AICAS Grand Challenge Track 3: LLM-Based Analog Operational Amplifier Circuit Design

Team Name: **iCASLLM4EDA**
Team Members: Xinyue Wu, Fan Hu, Jani Babu Shaik, Zijiao Zhang and Linyu Zhu
Advisor: Xinfei Guo
Shanghai Jiao Tong University, Shanghai, China

***Abstract*—We have developed a design methodology for analog operational amplifiers (OP-AMPs) with Qwen 2.5-7B series on the given 16-core CPU Yitian server. This approach, grounded in theoretical analysis, simulation, and iterative refinement, enables the LLM to directly generate NGSPICE netlists that fulfill all specified requirements.**

***Index Terms*—operation amplifier, large language model, agent, netlist**

I. Analysis of the Difficulties in the Semi-final

Compared with the preliminary round, the semi-final has been added some requirements including deploying the model locally on a 16-core CPU Linux AARCH64 server (without GPU), and arbitrarily determining one metric in five to let the agent achieved the standard. These requirements has mainly induced to three difficulties:

- The Python Ollama SDK's incompatibility with the system can fully occupy server threads, causing lag and program failure. Using Ollama directly for inference solves this but makes the Qwen Agent framework [1] unusable.
- The experiment indicates that models in resource-limited environments perform slightly worse than those on the cloud or with abundant resources, causing output instability.
- Any metric tests the flexibility of the agent. However, it is not easy to fine-tune the model so that it "truly understands" the meaning of each metric, given the limited model size and hardware conditions.

Thus in the semi-final, we newly enable the LLM to call other tools (e.g., Python programming) without Qwen Agent structure by ourselves, reconduct prompt engineering, and design a knowledge-based arbitrary metric requirement agent to improve our agent.

II. Agent Structure

The architecture of our proposed agent is illustrated in Figure 1. This framework integrates a Theoretical Calculation Agent, an Iteration Agent, a Netlist Generation Agent and an Arbitrary Metric Requirement Agent. Through this integration, the system is capable of producing a netlist in the correct format that fulfills all specified evaluation criteria.

Given the distinct proficiencies of the Qwen series LLMs in various domains, we have opted for two suitable 7B models

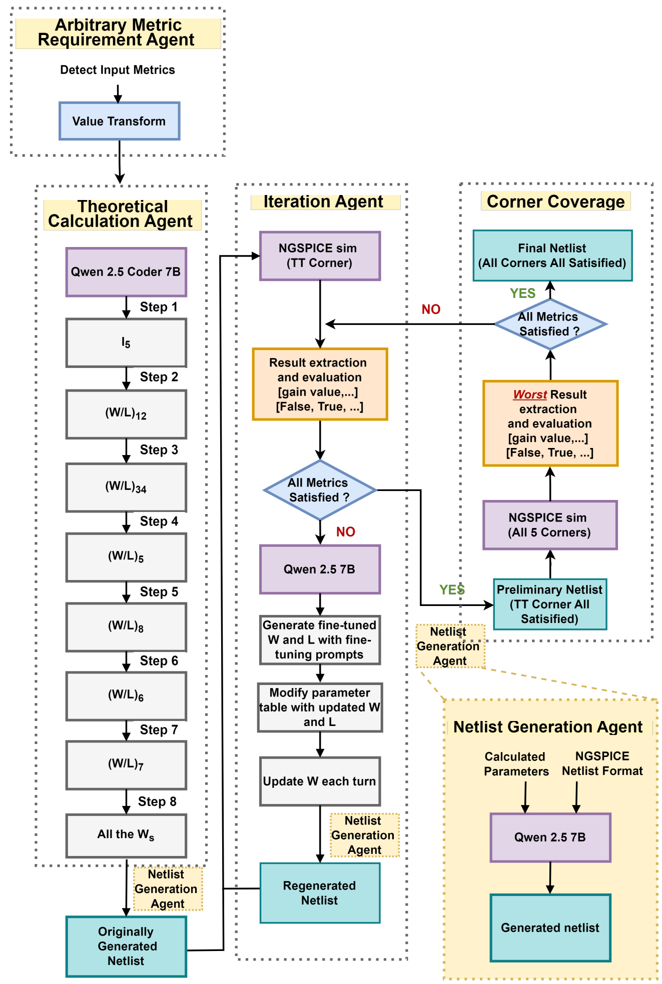

Fig. 1. Our proposed LLM-assisted OP-AMP design flow.

from this series for our approach, one designated for high-precision analytical computations and the other for netlist generation.

- **Qwen2.5-Coder-7B** [2] is the latest version of the code model in the Qwen series, inheriting the excellent performance of Qwen2-Coder and further enhancing its capabilities. This version not only performs well in multiple programming languages but also shows significant improvements in complex task handling and code logic understanding.

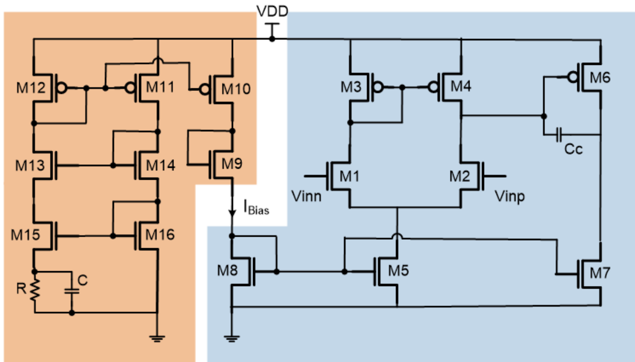

Fig. 2. Circuit diagram of Two-Stage OP-AMP with current bias generator design

- **Qwen2.5-7B** [3] is an advanced language model based on the Transformer architecture. Its innovations include technologies like RoPE, SwiGLU, and RMSNorm. It supports multilingual tasks, excels in long-text generation and structured data processing, offering outstanding performance.

A. Theoretical Calculation Agent

Figure 2 illustrates the two-stage operational amplifier (op-amp) circuit with current bias generator circuit highlighted in red-block. The reference bias current (I_{Bias}) is generated by the familiar bootstrapped current [6]. This op-amp circuit was developed using the square-law approach as referenced in [4]. The agent calculate all the W/L ratios through theoretical analysis steps below.

STEP-1: Calculation of the compensation capacitor (C_C) and the tail current through the M5 transistor (I_{D5})

1) For a phase margin of 60^0, the compensation capacitor (C_C) should be greater than 0.22 times the load capacitor (C_L), i.e., $C_C > 0.22C_L$.
2) The tail current through M5 transistor is given by $I_{D5} = SR * C_C$. Where SR is slewrate given the table III

STEP-2: Determine the (W/L) ratio for input transistors (M1 and M2).

The (W/L) ratio of input transistors is determined by following equation 1

$$(W/L)_{1,2} = \frac{g_{m1,2}^2}{K_n I_{D5}} \tag{1}$$

Where $g_{m1,2} = 2\pi * GBW * C_C$.

STEP-3: Determine the (W/L) ratio for mirror load (M3 and M4)

The (W/L) ratio for the MOSFETs of mirror load current (M3 and M4) are determined by the following equation 2

$$(W/L)_{3,4} = \frac{I_{D5}}{K_p(V_{DD} - V_{inMAX} - V_{THp} + V_{THn})^2} \tag{2}$$

STEP-4: Determine the (W/L) ratio for tail current source transistors (M5 and M8), M5 firstly calculated

The (W/L) ratio for the tail current transistor M5 is calculated by the following equation 3

$$(W/L)_5 = \frac{2I_{D5}}{K_n\left(V_{inMIN} - V_{SS} - \sqrt{\frac{I_{D5}}{K_n*(W/L)_{1,2}}} + V_{THn}\right)^2} \tag{3}$$

STEP-5: Determine the (W/L) ratio M8 According to M5

Since tail current is mirror of reference current (I_{ref} flowing through transistor M8. We can determine (W/L) ratio of M8 using current mirror property:

$$(W/L)_8 = (\frac{I_{ref}}{I_{D5}}) * (W/L)_5 \tag{4}$$

STEP-6: Determine the (W/L) ratio for second stage input transistor M6

We can determine the (W/L) ratio of M6 transistor and is given by equation 5.

$$(W/L)_6 = \left(\frac{g_{m6}}{g_{m4}}\right)(W/L)_{3,4} \tag{5}$$

Where transcondutance values for M6 needs to be 10 times greater than $g_{m1,2}$, i.e., $g_{m6} > 10g_{m1,2}$.

STEP-7: Determine the (W/L) ratio for tail current source transistor (M7) in the second stage

The (W/L) ratio of M7 can be determined with the current mirror properties and is given by equation 6

$$(W/L)_7 = \left(\frac{I_{D7}}{I_{D5}}\right)(W/L)_5 \tag{6}$$

STEP-8: Calculate all the Ws

Since in some occasion, a single W_{eff} should be divided into $multi$ of parallel MOSFETs with each W that

$$W_{eff} = W \times multi \tag{7}$$

we have to let the LLM calculate all the Ws, then put the params into Netlist Generation Agent to get an original netlist.

By running the Python scripts generated by Qwen, the agent can now theoretically design the two-stage operational amplifier circuit utilizing the measured (W/L) ratio of each MOSFET. Nevertheless, a trade-off must be optimized among gain, area, and power consumption. Thus, an Iteration Agent is introduced to finetune the params combining with NG-SPICE simulation results base on SkyWater130 PDK [5].

B. Iteration Agent

The iteration agent implements a **closed-loop optimization framework** (Algorithm 1) that systematically bridges theoretical OP-AMP designs with practical circuit implementation. To ensure robust performance across all five process corners (SS, SF, TT, FS and FF), the design iteration is performed on the TT corner while applying 5% guardbands to all specifications. For metrics with minimum requirements like Gain, GBW, PM, SR, CMRR and PSRR, we increase the targets by 5%, while for metrics with maximum limits including Idc and noise,

TABLE I

DEPENDENCE OF THE PERFORMANCE ON DRAIN CURRENT I_5 AND I_7, W/L RATIOS, AND THE COMPENSATING CAPACITOR C_c

Performance	**Drain Current**		**M1 and M2**		**M3 and M4**		**Inverter**	**Inverter Load**		**Compensation Capacitor**
	I_{D5}	I_{D7}	W/L	L	W	L	W_6/L_6	W_7	L_7	C_c
Increase Gain	$(\downarrow)^{1/2}$	$(\downarrow)^{1/2}$	$(\uparrow)^{1/2}$	$\uparrow$		$\uparrow$	$(\uparrow)^{1/2}$		$\uparrow$	
Increase GBW	$(\uparrow)^{1/2}$		$(\uparrow)^{1/2}$							$\downarrow$
Increase RHP Zero		$(\uparrow)^{1/2}$					$(\uparrow)^{1/2}$			$\downarrow$
Increase Slew Rate	$\uparrow$									$\downarrow$

Algorithm 1 AMP Netlist Fine-Tuning Iteration Algorithm

Require: Original LLM-generated OP-AMP netlist, fine-tuning prompts, expected specifications
Ensure: Fine-tuned netlist with all evaluation metrics optimized

```
Initialize maximum iterations max_iter ← 50
Initialize current netlist netlist ← original netlist
Initialize iteration counter iter ← 0
while iter < max_iter do
    metrics ← NgspiceSim(netlist) {Returns 5 evaluation metrics}
    is_valid, unsatisfied ← check_specifications(metrics)
    if is_valid then
        return netlist {All expected specifications satisfied}
    end if
    for all metric ∈ unsatisfied do
        tuning_strategy ← TuningStrategySelect(metric)
        netlist ← WL_Adjustment(netlist, tuning_strategy)
    end for
    iter ← iter + 1
end while
return netlist {Return optimized result}
```

we reduce the targets by 5%. This margin allocation strategy effectively accounts for potential corner variations while maintaining sufficient design flexibility during optimization.

Beginning with an LLM-generated netlist, this framework first executes automated Ngspice simulations to extract five key performance metrics (*e.g., gain, bandwidth, power consumption*). When inconsistances between simulated results and target specifications are detected, the agent initiates metric-driven W/L adjustments: 1) It maps each unsatisfied specification to predefined parameter tuning strategies via TUNINGSTRATEGYSELECT, which is given in iteration prompts and follows the dependence in Table I [4]. 2) Modifies transistor geometries through WL_ADJUSTMENT while preserving the core circuit topology. This process repeats for up to 50 iterations, employing gradient-free progressive refinement where each cycle's simulation results directly inform subsequent parameter adjustments. The self-correcting mechanism ensures continuous netlist optimization until all specifications are simultaneously met, effectively resolving fabrication-aware performance gaps that pure theoretical designs often exhibit.

C. Netlist Generation Agent

Given that the text comprehension capabilities of Qwen Math 2.5 are relatively limited, an integration strategy has been adopted to directly facilitate the generation of netlists incorporating calculated parameters and adhering to the correct format specifications. In this process, an agent is tasked with the transmission of a preliminary netlist document wherein variable placeholders, exemplified by notations such as $(W)_{12}$ and $(multi)_{12}$, are strategically positioned to indicate where specific numerical values must subsequently be inserted. This procedural step ensures that all necessary parameter substitutions are accurately executed, thereby ensuring the resultant netlist's functional integrity.

Following the completion of this substitution phase, the generated response from previous steps is meticulously preserved in a file designated as $AMP.cir$. The corresponding prompts utilized for orchestrating this sophisticated interaction are codified within the script located at $prompt/generate\_netlist.py$, which serves as the blueprint guiding this automated netlist generation procedure. Through this method, the inherent limitations in textual understanding are effectively circumvented, allowing for the precise formulation of netlists tailored to specified requirements.

D. Arbitrary Metric Requirement Agent

To verify the flexibility of our algorithm across diverse metrics, we have incorporated a dedicated agent module targeting metric requirements into the basic algorithm. This architectural enhancement endows our methodology with enhanced adaptability to accommodate expanded metric spaces as shown in Figure 3.

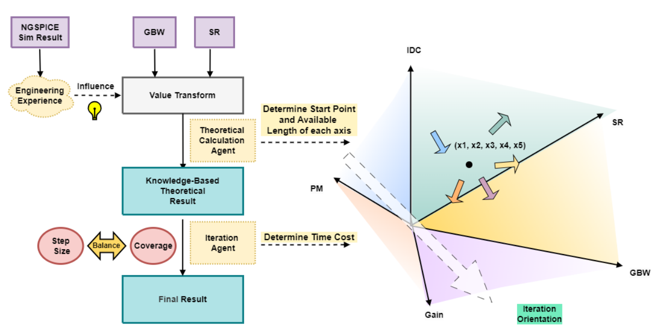

Fig. 3. The principle of adjusting arbitrary one in five indicators.

GBW and SR are two metrics critical for theoretical parameter calculations, and directly determined the theoretical calculation. Simulations show that for large requirements in these two metrics, some theoretical knowledge is in distortion, thus the agent first performs a slightly value transformation according to engineering experience. The theoretical result determines the start point and the length of each available range of metrics of the solution space, then iteration agent adjusts the iteration step size to make sure the process not too long but also can cover all the possible values in the workable ranges. Mostly, the iteration orientation is towards decreasing I_{DC} while increasing the other four metrics.

III. Results and Discussions

This section presents the simulated values of the two-stage operational amplifier, as seen in Figure 2. Initial theoretical calculations (as seen in table II) utilizing the qwen LLM demonstrate that the simulated parameters ($Gain = 51dB, GBW = 44.4M\ Hz, PM = 53^0, SR = 52.9\ V/\mu s,\ and\ I_{DC} = 300\ \mu A$) closely align with the target specifications. The iteration agent was developed as previously explained, to optimize the dimensions of MOSFETs to achieve the values that were specified. In the iteration steps, we focus on the gain, as the lengths of the M1, M2 and M3, M4 MOSFETs influence the gain according to Table I. Figure 4 demonstrates that our theory-driven iteration achieves a PM-Gain trade-off, with PM slightly decreasing while all four other metrics improve across iterations. Through iterative agent fine-tuning, we successfully meet the required values. Metrics before and after iteration are compared in Table II.

TABLE II
List of (W/L) ratio of each MOSFET derived using theoretical calculation from qwen LLM and optimized values after iteration agent

Parameters	Theoretical Calculation	After Iteration
I_{D5}	25.0	25.0
$(W/L)_{1,2}$	25.97/1.0	25.97/1.2
$(W/L)_{3,4}$	18.45/1.0	18.45/1.2
$(W/L)_5$	8.44/0.5	8.89/0.5
$(W/L)_6$	60.5/0.5	60.5/0.5
$(W/L)_7$	88.76/0.8	88.76/0.8
$(W/L)_8$	6.75/0.5	6.75/0.5

Table III summarizes the specified and post-optimization simulated values for five OP-AMP specifications. The results show that Gain and GBW exceed specifications, while SR and PM closely match the targets. This demonstrates that our qwen-LLM-driven methodology offers a streamlined approach to operational amplifier design.

For the arbitrary five-metric requirement, empirical validation confirms that the netlist generated by this agent achieves performance compliance within the tolerance ranges shown in Table IV for any arbitrarily selected metric among the five key parameters.

IV. Conclusion

The current work introduces a method for designing a standard two-stage operational amplifier with locally deployed Alibaba Qwen LLM series under hardware resource limited occasions. We initially created an agent to generate the (W/L) ratio for each MOSFET in the op-amp. An iteration agent was later developed to optimize the circuit in accordance with provided target values. Also, an Arbitrary Metric Requirement Agent was designed to perform slight value transformation according to human experiences, and was integrated to the whole flow to achieve the “arbitrarily determining one metric in five” requirement.

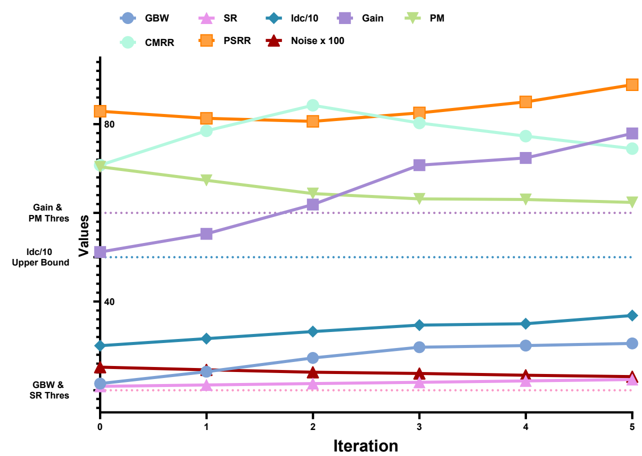

Fig. 4. Evolution of eight evaluation metrics across iteration steps.

TABLE III
List of required Op-Amp specifications and corresponding values. Also presents the Op-Amp simulated values

Specifications	Required Values	Tested Values
Tokens	-	372
Runtime	-	359.63 s
Gain Bandwidth (GBW)	$> 20M\ Hz$	$24.58M\ Hz$
DC gain (Av in dB)	$> 60\ dB$	$72.47\ dB$
Slew Rate (SR)	$> 20\ V/\mu s$	$17.89\ V/\mu s$
Phase Margin (PM)	$> 60^0$	60.66^0
DC current (I_{DC})	$< 500\ \mu A$	$435.11\ \mu A$
CMRR	-	74.48
PSRR	-	88.88
noise	-	$0.23/KHz$

TABLE IV
Available ranges of the five metrics

Metric	Unit	Available Range
DC Gain	dB	60-85
Gain Band Width (GBW)	MHz	20-40
Phase Margin (PM)	$degree$	60-90
Slew Rate (SR)	$V/\mu S$	20-85
DC Current (I_{DC})	μA	340-405

References

[1] QwenLM, “Qwen-Agent”, https://github.com/QwenLM/Qwen-Agent, Dec 2024

[2] A. Yang et al., “Qwen2. 5-Coder technical report,” arXiv preprint arXiv:2409.12186, 2024.

[3] A. Yang et al., “Qwen2. 5 technical report,” arXiv preprint arXiv:2412.15115, 2024.

[4] Phillip E. Allen, and Douglas R. Holberg “CMOS Analog Circuit Design” 3rd edition, Oxford University Press, Inc, 2012.

[5] SkyWater, “SkyWater SKY130 PDK”, https://skywater-pdk.readthedocs.io/, 2023

[6] T. Delbruck and A. van Schaik, ”Bias current generators with wide dynamic range, “ 2004 IEEE ISCAS, Vancouver, Canada, 2004, doi: 10.1109/ISCAS.2004.1328200.