SLIC: Reinforcement Fine-Tuning Small LMs for Multi-Turn Analog Circuit Optimization

Yan Xu[1], Zhengqi Gao[1], Ching-Yun Ko[3], Zhang-Wei Hong[3], Wenjie Lu[2]
Tao Yu[2], Xin Zhang[3], Duane S. Boning[1], Ruonan Han[1]

[1]MIT [2]Analog Devices, Inc. [3]IBM Research

Abstract

Analog integrated circuit (IC) design is a multi-turn, simulator-grounded engineering task that demands iterative reasoning over coupled performance trade-offs, yet proprietary process design kits (PDKs) make hosted large language model workflows difficult to deploy in practical design settings. We present **SLIC** – ***S**mall **L**M **I**terating on **C**ircuits* – a reinforcement fine-tuning framework that trains a 4B-scale, locally deployable language model for closed-loop analog circuit optimization. SLIC pairs structured circuit representations with discretized relative-update actions, domain- and success-gated multi-objective rewards, and Reward-Guided Rollout Steering (RGRS), which turns per-step simulator feedback into signals for both policy learning and online rollout construction in this sparse, highly constrained domain. We further release an open, goal-conditioned benchmark of 41 amplifier topologies with standardized protocols for in-distribution validation, held-out topology evaluation, and cross-technology-node transfer. Trained solely with open source 65 nm CMOS predictive technology models (PTM), SLIC-RANDOM achieves 87.1% in-distribution four-specification Pass@1 within eight turns, surpassing the 763B DeepSeek-V4.1-Flash THINK by 48.5 percentage points. Beyond the training distribution, SLIC-SOFTMAX reaches 72.6% Pass@1 on held-out topologies, and the trained policies consistently transfer across five unseen technology nodes without additional fine-tuning. This transfer extends to a commercial TSMC 65 nm CMOS PDK, where SLIC-RANDOM achieves 49.9% held-out-topology Pass@1, demonstrating practical deployment through local inference without transmitting proprietary data to external LLM services. Performance further scales with test-time simulator interactions, reaching up to 83.2% held-out-topology Pass@1 at a 32-turn budget, suggesting that SLIC enables the small policy to convert additional simulator calls into improved circuit designs. Project page: `https://hangroup.mit.edu/slic/`.

1 Introduction

Analog integrated circuits (ICs) [7, 49] sit at the interface between the physical world and modern computation. They perform high-fidelity conditioning and conversion of sensor signals [48], enable efficient power regulation in advanced integrated systems [23], support high-speed data conversion [4], and form the high-bandwidth, low-power I/O fabric of AI accelerators and wireless infrastructure [38, 57, 64]. Despite this importance, analog IC design remains highly manual, expert-driven, and tightly coupled to process technology, making design productivity difficult to scale across circuits, specifications, and technology nodes. As artificial intelligence (AI) advances rapidly, this long-standing scalability bottleneck has renewed interest in AI-driven analog circuit design automation.

However, translating this opportunity into practical analog IC design frameworks remains challenging, as several key issues must be addressed. First, any practical framework must **contend with severe**

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

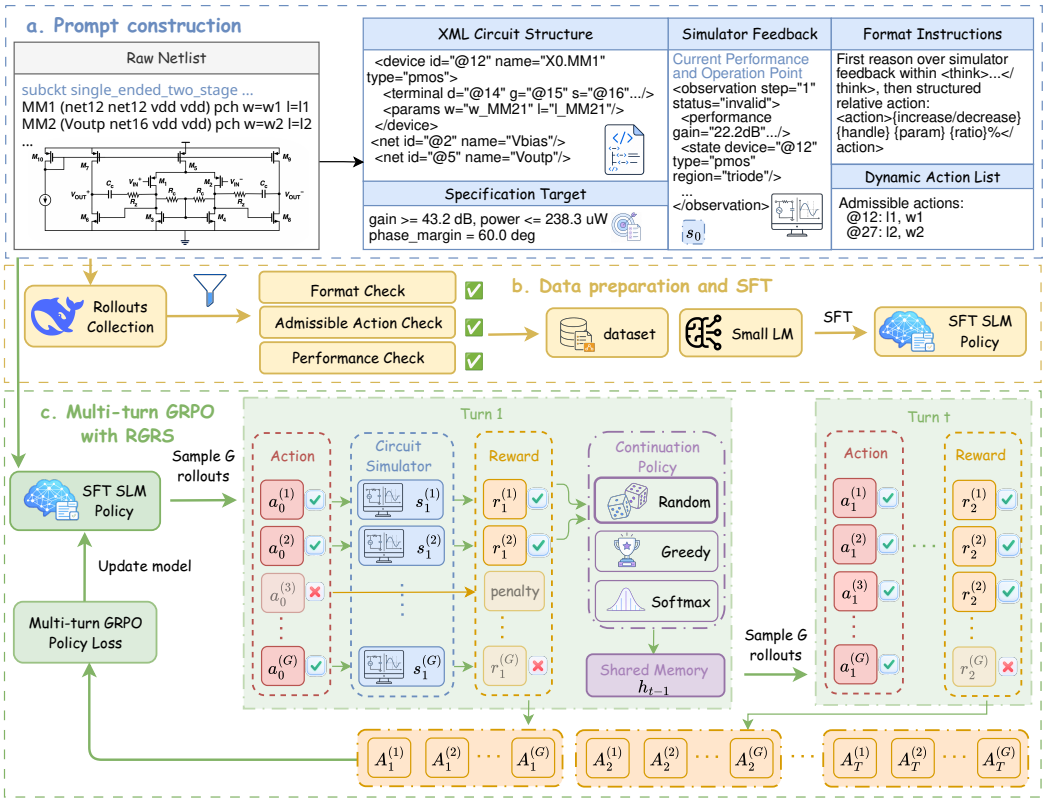


Figure 1: Overview of **SLIC**. Given a goal-conditioned analog circuit optimization task, SLIC builds a structured prompt from the circuit representation and target specifications, then trains a language-model agent through SFT and simulator-interactive RL. The upper path shows SFT data construction from filtered teacher rollouts. The lower loop shows multi-turn GRPO with Reward-Guided Rollout Steering (RGRS), where multiple candidate responses propose relative parameter updates that are evaluated through simulation, one eligible continuation is selected to advance the trajectory, and all candidates contribute to group-relative policy updates. Subscripts denote interaction steps, and superscripts index candidates within each group.

data scarcity and semiconductor technology fragmentation. Circuit performance and design decisions depend strongly on the underlying fabrication process, yet publicly available analog circuit designs are sparse due to proprietary protection and foundry confidentiality constraints. Many existing optimization [19, 22, 36] and learning-based approaches [6, 18, 51, 52, 62] remain specialized to particular circuit families or rely primarily on numerical search, limiting their ability to transfer across design settings. Therefore, a **reasoning-driven approach trained on synthetic, simulator-grounded data** is preferable: a model learns to navigate the design space rather than to memorize specific solutions, which in turn enables **generalization across circuit topologies and technology nodes**. When implemented using LLMs, **human-interpretable design behavior** is enabled, where proposed decisions and updates throughout the design process can be understood, inspected, and refined rather than treated as opaque black-box outputs. Second, **local deployability** is essential: exposing foundry process design kits (PDKs), device models, or proprietary circuit inventions to external LLM APIs is prohibited in most commercial and national-security-related settings [27, 29, 33]. Repeated reliance on external LLM APIs in agentic design workflows also incurs substantial token consumption and operational costs [2, 61]. Third, the framework should support **scalability and parallelism**, enabling multiple specialized agents to operate concurrently on different circuit sub-modules (e.g. amplifiers, mixers and comparators) and thereby improve system-level design efficiency. It should also remain composable within a unified framework that can scale to larger systems with additional specialized agents, so that updating one agent when new circuit-domain-specific data or methodologies become available does not disturb the rest of the framework.

In this paper, we introduce **SLIC** (Small LM Iterating on Circuits, as shown in Figure 1), a lightweight, circuit-specific and reasoning-driven small language model (SLM) training framework that addresses

the local deployability and scalability challenges above by performing multi-turn reinforcement learning with verifiable rewards (RLVR). RLVR has recently driven substantial progress in single-turn reasoning tasks such as mathematics and coding [8, 10, 12, 26, 35, 40, 41, 54, 59, 73, 77], and has been extended to multi-turn agentic settings [20, 30] where models act over multiple steps and adapt to environmental feedback [56, 71]. Such a paradigm shift from isolated answer generation toward stateful decision-making is a natural fit for analog circuit design: it is inherently multi-turn, requires reasoning over coupled trade-offs among performance objectives, and admits quantitatively verifiable feedback from circuit simulators at every step. Training small LMs under this paradigm reflects a practical trade-off between reasoning capability and parallel deployability. We focus on 4B-scale models as a balanced operating point: large enough to support structured reasoning and action generation, yet small enough to enable multiple specialized agents to run concurrently in local design environments. Our contributions are threefold:

1. **A reasoning-driven, locally deployable framework for analog design optimization.** We train SLIC on simulator-grounded synthetic data generated using public predictive technology models (PTM), addressing data scarcity without requiring proprietary PDK data. Trained solely using PTM 65 nm CMOS, the 4B SLIC-SOFTMAX policy maintains 61.9–71.0% Pass@1 on held-out topologies across five unseen technology nodes without additional fine-tuning, compared with 72.6% at the training node. Transfer to a commercial TSMC 65 nm CMOS PDK further demonstrates practical deployment through local inference and simulation without exposing proprietary process models or circuit data to external LLM services.
2. **Domain adaptations for simulator-interactive design optimization.** We combine domain- and success-gated rewards with Reward-Guided Rollout Steering (RGRS) to learn from intermediate simulator feedback. The rewards prioritize functional designs and target satisfaction, while RGRS guides rollout continuation and supports comparisons among candidate actions from the same interaction history. In our main setting, the 4B SLIC-RANDOM policy achieves 87.1% in-distribution four-specification Pass@1 within eight turns, outperforming the 763B DeepSeek-V4.1-Flash THINK baseline by 48.5 percentage points.
3. **Open simulator-interactive benchmark.** We release an open, goal-conditioned benchmark of 41 amplifier topologies paired with target specifications and initial states, with standardized protocols for in-distribution validation, held-out topology evaluation, and cross-technology-node transfer (22–130 nm CMOS), providing a reproducible testbed for closed-loop, multi-turn analog circuit design optimization.

2 Background and Problem Formulation

2.1 Multi-turn RL for Analog Circuit Optimization

We formulate analog circuit optimization as a multi-turn RLVR problem. At turn t, let $c_t \in \mathcal{C}$ denote the textual circuit representation and $f_t \in \mathcal{F}$ the simulator feedback, comprising measured performance metrics (e.g., gain, power) and device-level operating states (e.g., whether each transistor is in saturation or triode). The stepwise state is $s_t = (c_t, f_t)$. Let h_t contain the task specification q, the current state s_t, and the retained interaction history. The policy generates a variable-length sequence of circuit edit actions $a_t \sim \pi_\theta(\cdot \mid h_t)$ corresponding to a structured parameter update. The simulator-interactive environment $\mathcal{M}$ applies the update and returns $s_{t+1} = \mathcal{M}(s_t, a_t)$ together with a verifiable reward $r_t = \mathcal{R}(a_t, s_{t+1})$. An episode terminates when all target specifications are satisfied or when the maximum horizon is reached.

2.2 Multi-turn GRPO

Given a task instance (q, s_0), trajectory-level GRPO [54] samples a group of trajectories $\{\tau_i\}_{i=1}^{G}$ from the behavior policy $\pi_{\theta_{\text{old}}}$, where each trajectory $\tau_i = (s_{0,i}, a_{0,i}, r_{0,i}, \ldots, s_{T_i,i})$ is generated through simulator-interactive optimization. Let $o_i = (o_{i,1}, \ldots, o_{i,|o_i|})$ denote the model output sequence that induces trajectory τ_i through interaction with the environment. Each sampled trajectory τ_i receives a scalar return $R_i = R(\tau_i)$, defined by its terminal reward $R_i = r_{T_i-1,i}$ or by aggregating step rewards

as $R_i = \sum_{t=0}^{T_i-1} r_{t,i}$. GRPO normalizes these returns across the sampled trajectories:

$$\hat{A}_i = \frac{R_i - \mu_G}{\sigma_G + \epsilon}, \qquad \mu_G = \frac{1}{G}\sum_{i'=1}^{G} R_{i'}, \qquad \sigma_G = \mathrm{std}\big(\{R_{i'}\}_{i'=1}^{G}\big), \tag{1}$$

where $\epsilon > 0$ ensures numerical stability. The resulting advantage is shared across all turns and generated response tokens ℓ in the trajectory, so $\hat{A}_{i,\ell} = \hat{A}_i$. Turn-level variants retain the clipped policy objective below but construct a separate advantage for each turn.

The policy is then updated with a PPO-style clipped objective [50],

$$\begin{aligned}\mathcal{L}_{\mathrm{GRPO}}(\theta) =& \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{\ell=1}^{|o_i|}\min\Big(\rho_{i,\ell}(\theta)\hat{A}_{i,\ell},\ \mathrm{clip}(\rho_{i,\ell}(\theta), 1-\epsilon_{\mathrm{low}}, 1+\epsilon_{\mathrm{high}})\hat{A}_{i,\ell}\Big)\right] \\ & - \beta D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}}),\end{aligned} \tag{2}$$

where $\rho_{i,\ell}(\theta) = \pi_\theta(o_{i,\ell} \mid \cdot)/\pi_{\theta_{\mathrm{old}}}(o_{i,\ell} \mid \cdot)$ is the token-level importance ratio, with both policies conditioned on the same context and preceding tokens. We set $\beta = 0$ (Table 17). Following the Clip-Higher strategy in DAPO [73], we use an asymmetric clipping range with $\epsilon_{\mathrm{high}} > \epsilon_{\mathrm{low}}$, mildly relaxing the upper policy-ratio bound for positive-advantage tokens to preserve exploration during multi-turn training.

2.3 Challenges of Off-the-Shelf Multi-Turn GRPO for Analog Circuit Design Optimization

Although multi-turn GRPO provides a natural policy-optimization backbone for simulator-interactive circuit design optimization, applying it directly presents several challenges. First, analog circuit optimization involves tightly coupled performance requirements within a narrow feasible design space. Assessing progress requires considering both circuit validity and improvements toward satisfying all target specifications. Second, vanilla trajectory-level credit assignment can be ambiguous in multi-turn optimization. Since the final trajectory return is attached to multiple turns, invalid or unhelpful turns may inherit high rewards from trajectories that are later rescued, while locally beneficial turns may receive low scores when later turns fail. This motivates denser intermediate rewards that provide more direct feedback on individual updates. Finally, sampled groups can lack reward contrast, including when all candidates fail or all succeed. If their rewards are identical, group-relative advantages vanish. Maintaining informative reward differences within each group is therefore important for learning efficiently from sampled simulator feedback.

3 SLIC: Domain Adaptations for Simulator-Interactive Circuit Optimization

3.1 SLIC Framework

Figure 1 summarizes the SLIC pipeline. We collect training trajectories by running a strong prompted model (DeepSeek-V4.1-Flash) on simulator-grounded multi-turn optimization without specifying explicit goals, then relabel each trajectory with its best-observed performance tuple as the goal-conditioned training target, following the hindsight relabeling principle of HER [1]. This bypasses the lack of labeled (circuit, target-spec) pairs and yields a fully simulator-grounded synthetic training set. The dataset is filtered to retain valid and performance-improving turns for supervised fine-tuning (SFT), which initializes the policy with basic reasoning and action-generation behavior (Figure 1b). The SFT checkpoint is then further trained with multi-turn GRPO using the domain-gated rewards of Section 3.2 and the Reward-Guided Rollout Steering procedure of Section 3.3 (Figure 1c).

Each prompt contains the XML-structured circuit state, target specifications, simulator feedback, and a dynamically generated admissible action list (Figure 1a). The policy outputs reasoning and interpretable discretized relative-update actions rather than absolute parameter values. Full representation details, action grammar, and validity checks are provided in Appendix B.

3.2 Domain Reward Shaping

Analog circuit optimization requires satisfying coupled performance targets under physically meaningful operating conditions. Our reward design prioritizes valid amplification and joint specification satisfaction, while rewarding additional performance headroom among successful designs.

Domain-validity gates. Directly aggregating performance metrics can favor nonfunctional designs, such as circuits that consume little power because they fail to provide valid amplification. We therefore apply gates that reflect the physical prerequisites of amplifier operation before aggregating per-specification rewards. Non-positive gain suppresses power, bandwidth, and phase-margin credit. An in-band flatness gate additionally suppresses gain, bandwidth, and phase-margin credit when peaking exceeds 6 dB, preventing distorted frequency responses from receiving misleading performance credit. The same gates apply when checking specification satisfaction, with flatness used only to assess validity. Full definitions are provided in Appendix C.

Success-conditional slack bonus. To prioritize target satisfaction, we keep unsuccessful scores strictly below one and assign successful designs scores in $[1, 1.5]$, with a bonus based on mean normalized specification slack. A constant success reward would leave all-success groups without a performance learning signal. DAPO [73] addresses this through dynamic sampling, filtering out groups with uniform correctness outcomes. To obtain richer feedback from successful terminal candidates, we use the slack bonus to preserve differences in their measured specification headroom, allowing all-success groups to remain informative when their slack differs. The bonus also encourages additional headroom, providing margin against performance degradation and improving design robustness and policy generalization. The formulation and its tolerance alignment are detailed in Appendix C.

For policy updates, we normalize validity and performance feedback separately before combining their advantages, following the reward-decoupled normalization principle of GDPO [34]. The formulation is provided in Appendix C.4.

3.3 Reward-Guided Rollout Steering

Motivated by the credit-assignment challenges discussed in Section 2.3, we introduce **Reward-Guided Rollout Steering (RGRS)**, a search-augmented procedure for online rollout construction guided by intermediate simulator feedback. At each turn, RGRS samples G candidate responses from the same interaction history, each proposing an alternative set of parameter edits, and evaluates the resulting circuits through simulation. This procedure serves two roles. For policy learning, all sampled candidates remain available for group-relative updates, providing same-context comparisons for stepwise advantage estimation. For trajectory construction, RGRS selects one eligible successor as the parent state for the next turn, advancing the shared history. Continuation selection is restricted to valid candidates whenever any are available.

This design is motivated by the view of search as a policy-improvement operator. Best-of-G selection induces a reward-biased distribution over outputs, and prior work has analyzed the distributional shift and KL trade-offs of such best-of-N policies [3]. More broadly, expert-iteration and AlphaZero-style methods use search to produce improved behavior that is later internalized by a learned policy [11]. A related principle appears in automated discovery: TTT-Discover [76] selects promising parent states from previously evaluated candidates while retaining exploration through its reuse procedure. In our setting, RGRS applies candidate selection locally at each turn. We study three continuation policies to examine the trade-off between performance preference and exploration: RANDOM samples uniformly from eligible candidates, GREEDY selects a highest-scoring candidate, and SOFTMAX favors higher-scoring candidates while retaining nonzero selection probability for other eligible candidates. The full procedure is provided in Algorithm 1 in Appendix D.

4 Experiments

4.1 Setup

Dataset. We construct a benchmark of 41 amplifier topologies, using 35 for training and in-distribution (ID) evaluation and reserving six held-out topologies for out-of-distribution (OOD) evaluation. The dataset is further divided into Simple and Complex topology categories, each paired with multiple target goals and initial states. As shown in Table 15, the benchmark spans a wide range of topology complexity, tunable parameter counts, and valid/invalid initial-state ratios, providing a challenging testbed for multi-turn analog circuit optimization. The main experiments extend the original evaluation with a newer backbone, a larger benchmark, and an additional bandwidth objective. Training at 65 nm enables transfer evaluation toward both more advanced (22–45 nm) and

more mature (90–130 nm) technology nodes. Appendix G.1 reports this earlier configuration, which uses the Apache-2.0-licensed Qwen2.5-7B-Instruct [43, 69], 22 nm training, and three specifications (excluding bandwidth).

Simulation environment. Training tasks are simulated with ngspice [39] using the 65 nm bulk CMOS Predictive Technology Model (PTM) [37, 78]. In this setting, transistor sizing is constrained to channel length $L \in [45\,\mathrm{nm}, 100\,\mu\mathrm{m}]$ and width $W \in [45\,\mathrm{nm}, 500\,\mu\mathrm{m}]$. For cross-technology evaluation, each node uses its corresponding PTM models [37] and node-specific sizing bounds, as summarized in Table 5. We use Cadence Spectre [5] for evaluation with a commercial TSMC 65 nm CMOS foundry PDK.

Model training configuration. We initialize SFT from Apache-2.0-licensed Qwen3.5-4B [45, 46], choosing a 4B-scale model as a practical operating point for locally deployable specialized analog design agents under proprietary PDK constraints. RL is initialized from the resulting SFT checkpoint and optimized with GRPO using RGRS. Training uses an XML circuit representation and, unless another horizon is specified, a maximum rollout horizon of eight turns, taking approximately 39 hours on 2×B200 GPUs. Tables 16 and 17 summarize the SFT and RL configurations, respectively. We also train a variant of SLIC-RANDOM with dynamic filtering inspired by DAPO's dynamic sampling [73], which skips candidate groups with no group relative policy gradient signal. It is labeled "+ filter" in the figures and "+ dynamic filtering" in the tables. To assess applicability beyond Qwen3.5-4B, we also apply the same SFT-to-RL pipeline to the Apache-2.0-licensed Qwen3.5-9B [45, 47] and Qwen3-8B [44, 70] using SLIC-RANDOM. Results are reported in Table 4.

Baselines. We compare our method with several complementary baselines. We include Bayesian Optimization with TuRBO [19] as a representative black-box numerical optimizer, the Qwen3.5-4B model to evaluate the zero-shot capability of an untuned locally deployable LLM, and the supervised fine-tuned (SFT) model as an initialization without reinforcement learning. We also evaluate DeepSeek-V4.1-Flash [13, 14], a recent large open-weight model, in THINK mode as a strong prompt-based reasoning baseline. To compare SLIC with a strong closed-source model, we evaluate GPT-6-Astra [42] on the same tasks, with one rollout per task and up to eight interaction turns. Evaluation across both splits costs approximately $276 in API charges. We also include two GRPO baselines. Single-turn GRPO uses SLIC's reward design, with the effect of training horizon analyzed in Section 4.4. For multi-turn training, trajectory-level GRPO with only terminal supervision was unstable in our experiments. We therefore evaluate the branching-rollout formulation of multi-turn GRPO (MT-GRPO) [65], referred to here as Tree MT-GRPO, which combines intermediate and terminal feedback. Its budget of candidate evaluations is approximately matched to SLIC-RANDOM. Implementation details are provided in Appendix F.1.

Metrics. We evaluate four specifications: gain, power consumption, $-3\,\mathrm{dB}$ bandwidth, and phase margin. A rollout succeeds if it reaches a valid design satisfying all four target specifications within the prescribed 5% tolerance under the domain-validity gates. At a turn budget T, Pass@k measures the probability that at least one of k sampled rollouts succeeds within T interaction turns, averaged over evaluation tasks. We primarily report cumulative four-specification Pass@1 at Turns 4 and 8. Unless otherwise noted, we report means $\pm$ standard deviations over three evaluation runs.

4.2 Evaluation Results

In-distribution evaluation. Table 1 shows that all 4B SLIC variants substantially outperform SFT and DeepSeek-V4.1-Flash THINK at both evaluation horizons. At four turns, SLIC-RANDOM achieves 74.9% Pass@1, exceeding GPT-6-Astra's 67.1%. At eight turns, Astra achieves higher success at 90.6%, while the locally deployable 4B SLIC-RANDOM policy reaches 87.1%, a gap of 3.5 percentage points. Among SLIC variants, SOFTMAX performs best on simple topologies, while RANDOM leads on complex topologies and overall. Although DeepSeek-V4.1-Flash also generated the SFT data, RL lifts the student from below its teacher (26.5% vs. 38.6%) to well above it (87.1%).

Generalization to held-out topologies. Table 2 shows that SLIC generalizes to held-out topologies. At eight turns, SLIC-SOFTMAX achieves the highest Pass@1 among SLIC variants, exceeding DeepSeek by 45.9 percentage points. GPT-6-Astra achieves 86.2% on the same held-out tasks, with its advantage concentrated in structurally simple topologies, which can still require substantial

Table 1: In-distribution evaluation at the PTM 65 nm technology node over 840 tasks with breakdowns by topology complexity. GPT-6-Astra is evaluated once on the full evaluation set. Bayesian Optimization uses 96 and 192 simulator calls per topology to match SLIC's total simulation budget. **Bold** and underline denote the best and second-best mean results in each column respectively.

	4-spec Pass@1 (%) @Turn 4			4-spec Pass@1 (%) @Turn 8		
Method	**Simple**	**Complex**	**Total**	**Simple**	**Complex**	**Total**
Bayesian Optimization	54.3 ± 7.2	34.2 ± 1.9	43.4 ± 3.9	70.5 ± 7.5	54.1 ± 5.3	61.6 ± 6.1
GPT-6-Astra	77.6	58.3	67.1	**96.1**	**86.0**	**90.6**
DeepSeek-V4.1-Flash THINK (763B)	33.1 ± 0.0	11.1 ± 1.8	21.2 ± 1.0	56.0 ± 1.6	24.0 ± 1.1	38.6 ± 1.3
Qwen3.5-4B	2.4 ± 0.6	1.0 ± 0.3	1.7 ± 0.2	3.2 ± 1.1	1.5 ± 0.4	2.3 ± 0.5
SFT	22.4 ± 1.2	6.1 ± 0.1	13.5 ± 0.6	41.0 ± 1.8	14.4 ± 0.7	26.5 ± 0.7
Single-turn GRPO	66.4 ± 2.0	41.7 ± 0.8	53.0 ± 1.0	74.3 ± 0.8	51.0 ± 1.9	61.6 ± 1.4
Tree MT-GRPO	73.2 ± 1.2	37.8 ± 0.9	54.0 ± 0.8	89.7 ± 1.1	55.6 ± 0.8	71.2 ± 0.6
Proposed (Qwen3.5-4B)						
SLIC-GREEDY	72.4 ± 1.3	36.6 ± 1.5	53.0 ± 1.2	84.1 ± 0.9	51.6 ± 1.1	66.5 ± 0.8
SLIC-RANDOM	$\underline{86.8 \pm 0.8}$	$\mathbf{64.8 \pm 0.7}$	$\mathbf{74.9 \pm 0.0}$	92.7 ± 1.2	$\underline{82.5 \pm 1.0}$	$\underline{87.1 \pm 0.7}$
w/o domain-validity gates	82.2 ± 0.8	57.2 ± 0.1	68.6 ± 0.4	89.1 ± 0.5	77.6 ± 2.3	82.8 ± 1.2
SLIC-SOFTMAX	$\mathbf{87.7 \pm 0.3}$	$\underline{59.8 \pm 1.5}$	$\underline{72.5 \pm 0.7}$	$\underline{93.9 \pm 0.8}$	76.2 ± 1.5	84.3 ± 1.0

performance improvements to meet their targets. Both methods achieve 82.5% on complex topologies. Softmax's advantage over Random is not statistically significant (Appendix H.5), although both outperform Greedy, suggesting a benefit from preserving diversity during rollout construction for topology generalization. Rollout analysis associates Softmax with more consistent recovery and planning after performance decreases, while Greedy shows more hesitation and repeated reconsideration despite longer responses (Appendix H.1, Figure 6). Required performance improvements also help explain differences across topologies beyond circuit size alone (Appendix H.3).

Comparison with Tree MT-GRPO. Within eight evaluation turns, SLIC-RANDOM achieves higher Pass@1 than Tree MT-GRPO on both ID tasks (87.1% vs. 71.2%) and held-out topologies (70.6% vs. 55.1%) (Tables 1 and 2). In circuit optimization, each action jointly modifies multiple parameters, making it important to associate the resulting performance with the corresponding update. This association is less direct in Tree MT-GRPO, where aggregating advantages across subsequent turns mixes feedback on the current joint update with the effects of later edits. Moreover, Tree MT-GRPO exhibits a sharp increase in policy entropy during training. Its branching structure also makes sampling cost grow exponentially with the training horizon, limiting the depth of training rollouts. SLIC therefore uses RGRS to compare candidates from the same interaction history and learns from stepwise simulator feedback. This supports effective iterative circuit optimization by keeping credit assignment focused on local refinement, while advancing one selected successor allows larger candidate groups and longer training rollouts with linear sampling cost.

Technology node generalization. Figure 2 shows that policies trained at PTM 65 nm can transfer to both older and more advanced technology nodes without additional fine-tuning. Across the five unseen technology nodes, SLIC-SOFTMAX maintains 73.3–79.8% Pass@1 on ID tasks and 61.9–71.0% on held-out topologies, compared with 83.5% and 72.6%, respectively, at the training node. These results suggest that SLIC internalizes simulator-grounded optimization strategies that remain useful beyond the training node, supporting our hypothesis that some learned strategies are agnostic to the technology node. Valid parameter ranges for each node are reported in Table 5.

To assess effectiveness in a practical circuit design setting, we also evaluate SLIC on a commercial TSMC 65 nm CMOS foundry PDK without additional fine-tuning. SLIC-RANDOM achieves 55.8% ID and 49.9% OOD Pass@1 within eight turns, down from 86.9% and 70.6%, respectively, on the same tasks with PTM 65 nm. The foundry PDK captures process-specific effects that PTM simplifies. Differences in device behavior can therefore shift the trade-offs achievable with a given topology, making the four targets selected under PTM harder to satisfy jointly. This mismatch may help explain the observed drop in Pass@1 after transfer. Our task-level audit shows that SFT mainly succeeds when the initial design needs only minor adjustments to meet the targets, which may explain its similar Pass@1 across the two processes. On the foundry PDK, SLIC-RANDOM still outperforms SFT by about 29 percentage points on both splits. These results demonstrate the practical value of

Table 2: Out-of-distribution evaluation at the PTM 65 nm technology node over 240 tasks, with breakdowns by topology complexity (two simple and four complex). GPT-6-Astra is evaluated once on the full evaluation set. Bayesian Optimization uses 96 and 192 simulator calls per topology to match SLIC's total simulation budget. **Bold** and underline denote the best and second-best reported mean results in each column respectively.

	4-spec Pass@1 (%) @Turn 4			4-spec Pass@1 (%) @Turn 8		
Method	**Simple**	**Complex**	**Total**	**Simple**	**Complex**	**Total**
Bayesian Optimization	2.5 ± 0.0	36.7 ± 1.0	25.3 ± 0.6	3.3 ± 1.4	46.0 ± 1.4	31.8 ± 0.5
GPT-6-Astra	**66.2**	62.5	**63.7**	**93.8**	**82.5**	**86.2**
DeepSeek-V4.1-Flash THINK (763B)	10.8 ± 3.8	16.9 ± 1.1	14.9 ± 0.6	20.0 ± 3.7	30.0 ± 2.3	26.7 ± 2.2
Qwen3.5-4B	3.3 ± 0.7	3.8 ± 0.6	3.6 ± 0.5	3.3 ± 0.7	5.0 ± 1.1	4.4 ± 0.9
SFT	3.3 ± 3.1	14.8 ± 1.3	11.0 ± 0.2	10.4 ± 1.4	24.6 ± 1.3	19.9 ± 1.3
Single-turn GRPO	31.2 ± 2.5	41.2 ± 2.9	37.9 ± 2.7	49.2 ± 0.7	46.9 ± 1.7	47.6 ± 0.9
Tree MT-GRPO	25.8 ± 3.8	48.1 ± 0.6	40.7 ± 0.9	35.8 ± 1.4	64.8 ± 1.3	55.1 ± 0.6
Proposed (Qwen3.5-4B)						
SLIC-GREEDY	22.5 ± 2.5	36.2 ± 2.9	31.7 ± 1.8	32.1 ± 1.9	49.8 ± 2.9	43.9 ± 2.4
SLIC-RANDOM	35.0 ± 1.2	$\mathbf{67.7 \pm 2.2}$	56.8 ± 1.3	46.7 ± 1.4	$\mathbf{82.5 \pm 2.3}$	70.6 ± 1.2
w/o domain-validity gates	31.2 ± 4.5	55.2 ± 2.2	47.2 ± 2.3	46.2 ± 2.2	72.3 ± 1.9	63.6 ± 0.9
SLIC-SOFTMAX	36.7 ± 4.0	67.1 ± 3.0	56.9 ± 3.1	52.9 ± 4.7	$\mathbf{82.5 \pm 0.6}$	72.6 ± 2.0

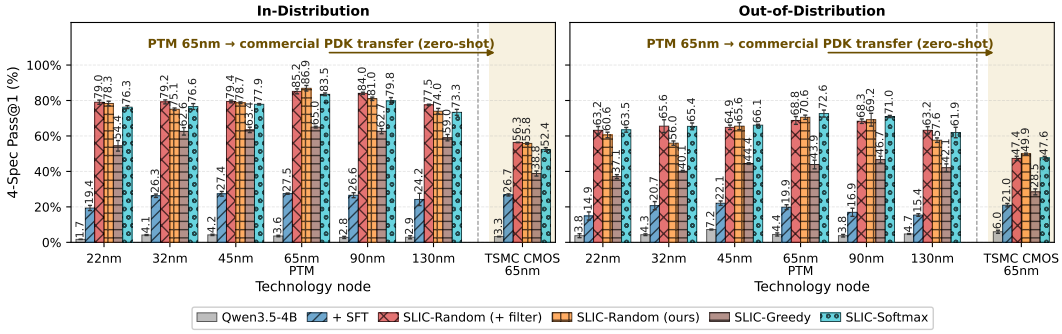


Figure 2: Technology transfer under four-specification Pass@1 evaluation within eight turns. Policies are trained on PTM 65 nm and evaluated across PTM nodes from 22 to 130 nm and a TSMC CMOS 65 nm PDK without additional fine-tuning. Initial parameters are adjusted to the valid ranges of each technology, while the PTM 65 nm target specifications are reused. Left: a 420-task subset of the full 840 in-distribution tasks. Right: 240 held-out topology tasks.

local policy inference and simulation, which keep proprietary device models and circuit data in-house while enabling automated circuit optimization in workflows where PDK confidentiality prevents the use of hosted LLM services.

4.3 Component Ablations

Domain-validity guidance. Removing domain-validity reward gates from SLIC-RANDOM reduces Turn 8 Pass@1 by 4.3 percentage points in-distribution and 7.0 points out-of-distribution (Tables 1 and 2), which supports our inductive bias that bringing a circuit into a feasible operating region before further optimization is beneficial.

Success-conditional slack bonus. Table 3 reports four-specification Pass@1 at Turn 8. With dynamic filtering enabled, adding the slack bonus improves Pass@1 by 8.5 percentage points on ID tasks and 12.1 points on held-out topologies. The larger gain on held-out topologies supports the benefit of this richer feedback for generalization.

Table 3: Slack bonus ablation with random continuation and dynamic filtering.

Slack bonus	ID	OOD
Without	76.4 ± 0.9	56.7 ± 1.3
With	84.9 ± 0.7	68.8 ± 2.2

Table 4: Generalization across backbones at Turn 8 on 840 ID and 240 OOD tasks. Qwen3 and Qwen3.5 backbones follow the main SLIC-RANDOM configuration. †Qwen2.5-7B-Instruct uses SLIC-SOFTMAX with 800 RL updates (Appendix G.2).

	In-distribution Pass@1 (%)			Out-of-distribution Pass@1 (%)		
Backbone	Base	SFT	SLIC	Base	SFT	SLIC
Qwen2.5-7B-Instruct†	1.0 ± 0.2	14.0 ± 0.7	32.2 ± 0.8	2.8 ± 0.6	13.3 ± 1.1	24.9 ± 1.7
Qwen3-8B	1.4 ± 0.0	13.0 ± 0.9	39.8 ± 0.5	3.2 ± 0.2	11.4 ± 1.3	29.0 ± 1.2
Qwen3.5-4B	2.3 ± 0.5	26.5 ± 0.7	87.1 ± 0.7	4.4 ± 0.9	19.9 ± 1.3	70.6 ± 1.2
Qwen3.5-9B	3.0 ± 0.2	27.1 ± 2.0	88.9 ± 0.4	4.2 ± 0.7	16.9 ± 1.0	76.1 ± 1.3

Additional ablations in Appendix E examine advantage normalization, rollout construction, episode termination, training horizon, and history representation and length.

4.4 Backbone Generalization and Test-Time Scaling

Generalization across backbones. Table 4 shows that SLIC improves over SFT across all four evaluated backbones. Within the Qwen3.5 family, scaling from 4B to 9B improves Pass@1 by 1.8 percentage points on ID tasks and 5.5 points on held-out topologies, suggesting additional benefits from scaling model capacity.

Pass@1 over interaction turns. Figure 3 shows that additional test-time simulator interactions improve cumulative Pass@1 on both splits, with the largest gains in early turns and diminishing returns at longer horizons. This suggests that much of the benefit can be captured within a moderate interaction budget, motivating us to examine shorter training horizons. Training with four turns remains effective and allows continued improvement beyond the training horizon. Further shortening the training horizon to single-turn GRPO lowers Pass@1 (Tables 1 and 2) and leads to earlier saturation, particularly on held-out topologies. This is consistent with its lack of training experience in successive simulator-guided refinement. These results suggest that a moderate multi-turn training budget may reduce training compute while preserving test-time scaling (Appendix E).

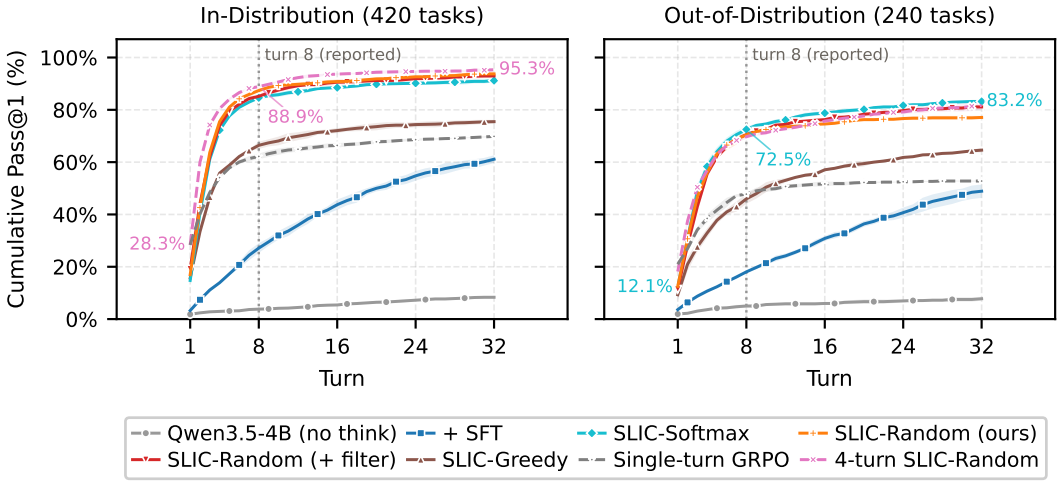


Figure 3: Test-time scaling of cumulative Pass@1 with up to 32 interaction turns on a 420-task subset of the full 840 ID tasks (left) and 240 OOD tasks (right). Qwen3.5-4B is evaluated in non-thinking mode; we also tested thinking mode with an 8k-token generation budget, but reasoning did not terminate within this limit. Numerical annotations highlight selected values along the best-performing curve in each panel.

Pass@k over k. Increasing the number of test-time rollouts raises Pass@k toward saturation across the evaluated methods, although SLIC's higher first-attempt success leaves less room for improvement (Figure 4). SFT attains substantially higher success rates with a large sampling budget, suggesting that successful optimization strategies are already accessible to the policy. SLIC-RANDOM

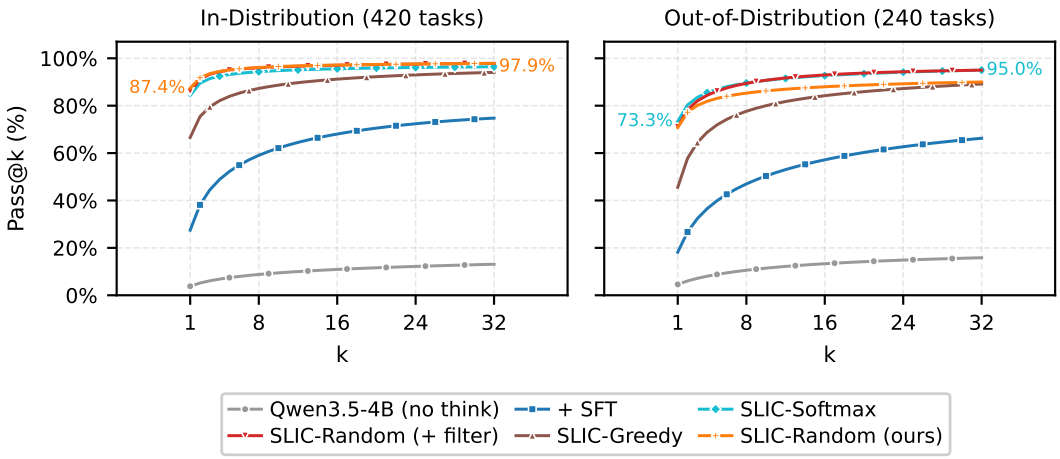


Figure 4: Test-time scaling of Pass@k with the number of sampled rollouts on a 420-task subset of the full 840 ID tasks (left) and 240 OOD tasks (right) with eight-turn simulator budget per rollout. Numerical annotations highlight selected values along the best-performing curve in each panel.

exceeds SFT's Pass@32 with a single rollout on both splits, consistent with reinforcement fine-tuning concentrating probability mass on effective optimization behaviors [75] and reducing the need for repeated sampling.

5 Conclusion and Future Work

We present SLIC, a simulator-interactive RLVR framework that formulates analog circuit optimization as a multi-turn language-model decision problem grounded in verifiable simulator feedback. With structured circuit representations, interpretable relative-update actions, domain-aware reward shaping, and reward-guided rollout steering, a 4B policy reaches 87.1% in-distribution Pass@1 within eight turns, 3.5 percentage points below GPT-6-Astra, and exceeds GPT-6-Astra at four turns. The trained policies transfer to held-out topologies, five unseen PTM nodes, and a commercial TSMC 65 nm PDK without additional fine-tuning, while running entirely with local inference. These results position simulator-grounded RLVR on small models as a practical path toward locally deployable analog circuit design automation.

Future work will extend this formulation toward the multi-agent analog design framework envisioned in the introduction. Rather than relying on a single design-optimization agent, future systems may coordinate multiple specialized agents for topology exploration, block-level optimization, verification, layout-aware refinement, and system-level specification decomposition. This direction better matches the modular and hierarchical nature of analog circuit design, but also raises new challenges in scaling to larger analog and mixed-signal systems, including long-context management, cross-agent coordination, and maintaining consistent design state across hierarchical blocks. Addressing these issues could enable scalable collaboration among lightweight, locally deployable agents for more complex circuit and system-level design workflows.

Additional experiments, implementation details, limitations, and broader impact discussions are provided in the Appendix.

Acknowledgments

This work was supported primarily by Analog Devices (ADI), with additional funding from the MIT Cadence Fellowship and the JLO/MACOM Endowment Fund. We thank the MIT Center for Integrated Circuits and Systems (CICS) for its support. We also acknowledge IBM, the MIT Office of Research Computing and Data (ORCD), and the Artificial Intelligence Compute Resources (AICR) cluster for providing computing and storage resources.

References

[1] Marcin Andrychowicz, Filip Wolski, Alex Ray, et al. Hindsight experience replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[2] Longju Bai, Zhemin Huang, Xingyao Wang, et al. How do AI agents spend your money? Analyzing and predicting token consumption in agentic coding tasks. *arXiv preprint arXiv:2604.22750*, 2026. doi: 10.48550/arXiv.2604.22750.

[3] Ahmad Beirami, Alekh Agarwal, Jonathan Berant, et al. Theoretical guarantees on the best-of-n alignment policy. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 3580–3602. PMLR, 2025. URL https://proceedings.mlr.press/v267/beirami25a.html.

[4] Chakravarti Bheemisetti, Karunanidhan Pandey, Idan Lotan, et al. A 7-bit 1.75-GS/s 6.9-fJ/conv.-step FoM_w loop-unrolled fully asynchronous SAR ADC in 3-nm CMOS for a 224-Gb/s SerDes receiver. *IEEE Journal of Solid-State Circuits*, 60(4):1486–1499, 2025. doi: 10.1109/JSSC.2024.3449115.

[5] Cadence Design Systems. Cadence Spectre circuit simulator. Computer software, 2026. URL https://www.cadence.com/en_US/home/tools/custom-ic-analog-rf-design/circuit-simulation.html. Commercial software used under institutional license.

[6] Weidong Cao, Mouhacine Benosman, Xuan Zhang, et al. Domain knowledge-based automated analog circuit design with deep reinforcement learning. In *1st Annual AAAI Workshop on AI to Accelerate Science and Engineering (AI2ASE)*, 2022. URL https://arxiv.org/abs/2202.13185.

[7] Tony Chan Carusone, David A Johns, and Kenneth W Martin. *Analog integrated circuit design*. John Wiley & Sons, 2nd edition, 2011.

[8] Guoxin Chen, Minpeng Liao, Chengxi Li, et al. Step-level value preference optimization for mathematical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP*, 2024.

[9] Xingyu Chen, Jiahao Xu, Tian Liang, et al. Do NOT think that much for 2+3=? On the overthinking of long reasoning models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 9487–9499. PMLR, 2025. URL https://proceedings.mlr.press/v267/chen25bx.html.

[10] Ganqu Cui, Lifan Yuan, Zefan Wang, et al. Process reinforcement through implicit rewards. 2025. URL https://arxiv.org/abs/2502.01456v2.

[11] Ivo Danihelka, Arthur Guez, Julian Schrittwieser, et al. Policy improvement by planning with Gumbel. In *International Conference on Learning Representations*, 2022.

[12] DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948v1.

[13] DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression, 2026. URL https://arxiv.org/abs/2609.19969.

[14] DeepSeek-AI. DeepSeek-V4.1-Flash. Hugging Face model card, 2026. URL https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash.

[15] DeepSeek-AI. DeepSeek-V4-Pro model card. Hugging Face, 2026. URL https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro.

[16] DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

[17] Yifeng Ding, Hung Le, Songyang Han, et al. Empowering multi-turn tool-integrated agentic reasoning with group turn policy optimization. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 42409–42423, 2026. URL https://aclanthology.org/2026.acl-long.1962/.

[18] Zehao Dong, Weidong Cao, Muhan Zhang, et al. CktGNN: Circuit graph neural network for electronic design automation. In *International Conference on Learning Representations (ICLR)*, 2023.

[19] David Eriksson, Michael Pearce, Jacob Gardner, et al. Scalable global optimization via local Bayesian optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[20] Lang Feng, Zhenghai Xue, Tingcong Liu, et al. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/420c9f777c0b4f78d515e53cf74d58b2-Abstract-Conference.html.

[21] Zhengqi Gao and Duane S Boning. A review of Bayesian methods in electronic design automation. *arXiv preprint arXiv:2304.09723*, 2023.

[22] Zhengqi Gao, Jun Tao, Fan Yang, et al. Efficient performance trade-off modeling for analog circuit based on Bayesian neural network. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019.

[23] Rinkle Jain, Shunjiang Xu, Rajiv Kaushal, et al. 28.6 An 87% efficient 2V-input, 200A voltage regulator chiplet enabling vertical power delivery in multi-kW systems-on-package. In *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 466–468, 2024. doi: 10.1109/ISSCC49657.2024.10454349. URL https://doi.org/10.1109/ISSCC49657.2024.10454349.

[24] Haoning Jiang, Han Wu, Zhuoli Ouyang, et al. FD-MAGRPO: Functionality-driven multi-agent group relative policy optimization for analog-LDO sizing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(27):22310–22317, 2026. doi: 10.1609/aaai.v40i27.39388. URL https://ojs.aaai.org/index.php/AAAI/article/view/39388.

[25] Lingyu Jiang, Zirui Li, Lingyu Xu, et al. PathCal: State-aware reflection-marker calibration for efficient reasoning, 2026. URL https://arxiv.org/abs/2605.23074.

[26] Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, et al. VinePPO: Refining credit assignment in RL training of LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 29557–29590. PMLR, 2025. URL https://proceedings.mlr.press/v267/kazemnejad25a.html.

[27] Dimple Vijay Kochar, Hanrui Wang, Anantha P. Chandrakasan, et al. LEDRO: LLM-enhanced design space reduction and optimization for analog circuits. In *2025 IEEE International Conference on LLM-Aided Design (ICLAD)*, 2025. doi: 10.1109/ICLAD65226.2025.00011. URL https://doi.org/10.1109/ICLAD65226.2025.00011.

[28] Kishor Kunal, Tonmoy Dhar, Meghna Madhusudan, et al. GANA: Graph convolutional network based automated netlist annotation for analog circuits. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 55–60, 2020. doi: 10.23919/DATE48585.2020.9116329.

[29] Yao Lai, Sungyoung Lee, Guojin Chen, et al. AnalogCoder: Analog circuit design via training-free code generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(1):379–387, 2025. doi: 10.1609/aaai.v39i1.32016. URL https://ojs.aaai.org/index.php/AAAI/article/view/32016.

[30] Junbo Li, Peng Zhou, Rui Meng, et al. Turn-PPO: Turn-level advantage estimation with PPO for improved multi-turn RL in agentic LLMs. *arXiv preprint arXiv:2512.17008*, 2025. URL https://arxiv.org/abs/2512.17008.

[31] Peiji Li, Linyang Li, Handa Sun, et al. TL-GRPO: Turn-level RL for reasoning-guided iterative optimization. *arXiv preprint arXiv:2601.16480*, 2026. URL https://arxiv.org/abs/2601.16480.

[32] Bo Liu, Qingfu Zhang, and Georges G. E. Gielen. A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems. *IEEE Transactions on Evolutionary Computation*, 18(2):180–192, 2014. doi: 10.1109/TEVC.2013.2248012. URL https://doi.org/10.1109/TEVC.2013.2248012.

[33] Chang Liu and Danial Chitnis. EEsizer: LLM-based AI agent for sizing of analog and mixed signal circuit. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2026. doi: 10.1109/TCSI.2025.3646359. URL https://doi.org/10.1109/TCSI.2025.3646359.

[34] Shih-Yang Liu, Xin Dong, Ximing Lu, et al. GDPO: Group reward-decoupled normalization policy optimization for multi-reward RL optimization. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 75978–76000. PMLR, 2026. URL https://proceedings.mlr.press/v306/liu26aa.html.

[35] Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, et al. ReFT: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7601–7614, 2024. doi: 10.18653/v1/2024.acl-long.410. URL https://aclanthology.org/2024.acl-long.410/.

[36] Wenlong Lyu, Fan Yang, Changhao Yan, et al. Batch Bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design. In *International Conference on Machine Learning (ICML)*, 2018.

[37] Microelectronics Co-design Research Group, University of Minnesota. Predictive Technology Model (PTM). https://mec.umn.edu/ptm, 2026. Accessed: 2026-05-07; PTM transistor model files.

[38] Susnata Mondal, Junyi Qiu, Sashank Krishnamurthy, et al. 18.2 A 4×64Gb/s NRZ 1.3pJ/b co-packaged and fiber-terminated 4-ch VCSEL-based optical transmitter. In *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 340–342, 2024. doi: 10.1109/ISSCC49657.2024.10454455.

[39] ngspice Developers. ngspice: Open-source SPICE circuit simulator. Computer software, n.d. URL https://ngspice.sourceforge.io/. Accessed: 2026-11-02.

[40] OpenAI. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*, 2024. URL https://arxiv.org/abs/2412.16720v1.

[41] OpenAI. Learning to reason with LLMs, September 2024. URL https://openai.com/index/learning-to-reason-with-llms/. Accessed: 2025-05-05.

[42] OpenAI. GPT-6 Astra. Model documentation, 2026. URL https://developers.openai.com/api/docs/models/gpt-6-astra.

[43] Qwen Team. Qwen2.5-7B-Instruct. https://huggingface.co/Qwen/Qwen2.5-7B-Instruct, 2024. Model card; Apache-2.0 license.

[44] Qwen Team. Qwen3-8B. Hugging Face model card, 2025. URL https://huggingface.co/Qwen/Qwen3-8B.

[45] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[46] Qwen Team. Qwen3.5-4B. Hugging Face model card, 2026. URL https://huggingface.co/Qwen/Qwen3.5-4B.

[47] Qwen Team. Qwen3.5-9B. Hugging Face model card, 2026. URL https://huggingface.co/Qwen/Qwen3.5-9B.

[48] Dakey Sudhir Nageswara Rao, Udara Yedukondalu, and N. Balaji. A comprehensive survey on low power analog front end circuits with digital converters for biomedical sensor application. *Analog Integrated Circuits and Signal Processing*, 126:11, 2026. doi: 10.1007/s10470-025-02531-9. URL https://doi.org/10.1007/s10470-025-02531-9.

[49] Behzad Razavi. *Design of analog CMOS integrated circuits*. McGraw-Hill Education, 2nd edition, 2017.

[50] John Schulman, Filip Wolski, Prafulla Dhariwal, et al. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[51] Keertana Settaluri, Ameer Haj-Ali, Qijing Huang, et al. AutoCkt: Deep reinforcement learning of analog circuit designs. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 490–495, 2020. doi: 10.23919/DATE48585.2020.9116200. URL https://ieeexplore.ieee.org/document/9116200/.

[52] Aditya Shahane, Saripilli Swapna Manjiri, Ankesh Jain, et al. Graph of circuits with GNN for exploring the optimal design space. In *Advances in Neural Information Processing Systems*, volume 36, pages 6014–6025, 2023. doi: 10.52202/075280-0263. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/12da92b7c64176eb6eb6ad0ae31554fd-Paper-Conference.pdf.

[53] Bobak Shahriari, Kevin Swersky, Ziyu Wang, et al. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016. doi: 10.1109/JPROC.2015.2494218. URL https://doi.org/10.1109/JPROC.2015.2494218.

[54] Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

[55] Jinyi Shen, Zihao Chen, Ji Zhuang, et al. 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. URL https://doi.org/10.1109/tcad.2025.3573228.

[56] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, et al. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

[57] Alan Smith, Eric Chapman, Chintan Patel, et al. 11.1 AMD Instinct™ MI300 series modular chiplet package – HPC and AI accelerator for exa-class systems. In *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, volume 67, pages 490–492, 2024. doi: 10.1109/ISSCC49657.2024.10454441.

[58] Michael Sullivan and Alexander Koller. GRPO is secretly a process reward model. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 116364–116379. PMLR, 2026. URL https://proceedings.mlr.press/v306/sullivan26a.html.

[59] Jonathan Uesato, Nate Kushman, Ramana Kumar, et al. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

[60] Prashanth Vijayaraghavan, Luyao Shi, Ehsan Degan, et al. AUTOCIRCUIT-RL: Reinforcement learning-driven LLM for automated circuit topology generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 61498–61512, 2025. URL https://proceedings.mlr.press/v267/vijayaraghavan25a.html.

[61] Noppanat Wadlom, Junyi Shen, and Yao Lu. Efficient LLM serving for agentic workflows: A data systems perspective. *Proceedings of the ACM on Management of Data*, 4(3):1–29, 2026. doi: 10.1145/3802046.

[62] Hanrui Wang, Kuan Wang, Jiacheng Yang, et al. GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning. In *Proceedings of the 57th ACM/IEEE Design Automation Conference (DAC)*, 2020.

[63] Ruiyi Wang and Prithviraj Ammanabrolu. A practitioner's guide to multi-turn agentic reinforcement learning. *arXiv preprint arXiv:2510.01132*, 2025. URL https://arxiv.org/abs/2510.01132.

[64] Zisong Wang, Huan Wang, Youssef O. Hassan, et al. A bits-to-antenna F-band 120-Gb/s CMOS RF-64QAM transmitter for FutureG wireless links. *IEEE Journal of Solid-State Circuits*, 60(8):2694–2710, 2025. doi: 10.1109/JSSC.2024.3523842.

[65] Quan Wei, Siliang Zeng, Chenliang Li, et al. Reinforcing multi-turn reasoning in LLM agents via fine-grained reward structure and credit assignment. *arXiv preprint arXiv:2505.11821v3*, 2026. URL https://arxiv.org/abs/2505.11821v3.

[66] Yutao Xie, Nathaniel Thomas, Nicklas Hansen, et al. TIPS: Turn-level information-potential reward shaping for search-augmented LLMs. *arXiv preprint arXiv:2603.22293*, 2026. URL https://arxiv.org/abs/2603.22293.

[67] Wei W. Xing, Weijian Fan, Zhuohua Liu, et al. KATO: Knowledge alignment and transfer for transistor sizing of different design and technology. In *Proceedings of the 61st ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2024. doi: 10.1145/3649329.3657380.

[68] Yihao Xue, Allan Zhang, Jianhao Huang, et al. Beyond what seems necessary: Hidden gains from scaling training-time reasoning length under outcome supervision, 2026. URL https://arxiv.org/abs/2602.00927.

[69] An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115v1.

[70] An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388v1.

[71] Shunyu Yao, Howard Chen, John Yang, et al. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pages 20744–20757, 2022. doi: 10.52202/068431-1508. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/82ad13ec01f9fe44c01cb91814fd7b8c-Paper-Conference.pdf.

[72] Yuxuan Yin, Yu Wang, Boxun Xu, et al. ADO-LLM: Analog design Bayesian optimization with in-context learning of large language models. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, pages 1–9, 2025. doi: 10.1145/3676536.3676816. URL https://doi.org/10.1145/3676536.3676816.

[73] Qiying Yu, Zheng Zhang, Ruofei Zhu, et al. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3775. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

[74] Xi Yu, Dmitrii Torbunov, Soumyajit Mandal, et al. AutoSizer: Automatic sizing of analog and mixed-signal circuits via large language model (LLM) agents. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 151161–151186. PMLR, 2026. URL https://proceedings.mlr.press/v306/yu26aw.html.

[75] Yang Yue, Zhiqi Chen, Rui Lu, et al. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *Advances in Neural Information Processing Systems*, volume 38, pages 57654–57689, 2025. doi: 10.52202/085713-1933. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/537d5aa768c2d534016a4d06f87bc8fb-Paper-Conference.pdf.

[76] Mert Yuksekgonul, Daniel Koceja, Xinhao Li, et al. Learning to discover at test time. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 152432–152496. PMLR, 2026. URL https://proceedings.mlr.press/v306/yuksekgonul26a.html.

[77] Kaiwen Zha, Zhengqi Gao, Maohao Shen, et al. RL Tango: Reinforcing generator and verifier together for language reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[78] Wei Zhao and Yu Cao. New generation of predictive technology model for sub-45 nm early design exploration. *IEEE Transactions on Electron Devices*, 53(11):2816–2823, 2006. doi: 10.1109/TED.2006.884077.

[79] Haitian Zhong, Jixiu Zhai, Lei Song, et al. RC-GRPO: Reward-conditioned group relative policy optimization for multi-turn tool calling agents. *arXiv preprint arXiv:2602.03025*, 2026. URL https://arxiv.org/abs/2602.03025.

[80] Yifei Zhou, Song Jiang, Yuandong Tian, et al. SWEET-RL: Training multi-turn LLM agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025.

A Related Work

Analog circuit design and automation. Prior work on analog circuit design automation mainly follows two directions. The first treats sizing as black-box optimization [53] over simulator feedback. Representative early efforts investigated single-objective [32] or multi-objective Bayesian optimization [22, 36] for analog circuit design. We refer readers to the review paper [21] for a comprehensive overview. Subsequent developments, such as TuRBO [19], further improved the scalability of Bayesian optimization by using local trust regions. More recently, KATO [67] studies knowledge transfer for transistor sizing across different circuits and technology nodes. Despite their effectiveness, these approaches still optimize through implicit black-box search, offering limited interpretability and no explicit multi-turn reasoning process. The second direction learns optimization policies directly from data or simulation [24]. AutoCkt [51] formulates analog design as a reinforcement learning problem over circuit environments. GCN-RL [62] further introduces graph convolutional neural networks to improve transfer across technology nodes and similar circuit topologies. While these learning-based approaches demonstrate the promise of neural automation, they typically train a topology-specific optimizer or policy network rather than a general-purpose reasoning model.

Multi-turn and multi-reward reinforcement learning. Our method is closely related to recent work on reinforcement learning for multi-turn agentic optimization [63, 79]. MT-GRPO [65] studies turn-level credit assignment in multi-turn reasoning trajectories, highlighting that purely trajectory-level returns can produce ambiguous supervision across turns [66, 80]. GiGPO [20] further provides a practical formulation for multi-turn policy optimization over structured agent trajectories, and our implementation builds on this line of formulation. GTPO [17] combines turn-level rewards and return-based advantages for tool-integrated reasoning. Complementary theoretical work analyzes the process-level credit assignment implicit in outcome-based GRPO [58]. Our setting is also related to multi-reward reinforcement learning, where multiple objectives must be combined into a stable scalar training signal. GDPO [34] addresses this issue by introducing decoupled normalization for GRPO under multiple reward channels. In addition, Hindsight Experience Replay (HER) [1] shows that achieved outcomes can be relabeled as training targets in goal-conditioned reinforcement learning, which is closely related to our automatic goal construction strategy. Concurrent work TL-GRPO [31] also studies turn-level GRPO-style reinforcement learning for analog sizing. In contrast, we focus on problem-driven, locally deployable small-model training, with domain adaptations for circuit representation, multi-objective rewards, reward-guided rollout sampling, and simulator-interactive evaluation across topologies and technology nodes.

Large language models for circuit design. Recent work has begun to explore LLMs for analog design automation [55, 72]. AnalogCoder [29] studies training-free analog circuit design through Python code generation with feedback-enhanced prompting. EEsizer [33] integrates an LLM with circuit simulation and custom analysis functions for closed-loop transistor sizing. LEDRO [27] uses LLMs to iteratively reduce the analog sizing search space and guide downstream optimization, with an emphasis on cross-topology and cross-technology generalization. AutoSizer [74] further proposes a reflective two-loop meta-optimization framework and introduces AMS-SizingBench for evaluation. These works introduce stronger knowledge priors and often improve interpretability over earlier black-box methods, but they primarily rely on prompting, reflection, or agent scaffolds around fixed foundation models. While prior work uses reinforcement learning to train an LLM for circuit topology generation [60], we focus on training lightweight, locally deployable SLM agents for closed-loop analog circuit optimization, which is important for realistic workflows where proprietary PDKs and simulator models cannot be exposed to external APIs. This shifts the emphasis from using large hosted models as design assistants to adapting smaller models themselves for simulator-grounded multi-turn optimization.

B Circuit Representation and Action Format

B.1 XML Circuit Representation

We represent circuits using an XML-like structured text format augmented with explicit handle annotations. Unlike raw SPICE netlists, which are designed for simulator compatibility and often encode many task-relevant relations only implicitly, this representation makes component types,

editable attributes, and structural boundaries explicit in the token stream. This is particularly useful in our multi-turn setting, where the model must repeatedly inspect the current circuit state and produce targeted edits. The handle annotations further provide stable references for editable circuit elements across turns, reducing ambiguity during iterative optimization. A representative prompt excerpt is shown in the case study (Appendix L).

B.2 Discretized Relative Actions

The policy outputs each circuit edit action as a tuple (*target component*, *parameter*, *direction*, *ratio*), which the simulator applies as a multiplicative update to the current parameter value rather than generating an absolute new value from scratch. The framework does not prescribe a fixed number of circuit parameter edits per turn; instead, the model may generate a variable-length sequence of circuit edit actions in each single turn. For example, `increase @27 l_MM4 50%` multiplies the current channel length of the device at handle `@27` by 1.5. The ratio is a positive integer percentage with a configurable upper bound; we cap it at 500% in our experiments, which bounds the per-action perturbation while still allowing coarse early-stage circuit edits. This converts continuous parameter tuning into a finite set of local refinement operations and matches the iterative nature of analog sizing: each action refines the current design state instead of respecifying the full parameter configuration. The relative formulation is also less sensitive to scale variation across devices, topologies, and technology nodes.

B.3 Action Validity

SLIC enforces a multi-layer validity check on every action, with three pre-invocation filters and one post-execution detector. (1) *Format check*: actions failing to parse against the relative-update template are rejected. (2) *Admissibility check*: actions referencing components or parameters outside the dynamically generated admissible list are rejected. (3) *Parameter Range check*: after applying the multiplicative update, parameters falling outside their node-specific valid ranges (Table 5) are rejected. (4) *Simulator failure*: actions that pass the three pre-invocation filters but trigger DC/AC convergence failures or non-finite measurements during simulation are treated as invalid post-execution. All four cases incur the invalid-transition penalty λ_{inv} and contribute to GRPO advantages under joint normalization. Under separate normalization, invalidity affects the validity advantage, while candidates without usable performance measurements are excluded from the performance comparison. The first three additionally save simulator budget by rejecting actions before any simulator call.

Table 5: Technology-specific valid ranges for editable transistor parameters used in the parameter range check (Appendix B.3, item (3)).

Technology	Channel length L	Channel width W
PTM 22 nm	22 nm–100 μm	45 nm–500 μm
PTM 32 nm	45 nm–100 μm	45 nm–500 μm
PTM 45 nm	45 nm–100 μm	45 nm–500 μm
PTM 65 nm (training)[†]	45 nm–100 μm	45 nm–500 μm
PTM 90 nm	56 nm–100 μm	45 nm–500 μm
PTM 130 nm	85 nm–100 μm	45 nm–500 μm
TSMC CMOS 65 nm	60 nm–20 μm	120 nm–500 μm

[†] At PTM 65 nm, nine training topologies have topology-specific upper bounds on L ranging from 2.5 to 40 μm. No held-out topology is affected.

C Reward Function Details

C.1 Per-Specification Reward

We consider four specifications indexed by $\mathcal{K} = \{\Gamma, \mathrm{B}, \mathrm{P}, \varphi\}$, corresponding to gain, bandwidth, power consumption, and phase margin. At turn t, their measured values are

$$m_t^{(\Gamma)} = \Gamma_t, \qquad m_t^{(\mathrm{B})} = B_t, \qquad m_t^{(\mathrm{P})} = P_t, \qquad m_t^{(\varphi)} = \varphi_t.$$

These measurements are obtained after executing the candidate action. We omit the candidate index when describing a single reward evaluation. Gain is expressed in dB.

For gain, bandwidth, and power, let $q^{(k)}$ denote the corresponding target in the task specification q, with $\Gamma^\star = q^{(\Gamma)}$. We define

$$r_t^{(k)} = \min\left(1, \left(\frac{m_t^{(k)} + \delta}{q^{(k)} + \delta}\right)^{\eta_k}\right), \qquad \eta_k = \begin{cases} +1, & k \in \{\Gamma, \mathrm{B}\}, \\ -1, & k = \mathrm{P}. \end{cases} \tag{3}$$

The direction η_k reflects that higher gain and bandwidth are preferred, whereas lower power is preferred. When δ is negligible, the scores approach $m_t^{(k)}/q^{(k)}$ for gain and bandwidth and $q^{(k)}/m_t^{(k)}$ for power, capped at one. The small constant $\delta > 0$ prevents division by zero for nonnegative measurements and targets.

For a finite phase-margin measurement, the reward is the indicator of whether it lies within the target window $[\varphi_{\mathrm{lo}}, \varphi_{\mathrm{hi}}]$:

$$r_t^{(\varphi)} = b_t^{(\varphi)} = \mathbb{I}[\varphi_{\mathrm{lo}} \le \varphi_t \le \varphi_{\mathrm{hi}}], \tag{4}$$

where $\mathbb{I}[\cdot]$ is the indicator function. The no-unity-gain-crossing case is handled separately under the domain-validity rules.

C.2 Domain-Validity Gates

Let F_t denote the in-band peaking measure in dB. We define $d_t = \mathbb{I}[\Gamma_t > 0]$ to indicate positive gain and $\chi_t = \mathbb{I}[F_t \text{ is finite and } F_t > 6\,\mathrm{dB}]$ to indicate excessive peaking. The gated per-specification rewards are

$$\widetilde{r}_t^{(k)} = r_t^{(k)} \begin{cases} 1 - \chi_t, & k = \Gamma, \\ d_t, & k = \mathrm{P}, \\ d_t(1 - \chi_t), & k \in \{\mathrm{B}, \varphi\}. \end{cases} \tag{5}$$

Non-positive gain therefore suppresses auxiliary performance credit, while excessive peaking suppresses gain, bandwidth, and phase-margin credit. Power remains eligible when gain is positive because it is measured from the DC operating point. For a simulator-reported no-unity-gain-crossing case, phase-margin credit requires positive gain and a finite flatness measurement satisfying $F_t \le 6\,\mathrm{dB}$.

The same gates apply to specification satisfaction. A design succeeds only when all target specifications pass their tolerance-relaxed thresholds under these conditions. We use relative tolerance $\varepsilon_{\mathrm{tol},k} = 0.05$: gain and bandwidth lower bounds are multiplied by $1 - \varepsilon_{\mathrm{tol},k}$, the power upper bound by $1 + \varepsilon_{\mathrm{tol},k}$, and the phase-margin window is widened about its center by a factor of $1 + \varepsilon_{\mathrm{tol},\varphi}$. Flatness is a validity condition rather than an additional optimization objective.

C.3 Success-Conditional Slack Bonus

Uniform rewards for successful candidates can eliminate group-relative learning signals, motivating the dynamic sampling and filtering strategy in DAPO [73]. In circuit optimization, successful candidates can still differ in specification headroom. We therefore retain this information through a bounded success-conditional slack bonus.

For gain, bandwidth, and power, we define relative slack as

$$u_t^{(k)} = \eta_k \frac{m_t^{(k)} - q^{(k)}}{|q^{(k)}|}, \qquad k \in \{\Gamma, \mathrm{B}, \mathrm{P}\}, \tag{6}$$

using the optimization direction η_k defined in Equation 3. Positive slack indicates performance beyond the target in the desired direction.

We map each slack to $[0, 1]$ before averaging, assigning zero at the tolerance-relaxed boundary and one at the prescribed headroom cap. For phase margin, the score instead measures proximity to the

center of the accepted window. Writing $\varphi_c = (\varphi_{\mathrm{lo}} + \varphi_{\mathrm{hi}})/2$ and $h_\varphi = (\varphi_{\mathrm{hi}} - \varphi_{\mathrm{lo}})/2$, we define

$$\begin{aligned} z_t^{(k)} &= \mathrm{clip}\left(\frac{u_t^{(k)} + \varepsilon_{\mathrm{tol},k}}{u_{\max} + \varepsilon_{\mathrm{tol},k}}, 0, 1\right), \qquad k \in \{\Gamma, \mathrm{B}, \mathrm{P}\}, \quad u_{\max} = 1, \\ z_t^{(\varphi)} &= \mathrm{clip}\left(1 - \frac{|\varphi_t - \varphi_c|}{h_\varphi(1 + \varepsilon_{\mathrm{tol},\varphi})}, 0, 1\right). \end{aligned} \tag{7}$$

The tolerance shift preserves distinctions between designs that succeed within the tolerance band, while $u_{\max} = 1$ saturates the one-sided terms at 100% relative headroom. The phase-margin term rewards centering within the accepted window. For a no-crossing case satisfying the domain-validity conditions, we set $z_t^{(\varphi)} = 1$.

C.4 Separate Normalization of Validity and Performance

Following the reward-decoupled normalization principle of GDPO [34], we normalize action validity and measured performance separately before combining their advantages. In mixed-validity groups, the separation between valid actions and constant-penalty failures can dominate aggregate reward variance, compressing performance contrast among measured designs.

At turn t, let $\mathcal{I}_t = \{1, \ldots, G\}$ index the G candidate actions, and let $v_{t,i} \in \{0, 1\}$ denote the action-validity label of candidate i. Let $\mathcal{I}_t^{\mathrm{meas}} \subseteq \mathcal{I}_t$ contain candidates for which the simulator returns sufficient measurements to compute a performance reward $r_{t,i}$.

We compute a validity advantage relative to the full candidate group and a performance advantage relative to the candidates with available performance rewards:

$$\begin{aligned} A_{t,i}^{\mathrm{val}} &= \mathcal{N}_{\mathcal{I}_t}(v_t)_i, \\ A_{t,i}^{\mathrm{perf}} &= \begin{cases} \mathcal{N}_{\mathcal{I}_t^{\mathrm{meas}}}(\mathbf{r}_t)_i, & i \in \mathcal{I}_t^{\mathrm{meas}}, \\ 0, & i \notin \mathcal{I}_t^{\mathrm{meas}}. \end{cases} \end{aligned} \tag{8}$$

Here, v_t and $\mathbf{r}_t$ collect the validity labels and performance rewards at turn t, respectively. For a comparison set $\mathcal{S}$ with at least two members and nonzero variance, the normalization operator is

$$\mathcal{N}_{\mathcal{S}}(x)_i = \frac{x_i - \mu_{\mathcal{S}}(x)}{\mathrm{std}_{\mathcal{S}}(x) + \epsilon},$$

where $\mu_{\mathcal{S}}(x)$ and $\mathrm{std}_{\mathcal{S}}(x)$ are the mean and standard deviation of $\{x_j : j \in \mathcal{S}\}$, and $\epsilon > 0$ ensures numerical stability. Empty or singleton comparison sets and sets with zero variance contribute zero advantage. Thus, each advantage measures how a candidate compares with the mean of its respective comparison set.

The combined turn-level advantage is

$$A_{t,i} = \lambda_{\mathrm{val}} A_{t,i}^{\mathrm{val}} + \lambda_{\mathrm{perf}} A_{t,i}^{\mathrm{perf}}, \tag{9}$$

where λ_{val} and λ_{perf} weight the validity and performance advantages, respectively. We set $\lambda_{\mathrm{val}} = \lambda_{\mathrm{perf}} = 1$. This separates validity discrimination from performance ranking without interpreting unavailable measurements as ordinary low-performance observations. The combined advantage is assigned to the generated response tokens, including reasoning and actions, and used in the clipped policy objective.

C.5 Final Reward

The final turn reward is

$$r_t = \begin{cases} \lambda_{\mathrm{inv}}, & \text{no usable performance measurement,} \\ 1 + \dfrac{w}{|\mathcal{K}|} \displaystyle\sum_{k \in \mathcal{K}} z_t^{(k)}, & \text{design succeeds within tolerance,} \\ 10^{(\Gamma_t - \Gamma^\star)/20}, & \Gamma_t < 0, \\ \min\left(1 - \epsilon_s,\ \dfrac{1}{|\mathcal{K}|} \displaystyle\sum_{k \in \mathcal{K}} \widetilde{r}_t^{(k)}\right), & \text{otherwise,} \end{cases} \tag{10}$$

where $w = 0.5$, $\lambda_{\text{inv}} = -0.1$, and $\epsilon_s > 0$ caps unsuccessful scores strictly below one. Successful scores lie in $[1, 1.5]$, with the bonus reflecting mean specification headroom. Under separate normalization, λ_{inv} does not enter the advantage calculation. The same reward applies whether episodes stop at success or continue optimizing afterward.

D Reward-Guided Rollout Steering Algorithm

At each turn t, the policy samples G candidate edits $a_{t,i} \sim \pi_\theta(\cdot \mid h_t)$ from the shared history h_t. Here, h_t denotes the prompt containing the task specification, current circuit state, and retained interaction history. The simulator executes each candidate, producing successor states $s_{t+1,i}$, validity labels $v_{t,i} \in \{0, 1\}$, and post-action rewards $r_{t,i}$ from Section 3.2. We then form the eligible set

$$\mathcal{E}_t = \begin{cases} \{i : v_{t,i} = 1\}, & \text{if nonempty,} \\ \{1, \ldots, G\}, & \text{otherwise (random fallback),} \end{cases} \tag{11}$$

and a continuation policy κ selects $i_t^\star \sim \kappa(\{r_{t,i}\}_{i \in \mathcal{E}_t})$ to advance the shared history. All G transitions are retained for group-relative policy updates. For softmax continuation, $\tau_{\text{sel}} > 0$ controls the preference for higher-scoring candidates. The full algorithm is provided in Algorithm 1.

Algorithm 1 Reward-Guided Rollout Steering (RGRS)

Require: Initial circuit state s_0, target specification q, policy π_θ, horizon T, group size G, continuation policy κ

Initialize shared history $h_0 \leftarrow \text{Prompt}(q, s_0)$
Initialize buffer $\mathcal{D} \leftarrow \emptyset$
for $t = 0, \ldots, T-1$ **do**
  Sample candidate edits $\{a_{t,i}\}_{i=1}^G$ from $\pi_\theta(\cdot \mid h_t)$
  for $i = 1, \ldots, G$ **do**
    Execute $a_{t,i}$ in the simulator to obtain successor state $s_{t+1,i}$
    Compute validity label $v_{t,i}$ and post-action reward $r_{t,i}$
    Store $(h_t, a_{t,i}, s_{t+1,i}, r_{t,i}, v_{t,i})$ in $\mathcal{D}$
  end for
  Construct valid set $\mathcal{V}_t \leftarrow \{i \in \{1, \ldots, G\} : v_{t,i} = 1\}$
  if $\mathcal{V}_t = \emptyset$ **then**
    Set eligible set $\mathcal{E}_t \leftarrow \{1, \ldots, G\}$
  else
    Set eligible set $\mathcal{E}_t \leftarrow \mathcal{V}_t$
  end if
  if $\kappa =$ RANDOM or $\mathcal{V}_t = \emptyset$ **then**
    Sample $i_t^\star$ uniformly from $\mathcal{E}_t$
  else if $\kappa =$ GREEDY **then**
    Set $i_t^\star \leftarrow \arg\max_{i \in \mathcal{E}_t} r_{t,i}$
  else if $\kappa =$ SOFTMAX **then**
    Sample $i_t^\star$ from $\mathcal{E}_t$ with $p(i) \propto \exp(r_{t,i}/\tau_{\text{sel}})$
  end if
  Update shared history $h_{t+1} \leftarrow \text{Update}(h_t, a_{t,i_t^\star}, s_{t+1,i_t^\star})$
  if $s_{t+1,i_t^\star}$ satisfies all target specifications **then**
    break
  end if
end for
Compute turn-level advantages as in Appendix C.4 and update π_θ using all transitions in $\mathcal{D}$

E Additional Ablations

We report four-specification Pass@1 (%) at Turn 8 on 840 in-distribution and 240 out-of-distribution tasks. Each comparison changes only the indicated component within its specified training setting. All variants use the same evaluation targets and domain-validity criteria.

Table 6: Trajectory construction with dynamic filtering, no slack bonus, and stopping at success. RGRS uses random continuation. Turn 8 Pass@1 (%).

Construction	In-distribution	Out-of-distribution
Fixed-base (TL-GRPO)	73.8 ± 0.5	57.6 ± 0.6
Online (RGRS)	76.4 ± 0.9	56.7 ± 1.3

Table 7: Continuation candidate set with softmax selection, joint normalization, and continued optimization after success. Turn 8 Pass@1 (%).

Candidate set	In-distribution	Out-of-distribution
Eligible only	77.5 ± 0.3	59.3 ± 1.3
All candidates	76.9 ± 0.4	59.6 ± 0.7

Table 8: Normalization ablation for Qwen3.5-4B with random continuation and continued optimization after success. Results are four-specification Pass@1 (%) at Turn 8.

Normalization	In-distribution	Out-of-distribution
Joint	79.0 ± 0.7	60.4 ± 2.3
Separate	85.2 ± 2.2	65.3 ± 2.4

Table 9: History-length ablation for Qwen3.5-4B with 2k training tasks, softmax continuation, and joint normalization. Results are four-specification Pass@1 (%) at Turn 8.

History length	In-distribution	Out-of-distribution
$H = 1$	82.8 ± 0.7	64.9 ± 2.1
$H = 8$	83.6 ± 0.8	69.2 ± 2.5

E.1 Rollout Construction

Trajectory construction. With dynamic filtering [73] enabled and the slack bonus disabled, online construction improves in-distribution Pass@1 by 2.6 percentage points over fixed-base TL-GRPO [31], while out-of-distribution Pass@1 decreases by 0.9 points (Table 6). Thus, this comparison shows an in-distribution benefit but does not establish a generalization advantage for online construction alone.

Continuation candidate set. Restricting softmax selection to eligible candidates changes in-distribution and out-of-distribution Pass@1 by $+0.6$ and -0.3 percentage points, respectively (Table 7). The small, mixed differences suggest limited sensitivity to this restriction.

E.2 Advantage Normalization

With episodes continuing after success, separate normalization improves in-distribution and out-of-distribution Pass@1 by 6.2 and 4.9 percentage points, respectively (Table 8). Under joint normalization, the reward gap between invalid actions and measured designs can inflate group variance, compressing advantage differences among designs with distinct performance. Normalizing validity and performance separately preserves these comparisons and avoids using the invalid-action penalty to calibrate the scale of performance advantages.

E.3 Training Configuration and History

Termination after success. Stopping at success yields higher mean Pass@1 on both splits, with a larger improvement on held-out topologies (Table 10). Continuing after success is better suited to searching for higher-performing designs beyond target satisfaction, while stopping at success fits our goal-directed setting.

Training horizon. Training with four turns remains competitive with the main eight-turn configuration, whereas single-turn training substantially reduces performance on both splits when evaluated at eight turns (Table 10). For the main policy, rollouts that succeed within eight evaluation turns reach their first success after an average of 2.87 turns in-distribution and 3.16 turns out-of-distribution. This concentration of successful refinement within a few turns is consistent with the effectiveness of four-turn training.

Four-turn training also preserves the ability to improve through additional test-time simulator feedback. In contrast, single-turn training exhibits more pronounced diminishing returns from additional evaluation turns and saturates earlier, particularly on held-out topologies. A single-turn budget provides no RL experience of refinement through successive updates, which may limit the ability to sustain improvement during evaluation. Together, these observations suggest a trade-off between

Table 10: Ablations relative to the main SLIC-RANDOM configuration, which uses separate normalization, the success-conditional slack bonus, stopping at success, an eight-turn training budget, and structured history. All policies are evaluated at Turn 8.

Training configuration	In-distribution	Out-of-distribution
Main configuration	87.1 ± 0.7	70.6 ± 1.2
Continue optimization after success	85.2 ± 2.2	65.3 ± 2.4
Single-turn GRPO	61.6 ± 1.4	47.6 ± 0.9
Four-turn SLIC-RANDOM	89.5 ± 0.1	69.3 ± 0.6
Retain raw model outputs in history	76.8 ± 0.5	55.7 ± 1.7
Train with zero tolerance	81.7 ± 0.5	60.1 ± 0.6

training and test-time compute. A moderate training horizon may reduce training cost while retaining the iterative refinement behavior needed to benefit from further simulator interactions.

History representation. The main configuration retains the parsed action and simulator observation from the most recent turn ($H = 1$). The raw-history variant also uses $H = 1$ but retains the full model response, including reasoning and actions, along with the simulator observation. This change reduces performance on both splits (Table 10). Both variants start from the same SFT checkpoint, without additional SFT for the raw-history format. The degradation may partly reflect this format mismatch, as well as greater difficulty identifying relevant actions and simulator feedback within the additional reasoning text.

History length. We study retained history length using SLIC-SOFTMAX trained on a smaller dataset of 2,000 tasks, with joint normalization held fixed. Increasing history length from $H = 1$ to $H = 8$ yields a small in-distribution improvement and a larger gain on held-out topologies (Table 9). A longer history may help the policy compare earlier parameter edits and their outcomes, guiding subsequent exploration on unfamiliar circuit structures. Both variants start from the same SFT checkpoint trained with $H = 1$. Mean training prompt lengths with $H = 8$ are approximately 5.8–6.6k tokens, exceeding the 4.6–4.8k tokens observed in the raw-history experiment. These results suggest that longer inputs need not impair small-model performance when they provide useful, structured interaction history.

Training tolerance. Using zero tolerance during training reduces Turn 8 in-distribution and out-of-distribution Pass@1 by 5.4 and 10.5 percentage points, respectively, under the same evaluation criterion (Table 10). This suggests that tolerance-relaxed training thresholds provide a more effective learning signal near specification boundaries, whereas strict thresholds may place excessive emphasis on small residual violations.

E.4 Tolerance Sensitivity

SLIC-RANDOM trained with the default 5% success tolerance outperforms its zero-tolerance training counterpart across the evaluated range, even when evaluation requires strict target satisfaction (Figure 5). This suggests that modestly relaxing the training success threshold can improve learning and generalization. One possible explanation is that relaxing the success threshold makes higher rewards more accessible early in training, while the success-conditional slack bonus continues to encourage specification headroom. This may improve the utilization of sampled simulator evaluations and help the policy progress toward strict target satisfaction.

F Baseline Tree MT-GRPO Implementation

F.1 Tree MT-GRPO

We implement Tree MT-GRPO [65] with 16 tasks per training step, matching the main SLIC-RANDOM run. To keep the simulation budgets comparable, we use a four-turn training horizon and group size $G = 3$. Each active state branches into three candidate actions in the first three turns, and each surviving branch takes one action in the final turn. A fully expanded tree thus contains

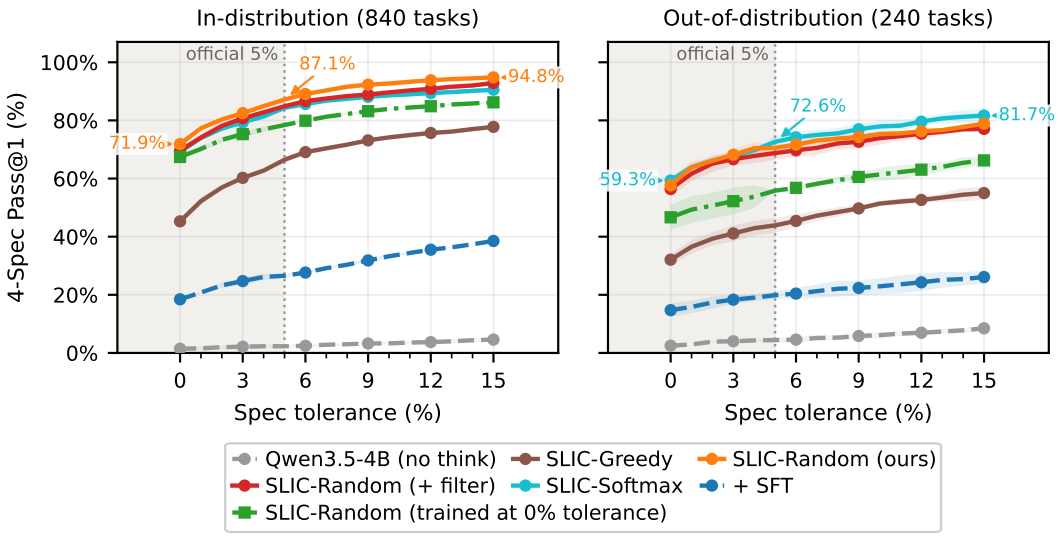


Figure 5: Tolerance sweep of four-specification Pass@1 within eight turns on 840 in-distribution tasks (left) and 240 out-of-distribution tasks (right), with evaluation tolerances ranging from 0% to 15%. The vertical dotted line marks the default 5% tolerance. Trajectories are generated with strict targets and reevaluated at each tolerance. Numerical annotations highlight selected values along the best-performing curve in each panel.

$3 + 9 + 27 + 27 = 66$ candidate actions per task, close to the $8 \times 8 = 64$ candidates in a full SLIC-RANDOM rollout. Episodes terminate at first success. All other optimization and generation settings follow the main SLIC-RANDOM configuration. Both methods are evaluated with the standard eight-turn interaction budget.

Intermediate rewards use SLIC's performance score for the updated circuit, with -0.1 assigned to invalid actions. The outcome reward evaluates the terminal circuit and includes the success-conditional slack bonus for successful designs. We normalize intermediate rewards among sibling actions to obtain A^{int}, and outcome rewards across trajectories for the same task to obtain A^{out}.

Following MT-GRPO [65, Eq. 5], we combine current and future intermediate advantages with the outcome advantage. With unit discount, the advantage at turn t of trajectory i, whose length is T_i, is

$$A^{\text{tree}}_{t,i} = \sum_{t'=t}^{T_i-2} A^{\text{int}}_{t',i} + A^{\text{out}}_{i}. \tag{12}$$

The terminal turn $t = T_i - 1$ receives only the outcome advantage A^{out}_i. For a shared response node n at turn t, Eq. 12 gives a trajectory advantage $A^{\text{tree}}_{t,i}$ for each continuation i. We average these advantages to obtain a single training advantage $\overline{A}(n)$. With $\text{Ch}(n)$ denoting its child responses,

$$\begin{aligned} \overline{A}(n) &= \mathbb{E}_{i|n}\left[A^{\text{tree}}_{t,i}\right] \\ &= \begin{cases} A^{\text{out}}(n), & \text{if } n \text{ is terminal,} \\ A^{\text{int}}(n) + \text{mean}_{n' \in \text{Ch}(n)}\, \overline{A}(n'), & \text{otherwise.} \end{cases} \end{aligned} \tag{13}$$

The expectation assigns equal weight to each child branch recursively. At each non-terminal node, we add its intermediate advantage to the mean of its children's advantages, which already include all subsequent intermediate and outcome terms. Each shared response is trained once, with $\overline{A}(n)$ assigned to all its generated tokens.

Tree depth is the main scalability bottleneck. At $G = 3$, extending the fully expanded tree from four to eight turns increases the number of trajectories per task from $3^3 = 27$ to $3^7 = 2{,}187$. At eight turns, increasing the group size from three to four raises this count to $4^7 = 16{,}384$. Under a fixed simulation budget, this exponential growth makes it difficult to scale tree-based training to more complex circuits and tasks that require more successive refinement steps to reach a successful design.

Table 11: Three-specification setting with Qwen2.5-7B-Instruct at 22 nm, evaluated on 196 in-distribution and 246 held-out-topology tasks. All trained policies use a four-turn training horizon. Bold denotes the highest mean in each column.

	In-distribution		Out-of-distribution	
Method	Turn 4	Turn 8	Turn 4	Turn 8
Qwen2.5-7B-Instruct	10.4 ± 0.8	12.4 ± 0.8	5.6 ± 0.6	7.3 ± 1.2
+ SFT	36.1 ± 0.6	47.6 ± 2.4	22.5 ± 1.9	33.9 ± 1.4
+ Trajectory-level GRPO	44.2 ± 1.6	48.6 ± 2.1	49.2 ± 2.2	55.4 ± 2.4
1.6T DeepSeek-V4-Pro THINK [15, 16]	53.6 ± 1.8	$\mathbf{67.9 \pm 0.5}$	43.0 ± 3.8	58.7 ± 0.6
SLIC-GREEDY	$\mathbf{60.7 \pm 1.3}$	63.4 ± 1.6	54.2 ± 1.3	58.7 ± 2.1
SLIC-RANDOM	$\mathbf{60.7 \pm 2.7}$	65.0 ± 1.8	50.0 ± 5.3	52.6 ± 5.5
SLIC-SOFTMAX	58.8 ± 2.9	64.6 ± 1.5	$\mathbf{59.2 \pm 2.1}$	$\mathbf{66.1 \pm 2.5}$

Table 12: Qwen2.5-7B-Instruct four-specification Pass@1 (%) at Turn 8.

Normalization	In-distribution	Out-of-distribution
Joint	32.2 ± 0.8	24.9 ± 1.7
Separate	41.8 ± 0.5	28.7 ± 1.1

G Training and Evaluation with Qwen2.5-7B-Instruct

G.1 Three-Specification Setting at 22 nm

We evaluate an earlier configuration that trains Qwen2.5-7B-Instruct at the 22 nm PTM node with a four-turn horizon and optimizes gain, power consumption, and phase margin. Evaluation covers 196 in-distribution tasks across 31 training topologies and 246 tasks across three held-out topologies at four and eight turns. The trajectory-level GRPO baseline uses independently sampled episodes with terminal-return supervision and trains stably in this three-specification setting.

Table 11 shows that all three SLIC variants outperform SFT on both splits at both horizons. At four turns, Greedy and Random achieve the highest in-distribution success, while Softmax performs best on held-out topologies. Softmax retains this OOD advantage at eight turns, reaching 66.1% Pass@1. All SLIC variants improve beyond their four-turn training horizon, demonstrating that the learned policies can benefit from additional simulator interactions.

G.2 Setting at 65 nm

We train Qwen2.5-7B-Instruct on approximately 2k tasks at the 65 nm PTM node, using Cadence Spectre for simulation. RL uses SLIC-SOFTMAX with group size $G = 4$ for 800 updates, without the success-conditional slack bonus. The joint-normalization run is used in the main backbone comparison. Separate normalization improves Pass@1 by 9.6 percentage points in-distribution and 3.8 points out-of-distribution (Table 12), supporting its effectiveness on Qwen2.5-7B-Instruct.

H Additional Experiments and Analysis

H.1 Training Dynamics

Figure 6 shows that Greedy develops the longest responses while remaining less successful than Random and Softmax. Its sampled responses repeatedly reconsider edits, using expressions such as "Actually," "Maybe," and "let's revert" (Figure 7). This pattern is consistent with unproductive overthinking, where additional tokens are spent revisiting decisions without comparable performance gains [9]. Individual reflection markers are not sufficient evidence of poor reasoning, as their effects depend on context [25].

Softmax's responses more often organize simulator feedback into a diagnosis, proposed edits, and expected effects. An auxiliary analysis of eight-turn evaluation rollouts also finds that its mean

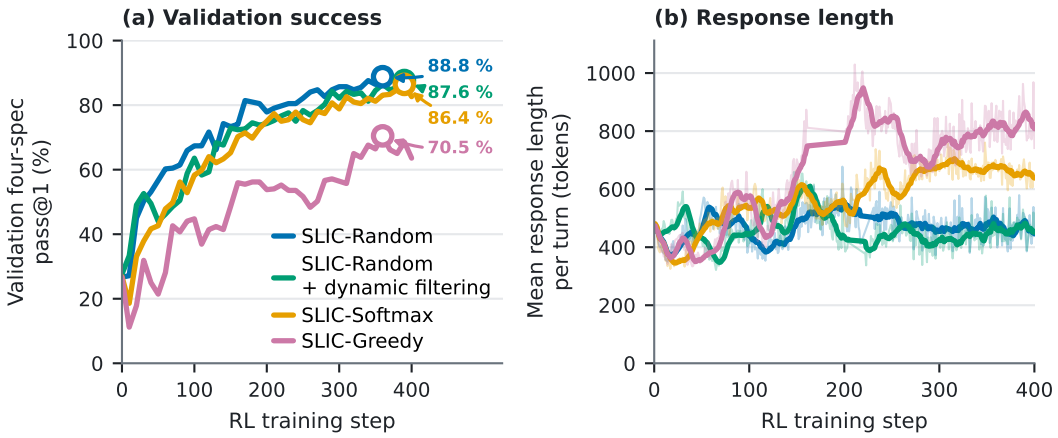


Figure 6: Training dynamics of SLIC under different continuation strategies. (a) Four-specification Pass@1 on the validation set used for checkpoint selection; open circles mark the best validation checkpoint for each variant. (b) Mean generated response length per turn during training, including both reasoning and actions.

SLIC-SOFTMAX

[...] **Diagnosis:** [...]

3. Gain is adequate at 64 dB but will decrease after these changes, which is acceptable since target is $\geq$ 49.8 dB.

Strategy: [...]

- Increase `w_MM12` slightly to increase output nMOS current (increasing gm). [...]

Expected results after actions:

- Bandwidth should increase from 11 kHz to $>$ 75 kHz

- Phase margin should improve from -9° to 30–90° [...]

SLIC-GREEDY

[...] Increasing gm or reducing parasitic? **Actually** previous step moved in wrong direction. [...]

Let's **revert** a few to move back to PM 2 deg, then try to push PM. [...]

Let's **revert** some and also try to push gain.

Maybe increase second stage gain devices or first stage. [...]

Ok, let's try **revert** and try different.

Figure 7: Illustrative response excerpts from two different training tasks. Softmax organizes its response around diagnosis, proposed edits, and expected effects. Greedy repeatedly revisits the decision to revert earlier changes. Bold text highlights these patterns, and brackets mark omissions.

next-turn improvement after a performance decrease remains similar across the two splits, whereas Random's declines on held-out topologies. These observations suggest that Softmax uses additional reasoning to support corrective progress on unfamiliar circuits. Although longer training-time reasoning can benefit OOD generalization [68], our results indicate that response length alone does not explain the differences between continuation strategies.

H.2 Evaluation on Harder Tasks

We additionally evaluate HARD, a more challenging benchmark variant at the same 65 nm technology node, retaining the 35 training topologies, six held-out topologies, and parameter ranges of STANDARD (Table 15). The fraction of valid initial designs decreases to 50.0% and 52.5% on the in-distribution and out-of-distribution evaluation splits, respectively. To assess target difficulty, we use the dataset builder's joint target-hit rate: the fraction of sampled candidate designs satisfying all target specifications. On held-out topologies, this rate drops from 2.3–3.4% on STANDARD to 0.18–0.40% on HARD, indicating rarer joint target attainment under the same sampling procedure.

Transfer from standard training. Table 13 shows that policies trained on STANDARD retain substantial success on HARD, with random and softmax continuation outperforming greedy continuation on both splits. The comparable success rates between initially valid and invalid tasks may partly reflect our effort to balance task difficulty during dataset construction. Valid initial designs are

Table 13: Four-specification Pass@1 (%) within eight turns on HARD, evaluated on 420 in-distribution and 240 out-of-distribution tasks. Reported overall and separately for tasks with valid and invalid initial designs. The upper group evaluates models trained on STANDARD without additional fine-tuning; the lower group compares training on HARD with and without the gain-prioritized difficulty gate. All models use Qwen3.5-4B unless otherwise specified.

	In-distribution			Out-of-distribution		
Method	Overall	Valid init.	Invalid init.	Overall	Valid init.	Invalid init.
Trained on STANDARD*; evaluated on* HARD						
SLIC-GREEDY	36.2 ± 2.3	36.7 ± 2.5	35.7 ± 2.1	32.5 ± 2.9	29.9 ± 3.6	35.4 ± 3.5
SLIC-SOFTMAX	54.3 ± 0.4	52.2 ± 1.0	56.3 ± 0.7	54.4 ± 2.3	51.9 ± 2.6	57.3 ± 2.0
SLIC-RANDOM	57.9 ± 1.8	56.2 ± 1.3	59.5 ± 2.4	53.9 ± 0.5	54.2 ± 1.7	53.5 ± 2.3
+ dynamic filtering	59.9 ± 1.6	59.4 ± 1.1	60.5 ± 2.1	59.4 ± 2.1	56.1 ± 2.8	63.2 ± 3.0
w/o domain-validity gates	56.7 ± 1.0	55.6 ± 0.7	57.8 ± 1.5	49.2 ± 0.7	49.2 ± 0.8	49.1 ± 1.8
Qwen3.5-9B SLIC-RANDOM	62.5 ± 1.6	60.8 ± 3.8	64.3 ± 0.8	58.1 ± 1.3	58.2 ± 0.5	57.9 ± 2.3
SLIC-RANDOM *trained and evaluated on* HARD						
Without difficulty gate	58.0 ± 0.5	56.3 ± 1.2	59.7 ± 2.0	50.3 ± 1.0	48.9 ± 3.7	51.8 ± 6.1
With difficulty gate	63.5 ± 0.6	60.8 ± 1.5	66.2 ± 1.4	56.5 ± 1.0	57.4 ± 2.3	55.6 ± 1.0

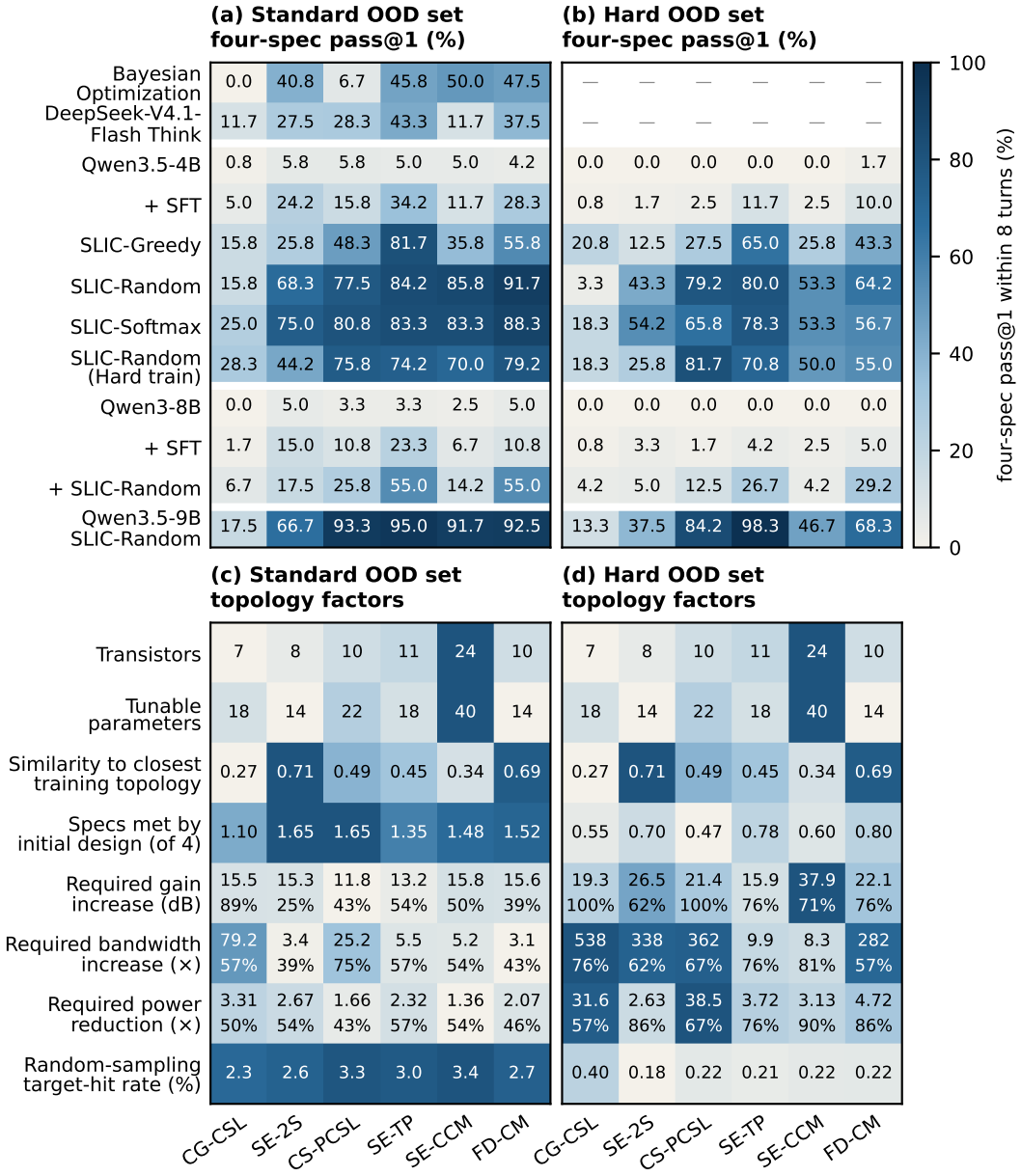
selected to perform poorly on the designated hard specification, while the selection criteria tend to yield less demanding targets for invalid initial designs, which additionally require recovery to a valid operating state.

Gain-prioritized difficulty gate. This additional gate activates reward terms for easier specifications only after gain meets its target within the 5% tolerance, preventing easier objectives from dominating the reward before gain is sufficient [34]. Specifically, the gain prerequisite is $\Gamma_t \geq (1 - 0.05)\Gamma^\star$. We compare training with and without this difficulty gate, while retaining the domain-validity gates and the same 5% success tolerance in both configurations. Although we observed no clear benefit on STANDARD, enabling it during training on HARD improves in-distribution and out-of-distribution Pass@1 by 5.5 and 6.2 percentage points, respectively (Table 13). The larger initial gain deficits on HARD suggest that gain is a more prominent bottleneck, making gain prioritization useful for identifying suitable operating points. On STANDARD, the difficulty differences among specifications are less pronounced, which may explain the limited benefit of imposing this priority.

H.3 Factors Associated with Topology Generalization

Figure 8 highlights contrasting cases of topology generalization. On STANDARD, CG-CSL has the fewest transistors among the held-out topologies, yet SLIC-RANDOM achieves only 15.8% Pass@1. Its tasks require large bandwidth increases and power reductions, making this structurally simple circuit challenging to optimize. Conversely, SE-CCM has the most tunable parameters (40) but reaches 85.8% Pass@1, with smaller required bandwidth increases and power reductions. For the same SE-CCM topology, success drops to 53.3% on HARD, where the required gain improvement is substantially larger. These cases illustrate that circuit size alone does not determine optimization difficulty: the performance changes required by the tasks are also important for interpreting success across topologies.

Correlation analysis. Figure 9 relates these performance differences to topology-level factors. On STANDARD, larger required bandwidth increases and power reductions are associated with lower success for the 4B RANDOM and SOFTMAX variants (Spearman $\rho = -0.71$ to -0.64). On HARD, required gain increases become more variable across topologies and are negatively associated with success across all four variants, while the negative association with required power reduction does not persist. Transistor count is positively associated with success on both datasets, whereas tunable parameter count shows little association, indicating that larger circuits are not necessarily harder to optimize in this benchmark. This is consistent with our goal construction, which aims to balance task difficulty across topologies: even structurally simple circuits can be assigned demanding performance targets and consequently exhibit low success rates. Together, these observations support interpreting difficulty through the performance changes required of each topology, rather than circuit size alone.



Gain / bandwidth / power rows: top = median gap between the initial design and the goal (dB of gain to add, × bandwidth increase, × power reduction), over the tasks where that spec is not met initially; bottom = initial unmet rate: % of tasks with a valid initial design that do not meet that spec (5 % tolerance).

Figure 8: Topology-level performance and task characteristics on STANDARD (left) and HARD (right). (a,b) Four-specification Pass@1 (%) on each held-out topology. Language models use up to eight turns, and BO uses 192 simulator calls per topology. SLIC checkpoints are trained on STANDARD unless labeled otherwise. Dashes denote unavailable results. (c,d) Circuit characteristics and task difficulty. For each specification gap, the main value is the median gap among tasks that initially miss the target. The smaller percentage gives the fraction of initially valid designs that miss it. Color scales are shared across datasets: one for Pass@1 and one per factor, with darker cells indicating larger values. Topology abbreviations: CG-CSL, common-gate amplifier with a current-source load; SE-2S, single-ended two-stage amplifier; CS-PCSL, common-source amplifier with a PMOS cascode current-source load; SE-TP, single-ended telescopic PMOS amplifier; SE-CCM, single-ended cascode current-mirror amplifier; FD-CM, fully differential current-mirror amplifier.



Figure 9: Spearman rank correlations between topology-level factors and mean four-specification Pass@1 within eight turns on STANDARD (left) and HARD (right). Each coefficient is computed across the six held-out topologies for the indicated SLIC variant. Gain, bandwidth, and power factors describe required improvements rather than raw performance values; larger values denote larger required changes. Blue and orange indicate positive and negative associations with success, respectively, on a shared scale centered at zero. Dashes denote unavailable results.

H.4 Distribution Shift

Figure 10 compares initial conditions, target specifications, initial target gaps, and circuit structure across dataset splits. Within both STANDARD and HARD, in-distribution tasks closely match their respective training sets, whereas held-out tasks show larger differences in device and parameter counts and architectural properties. Target specifications also differ across splits, particularly for bandwidth, so held-out evaluation involves changes in both circuit structure and target distributions. Comparing HARD training data with STANDARD, the structural distributions remain unchanged, while initial validity, target specifications, and initial target gaps shift substantially, especially for power. These changes reflect how HARD modifies the starting conditions and performance requirements within the same topology pool.

H.5 Statistical Analysis

We conduct task-paired comparisons on the 840 in-distribution and 240 out-of-distribution evaluation tasks. Each task's score is the mean of three binary indicators of four-specification success within eight turns. We estimate mean task-wise differences and their 95% percentile confidence intervals using paired bootstrap resampling over tasks (10,000 resamples, seed 7). Each resampled task retains all three evaluation repetitions and the pairing between methods.

Table 14 reports selected comparisons. Significance is assessed using two-sided bootstrap tests with Holm correction across all 20 Turn 8 tests in the analysis. The analysis treats tasks as independent sampling units and conditions on the evaluated checkpoints.

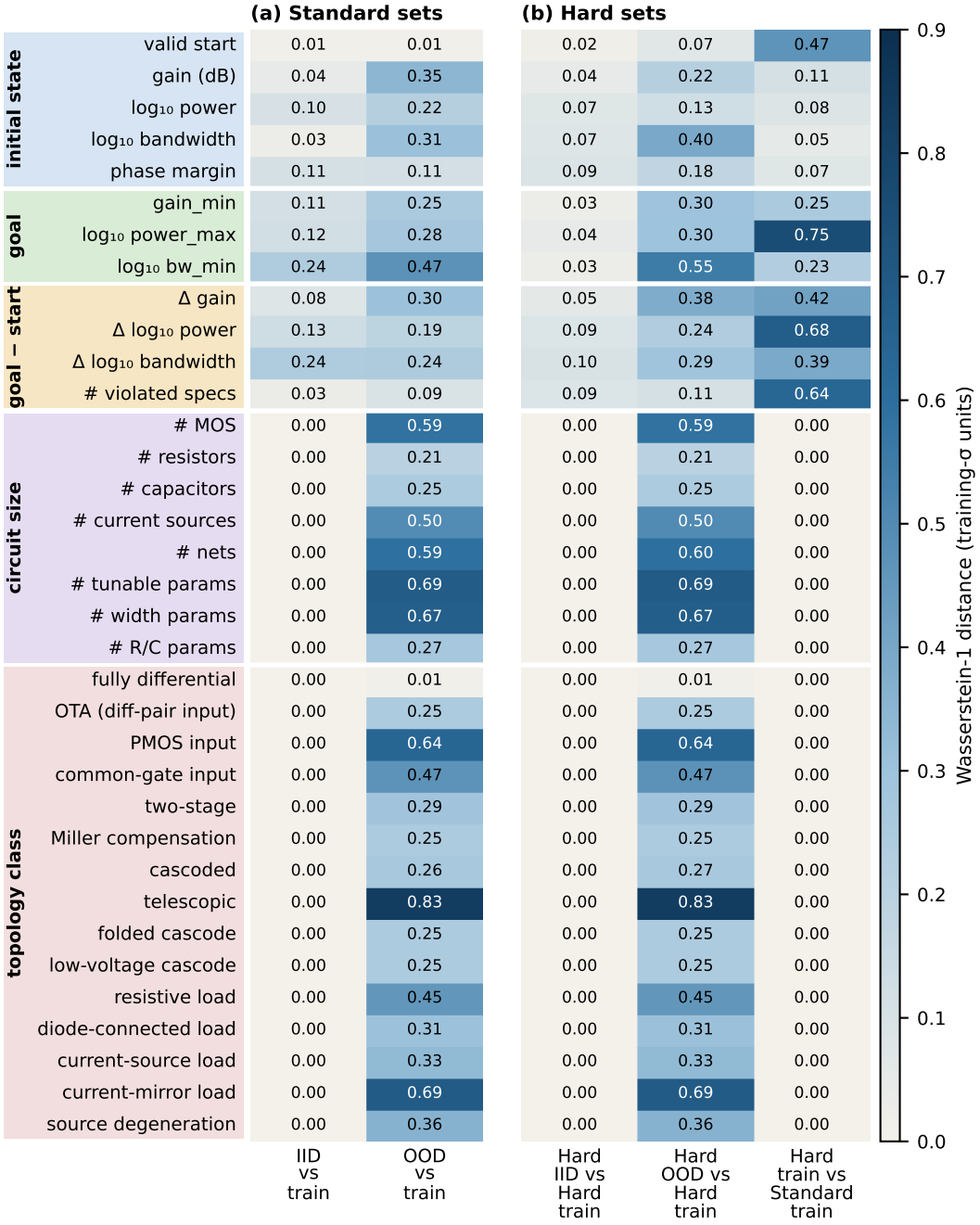


Figure 10: Feature-wise distribution shifts for STANDARD and HARD. Each column compares the indicated dataset with its reference training set using Wasserstein-1 distances computed after standardization using the same reference-training statistics for both compared sets. Power and bandwidth are log-transformed, and distances are expressed in reference-training standard-deviation units. Features describe initial conditions, specification targets, initial target gaps, circuit-size counts, and 15 expert-defined binary design descriptors. The descriptors encode architectural properties using fixed rules applied to circuit metadata and canonical topology names; multiple descriptors may apply to the same topology. For binary descriptors, distances reflect differences in attribute prevalence after normalization. Larger distances indicate stronger distribution shifts.

Table 14: Task-paired differences in four-specification Pass@1 within eight turns. Differences (Δ, in percentage points) are computed as the first method minus the second. Random, Softmax, and Greedy denote the corresponding SLIC variants. The normalization and slack-bonus comparisons use their respective matched training settings. Confidence intervals are pointwise 95% percentile bootstrap intervals. * indicates significance at $\alpha = 0.05$ after Holm correction across all 18 Turn 8 tests in the analysis.

	In-distribution		Out-of-distribution	
Comparison	Δ	95% CI	Δ	95% CI
Random vs. SFT	$+60.6^*$	[57.9, 63.3]	$+50.7^*$	[45.0, 56.4]
Random vs. DeepSeek-V4.1-Flash THINK	$+48.5^*$	[45.6, 51.5]	$+43.9^*$	[38.2, 49.4]
Random vs. Greedy	$+20.7^*$	[18.1, 23.3]	$+26.7^*$	[21.2, 32.1]
Random vs. w/o domain-validity gates	$+4.3^*$	[2.4, 6.2]	$+6.9^*$	[3.1, 11.0]
Softmax vs. Random	-2.8^*	[−4.7, −1.0]	$+2.1$	[−1.7, 5.8]
Separate vs. joint normalization	$+6.2^*$	[4.2, 8.3]	$+4.9$	[0.1, 9.7]
With vs. without success-conditional slack bonus	$+8.5^*$	[6.3, 10.8]	$+12.1^*$	[7.5, 16.7]

SLIC-RANDOM significantly outperforms SFT, DeepSeek, SLIC-GREEDY, and its variant without domain-validity gates on both splits after correction. SLIC-SOFTMAX has significantly lower in-distribution performance than SLIC-RANDOM, while its higher out-of-distribution mean does not establish a statistically significant difference. Among the component ablations, separate normalization significantly improves in-distribution performance, while its out-of-distribution improvement does not remain significant after correction. The success-conditional slack bonus significantly improves performance on both splits under dynamic filtering.

I Implementation Details

I.1 Dataset

We build our benchmark by adapting circuit topologies from the GANA dataset [28]. For each topology, we retain the main amplifier circuit together with its associated current-bias circuit, and curate the resulting circuits into train, validation, and held-out test splits for simulator-interactive optimization. We group the topologies by complexity, treating differential-input amplifiers as complex circuits and single-input amplifiers as simple circuits. The dataset is constructed by varying both the initial circuit states and the target goal specifications. Within the prescribed topology split, training and evaluation tasks are sampled with stratification by circuit topology and task difficulty to control the representation of different circuit structures and optimization demands. The resulting category-level dataset statistics are summarized in Table 15.

SFT data. The SFT corpus contains 6,981 single-turn samples from 1,568 teacher-generated episodes across the 35 training topologies. We split the corpus by episode into 6,233 training samples from 1,400 episodes and 748 test samples from 168 episodes. All samples from the same episode remain in one split, while both splits cover the same 35 topologies.

I.2 Hosted Model Evaluation

DeepSeek-V4.1-Flash was evaluated on the 840 ID tasks in two 420-task batches at different times, using the same API model identifier, prompts, and protocol. The two batches achieved 35.3% and 41.9% Pass@1, while GPT-6-Astra achieved similar scores on both (91.0% and 90.2%). The DeepSeek difference may therefore reflect changes in the served model or inference backend. The main results aggregate both batches, with three evaluation repetitions per task. Using fixed local checkpoints avoids uncertainty about model version changes between evaluations. The total API cost for the DeepSeek evaluation was about $227.

Table 15: Category-wise dataset statistics for STANDARD and HARD at the 65 nm technology node. Both variants share the topology sets and parameter ranges within each split and category. Tunable parameters are reported as min–max ranges across topologies. Valid init. denotes the percentage of tasks with a valid initial design.

				STANDARD		HARD	
Split	**Category**	**Topology count**	**Tunable parameters (min–max)**	**Samples**	**Valid init. (%)**	**Samples**	**Valid init. (%)**
Train	Complex	19	10–36	3,605	70.5	2,432	49.2
	Simple	16	4–22	3,027	70.4	2,048	49.2
	Total	35	4–36	6,632	70.4	4,480	49.2
In-distribution test	Complex	19	10–36	456	71.1	228	50.0
	Simple	16	4–22	384	70.8	192	50.0
	Total	35	4–36	840	71.0	420	50.0
Out-of-distribution test	Complex	4	14–40	160	70.0	160	52.5
	Simple	2	18–22	80	70.0	80	52.5
	Total	6	14–40	240	70.0	240	52.5

I.3 Training Configurations

Table 16: Main supervised fine-tuning (SFT) configuration.

Category	**Setting**	**Value**
Hardware	Nodes / GPUs	1 node, 8 $\times$ H100 (80 GB)
Model	Base model	Qwen3.5-4B
	Training strategy	FSDP2
	Model load dtype	bfloat16
	Gradient checkpointing	Enabled
Data	Dataset	Single-turn step samples
	Training samples	6,233
	Test samples	748
	Train batch size	8
	Micro-batch size / GPU	1
	Max sequence length	11,776
	Truncation	Left
Environment / Task	Representation	XML
	Technology node	65 nm (PTM)
	Max episode steps	8
	History length	1
Optimization	Optimizer	AdamW
	Learning rate	1×10^{-5}
	Warmup ratio	0.1
	Total epochs	4

J Limitations

While SLIC demonstrates strong capabilities in analog circuit optimization, several limitations remain. First, SLIC focuses on device sizing within a given circuit topology. Extending it to a complete design flow that integrates topology generation, sizing, robustness verification under process, voltage, and temperature (PVT) variations, and physical layout remains future work. Second, our evaluation currently focuses on amplifier optimization with four specifications. Extending SLIC to a broader range of circuits, including low-dropout regulators (LDOs), low-noise amplifiers (LNAs), and power amplifiers (PAs), and incorporating additional coupled objectives such as noise and offset remain directions for future work.

Table 17: Main RL training configuration for SLIC-RANDOM.

Category	Setting	Value
Hardware	Nodes / GPUs	1 node, 2 × B200
	CPUs per GPU	16
	Host memory	400 GB
Model	Initialization checkpoint	Qwen3.5-4B SFT
	Actor learning rate	1×10^{-6}
	Learning-rate schedule	Cosine decay to 1×10^{-7}
	Warmup	None
	Optimizer	AdamW
	Weight decay	0.01
	Gradient checkpointing	Enabled
Data	Train batch size	16 tasks
	Rollout group size	8
	Training turns per step	1,024
	Validation set size	420 tasks
	Max prompt length	13,072
	Max response length	2,048
	Truncation	Middle
Environment	Representation	XML
	Technology node	65 nm (PTM)
	Max steps	8
	History length	1
RL algorithm	Algorithm	GRPO with RGRS
	Advantage normalization	Separate validity / performance
	Validity / performance weights	1.0 / 1.0
	PPO epochs	1
	PPO mini-batch size	256
	PPO micro-batch size / GPU	1
	Clip ratio (low / high)	0.2 / 0.28
	KL loss	Disabled
Rollout	Backend	vLLM
	Tensor parallel size	1
	Temperature	1.0
	Top-p	0.9
	Samples per prompt	1
Reward shaping	Invalid action penalty magnitude	0.1
	Simulation failure penalty magnitude	0.1
Training schedule	Total training steps	400

K Broader Impact

SLIC aims to advance practical AI-driven automation for analog circuit design, a field that remains highly manual, expert-intensive, and difficult to scale across circuit families, specifications, and technology nodes. By training a locally deployable language-model agent to iteratively optimize circuits using verifiable simulator feedback, this work points toward design tools that can reduce repetitive manual tuning and improve the scalability of analog and mixed-signal development. The local deployment setting is especially important for industry, where proprietary PDKs, device models, and circuit IP often cannot be exposed to hosted LLM APIs. Circuit-specialized small models could therefore provide a practical alternative for secure in-house design assistance and parallel multi-agent workflows across circuit sub-blocks.

Local deployment could also support education in analog integrated circuit design, particularly when repeated use of hosted LLM APIs is costly and confidentiality restrictions prevent sharing proprietary commercial PDK data with external services. Teaching analog integrated circuit design often requires instructors to demonstrate how to make design decisions, interpret simulation results, and debug circuits, yet such individual guidance can be difficult to provide when teaching resources are limited.

Analog integrated circuit design agents running locally on students' computers could complement this guidance by making the design process visible and interactive. Students could follow proposed changes, inspect simulator feedback, and explore subsequent revisions, making practical design experience more accessible.

However, such systems should remain coupled with expert review and standard verification flows, since final analog designs must satisfy reliability, manufacturability, and safety requirements beyond the simulated objectives considered in this work.

L Case Studies

We present two evaluation rollouts of SLIC-RANDOM at the 65 nm technology node, covering different topologies, initial operating conditions, and target specifications. Case 1 uses a fully-differential folded-cascode amplifier from the in-distribution split and starts from a valid design that meets only the bandwidth target. Case 2 uses a held-out fully-differential current-mirror amplifier from the out-of-distribution split and starts from an invalid design. Both reach all four targets in four turns, including recovery from an invalid intermediate design. Model responses are reproduced as generated, including their intermediate interpretations. Long graph descriptions and repeated device features are omitted where indicated.

L.1 Case 1: In-Distribution Folded-Cascode Amplifier

Figure 11 shows the folded-cascode amplifier used in this case. This topology contains 20 transistors across the amplifier core and three bias-generator blocks, with 31 tunable parameters. The initial gain is 36.7 dB below the target, power is about 34 times the budget, and phase margin is outside the required window. The policy first recovers from an invalid update, then increases bandwidth while retaining sufficient gain and acceptable power.

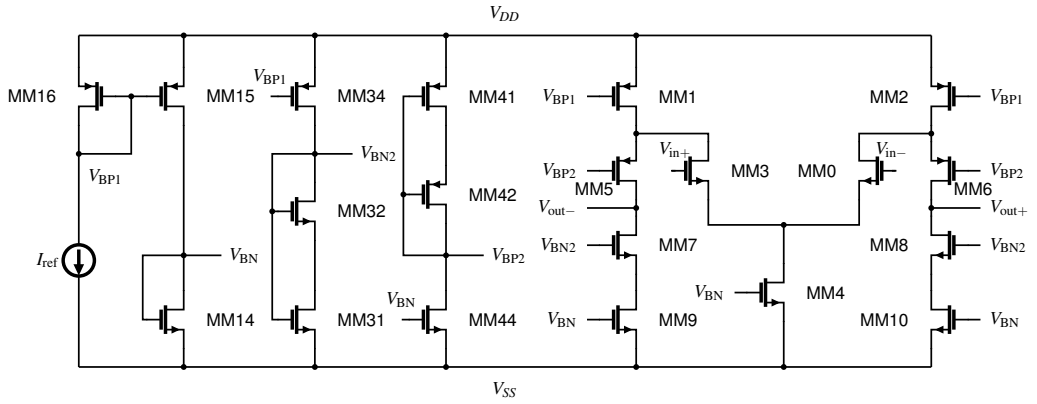


Figure 11: Fully differential folded-cascode amplifier used in Case 1. X0 is the amplifier core and X1–X3 generate its bias voltages. Device identifiers and connections follow the supplied netlist. In the top-level instance, V_{biasn1} is tied to V_{biasn}. Identical bias labels denote the same electrical net.

Task Setup

Topology	Fully-differential folded-cascode amplifier
Evaluation split	In-distribution
Technology	65 nm CMOS (PTM), $V_{DD} = 1.2$ V
Gain target	≥ 61.7 dB
Power target	≤ 425.7 μW
Bandwidth target	≥ 43.1 kHz (-3 dB bandwidth)
Phase-margin target	$30^\circ \leq \mathrm{PM} \leq 90^\circ$
Evaluation tolerance	5%, as defined in the main evaluation
Initial gain	25.0 dB
Initial power	14,349 μW
Initial bandwidth	111 kHz
Initial phase margin	-36°
Initial in-band peaking	0.0 dB
Initial validity	Valid. Only the bandwidth target is met.

Structured System Input Excerpt

```
<GRAPH_INFO>
TOPOLOGY_NAME: Fully_differential_folded_cascode
</GRAPH_INFO>
<GRAPH_NODE_IDMAP>
type=pin: @0=vdd @1=vss
type=current_source: @70=X1.I0
type=n_mosfet: @10=X0.MM0 @114=X3.MM44 @16=X0.MM10 @30=X0.MM3 @35=X0.MM4 @50=X0.MM7
  @55=X0.MM8 @60=X0.MM9 @73=X1.MM14 @88=X2.MM31 @93=X2.MM32
type=p_mosfet: @104=X3.MM41 @109=X3.MM42 @15=X0.MM1 @25=X0.MM2 @40=X0.MM5 @45=X0.MM6
  @78=X1.MM15 @83=X1.MM16 @98=X2.MM34
type=net: LG_Vbiasn LG_Vbiasn2 LG_Vbiasp1 LG_Vbiasp2 Vinn Vinp Voutn Voutp ...
</GRAPH_NODE_IDMAP>
<PINMAP> ... </PINMAP>
<EDGES> ... </EDGES>

Design bounds:
For tech 65nm, length l must stay within [45nm, 100um] and width w must stay
within [45nm, 500um]. Designs outside this range are invalid.
Fixed global: vdd=1.2V (user-set, not tunable).

Allowed valid target handles and parameter tokens:
  @10: l_MM30, w_MM30      (MM0 -- NMOS input pair, matched with @30)
  @15: l_MM12, w_MM12      (MM1 -- PMOS current source, matched with @25)
  @16: l_MM910, w_MM910    (MM10 -- NMOS current source, matched with @60)
  @35: l_MM4, w_MM4        (MM4 -- NMOS tail)
  @40: l_MM56, w_MM56      (MM5 -- PMOS cascode, matched with @45)
  @50: l_MM78, w_MM78      (MM7 -- NMOS cascode, matched with @55)
  @70: iref                (I0 -- reference current)
  @73, @78, @83: MM14/MM15/MM16   (X1: LG_nmos_l1 -> Vbiasn, Vbiasp1)
  @88, @93, @98: MM31/MM32/MM34   (X2: LG_load_biasn_LV -> Vbiasn2)
  @104, @109, @114: MM41/MM42/MM44 (X3: LG_load_biasp_LV -> Vbiasp2)
```

Turn 1 User Input Excerpt

```
Your goal is: Achieve gain >= 61.7 dB, power <= 425.7 uW,
-3dB bandwidth >= 43.1 kHz,
30.0 deg <= phase_margin <= 90.0 deg
(each spec has 5% tolerance). Prioritize gain first.
[...]

Your current observation is:
<PERFORMANCE>
<GainAt1kHz>25.0 dB</GainAt1kHz>
<Power>14349 uW</Power>
<Bandwidth3dB>111 kHz</Bandwidth3dB>
<PhaseMargin>-36 deg</PhaseMargin>
<InBandFlatness>0.0 dB (limit 6)</InBandFlatness>
</PERFORMANCE>
<FEATURES>
@10 id=X0.MM0 type=n_mosfet w=w_MM30=34.78u l=l_MM30=166.8n Vgs=496 mV
    Vds=698 mV Vdsat=83.9 mV Ids=438 uA Gm=7 mS region=sat
@15 id=X0.MM1 type=p_mosfet w=w_MM12=17.5u l=l_MM12=69.58n Vgs=-476 mV
    Vds=-286 mV Vdsat=-157 mV Ids=-436 uA Gm=4.6 mS region=sat
@16 id=X0.MM10 type=n_mosfet w=w_MM910=175.1u l=l_MM910=6.297u Vgs=834 mV
    Vds=5 mV Vdsat=367 mV Ids=33.6 uA Gm=66.1 uS region=triode
@35 id=X0.MM4 type=n_mosfet w=w_MM4=4.998u l=l_MM4=199.9n Vgs=834 mV
    Vds=216 mV Vdsat=315 mV Ids=872 uA Gm=2.2 mS region=triode
@40 id=X0.MM5 type=p_mosfet w=w_MM56=117.9n l=l_MM56=20u Vgs=-914 mV
    Vds=-202 mV Vdsat=-471 mV Ids=-22.0 nA Gm=48.2 nS region=triode
@50 id=X0.MM7 type=n_mosfet w=w_MM78=75.97n l=l_MM78=20u Vgs=502 mV
    Vds=706 mV Vdsat=114 mV Ids=11.9 nA Gm=167 nS region=sat
@70 id=X1.I0 type=current_source dc=iref=23u ...
</FEATURES>
```

Turn 1 Original LLM Output Excerpt

```
<think>
The current operating point is far from specs: DC gain is only 25 dB (needs >=61.7 dB),
power is 14.35 mW (needs <=425.7 uW), and phase margin is -36 deg (needs 30-90 deg).
Bandwidth is 111 kHz, which is acceptable, but the main problems are: 1. Excessive bias
current: iref=23u causes excessive DC current. 2. PMOS current source MM1/MM2 is too
short (69.58nm) and wide, allowing too much DC current. 3. NMOS current source MM9/MM10
is very wide (175.1u) and acts as resistor rather than current source; needs moderate
W/L. 4. Input pair MM3/MM0 is too wide and short; needs stronger gm with better ro.
5. Tail MM4 is over-sized in current. 6. PMOS current source MM5/MM6 and NMOS cascode
   MM7/MM8 are weak (only 117.9n/75.97n). 7. DC gain is low because ro of signal devices
   are too small. Strategy: Reduce iref, make PMOS load stronger but less conductive,
   improve input pair, moderate tail, strengthen cascode current source, and fix bias
   blocks. ...
</think>
<action>decrease @70 iref 95%</action>
<action>increase @15 l_MM12 200%</action>
<action>decrease @15 w_MM12 50%</action>
<action>decrease @30 w_MM30 70%</action>
<action>increase @30 l_MM30 200%</action>
<action>decrease @35 w_MM4 50%</action>
<action>increase @35 l_MM4 20%</action>
<action>decrease @16 w_MM910 95%</action>
<action>increase @16 l_MM910 50%</action>
<action>increase @40 w_MM56 500%</action>
<action>decrease @40 l_MM56 99%</action>
<action>increase @50 w_MM78 500%</action>
<action>decrease @50 l_MM78 99%</action>
<action>decrease @73 w_MM14 95%</action>
<action>decrease @73 l_MM14 85%</action>
<action>decrease @78 w_MM15 98%</action>
<action>increase @78 l_MM15 300%</action>
<action>decrease @83 w_MM16 95%</action>
<action>decrease @83 l_MM16 70%</action>
<action>increase @104 w_MM41 500%</action>
<action>increase @109 w_MM42 500%</action>
<action>increase @114 w_MM44 10%</action>
```

Simulator Result After Turn 1

Gain / power	-21.9 dB / 23 μW
Bandwidth	Not measurable
Phase margin	Unavailable

The 22 simultaneous updates produce an invalid design with non-positive gain. The policy receives the simulator measurements and the `design_error` message before the next turn.

Turn 2 User Input Excerpt

```
<PERFORMANCE>
<Status>INVALID - design_error</Status>
<Error>Design produces non-positive gain: -21.86 dB.</Error>
<GainAt1kHz>-21.9 dB</GainAt1kHz>
<Power>23 uW</Power>
<Bandwidth3dB>not measurable</Bandwidth3dB>
<Validity>non-positive gain; power/bandwidth/PM not meaningful until gain > 0</Validity>
</PERFORMANCE>
<FEATURES>
@10 id=X0.MM0 type=n_mosfet w=w_MM30=10.4u l=l_MM30=500.4n
@15 id=X0.MM1 type=p_mosfet w=w_MM12=8.8u l=l_MM12=208.7n
@16 id=X0.MM10 type=n_mosfet w=w_MM910=8.8u l=l_MM910=9.4u
@35 id=X0.MM4 type=n_mosfet w=w_MM4=2.5u l=l_MM4=239.9n
@40 id=X0.MM5 type=p_mosfet w=w_MM56=707.4n l=l_MM56=200n
@50 id=X0.MM7 type=n_mosfet w=w_MM78=455.8n l=l_MM78=200n
@70 id=X1.I0 type=current_source dc=iref=1.2u ...
</FEATURES>
```

Turn 2 Original LLM Output Excerpt

```
<think>
The current operating point is invalid: DC gain is -21.9 dB (magnitude only 0.219x). The
previous aggressive changes destroyed the DC bias: the PMOS current source/load
(MM1/MM2) is too strong, tail MM4 is weak, input pair MM3/MM0 is weak, and NMOS current
source MM9/MM10 is overkill. The core needs to be restored: make PMOS load weaker
(larger l_MM12, smaller MM1/MM2), stronger input pair MM3/MM0, stronger tail MM4,
moderate iref, and adjust NMOS cascode/load. Need positive DC gain first.
</think>
<action>increase @15 l_MM12 400.0%</action>
<action>decrease @15 w_MM12 30.0%</action>
<action>increase @10 w_MM30 100.0%</action>
<action>increase @10 l_MM30 100.0%</action>
<action>increase @35 w_MM4 300.0%</action>
<action>increase @35 l_MM4 50.0%</action>
<action>decrease @16 w_MM910 70.0%</action>
<action>increase @16 l_MM910 20.0%</action>
<action>increase @40 w_MM56 300.0%</action>
<action>increase @40 l_MM56 20.0%</action>
<action>increase @50 w_MM78 200.0%</action>
<action>increase @50 l_MM78 20.0%</action>
<action>decrease @70 iref 10.0%</action>
<action>decrease @73 w_MM14 70.0%</action>
<action>decrease @73 l_MM14 20.0%</action>
<action>increase @78 w_MM15 100.0%</action>
<action>decrease @78 l_MM15 10.0%</action>
<action>decrease @83 w_MM16 70.0%</action>
<action>increase @83 l_MM16 10.0%</action>
<action>increase @98 w_MM34 150.0%</action>
<action>decrease @93 w_MM32 85.0%</action>
<action>decrease @88 w_MM31 70.0%</action>
<action>decrease @88 l_MM31 60.0%</action>
```

Simulator Result After Turn 2

Gain / power	75.6 dB / 98 μW
Bandwidth / phase margin	15 kHz / 42°
In-band peaking	0.0 dB

The design becomes valid and meets the gain, power, and phase-margin targets. Bandwidth remains below its 43.1 kHz target, while gain now has substantial headroom.

Turn 3 User Input Excerpt

```
<PERFORMANCE>
<GainAt1kHz>75.6 dB</GainAt1kHz>
<Power>98 uW</Power>
<Bandwidth3dB>15 kHz</Bandwidth3dB>
<PhaseMargin>42 deg</PhaseMargin>
<InBandFlatness>0.0 dB (limit 6)</InBandFlatness>
</PERFORMANCE>
<FEATURES>
@10 id=X0.MM0 type=n_mosfet w=w_MM30=20.8u l=l_MM30=1u Vgs=324 mV
    Vds=434 mV Vdsat=48.5 mV Ids=3 uA Gm=63.6 uS region=subthreshold
@15 id=X0.MM1 type=p_mosfet w=w_MM12=6.2u l=l_MM12=1u Vgs=-477 mV
    Vds=-760 mV Vdsat=-139 mV Ids=-3.3 uA Gm=42.6 uS region=sat
@16 id=X0.MM10 type=n_mosfet w=w_MM910=2.6u l=l_MM910=11.3u Vgs=446 mV
    Vds=281 mV Vdsat=83.1 mV Ids=351 nA Gm=6 uS region=sat
@35 id=X0.MM4 type=n_mosfet w=w_MM4=10u l=l_MM4=359.8n Vgs=446 mV
    Vds=6.2 mV Vdsat=82.0 mV Ids=6 uA Gm=71.8 uS region=triode
@40 id=X0.MM5 type=p_mosfet w=w_MM56=2.8u l=l_MM56=240n Vgs=-440 mV
    Vds=-110 mV Vdsat=-61.9 mV Ids=-342 nA Gm=7.5 uS region=subthreshold
@50 id=X0.MM7 type=n_mosfet w=w_MM78=1.4u l=l_MM78=240n Vgs=345 mV
    Vds=49.2 mV Vdsat=45.8 mV Ids=341 nA Gm=7.7 uS region=cutoff
@70 id=X1.I0 type=current_source dc=iref=1.1u ...
</FEATURES>
```

Turn 3 Original LLM Output Excerpt

```
<think>
Current specs: Gain is 75.6 dB, well above the 61.7 dB target. Power is only 98 uW, well
within 425.7 uW. Phase margin is 42 deg, which meets 30-90 deg. The unmet spec is
bandwidth: only 15 kHz, needs >=43.1 kHz. The main bottleneck is the high output
resistance: MM3/MM0 are long (1u), MM1/MM2 PMOS load is long (1u), MM9/MM10 current
source is very long (11.3u), and PMOS/NMOS cascode/load devices MM5/MM6 and MM7/MM8 are
modest. The DC gain is excessive because ro of input pair and PMOS load are large. Need
to reduce device lengths, especially ro of signal path devices ... Goal is to lower ro
to reduce Rout, improving -3dB bandwidth while maintaining adequate DC gain.
</think>
<action>decrease @10 l_MM30 70%</action>
<action>increase @10 w_MM30 10%</action>
<action>decrease @15 l_MM12 60%</action>
<action>increase @15 w_MM12 20%</action>
<action>decrease @16 l_MM910 80%</action>
<action>increase @16 w_MM910 50%</action>
<action>increase @40 w_MM56 100%</action>
<action>decrease @40 l_MM56 20%</action>
<action>increase @50 w_MM78 100%</action>
<action>decrease @50 l_MM78 15%</action>
<action>decrease @70 iref 10%</action>
```

Simulator Result After Turn 3

Gain / power	75.9 dB / 169 μW
Bandwidth / phase margin	37 kHz / 52°

Bandwidth increases from 15 to 37 kHz following the combined length, width, and bias updates. It remains below the target even with the 5% evaluation tolerance. The other three targets remain met.

Turn 4 Observation Summary

```
(Observation: <GainAt1kHz>75.9 dB</GainAt1kHz><Power>169 uW</Power><Bandwidth3dB>37 kHz
 </Bandwidth3dB><PhaseMargin>52 deg</PhaseMargin>; input pair w=22.9u l=300n Ids=7.3 uA
 Gm=160 uS; MM1 w=7.4u l=400n; MM10 w=3.9u l=2.3u; MM5 w=5.6u l=192n; MM7 w=2.8u l=204n;
 iref=990n)
```

Turn 4 Original LLM Output Excerpt

```
<think>
Current specs: Gain is 75.9 dB (well above 61.7 dB target), Power is only 169 uW (well
within 425.7 uW), Phase Margin is 52 deg (meets 30-90 deg). The only unmet spec is
Bandwidth: 37 kHz requires >=43.1 kHz. The main bottleneck is the DC gain, dominated by
ro of signal path devices. Need to reduce ro of MM0/MM3 (input pair), MM1/MM2 PMOS load,
MM9/MM10 current source, MM5/MM6 PMOS cascode, MM7/MM8 NMOS cascode. Strategy: decrease
device lengths, especially l_MM30, l_MM12, l_MM910, l_MM56, l_MM78. Also improve iref
slightly for more current.
</think>
<action>decrease @10 l_MM30 50.0%</action>
<action>decrease @15 l_MM12 40.0%</action>
<action>increase @15 w_MM12 10.0%</action>
<action>decrease @16 l_MM910 55.0%</action>
<action>increase @16 w_MM910 15.0%</action>
<action>decrease @40 l_MM56 25.0%</action>
<action>increase @40 w_MM56 10.0%</action>
<action>decrease @50 l_MM78 30.0%</action>
<action>increase @50 w_MM78 15.0%</action>
<action>increase @70 iref 15.0%</action>
```

Final Simulator Result After Turn 4

Gain / power	68.4 dB / 318 μW
Bandwidth / phase margin	172 kHz / 56.4°
In-band peaking	0.0 dB

All four original targets are satisfied without relying on the evaluation tolerance. The episode terminates after four turns.

L.2 Case 2: Out-of-Distribution Current-Mirror Amplifier

Figure 12 shows the current-mirror amplifier used in this case. The second task starts with a gain of -123.2 dB and 61.2 dB of in-band peaking. The input pair and PMOS output mirror are in triode, and bandwidth is not measurable. After two turns, the policy reaches a valid operating point with sufficient gain and bandwidth. Two further updates reduce power and improve phase margin while retaining enough gain to meet the target.

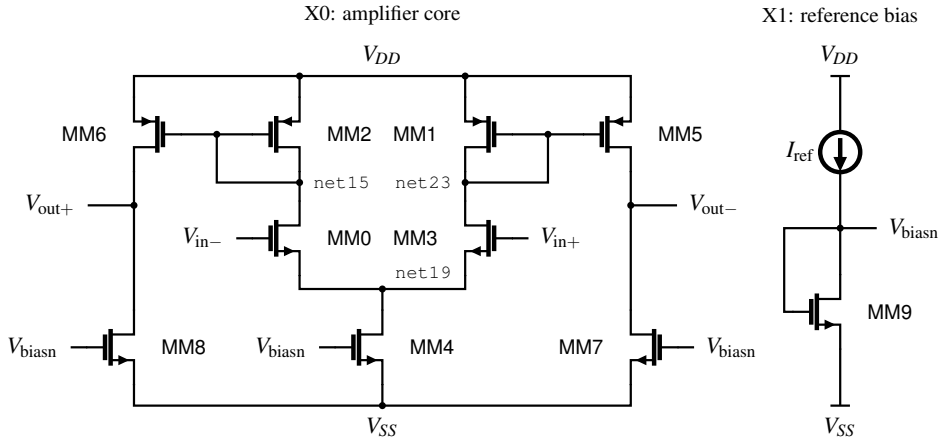


Figure 12: Fully differential current-mirror amplifier used in Case 2. X0 contains the amplifier and X1 supplies its shared bias voltage. Device identifiers and connections follow the supplied netlist. Identical bias labels denote the same electrical net. In both schematics, $V_{\mathrm{in}+}/V_{\mathrm{in}-}$ and $V_{\mathrm{out}+}/V_{\mathrm{out}-}$ correspond to `Vinp`/`Vinn` and `Voutp`/`Voutn`, respectively.

Task Setup

Topology	Fully-differential current-mirror amplifier
Evaluation split	Out-of-distribution
Technology	65 nm CMOS (PTM), $V_{DD} = 1.2$ V
Gain target	≥ 26.4 dB
Power target	≤ 170.1 μW
Bandwidth target	≥ 1.25 MHz (-3 dB bandwidth)
Phase-margin target	$30^\circ \leq \mathrm{PM} \leq 90^\circ$
Evaluation tolerance	5%, as defined in the main evaluation
Initial gain	-123.2 dB
Initial power	52 μW
Initial bandwidth	Not measurable
Initial phase margin	Unavailable
Initial in-band peaking	61.2 dB
Initial validity	Invalid under the domain-validity gates.

Structured System Input Excerpt

```
<GRAPH_INFO>
TOPOLOGY_NAME: Fully_differential_current_mirror
</GRAPH_INFO>
<GRAPH_NODE_IDMAP>
type=pin: @0=vdd @1=vss
type=current_source: @55=X1.I0
type=n_mosfet: @22=X0.MM3 @27=X0.MM4 @42=X0.MM7 @47=X0.MM8 @58=X1.MM9 @7=X0.MM0
type=p_mosfet: @12=X0.MM1 @17=X0.MM2 @32=X0.MM5 @37=X0.MM6
type=net: @2=LG_Vbiasn @3=Vinn @4=Vinp @5=Voutn @6=Voutp ...
</GRAPH_NODE_IDMAP>
<PINMAP> ... </PINMAP>
<EDGES> ... </EDGES>

Design bounds:
For tech 65nm, length l must stay within [45nm, 100um] and width w must stay
within [45nm, 500um]. Designs outside this range are invalid.
Fixed global: vdd=1.2V (user-set, not tunable).

Allowed valid target handles and parameter tokens:
  @7: l_MM30, w_MM30      (MM0 -- NMOS input pair, matched with @22)
  @12: l_MM21, w_MM21     (MM1 -- PMOS diode load, matched with @17)
  @27: l_MM4, w_MM4       (MM4 -- NMOS tail)
  @32: l_MM56, w_MM56     (MM5 -- PMOS output mirror, matched with @37)
  @42: l_MM78, w_MM78     (MM7 -- NMOS output sink, matched with @47)
  @55: iref               (I0 -- bias current source)
  @58: l_MM9, w_MM9       (MM9 -- NMOS bias diode)
```

Turn 1 User Input Excerpt

```
Your goal is: Achieve gain >= 26.4 dB, power <= 170.1 uW,
-3dB bandwidth >= 1.25 MHz,
30.0 deg <= phase_margin <= 90.0 deg
(each spec has 5% tolerance). Prioritize gain first.
[...]

Your current observation is:
<PERFORMANCE>
<GainAt1kHz>-123.2 dB</GainAt1kHz>
<Power>52 uW</Power>
<Bandwidth3dB>not measurable</Bandwidth3dB>
<InBandFlatness>61.2 dB (limit 6)</InBandFlatness>
<Validity>non-positive gain; power/bandwidth/PM not meaningful until gain >
  0</Validity>
</PERFORMANCE>
<FEATURES>
@7 id=X0.MM0 type=n_mosfet w=w_MM30=462.5n l=l_MM30=185.9n Vgs=647 mV
    Vds=970 uV Vdsat=132 mV Ids=188 nA Gm=1.3 uS region=triode
@12 id=X0.MM1 type=p_mosfet w=w_MM21=1.6u l=l_MM21=20u Vgs=-646 mV
    Vds=-646 mV Vdsat=-276 mV Ids=-188 nA Gm=1.2 uS region=sat
@27 id=X0.MM4 type=n_mosfet w=w_MM4=1.8u l=l_MM4=20u Vgs=522 mV
    Vds=553 mV Vdsat=127 mV Ids=428 nA Gm=5.5 uS region=sat
@32 id=X0.MM5 type=p_mosfet w=w_MM56=22u l=l_MM56=417.4n Vgs=-646 mV
    Vds=-32.0 uV Vdsat=-277 mV Ids=-30.5 nA Gm=102 nS region=triode
@42 id=X0.MM7 type=n_mosfet w=w_MM78=58.7n l=l_MM78=7.9u Vgs=522 mV
    Vds=1.2 V Vdsat=127 mV Ids=29.9 nA Gm=381 nS region=sat
@55 id=X1.I0 type=current_source dc=iref=8.2u
@58 id=X1.MM9 type=n_mosfet w=w_MM9=196.1n l=l_MM9=140.5n Vgs=522 mV
    Vds=522 mV Vdsat=118 mV Ids=8.1 uA Gm=93.6 uS region=sat
</FEATURES>
```

Turn 1 Original LLM Output Excerpt

```
<think>
The current operating point is invalid: DC gain is -123.2 dB (magnitude only
0.123 dB), power 52 uW is fine but bandwidth is not measurable because the
low-frequency gain is essentially zero. The main problems are:
1. PMOS load MM1/MM2 is very weak: W/L only 1.6u/20u = 0.08. Needs stronger PMOS
   current source.
2. NMOS pair MM3 is too weak: W/L only 462.5n/185.9n = 2.49, needs wider input
   pair.
3. Tail MM4 is poorly sized: W/L only 1.8u/20u = 0.09, should provide more tail
   current.
4. PMOS current source MM5/MM6 is too wide/triode: 22u/417n acts as resistor
   rather than DC current source.
5. NMOS load MM7/MM8 is too weak: W=58.7n, L=7.9u gives poor DC level shifting.
   Need stronger NMOS load.
6. iref can be reduced later, but first establish DC bias. Strategy: Make MM3
   strong pair, MM1/MM2 moderate PMOS load, stronger tail, better PMOS current
   source, stronger NMOS load, moderate iref.
</think>
<action>increase @22 w_MM30 500%</action>
<action>decrease @22 l_MM30 20%</action>
<action>increase @12 w_MM21 500%</action>
<action>decrease @12 l_MM21 99%</action>
<action>increase @27 w_MM4 500%</action>
<action>decrease @27 l_MM4 99%</action>
<action>decrease @32 w_MM56 95%</action>
<action>increase @32 l_MM56 100%</action>
<action>increase @42 w_MM78 500%</action>
<action>decrease @42 l_MM78 99%</action>
<action>decrease @55 iref 50%</action>
<action>increase @58 w_MM9 100%</action>
<action>increase @58 l_MM9 50%</action>
```

Simulator Result After Turn 1

Gain / power	-118.4 dB / 25 μW
Bandwidth / phase margin	Not measurable
In-band peaking	39.5 dB

The first update does not restore valid operation. Gain remains non-positive and in-band peaking exceeds the validity threshold. The simulator again returns a `design_error`.

Turn 2 Observation Summary

```
(observation: INVALID - design_error, gain -118.4 dB, power 25 uW,
 in-band flatness 39.5 dB; sizes after Turn 1: MM0/MM3 2.8u/148.7n, MM1/MM2
   9.6u/200n,
 MM4 10.8u/200n, MM5/MM6 1.1u/834.8n, MM7/MM8 352.2n/79n, MM9 392.2n/210.8n,
   iref 4.1u)
```

Turn 2 Original LLM Output Excerpt

```
<think>
The current operating point is invalid: DC gain is -118.4 dB (magnitude only
0.118 V/V). The DC gain polarity and magnitude are wrong. The main problems are
the bias: PMOS load MM1/MM2 is too strong compared to NMOS pair MM0/MM3, tail
MM4 current is moderate, PMOS current source MM5/MM6 and NMOS current mirror
MM7/MM8 are not properly sized. The DC operating point is such that Vout is
wrong. Need to restore DC: make NMOS pair MM0/MM3 stronger (larger W, longer L),
PMOS load MM1/MM2 weaker (smaller W, longer L), stronger tail MM4, stronger PMOS
current source MM5/MM6, and better NMOS current mirror MM7/MM8. iref can be
reduced later for power.
</think>
<action>increase @22 w_MM30 500.0%</action>
<action>increase @22 l_MM30 200.0%</action>
<action>decrease @12 w_MM21 80.0%</action>
<action>increase @12 l_MM21 400.0%</action>
<action>increase @27 w_MM4 200.0%</action>
<action>decrease @27 l_MM4 10.0%</action>
<action>increase @32 w_MM56 300.0%</action>
<action>decrease @32 l_MM56 50.0%</action>
<action>increase @42 w_MM78 500.0%</action>
<action>increase @42 l_MM78 100.0%</action>
<action>decrease @55 iref 20.0%</action>
<action>decrease @58 w_MM9 20.0%</action>
<action>decrease @58 l_MM9 20.0%</action>
```

Simulator Result After Turn 2

Gain / power	40.2 dB / 224 μW
Bandwidth / phase margin	6.9 MHz / 8.5°
In-band peaking	0.0 dB

The design becomes valid and exceeds the gain and bandwidth targets. Power remains above the budget, and phase margin is below the accepted range.

Turn 3 Observation Summary

```
(observation: gain 40.2 dB, power 224 uW, bandwidth 7 MHz, PM 8 deg; MM0/MM3
  16.8u/446.1n
 Ids=4 uA, MM5/MM6 4.4u/417.4n Ids=-24.2 uA, MM7/MM8 2.1u/158n Ids=24.2 uA, iref
   3.3u)
```

Turn 3 Original LLM Output Excerpt

```
<think>
Current specs: Gain is 40.2 dB (well above 26.4 dB target), BW is 7 MHz (meets
1.25 MHz requirement), but Power is 224 uW (exceeds 170.1 uW) and Phase Margin
is only 8 deg (needs 30-90 deg). The main issues are excessive bias current and
over-sized PMOS current source/load devices causing too much unity-gain
frequency phase lag. Need to reduce iref, scale down MM5/MM6 PMOS current
source, MM7/MM8 NMOS load, and iref/MM9 bias to save power and improve phase
margin while maintaining gain and bandwidth.
</think>
<action>decrease @55 iref 65.0%</action>
<action>decrease @32 w_MM56 60.0%</action>
<action>decrease @32 l_MM56 40.0%</action>
<action>decrease @42 w_MM78 50.0%</action>
<action>decrease @42 l_MM78 20.0%</action>
<action>decrease @27 w_MM4 40.0%</action>
<action>decrease @27 l_MM4 15.0%</action>
<action>decrease @12 w_MM21 20.0%</action>
<action>decrease @12 l_MM21 20.0%</action>
<action>decrease @58 w_MM9 30.0%</action>
<action>increase @58 l_MM9 10.0%</action>
```

Simulator Result After Turn 3

Gain / power	36.2 dB / 90 μW
Bandwidth / phase margin	4.8 MHz / 23.6°

The combined bias and sizing updates bring power within the budget and improve phase margin from 8.5° to 23.6°. Only the phase-margin specification remains unmet.

Turn 4 Observation Summary

```
(observation: gain 36.2 dB, power 90 uW, bandwidth 5 MHz, PM 24 deg; MM0/MM3
  16.8u/446.1n
 Ids=2.1 uA, MM5/MM6 1.8u/250.4n Ids=-9.4 uA, MM7/MM8 1u/126.4n Ids=9.4 uA, iref
   1.2u)
```

Turn 4 Original LLM Output Excerpt

```
<think>
Current specs: Gain is 36.2 dB (well above 26.4 dB target), Power is only 90 uW
(well within 170.1 uW), Bandwidth is 5 MHz (exceeds 1.25 MHz requirement). The
only unmet spec is Phase Margin, which is 24 deg but requires 30.0 deg to 90.0
deg. The main issue is excessive DC gain due to large ro of signal path devices,
causing too much phase lag at unity-gain frequency. Need to reduce ro of
output/load devices to lower DC gain and improve phase margin while maintaining
gain and bandwidth. Strategy: reduce device lengths (MM0/MM3, MM1/MM2, MM5/MM6,
MM7/MM8) and iref to lower ro and DC current, decreasing DC gain and improving
phase margin. Decrease iref moderately to save power.
</think>
<action>decrease @7 w_MM30 50.0%</action>
<action>decrease @7 l_MM30 40.0%</action>
<action>decrease @12 w_MM21 30.0%</action>
<action>decrease @12 l_MM21 40.0%</action>
<action>decrease @32 w_MM56 50.0%</action>
<action>decrease @32 l_MM56 40.0%</action>
<action>decrease @42 w_MM78 40.0%</action>
<action>decrease @42 l_MM78 30.0%</action>
<action>decrease @55 iref 30.0%</action>
<action>decrease @27 w_MM4 30.0%</action>
<action>decrease @27 l_MM4 15.0%</action>
<action>decrease @58 w_MM9 20.0%</action>
<action>decrease @58 l_MM9 10.0%</action>
```

Final Simulator Result After Turn 4

Gain / power	28.4 dB / 78.8 μW
Bandwidth / phase margin	10.2 MHz / 46.8°
In-band peaking	0.0 dB

All four original targets are satisfied without relying on the evaluation tolerance. Gain remains 2.0 dB above the target, while phase margin reaches 46.8° and power remains within the budget. The episode terminates after four turns.

These examples illustrate how simulator feedback can guide recovery from invalid operation and subsequent refinement of coupled performance metrics. The two rollouts demonstrate successful optimization from different initial conditions and specification targets on an in-distribution topology and a held-out topology.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions stated in the abstract and introduction (Section 1) are empirically supported by Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Appendix J.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We described the experiment details and setup in Section 4.1, with SFT and RL hyperparameters in Table 16 and Table 17, circuit representation and action grammar in Appendix B, reward details in Appendix C, and the RGRS procedure in Algorithm 1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification:The project page (`https://slic-agent.github.io/`) links to the code, benchmark data, and instructions for reproducing the main PTM experiments.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: We specify those details in Section 4.1 and Appendix I.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report means and standard deviations over three evaluation runs, except for GPT-6-Astra, which is evaluated once. Appendix H.5 provides task-paired bootstrap confidence intervals and significance tests with Holm correction. Due to limited computational resources, these analyses are based on a single training run per method and do not capture variability across training seeds.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section 4.1 and Tables 16–17

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We fully adhere to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed the potential societal impacts in Appendix K.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly credit all existing assets used in this work, including model, tools, code and data.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Appendices I.1 and B document the benchmark splits, circuit representation, and action format.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper explicitly describes the use of LLMs/SLMs as core components of the method, including their roles as reinforcement learning backbones, teachers for supervised fine-tuning and evaluation baselines.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.