Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 7** : 1–14.
ISSN-0037-8712
doi:10.5269/bspm.81896

Applications of AI in VLSI Design: A Comprehensive Review

Deepak Garg[1], Manoj Gupta[2], Sumit Kumar Gupta[3]

ABSTRACT: This paper presents a comprehensive review of the rapidly expanding role of AI in transforming the VLSI design ecosystem. As technology nodes advance toward 3 nm and beyond, traditional EDA tools struggle with escalating complexity, rising verification costs, and increasingly interdependent power, performance, and area (PPA) constraints. The review examines how AI methodologies-particularly Deep Reinforcement Learning (DRL), supervised learning, and Graph Neural Networks (GNNs)-address these challenges across the complete VLSI design flow. AI-driven approaches are shown to significantly enhance RTL optimization, autonomous Design Space Exploration (DSE), physical design stages such as floor planning, placement, routing, and timing closure, as well as verification and testing processes. The paper further highlights AI's expanding impact in analog and mixed-signal design through variation-aware modeling and intelligent simulation reduction. Industrial adoption is validated through case studies demonstrating substantial productivity gains and measurable PPA improvements delivered by commercial AI-enabled EDA platforms. Emerging research directions-including Circuit Foundation Models (CFMs) and Explainable AI (XAI)-are identified as critical enablers for scalable, interpretable, and fully autonomous design flows. Overall, the review underscores AI's pivotal role in shaping the next generation of high-performance, energy-efficient semiconductor design methodologies.

Key Words: Artificial intelligence, electronic design automation, deep reinforcement learning, graph neural networks, design space optimization, circuit foundation models.

Contents

2020 *Mathematics Subject Classification:* 68T07, 68W01, 65K10.
Submitted February 11, 2026. Published April 11, 2026

1. Introduction

1.1. The Scaling Crisis and the Mandate for AI in EDA

The semiconductor industry is currently at a major turning point because technology nodes keep getting smaller. As feature sizes shrink to 3nm and lower, the difficulties of VLSI design are growing at an alarming rate, surpassing the limits of traditional EDA methods [1]. This complexity shows up in a few important ways: verification costs are rising so much that they now take up to 70 % of the total chip development cycle; there are more and more complex design rules to keep track of; and there are complicated trade-offs to make between power, performance, and area [2].

To keep innovation going and meet strict deadlines for getting products to market, smarter tools and automated methods are no longer just nice to have; they are now required [3]. The main goal of adding AI and ML to VLSI design is to completely change the way designs are made. AI systems help engineers get products to market faster, make silicon more predictable, find important design bugs much earlier in the process, and make design improvements that are often better than what humans can do [1].

The need for AI comes from the fact that traditional DSE is not very efficient. Traditionally, optimization depends on people manually going through huge parameter spaces over and over again, mostly based on their own experience and the knowledge that has built up in the institution. This process takes a lot of work and can take weeks. It often results in less-than-ideal solutions because the designer is tired, the analysis is incomplete or the design parameters are too strict. The potential solution space in modern chip design is trillions of times bigger than that of complex games like Go-renders. The use of AI is a key way to automate this exploration, moving the industry from traditional, manual DSE to generative, autonomous Design Space Optimization (DSO).

Nomenclature

AI	Artificial Intelligence
LLMs	Large Language Models
AMS	Analog and Mixed-Signal
MDP	Markov Decision Process
ATPG	Automated Test Pattern Generation
ML	Machine Learning
CFMs	Circuit Foundation Models
MPNNs	Message-Passing Neural Networks
CTS	Clock Tree Synthesis
P&R	Place and Route
DRC	Design Rule Check
PPA	Power, Performance and Area
DRL	Deep Reinforcement Learning
QoR	Quality of Results
DSE	Design Space Exploration
RL	Reinforcement Learning
DSO	Design Space Optimization
RTL	Register-Transfer Level
EDA	Electronic Design Automation
SI	Signal Integrity
GCNs	Graph Convolutional Networks
SP	Sequence Pair
GNNs	Graph Neural Networks
VLSI	Very Large-Scale Integration
HLS	High-Level Synthesis
XAI	Explainable AI

1.2. Scope, Contribution, and Organization of the Review

This review provides a detailed and critical discussion on the use of AI applications in the whole VLSI design lifecycle. It will start with the analysis by defining taxonomy of core AI techniques used in EDA, in terms of the representation and processing of circuit data. Further on, the introduction of AI methods in the key design phases Register-Transfer Level (RTL) optimization and synthesis, physical design (floor planning, placement, and routing), verification, testing, and new applications in analog and mixed-signal design are described [4]. The present paper is a critical evaluation of the measurable affects of AI on the major industrial indicators, specifically, PPA. Moreover, it pinpoints two key, introspective frontiers on future research, the introduction of Circuit Foundation Models to reach generalization across designs, and the Explain ability Imperative, which is the crucially important demand of transparency and trustworthiness in automated design choices [5].

2. Foundational AI Methodologies and Data Representation in EDA

2.1. Taxonomy of Machine Learning Paradigms for Circuit Optimization

The successful integration of AI into EDA relies on the tailored application of specific machine learning paradigms to complex optimization problems.

Reinforcement Learning (RL) and Deep Reinforcement Learning (DRL): The leading approach to autonomous design optimization is placed on RL. The RA agents engage the design environment (the Markov Decision Process), take a sequence of decisions to maximize a specified reward, where it is typically the optimal PPA targets or reduced wire length and congestion [6]. DRL is used to control highly dynamic processes, including the creation of complex test cases during verification and the refinement of floor plans and place strategies through the learning of the optimal strategies to use in a continuous feedback process [1].

Supervised Learning: Predictive models are developed based on this paradigm. With trained models

on large historical design data, predictions of important results, including congestion hotspots, probability of timing violation, or expected Quality of Results (QoR) measures, are made based on design characteristics at early design phases [4]. It is this prediction ability that forms the core of the shift-left approach in EDA, as design problems can be signaled, and fixed before the downstream implementation activities prove extremely expensive.

2.2. Graph Representation and Graph Neural Networks (GNNs)

A critical architectural consideration in applying AI to VLSI is the representation of the circuit itself. VLSI net lists are inherently massive, non-Euclidean data structures, characterized by complex connections between functional units (cells) via wires (nets). These net lists are naturally and most effectively modeled as graphs or hyper graphs, which necessitates specialized learning architectures.

GNNs for Circuit Analysis: The vital components of modeling these complex relationships are Graph Neural Networks (GNNs) which also covers Graph Convolutional Networks (GCNs), and Message-Passing Neural Networks (MPNNs). GNNs can be implemented at different levels of abstraction, including optimization target analysis in high-level synthesis (in the form of Data Flow Graphs, or DFGs) to logic synthesis (in the form of Boolean Networks) [7]. Such networks are learned with high-dimensional embeddings and features of the graph topology, and are thus able to classify sub-circuit functionality, analyze the effects of logic rewrite, and comprehend the complex long-range interactions which are important to predict post-routing properties.

The fact that GNNs are efficient to learn the graph topology has a significant impact on the design flow speed. Hard optimization steps, like place and route, are infamously expensive in the iterative design cycle and the solutions of the wire length and congestion-are intricate functions of the net listing topology. GNNs offer a convenient and effective solution to these post-routing properties to predict them efficiently and accurately without needing to run the entire, time-intensive P&R utility based on the synthesized net list. This capability change significantly speeds up Design Space Exploration fidelity as it enables quick Quality of Results analysis and thus satisfies the industry-wide objective of identifying and eliminating performance risks far earlier in the development pipeline [4].

Figure 1 illustrates a progressive workflow of AI/ML techniques applied to VLSI and circuit design, moving from problem formulation to advanced structural analysis. It is organized left to right as a pipeline with arrows indicating information flow and increasing design intelligence. The descriptions of all four stages are discussed as:

First Stage: This stage represents the application of generic machine learning techniques for design parameter optimization. Raw design data (shown as scattered dots) are processed to identify patterns and correlations that help tune circuit parameters such as delay, power, or area.

Second Stage: In this stage, the model interacts with the design environment and learns through trial-and-error. Reinforcement learning is used for autonomous design optimization, such as transistor sizing, placement decisions, or routing strategies, guided by reward signals.

Third Stage: In this stage, labeled design data are used to build predictive models. Supervised learning enables fast estimation of circuit metrics (e.g., power, timing, or yield) without running expensive simulations, supporting rapid design-space exploration.

Fourth Stage: The final stage focuses on analyzing circuit topology. Circuits are modeled as graphs where nodes represent devices or gates and edges represent interconnections. Graph Neural Networks (GNNs) capture structural dependencies, making them suitable for tasks such as timing prediction, fault detection, and layout-aware analysis.

3. AI in Digital Synthesis and Front-End Optimization

3.1. High-Level Synthesis (HLS) and RTL Optimization

The front-end of the design flow, encompassing RTL design and logic synthesis, is benefiting significantly from AI-assisted automation. AI is enhancing RTL design by automating traditionally manual operations such as logic synthesis, data path optimization, and efficient resource allocation [4]. This automation helps overcome the limitations of conventional heuristic compilers, which struggle to achieve global optimization across large, complex designs.

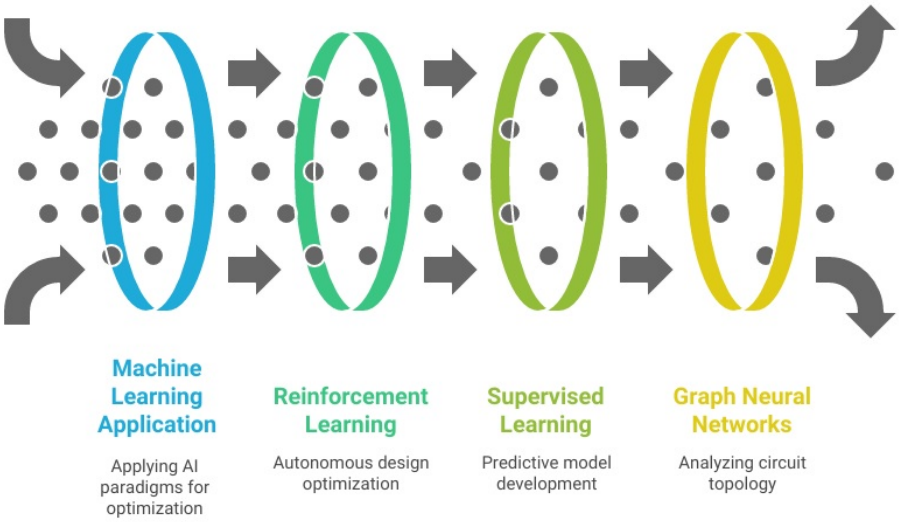

Figure 1: AI-Driven Circuit Optimization Process

Furthermore, the design process is streamlined by the introduction of tools that automatically generate design constraints based on high-level specifications [8]. This automation ensures that the physical implementation adheres to the desired performance targets early in the flow, minimizing constraint closure headaches later on.

3.2. Predictive Modeling for Quality of Results (QoR) Forecasting

At the synthesis stage, machine learning models are indispensable for guiding the optimization process. Predictive models utilize historical design data and layout features to forecast PPA targets [4]. This guidance allows design engineers to set optimal constraints and choose effective mapping strategies with high confidence, leading to improved convergence on PPA goals.

The culmination of AI in front-end optimization is the concept of Autonomous DSE, now realized in commercial tools like Synopsys DSO.ai. It is an independent optimization paradigm that applies the Reinforcement Learning technology to explore the expansive design space to seek the best solution. The AI engine can search through an enormous number of possible design solutions, by automatically optimization of the entire flow of RTL-to-GDSII, in the process continuously discovering recipes that open the potential of superior PPA both on the logical and physical sides, with less manual effort.

The figure 2 illustrates a funnel-shaped AI-assisted VLSI/SoC design flow, showing how a design evolves from abstract requirements to an optimized implementation through successive intelligence-driven stages. The descriptions of figure 2 are discussed as:

• High-Level Specifications (Top of the funnel): The process begins with broad and abstract system requirements such as functionality, performance targets, power constraints, and area goals. At this stage, design freedom is high and details are minimal.

• RTL Design Enhancement: In the first refinement stage, AI techniques assist in improving the Register Transfer Level description. This includes optimizing logic structures, improving coding efficiency, and identifying potential performance or power bottlenecks early in the design cycle.

• Constraint Generation: As the funnel narrows, AI models automatically derive and refine design constraints related to timing, power, and area. This step helps translate high-level specifications into actionable constraints for downstream synthesis and implementation tools.

• Predictive Modeling: This stage uses machine learning models to predict key design metrics (such as delay, power consumption, or congestion) without requiring full simulations. Predictive modeling accelerates design-space exploration and supports informed decision-making.

• Autonomous Optimization (Bottom of the funnel): At the narrowest stage, AI-driven autonomous optimization techniques often based on reinforcement learning fine-tune the design parameters. The goal

is to achieve optimal trade-offs among power, performance, and area with minimal human intervention.

• Optimized Design (Output): The final outcome is a highly optimized design that meets specifications efficiently, with reduced design iterations and improved productivity.

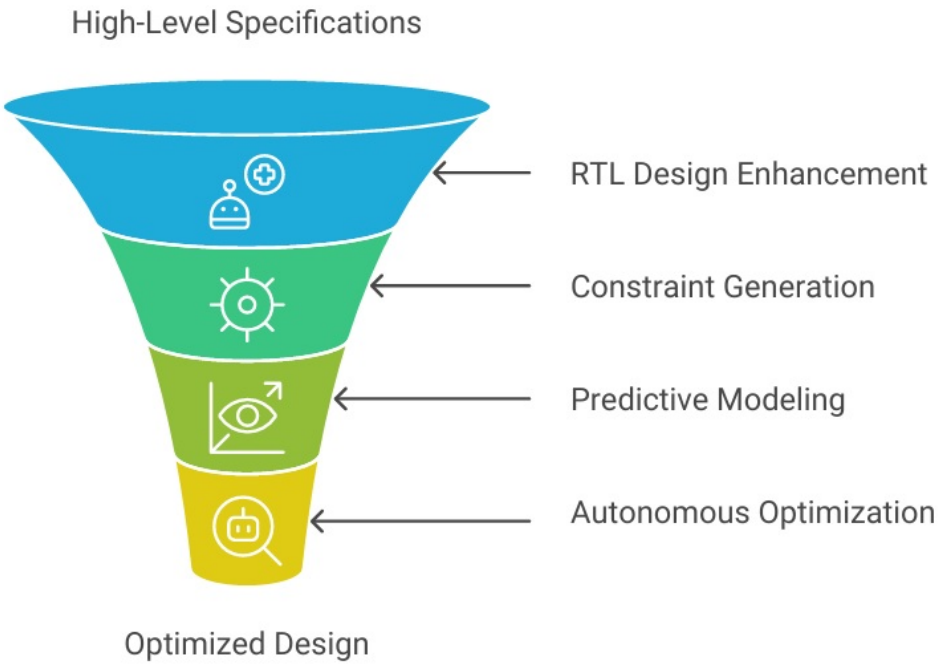

Figure 2: AI-Driven Circuit Optimization Process

4. AI-Driven Physical Design Automation (Back-End)

Physical design-including partitioning, floor planning, placement, clock network synthesis (CTS), and routing-remains a crucial stage, as it centrally determines the overall quality metrics, including PPA, manufacturability, and reliability. This stage involves resolving highly interdependent problems such as timing closure, congestion, and signal integrity under aggressive turnaround constraints [10].

4.1. Intelligent Floor planning and Placement

The first stage of the physical design is the floor planning problem which entails setting up the relative location of the major functional blocks so as to maximize the area and interconnection properties [6]. The formulation of this task is that of a Markov Decision Process (MDP) and thus best fit in the training of Reinforcement Learning [11].

The implementation of RL, including the offered Deep Reinforcement Learning (DRL) algorithms, makes use of such representations as Sequence Pair (SP) encoding to search the solution space on their own [6]. Through the power of RL to learn and generalize, such algorithms have shown themselves to be able to find better solutions to floor plans than traditional approaches to the problem which rely on heuristic approaches, especially when dealing with fixed obstructions to optimize the total area and wire length [11].

Equally, the process of placement optimization has changed the focus of being a purely heuristic process to a data-driven DRL task. The agents of DRL are taught to find optimal placement strategies by acting upon the design environment and obtaining feedback on the key performance indicators such as timing, congestion, and wire length on an iterative basis [4].

4.2. Routing, Clock Tree Synthesis, and Timing Closure

AI techniques are essential for optimizing the interconnected stages following placement. In Clock Tree Synthesis (CTS), AI methodologies use predictive models and topology search algorithms to evaluate multiple tree structures rapidly, optimizing the clock distribution network to minimize skew and power consumption, particularly in large-scale designs where manual tuning is impractical [4]. In the case of Routing, AI reduces the number of cycles by estimating the level of congestion and informing the direction of paths. Predicting the congestion hotspots is carried out using Neural Networks and the routing grid using Graph Search Algorithms such that timing and Design Rule Check (DRC) are met.

Finally, AI dramatically accelerates Timing Closure. Predictive models predict timing violations at the initial stage of the flow, depending on past information and layout characteristics. More modern systems use automated Engineering Change Order (ECO) Suggestion Engines, or automated suggestions and corrective actions based on Reinforcement Learning, to propose and apply the best fixes, whether this be gate sizing or buffer additions, by responding to larger and more complex design layouts, which cannot be effectively addressed by human effort [4].

Predictive models forecast timing violations early in the flow based on historical data and layout features. More advanced systems utilize automated Engineering Change Order (ECO) Suggestion Engines, which leverage Reinforcement Learning to suggest and implement optimal fixes, such as gate sizing or buffer insertions, through iterative feedback, overcoming the limitations of manual intervention in large, complex designs [4].

The physical design dependency on AI is due to the fact that according to the conventional EDA, the objectives (PPA, manufacturability and timing) are regarded as independent constraints. Physical design goals, however, in practice are very interdependent; high performance goals usually lead to power or area optimization, at the cost of complex trade-offs [10]. The complexity of these multi-objective trade-offs operating in parallel in different interdependent constraints can only be dealt with in a special way using DRL methodologies. As a good mediator between competing objectives, AI can save a lot of expensive manual steps needed to reach design closure, making the design schedule shorter without compromising design quality [4].

5. AI in Verification, Testing, and Analog Design

5.1. Intelligent Verification and Coverage Closure

Verification remains the single most resource-intensive phase, often accounting for nearly two-thirds of the total chip development time [1]. AI integration targets this bottleneck to achieve significant productivity gains.

Reinforcement Learning (RL) plays a crucial role in dynamic test generation. RL algorithms are employed to guide the test generation process, dynamically learning which input stimuli are most effective at exploring complex state spaces and revealing latent design bugs, thereby ensuring efficient verification coverage closure. Commercial tools like Synopsys VSO.ai utilize AI engines to achieve higher coverage quality faster, eliminating redundancies in regression tests, and automating root cause analysis [9]. Furthermore, ML models analyze verification coverage data and past simulation results to predict the likelihood of bugs in specific design blocks, allowing engineers to "shift left" their focus and target verification resources efficiently where they are most needed [1].

5.2. Testing, Fault Tolerance, and Manufacturing

In the post-design phases, AI provides substantial benefits in testing and manufacturing. Automated Test Pattern Generation (ATPG), which is critical for identifying faults, is optimized by AI tools (e.g., Synopsys TSO.ai) to achieve maximum defect coverage using the fewest possible test patterns, thereby reducing testing time and associated costs [4].

In manufacturing, AI models are indispensable for ensuring high yield and fault tolerance. They enable robust statistical modeling necessary to handle process variations at nanoscale geometries, which otherwise lead to unpredictable behavior [3]. AI facilitates real-time process optimization and yield prediction,

resulting in increased overall efficiency and reduced defect rates [1].

5.3. Analog and Mixed-Signal (AMS) Applications

The amplification and sensitivity of analog data, the large degree of variability, and the lengthy and compute intensive simulating of analog data makes AMS design uniquely challenging to automate [12]. Such complications normally cause design mistakes and expensive re-spins of chips.

Special tools help overcome these problems with the help of AI. As an example, Siemens Solido incorporates machine learning to support variation-aware design and analog/RF verification, by a significant factor lowering the simulation needs to signoff. Commercial tools such as Synopsys ASO.ai are specifically aimed at providing accelerated analog implementation and verification, allowing it to be used features like automated schematic migration to facilitate the task of updating the devices and parameters to new process design kits [9].

6. Commercial Adoption and Quantified Impact

6.1. The Shift to Integrated, Autonomous EDA Solutions

The adoption of AI in VLSI design is no longer purely academic; it is rapidly transitioning into integrated, autonomous commercial EDA solutions. This represents a foundational shift from fragmented, task-specific ML algorithms to comprehensive AI-powered EDA suites. Tools like Synopsys's AI portfolio (DSO.ai, VSO.ai, TSO.ai and ASO.ai) and Cadence Cerebrus manage the flow seamlessly from system architecture through manufacturing [9]. These AI engines act as centralized intelligence layers, designed to ingest and analyze terabytes of high-velocity, multi-dimensional data-including RTL models, net lists, timing libraries, physical characteristics, and Design for Manufacturing (DFM) models to make complex, autonomous optimization decisions.

6.2. Quantifiable PPA Metrics and Case Studies

Industrial case studies provide concrete validation of the superior performance achievable through AI-driven design.

Cadence has reported that its Cerebrus tool, which uses a unique ML technology to drive the RTL-to-signoff flow, delivers improvements of up to 10x productivity and 20 % PPA improvements over traditional manual flows [1]. Similarly, Synopsys DSO.ai has demonstrated significant quantitative gains, including a reported > 10 % power reduction on commercial designs achieved by autonomously searching for superior synthesis and layout strategies [13].

A specific example illustrating the power of autonomous optimization involves an AI engine achieving a challenging 1.95 GHz speed target (a 12 % speed-up) while simultaneously reducing power consumption (27.9 mW) and meeting area constraints. Critically, this result was achieved autonomously in just two days with zero human intervention [13].

This body of commercial evidence confirms the transformative capability of RL-based optimization engines. The initial hypothesis—that autonomous DRL optimization could systematically exceed human capability-is validated by the achieved quantifiable PPA gains (e.g., 20 % PPA, 12 % speed-up) that were unattainable within aggressive timelines using months of manual effort [13]. This conclusively demonstrates that RL engines effectively manage the colossal search space (trillions of solutions) of complex chip design, firmly establishing AI as the dominant methodology for future energy-efficient and high-performance computing designs. The different Commercial AI Platforms for VLSI Optimization and Reported Performance Metrics are summarized in table 1.

7. Emerging Research Frontiers: Circuit Foundation Models and Generative AI

7.1. The Circuit Foundation Model (CFM) Paradigm

As AI matures in EDA, a new technology trend Circuit Foundation Models (CFMs)-is emerging. CFMs represent a paradigm shift away from traditional task-specific AI solutions, which require extensive labeled data for every new design and process node [5].

Table 1: Commercial AI Platforms for VLSI Optimization and Reported Performance Metrics

S. No.	Platform	Vendor	Optimization Scope	Ref	Primary AI Method	Reported PPA
1	DSO.ai	Synopsys	Autonomous RTL-to-GDSII Digital Flow	[14]	Deep Reinforcement Learning	>10% power reduction; 12% speed-up; Autonomous Search
2	Cerebrus	Cadence	Autonomous RTL-to-Signoff Flow	[15]	Machine Learning	Up to 10× productivity; 20% PPA improvements
3	VSO.ai	Synopsys	Verification	[16]	AI Engine	Higher coverage quality; faster closure; Redundancy elimination
4	Solido	Siemens	Analog/RF Design	[15]	Machine Learning	Variation-aware design; Improved yield prediction
5	TSO.ai	Synopsys	Testing	[16]	AI Engine	Optimized test program generation; Maximum defect coverage

CFMs are developed through a two-stage process: first, self-supervised pre-training is conducted on a massive, unlabeled dataset of circuits to learn intrinsic, generalized circuit properties; second, efficient fine-tuning adapts the model for specific, downstream applications [5]. This architectural approach directly addresses the critical challenge of generalization, a known limitation where models trained on one architecture often fail to perform adequately on others [4]. The fundamental advantages of the CFM approach are profound: superior model generalization across different architectures, significantly reduced reliance on expensive labeled circuit data, efficient adaptation to new tasks and technology nodes, and unprecedented generative capability [5]. This generalization is the key architectural solution necessary for AI to move beyond single-design success and become a universally deployable tool.

7.2. Taxonomy and Architecture of CFMs

CFMs are broadly categorized into two primary types based on their underlying architecture and function [5]:

Encoder-Based Methods: These models focus primarily on general circuit representation learning for predictive tasks. They are essential for applications such as early-stage design quality evaluation, allowing engineers to forecast potential PPA issues based on learned intrinsic circuit properties before substantial resources are committed to implementation [5].

Decoder-Based Methods and Large Language Models (LLMs): These leverage the architecture of LLMs for generative applications. While LLMs are traditionally used for text, their architecture can be adapted for generative tasks in EDA, such as creating circuit-related context and automating the review of VLSI design specifications at various abstraction levels. Studies in this area demonstrate a novel approach to optimizing the accuracy and efficiency of the initial stages of IC design, although these methods also necessitate a focus on explainability to ensure trust [17].

7.3. Practical Implementations of CFM Paradigm

Pre-trained Encoder Models for Predictive Design Tasks Circuit representation learning encoders form the backbone of CFMs by learning generalized latent embeddings of circuits from large unlabeled datasets. These learned embeddings can then predict design quality metrics (e.g., timing, power, logic correctness) early in the VLSI flow. Encoders reduce human effort in early evaluation and improve scalability for large designs by predicting potential design issues from raw circuit representations. Some examples for this design task are given as:

• Net-TAG is a multimodal foundation model that fuses textual gate semantics with graph structure to support multiple downstream tasks such as functional and physical property prediction from net-lists.

This helps automate prediction of circuit behavior at various stages of design [18].

• General encoder-based CFMs capture functional and structural aspects of circuits, enabling predictive tasks like early quality evaluation before expensive simulation/physical design [5].

Generative Decoder Models for RTL & Specification Generation Decoder-type CFMs often based on LLMs are used to generate HDL code, design specifications, assertions, or test benches automatically from high- or low-level prompts. Decoder models enable automation of creative and standardized tasks (like code generation), freeing engineers from repetitive HDL coding. Some examples for this design task are given as:

• Models aligned with LLMs can generate RTL code or functional specifications from natural language descriptions or design intents. This expands EDA workflows by transforming human design intent into synthesizable hardware descriptions.

• Specific research shows LLM-based CFMs generating high-level functionality directly from low-level net lists, bypassing manual writing of specification and structural RTL. This capacity dramatically shortens design entry and iteration cycles.

Hybrid CFM Frameworks Aligning Encoders & Decoders More advanced implementations combine both predictive and generative capabilities in a shared model space. This allows models to reason across representations from circuit graphs to text and back improving accuracy and utility. These hybrid CFMs can significantly reduce manual verification effort and improve early design feedback, key bottlenecks in modern chip development.
Gen-EDA introduces a cross-modal encoder-decoder alignment, enabling generative reasoning tasks on net-lists such as reconstructing high-level specifications or RTL from low-level gate connections. This is the first demonstration of a foundation model that integrates structural and textual reasoning within VLSI EDA.

Integration with Conventional EDA Tools CFMs are not isolated research artifacts they are being conceived for integration into traditional EDA flows:

• Self-supervised pre-trained models can feed quality predictions into mainstream synthesis/physical tools to influence placement, routing, and routing decisions earlier.

• Foundation models capable of generating verification/test bench assets complement tools like Synopsys, Cadence, and Siemens EDA flows by adding intelligent, learned automation layers. The different Practical Implementation Examples for VLSI Optimization and Reported Performance Metrics are summarized in table 2.

Table 2: Summary of Practical Implementation Examples

S. No.	CFM Implementation Class	Practical Output	Main Benefits
1	Encoder-based CFMs	Circuit property prediction	Early quality evaluation; error avoidance
2	Decoder-based CFMs / LLMs	RTL, specification, assertions	Code automation; synthesis assistance
3	Hybrid Encoder–Decoder Models	Functional reasoning from netlists	Bridging representation gaps; deeper insight
4	Verification Generation	Test scripts; assertions	Reduced manual verification toil
5	EDA Integration	Guidance for synthesis and physical tools	Smarter automation & faster cycles

8. Critical Challenges and the Explain ability Imperative

8.1. Integration, Scalability, and Physical Awareness

Despite the demonstrated success, the widespread industrial adoption of AI in VLSI faces significant technical and philosophical hurdles.

Integration and Interoperability: AI tools must seamlessly integrate and interoperate with the established EDA ecosystems provided by major vendors like Cadence and Synopsys [3]. Disrupting existing, heavily validated workflows is a high barrier to entry, demanding that new AI tools function without disrupting the current design methodology [8].

Scalability and Complexity: The continued scaling of designs to accommodate billions of components requires continuous innovation in model architectures to handle massive data scales efficiently [3].

Generalization and Physical Constraints: AI models require robust, diverse, and high-quality labeled datasets to generalize effectively across different design styles and architectures [4]. Critically, models must incorporate awareness of physical constraints beyond standard PPA metrics, such as complex thermal profiles, Signal Integrity (SI), and manufacturability rules, which are essential for guaranteeing yield and reliability at nanoscale geometries [3].

8.2. The Explain ability Mandate (XAI): Ensuring Trust and Audit ability

The most notable non-technical hurdle to complete adoption of AI is, perhaps, the black-box quality of the Deep Learning and Reinforcement Learning models. The transparency of AI choices causes erosion of industrial confidence. Designers should have the background of the motivation of an AI-driven decision, such as why a certain location was selected or why a certain time fix was proposed to debug possible erroneous scenarios, address infrequent edge cases, and finally support the output [4].
The mandate for XAI is twofold: it is necessary for engineering validation and for governance. From a governance perspective, explainability is essential for accountability, auditing, and adhering to regulatory requirements by providing clear, understandable justification for automated decisions [19]. Without transparent decision-making, there is a risk of overreliance on opaque automation, potentially obscuring designer intuition and masking edge cases.

8.3. Techniques for Interpretability in EDA

Explainable AI (XAI) refers to the techniques used to provide the rationale behind the output of a machine learning algorithm, fostering confidence and enabling debugging [19].
Feature Importance: These techniques highlight the most influential input features that contributed to a specific AI output or prediction, such as identifying if wire length minimization or timing criticality drove a placement decision [20].
Local Interpretable Model-Agnostic Explanations (LIME): LIME is a powerful method used to explain the predictions of any complex machine learning model. LIME generates local, instance-level explanations by perturbing the input data and observing the corresponding effect on the model's output [19]. This capability is crucial for providing granular debug information for specific placement violations or timing optimizations.
Shapley Additive Explanations (SHAP): SHAP values quantify the contribution of individual features to a prediction, rooted in cooperative game theory. Both LIME and SHAP are vital tools for moving black-box models toward greater interpretability, helping cyber security professionals and, by extension, EDA engineers, analyze model results and ensure continuous improvement [21]. The incorporation of XAI methods provides the rationale (the local explanation) for the black-box prediction, effectively acting as a necessary trust bridge between the high performance of AI and the human requirement for audit ability [19]. This integration transforms the AI from an un-auditable oracle into a transparent engineering tool, empowering designers to refine model behaviors, mitigate rare corner cases, and overcome the debilitating risk of overreliance on opaque automation [4]. The different Critical Challenges in AI Integration for VLSI/EDA and Mitigation Strategies are summarized in table 3.

9. Limitations and Failure Cases of AI Techniques in VLSI Design

Despite the growing adoption of AI across various stages of VLSI design automation, several limitations and potential failure scenarios remain that constrain its widespread industrial deployment. Recog-

Table 3: Critical Challenges in AI Integration for VLSI/EDA and Mitigation Strategies

S. No.	Ref.	Challenge Domain	Impact/Consequence	Required Research / Mitigation Strategy
1	[23]	Explain ability (XAI)	Opaque decisions hinder designer trust, debugging, and audit ability	Implement XAI techniques (LIME, SHAP); Develop Hybrid Rule-Based + AI Systems
2	[5]	Generalization/ Data Dependency	Models fail across diverse architectures; reliance on high-quality labeled data	Develop Circuit Foundation Models (CFMs) via self-supervised pre-training
3	[23]	Scalability	Inefficient handling of large-scale designs (millions of cells)	Efficient graph learning architectures (GNNs); Optimized model partitioning
4	[22]	Physical Awareness	AI may ignore thermal /manufacturability constraints beyond PPA	Integration of robust statistical models and physical rule checking into reward functions

nizing these challenges is essential for realistic performance assessment and responsible design integration.

1. Limited Generalization across Technology Nodes: Modern AI models often exhibit poor cross-node generalization due to differences in device physics, physical effects, and design constraints between technology generations. This necessitates frequent retraining per node, reducing the reusability of learned models across industrial design flows. Such node-specific dependencies have been highlighted in recent reviews exploring machine learning for design space exploration and optimization [24].

2. Dependence on High-Quality Labeled Data: Supervised learning approaches for placement, routing, power prediction, and timing closure rely on large volumes of high-fidelity design datasets. However, chip design data is proprietary, confidential, and often unavailable outside EDA vendors or foundries, creating a data scarcity bottleneck that limits model robustness and real-world deployment.

3. Black-Box Behavior and Lack of Explainability: Deep learning and reinforcement learning models provide little interpretability of their decision processes, especially in critical subsystems like physical design or verification. The lack of transparent reasoning can undermine designer trust and make debugging or correcting AI-generated suggestions difficult, an issue stressed in surveys of AI-assisted EDA methodologies [5].

4. Failure in Corner and Rare Event Scenarios: AI models tend to underperform on rare but critical corner cases, such as extreme PVT (process, voltage, temperature) variations, noise sources, or adjacent interference patterns in layout. These corner conditions are underrepresented in training sets, leading the models to produce overly optimistic or inaccurate predictions, a common complaint noted in machine learning reviews on design space exploration [24].

5. Over fitting to Benchmark Conditions: Many AI evaluations in VLSI are conducted on synthetic benchmarks or academic test cases that lack the complexity of real SoC designs. Models tuned on narrow cases risk over fitting and may fail when confronted with large, heterogeneous designs typical of modern chips. This issue is widely discussed in general AI literature and applies to specialized EDA models [5].

6. High Computational and Energy Overheads: Training and running advanced models (e.g., graph neural networks or reinforcement learning for placement) can demand extensive compute and memory resources that are prohibitive for smaller research teams or startups. Although such models accelerate some design phases, the total cost of ownership in compute still remains high.

7. Integration Challenges with Legacy EDA Flows: Traditional EDA tools follow decades of rule-based workflows. Introducing AI components often results in format incompatibilities, tool certification obstacles, and workflow disruptions, especially when AI outputs deviate from deterministic rule engines. Hybrid AI–EDA workflows require careful interfacing and validation loops, as emphasized in recent surveys on circuit foundation models.

8. Security and Intellectual Property Risks: AI models trained on sensitive design data may inadvertently learn proprietary features, leading to IP leakage or vulnerabilities through model extraction or reverse engineering. Security concerns around data provenance and model trustworthiness are emerging as critical considerations in AI-augmented design automation research. The graphical representation of Challenges of AI Techniques in VLSI Design is shown in figure 3.

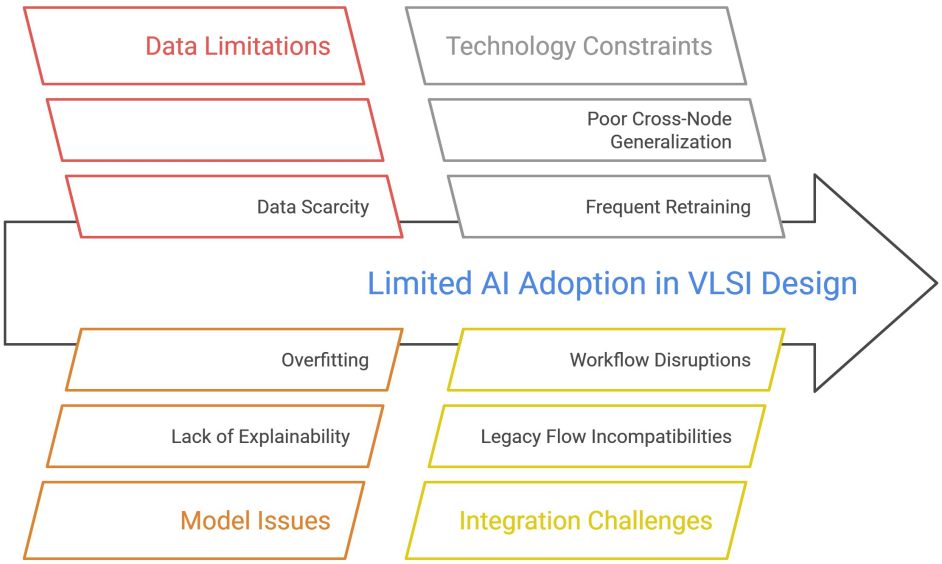

Figure 3: AI in VLSI Design: Challenges

10. Conclusion and Future Outlook

Synthesis of AI's Transformative Impact

The introduction of Artificial Intelligence has revolutionized VLSI design process to the point where it is no longer possible to rely on traditional methods like exploration that are based on heuristics: autonomous, quantified optimization is now deployed. AIs, especially Deep Reinforcement Learning and Graph Neural Networks, are shown to be competent in terms of addressing the astronomical complexity of advanced technology nodes, achieving provable and significant gains in terms of PPA and productivity, including the 20 percent reported improvements in PPA and 10-fold improvements in productivity in industrial applications [14]. This has been done by successfully automating the most time consuming bottlenecks or physically designing the physical design and the extremely complex verification step that consumes between 70-95 percent of the development [1].

Predicted Trajectory and Research Roadmap

AI in VLSI is evidently heading the way to entirely autonomous design flows, where AI systems handle and co-optimize all the conflicting goals throughout the whole life-cycle-from RTL specification, down to manufacturing yield prediction [25]. To allow this complete automation, future research needs to address two frontiers that are related to each other. To begin with, the success of AI depends on the issue of generalization. The needed architecture solution is the fast development of CFMs, which is based on self-supervised pre-training and which is expected to release powerful performance across varied chip architectures and technology nodes without the need to re-label them in an exhaustive and expensive way [5]. Second, the condition of the ethical and commercial feasibility of autonomous EDA systems lies in overcoming the Explain ability Imperative. It is important to continue with research and application of XAI methods, including LIME and SHAP, to guarantee model transparency, keep designers trusting, allow effective debugging, and guarantee the required level of accountability and governance that the next generation of semiconductor manufacturing will need.

References

1. J. Singh and D. Singh, Applications of AI/ML algorithms in VLSI design and technology, *Integrated Devices for Artificial Intelligence and VLSI*, pp. 157–191, 2024.

2. D. Garg and D. K. Sharma, Towards the evaluation from low power VLSI to quantum circuits, in *Quantum-Dot Cellular Automata Circuits for Nanocomputing Applications*, CRC Press, pp. 1–24, 2023.

3. A. Chauhan and D. Patle, A review on digital design and optimization of VLSI architecture.

4. G. Thakur and S. Jain, Role of artificial intelligence in VLSI design: A review, *Recent Advances in Computer Science and Communications*, vol. 18, no. 1, 2025.

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

6. S. Yu, S. Du, and C. Yang, A deep reinforcement learning floor planning algorithm based on sequence pairs, *Applied Sciences*, vol. 14, no. 7, p. 2905, 2024.

7. Y. Li, M. Liu, A. Mishchenko, and C. Yu, Verilog-to-PyG-A: A framework for graph learning and augmentation on RTL designs, in *Proc. IEEE/ACM Int. Conf. on Computer Aided Design (ICCAD)*, pp. 1–4, 2023.

8. D. Z. Pan, Y. W. Chang, C. H. Hsu, and Y. Hsu, From classical algorithms to AI: Evolving trends in VLSI physical design automation, *IEEE Design & Test*, 2025.

9. Y. Shin, AI-EDA: Toward a holistic approach to AI-powered EDA, in *Proc. ACM/IEEE Workshop on Machine Learning for CAD (MLCAD)*, pp. 1–3, 2023.

10. S. Yu and S. Du, VLSI floor planning algorithm based on reinforcement learning with obstacles, in *Biologically Inspired Cognitive Architectures Meeting*, Springer, pp. 1034–1043, 2023.

11. Z. H. Kong, A. James, S. K. Ata, K. M. M. Aung, C. S. Foo, A. James, and K. S. Yeo, AI-powered revolution: Redefining analog circuit design and optimization for tomorrow.

12. B. R. Bhowmik, AI technology in networks-on-chip, in *Industrial Transformation*, CRC Press, pp. 99–128, 2022.

13. DSO.ai: AI-driven design applications – Synopsys, available online, accessed Nov. 21, 2025.

14. How AI/ML is being integrated into VLSI design and verification – Inskill Courses, available online, accessed Nov. 21, 2025.

15. AI chip design – AI-powered EDA solutions, Synopsys, available online, accessed Nov. 21, 2025.

16. J. Pan, G. Zhou, C. C. Chang, I. Jacobson, J. Hu, and Y. Chen, A survey of research in large language models for electronic design automation, *ACM Trans. Design Automation of Electronic Systems*, vol. 30, no. 3, pp. 1–21, 2025.

17. W. Fang, W. Li, S. Liu, Y. Lu, H. Zhang, and Z. Xie, NetTAG: A multimodal RTL- and layout-aligned netlist foundation model via text-attributed graph, *arXiv preprint arXiv:2504.09260*, 2025.

18. R. Dwivedi *et al.*, Explainable AI (XAI): Core ideas, techniques, and solutions, *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–33, 2023.

19. T. Rieger, D. Manzey, B. Meussling, L. Onnasch, and E. Roesler, Be careful what you explain: Benefits and costs of explainable AI in a simulated medical task, *Computers in Human Behavior: Artificial Humans*, vol. 1, no. 2, p. 100021, 2023.

20. P. Hermosilla, S. Berríos, and H. Allende-Cid, Explainable AI for forensic analysis: A comparative study of SHAP and LIME, *Applied Sciences*, vol. 15, no. 13, p. 7329, 2025.

21. How is AI revolutionizing physical design optimization in VLSI engineering? – ACL Digital, available online, accessed Nov. 21, 2025.

22. AI in VLSI physical design: Opportunities and challenges, available online, accessed Nov. 21, 2025.

23. E. Celik and D. Dal, A systematic review of machine learning-driven design space exploration in high-level synthesis, *Integration*, p. 102513, 2025.

24. A. Dev, M. M. Fouda, and S. Chiu, The role of artificial intelligence in transforming VLSI design, testing, and manufacturing, in *Proc. Intermountain Engineering, Technology and Computing (IETC)*, pp. 1–6, 2025.

[1] *Department of Electronics and Communication Engineering, IIMT Engineering College, Meerut, India*

[2] *Department of Electrical and Electronics Engineering, IIMT Engineering College, Meerut, India*

[3] *Department of Electronics and Telecommunications Engineering, Symbiosis Institute of Technology, Pune Campus, Symbiosis International (Deemed University), Pune India*

E-mail address: deepakgarg1985@gmail.com, guptamanoj8485@gmail.com, sumikg@gmail.com