RGB-ACD: RGB-INSPIRED TOPOLOGY GENERATION FOR LLM-GUIDED ANALOG CIRCUIT DESIGN

ABSTRACT

Automating analog circuit design requires searching discrete topologies and continuous device parameters under coupled performance constraints. Translating natural-language requirements into circuit structures that satisfy these constraints remains challenging. We introduce RGB-inspired Analog Circuit Design (RGB-ACD), a large language model (LLM)-guided agent for topology generation and device sizing. RGB-ACD represents circuit topologies as three-channel RGB images and uses this encoding to learn a mapping from text to circuit connectivity. We construct a dataset of 35,200 text-topology pairs and train a generator that predicts electrical connections while preserving fixed device structure. To meet performance targets, the agent couples an outer topology-refinement loop with an inner parameter-optimization loop. The outer LLM uses simulation residuals to revise topology-generation instructions while retaining the original targets. For each candidate topology, the inner loop uses finite-difference gradient estimates to update sensitive parameter groups under process constraints. RGB-ACD outperforms all evaluated baselines in success rate across all three difficulty levels on OCB:Ckt-Bench-101 and AMSBench-Design. On hard tasks, success rates reach 66% and 51%, exceeding the strongest baselines by 42 and 30 percentage points, respectively.

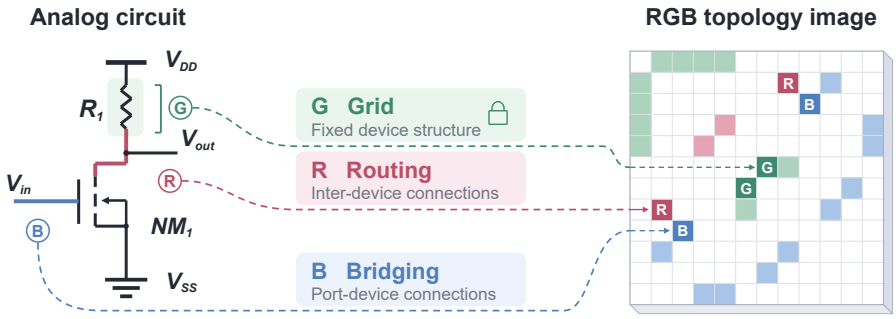


Figure 1: Circuit connections mapped to RGB pixels. Colored leaders link example Routing (R), Grid (G), and Bridging (B) relations to their encoded entries; matching letters mark symmetric pixel pairs. Grid preserves fixed device structure, while Routing and Bridging are predicted. Rows and columns use the same node order.

1 INTRODUCTION

Conventional analog circuit front-end design, which involves circuit topology development and device sizing, relies heavily on engineers' experience and demands substantial time investment. To effectively accelerate this flow, methods and frameworks based on artificial intelligence (AI) have been introduced into electronic design automation for analog circuit design (Shen et al., 2026; Lai et al., 2026; Gao et al., 2025; Liu et al., 2025b). Among these, early methods primarily rely on classical machine learning algorithms, such as Bayesian Optimization (Lu et al., 2021) and Evolutionary Strategy (Zhou et al., 2022), demonstrating promising performance. Furthermore, deep learning methods based on neural networks (Pan et al., 2026; Leng et al., 2025; Liu et al., 2025b) and reinforcement learning (Vijayaraghavan et al., 2025) have been applied by researchers to accomplish various analog circuit automated design tasks. In particular, graph neural network-based methods such as GCN-RL (Wang et al., 2020), TSO-Flow (Han et al., 2024) and CktGNN (Dong et al., 2023)

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

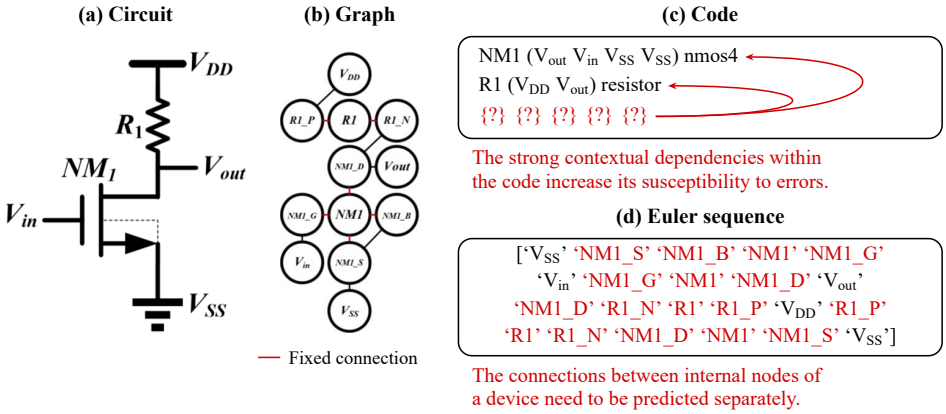


Figure 2: Representations of the same analog circuit. (a) Circuit schematic. (b) Graph representation, with fixed intra-device relations highlighted in red. (c) Netlist-style code (Lai et al., 2025). Arrows indicate contextual dependencies. (d) Eulerian token sequence (Gao et al., 2025). Red tokens identify devices and their terminals.

utilize graph-modal to model the interrelations among circuit devices, thereby enabling efficient analog circuit topology design and device sizing optimization. However, due to the vast search space of analog circuit topologies and parameters, as well as the limitation of computational resources, these methods face challenges of long search times and low efficiency (Scheible, 2022).

Benefiting from large language models (LLMs) pretrained on extensive corpora of human knowledge (OpenAI, 2023), the intelligent agents based on LLMs for simulating circuit design have provided new ideas for addressing the challenges faced by AI-assisted circuit design (Chen et al., 2024; Shen et al., 2026; Liu et al., 2025a; Yin et al., 2024; Lai et al., 2025; Chen et al., 2025b; Li et al., 2025; Chang et al., 2024; Lai et al., 2026; Abbineni et al., 2026; Liu et al., 2024; Chen et al., 2025a). Techniques such as chain-of-thought reasoning (Chen et al., 2024; Lai et al., 2025), retrieval-augmented generation (RAG) (Li et al., 2025; Abbineni et al., 2026) and fine-tuning (Shen et al., 2026; Chen et al., 2025b) have been incorporated into the design of agent to facilitate interactive, dialogue-driven analog circuit design.

Despite these advances, three challenges remain in automated analog circuit design:

Distinguishing circuit connection types. Graphs, netlist-style code, and token sequences encode connectivity differently (Figure 2(b)–(d)). Topology generators must distinguish fixed intra-device relations, such as the red edges in Figure 2(b), from inter-device routing and port connections. Predicting fixed structure adds redundant targets and enlarges the topology search space.

Aligning language with circuit topology. Hallucinations in general-purpose LLMs can cause generated topologies to deviate from natural-language requirements. An accurate, interpretable mapping from user intent to circuit structure remains difficult.

Satisfying coupled performance constraints. Structurally valid topologies may still violate gain, bandwidth, power, phase margin, or noise requirements. Trade-offs among these metrics can prevent device sizing alone from satisfying all specifications.

To address these challenges, we propose RGB-inspired Analog Circuit Design (RGB-ACD), an agent that combines RGB-encoded topology generation with specification-guided circuit refinement. Our main contributions are as follows:

- RGB-ACD offers a general agent framework for analog circuits automated topology design and device sizing optimization.
- We firstly introduce an innovative RGB-inspired representation scheme that leverages different image layers to encode and differentiate distinct types of connections in analog circuits (Figure 1), and we construct a dataset pairing textual descriptions with circuit topologies.

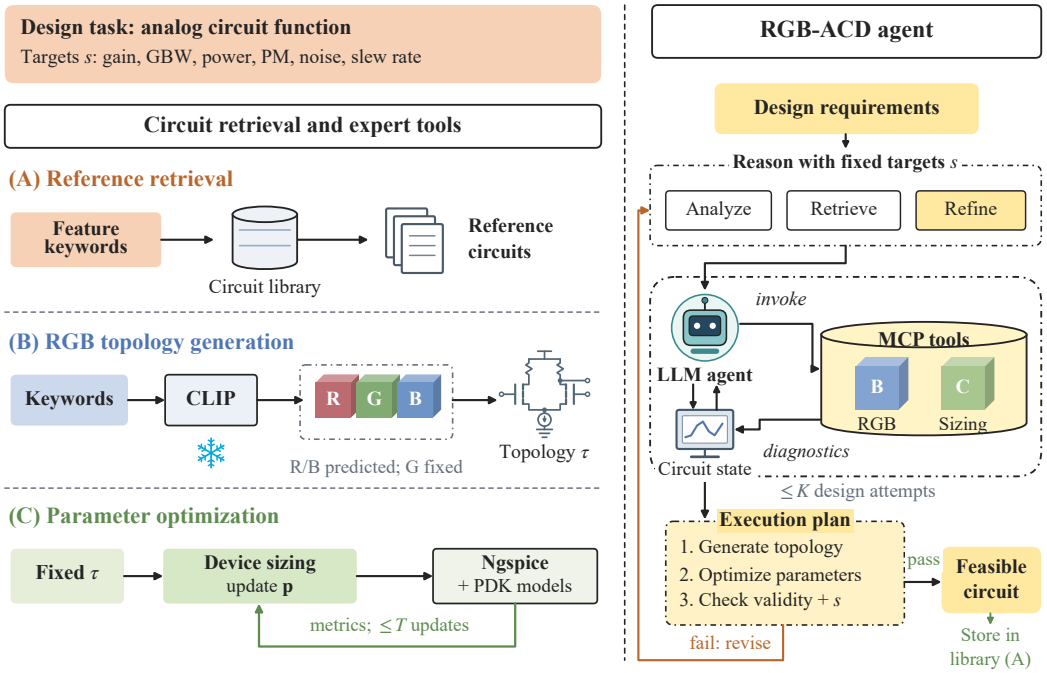


Figure 3: Overview of RGB-ACD. Left: reference retrieval (A), RGB topology generation (B), and fixed-topology parameter optimization (C). Right: the LLM coordinates retrieval and invokes two expert MCP tools. Simulation metrics, residuals, and topology observations guide instruction refinement. T and K specify the maximum numbers of inner updates and outer attempts, respectively.

- We develop a text-conditioned topology generator that predicts symmetric Routing and Bridging maps while preserving fixed device structure.
- We couple topology generation and parameter optimization through specification-guided iterative refinement. The outer LLM revises topology-generation instructions using simulation residuals while keeping design targets fixed. The inner loop fixes the topology and uses finite-difference gradient estimates to update sensitive parameter groups within PDK bounds.

2 Methodology

In this section, we introduce RGB-ACD, an LLM-guided agent for analog circuit topology generation and device sizing.

2.1 Framework Overview

RGB-ACD couples text-conditioned topology generation with simulation-driven device sizing under fixed design specifications (Figure 3). An LLM agent retrieves reference circuits using feature keywords (A) and forms a design instruction while retaining the original specification vector s. It invokes two expert tools through the Model Context Protocol (MCP). The topology-generation tool (B) uses a frozen CLIP encoder to condition the RGB model on instruction-derived keywords. The model predicts Routing and Bridging while preserving fixed device structure, yielding a candidate topology τ.

The parameter-optimization tool (C) holds τ fixed and updates device parameters $\mathbf{p}$ using Ngspice simulations under the target process design kit (PDK). This defines an inner loop with at most T parameter-update attempts per candidate. If the evaluated circuit remains infeasible, the outer loop uses topology observations and available simulation residuals to revise the next topology-generation instruction. The targets s remain fixed across at most K design attempts. Acceptance requires structural validity and satisfaction of all requested specifications within the stated numerical

tolerance; successful designs are stored for subsequent retrieval. Section 2.4 specifies the feedback exchanged between these loops and their termination rules.

2.2 Represent Circuit as Images

2.2.1 Circuit Connection Encoding

We encode circuit topology as three aligned relation maps that distinguish electrical connectivity from fixed device structure (Figure 1). **Routing** represents electrical connections between device terminals, whereas **Bridging** represents connections between device terminals and circuit ports. **Grid** records fixed device-to-terminal incidence, describing device structure rather than electrical shorts. All three maps use the same node ordering for their rows and columns.

For each relation type $c \in \{r, g, b\}$, let $A_c \in \{0,1\}^{H\times W}$ denote its binary relation map. An entry $A_c(i, j)$ equals one when the corresponding relation exists between nodes i and j, and zero otherwise. We construct the RGB image by stacking the three channel maps:

$$I = \mathrm{stack}(L_r, L_g, L_b) \in \{0, 255\}^{H\times W\times 3}, \qquad L_c = 255(1 - A_c). \tag{1}$$

Thus, a present relation has intensity 0, whereas an absent relation has intensity 255. The Grid channel is specified by the device template and remains fixed during generation. The model therefore predicts only Routing and Bridging, excluding predefined device structure from its prediction targets.

At inference, let $P_c(i, j) \in [0, 1]$ denote the predicted connection probability for each generated channel. We discretize these probabilities to recover the two variable relation maps:

$$\widehat{A}_c(i, j) = \mathbf{1}[P_c(i, j) \geq t_{\mathrm{pix}}], \qquad c \in \{r, b\}, \tag{2}$$

where t_{pix} is the probability threshold and $\mathbf{1}[\cdot]$ is the indicator function. The discretized maps are combined with the fixed Grid channel to recover circuit connectivity.

2.2.2 Fixed-Size Circuit Representation

We use a fixed $H = W = 1024$ canvas to represent circuits with different device counts. Rows and columns share a predefined index over device nodes, device terminals, and circuit ports. The template allocates slots for transistors, resistors, capacitors, current sources, and ports. Each circuit occupies a subset of these slots, with the remaining slots left unused. The allocated device and port capacity substantially exceeds the requirements of the analog circuits considered in this work. This spare capacity accommodates variations in circuit size without requiring a separate canvas configuration for each circuit.

This shared indexing scheme preserves the tensor shape across circuit instances and maps each relation entry to a specific node pair. The template defines the available device and port capacity, while the relation maps specify connectivity among the selected nodes. The blank connectivity matrix template and its node indexing are provided in Supplementary File 1.

2.3 Expert Model for Circuit Design

The expert model consists of two components: a topology generation model and a parameter optimization model, both of which are integrated into the circuit design agent framework as MCP modules. The details are described as follows.

2.3.1 Text-Conditioned Topology Generation

We formulate topology generation as conditional prediction of circuit connectivity on a given device template (Figure 4). The generator predicts Routing and Bridging relations while preserving Grid, which specifies device-to-terminal incidence. Its inputs are a circuit description q, the Grid map A_g, and an active-node indicator $\mathbf{a} \in \{0,1\}^H$. The template and active-node set specify the available devices and ports for each candidate; their selection is outside this generator. This separates connectivity prediction from the fixed device structure defined in Section 2.2.

Text-conditioned residual blocks. The LLM extracts circuit-design keywords from its current instruction to form q. A frozen CLIP text encoder (Radford et al., 2021) produces an ℓ_2-normalized

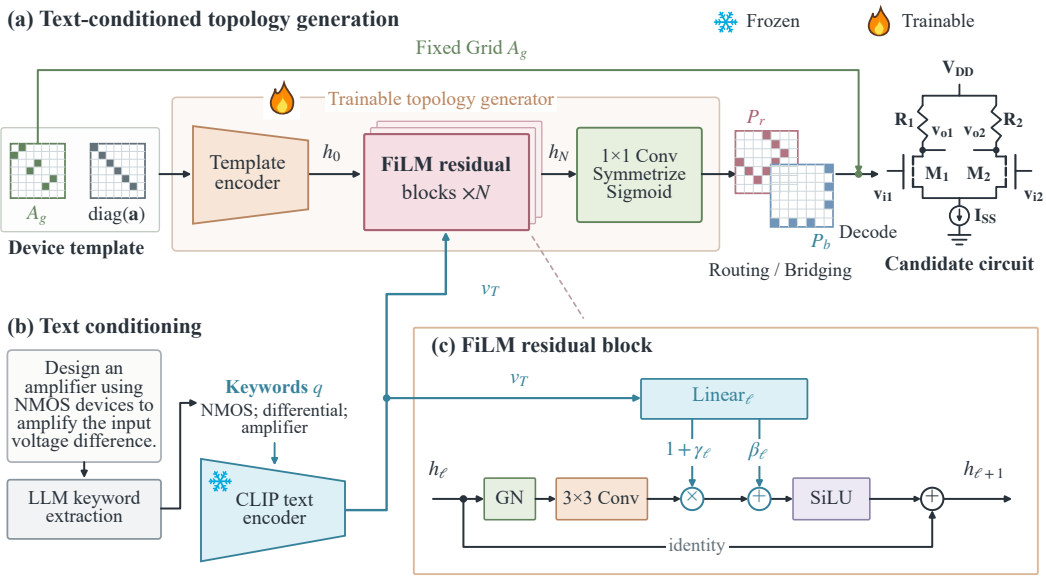


Figure 4: Text-conditioned topology generation. (a) The generator predicts Routing and Bridging while preserving Grid. (b) Frozen CLIP encodes instruction-derived keywords. (c) Each residual block applies its own text-conditioned scale and shift. Relation maps and the circuit are schematic illustrations.

embedding $v_T \in \mathbb{R}^{d_T}$. A convolutional stem encodes the template and positional information into an initial feature map h_0. We then apply N residual convolutional blocks with feature-wise linear modulation (FiLM) (Perez et al., 2018). Each block has its own parameters and directly maps v_T to channel-wise scale and shift vectors:

$$\begin{aligned} [\gamma_\ell; \beta_\ell] &= W_\ell v_T + b_\ell, \qquad u_\ell = \mathrm{Conv}_{3\times3,\ell}(\mathrm{GN}_\ell(h_\ell))\,, \\ h_{\ell+1} &= h_\ell + \mathrm{SiLU}((1+\gamma_\ell)\odot u_\ell + \beta_\ell)\,, \qquad \ell = 0, \ldots, N-1. \end{aligned} \tag{3}$$

Here, GN denotes group normalization, and the modulation vectors are broadcast across spatial locations. Thus, N specifies network depth within one forward pass. Appendix E gives the template encoding and initialization of h_0.

Symmetric connectivity prediction. A 1×1 convolution maps h_N to two logit maps U_r and U_b. Because electrical connections are undirected, we symmetrize each map before applying the logistic sigmoid σ:

$$P_c = \sigma\left(\frac{U_c + U_c^\top}{2}\right), \qquad c \in \{r, b\}. \tag{4}$$

This enforces $P_c(i,j) = P_c(j,i)$ by construction. The fixed Grid map is excluded from the prediction targets.

Training and decoding. We supervise the two predicted relation maps using positive-class-weighted binary cross-entropy and add two differentiable structural regularizers:

$$\mathcal{L}_{\mathrm{topo}} = \mathcal{L}_{\mathrm{WBCE}} + \lambda_{\mathrm{diag}}\mathcal{L}_{\mathrm{diag}} + \lambda_{\mathrm{deg}}\mathcal{L}_{\mathrm{deg}}. \tag{5}$$

The self-loop penalty $\mathcal{L}_{\mathrm{diag}}$ suppresses diagonal connection probabilities. The soft-degree regularizer $\mathcal{L}_{\mathrm{deg}}$ penalizes active nodes whose summed connection probabilities fall below a prescribed margin. Appendix E defines both terms and their normalization. These penalties provide local structural preferences but do not guarantee freedom from short circuits or connectivity to legal ports after thresholding.

At inference, Equation 2 discretizes P_r and P_b, which are combined with the fixed Grid map to form a candidate topology. Numerical targets remain in the specification vector s (Equation 6) and influence generation through the LLM's revised instructions. Structural acceptance and specification satisfaction are evaluated by the downstream design loop, rather than guaranteed by the topology predictor.

2.3.2 PARAMETER OPTIMIZATION

Given a candidate topology τ, we formulate device sizing as constrained black-box optimization with Ngspice as the evaluator. Matched devices, including differential pairs and current mirrors, share parameter groups that enforce symmetry constraints. The resulting vector $\mathbf{p} \in \mathcal{P}_\tau$ contains the positive free parameters, where $\mathcal{P}_\tau$ specifies their admissible bounds under the target PDK. Circuit connectivity remains fixed throughout this inner optimization loop.

Specification objective. The optimizer receives the original numerical targets as a structured vector

$$s = [g, \mathrm{GBW}, P, \mathrm{PM}, n, \mathrm{SR}_{\mathrm{slew}}, \ldots], \tag{6}$$

whose entries specify gain, gain-bandwidth product, power, phase margin, noise, and slew rate, respectively. Each target retains its bound direction and units across design attempts. For simulated metrics $m(\tau, \mathbf{p})$, we minimize the objective

$$\underset{\mathbf{p}\in\mathcal{P}_\tau}{\text{minimize}} \quad \mathcal{L}_{\mathrm{opt}}(\mathbf{p} \mid \tau, s) = \sum_k w_k \, v_k(\mathbf{p}) + \eta \, \mathcal{R}(\mathbf{p}), \tag{7}$$

where v_k is the positive bound violation normalized by $\max(|s_k|, \epsilon_k)$, with a fixed floor $\epsilon_k > 0$ in the metric's units. It penalizes deficiency for lower bounds and excess for upper bounds, and vanishes when the bound is met. This normalization yields dimensionless penalties across metrics with different units. We set $w_k = 1$ and $\eta = 10^{-3}$; the dimensionless regularizer $\mathcal{R}$ penalizes unrealistic parameter values within $\mathcal{P}_\tau$.

Sensitivity-guided search. At each iteration, paired multiplicative perturbations estimate local objective sensitivities using their actual separation in log space. We select up to $\min(8, \lceil 0.2d \rceil)$ groups with the largest absolute sensitivities, where d is the number of free parameter groups. A projected step in $\mathbf{x} = \log \mathbf{p}$ updates only these groups while preserving positivity and PDK bounds. We accept the trial only when the objective decrease meets the prescribed threshold; otherwise, we retain the current parameters and halve the step size. Algorithm 1 in Appendix C specifies the perturbations, projection, acceptance threshold, and failure handling.

Each candidate receives at most T update attempts and 400 Ngspice calls, including independent final verification. Success requires this final simulation to confirm circuit validity, finite values for every required metric, and $\max_k v_k \leq 10^{-3}$. Otherwise, the returned parameters and available simulation residuals inform the outer refinement loop in Section 2.4.

2.4 SPECIFICATION-GUIDED ITERATIVE REFINEMENT

RGB-ACD uses simulation feedback to revise topology proposals when bounded parameter optimization does not yield a verified feasible circuit. This outer loop complements the fixed-topology optimization in Section 2.3.2, while preserving the original specification vector s throughout the search.

Specification-preserving feedback. At outer round r, the inner loop returns parameters $\mathbf{p}_r$ for topology τ_r, together with the corresponding simulation status and diagnostics. The feedback combines topology-level observations with available metrics $m_k(\tau_r, \mathbf{p}_r)$ and signed residuals $m_k(\tau_r, \mathbf{p}_r) - s_k$. Residuals retain metric identities, target bounds, inequality directions, and units. A negative residual indicates a deficit for a lower-bound constraint, whereas a positive residual indicates excess for an upper-bound constraint. Unavailable metrics remain explicitly marked as missing in the feedback.

Feedback-conditioned topology revision. After an unsuccessful attempt, the LLM uses these diagnostics and the original targets to construct the next design instruction d_{r+1}. Frozen CLIP encodes circuit-design keywords extracted from d_{r+1} to condition the topology generator in Section 2.3.1. The resulting candidate τ_{r+1} undergoes parameter optimization against the same specification vector s. Numerical requirements thus guide topology generation through instruction refinement and remain explicit constraints during sizing and verification.

Budgets and termination. The search permits at most K outer design attempts and T inner parameter-update attempts per candidate, subject to the simulation budget in Section 2.3.2. Acceptance requires independent final simulation to confirm structural validity, finite values for all required metrics, and normalized violations $\max_k v_k \leq 10^{-3}$. The procedure returns the first verified feasible circuit or reports failure when the outer budget is exhausted. Successful topologies, parameters, and features are stored for subsequent retrieval. Algorithm 2 in Appendix C summarizes the procedure.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETUP

Implementation Details. We implement the LLM agent using the GPT-5.5 API. The topology generator uses $N = 8$ FiLM-conditioned residual blocks and a frozen CLIP text encoder with embedding dimension $d_T = 512$. We train the generator on 35,200 circuit representations paired with textual descriptions, using AdamW for 1,000 epochs on a single NVIDIA H100 SXM GPU. The batch size is 16, and each circuit is encoded on a 1024×1024 canvas. Dataset construction and augmentation are detailed in Appendix B. Here, N denotes network depth. For RGB-ACD, we set $T = 10$ inner parameter-update attempts per candidate topology and $K = 15$ outer design attempts (Section 2.4).

Simulation Environment. We use Ngspice with the SkyWater 130 nm process design kit (SKY130 PDK) (SkyWater PDK Authors, 2020) for circuit evaluation and parameter optimization. Simulations include direct-current (DC), alternating-current (AC), transient, and noise analyses. Circuit acceptance follows the structural validity and specification satisfaction criteria in Section 2.4.

Baselines. We assess functional coverage and end-to-end design performance separately. Table 1 compares topology generation, parameter optimization, open-ended design, and support for multiple circuit types. This qualitative comparison covers AnalogGenie (Gao et al., 2025), ATOM (Shen et al., 2025), DiffCkt (Liu et al., 2025b), CktGNN (Dong et al., 2023), ADO-LLM (Yin et al., 2024), PZ-Agent (Li et al., 2025), Artisan (Chen et al., 2024), LaMAGIC (Chang et al., 2024), and AnalogSeeker (Chen et al., 2025b).

For quantitative evaluation, we compare RGB-ACD against four baselines. These include the circuit-design method AnalogSeeker and a retrieval-augmented Qwen3-32B model using our dataset, denoted Qwen3-32B (RAG). We also evaluate two general-purpose LLMs: ChatGPT-5.5 and DeepSeek-V4-Pro. We report success rate (SR), average number of attempts (ANoA), and runtime for each method.

3.2 BENCHMARK

We evaluate on OCB:Ckt-Bench-101 for design under explicit numerical constraints and AMSBench-design for open-ended design across diverse circuit categories. Task definitions, construction details, and difficulty-level counts are provided in Appendix D.

3.3 EVALUATION METRICS

We evaluate end-to-end circuit design using success rate (SR), average number of attempts (ANoA), and runtime. Tables 2 and 3 report these metrics separately for each benchmark and difficulty level.

Success rate (SR). SR is the fraction of tasks solved within the allotted attempt budget. A successful design must be structurally valid and satisfy all original performance specifications after parameter optimization, following the acceptance criterion in Section 2.4.

Average number of attempts (ANoA). ANoA is the reported mean number of attempts required to obtain a successful design. The available aggregates do not distinguish outer topology attempts from inner parameter updates, limiting their interpretation as a measure of computational cost.

Runtime. Runtime measures task execution time in seconds, including topology generation and parameter optimization. We interpret runtime jointly with SR and ANoA to assess the trade-off between successful task completion and execution cost.

3.4 EXPERIMENTAL RESULTS

End-to-end design performance. RGB-ACD achieved the highest SR across both benchmarks and all three difficulty levels, outperforming the four baselines in Table 2. The improvement over the strongest baseline in each setting ranged from 15 to 42 percentage points. On Hard tasks, SR reached 66% on OCB:Ckt-Bench-101 and 51% on AMSBench-Design. The corresponding best baseline SRs

Table 1: Capability comparison of different methods.

Method	Topology Generation	Parameter Optimization	Open-ended Design	Multiple Circuits
AnalogGenie (Gao et al., 2025)	✓	✗	✗	✓
ATOM (Shen et al., 2025)	✓	✓	✗	✗
DiffCkt (Liu et al., 2025b)	✓	✓	✗	✗
CktGNN (Dong et al., 2023)	✓	✓	✗	✗
ADO-LLM (Yin et al., 2024)	✗	✓	✓	✓
PZ-Agent (Li et al., 2025)	✓	✗	✓	✗
Artisan (Chen et al., 2024)	✓	✓	✓	✗
LaMAGIC (Chang et al., 2024)	✓	✗	✓	✗
AnalogSeeker (Chen et al., 2025b)	✓	✓	✓	✗
RGB-ACD (ours)	✓	✓	✓	✓

Table 2: Results on OCB:Ckt-Bench-101 and AMSBench-Design.

Method	OCB:Ckt-Bench-101 (Dong et al., 2023)								
	Easy			Medium			Hard		
	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓
AnalogSeeker (Chen et al., 2025b)	0.50	5.0	101	0.46	6.1	123	0.17	6.7	139
ChatGPT-5.5	0.42	4.1	223	0.29	5.7	249	0.19	8.1	301
DeepSeek-V4-Pro	0.52	4.0	186	0.33	6.9	229	0.15	8.3	256
Qwen3-32B (RAG)	0.53	5.9	145	0.36	6.8	156	0.24	7.6	172
RGB-ACD (ours)	**0.86**	**2.9**	**78**	**0.73**	**3.6**	**92**	**0.66**	**4.4**	**109**
Method	AMSBench-Design								
	Easy			Medium			Hard		
	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓
AnalogSeeker (Chen et al., 2025b)	0.56	4.1	103	0.32	4.5	125	0.21	5.9	147
ChatGPT-5.5	0.50	4.7	195	0.44	6.3	214	0.21	6.5	239
DeepSeek-V4-Pro	0.52	5.9	197	0.31	7.1	217	0.17	7.7	243
Qwen3-32B (RAG)	0.35	5.5	90	0.31	6.9	108	0.18	7.1	132
RGB-ACD (ours)	**0.71**	**3.2**	**82**	**0.62**	**3.9**	**99**	**0.51**	**4.8**	**117**

were 24% for Qwen3-32B (RAG) and 21% for both AnalogSeeker and ChatGPT-5.5. These results represent gains of 42 and 30 percentage points under the reported evaluation, respectively.

Reported search cost. RGB-ACD recorded lower ANoA and runtime than all four baselines in each evaluation setting. On Hard tasks, ANoA was 4.4 versus 7.6 for Qwen3-32B (RAG) on OCB:Ckt-Bench-101, and 4.8 versus 5.9 for AnalogSeeker on AMSBench-Design. These correspond to approximately 42% and 19% fewer reported attempts, respectively. Hard-task runtimes were 109 s and 117 s, below the fastest baseline times of 139 s and 132 s, respectively. The ANoA aggregates do not separate outer topology attempts from inner parameter updates, so they do not establish equal computational effort. The variant without outer-loop feedback ran faster but had lower SR in every setting, a trade-off examined in the following ablation study.

Capability coverage. Table 1 summarizes functional support for topology generation, parameter optimization, open-ended design, and multiple circuit types. RGB-ACD integrates all four capabilities, while the listed baselines support different subsets under the table's definitions. This qualitative comparison describes functional coverage, complementing the end-to-end performance measured in Table 2.

Table 3: Ablation results on OCB:Ckt-Bench-101 and AMSBench-Design.

Method	OCB:Ckt-Bench-101 (Dong et al., 2023)								
	Easy			Medium			Hard		
	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓
RGB-ACD (ours)	**0.86**	2.9	78	**0.73**	3.6	92	**0.66**	4.4	109
RGB-ACD (w/o multi-layer representation)	0.53	3.8	70	0.38	4.8	88	0.31	5.9	111
RGB-ACD (w/o outer feedback)	0.54	**1.0**	**24**	0.51	**1.0**	**33**	0.43	**1.0**	**43**
Method	AMSBench-Design								
	Easy			Medium			Hard		
	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓	SR↑	ANoA↓	Runtime (s)↓
RGB-ACD (ours)	**0.71**	3.2	82	**0.62**	3.9	99	**0.51**	4.8	117
RGB-ACD (w/o multi-layer representation)	0.38	3.5	65	0.32	4.4	81	0.26	5.5	102
RGB-ACD (w/o outer feedback)	0.54	**1.0**	**30**	0.43	**1.0**	**36**	0.28	**1.0**	**37**

3.5 Ablation Study

We ablate two components of RGB-ACD: the multi-layer RGB representation and the outer-loop feedback used for specification-guided iterative refinement. Table 3 compares both variants with the full RGB-ACD model.

Effect of Multi-layer Representation. We evaluate the contribution of the proposed RGB-style multi-layer representation by replacing it with a single-layer topology encoding. The corresponding ablation variant is denoted as RGB-ACD (w/o multi-layer representation). As shown in Table 3, removing the multi-layer representation reduced SR and increased ANoA across both benchmarks and all three difficulty levels. On Hard tasks, SR fell from 66% to 31% on OCB:Ckt-Bench-101 and from 51% to 26% on AMSBench-Design.

Effect of Outer-Loop Feedback. The RGB-ACD (w/o outer feedback) variant disables outer-loop feedback between the LLM and the expert model while retaining topology generation and parameter optimization. Removing this feedback does not remove the inner parameter-optimization procedure. The reported ANoA is 1.0 across both benchmarks and all three difficulty levels.

On hard tasks, disabling outer-loop feedback reduces the reported success rate from 66% to 43% on OCB:Ckt-Bench-101 and from 51% to 28% on AMSBench-Design. Both differences are 23 percentage points. The corresponding runtime decreases from 109 to 43 seconds and from 117 to 37 seconds, respectively. Across all difficulty levels, the feedback-enabled configuration has higher success rates and longer runtime than the variant without outer-loop feedback.

4 Conclusion

We introduced RGB-ACD, an LLM-guided framework for analog circuit design that couples RGB-based topology generation with specification-guided refinement. Its multi-layer representation separates circuit connection types, while nested optimization loops use simulation feedback to refine topology and device parameters against fixed design targets. Across OCB:Ckt-Bench-101 and AMSBench-Design, RGB-ACD achieved higher success rates and lower runtime than all evaluated baselines at every difficulty level. These results support combining structured circuit representations with specification-guided simulation feedback for automated analog circuit design.

REFERENCES

Pravallika Abbineni, Saoud Aldowaish, Colin Liechty, Soroosh Noorzad, Ali Ghazizadeh Ghalati, and Morteza Fayazi. MuaLLM: A multimodal large language model agent for circuit design assistance with hybrid contextual retrieval-augmented generation. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 646–652, 2026. doi: 10.1109/asp-dac66049.2026.11420505.

Chen-Chia Chang, Yikang Shen, Shaoze Fan, Jing Li, Shun Zhang, Ningyuan Cao, Yiran Chen, and Xin Zhang. LaMAGIC: Language-model-based topology generation for analog integrated circuits. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 6253–6262. PMLR, 2024. URL https://proceedings.mlr.press/v235/chang24c.html.

Jianqiu Chen, Siqi Li, and Xu He. White-box reasoning: Synergizing LLM strategy and gm/Id data for automated analog circuit design. arXiv preprint arXiv:2508.13172, 2025a. URL https://arxiv.org/abs/2508.13172.

Zihao Chen, Jiangli Huang, Yiting Liu, Fan Yang, Li Shang, Dian Zhou, and Xuan Zeng. Artisan: Automated operational amplifier design via domain-specific large language model. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2024. doi: 10.1145/3649329.3655903.

Zihao Chen, Ji Zhuang, Jinyi Shen, Xiaoyue Ke, Xinyi Yang, Mingjie Zhou, Zhuoyao Du, Xu Yan, Zhouyang Wu, Zhenyu Xu, Jiangli Huang, Li Shang, Xuan Zeng, and Fan Yang. AnalogSeeker: An open-source foundation language model for analog circuit design. arXiv preprint arXiv:2508.10409v2, 2025b. URL https://arxiv.org/abs/2508.10409v2.

Zehao Dong, Weidong Cao, Muhan Zhang, Dacheng Tao, Yixin Chen, and Xuan Zhang. CktGNN: Circuit graph neural network for electronic design automation. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=NE2911Kq1sp.

Jian Gao, Weidong Cao, Junyi Yang, and Xuan Zhang. AnalogGenie: A generative engine for automatic discovery of analog circuit topologies. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f0552f14388d95b19740dee809f5cad1-Abstract-Conference.html.

Jinglin Han, Yuhao Leng, Xiuli Zhang, and Peng Wang. TSO-Flow: A topology synthesis and optimization workflow for operational amplifiers with invertible graph generative model. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–9, 2024. doi: 10.1145/3676536.3676693.

Yao Lai, Sungyoung Lee, Guojin Chen, Souradip Poddar, Mengkang Hu, David Z. Pan, and Ping Luo. AnalogCoder: Analog circuit design via training-free code generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 379–387, 2025. doi: 10.1609/aaai.v39i1.32016.

Yao Lai, Souradip Poddar, Sungyoung Lee, Guojin Chen, Mengkang Hu, Bei Yu, Ping Luo, and David Z. Pan. AnalogCoder-Pro: Unifying analog circuit generation and optimization via multimodal LLMs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2026. doi: 10.1109/tcad.2026.3673493. Early access.

YuHao Leng, Jinglin Han, Yining Wang, and Peng Wang. TPC-GAN: Batch topology synthesis for performance-compliant operational amplifiers using generative adversarial networks. In *Proceedings of the Design, Automation & Test in Europe Conference (DATE)*, pp. 1–7, 2025. doi: 10.23919/date64628.2025.10992776.

Mingzhen Li, Bo Li, Tingjie Yang, and Guoyong Shi. PZ-Agent: A symbolic-engine enhanced LLM agent for op-amp topology inquisition. In *Proceedings of the International Symposium of Electronics Design Automation (ISEDA)*, pp. 730–735, 2025. doi: 10.1109/iseda65950.2025.11100744.

Chengjie Liu, Yijiang Liu, Yuan Du, and Li Du. LADAC: Large language model-driven auto-designer for analog circuits. TechRxiv preprint, 2024. URL https://doi.org/10.36227/techrxiv.170473941.10097233/v1.

Chengjie Liu, Weiyu Chen, Huiyao Xu, Yuan Du, Jun Yang, and Li Du. A large language model-based multi-agent framework for analog circuits' sizing relationships extraction. In *Proceedings of the*

International Symposium of Electronics Design Automation (ISEDA), pp. 181–187, 2025a. doi: 10.1109/iseda65950.2025.11101698.

Chengjie Liu, Jiajia Li, Yabing Feng, Wenhao Huang, Weiyu Chen, Yuan Du, Jun Yang, and Li Du. DiffCkt: A diffusion model-based hybrid neural network framework for automatic transistor-level generation of analog circuits. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–9, 2025b. doi: 10.1109/iccad66269.2025.11240784.

Jialin Lu, Liangbo Lei, Fan Yang, Changhao Yan, and Xuan Zeng. Automated compensation scheme design for operational amplifier via Bayesian optimization. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*, pp. 517–522, 2021. doi: 10.1109/dac18074.2021.9586306.

OpenAI. GPT-4 technical report. arXiv preprint arXiv:2303.08774, 2023. URL https://arxiv.org/abs/2303.08774.

Guanyuan Pan, Tiansheng Zhou, Jianxiang Zhao, Zhi Li, Yugui Lin, Bingtao Ma, Yaqi Wang, Pietro Liò, and Shuai Wang. Graph neural networks based analog circuit link prediction. *Engineering Applications of Artificial Intelligence*, 163:113035, 2026. doi: 10.1016/j.engappai.2025.113035.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 3942–3951. AAAI Press, 2018. doi: 10.1609/aaai.v32i1.11671.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Juergen Scheible. Optimized is not always optimal - the dilemma of analog design automation. In *Proceedings of the International Symposium on Physical Design (ISPD)*, pp. 151–158, 2022. doi: 10.1145/3505170.3511042.

Jinyi Shen, Fan Yang, Li Shang, Changhao Yan, Zhaori Bi, Dian Zhou, and Xuan Zeng. ATOM: An automatic topology synthesis framework for operational amplifiers. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 44(3):1193–1198, 2025. doi: 10.1109/tcad.2024.3463534.

Jinyi Shen, Zihao Chen, Ji Zhuang, Jiangli Huang, Fan Yang, Li Shang, Zhaori Bi, Changhao Yan, Dian Zhou, and Xuan Zeng. Atelier: An automated analog circuit design framework via multiple large language model-based agents. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 45(1):31–44, 2026. doi: 10.1109/tcad.2025.3573228.

Yichen Shi, Ze Zhang, Hongyang Wang, Zhuofu Tao, Zhongyi Li, Bingyu Chen, Yaxin Wang, Zhen Huang, Xuhua Liu, Quan Chen, Zhiping Yu, Ting-Jung Lin, and Lei He. AMSbench: A comprehensive benchmark for evaluating MLLM capabilities in AMS circuits. arXiv preprint arXiv:2505.24138, 2025. URL https://arxiv.org/abs/2505.24138.

SkyWater PDK Authors. SkyWater SKY130 PDK, 2020. URL https://skywater-pdk.readthedocs.io/en/main/.

Prashanth Vijayaraghavan, Luyao Shi, Ehsan Degan, Vandana Mukherjee, and Xin Zhang. AUTOCIRCUIT-RL: Reinforcement learning-driven LLM for automated circuit topology generation. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 61498–61512. PMLR, 2025. URL https://proceedings.mlr.press/v267/vijayaraghavan25a.html.

Hanrui Wang, Kuan Wang, Jiacheng Yang, Linxiao Shen, Nan Sun, Hae-Seung Lee, and Song Han. GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2020. doi: 10.1109/dac18072.2020.9218757.

Yuxuan Yin, Yu Wang, Boxun Xu, and Peng Li. ADO-LLM: Analog design Bayesian optimization with in-context learning of large language models. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–9, 2024. doi: 10.1145/3676536.3676816.

Ranran Zhou, Peter Poechmueller, and Yong Wang. An analog circuit design and optimization system with rule-guided genetic algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(12):5182–5192, 2022. doi: 10.1109/tcad.2022.3166637.

A Background and Problem Formulation

A.1 Topology and Device Sizing

Analog circuit design couples discrete topology selection with continuous device sizing (Dong et al., 2023; Gao et al., 2025). We represent a circuit by $(\tau, \mathbf{p})$, where τ specifies device types, counts, and terminal connections. The parameter vector $\mathbf{p}$ contains transistor dimensions, resistance values, and capacitance values. Under a fixed process design kit (PDK) and simulation conditions, the simulator evaluates the performance vector $m(\tau, \mathbf{p})$, including gain, bandwidth, and power. These metrics depend jointly on topology and sizing; a structurally valid topology need not admit parameters that satisfy every performance bound.

We consider topology generation and device sizing before physical layout. Each generation call receives a device template and an active-node set that specify the available devices and ports. The generator predicts their connectivity; device-inventory selection lies outside its scope (Section 2.3.1).

A.2 Terminal-Level Circuit Representation

Circuit representations must distinguish electrical connectivity from device-to-terminal incidence (Gao et al., 2025). A MOS transistor, for example, has gate, drain, source, and body terminals. Their shared device membership does not imply an electrical connection between them. External ports provide the circuit's interfaces for signal and supply connections.

With a shared node ordering, binary relation matrices encode connections or incidence through pairwise entries. RGB-ACD separates terminal-to-terminal connections (Routing), device-to-terminal incidence (Grid), and terminal-to-port connections (Bridging), as defined in Section 2.2. Matrix indices identify circuit entities, so image coordinates do not encode physical layout positions. Fixing Grid reduces topology generation to predicting the two electrical relation maps.

A.3 Specification-Constrained Circuit Design

Given a natural-language request and specification vector s, we seek a circuit that implements the requested functionality and satisfies all performance bounds. Each target retains its metric, units, and inequality direction, such as minimum gain or maximum power. Let $\mathcal{T}_{\text{valid}}$ denote the admissible, structurally valid topologies for the task, and $\mathcal{P}_\tau$ the parameter domain allowed by the target PDK. Let $\mathcal{S}(s)$ contain all finite-valued performance vectors satisfying every numerical bound in s. We formulate design as a joint discrete–continuous feasibility problem:

$$\begin{aligned} \text{find} \quad & (\tau, \mathbf{p}) \\ \text{subject to} \quad & \tau \in \mathcal{T}_{\text{valid}}, \qquad \mathbf{p} \in \mathcal{P}_\tau, \qquad m(\tau, \mathbf{p}) \in \mathcal{S}(s). \end{aligned} \tag{8}$$

RGB-ACD searches this space by alternating topology proposals and device sizing while keeping s fixed. Acceptance requires independent simulation to verify circuit validity and satisfaction of every required performance bound. Section 2.4 specifies the search budgets and normalized feasibility tolerance used in practice. Exhausting the search budget without a verified design does not establish infeasibility.

B Dataset and Data Augmentation

We first convert circuit structures from the AnalogGenie dataset (Gao et al., 2025) into the image-based representation introduced in Section 2.2. By integrating multimodal inputs—including circuit images, Cadence schematic renderings, node-connection adjacency matrices, and specification tables—into the LLM for unified analysis, we derive high-quality textual descriptions that capture both the essential structural features and the quantitative performance intent of each circuit. We manually check these descriptions for consistency with the corresponding circuit structures and numerical specifications. We further enrich the dataset by substituting functionally equivalent devices within the image-based circuit representations, varying specification ranges under the same topology family, and introducing synonym-driven paraphrasing of the circuit feature descriptions to enhance linguistic diversity. Overall, we obtain 35,200 image-based circuit representations together with their aligned feature descriptions for model training.

C Inner Optimization and Outer Refinement

Algorithm 1 details parameter optimization for a fixed topology. Algorithm 2 describes the outer feedback flow from Section 2.4. Both retain the original targets s; T bounds inner update attempts and K bounds outer design attempts.

Algorithm 1 Detailed parameter optimization for a fixed topology

Require: Fixed topology τ, original targets s, update budget T; configured ϵ_k, δ_j, and $\mathcal{R}$
Ensure: Success or failure, current parameters $\mathbf{p}$, and simulation diagnostics $\mathcal{D}$

Bind matched-device constraints; initialize d positive free parameter groups within PDK bounds
Set $\mathbf{x} = \log \mathbf{p}$ and $\mathcal{B} = [\log \mathbf{p}^{\min}, \log \mathbf{p}^{\max}]$
Define normalized violations for each required metric $m_k = m_k(\tau, \mathbf{p})$:

$$v_k(\mathbf{p}) = \begin{cases} \dfrac{\max(0, s_k - m_k)}{\max(|s_k|, \epsilon_k)}, & m_k \geq s_k \text{ required,} \\ \dfrac{\max(0, m_k - s_k)}{\max(|s_k|, \epsilon_k)}, & m_k \leq s_k \text{ required.} \end{cases}$$

Define $J(\mathbf{p}) = \sum_k w_k v_k(\mathbf{p}) + \eta \mathcal{R}(\mathbf{p})$, with $w_k = 1$ and $\eta = 10^{-3}$
Set $\alpha = 0.05$, $\alpha_{\min} = 0.005$, $\varepsilon_{\text{feas}} = 10^{-3}$, $B = 400$, $c = 0$, $\mathcal{D} = \emptyset$
Evaluate $\mathbf{p}$ using Ngspice; each evaluation increments c and records diagnostics in $\mathcal{D}$
if simulation fails, the circuit is invalid, or any required metric is unavailable or non-finite **then**
    return failure, $\mathbf{p}$, $\mathcal{D}$
end if
for $t = 1, \ldots, T$ **do**
    if $\max_k v_k(\mathbf{p}) \leq \varepsilon_{\text{feas}}$ or $\alpha < \alpha_{\min}$ or $c + 2d + 1 > B - 1$ **then**
        break ▷ Reserve one call for independent final verification
    end if
    Rank specifications by increasing normalized slack; record their order for feedback
    for each free parameter group $j = 1, \ldots, d$ **do**
        Form $\mathbf{p}_{j,\pm}$ by replacing p_j with $p_{j,\pm} = \text{clip}(p_j(1 \pm \delta_j), p_j^{\min}, p_j^{\max})$
        Evaluate both perturbations; exclude group j if either fails or $p_{j,+} = p_{j,-}$
        For retained groups, set $g_j = \dfrac{J(\mathbf{p}_{j,+}) - J(\mathbf{p}_{j,-})}{\log p_{j,+} - \log p_{j,-}}$
    end for
    Select up to $q = \min(8, \lceil 0.2d \rceil)$ retained groups with largest $|g_j|$ into S
    Set $\widetilde{g}_j = g_j$ for $j \in S$ and $\widetilde{g}_j = 0$ otherwise
    if $S = \emptyset$ or $\|\widetilde{\mathbf{g}}\|_\infty = 0$ **then**
        break
    end if
    Form $\mathbf{x}' = \Pi_{\mathcal{B}}\left[\mathbf{x} - \alpha \dfrac{\widetilde{\mathbf{g}}}{\max(\|\widetilde{\mathbf{g}}\|_\infty, 10^{-12})}\right]$ and $\mathbf{p}' = \exp(\mathbf{x}')$
    Evaluate $\mathbf{p}'$; set $J(\mathbf{p}') = +\infty$ if the evaluation fails
    if $J(\mathbf{p}) - J(\mathbf{p}') \geq 10^{-4} \max(1, J(\mathbf{p}))$ **then**
        Accept $\mathbf{x} \leftarrow \mathbf{x}'$, $\mathbf{p} \leftarrow \mathbf{p}'$ and the corresponding evaluated metrics
    else
        Reject the trial; set $\alpha \leftarrow \alpha/2$
    end if
end for
Independently re-simulate the current $\mathbf{p}$ and append its diagnostics to $\mathcal{D}$
if final simulation succeeds, the circuit is valid, all required metrics are finite, and $\max_k v_k \leq \varepsilon_{\text{feas}}$ **then**
    return success, $\mathbf{p}$, $\mathcal{D}$
else
    return failure, $\mathbf{p}$, $\mathcal{D}$
end if

Parameterization and objective. Each coordinate of $\mathbf{p}$ represents a free parameter group, with matched-device constraints enforced when reconstructing the circuit parameters. Only groups with $0 < p_j^{\min} < p_j^{\max}$ enter the log-space search; fixed parameters remain unchanged. The operator $\Pi_{\mathcal{B}}$ clips each log-coordinate to its admissible interval. The normalization floor $\epsilon_k > 0$ has the same units as metric k, and each relative perturbation satisfies $0 < \delta_j < 1$. These settings and the dimensionless regularizer $\mathcal{R}$ are fixed before optimization, with $\mathcal{R}$ finite throughout the admissible parameter domain. Here, J denotes $\mathcal{L}_{\text{opt}}$ for fixed τ and s. Sensitivities use the actual log-space

separation after clipping, keeping their coordinates consistent with the update rule. A group is excluded if its perturbations coincide numerically or either simulation lacks a finite required metric.

Selection, acceptance, and feedback. Normalized slack is $(m_k - s_k)/\max(|s_k|, \epsilon_k)$ for lower bounds and $(s_k - m_k)/\max(|s_k|, \epsilon_k)$ for upper bounds. Increasing slack orders the largest violations first for feedback; parameter selection instead ranks the objective sensitivities by magnitude. Trial parameters are accepted only when the objective decrease reaches the stated threshold. Rejection preserves the current parameters and halves the step size. Each diagnostic record retains the evaluated parameters, simulation status, available metrics, and signed residuals $m_k - s_k$, together with bound directions and units. Missing metrics remain explicitly unavailable. Algorithm 2 receives diagnostics tied to the returned parameters, including the final verification record when available.

Budgets and verification. The limit of $B = 400$ Ngspice calls includes initialization, perturbations, trial evaluations, failed calls, and independent final verification. A new update attempt requires enough remaining calls for all $2d$ perturbations, one trial, and the reserved verification. Thus, T bounds update attempts rather than simulator calls or accepted updates. Every successful return requires independent verification of circuit validity and all original bounds within the normalized tolerance $\varepsilon_{\text{feas}} = 10^{-3}$. The original targets remain fixed across both loops; the tolerance is a numerical acceptance criterion. Failure denotes termination without a verified feasible design, rather than proof that the topology admits no feasible parameters.

Algorithm 2 Specification-guided outer refinement: schematic information flow

Require: Task x, original targets s, outer budget K, inner budget T
Ensure: A feasible circuit or failure

```
Analyze x and s; retrieve reference circuits and form instruction d_1
for r = 1 to K do
    Extract circuit keywords from d_r and encode them with CLIP
    Generate candidate topology τ_r using the conditioned RGB model
    Run Algorithm 1 with τ_r, s, and T
    Collect its status, returned parameters, and corresponding simulation diagnostics
    if Algorithm 1 reports success after independent final verification then
        Store its topology, parameters, and features for retrieval
        return the feasible circuit
    end if
    if r < K then
        Form feedback from topology observations and available metric residuals
        Use the LLM to construct d_{r+1} from the feedback and original targets s
    end if
end for
return failure
```

The outer-loop pseudocode specifies the exchange of design instructions and simulation feedback. The residuals are interpreted against the original metric bounds; changing an instruction does not relax those bounds. The description does not imply monotonic improvement across rounds or an independent learning update to the topology generator.

D Benchmark Construction and Difficulty Levels

OCB:Ckt-Bench-101. OCB:Ckt-Bench-101 is derived from the Open Circuit Benchmark (OCB) introduced in CktGNN (Dong et al., 2023), which is the first open-source dataset for analog circuit topology generation and parameter optimization. It consists of 10,000 operational amplifiers, where each circuit is represented as a graph and paired with performance specifications obtained from circuit simulations. We select 1,000 circuits from the full dataset to construct our evaluation set.

Each task pairs natural-language instructions with explicit quantitative constraints, represented by the specification vector s. Targets include gain, bandwidth, power, phase margin, and noise, together with additional conditions when provided. A design is successful only if it satisfies all specifications after parameter optimization, following the acceptance criterion in Section 2.4.

AMSBench-design. To further evaluate open-ended analog circuit design capability, we construct AMSBench-design as an extension of the circuit design benchmark released in AMSBench (Shi et al., 2025). Specifically, we align circuit topologies with their corresponding textual descriptions to support natural-language-driven design. For each circuit, we perform process-aware simulations using Ngspice under the SKY130 PDK, where device parameters are swept within predefined ranges to establish the mapping between circuit structures and performance metrics. Based on this process, we curate 1,200 valid circuit–performance pairs. All simulations are conducted on an Intel Xeon Gold 6330 platform (28 cores, 2.0 GHz), with a total runtime of 154 hours.

AMSBench-design evaluates design across diverse analog circuit categories and also includes parameter optimization. High-quality simulation data provide prior guidance for parameter optimization to obtain feasible implementations across these categories.

Difficulty Levels. To evaluate robustness under varying design complexity, tasks are categorized into three difficulty levels based on circuit difficulty: **Easy**, **Medium**, and **Hard**. For OCB:Ckt-Bench-101, the Easy, Medium, and Hard levels contain 300, 300, and 400 tasks, respectively. For AMSBench-design, the corresponding numbers are 400, 400, and 400.

E Topology Generator and Training Objective

Template encoding. The generator uses binary relation maps; Equation 1 specifies their 8-bit visualization. Let $\mathbf{a} \in \{0,1\}^H$ identify active nodes, and let C_{row} and C_{col} encode row and column coordinates in $[-1,1]$. The initial features are

$$\begin{aligned} X_0 &= \text{concat}(A_g, \text{diag}(\mathbf{a}), C_{\text{row}}, C_{\text{col}}), \\ h_0 &= \text{SiLU}(\text{GN}(\text{Conv}_{5\times5}(X_0))) + E_{\text{pos}}, \end{aligned} \tag{9}$$

where $X_0 \in \mathbb{R}^{4\times H\times H}$ and $E_{\text{pos}} \in \mathbb{R}^{C\times H\times H}$ is a learnable positional canvas. All convolutions preserve resolution. Each FiLM projection uses $W_\ell \in \mathbb{R}^{2C\times d_T}$ and $b_\ell \in \mathbb{R}^{2C}$ for C feature channels.

The data loader supplies each sample's Grid map and active-node set. Prediction is conditional on these inputs; the generator does not select the device inventory.

Reconstruction objective. For B circuits, let $A_{n,c,i,j}$ denote a binary target and $P_{n,c,i,j}$ its predicted probability. With positive-class weight w_+, the reconstruction loss is

$$\begin{aligned} \mathcal{L}_{\text{WBCE}} &= \frac{1}{2BH^2} \sum_{n=1}^{B} \sum_{c\in\{r,b\}} \sum_{i,j=1}^{H} \ell_{w_+}(A_{n,c,i,j}, P_{n,c,i,j}), \\ \ell_{w_+}(a,p) &= -w_+ a \log p - (1-a)\log(1-p). \end{aligned} \tag{10}$$

The implementation evaluates this objective directly from the symmetrized logits for numerical stability. It includes all entries of both predicted maps, including unused slots, and excludes Grid.

Structural regularizers. We penalize self-connections using the mean diagonal probability and define each node's soft degree by summing both generated relation types:

$$\mathcal{L}_{\text{diag}} = \frac{1}{2BH} \sum_{n=1}^{B} \sum_{c\in\{r,b\}} \sum_{i=1}^{H} P_{n,c,i,i}, \qquad d_{n,i} = \sum_{c\in\{r,b\}} \sum_{j=1}^{H} P_{n,c,i,j}. \tag{11}$$

For active-node indicators $a_{n,i}$, the soft-degree penalty is

$$\mathcal{L}_{\text{deg}} = \frac{\sum_{n=1}^{B}\sum_{i=1}^{H} a_{n,i} \max(0, \rho - d_{n,i})}{\max\left(1, \sum_{n=1}^{B}\sum_{i=1}^{H} a_{n,i}\right)}, \qquad \rho = 0.5. \tag{12}$$

F AI Use Statement

OpenAI ChatGPT was used to assist with language editing, translation, and LaTeX formatting during manuscript preparation. The use of LLMs for circuit design and dataset construction is described in the methodology and Appendix B. The authors retain full responsibility for the accuracy and integrity of the manuscript, including its technical content, references, and reported results.