Language-Model Interfaces for SerDes Design Optimization: Comparing Prompt-Based and Agentic Approaches

Nirjhor Rouf
ECE Department
North Carolina State University
Raleigh, NC, United States
nrouf2@ncsu.edu

Priyank Kashyap
Storage Division
Hewlett Packard Enterprise
Durham, NC, United States
priyank.kashyap@hpe.com

Chris Cheng
Storage Division
Hewlett Packard Enterprise
Milpitas, CA, United States
chris.cheng@hpe.com

Soumyadeep Chatterjee
ECE Department
North Carolina State University
Raleigh, NC, United States
schatte5@ncsu.edu

Sounak Dutta
ECE Department
North Carolina State University
Raleigh, NC, United States
sdutta6@ncsu.edu

Paul D. Franzon
ECE Department
North Carolina State University
Raleigh, NC, United States
paulf@ncsu.edu

***Abstract*—Modern SerDes/SI-PI optimization workflows are effective but often require significant domain expertise and careful manual iterations, creating friction in early design-space exploration. While LLM-based natural language interfaces offer a path to lower this interaction barrier, it remains unclear whether prompting alone can substitute for algorithmic optimizers in practice. We present the first comparative study of LLM-driven design optimization for SerDes/SI-PI, contrasting a purely Prompt-Based scheme against an agentic workflow that integrates Bayesian optimization. Our experiments show that Prompt-Based search under-explores the design space, while the agentic approach varies the design parameters effectively. We evaluate both interfaces on a 50 Gb/s NRZ receiver in Ansys Electronic Desktop, jointly tuning three DFE taps and CTLE gain to maximize eye height. Across multiple runs, the Prompt-Based loop often stays near the initial samples and reuses settings, limiting exploration. These results suggest that LLMs are most effective as an interface layer over established optimizers, rather than as a standalone search strategy.**

***Index Terms*—SerDes, SI-PI, LM-guided optimization, LM-driven design**

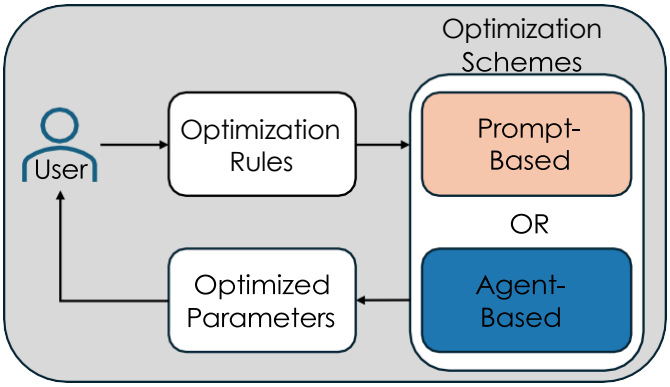


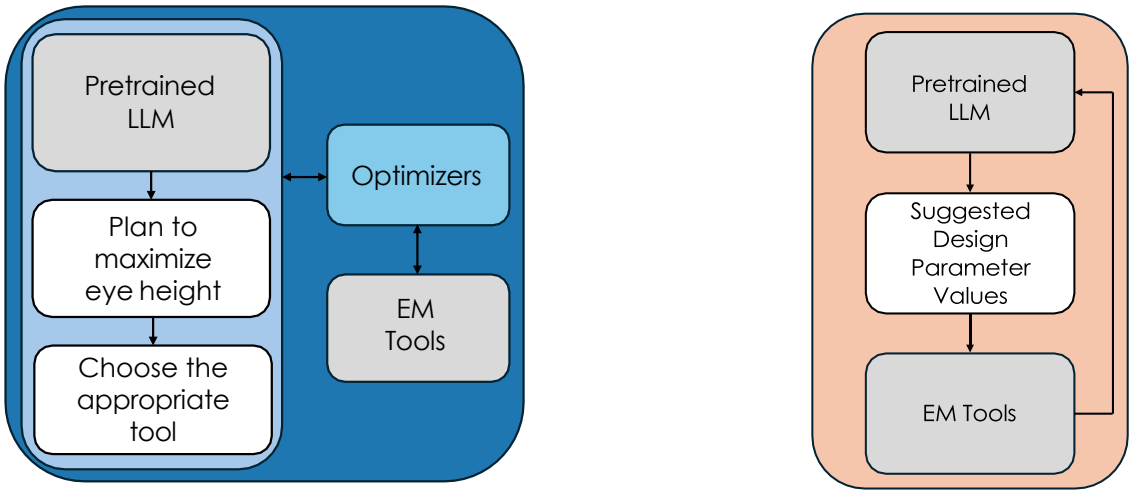
Fig. 1: The typical workflow of how designers interact with optimization tools. EDA engineers consider design targets and constraints for their task and use them with optimizers. Our experiments investigate an emerging paradigm where the optimization schemes are LLM-guided.

I. Introduction and Motivation

With the growing demand for data-intensive workloads — such as generative AI, graph analytics, and image processing — the need for robust high-speed Serializer/Deserializer (SerDes) links is more pressing than ever. Consequently, prior work has leveraged optimization techniques such as Bayesian Optimization (BO), Markov Chain Monte Carlo, and genetic algorithms to tune SerDes links, performing crucial tasks such as optimizing the channel, the receiver, and improving end-to-end performance to efficiently utilize limited channels. [1], [2]. These advancements have greatly accelerated data throughput and are a central component in modern high-speed electronics systems. However, despite their effectiveness, current approaches often require substantial domain expertise and careful manual configuration.

In parallel, large language models (LLMs) have transformed natural language processing by enabling systems to ingest large volumes of information, reason over it, and provide intuitive human-facing interfaces [3]. This capability has made LLMs increasingly attractive for domain-specific workflows, including engineering automation and decision support.

Within electronic design automation (EDA), LLM applications remain relatively early-stage, yet the potential impact is clear: well-integrated LLM workflows can streamline complex de-



(a) Agent-Based (b) Prompt-Based

Fig. 2: We compare two optimization schemes. (a) The Agent-Based approach allows an LLM to directly drive an optimizer. (b) The Prompt-Based approach has an LLM use its common sense reasoning to suggest design parameter values.

sign processes and potentially improve manufacturing yield by accelerating iteration and reducing human bottlenecks in IC design. Recent efforts have already demonstrated progress across multiple EDA subdomains — particularly in physical and analog design — using LLM-driven pipelines for tasks ranging from code/script generation to specification reasoning and verification-oriented assistance [4]–[11]. In contrast, we have seen comparatively limited adoption of LLM-based methods in optimizing system-level design and signal integrity/power integrity (SI/PI) workflows.

Deploying LLMs effectively in EDA requires addressing several practical and methodological challenges. One key concern is unintentional data retention in deep neural networks trained on large corpora [12]. This risk has two important facets - first, sensitive or proprietary IP may be exposed if user inputs are stored or retained by an API; second, models may inadvertently reproduce copyrighted design content without proper attribution, potentially creating downstream legal and compliance issues. A further challenge is the significant compute burden of modern LLMs — for instance, Meta's Llama2-70B requires 130 GB of memory to load [13] — which can limit feasibility in many engineering environments.

Crucially, choosing the model is a non-trivial process, especially when considering the trade-offs between proprietary models and open-source alternatives. Proprietary models such as ChatGPT-4 [14] can be highly capable, but they typically come with restricted accessibility, pay-to-use constraints, limited transparency, and concerns about data handling and storage. In addition, the inability to fine-tune the proprietary models can result in sub-optimal performance in domain-specific tasks. Meanwhile, open-source LLMs are more accessible and customizable, but often face limitations in scale and training resources relative to proprietary systems, which can translate into lower performance on demanding tasks [15].

Motivated by these considerations, we have selected 3 open-source models to effectively perform a comparative study involving two LLM-centered strategies for design optimization of a high-speed SerDes receiver. Specifically, we evaluate a purely prompt-driven method that relies on model reasoning alone (Prompt − Based) against an agentic workflow that integrates BO into the loop (Agent − Based), with the goal of characterizing their respective strengths and tradeoffs in SerDes receiver design.

The paper is structured as follows: Section II presents relevant foundational knowledge, Section III provides information on prior work done on LLM applications in the field of EDA. Section IV details the experimental setup for the presented work, while Section V highlights the key results of the paper. Sections VI and VII summarize the key takeaways and future directions.

II. Preliminaries

A. LLMs

LLMs are transformer-based [16] architectures specialized in natural language processing tasks. While transformers were originally developed with language-to-language translation in mind, modern LLMs, such as GPT-5 [17], have greatly broadened their scope and can now handle a wide range of complex tasks [18]. These models are capable of in-context learning, which refers to the ability to generate outputs for given natural language instructions without additional training. In this work, we utilize in-context learning for both approaches, as we are

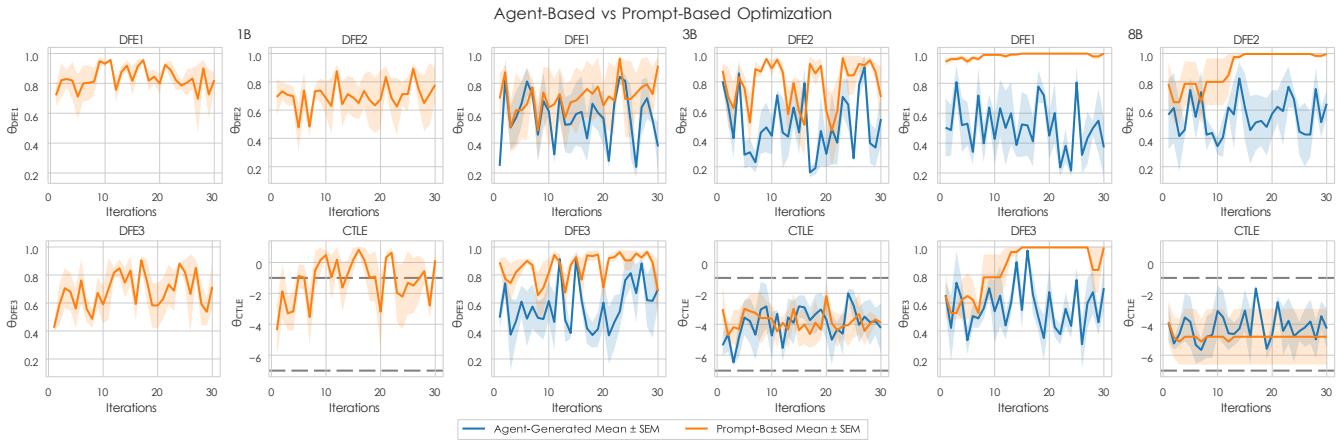


Fig. 3: The design parameters explored by both schemes over the optimization iterations. The gray dashed lines within the CTLE plots denote the minimum and maximum allowable parameter values. Designs induced by parameters outside of this region were considered to be invalid. Note that for the 1B model, the Agent-Based scheme failed to suggest any parameters.

using off-the-shelf pretrained LLMs without any additional fine-tuning. A pretrained LLM can be further fine-tuned on datasets of input-output pairs (prompts and responses). This method, called instruction tuning, enhances the language model's capabilities in following user instructions more closely [19]. Moreover, this helps the model generalize better and perform well on previously unseen tasks [18]. The Llama-3-instruct [20] models used in this work are examples of such instruction-tuned models [18]. Furthermore, it has been demonstrated that LLMs can leverage intermediate reasoning to solve multi-step problems [21]. Lastly, there is a strong correlation between model size and model performance. It has been observed that larger models and data size lead to improved model performance [18] as seen in Section V.

B. Agents

Agents are systems driven by language models that can interact with their environment to achieve a goal across multiple steps. Depending on the design, an agentic system may involve a single agent operating alone or a collection of agents coordinating to solve a task. In many architectures, each agent is assigned a specific role and equipped with tools that enable it to carry out its responsibilities—either independently or in collaboration with others [22]. In our agentic approach, we have leveraged the ReAct [23] framework, where the agent reasons, plans, and then takes action using external tools to fulfill the given optimization task. Specifically, our agent is tasked with optimizing the design of a high-speed SerDes link.

C. Exploration vs Exploitation

The exploration vs exploitation tradeoff is an integral concept in decision-making. In our case, we are attempting to optimize a SerDes channel to achieve the maximum eye height. However, since the best possible eye-opening is unknown to us, the optimization problem heavily involves exploration versus exploitation. Exploitation refers to selecting the action with the highest estimated action-value based on current knowledge. In contrast, exploration involves choosing actions that are not currently estimated in order to gain additional information about the reward structure [24]. This presents a dilemma, as we cannot maximize both factors simultaneously - exploration is inherently costly in terms of resources [25], while exploitation can result in missed opportunities. In this work, the problem reduces to either exploiting the design space to converge to a known eye-height value or effectively exploring the design space to identify a potentially superior solution.

D. Bayesian Optimization

Bayesian Optimization is a sample-efficient method for exploring high-dimensional black box functions, using a surrogate model of the objective function, and efficiently choosing the next sample with an acquisition function to quickly find minima and maxima [26]. This approach is particularly effective for EDA design tasks where design optimization relies on expensive and time-consuming simulations, such as circuit parameter tuning, layout optimization, and design-space exploration under stringent performance and design constraints [27]. In this work, we have utilized an LLM-driven external BO module.

III. Prior Work on LLMs for EDA

Early investigations into the use of LLMs for EDA have yielded encouraging results, demonstrating their applicability across a range of tasks, including chip design, debugging, and script generation. A prominent example is ChatEDA [4], which introduces an RTL-to-GDSII design flow orchestrated by Automage, a fine-tuned variant of Llama-2-70B [13]. In this framework, the LLM serves as a high-level controller that

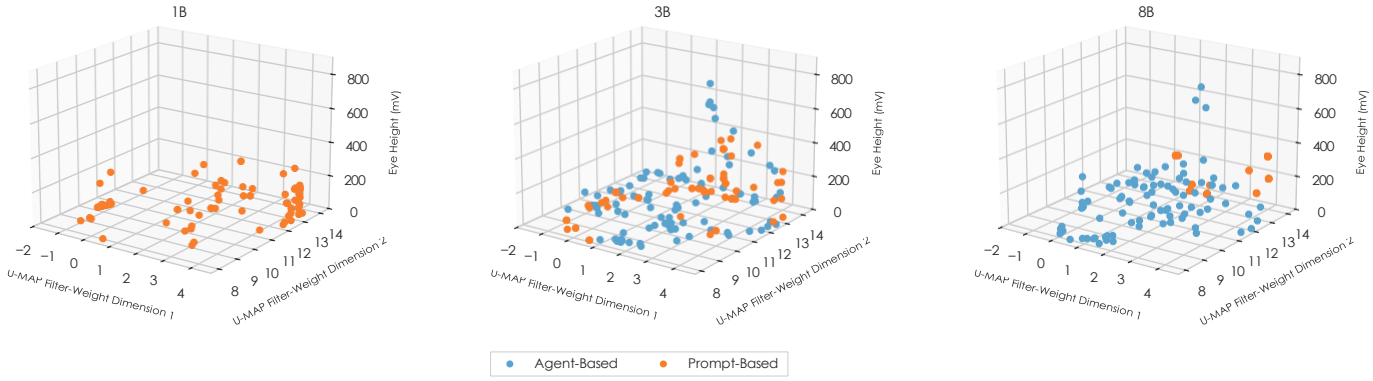


Fig. 4: A U-MAP projection of the design space explored by the two schemes. When using the 8B model, the Prompt-Based scheme suggested very similar parameter ranges numerous times. On the other hand, the Agent-Based approach was able to meaningfully explore the design space and improve eye height.

interacts with EDA tools via APIs to perform performance analysis, grid searches, parameter tuning, custom optimization, and clock-period minimization. This line of work highlights the effectiveness of LLMs as brains that coordinate complex design workflows rather than directly executing low-level design tasks.

In contrast, other efforts position LLMs primarily as productivity-enhancing assistants for hardware engineers. NVIDIA's ChipNeMo adopts this paradigm, leveraging a Llama-2–based chatbot to support engineering workflows through script generation, bug summarization, and analytical assistance [6]. Reported results indicate that ChipNeMo outperforms GPT-4 on engineering assistant and script-generation tasks, while achieving comparable performance on bug summarization. Similarly, ChatEDA reports comparable or superior outcomes relative to GPT-4 across its evaluated scenarios. Complementary work has explored the use of LLMs in conversational hardware design, in which a human engineer collaborates with an LLM to iteratively develop hardware through natural language interaction. Notably, one such effort reports the co-design and successful tapeout of an 8-bit microprocessor using GPT-4, which the authors describe as the first fully AI-written HDL design sent to fabrication [5].

More recently, retrieval-augmented generation (RAG) has emerged as an effective mechanism for improving the reliability and domain specificity of LLMs in EDA contexts. The EDA-Copilot framework [28] applies RAG to RTL code generation by integrating external knowledge retrieval into the LLM inference process, resulting in substantial performance improvements over earlier methods. By grounding generation in retrieved domain-relevant information, this approach enhances both query response accuracy and the functional correctness of generated RTL. These results underscore the importance of hybrid LLM architectures that combine generative capabilities with structured knowledge access to enable robust and trustworthy LLM-assisted EDA workflows.

Despite this, large language models remain limited in their ability to perform precise mathematical reasoning and numerical optimization, both of which are central to many EDA workflows [29]. Although state-of-the-art models exhibit improved reasoning capabilities, these limitations motivate integration strategies that avoid assigning LLMs direct responsibility for optimization tasks. Instead, LLMs are increasingly employed in agentic roles, where they act as high-level coordinators that interface with established optimization and design tools. Within this paradigm, the LLM interprets user intent, structures the optimization problem, and orchestrates tool execution, while numerical computation and optimization are delegated to external solvers such as Bayesian optimization.

The effectiveness of this approach in practical design settings is a field of ongoing research; for example, [30] presents an agentic, human-in-the-loop framework for end-to-end hardware design and verification, achieving over 95% coverage while reducing verification time. Similarly, AnaFlow [8] proposes a multi-agent workflow in which specialized LLM agents collaborate to analyze circuit topology and design objectives before optimizing circuit parameters to meet sizing requirements. This framework decomposes optimization into analysis, DC refinement, and heuristic- and optimization-based refinement stages, illustrating how LLMs can guide and structure optimization processes rather than directly performing them.

The agentic approach in ORFS [31] further validates this design philosophy by demonstrating an agentic LLM-driven optimization flow for chip design. The proposed open-source framework reports improved performance while requiring fewer design flow executions. Collectively, these works underscore the emerging role of LLMs as intelligent interfaces to optimization algorithms, motivating hybrid architectures that combine LLM-based reasoning with efficient external optimization modules, as adopted in this work.

```
Run an optimization when number of loops is 2,
number of initial samples are 10,
params_range:  [[0,1], [0,1], [0,1], [-7,-1]],
channel file is located at:
'file_path/channel.sp',
sample type is random,
pre-loading initial data is true,
path_data_name is 'file_path/initial_data.npz'
the file should be saved as
'file_path/save_file_'.
```

Fig. 5: An example user-prompt for Agent-Based approach

IV. Methodologies

We used Ansys Electronic Desktop (AEDT) to simulate a high-speed receiver at 50 Gb/s NRZ with four filter settings, which the LLM aimed to optimize. The filter settings included three decision feedback equalizer (DFE) taps and a gain control for a continuous time linear equalizer (CTLE). To assess the marked improvement in either LLM approach, we utilized a low-loss channel to ensure a valid eye opening; the presence of equalization further enhances the eye opening by tuning our objective to the eye height. We investigate two schemes, one relying on the LLMs' inherent reasoning capabilities to suggest values for design parameters, whereas in the second method, the LLM drives an external optimizer to optimize the channel eye height. Fig. 1 shows a high-level abstraction of both workflows, explained below.

A. Agent-Based

The agentic workflow is comprised of an LLM interfaced with a variety of external tools; in this configuration, we prompt the LLM with the design constraints as shown in Fig. 5. The LLM processes the prompt and extracts the necessary parameters from it. Then, the LLM selects the appropriate tool(s), in this case, a Bayesian optimizer, and organizes the information extracted from the prompt to drive the optimizer. The optimizer interfaces with AEDT to run simulation loops and save the results. Once the user-defined end of the loop has been reached, the LLM reads the results, picks the best eye height value, and sends it to the user. The detailed Agent-Based workflow is depicted in Fig. 2a.

B. Prompt-Based

In the Prompt-Based workflow, as shown in Fig. 2b, we first prompted the LLM with initial samples and currently available eye heights in natural language, asking it to generate a new set of receiver filter coefficients based on the priors. The LLM then drives the AEDT simulation based on these provided values, generates the eye diagrams, and feeds the eye height result back to the LLM, before asking it for the next set of parameters. This process is repeated for a pre-specified number of times, and the best obtained eye height is chosen as the optimized value. The LLM then sends the optimum eye height value, along with the newly optimized SerDes tap coefficient values, to the user. An example prompt for this technique is shown in Fig. 6.

```
We are trying to optimizing a SerDes
channel.  The channel can be optimized using
4 parameters:
1 CTLE filter whose value ranges from -1 to -7
and 3 DFE filters whose values range from 0
to 1.
The optimized channel will have a big eye
diagram height or eye opening.
Below are some previous samples and their
values:
<DFE1> 0.118574243 </DFE1>
<DFE2> 0.885983655 </DFE2>
<DFE3> 0.653941548 </DFE3>
<CTLE> -4.895275878 </CTLE>
eye height (result):  0
Can you please recommend a new set of DFE1,
DFE2, DFE3 and CTLE values that will result
in a bigger eye height?
Answer with only the values.  If you add other
text, I cannot check your result.
The values should be in the format:
<DFE1> value </DFE1>
<DFE2> value </DFE2>
<DFE3> value </DFE3>
<CTLE> value </CTLE>
```

Fig. 6: An example user-prompt for Prompt-Based approach

C. Benchmarking

We benchmarked our results across 3 open-source language models: Llama-3.1-8B-Instruct, Llama-3.2-3B-Instruct, and Llama-3.2-1B-Instruct, to comprehensively investigate the effects of increasing language model parameters on SerDes optimization tasks. We selected these models to leverage their ease of deployment and ability to run on relatively computationally light hardware. For each method, we collected three sets of initial samples and then ran the optimization process for 30 iterations, as the low-loss channel coupled with the small design space allows for a lower number of iterations.

V. Results

A. Design Space Exploration and Eye Height Optimization

As seen in Fig. 7, the Agent-Based approach outperforms the Prompt-Based approach in finding a bigger eye-opening, with

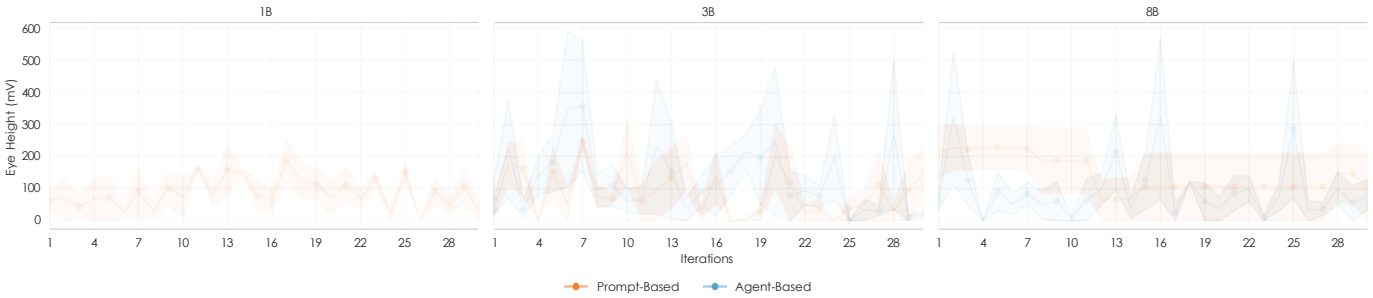


Fig. 7: The Agent-Based scheme explores more of the design space (see Fig. 4), causing higher variance in eye heights. The Prompt-Based scheme has lower variance in eye heights; a lower variance is not undesirable behavior if the resulting eye heights are higher, as this would indicate *exploitation* of a promising region. However, this is not the case for the Prompt-Based scheme, as the resulting eye heights are lower, as shown in the figure above.

the best eye heights being 832 mV and 364 mV across all models, for Agent-Based and Prompt-Based, respectively. We examine the distribution and spread of the filter coefficients explored by both approaches during user interactions to evaluate their effectiveness in design space exploration. Analysis of the predicted coefficients reveals inherent limitations of the Prompt-Based method. Fig. 3 shows the mean values of the filter coefficients generated by each approach across all iterations, while Fig. 4 uses UMAP to visualize the explored coefficient space in two dimensions as a function of eye height.

B. Adherence to Design Specifications

When the Agent-Based approach was used with the 1B model, it failed to utilize the user-given tools properly and ignored the instructions provided in the system prompt. As a result, the agent hallucinated and failed to execute the optimization task completely. Conversely, the 3B and 8B model-based Agents did not exhibit this behavior, consistently utilizing the appropriate tools every time.

It should be noted that certain PCIe design specifications have limits on the values of the gain; these limits were specified in the user-prompts as seen in Figs. 5 and 6. Providing these constraints is necessary to ensure that our SerDes remains compliant with the design specifications. When evaluating the Prompt-Based approach with the 1B model, the model ignored these given design specifications and suggested values outside the normal mode of operation for CTLE over 50% of the time. However, the 3B and 8B models were successful in adhering to the instructions provided by the user and executing the optimization tasks without violating the design specifications.

VI. Discussion

We see from Fig. 4 that the Prompt-Based approach is reluctant to vary all four weights simultaneously. This behavior is more prominent in the 8B model, where almost all the tap coefficients tend to be the same across different simulation runs, indicating more exploitation of the design space. This may be attributed to the LLM's inherent weakness in effectively comprehending complex mathematical problems [29]. Crucially, this may indicate the Prompt-Based approach's inability to scale up with higher dimensional problems. In contrast, the agentic flow leverages the external optimizer to explore the design space efficiently, indicating a clear architectural advantage.

Further, it is evident from Fig. 4 that Prompt-Based tends to reuse samples, which is reflected through the numerous clusters of orange samples. This behavior is further quantified in Table I. While having a lower variation may not necessarily be a bad thing, in the case of the Prompt-Based approach, it points towards a failure to capitalize on exploitation, resulting in a lower likelihood of reaching an optimum eye height.

These results showcase that agentic LLM workflows could be well-suited for EDA tasks involving time-intensive simulations and high-dimensional parameter spaces, such as analog circuit sizing. Here, we could potentially translate high-level performance targets such as gain, bandwidth, and power constraints into structured optimization problems, and delegate transistor-level parameter tuning to an external optimizer driven by circuit simulations. On the other hand, Prompt-Based approaches may exhibit promise in rapid prototyping, coarse design-space pruning, and other scenarios where approximate solutions are sufficient.

Overall, our work shows that treating LLMs as tool-orchestrating agents, rather than standalone optimizers, enables more effective and reliable exploration of complex design spaces. The Agent-Based approach is both tool-agnostic and decision-capable, supporting its extension to a wide range of EDA workflows beyond the specific SerDes application considered.

VII. Conclusion

In summary, our results motivate future research in balancing exploration and exploitation of design spaces via LLM-guided frameworks. We highlight how language model interfaces

TABLE I: Absolute Deviation of Design Parameters

	Llama-3.2-1B		Llama-3.2-3B		Llama-3.1-8B	
	Agent-Based	Prompt-Based	Agent-Based	Prompt-Based	Agent-Based	Prompt-Based
DFE 1	Failed	0.1044	0.3192	0.1530	0.2785	0.0073
DFE 2	Failed	0.1511	0.3318	0.1961	0.2982	0.0408
DFE 3	Failed	0.1980	0.3035	0.1265	0.2941	0.0514
CTLE	Failed	2.4562	2.0295	1.3859	2.1016	0.1630

The Agent-Based approach has a higher absolute deviation across all iteration runs compared to its Prompt-Based counterpart. A higher deviation corresponds to a more effective exploration of the design space and thus has a higher likelihood of reaching an optimized eye height.

can complement traditional optimization workflows and prove that domain-specific algorithms interfaced with LLMs are a promising approach for high-dimensional optimization in EDA workflows. In particular, Agent-Based integration of LLMs with domain-specific algorithms enables more effective and consistent exploration of high-dimensional SerDes design spaces compared to Prompt-Based approaches, indicating that hybrid architectures combining LLM reasoning with established optimization methods can represent a promising direction for scalable and robust design automation in EDA.

Acknowledgment

This research is supported in part by the NSF under Grant No. CNS 2137283 (Center for Advanced Electronics through Machine Learning) and the industry members of the CAEML IUCRC.

References

[1] M. A. Dolatsara, J. A. Hejase, W. D. Becker, J. Kim, S. K. Lim, and M. Swaminathan, "Worst-case eye analysis of high-speed channels based on bayesian optimization," *IEEE Transactions on Electromagnetic Compatibility*, vol. 63, no. 1, pp. 246–258, 2021.

[2] X. Wei, "The application of the optimized genetic algorithm in SerDes circuit design," *Microelectronics Journal*, vol. 156, p. 106509, 2025.

[3] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozie`re, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "Llama: Open and efficient foundation language models," 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

[4] Z. He, H. Wu, X. Zhang, X. Yao, S. Zheng, H. Zheng, and B. Yu, "Chateda: A large language model powered autonomous agent for eda," 2024.

[5] J. Blocklove, S. Garg, R. Karri, and H. Pearce, "Chip-chat: Challenges and opportunities in conversational hardware design," in *2023 ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD)*. IEEE, 2023, pp. 1–6.

[6] M. Liu, T.-D. Ene, R. Kirby, C. Cheng, N. Pinckney, R. Liang, J. Alben, H. Anand, S. Banerjee, I. Bayraktaroglu, B. Bhaskaran, B. Catanzaro, A. Chaudhuri, S. Clay, B. Dally, L. Dang, P. Deshpande, S. Dhodhi, S. Halepete, E. Hill, J. Hu, S. Jain, A. Jindal, B. Khailany, G. Kokai, K. Kunal, X. Li, C. Lind, H. Liu, S. Oberman, S. Omar, S. Pratty, J. Raiman, A. Sarkar, Z. Shao, H. Sun, P. P. Suthar, V. Tej, W. Turner, K. Xu, and H. Ren, "Chipnemo: Domain-adapted llms for chip design," 2024.

[7] C.-C. Chang, Y. Shen, S. Fan, J. Li, S. Zhang, N. Cao, Y. Chen, and X. Zhang, "Lamagic: language-model-based topology generation for analog integrated circuits," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24. JMLR.org, 2024.

[8] M. Ahmadzadeh, K. Chen, and G. Gielen, "Anaflow: Agentic llm-based workflow for reasoning-driven explainable and sample-efficient analog circuit sizing," 2025. [Online]. Available: https://arxiv.org/abs/2511.03697

[9] G. Kokolakis, A. Moschos, and A. D. Keromytis, "Harnessing the power of general-purpose llms in hardware trojan design."

[10] M. Li, W. Fang, Q. Zhang, and Z. Xie, "Specllm: Exploring generation and review of vlsi design specification with large language model," 2024.

[11] M. Liu, N. Pinckney, B. Khailany, and H. Ren, "Invited paper: Verilogeval: Evaluating large language models for verilog code generation," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023, pp. 1–8.

[12] E. Latif, L. Fang, P. Ma, and X. Zhai, "Knowledge distillation of llm for automatic scoring of science education assessments," 2024.

[13] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[14] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[15] X. Xu, M. Li, C. Tao, T. Shen, R. Cheng, J. Li, C. Xu, D. Tao, and T. Zhou, "A survey on knowledge distillation of large language models," 2024.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[17] A. Singh, A. Fry, A. Perelman, A. Tart, A. Ganesh, A. El-Kishky, A. McLaughlin, A. Low, A. Ostrow, A. Ananthram *et al.*, "Openai gpt-5 system card," *arXiv preprint arXiv:2601.03267*, 2025.

[18] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, "A survey of large language models," 2025. [Online]. Available: https://arxiv.org/abs/2303.18223

[19] K. L. Aw, S. Montariol, B. AlKhamissi, M. Schrimpf, and A. Bosselut, "Instruction-tuning aligns llms to the human brain," 2024. [Online]. Available: https://arxiv.org/abs/2312.00575

[20] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[21] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2022.

[22] T. Masterman, S. Besen, M. Sawtell, and A. Chao, "The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2404.11584

[23] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," 2023. [Online]. Available: https://arxiv.org/abs/2210.03629

[24] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[25] H. Wang, T. Zariphopoulou, and X. Zhou, "Exploration versus exploitation in reinforcement learning: a stochastic control approach," 2019. [Online]. Available: https://arxiv.org/abs/1812.01552

[26] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[27] Y. Wen, J. Dean, B. A. Floyd, and P. D. Franzon, "High dimensional optimization for electronic design," in *2022 ACM/IEEE 4th Workshop on Machine Learning for CAD (MLCAD)*, 2022, pp. 153–157.

[28] Z. Xiao, X. He, H. Wu, B. Yu, and Y. Guo, "Eda-copilot: A rag-powered intelligent assistant for eda tools," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 30, no. 6, Oct. 2025. [Online]. Available: https://doi.org/10.1145/3715326

[29] J. Boye and B. Moell, "Large language models and mathematical reasoning failures," 2025. [Online]. Available: https://arxiv.org/abs/2502.11574

[30] D. N. Gadde, K. K. Radhakrishna, V. N. Viswambharan, A. Kumar, D. Lettnin, W. Kunz, and S. Simon, "Hey ai, generate me a hardware code! agentic ai-based hardware design amp; verification," in *2025 38th SBC/SBMicro/IEEE Symposium on Integrated Circuits and Systems Design (SBCCI)*. IEEE, Aug. 2025, p. 1–5. [Online]. Available: http://dx.doi.org/10.1109/SBCCI66862.2025.11218681

[31] A. Ghose, A. B. Kahng, S. Kundu, and Z. Wang, "Orfs-agent: Tool-using agents for chip design optimization," 2025. [Online]. Available: https://arxiv.org/abs/2506.08332