE3.0: a systematic framework for design-technology pathfinding with improved design enablement. IEEE Trans Comput-Aided Des Integr Circ Syst, 2024, 43: 1218–1231
59 Beaumont-Smith A, Lim C C. Parallel prefix adder design. In: Proceedings of the 15th IEEE Symposium on Computer Arithmetic, 2001. 218–225
60 Rakesh S, Grace K S V. A comprehensive review on the VLSI design performance of different Parallel Prefix Adders. Mater Today-Proc, 2019, 11: 1001–1009
61 Liu J, Zhou S, Zhu H, et al. An algorithmic approach for generic parallel adders. In: Proceedings of International Conference on Computer Aided Design, 2003. 734–740
62 Matsunaga T, Matsunaga Y. Area minimization algorithm for parallel prefix adders under bitwise delay constraints. In: Proceedings of the 17th ACM Great Lakes Symposium on VLSI, 2007. 435–440
63 Roy S, Choudhury M, Puri R, et al. Towards optimal performance-area trade-off in adders by synthesis of parallel prefix structures. In: Proceedings of the 50th Annual Design Automation Conference, 2013. 1–8
64 Wallace C S. A suggestion for a fast multiplier. IEEE Trans Electron Comput, 1964, EC-13: 14–17
65 Dadda L. Some Schemes for Parallel Multipliers. Palo Alto: IEEE Computer Society Press, 1990
66 Xiao W, Qian W, Liu W. GOMIL: global optimization of multiplier by integer linear programming. In: Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), 2021. 374–379
67 Synopsys. Designware ip. http://www.synopsys.com/designware, 2015
68 Corporation S P E. SPEC CPU 2017 benchmark. https://www.spec.org/cpu2017/
69 Haseeb M, Saeed F. High performance computing framework for tera-scale database search of mass spectrometry data. Nat Comput Sci, 2021, 1: 550–561
70 Buchmann J. Introduction to Cryptography. New York: Springer, 2004
71 Mulgrew B, Grant P, Thompson J. Digital Signal Processing: Concepts and Applications. London: Red Globe Press, 2002
72 Parashar A, Raina P, Shao Y S, et al. Timeloop: a systematic approach to DNN accelerator evaluation. In: Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2019. 304–315
73 Bai C, Sun Q, Zhai J, et al. BOOM-Explorer: RISC-V BOOM microarchitecture design space exploration framework. In: Proceedings of IEEE/ACM International Conference On Computer Aided Design (ICCAD), 2021
74 Chen S, Zheng S, Bai C, et al. SoC-Tuner: an importance-guided exploration framework for DNN-targeting soc design. In: Proceedings of the 29th Asia and South Pacific Design Automation Conference (ASP-DAC), 2024. 207–212
75 Venieris S I, Bouganis C S. fpgaConvNet: a framework for mapping convolutional neural networks on FPGAs. In: Proceedings of IEEE 24th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2016. 40–47
76 Wang J, Cong J. Search for optimal systolic arrays: a comprehensive automated exploration framework and lessons learned. 2021. ArXiv:2111.14252
77 Zhang D, Huda S, Songhori E, et al. A full-stack search technique for domain optimized deep learning accelerators. In: Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2022. 27–42
78 Bai C, Huang J, Wei X, et al. ArchExplorer: microarchitecture exploration via bottleneck analysis. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2023. 268–282
79 Dave S, Nowatzki T, Shrivastava A. Explainable-DSE: an agile and explainable exploration of efficient HW/SW codesigns of deep learning accelerators using bottleneck analysis. In: Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2023. 87–107
80 Budak A F, Pan D Z, Chen H, et al. CAD for Analog/Mixed-Signal Integrated Circuits. Piscataway: Wiley Press, 2023. 43–60
81 Budak A F, Zhang S, Liu M, et al. Machine Learning for Analog Circuit Sizing. Cham: Springer International Publishing, 2022. 307–335

82 Chen H, Liu M, Tang X, et al. Challenges and opportunities toward fully automated analog layout design. J Semicond, 2020, 41: 111407
83 Zhao Z, Zhang L. An automated topology synthesis framework for analog integrated circuits. IEEE Trans Comput-Aided Des Integr Circ Syst, 2020, 39: 4325–4337
84 Lyu W, Xue P, Yang F, et al. An efficient Bayesian optimization approach for automated optimization of analog circuits. IEEE Trans Circ Syst I, 2018, 65: 1954–1967
85 Zhu K, Chen H, Liu M, et al. Effective analog/mixed-signal circuit placement considering system signal flow. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020
86 Wu N, Xie Y, Hao C. AI-assisted synthesis in next generation EDA: promises, challenges, and prospects. In: Proceedings of IEEE 40th International Conference on Computer Design (ICCD), 2022. 207–214
87 Goswami P, Bhatia D. Application of machine learning in FPGA EDA tool development. IEEE Access, 2023, 11: 109564
88 Koblah D, Acharya R, Capecci D, et al. A survey and perspective on artificial intelligence for security-aware electronic design automation. ACM Trans Des Autom Electron Syst, 2023, 28: 1–57
89 Huang G, Hu J, He Y, et al. Machine learning for electronic design automation: a survey. ACM Trans Des Autom Electron Syst, 2021, 26: 1–46
90 Lopera D S, Servadei L, Kiprit G N, et al. A survey of graph neural networks for electronic design automation. In: Proceedings of ACM/IEEE 3rd Workshop on Machine Learning for CAD (MLCAD), 2021. 1–6
91 Lin Y, Ziv A, Ren H. Introduction to the special issue on machine learning for CAD/EDA. ACM Trans Des Autom Electron Syst, 2023, 28: 1–2
92 Ren H, Hu J. Machine Learning Applications in Electronic Design Automation. Berlin: Springer, 2022
93 Joseph P, Vaswani K, Thazhuthaveetil M J. Construction and use of linear regression models for processor performance analysis. In: Proceedings of IEEE International Symposium on High Performance Computer Architecture (HPCA), 2006
94 Mendis C, Renda A, Amarasinghe S, et al. Ithemal: accurate, portable and fast basic block throughput estimation using deep neural networks. In: Proceedings of International Conference on Machine Learning (ICML), 2019
95 Zhai J, Bai C, Zhu B, et al. McPAT-Calib: a microarchitecture power modeling framework for modern CPUs. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2021. 1–9
96 Zhang Q, Li S, Zhou G, et al. PANDA: architecture-level power evaluation by unifying analytical and machine learning solutions. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2023. 1–9
97 Ardalani N, Lestourgeon C, Sankaralingam K, et al. Cross-architecture performance prediction (XAPP) using CPU code to predict GPU performance. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2015
98 Wu G, Greathouse J L, Lyashevsky A, et al. GPGPU performance and power estimation using machine learning. In: Proceedings of IEEE International Symposium on High Performance Computer Architecture (HPCA), 2015
99 Qian Z, Juan D C, Bogdan P, et al. SVR-NoC: a performance analysis tool for network-on-chips using learning-based support vector regression model. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2013. 354–357
100 Shi Z, Huang X, Jain A, et al. Applying deep learning to the cache replacement problem. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2019. 413–425
101 Bera R, Kanellopoulos K, Nori A, et al. Pythia: a customizable hardware prefetching framework using online reinforcement learning. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2021
102 Lu S, Tessier R, Burleson W. Reinforcement learning for thermal-aware many-core task allocation. In: Proceedings of Great Lakes Symposium on VLSI, 2015
103 AbouGhazaleh N, Ferreira A, Rusu C, et al. Integrated CPU and L2 cache voltage scaling using machine learning. In: Proceedings of ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES), 2007
104 Dubach C, Jones T M, Bonilla E V, et al. A predictive model for dynamic microarchitectural adaptivity control. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2010. 485–496
105 Kao S C, Jeong G, Krishna T. ConfuciuX: autonomous hardware resource assignment for DNN accelerators using reinforcement learning. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2020. 622–636
106 Dai S, Zhou Y, Zhang H, et al. Fast and accurate estimation of quality of results in high-level synthesis with machine learning. In: Proceedings of Symposium on Field-Programmable Custom Computing Machines (FCCM), 2018
107 Makrani H M, Farahmand F, Sayadi H, et al. Pyramid: machine learning framework to estimate the optimal timing and resource usage of a high-level synthesis design. In: Proceedings of International Conference on Field-Programmable Logic and Applications (FPL), 2019
108 Ustun E, Deng C, Pal D, et al. Accurate operation delay prediction for FPGA HLS using graph neural networks. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020. 1–9
109 Zhao J, Liang T, Sinha S, et al. Machine learning based routing congestion prediction in FPGA high-level synthesis. In: Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), 2019. 1130–1135
110 Lin Z, Yuan Z, Zhao J, et al. PowerGear: early-stage power estimation in FPGA HLS via heterogeneous edge-centric GNNs. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2022. 1341–1346
111 Liu H Y, Carloni L P. On learning-based methods for design-space exploration with high-level synthesis. In: Proceedings of Design Automation Conference (DAC), 2013
112 Meng P, Althoff A, Gautier Q, et al. Adaptive threshold non-Pareto elimination: re-thinking machine learning for system level design space exploration on FPGAs. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2016. 918–923
113 Kim R G, Doppa J R, Pande P P. Machine learning for design space exploration and optimization of manycore systems. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018. 1–6
114 Mahapatra A, Schafer B C. Machine-learning based simulated annealer method for high level synthesis design space exploration. In: Proceedings of Electronic System Level Synthesis Conference (ESLsyn), 2014. 1–6
115 Wang Z, Schafer B C. Machine leaming to set meta-heuristic specific parameters for high-level synthesis design space exploration. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2020. 1–6
116 Sun Q, Chen T, Liu S, et al. Correlated multi-objective multi-fidelity optimization for HLS directives design. ACM Trans Des Autom Electron Syst, 2022, 27: 1–27
117 Yu Z, Bail C, Hu S, et al. IT-DSE: invariance risk minimized transfer microarchitecture design space exploration. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9

118 Xiao Q, Zheng S, Wu B, et al. HASCO: towards agile hardware and software co-design for tensor computation. In: Proceedings of IEEE/ACM International Symposium on Computer Architecture (ISCA), 2021. 1055–1068
119 Xu C, Kjellqvist C, Wills L W. SNS's not a synthesizer: a deep-learning-based synthesis predictor. In: Proceedings of International Symposium on Computer Architecture (ISCA), 2022
120 Sengupta P, Tyagi A, Chen Y, et al. 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
121 Xu C, Sharma P, Wang T, et al. Fast, robust and transferable prediction for hardware logic synthesis. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture, 2023. 167–179
122 Fang W, Lu Y, Liu S, et al. MasterRTL: a pre-synthesis PPA estimation framework for any RTL design. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2023
123 Lopera D S, Ecker W. Applying GNNs to timing estimation at RTL. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2022
124 Wu N, Lee J, Xie Y, et al. LOSTIN: logic optimization via spatio-temporal information with hybrid graph models. In: Proceedings of International Conference on Application-specific Systems, Architectures and Processors (ASAP), 2022
125 Zhou Y, Ren H, Zhang Y, et al. PRIMAL: power inference using machine learning. In: Proceedings of Design Automation Conference (DAC), 2019
126 Lee D, John L K, Gerstlauer A. Dynamic power and performance back-annotation for fast and accurate functional hardware simulation. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2015
127 Kumar A K A, Gerstlauer A. Learning-based CPU power modeling. In: Proceedings of ACM/IEEE Workshop on Machine Learning for CAD (MLCAD), 2019
128 Xie Z, Li S, Ma M, et al. DEEP: developing extremely efficient runtime on-chip power meters. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2022
129 Zoni D, Cremona L, Fornaciari W. PowerProbe: run-time power modeling through automatic RTL instrumentation. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2018
130 Pagliari D J, Peluso V, Chen Y, et al. ALL-digital embedded meters for on-line power estimation. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2018
131 Xie Z, Xu X, Walker M, et al. APOLLO: an automated power modeling framework for runtime power introspection in high-volume commercial microprocessors. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2021
132 Kim D, Zhao J, Bachrach J, et al. Simmani: runtime power modeling for arbitrary RTL with automatic signal selection. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO), 2019
133 Yang J, Ma L, Zhao K, et al. Early stage real-time SoC power estimation using RTL instrumentation. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2015
134 Fine S, Ziv A. Coverage directed test generation for functional verification using Bayesian networks. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2003
135 Vasudevan S, Jiang W J, Bieber D, et al. Learning semantic representations to verify hardware designs. In: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), 2021. 34: 23491–23504
136 Katz Y, Rimon M, Ziv A, et al. Learning microarchitectural behaviors to improve stimuli generation quality. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2011
137 Rai S, Neto W L, Miyasaka Y, et al. Logic synthesis meets machine learning: trading exactness for generalization. In: Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), 2021. 1026–1031
138 Kahng A B, Wang Z. ML for design QoR prediction. In: Proceedings of Machine Learning Applications in Electronic Design Automation, 2022
139 Gogri S, Hu J, Tyagi A, et al. Machine learning-guided stimulus generation for functional verification. In: Proceedings of the Design and Verification Conference (DVCON-USA), 2020. 2–5
140 Xie Z, Pan J, Chang C C, et al. The dark side: security and reliability concerns in machine learning for EDA. IEEE Trans Comput-Aided Des Integr Circ Syst, 2022, 42: 1171–1184
141 Neto W L, Austin M, Temple S, et al. LSOracle: a logic synthesis framework driven by artificial intelligence. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019. 1–6
142 Neto W L, Moreira M T, Li Y, et al. SLAP: a supervised learning approach for priority cuts technology mapping. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2021. 859–864
143 Neto W L, Moreira M T, Amaru L, et al. Read your circuit: leveraging word embedding to guide logic optimization. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2021. 530–535
144 Yu C, Xiao H, Micheli G. Developing synthesis flows without human knowledge. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2018
145 Yu C, Zhou W. Decision making in synthesis cross technologies using LSTMs and transfer learning. In: Proceedings of ACM/IEEE Workshop on Machine Learning for CAD (MLCAD), 2020. 55–60
146 Pei Z, Liu F, He Z, et al. AlphaSyn: logic synthesis optimization with efficient Monte Carlo tree search. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9
147 Yuan J, Wang P, Ye J, et al. EasySO: exploration-enhanced reinforcement learning for logic synthesis sequence optimization and a comprehensive RL environment. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9
148 Xie Z, Liang R, Xu X, et al. Preplacement net length and timing estimation by customized graph neural network. IEEE Trans Comput-Aided Des Integr Circ Syst, 2022, 41: 4667–4680
149 Zhong R, Ye J, Tang Z, et al. PreRoutGNN for timing prediction with order preserving partition: global circuit pre-training, local delay learning and attentional cell modeling. AAAI, 2024, 38: 17087–17095
150 Zhang Y, Ren H, Khailany B. GRANNITE: graph neural network inference for transferable power estimation. In: Proceedings of Design Automation Conference (DAC), 2020
151 Rakesh M, Das P, Terkar A, et al. GRASPE: accurate post-synthesis power estimation from RTL using graph representation learning. In: Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), 2023. 1–5
152 Khan S, Shi Z, Li M, et al. DeepSeq: deep sequential circuit learning. 2023. ArXiv:2302.13608
153 Chowdhury S D, Yang K, Nuzzo P. ReIGNN: state register identification using graph neural networks for circuit reverse engineering. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2021. 1–9
154 Alrahis L, Sengupta A, Knechtel J, et al. GNN-RE: graph neural networks for reverse engineering of gate-level netlists.

IEEE Trans Comput-Aided Des Integr Circ Syst, 2021, 41: 2435–2448
155 He Z, Wang Z, Bail C, et al. Graph learning-based arithmetic block identification. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2021. 1–8
156 Wu N, Li Y, Hao C, et al. Gamora: graph learning based symbolic reasoning for large-scale Boolean networks. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023
157 Shin Y. AI-EDA: toward a holistic approach to AI-powered EDA. In: Proceedings of ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD), 2023. 1–3
158 Amuru D, Zahra A, Vudumula H V, et al. AI/ML algorithms and applications in VLSI design and technology. Integration, 2023, 93: 102048
159 Li W, Chen G, Yang H, et al. Learning point clouds in EDA. In: Proceedings of the International Symposium on Physical Design, 2021. 55–62
160 Chen T, Zhang G L, Yu B, et al. Machine learning in advanced IC design: a methodological survey. IEEE Des Test, 2023, 40: 17–33
161 Ward S, Ding D, Pan D Z. PADE: a high-performance placer with automatic datapath extraction and evaluation through high dimensional data learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2012. 756–761
162 Lin Y, Dhar S, Li W, et al. DREAMPlace: deep learning toolkit-enabled GPU acceleration for modern VLSI placement. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2019. 1–6
163 Agnesina A, Rajvanshi P, Yang T, et al. AutoDMP: automated DREAMPlace-based macro placement. In: Proceedings of ACM International Symposium on Physical Design (ISPD), 2023
164 Xie Z, Huang Y H, Fang G Q, et al. RouteNet: routability prediction for mixed-size designs using convolutional neural network. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018
165 Huang Y H, Xie Z, Fang G Q, et al. Routability-driven macro placement with embedded CNN-based prediction model. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2019
166 Chang C C, Pan J, Zhang T, et al. Automatic routability predictor development using neural architecture search. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2021
167 Pan J, Chang C C, Xie Z, et al. Towards collaborative intelligence: routability estimation based on decentralized private data. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2022
168 Zheng S, Zou L, Xu P, et al. Lay-Net: grafting netlist knowledge on layout-based congestion prediction. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9
169 Liu S, Sun Q, Liao P, et al. Global placement with deep learning-enabled explicit routability optimization. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE), 2021. 1821–1824
170 Chen J, Kuang J, Zhao G, et al. PROS: a plug-in for routability optimization applied in the state-of-the-art commercial EDA tool using deep learning. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020
171 Zheng S, Zou L, Liu S, et al. Mitigating distribution shift for congestion optimization in global placement. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
172 Barboza E C, Shukla N, Chen Y, et al. Machine learning-based pre-routing timing prediction with reduced pessimism. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2019
173 He X, Fu Z, Wang Y, et al. Accurate timing prediction at placement stage with look-ahead RC network. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2022. 1213–1218
174 Cao P, He G, Yang T. TF-Predictor: transformer-based prerouting path delay prediction framework. IEEE Trans Comput-Aided Des Integr Circ Syst, 2023, 42: 2227–2237
175 Guo Z, Liu M, Gu J, et al. A timing engine inspired graph neural network model for pre-routing slack prediction. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2022. 1207–1212
176 Wang Z, Liu S, Pu Y, et al. Restructure-tolerant timing prediction via multimodal fusion. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
177 Liang R, Xie Z, Jung J, et al. Routing-free crosstalk prediction. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020
178 Liu S, Wang Z, Liu F, et al. Concurrent sign-off timing optimization via deep Steiner points refinement. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
179 Kahng A B, Mallappa U, Saul L. Using machine learning to predict path-based slack from graph-based timing analysis. In: Proceedings of IEEE International Conference on Computer Design (ICCD), 2018. 603–612
180 Ye Y, Chen T, Gao Y, et al. Graph-learning-driven path-based timing analysis results predictor from graph-based timing analysis. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2023. 547–552
181 Ho C T, Kahng A B. IncPIRD: fast learning-based prediction of incremental IR drop. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019
182 Pao C H, Su A Y, Lee Y M. XGBIR: an XGBoost-based IR drop predictor for power delivery network. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2020. 1307–1310
183 Fang Y C, Lin H Y, Sui M Y, et al. Machine-learning-based dynamic IR drop prediction for ECO. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018. 1–7
184 Alawieh M B, Lin Y, Zhang Z, et al. GAN-SRAF: subresolution assist feature generation using generative adversarial networks. IEEE Trans Comput-Aided Des Integr Circ Syst, 2020, 40: 373–385
185 Yang H, Li S, Deng Z, et al. GAN-OPC: mask optimization with lithography-guided generative adversarial nets. IEEE Trans Comput-Aided Des Integr Circ Syst, 2020, 39: 2822–2834
186 Chen G, Yu Z, Liu H, et al. DevelSet: deep neural level set for instant mask optimization. IEEE Trans Comput-Aided Des Integr Circ Syst, 2023, 42: 5020–5033
187 Zhu B, Zheng S, Yu Z, et al. L2O-ILT: learning to optimize inverse lithography techniques. IEEE Trans Comput-Aided Des Integr Circ Syst, 2024, 43: 944–955
188 Watanabe Y, Kimura T, Matsunawa T, et al. Accurate lithography simulation model based on convolutional neural networks. In: Proceedings of SPIE, 2017. 137–145
189 Ye W, Alawieh M B, Lin Y, et al. LithoGAN: end-to-end lithography modeling with generative adversarial networks. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2019
190 Lin Y, Li M, Watanabe Y, et al. Data efficient lithography modeling with transfer learning and active data selection. IEEE Trans Comput-Aided Des Integr Circ Syst, 2018, 38: 1900–1913

191 Chen G, Pei Z, Yang H, et al. Physics-informed optical kernel regression using complex-valued neural fields. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
192 Yang H, Luo L, Su J, et al. Imbalance aware lithography hotspot detection: a deep learning approach. J Micro Nanolith MEMS MOEMS, 2017, 16: 033504
193 Chen J, Lin Y, Guo Y, et al. Lithography hotspot detection using a double inception module architecture. J Micro Nanolith MEMS MOEMS, 2019, 18: 013507
194 Jiang Y, Yang F, Yu B, et al. Efficient layout hotspot detection via binarized residual neural network ensemble. IEEE Trans Comput-Aided Des Integr Circ Syst, 2020, 40: 1476–1488
195 Ciccazzo A, Pillo G D, Latorre V. A SVM surrogate model-based method for parametric yield optimization. IEEE Trans Comput-Aided Des Integr Circ Syst, 2015, 35: 1224–1228
196 Nakata K, Orihara R, Mizuoka Y, et al. A comprehensive big-data-based monitoring system for yield enhancement in semiconductor manufacturing. IEEE Trans Semicond Manufact, 2017, 30: 339–344
197 Alawieh M B, Boning D, Pan D Z. Wafer map defect patterns classification using deep selective learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2020. 1–6
198 Kwon J, Ziegler M M, Carloni L P. A learning-based recommender system for autotuning design flows of industrial high-performance processors. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2019
199 Xie Z, Fang G Q, Huang Y H, et al. FIST: a feature-importance sampling and tree-based method for automatic design flow parameter tuning. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2020
200 Geng H, Chen T, Ma Y, et al. PTPT: physical design tool parameter tuning via multi-objective Bayesian optimization. IEEE Trans Comput-Aided Des Integr Circ Syst, 2022, 42: 178–189
201 Cho M, Yuan K, Ban Y, et al. ELIAD: efficient lithography aware detailed routing algorithm with compact and macro post-OPC printability prediction. IEEE Trans Comput-Aided Des Integr Circ Syst, 2009, 28: 1006–1016
202 Synopsys. Synopsys.ai unveiled as industry's first full-stack, AI-driven EDA suite for chipmakers. 2023. https://news.synopsys.com/2023-03-29-Synopsys-ai-Unveiled-as-Industrys-First-Full-Stack,-AI-Driven-EDA-Suite-for-Chipmakers
203 Liu G, Zhang Z. PIMap: a flexible framework for improving LUT-based technology mapping via parallelized iterative optimization. ACM Trans Reconfig Technol Syst, 2018, 11: 1–23
204 Yu C. FlowTune: practical multi-armed bandits in Boolean optimization. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020. 1–9
205 Zhu K, Liu M, Chen H, et al. Exploring logic optimizations with reinforcement learning and graph convolutional network. In: Proceedings of ACM/IEEE Workshop on Machine Learning for CAD (MLCAD), 2020. 145–150
206 Hosny A, Hashemi S, Shalan M, et al. DRiLLS: deep reinforcement learning for logic synthesis. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC), 2020. 581–586
207 Peruvemba Y V, Rai S, Ahuja K, et al. RL-guided runtime-constrained heuristic exploration for logic synthesis. In: Proceedings of IEEE/ACM International Conference On Computer Aided Design (ICCAD), 2021. 1–9
208 Haaswijk W, Collins E, Seguin B, et al. Deep learning for logic optimization algorithms. In: Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), 2018. 1–4
209 Timoneda X, Cavigelli L. Late breaking results: reinforcement learning for scalable logic optimization with graph neural networks. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2021. 1378–1379
210 Mirhoseini A, Goldie A, Yazgan M, et al. A graph placement methodology for fast chip design. Nature, 2021, 594: 207–212
211 Xu Q, Geng H, Chen S, et al. GoodFloorplan: graph convolutional network and reinforcement learning-based floorplanning. IEEE Trans Comput-Aided Des Integr Circ Syst, 2021, 41: 3492–3502
212 Cheng R, Yan J. On joint learning for solving placement and routing in chip design. In: Proceedings of Advances in Neural Information Processing Systems, 2021. 34: 16508–16519
213 Cheng R, Lyu X, Li Y, et al. The policy-gradient placement and generative routing neural networks for chip design. In: Proceedings of Advances in Neural Information Processing Systems, 2022. 35: 26350–26362
214 Du X, Wang C, Zhong R, et al. HubRouter: learning global routing via hub generation and pin-hub connection. In: Proceedings of Advances in Neural Information Processing Systems, 2024
215 Agnesina A, Chang K, Lim S K. VLSI placement parameter optimization using deep reinforcement learning. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design, 2020. 1–9
216 Lu Y C, Nath S, Khandelwal V, et al. RL-Sizer: VLSI gate sizing for timing optimization using deep reinforcement learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2021. 733–738
217 Lu Y C, Chan W T, Guo D, et al. RL-CCD: concurrent clock and data optimization using attention-based self-supervised reinforcement learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
218 Liang X, Ouyang Y, Yang H, et al. RL-OPC: mask optimization with deep reinforcement learning. IEEE Trans Comput-Aided Des Integr Circ Syst, 2024, 43: 340–351
219 Lu Y C, Lee J, Agnesina A, et al. GAN-CTS: a generative adversarial framework for clock tree prediction and optimization. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019
220 Lu Y, Liu S, Zhang Q, et al. RTLLM: an open-source benchmark for design RTL generation with large language model. 2023. ArXiv:2308.05345
221 Liu M, Pinckney N, Khailany B, et al. VerilogEval: evaluating large language models for Verilog code generation. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2023
222 Liang X. Hardware descriptions code completion based on a pre-training model. In: Proceedings of IEEE Conference on Telecommunications, Optics and Computer Science (TOCS), 2021. 228–232
223 Chang K, Wang Y, Ren H, et al. ChipGPT: how far are we from natural language hardware design. 2023. ArXiv:2305.14019
224 Thakur S, Blocklove J, Pearce H, et al. AutoChip: automating HDL generation using LLM feedback. 2023. ArXiv:2311.04887
225 Blocklove J, Garg S, Karri R, et al. Chip-Chat: challenges and opportunities in conversational hardware design. In: Proceedings of ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD), 2023
226 Liu M, Ene T D, Kirby R, et al. ChipNeMo: domain-adapted LLMs for chip design. 2023. ArXiv:2311.00176
227 Liu S, Fang W, Lu Y, et al. RTLCoder: outperforming GPT-3.5 in design RTL generation with our open-source dataset and lightweight solution. 2023. ArXiv:2312.08617
228 Pei Z, Zhen H L, Yuan M, et al. BetterV: controlled Verilog generation with discriminative guidance. 2024. ArXiv:2402.03375
229 Orenes-Vera M, Martonosi M, Wentzlaff D. Using LLMs to facilitate formal verification of RTL. 2023. ArXiv:2309.09437
230 Sun C, Hahn C, Trippel C. Towards improving verification productivity with circuit-aware translation of natural language to SystemVerilog assertions. In: Proceedings of the 1st International Workshop on Deep Learning-aided Verification (DAV),

2023
231 Fang W, Li M, Li M, et al. AssertLLM: generating and evaluating hardware verification assertions from design specifications via multi-LLMs. 2024. ArXiv:2402.00386
232 Zhang Y, Zhen H L, Pei Z, et al. SoLA: solver-layer adaption of LLM for better logic reasoning. 2024. ArXiv:2402.11903
233 Ahmad B, Thakur S, Tan B, et al. Fixing hardware security bugs with large language models. 2023. ArXiv:2302.01215
234 Nair M, Sadhukhan R, Mukhopadhyay D. Generating secure hardware using ChatGPT resistant to CWEs. Cryptology ePrint Archive, 2023. https://eprint.iacr.org/2023/212
235 Kande R, Pearce H, Tan B, et al. LLM-assisted generation of hardware assertions. 2023. ArXiv:2306.14027v1
236 He Z, Wu H, Zhang X, et al. ChatEDA: a large language model powered autonomous agent for EDA. 2023. ArXiv:2308.10204
237 Fu Y, Zhang Y, Yu Z, et al. GPT4AIGChip: towards next-generation AI accelerator design automation via large language models. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2023. 1–9
238 Yan Z, Qin Y, Hu X S, et al. On the viability of using LLMs for SW/HW co-design: an example in designing CiM DNN accelerators. 2023. ArXiv:2306.06923
239 Liang Z, Cheng J, Yang R, et al. Unleashing the potential of LLMs for quantum computing: a study in quantum architecture design. 2023. ArXiv:2307.08191
240 Li M, Fang W, Zhang Q, et al. SpecLLM: exploring generation and review of VLSI design specification with large language model. 2024. ArXiv:2401.13266
241 Ren H, Fojtik M. Invited-NVCell: standard cell layout in advanced technology nodes with reinforcement learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2021. 1291–1294
242 Ren H, Fojtik M. Standard cell routing with reinforcement learning and genetic algorithm in advanced technology nodes. In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2021. 684–689
243 Liang A C W, Wen C H P, Huang H M. A general and automatic cell layout generation framework with implicit learning on design rules. IEEE Trans VLSI Syst, 2022, 30: 1341–1354
244 Roy S, Ma Y, Miao J, et al. A learning bridge from architectural synthesis to physical design for exploring power efficient high-performance adders. In: Proceedings of IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), 2017. 1–6
245 Geng H, Ma Y, Xu Q, et al. High-speed adder design space exploration via graph neural processes. IEEE Trans Comput-Aided Des Integr Circ Syst, 2022, 41: 2657–2670
246 Cheng J, Xiao Y, Shao Y, et al. Machine-learning-driven architectural selection of adders and multipliers in logic synthesis. ACM Trans Des Autom Electron Syst, 2023, 28: 1–16
247 Zuo D, Ouyang Y, Ma Y. RL-MUL: multiplier design optimization with deep reinforcement learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
248 Zhu K, Chen H, Turner W J, et al. TAG: learning circuit spatial embedding from layouts. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2022
249 Lu J, Lei L, Yang F, et al. Topology optimization of operational amplifier in continuous space via graph embedding. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2022. 142–147
250 Fan S, Cao N, Zhang S, et al. From specification to topology: automatic power converter design via reinforcement learning. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2021
251 Zhao Z, Luo J, Liu J, et al. Signal-division-aware analog circuit topology synthesis aided by transfer learning. IEEE Trans Comput-Aided Des Integr Circ Syst, 2023, 42: 3481–3490
252 Poddar S, Budak A, Zhao L, et al. A data-driven analog circuit synthesizer with automatic topology selection and sizing. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2024
253 Lu J, Li Y, Yang F, et al. High-level topology synthesis method for $\Delta$-$\Sigma$ modulators via bi-level Bayesian optimization. IEEE Trans Circ Syst II, 2023, 70: 4389–4393
254 Fayazi M, Taba M T, Afshari E, et al. AnGeL: fully-automated analog circuit generator using a neural network assisted semi-supervised learning approach. IEEE Trans Circuits Syst I, 2023, 70: 4516–4529
255 Hakhamaneshi K, Nassar M, Phielipp M, et al. Pretraining graph neural networks for few-shot analog circuit modeling and design. IEEE Trans Comput-Aided Des Integr Circ Syst, 2023, 42: 2163–2173
256 Budak A F, Gandara M, Shi W, et al. An efficient analog circuit sizing method based on machine learning assisted global optimization. IEEE Trans Comput-Aided Des Integr Circ Syst, 2022, 41: 1209–1221
257 Wang H, Wang K, Yang J, et al. GCN-RL circuit designer: transferable transistor sizing with graph neural networks and reinforcement learning. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2020
258 Zhao A, Wang X, Lin Z, et al. cVTS: a constrained Voronoi tree search method for high dimensional analog circuit synthesis. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
259 Burns S M, Chen H, Dhar T, et al. Machine Learning for Analog Layout. Cham: Springer International Publishing, 2022. 505–544
260 Kunal K, Poojary P, Dhar T, et al. A general approach for identifying hierarchical symmetry constraints for analog circuit layout. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020
261 Zhu K, Chen H, Liu M, et al. Automating analog constraint extraction: from heuristics to learning: (invited paper). In: Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2022. 108–113
262 Zhu K, Liu M, Lin Y, et al. GeniusRoute: a new analog routing paradigm using generative neural network guidance. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019
263 Xu B, Lin Y, Tang X, et al. WellGAN: generative-adversarial-network-guided well generation for analog/mixed-signal circuit layout. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2019. 1–6
264 Gusmão A, Horta N, Lourenço N, et al. Late breaking results: attention in Graph2Seq neural networks towards push-button analog IC placement. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2021. 1360–1361
265 Wang P C, Lin M P H, Liu C N J, et al. Layout synthesis of analog primitive cells with variational autoencoder. In: Proceedings of International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), 2023
266 Liu M, Zhu K, Gu J, et al. Towards decrypting the art of analog layout: placement quality prediction via transfer learning. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2020. 496–501
267 Lin Y, Li Y, Fang D, et al. Are analytical techniques worthwhile for analog IC placement? In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2022. 154–159
268 Xu P, Li J, Ho T Y, et al. Performance-driven analog layout automation: current status and future directions. In: Proceedings

of IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC), 2024
269 Ren H, Kokai G F, Turner W J, et al. ParaGraph: layout parasitics and device parameter prediction using graph neural networks. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2020
270 Zhang Q, Su S, Liu J, et al. CEPA: CNN-based early performance assertion scheme for analog and mixed-signal circuit simulation. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2020
271 Hakhamaneshi K, Werblun N, Abbeel P, et al. BagNet: Berkeley analog generator with layout optimizer boosted with deep neural networks. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019
272 Fang Y, Liu Z, Lu Y, et al. NPS: a framework for accurate program sampling using graph neural network. 2023. ArXiv:2304.08880
273 Li L, Flynn T, Hoisie A. Learning independent program and architecture representations for generalizable performance modeling. 2023. ArXiv:2310.16792
274 Yi X, Lu J, Xiong X, et al. Graph representation learning for microarchitecture design space exploration. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
275 Sakhuja C, Shi Z, Lin C. Leveraging domain information for the efficient automated design of deep learning accelerators. In: Proceedings of IEEE International Symposium on High-Performance Computer Architecture (HPCA), 2023. 287–301
276 Shi Z, Zheng Z, Khan S, et al. DeepGate3: towards scalable circuit representation learning. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2024
277 Li M, Shi Z, Lai Q, et al. On EDA-driven learning for SAT solving. In: Proceedings of the 60th ACM/IEEE Design Automation Conference (DAC), 2023. 1–6
278 Shi Z, Li M, Khan S, et al. DeepTPI: test point insertion with deep reinforcement learning. In: Proceedings of IEEE International Test Conference (ITC), 2022. 194–203
279 Wang Z, Bai C, He Z, et al. Functionality matters in netlist representation learning. In: Proceedings of ACM/IEEE Design Automation Conference, 2022. 61–66
280 Xie Z, Ren H, Khailany B, et al. PowerNet: transferable dynamic IR drop estimation via maximum convolutional neural network. In: Proceedings of Asia and South Pacific Design Automation Conference (ASP-DAC), 2020
281 Feng Z, Guo D, Tang D, et al. CodeBERT: a pre-trained model for programming and natural languages. 2020. ArXiv:2002.08155
282 Orenes-Vera M, Martonosi M, Wentzlaff D. From RTL to SVA: LLM-assisted generation of formal verification testbenches. 2023. ArXiv:2309.09437
283 Sorensson N, Een N. MiniSAT v1.13 - a SAT solver with conflict-clause minimization. SAT, 2005, 2005: 1–2
284 Fleury A, Heisinger M. CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling entering the SAT competition 2020. SAT Competition, 2020, 2020: 50
285 Eén N, Mishchenko A, Sörensson N. Applying logic synthesis for speeding up SAT. In: Proceedings of Theory and Applications of Satisfiability Testing, 2007. 272–286
286 Akers S B. A truth table method for the synthesis of combinational logic. IRE Trans Electron Comput, 1961, EC-10: 604–615
287 Zimmermann R, Tran D Q. Optimized synthesis of sum-of-products. In: Proceedings of the 37th Asilomar Conference on Signals, Systems & Computers, 2003. 867–872
288 Malik S, Wang A R, Brayton R K, et al. Logic verification using binary decision diagrams in a logic synthesis environment. In: Proceedings of IEEE International Conference on Computer-Aided Design, 1988. 6–7
289 Mishchenko A, Chatterjee S, Brayton R. Dag-aware aig rewriting a fresh look at combinational logic synthesis. In: Proceedings of the 43rd annual Design Automation Conference, 2006. 532–535
290 Zhang H T, Jiang J H R, Amarú L, et al. Deep integration of circuit simulator and SAT solver. In: Proceedings of the 58th ACM/IEEE Design Automation Conference (DAC), 2021. 877–882
291 Pan H, Lan C, Liu Y, et al. Physically aware synthesis revisited: guiding technology mapping with primitive logic gate placement. In: Proceedings of IEEE/ACM International Conference On Computer Aided Design (ICCAD), 2024. 1–9
292 Zou S, Zhang J, Shi B, et al. BESWAC: boosting exact synthesis via wiser SAT solver call. In: Proceedings of IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE), 2024
293 Liu L, Fu B, Wong M D F, et al. Xplace: an extremely fast and extensible global placement framework. In: Proceedings of the 59th ACM/IEEE Design Automation Conference, 2022
294 Wang B, Shen G, Li D, et al. LHNN: lattice hypergraph neural network for VLSI congestion prediction. In: Proceedings of ACM/IEEE Design Automation Conference (DAC), San Francisco, 2022
295 Pu Y, Shi C, Samson G, et al. A 9-mm$^2$ ultra-low-power highly integrated 28-nm CMOS SoC for Internet of Things. IEEE J Solid-State Circ, 2018, 53: 936–948
296 Jain S, Khare S, Yada S, et al. A 280mV-to-1.2V wide-operating-range IA-32 processor in 32nm CMOS. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2012. 66–68
297 Klemme F, Amrouch H. Efficient learning strategies for machine learning-based characterization of aging-aware cell libraries. IEEE Trans Circ Syst I, 2022, 69: 5233–5246
298 Chowdhury A B, Tan B, Karri R, et al. OpenABC-D: a large-scale dataset for machine learning guided integrated circuit synthesis. 2021. ArXiv:2110.11292
299 Chai Z, Zhao Y, Liu W, et al. CircuitNet: an open-source dataset for machine learning in VLSI CAD applications with improved domain-specific evaluation metric and learning strategies. IEEE Trans Comput-Aided Des Integr Circ Syst, 2023, 42: 5034–5047
300 Willsey M, Nandi C, Wang Y R, et al. egg: fast and extensible equality saturation. Proc ACM Program Lang, 2021, 5: 1–29