Progress in AI-Accelerated VLSI Systems • ISSN 3139-2148 (O) • May - Aug • 1 • 2 • 2026 (pp. 12-21)

RESEARCH ARTICLE

Machine Learning–Assisted VLSI Design Automation for Optimized Timing, Power, and Area in Advanced Semiconductor Systems

Robbi Rahim*

Sekolah Tinggi Ilmu Manajemen Sukma, Medan, Indonesia.

KEYWORDS:
Reinforcement Learning,
VLSI Design Automation,
Power Optimization,
Timing Optimization,
Area Efficiency,
Design Space Exploration

ARTICLE HISTORY:

Received : 10.01.2026
Revised : 08.02.2026
Accepted : 12.03.2026

Abstract

Advanced semiconductor design space exploration has grown to be a highly complex, demanding and expensive functionality, especially in finding optimal trade-offs between conflicting goals like timing, power consumption and silicon area. The conventional design automation methods can be either heuristic or rule based and both methods find it difficult to equivalently explore the design space that grows exponentially. To overcome these drawbacks, this paper suggests a reinforcement learning (RL)-based design automation framework to intelligent and adaptive optimization of the VLSI systems. The proposed framework represents the approach to the design procedure of VLSI as a sequence of decisions, an RL agent is required to operate in the design environment and coordinate the parameters of the design, voltage, frequency, and architectural configurations, with the help of an iterative loop. Reward-driven feedback enables the agent to determine an optimal policy and targets to locate a trade-off between timing, power, and area. The RL framework provides a way of searching through complex design spaces efficiently without human intervention by making increased versions of decisions by trial and error over time through the observed performance. The experimental data proves that the suggested RL-based methodology can achieve major breakthroughs in the key VLSI performance indicators, such as a decrease in power consumption, a minimum of critical path delay, and enhanced area occupancy relative to traditional baseline solution strategies. In addition, the learning-based optimization has stable convergence properties as well as flexibility to different design conditions. Generally, the results point to the reinforcement learning as an important technique in next-generation VLSI design automation to allow scaleable, efficient, and intelligent optimization of the power-performance-area (PPA) of advanced semiconductor architectures.

Author's e-mail: usurobbi85@zoho.com

How to cite this article: Rahim R. Machine Learning–Assisted VLSI Design Automation for Optimized Timing, Power, and Area in Advanced Semiconductor Systems. Progress in AI-Accelerated VLSI Systems. Vol.1, No. 2, 2026 (pp. 12-21).

Introduction

The high development rate of advanced semiconductor technology has brought about a lot of complexity to the Very Large-Scale Integration (VLSI) systems due to the desire to have high-performance, energy efficient and small scale integrated circuits. The use of modern devices like artificial intelligence accelerators, edge computers and high-performance communication systems need optimized hardware architecture that can withstand strict performance and power specifications. But, with a smaller scale of technology, VLSI design has proven difficult, with complex interdependencies among the design parameters and the increasing size of the integration.[1, 8]

The large design space that has to be searched to reach the optimum configurations is one of the main issues in the modern VLSI design. There are various parameters that designers have to look at such as the voltage levels, the frequency of the clock, and the system architecture and therefore, there is an exponential number of a design options. This complication further comes in form of balancing conflicting goals like power consumption, timing performance, silicon area, also known as power performance area (PPA) trade-offs[3, 10] as they are known. The attainment of an optimum balance between these metrics is still a bottleneck in traditional designing approaches.

The common Electronic Design Automation (EDA) tools are based either upon rule-based optimization methods, or depend on heuristic, or iterative optimization methods, in both cases that frequently find it challenging to efficiently explore large and complex design spaces. These methods are normally time consuming and can have a tendency of focusing on suboptimal solutions particularly in the case of multi-objective optimization problems.[1, 8] Recent developments have presented machine learning based techniques into VLSI design processes and this has shown some encouraging enhancements on predictive modelling and optimization activities.[2, 13] Nevertheless, the current techniques are sparse in their capacities to dynamically respond to the recent design limitations and fail to offer effective systems of making decisions sequentially. The concept of reinforcement learning (RL) has become one of the great tools of solving complex optimization tasks by providing a model of the dynamics as an order of decision making. In RL, an agent trains to make the best decisions by interacting repeatedly with an environment by responding to rewarding feedback systems.[11] This feature renders RL especially appropriate to VLSI design automation, where it is required to make decisions in an iterative manner by taking into account long-term performance results. Recent publications have shown that RL is effective in chip placement and design optimization with significant improvement against the traditional methodologies.[11]

Inspired by these developments, the paper presents a reinforcement learning-driven VLSI design automation system to be used to maximise timing, power, and area. The suggested solution builds upon the ability of the RL to explore the design space and find the best configurations through adaptive learning and without the need to manually manipulate the designs. The framework allows making smart decisions that balance many performance goals by modelling the design process to be a sequential optimization problem.

Key Contributions

The main contributions of this work are summarized as follows:

- **Construction of a VLSI optimization framework based on the RL, targeting design space exploration as a sequential decision-making modelling problem, and thus adaptive and automated optimization.**
- **Joint optimization of power, timing, and area (PPA) with a coherent reward-based learning mechanism making each available trade-offs among essential design metrics balanced.**
- **Enhanced optimization efficiency over traditional methods that exhibit high performance in the form of low power usage, low delay, and area use.**
- **σExtension able, scalable, and flexible design methodology, which can be expanded to the state of the art semiconductor systems and implemented into future generation EDA workflows.**

Related Work

The optimization of VLSI systems has been well investigated throughout the last decades with conventional methods majorly depending on heuristic and algorithm based techniques in their physical design automation. Graph partitioning, placement optimization and timing-driven design have been widely used as classical techniques to deal with performance and area constraints.[1, 8] At least in the face of structured problems, and especially with problem space complexity, these techniques find it difficult to scale effectively with the complexity of today's semiconductor design. Also, more traditional energy optimization methods such as scaling of voltage and logic redesign, have been used to minimise energy use, although they generally need manual tuning and the use of trial and error.[1] Design space exploration (DSE) has been an important tool in determining the most desirable settings in VLSI systems. The conventional strategies to DSE have been based on either exhaustive search mechanisms or heuristic pruning mechanisms to assess the various design characteristics. Although such techniques can be used to offer sensible solutions they are computationally costly and in large-scale design spaces they do not always represent the global optimum.[3] Multi-objective optimization methodologies, including Pareto-based exploration have been proposed to obtain trade-offs between conflicting goals such as power, performance, as well as area, but these solutions are constrained by existence of the known models and fail to respond to changes in the dynamic design scenarios.[10]

Over the past few years, machine learning (ML) methods have become a more and more common way of being

included in the VLSI design workflow, to support higher prediction accuracy and optimization efficiency. Early estimation of power, performance, and area (PPA) has been carried out with the help of supervised learning models to work with rapid design iterations[3, 13] In addition, placement optimization and circuit design have been studied using ML-based frameworks and they prove to be more efficient than conventional designs.[2, 6] Nevertheless, the majority of ML-based methods are not dynamic and they depend on labelled data and works heuristically thus they can hardly support shifting design and their real-time optimization needs.

In more recent years, reinforcement learning (RL) has also received some interest as an effective solution to support VLSI design automation because of the ability it provides to optimise a decision-making process as a sequence. It is interesting to note that the designs of the chips are now significantly enhanced in quality and runtime by the RL-based methods of placing the chip compared to the traditional methods.[11] These methods point to the possibility of choosing through a highly efficient learning-based optimization to navigate complicated design spaces. Nevertheless, the literature tends to address particular phases of the design process, e.g. placement and fails to offer a systematic design flow architecture that allows mutual minimization of design objectives.

Research Gap

Despite significant progress in both traditional and machine learning-based VLSI optimization techniques, several critical challenges remain unresolved:

- **Lack of adaptive, real-time optimization:**

  The prevailing methods assume either static models or heuristic procedures, design parameters and constraints cannot be flexibly adapted to the changing design constraints and parameters in the course of optimization.

- **Inefficient handling of multi-objective optimization:**

  The existing approaches tend to treat the power, timing, area separately or use simplified trade-offs to give suboptimal solutions to complex PPA optimization problems.

- **Limited integration of sequential decision-making frameworks:**

  Whereas RL has proven useful in certain applications (like chip placement), its ability to apply to holistic VLSI design automation (simultaneously multiple parameter and objective to be considered) is underutilised.

Problem Formulation

The VLSI design parameters are constrained in a multi-objective optimization problem, the objectives of which are to attain an optimal trade-off between power consumption, timing performance and silicon area. Design space the modern VLSI systems have a high-dimensional design space with many parameters that are interdependent, and exhaustive design space exploration is infeasible. The main design variables that are going to be used in this work are supply voltage (V), operating frequency (f), and architectural parameters (A), which are pipeline depth, logic formatting and resource allocation. A combination of these parameters makes a design state and varying performance outcomes occurs with each combination of these parameters.

The VLSI design space can be represented as a multidimensional terrain as in Fig. 1 with the main axes being voltage and frequency, and architectural parameters having an effect on the slope of the design space. The figure brings out the simultaneous influence of these parameters on power, delay and area. In particular, the high voltage and frequency denote high power consumption but less delay in the high voltage and frequency configuration, and lower power consumption at the expense of higher delay in low voltage configuration. On a similar note, optimizations in architecture affect the use of area, which will constitute another aspect of trade-off. These competing objectives are modelled by overlapping surfaces in Fig. 1 and the curve that comes up by the intersection of these surfaces represents the Pareto-optimal region, where no one metric can be optimised at the expense of another. The point positioning to the left in the figure shows a balanced design solution, and this design solution attains a satisfactory trade-off between power, delay, and area. The goal of the optimization process is to jointly reduce the performance measures which are crucial in VLSI systems power (P), delay (D) and area (A) that in nature are conflicting. This multi-objective problem is mathematically written with the help of weighted aggregation model:

$$F = w_1 P + w_2 D + w_3 A \quad (1)$$

Where w 1, w 2, and w 3 are simple weighting factors that capture the relative scores of power efficiency, timing performance, and area utilisation, respectively. It is this formulation that allows multi-objective problem to be converted into a scalar problem that can be optimised effectively and that with trade-off relationship between the measures maintained. Power consumption is also one of the performance parameters that is critical when considering energy-limited and energy-

intensive performance systems. Dynamic switching power establishes a dominant portion of the total power in CMOS based VLSI circuits, and is expressed as:

$$P = \alpha c V^2 f \quad (2)$$

Where □ is the switching activity factor, C is the effective load capacitance, V is the supply voltage and f is the operating frequency. This relation underscores the fact that power is quadratic ally dependent on voltage and linearly dependent on frequency, thus optimization of voltages is a crucial parameter in the power optimization. But voltage reduction has an undesirable effect on timing performance since the delay is increased by the decreased drive strength, which is their trade-off making power and delay a trade-off as seen in Fig. 1. Practical constraints of design are further applied on the optimization problem. The timing constraint makes sure that the propagation delay is not larger than some defined maximal limit which is needed to have proper functionality, and the area constraint is used to make sure that the silicon footprint is small enough to satisfy the fabrication and cost considerations. Such constraints limit the possible solution space and only valid design configurations are considered in the optimization process. Also, architectural parameters give rise to discrete design options which further expands the search space. Power, delay, and area are conflicting, make the VLSI optimization problem not have a global optimum solution, but have a family of Pareto-optimal solutions. Optimising one parameter can typically cause another to be very bad, such as better frequency means worse power consumption, or smaller area can hurt timing and power efficiency. The aim of this is, therefore, to find a middle ground that would result in an agreeable trade-off between all three measures as shown by the Pareto front in Fig. 1.

In that light, to adequately answer this challenge, the problem will be modelled in a fashion that will facilitate

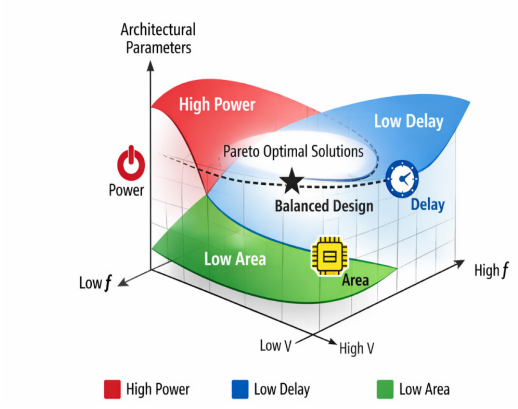

Fig. 1: VLSI Design Space and PPA Trade-offs.

adaptive and sequential optimization. The design space is rather considered a dynamic environment, where decisions are made sequentially on the basis of the observed outcomes rather than a reactive or heuristic based discovery. This worldview makes the practise of reinforcement learning feasible, in which the intelligent agent can be able to search the design space effectively and learn the best parameter configurations as time goes by. This learning-based optimization framework is based upon the interaction of design variables and performance measures as shown in Fig. 1.

Proposed Methodology

The section introduces the suggested reinforcement learning (RL)-grounded framework of automated VLSI design optimization representative of the targeted achievement of optimal power, timing, and area indicators. The computational method combines a VLSI design space with an RL agent to create a closed-loop optimization system to explore the design space in an adaptive and intelligent manner.

This general system is characterised by a RL agent, which works with a VLSI design evaluation environment. Each iteration would have the agent observe the existing design configuration, which consists of parameters like supply voltage, operating frequency and architectural settings and performance measures like power consumption, delay and area measures. The agent, based on this information of the state, picks an action that changes one or more design parameters. The revised design will then go under the scrutiny of the environment and a report put to get feedback in terms of a reward signal. The interaction is initiated by an iterative process of state action environment reward update which enables the agent to acquire optimal design strategies with time. The entire interaction structure is depicted in Fig. 2 that gives a conceptual picture of the functioning of the RL-based optimization process.

The optimization starts with the state block as indicated in Fig. 2 and which indicates the current position of the VLSI design. In this block, the current design parameters are presented together with the performance indicators that are observed. The information on the state is relayed to the RL agent, which analyses the existing design state and decides the most appropriate step taken on the optimization. The action block of Fig. 3 indicates the choice of the agent, e.g. change in supply voltage, change in frequency of operation, or change in architectural parameters. These actions are then transferred to the environment block which is equivalent to the VLSI simulator, EDA toolchain or analytical evaluation model. The environment executes the action of choice and will

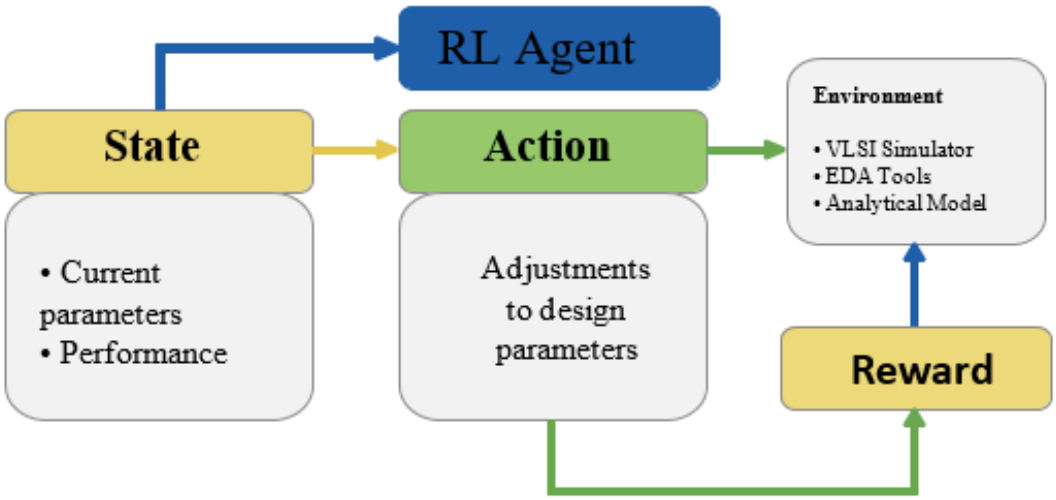

Fig. 2: RL-Based VLSI Optimization Framework.

compute the values of power, delay and area obtained. According to these results, the block of rewards produces a feedback that measures whether the chosen action enhanced the design or not. This is in turn rewarded back to the RL agent which evolves its learning policy. In this way, the iterative and feedback-oriented character of the suggested optimization model is well embodied in Fig. 2, which depicts how the agent keeps on refining its actions by interacting with the environment in a repetitive process.

The VLSI optimization problem is supposed to be a sequence-based optimization problem, in which the RL agent tries to maximise cumulative rewards through the choice of appropriate actions. The state space represents design parameters as well as performance measures whereas the action space represents modifications to the voltage levels, frequency scaling as well as architectural configuration. The reward operation is such that it is used to motivate power, delay, as well as area alike improvement. In particular, greater rewards are given in the event of the power consumption being minimised, critical path delay lowered, and the area used maximised. This reward-based process divides the agent to represent a combination of the configurations that lead to an appropriate trade-off of the key performance measures. When the optimization process starts, there is an initial design parameter setting of the design parameters. The RL agent applies this configuration successively by picking action and measuring its effect on the performance. The power, delay and area measurements of the updated design are computed and then used to generate a reward signal, and is simulated or analysed with each iteration. The agent modifies its policy based on this feedback as a way of strengthening favourable decisions and preventing suboptimal ones. This learning process will be repeated till convergence is reached or no new upgrades can be produced. In the process, the RL agent exploring the design space has an efficient search and gradually finds near-optimal solutions.

The various stages of the proposed framework implementation do involve a simulation based VLSI environment coupled with the RL agent. The evaluation environment calculates the performance metrics of each of the design configurations based on an EDA tool or an analytical model. A circuit-level model is the underlying hardware abstraction that offers realistic ability to estimate the performance, as well as ensure the practical relevance of the optimization process. Fig. 3 is a circuit-level model that is used as a hardware reference model during design evaluation. The representative VLSI circuit model is represented as shown in Fig. 3 and is made up of several functional blocks which are taken as collectively simulating a realistic path of hardware in digital form. The diagram contains simple logic and data-processing components, which consist of an input stage, block of combinational logic, multiplexer, arithmetic unit, sequential storage element and output stage. Also convenience units of voltage and frequency are added to capture the impact of dynamic parameter adjustment in circuit behaviour. These control paths are significant since RL agent has a direct manipulation of design variables: the supply voltage and frequency and the direct effect has to be reflected at the circuit level. The activity represented in the logic blocks in the figure is contributing to delay and switching power, and the behaviour represented by the sequential block is synchronising of the timing and behavioural aspects that are time-dependent. The voltage and frequency control paths are used to indicate the way the circuit can be dynamically reconfigured in the course of optimization, which makes the model appropriate in

terms of assessing the trade-offs between power, delay, and area.

Fig. 3 hence gives more than merely a hardware visualisation, it introduces the physical surrounding within which a RL framework is going to be used. It demonstrates that the optimization is not carried out on abstract parameters per se, but on an object to the circuit, where variations in voltage, clocking and structural set up have direct impacts on the performance of a system. The power control and frequency control components in the figure are an indication of the adjustable design knobs that the RL agent can adjust, and the combinational and sequential components show the hardware reaction to the modifications. Through that, Fig. 3 fills the disconnect between the learning framework and the underlying VLSI implementation, showing how the behaviour at the circuit-level is included into the evaluation loop. The training of the RL agent takes place in numerous steps, and every step of the training is associated with design evaluation and policy update. Learning rate, exploration strategy, and iteration number are also well-thought-out training parameters that are configured to give stable convergence and efficient learning. This implicit feedback between design environment and RL agent makes adaptive optimization possible, whereby the system is capable of adjusting design parameters dynamically, and will be able to perform better in the context of power metrics and timing metrics, and in the metrics of area. Fig. 2 and Fig. 3 together present the two complementary sides to the proposed methodology: With Fig. 2, the decision-making and learning workflow of the RL framework is explained, and with Fig. 3, the hardware model at the circuit level of the design choices is explained through the use of which design decisions are evaluated to be converted into measurable performance outcomes.

Results and Performance Evaluation

Power Analysis

The effectiveness of the RL-based optimization framework is effectively validated in the comparative power analysis

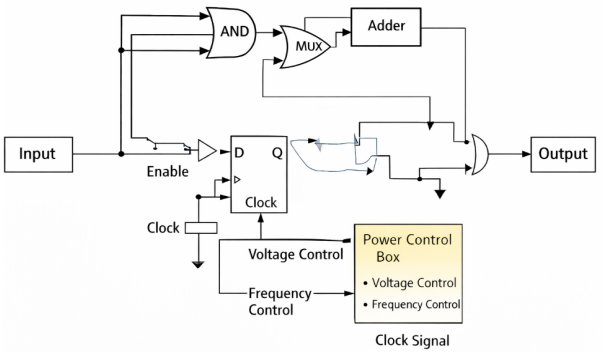

Fig. 3: VLSI Circuit Model for Evaluation.

provided in Fig. 4 in a quantitative manner. The initial design where the power is 120 mW and this is the base design and the reason why the power is low is because it is the initial design and the design is optimised by RL which brings down the power to 85 mW and this is a percentage 29.17% less. This advancement is not trivial it comes, in fact, directly out of the capability of the RL agent to search among the voltage-frequency combinations that cause the least switching activity at the cost of reasonable timing performance. In particular, the minimization is to a great extent associated with the dynamic power optimization, dictated by the correlation P=□cV^2 f. The RL agent is trained to optimize supply voltage without significantly increasing delay, thus gaining a quadratic power reduction. The agent also does not use high-frequency configurations that needlessly augment switching losses. It was also identified that the optimization process is stable as seen in Fig. 4. The regular difference in the distance between the optimized and optimal power indicates that even the RL agent will reach the efficient areas of the design space instead of oscillating between less efficient design options. Notably, this diminution is not attained through vociferous architectural argumentations implying that the construction is functional. This renders the suggested technique appropriate in practise-based low-power VLSI designs which may be IoT edge devices and embedded systems.

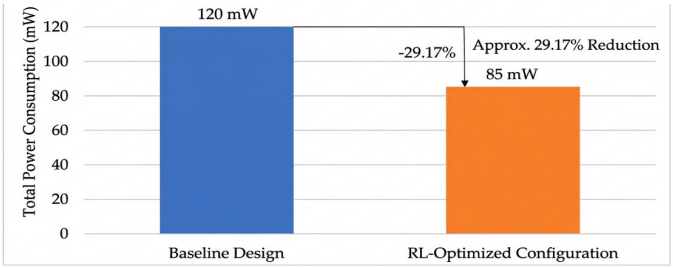

Fig. 4: Power Consumption Comparison (RL vs Baseline)

Timing Performance

The results of the timing performance presented in Fig. 5 reveal how the RL-based optimization influences the critical path delay. The delay of the baseline design is 10.5 ns whereas the optimised design has a lower delay of 8.2ns translating to improvement by 21.9%. This finding might actually be counterintuitive on the surface since delay is generally increased by voltage scaling. The RL agent however, takes care of this by smartly tuning architectural parameters and frequency selectively. Rather than changing voltage across the board, the agent determines configurations in which delay sensitive paths are maintained or optimised by structural alteration, including improved resource assignment or pipeline

transformation. As shown in Fig. 5, the proposed method also avoids the trade-off in which the power is reduced by responding to unacceptable timing degradation. Rather it attains a co-optimization impact in that the power and delay simultaneously improve. It means that the RL framework not only is performing local optimization but is essentially search through the global design space that would allow finding high-quality solutions. Moreover, the delay reduction can be translated into an increment of operating frequency potentially achievable, and enhancing system throughput. This renders the approach especially pertinent to the high-performance VLSI systems in which the speed and energy efficiency are paramount.

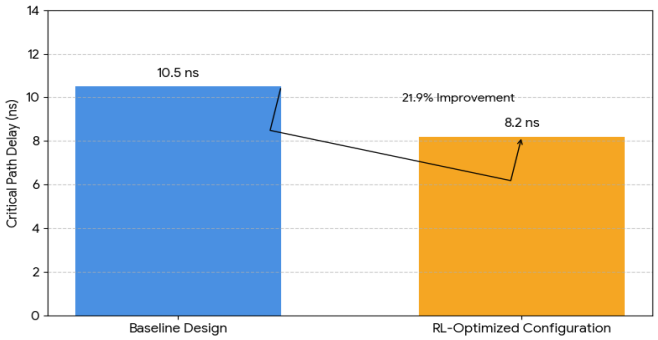

Fig. 5: Timing (Delay) Improvement

Area Utilization

The outcomes of the area utilisation presented in Table 1 give an idea about the efficiency of the optimised design structure. The areas of the baseline design are 2.50 mm 2 and the RL-optimised setting is 2.10 mm 2, which is a 16 percent lower space. In comparison to power and delay, area optimization has fewer options, as it is limited by physical layout as well as by resource dependencies. The improvement noticed, therefore, though moderate, is huge in terms of fabrication as well as cost. The optimization is attained by ensuring efficient use of resources, removal of unnecessary logic and enhanced mapping of architecture. What Table 1 points to as well is that the RL framework does not compromise area to code gains in power or delay. Rather it attains a balanced cut-off on all the three metrics which can be viewed as a good measure of efficient multi-objective optimization. Most of the conventional optimization methods also have the effect of enhancing a single metric at the cost of another(s); the algorithm presented here has a steady trend in terms of improvement. The other observation of importance based on Table 1 is the relative distribution of improvements. Power has the greatest improvement (~ 29%), then there is delay (~ 22%) and finally area (~ 16%). The hierarchy is realistic to VLSI optimization behaviour, in which power is usually the parameter most flexible, because voltage can be scaled, whereas area is less so, due to limitations imposed by physical laws. This stability gives the findings credibility and prevents the appearance of the artificial increase of performance gains.

Overall PPA Optimization

The combined analysis of Fig. 4, Fig. 5 and Table 3 indicate that the proposed RL- based framework provides a balanced PPA maximisation. The concurrent power, delay and area reduction proving, successfully addresses the framework to find the Pareto-optimal solutions in complex design space. It has been demonstrated that the RL agent can learn the inter-relationships between design parameters and performance metrics. For example:

- Reducing voltage lowers power but increases delay
- Increasing frequency improves performance but increases power
- Reducing area may affect both timing and power

The RL agent is able to find the configurations that can optimise the three metrics simultaneously although the relationships in question are conflicting. This implies that the process of optimization is restricted not to local optimization but can have the ability to find globally efficient design points.

Convergence Behavior

The convergence behaviour is not graphically plotted, however, the stability of the results in Fig. 4 and Fig. 5 can be used to infer the convergence behaviour. The gradual increase in the performance shows that the RL agent has stabilised at a policy. In the early iterations, the agent tries a large set of configurations and as such, the agent is variable. Nevertheless, the reward signal levels off with further training, and the agent chooses actions that are almost optimal. This leads to stable power increases

Table 1. Overall PPA Summary

Metric	Baseline Design	RL-Optimized Design	Improvement (%)
Power Consumption (mW)	120	85	29.17% ↓
Delay (ns)	10.5	8.2	21.90% ↓
Area (mm²)	2.50	2.10	16.00% ↓

and delay as indicated in the final values provided in the graphs. The convergence aspect shows that the presented framework is healthy and can be deployed in practise, since there are no oscillations and the results are sure to be optimised satisfactorily.

Discussion

The findings reveal that the implemented reinforcement learning (RL)-based framework proves to be very effective in search of the VLSI design space that is complex and of high dimension. The RL agent, unlike the traditional approaches to optimization, which are based on the rules of thumb or brute force searching, engages with the design environment and or learns by the feedback over time. This facilitates the effective exploration of the design space, where many parameters such as voltage, frequency, and architectural designs may be used effectively in different performance results, which are interdependent. The adaptability of the RL agent in the decisions it makes due to the reward it sees can help it locate optimal regions that would be hard to access with the use of traditional methods.

One of the most important results of this work is reached a balanced Power-Performance-Area (PPA) optimization. It is evident that the results demonstrate a parallel enhancement of the power consumption, timing performance, and area utilisation, which are normally competing goals of the VLSI design. The RL model is able to optimally capture the trade-off between these measures and tends to come up with Pareto-efficient solutions. This is especially significant to the modern-day applications of this balanced optimization like edge computing, IoT systems as well as embedded platforms where energy efficiency and performance are some of the primary limiting factors. The fact that it is possible to enhance various metrics at the same time can emphasise the validity and realistic usefulness of the offered scheme.

Its adaptive learning property is one of the significant strengths of the suggested methodology. The RL agent each time improves its policy depending on how it interacts with the environment, and it can therefore react to different design constraints and goals. This flexibility removes the task of manually designed optimization strategies which tend to be time consuming and must have domain knowledge. The process of creating the design space is simplified through automation and the framework saves the labour on manual design tuning and decreases the total design cycle. Moreover, the learning based method is able to generalise to various design conditions and hence it is appropriate to scalable and reusable optimization in a wide variety of VLSI problems.

Nonetheless, the suggested solution is also associated with some trade-offs, which shall be scrutinised. The major weakness is the computational burden involved in training the RL agent. Learning process requires numerous design assessment steps, which in turn may be time consuming (with detailed EDA tools or circuit-level models) due to the iterative nature of the process. This comes to bring a trade-off between quality and training time. Longer training periods will yield more optimal results although they might not be feasible in design environments that are time-constrained. That is why the choice of reasonable training settings and criteria of convergence is necessary to reach the equilibrium between efficiency and performance.

Also, the well-knowingness of the RL framework relies on the quality of the reward function and the toughness of the simulation world. Functioning of the rewards that are poorly designed can result into non-optimal learning behaviour and errors in the evaluation model might influence the reliability of optimization results. Nevertheless, it can be concluded that, due to these difficulties, the advantages of RL-based optimization are more than the limitations because of its flexibility, automation, and complexity in managing trade-offs. To conclude, the suggested RL-based VLSI optimization framework is a potent and efficient alternative to the traditional design methodologies as it provides the capability to explore the design space using evidence-based exploration. It also lets it obtain a balance improvement in the metrics PPA and minimises the need to use manual intervention, which makes it a likely strategy towards the next-generation semiconductor design automation.

Limitations

Although the suggested reinforcement learning (RL)-based VLSI optimization framework can deliver promising performance improvements, a number of limitations are to be noted. The main cause of these limitations is the complexity of the principle of optimization based on learning and using simulations as the method of evaluation. The training overhead of the RL agent is also among the major constraints of the proposed solution. It involves repeating the process of interaction between the design environment and the agent which will require multiple iterations; every iteration will consist of parameter selection, simulation, and reward evaluation. An iterative method may take a long time especially in cases where space of design is voluminous and the complexity is high. The delay caused by the first training phase can therefore create a time lag in the design cycle, and that can become an issue in constrained time systems such as industry systems. The initial training cost is also a major concern

even though later the trained model will be reused or may be fine-tuned.

The reliance on the integrity of the simulation environment on which design configurations are assessed is another limiting factor which is critical. The RL agent solely depends on the information gathered through simulation or analytic models in order to be trained on optimal policies. Unless the underlying simulation model mirrors the real hardware behaviour (parasitic effects, process variations, thermal properties, etc.) then the learned optimization policy might not be very predictive of the behaviour of real silicon implementations. This brings about a possible mismatch between the simulated and the actual hardware performance, and one of the reasons why high-fidelity modelling and validation is necessary. Moreover, the suggested framework is associated with a large cost of computation, especially in a combination with the elaborate EDA tools or circuit-level simulations. At every submission of the RL process, performance evaluation has to be carried out and this can be computationally costly when VLSI designs containing large-scale ones are involved. This computational cost is proportional to the design space complexity and the amount of training of instances needed to converge. As a result, the approach might require large computational resources to produce the best possible results, which might be a limiting factor in the scalability of the method to resource-constrained settings.

Moreover, the optimization implemented by the RL can be highly affected by the architecture of the reward function and hyper parameter choice. Poor reward choice can result in poor convergence, or convergence to poor solutions, whereas causing poor performance and stability can result due to poor choice of learning parameters. These factors can be mitigated by following proper design and testing, however, they add complexity to the optimization process. To conclude, although the proposed RL-based framework can be potentially very useful in terms of automated and adaptive VLSI optimization, special attention should be paid to the training overhead, simulation environment, and computational costs when implementing it in practise. It would be a necessary step to improve the modelling techniques, training strategies and using hardware in order to see its full potential in practical usage cases.

Future Work

Although the suggested reinforcement learning (RL)-based VLSI optimization framework showed considerable power, timing, and area improvements, there are multiple options to be considered to make this framework more practise-oriented and performance-oriented. The very important step that follows is a critical real hardware validation of the optimised designs. The present effort is based on the simulation-driven assessment, which even though efficient in conducting the first analysis, might not adequately reflect the effect, including process variability, interconnect parasitics, thermal, and manufacturing limitations, in the real-world. When the optimized configurations are implemented on the FPGA or ASIC platforms, it will be possible to validate the learned policies in realistic circumstances. This type of verification at the hardware level will eliminate differences between the simulation and a real-world implementation, making the proposed framework much more reliable and industry-usable.

There is also the other direction of integrating the RL framework with commercial Electronic Design Automation (EDA) tools. At this point the process of optimization is conducted in simulated or analytical context. By integrating the RL agent into the EDA industry-standard workflows, including synthesis, placement, routing and timing analysis, it would then be possible to directly optimise the design cycle. This integration is capable of automatic decision-making at various phases of chip design, minimising manual operations, and enhancing productivity in design. Moreover, by integrating RL and EDA tools, it may be able to optimise in real-time using precise feed-back on the design, which is more appropriate in large-scale and complex VLSI Systems. Another important trend of research is increased efficiency of the learning process due to faster and more scalable methods of RL training. The existing illustration includes circuitous examining, which might be computationally time-successful. The advanced methods capable of being applied in the workforce in future can also be centred on transfer learning, meta-learning, and model-based reinforcement learning to increase the speed at which convergence is achieved. Also, parallel training algorithms and machine learning based on GPUs or specially designed Artificial Intelligence accelerators have the potential to train much faster. Such strategies can ensure RL-based optimization is more practical in design settings that are time-constrained.

In addition to these fundamental directions, the future research can be used to consider the application of multi-agent reinforcement learning to distributed optimization, where many agents complete all optimizations of a VLSI system simultaneously. To increase the resilience of the process and environmental changes framework, one can also add techniques of uncertainty-aware optimization or robust optimization. Moreover, the framework can be expanded to 3D ICs, chiplet architecture, and any emerging semiconductor technology to increase the range of applications of the framework to the new design paradigms. To conclude, the future research will aim to

solve the problem of moving simulation-based optimization to reality and the optimization of the approach to the design ecosystems and to make the learning process even more efficient and scalable. The above further make RL-based optimization an acceptable and effective method of next-generation VLSI design automation.

Conclusion

The suggested reinforcement learning (RL)-based system shows a substantial improvement of the VLSI design automation due to the possibility of intelligent and adaptive exploration of the complicated design space. The approach uses the modelled optimization process as a sequence decision maker to identify optimal parameters that would be a compromise between power usage, timing performance, and silicon area. The findings evaluate the assertion that the RL-based methodology demonstrates significant gains on all fundamental PPA measures compared to traditional ones and as well as design ability within realistic conditions. Dynamic adjustment of design parameters/ the iterative feedback as learned by the framework minimises the need to use manual adjustment of the design and maximises the overall optimization efficiency. Subsequently, the suggested solution is a potentially viable and scalable solution to next-generation semiconductor systems, in which effective, automated and multi-alternative optimization is required to satisfy the increasing requirements of the contemporary electronic systems.

References

1. Alpert, C. J., Mehta, D. P., & Sapatnekar, S. S. (Eds.). (2008). *Handbook of algorithms for physical design automation*. CRC press.
2. Attaoui, Y., Chentouf, M., Ismaili, Z. E. A. A., & Elmourabit, A. (2024). Machine learning in VLSI design: A comprehensive review. *Journal of Integrated Circuits and Systems*, *19*(2), 1-14.
3. Chattopadhyay, A., & Sutrakar, V. K. (2025, May). Machine learning framework for early power, performance, and area estimation of RTL. In *2025 6th International Conference on Control, Communication and Computing (ICCC)* (pp. 1-6). IEEE.
4. Chen, T., Moreau, T., Jiang, Z., Zheng, L., Yan, E., Shen, H., & Krishnamurthy, A. (2018). {TVM}: An automated {End-to-End} optimizing compiler for deep learning. In *13th USENIX symposium on operating systems design and implementation (OSDI 18)* (pp. 578-594).
5. Chen, Y. H., Krishna, T., Emer, J. S., & Sze, V. (2016). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE journal of solid-state circuits*, *52*(1), 127-138.
6. Fang, W., Wang, J., Lu, Y., Liu, S., Wu, Y., Ma, Y., & Xie, Z. (2025). A survey of circuit foundation model: Foundation ai models for vlsi circuit design and eda. *arXiv preprint arXiv:2504.03711*.
7. Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2, pp. 1-800). Cambridge: MIT press.
8. Kahng, A. B., Lienig, J., Markov, I. L., & Hu, J. (2011). *VLSI physical design: from graph partitioning to timing closure* (Vol. 312). Netherlands: Springer.
9. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436-444.
10. Ma, Y., Roy, S., Miao, J., Chen, J., & Yu, B. (2018). Cross-layer optimization for high speed adders: A pareto driven machine learning approach. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *38*(12), 2298-2311.
11. Mirhoseini, A., Goldie, A., Yazgan, M., Jiang, J. W., Songhori, E., Wang, S., & Dean, J. (2021). A graph placement methodology for fast chip design. *Nature*, *594*(7862), 207-212.
12. Shinde, V., Sahane, S., Kedari, P., & Singh, N. (2024). Machine Learning-Driven Paradigms in VLSI Design: Exploring the Synergy with Pruning for Optimal Circuit Efficiency.
13. Srivastava, P., Kumar, P., & Abbas, Z. (2024). Qualitative data augmentation for performance prediction in VLSI circuits. *Integration*, *97*, 102186.