Exploration of an agent-driven LLM fully automated simulation circuit netlist generation architecture

Awinic Technology Co., Ltd.Shanghai, China  
wangwei.c@awinic.com

*Abstract*—The automation of analog circuit design continues to face numerous challenges due to its heavy reliance on expert experience and iterative SPICE simulations. This paper presents an agent-driven architecture that utilizes large language models (LLMs) to achieve fully automated circuit netlist generation. Our framework integrates a domain-enhanced LLM engine with a knowledge base containing topological templates and parameter optimizations based on simulation results and target discrepancies for each device, forming a closed-loop workflow that encompasses requirement parsing, netlist generation, NGSPICE-based simulation verification, and multi-objective optimization. Testing reveals that the proposed method achieves 78-82% success rate in generating specification-compliant operational amplifiers within ≤10 iterations (n=50 test cases).. The architecture's unique error self-correction mechanism effectively addresses a significant number of functional mismatches through automated parameter deviation correction. This work not only advances the application of LLMs in EDA but also provides a possible foundational framework for the Grand Challenge's goal of achieving circuit design automation.

Keywords—Analog circuit design, Design automation, LLM, Parameter optimization

# Introduction

Research Background and Motivation. The automation of analog circuit design has long been a critical area of research in the field of Electronic Design Automation (EDA). As electronic systems become increasingly complex, the demand for efficient, reliable, and scalable design methodologies grows. Traditional approaches to analog circuit design rely heavily on expert knowledge and manual intervention, which not only increases the time-to-market but also poses significant challenges in maintaining consistency and quality across designs. These challenges are particularly pronounced in high-performance applications such as operational amplifiers, where stringent performance metrics must be met.

Challenges in Analog Circuit Design Automation. Analog circuit design is inherently complex due to the intricate relationships between various electrical parameters and the need for precise tuning to meet specific performance criteria. The traditional design process involves iterative simulations and adjustments, guided by the expertise of experienced engineers. This approach, while effective, suffers from several limitations:

High Expertise Threshold: Significant domain knowledge is required to navigate the complexities of analog circuit design.

Time-Consuming: Manual iterations and fine-tuning can be labor-intensive and time-consuming.

Scalability Issues: Scaling up the design process to handle multiple constraints and specifications efficiently remains challenging.

Emergence of Large Language Models in EDA. Recent advancements in artificial intelligence, particularly the development of large language models (LLMs), have shown promise in addressing these challenges. LLMs possess the capability to understand and generate human-like text, making them suitable for tasks that require complex reasoning and decision-making. In the context of EDA, LLMs can be leveraged to automate various stages of the design process, from initial netlist generation to final optimization. By integrating LLMs into the design workflow, it becomes possible to reduce the dependency on manual interventions and enhance the efficiency of the overall design process.

Proposed Approach: Integrating LLM with Intelligent Agents. In this work, we propose a novel architecture that integrates LLMs with intelligent agents to achieve end-to-end automated generation of netlists directly from design requirements. Our approach leverages the strengths of LLMs in understanding and processing natural language descriptions of design specifications, combined with the computational power of intelligent agents to optimize and refine the generated netlists. This fusion allows for a seamless transition from requirement definition to netlist generation, significantly reducing the time and effort required for analog circuit design.

# designing scheme

## Final Round Circuit Design

According to the requirements of the final round, we replaced the ideal current source with a degeneration resistor while keeping the overall circuit structure unchanged, as shown in the figure below:



Fig.1. Circuit Structure

The depicted circuit represents a differential amplifier, a fundamental building block in operational amplifiers (op-amps).

The input stage comprises a differential pair (M1, M2) driven by differential signalsand enabling common-mode noise rejection. A current mirror (M3, M4) biases the input pair, establishing stable operating conditions. The output stage employs an active load (M6) to convert the differential signal into a single-ended output (( )) via current-to-voltage conversion, enhanced by a compensation capacitor (( )) for phase margin stabilization and bandwidth optimization. Load capacitance () governs settling time and frequency response.A current mirror network (M5, M7, M8) ensures proportional current replication from the input to output stages, maintaining signal fidelity. Key features include high gain (via differential amplification), low noise (symmetrical topology), and robust linearity (active load design).This architecture is widely utilized in high-speed op-amps and precision analog systems requiring stability, noise suppression, and wide bandwidth.

## Automated Netlist Generation Framework

Architecture Overview Our framework employs a three-agent architecture comprising initialization, optimization, and annotation agents. The system implements an iterative refinement process with maximum 50 optimization cycles, terminating when design specifications are met or reaching iteration limits. This hierarchical structure combines design expertise with automated simulation verification.

## Core Agent Design

Initialization Agent. The initialization agent integrates domain-specific knowledge including:

* Design guidelines data.
* Circuit topology.
* Design rules.

Its workflow contains:

1. Requirement parsing through natural language processing.
2. Parameter initialization via constraint programming.
3. SPICE-compatible netlist generation.
4. Automated simulation invocation.

Optimization Agent. The optimization agent implements:

* Determine which component parameters to optimize next based on the simulation results.
* Design tailored optimization strategies for different components.
* Simulation feedback analyzer.
* Maintain continuous iterative cycles until simulation results fully meet all specifications.

Key features include:

1. Leverage large language models (LLMs) to analyze subsequent optimization targets.
2. Refine optimization schemes through component-specific optimizations addressing distinct performance metric deficiencies, while mitigating LLM erroneous reasoning probability.
3. Automated simulation invocation.

Annotation Agent. The annotation subsystem enhances readability through:

* Automatic comment generation.
* Parameter significance tagging.

## Collaborative Workflow Design

The proposed framework executes the following steps to achieve automated netlist generation and optimization:

* Initial Netlist Generation (T0):

The system generates an initial netlist based on the provided design requirements using a large language model (LLM) integrated with an intelligent agent.

* Iterative Optimization Loop:

While (iteration < 50 && specifications not met):

Simulation Result Analysis: Analyze current netlist simulation results to identify performance gaps against target specifications, while incorporating supplementary operational parameters (e.g., component operating regions, Iref current) into simulations to facilitate LLM-enhanced optimization scheme formulation.

Optimized Subdivision: Instruct LLMs to perform performance-prioritized decision-making for identifying critical optimization metrics, followed by component parameter adjustments required for target metric enhancement.

Netlist Modification: Direct LLMs to perform targeted component optimization based on simulation results, thereby enhancing optimization accuracy.

Verification Checkpoint: Verify whether the updated netlist now meets the specified performance criteria. If not, proceed to the next iteration.

* Final Annotation Injection:

Once the netlist meets all the required specifications or the maximum number of iterations is reached, inject final annotations to document the design process and ensure traceability.



Fig.2. Core design iteration diagram.

## Initial Netlist Initialization Scheme

Based on the aforementioned circuit structure, we analyzed and identified 35 variables, including 8 MOS transistors, each with 4 variables: W, L, nf, and m, in addition to bias resistors, compensation resistors, and compensation capacitors, totaling 35 variables.

To reduce the difficulty of LLM reasoning, we imposed certain constraints, such as W1=W2, L1=L2, m1=m2; W3=W4=W6, L3=L4=L6, m3=m4; W5=W7=W8, L5=L7=L8; and m6:m4=2\*(m7:m5), among others. Since nf has minimal impact on the design, it was fixed to 1 during initialization.

As a result, the number of variables requiring LLM reasoning was significantly reduced. Subsequently, we informed the LLM that MOS transistors 1 and 2 serve as the first-stage amplifying transistors. To achieve better gain, the ratio of W to L should be as large as possible. For transistors 3, 4, and 6, the ratio should be moderate, while for transistor 8, the ratio should be relatively small. This enables the LLM to generate a relatively accurate initial netlist.

## Core Design Criteria

The core methodology involves iterative performance metric optimization following each netlist simulation cycle, with optimization prioritization ordered as Gain > Slew Rate (SR) > Phase Margin (PM) > Gain-Bandwidth Product (GBW). This process accommodates potential iterative regressions, such as PM compliance potentially causing SR non-compliance after previous optimization. The entire workflow is orchestrated by LLMs, with human engineers providing prompt-based guidance to establish component-specific optimization rules and strategies for each underperforming metric.

## Core Optimization Experience

We have designed a set of optimization logic and instructed the large model to sequentially check each optimization scheme. Once a matching optimization scheme is found, it outputs the result and stops further thinking.

1. Regardless of the situation, the priority is to let the large model adjust the Gain to meet the target. Based on a netlist with moderate relative indicators initialized, we only need to adjust Rref to achieve gain regulation.

2. After the gain meets the target, adjust SR to meet the target. This is relatively complex and may require adjustments to four components: XM4, XM5, XM6, and XM7. If the L and W parameters of XM5 and XM7 are small, prioritize increasing them proportionally. Otherwise, adjust the m values of XM6 and XM7, as proportionally amplifying them can also achieve SR improvement.

3. After SR meets the target, we adjust PM. Generally, only R1 and C1 need to be adjusted at this stage. In most cases, gradually increasing C1 helps improve PM.

Of course, there may be instances where SR no longer meets the target after PM is achieved. In such cases, it is necessary to revert to step 2 for further optimization. By following this iterative process, we can eventually obtain a netlist that meets the requirements.

# Experimental Results

Research indicates that this optimization scheme surpasses the one we designed during the preliminary competition, boasting higher stability and faster convergence rates. Moreover, it possesses the capability for independent optimization of individual metrics, making it more suited to the requirements of the finals. Under the finals' scheme, the success rate of optimization reached 100% within just 20 iterations, with no further instances of unstable optimization occurring. These insights provide valuable guidance for future algorithmic improvements.

# CONCLUSION

From the perspective of industrial chip design workflows, we posit that integrating corporate IP libraries and design experience repositories into the AI-driven design process can significantly enhance design efficiency for a majority of requirements. In practical production design scenarios, relying solely on an AI agent's autonomous capability to generate designs from scratch often faces challenges, especially in complex circuit design tasks where numerous constraints and specifications must be met.

By enabling the intelligent agent to access and leverage these knowledge bases, it can retrieve initial solutions that are closer to the desired specifications. This approach leverages existing expertise and proven designs, providing a more robust starting point for subsequent optimization processes. For instance, when designing operational amplifiers with stringent performance metrics such as those outlined in our study (Gain > 60 dB, GBW > 20 MHz, PM > 60°, SR > 20 V/μs, and Idc < 500 μA), initializing the design process with a netlist derived from a well-curated IP library can drastically reduce the number of iterations needed to achieve optimal results.

This hybrid methodology not only accelerates the design cycle but also improves the reliability of the final product by grounding the AI's efforts in established best practices. Consequently, while fully autonomous design generation by AI remains a challenging task in complex circuit design, augmenting the AI's capabilities with comprehensive knowledge bases offers a promising pathway to achieving both efficiency and effectiveness in industrial settings.

##### Acknowledgment

The authors would like to express their sincere gratitude to the organizers of the competition for providing the platform and resources that facilitated this research. Special thanks go to AICAS (AI Chip Advanced Solutions), T-HEAD, and Tianchi for their support and encouragement throughout the project.