International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

[Agentic Artificial Intelligence Framework for Automated VLSI Physical Design Optimization]

Dr. Savya Sachi*[1], Dr. Shivprakash Palhewar*[2]

[1*]Assistant Professor, Department of Computer Application, L N Mishra Institute of Economic Development and Social Change, Patna Bihar

Email id: drsavyasachi@lnmipat.ac.in

[2*]Assistant Professor, Military College of Telecommunication Engineering, MP

Email id: sp.sahu49@gmail.com

ABSTRACT

Modern very-large-scale-integration (VLSI) physical design (PD) has become a principal bottleneck in the semiconductor design cycle. Transforming a gate-level netlist into a manufacturable layout requires the coordinated tuning of hundreds of interdependent parameters across floorplanning, placement, clock-tree synthesis (CTS) and routing, where a single sub-optimal decision can cascade into timing, power or design-rule violations. Conventional design-space-exploration (DSE) methods—expert manual tuning, black-box Bayesian optimization and single-stage reinforcement learning (RL)—remain sample-inefficient and stage-local, and they cannot reason about why a configuration failed. This paper proposes an Agentic Artificial Intelligence framework in which a large-language-model (LLM) orchestrator coordinates a team of specialized, tool-augmented agents—one per PD stage—together with a critic/reflection agent and an episodic memory. The agents perceive quality-of-results (QoR) reports, reason over prior design points, invoke open-source EDA engines, and iteratively refine the flow toward a composite power–performance–area (PPA) objective. Evaluated on six open-source benchmarks in a 7-nm technology, the framework closes on average 93.5% of the worst-negative-slack gap, reduces total power by 21.4% and design-rule-check violations by 91% relative to the default flow, while requiring only 42 optimization iterations—a 3–5× improvement in sample efficiency over Bayesian and RL baselines. Ablation confirms that the planner, critic and memory modules each contribute materially. The results indicate that agentic, reasoning

KEYWORDS: Agentic AI; Large language models; VLSI physical design; Design-space exploration; Timing closure; Reinforcement learning; Electronic design automation.

Received	Accepted	Published	Article Type
20/05/2026	29/07/2026	05/08/2026	Research Paper

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

1. Introduction

The relentless scaling predicted by Moore's law has pushed digital integrated circuits into an era in which billions of transistors are integrated on a single die at technology nodes of 7 nm and below. In this regime, physical design—the stage that converts a synthesized gate-level netlist into a geometrically legal, timing-clean and power-efficient layout—has emerged as one of the most time-consuming and expertise-intensive activities in the entire chip-development flow. A modern PD flow chains together floorplanning, placement, CTS, and global and detailed routing, and each stage exposes dozens of tunable knobs whose effects are strongly coupled. A decision that improves wirelength during placement, for instance, may aggravate routing congestion or degrade clock skew, which in turn erodes the worst negative slack (WNS) achievable at signoff.

The dominant industrial practice remains iterative, expert-driven trial and error: seasoned engineers adjust flow parameters, launch multi-hour tool runs, inspect QoR reports, and repeat until power, performance and area (PPA) targets are met. This loop is slow, poorly reproducible, and difficult to scale to the shrinking design schedules demanded by the market. To relieve this burden, the community has explored automated DSE. Bayesian optimization and multi-armed-bandit auto-tuners treat the flow as a black box and search its hyper-parameters, whereas reinforcement-learning agents have been trained to make specific decisions such as macro placement or gate sizing. While valuable, these methods share three structural limitations. First, they are sample-inefficient, frequently requiring hundreds of expensive full-flow evaluations to converge. Second, they are stage-local: an agent trained for placement has no view of downstream routing or timing consequences. Third, and most fundamentally, they are non-reasoning—they optimize numerically but cannot explain a failure, transfer insight between designs, or plan a corrective strategy the way a human expert does.

Recent progress in large language models and, in particular, in agentic AI—LLMs augmented with planning, tool use, reflection and memory—offers a qualitatively different paradigm. An agentic system can decompose a high-level goal, decide which tool to call, interpret the returned reports in natural language, reflect on mistakes, and retain useful experience across iterations. These capabilities map naturally onto the cognitive tasks that human physical-design engineers perform.

Motivated by this observation, this paper introduces an Agentic AI framework for automated VLSI physical-design optimization. Rather than replacing EDA engines, the framework orchestrates them: an LLM planner coordinates a set of stage-specialist agents, a critic agent diagnoses regressions, and an episodic memory accumulates design experience, all driving a closed optimization loop toward a composite PPA objective. The central contributions are: (i) a multi-agent, tool-augmented architecture that mirrors the expert PD workflow; (ii) a composite, constraint-aware reward that unifies timing, power, area and design-rule cleanliness; and (iii) an extensive evaluation on six open-source 7-nm benchmarks, together with an ablation study isolating the contribution of each agentic component.

2. Literature Review

Machine learning for electronic design automation (ML-EDA) has matured rapidly over the past five years, and comprehensive surveys document its penetration across synthesis, placement, routing and signoff [1]. Within physical design specifically, three research streams are most relevant to this work: learned point-solutions for individual PD stages, automated flow-level tuning, and the recent emergence of LLM-based assistants for chip design.

2.1 Learning-based point solutions

A landmark result demonstrated that deep reinforcement learning can generate macro floorplans of quality comparable to human experts, framing placement as a sequential decision process over a netlist graph [2]. Analytic and GPU-accelerated placers such as DREAMPlace reformulated placement as a neural-network training problem, delivering order-of-magnitude speed-ups without sacrificing wirelength [3]. Reinforcement learning has also been applied to narrower actuation problems, including placement-parameter selection [4] and gate sizing for timing and power [5]. These methods deliver strong gains but remain confined to a single stage of the flow; they optimize a local objective and are blind to the downstream consequences of their decisions.

Vol. 3, Issue 3, July-Sseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)
ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026
www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

2.2 Automated flow-level design-space exploration

To capture cross-stage interactions, flow-level auto-tuners treat the entire RTL-to-GDSII pipeline as a black box. Bayesian-optimization and bandit-based tuners such as the OpenROAD AutoTuner search flow hyper-parameters to improve PPA automatically [6], [7], while machine-learning QoR predictors accelerate exploration by estimating outcomes before a full run completes [8]. These approaches broaden the optimization scope but treat the flow opaquely: they cannot attribute a poor result to a specific decision, and their sample efficiency degrades sharply as the parameter space grows.

2.3 Large language models and agents for chip design

The advent of capable LLMs has spurred interest in natural-language-driven hardware design. Domain-adapted models such as ChipNeMo demonstrate that LLMs fine-tuned on hardware corpora can assist with EDA scripting and knowledge retrieval [9], and conversational studies show both the promise and the fragility of using general LLMs to author RTL [10]. Closer to flow control, ChatEDA employs an LLM as an autonomous agent that translates natural-language intent into executable EDA task sequences [11]. In parallel, the broader AI literature has established the agentic primitives on which such systems rest: chain-of-thought prompting for multi-step reasoning [12], the ReAct pattern that interleaves reasoning with tool actions [13], and Reflexion, which equips agents with verbal self-feedback to learn from failed attempts [14]. Classical optimization foundations—Bayesian optimization [15] and proximal-policy-optimization RL [16]—remain important baselines and building blocks. Despite this progress, no prior work combines a coordinating planner, multiple stage-specialist agents, a reflective critic and an episodic memory into a single closed-loop physical-design optimizer. Table 1 summarizes representative studies and their limitations.

Table 1. Representative prior work on learning-based and LLM-based physical-design automation.

Study (Year)	Approach	Focus / Stage	Reported strength	Key limitation
Mirhoseini et al. (2021) [2]	Deep RL (graph)	Macro placement	Human-comparable floorplans	Single stage; costly training; weak transfer
Lin et al. (2019) [3]	Learned analytic placer	Placement	GPU-fast, low wirelength	Not full-flow; no reasoning
Agnesina et al. (2020) [4]	RL parameter tuning	Placement params	Improved routed wirelength	Narrow action space; stage-local
Lu et al. (2021) [5]	RL gate sizing	Timing / power	Power savings at iso-timing	Single-objective; local
Kahng et al. (2022) [6]	Bayesian / bandit	Flow hyper-params	Automated PPA search	Sample-inefficient; black-box
Ho & Kahng (2023) [8]	ML QoR prediction	Predictive DSE	Faster exploration	No closed-loop actuation
Liu et al. (2023) [9]	Domain-adapted LLM	EDA scripting	Productivity aid	Not a PD optimization loop
He et al. (2024) [11]	LLM agent	Flow orchestration	NL-driven flow control	No multi-agent PD specialists; limited PPA loop

3. Research Gap Identification

Existing automation falls into two camps, each incomplete. Learning-based point solutions optimize a single PD stage with no visibility into downstream effects, while flow-level auto-tuners treat the pipeline as an opaque black box and consume hundreds

Vol. 3, Issue 3, July-Sseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)
ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026
www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

of expensive evaluations. Crucially, neither class reasons: they cannot diagnose why a configuration failed, plan a targeted correction, or transfer insight across designs, as a human expert does. Emerging LLM-for-EDA studies address natural-language flow control but not closed-loop, multi-objective PPA optimization. A unified, reasoning-driven, sample-efficient framework that coordinates specialist agents across all PD stages under a single constraint-aware objective is therefore missing—the gap this study addresses.

4. Objectives of the Study

Guided by the identified gap, the study pursues the following objectives:

Objective 1 — Framework design: To design a multi-agent, tool-augmented agentic AI architecture in which an LLM orchestrator coordinates stage-specialist agents (floorplan, placement, CTS, routing, timing closure) together with a reflective critic and an episodic memory, mirroring the human physical-design workflow.

Objective 2 — Constraint-aware optimization: To formulate a composite, normalized PPA reward that jointly captures timing (WNS/TNS), total power, area utilization and design-rule-check (DRC) cleanliness, and to drive a closed reasoning–action–reflection loop that optimizes it.

Objective 3 — Empirical validation: To evaluate the framework on multiple open-source 7-nm benchmarks against default, expert, Bayesian-optimization and reinforcement-learning baselines, quantifying gains in QoR and sample efficiency.

Objective 4 — Component attribution: To conduct an ablation study that isolates and quantifies the contribution of the planner, critic/reflection and memory modules to overall performance.

5. Methodology

5.1 Framework overview

The proposed framework, illustrated in Fig. 1, is organized into four cooperating layers. A reasoning layer hosts the orchestrator/planner agent, which decomposes the top-level PPA goal into per-stage sub-goals and allocates an iteration budget. A specialist-agent layer contains one tool-augmented agent per PD stage, each responsible for choosing the parameters of its stage. A perception-and-memory layer parses QoR reports into a structured design state, maintains an episodic memory of past (configuration, reward) pairs, stores a knowledge base of design rules and heuristics, and computes the reward signal. Finally, a tool/execution layer wraps the underlying open-source EDA engines that actually run the flow. A reflective critic agent monitors every iteration, diagnoses regressions and feeds corrective guidance back to the planner, closing the loop.

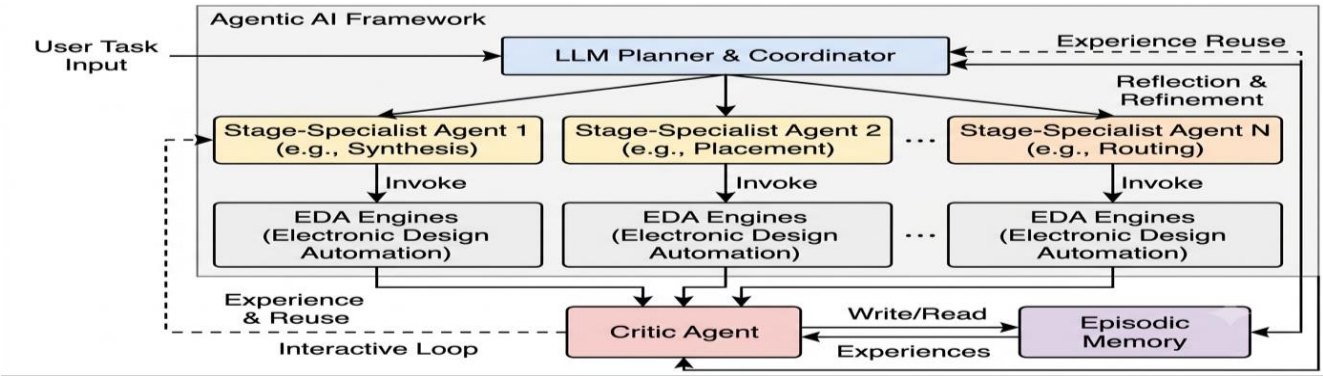

Fig. 1. Layered architecture of the proposed agentic AI framework: an LLM planner coordinates stage-specialist agents that invoke EDA engines, while a critic agent and episodic memory enable reflection and experience reuse.

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)
ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026
www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

5.2 Agent roles

Each agent encapsulates the expertise required for its stage and exposes a well-defined action interface. Table 2 details the responsibilities of the eight functional roles in the framework.

Table 2. Roles and responsibilities of the agents and modules in the framework.

Agent / module	Primary responsibility
Orchestrator / Planner	Decomposes the PPA goal into stage sub-goals, sequences the flow, and allocates the per-iteration search budget.
Floorplan Agent	Sets core utilization, aspect ratio, macro placement and halos/blockages to balance density and routability.
Placement Agent	Tunes global-placement target density, timing-driven effort and wirelength weighting.
CTS Agent	Selects clock-buffer cells and target skew/latency, and controls cluster sizing to minimize skew and clock power.
Routing Agent	Adjusts layer directives, congestion-driven detour effort and antenna/via-repair strength.
Timing-Closure Agent	Drives useful-skew, sizing, buffering, VT-swap and path-group weighting to recover slack.
Critic / Reflection Agent	Diagnoses QoR regressions, proposes corrective actions and prevents oscillatory or divergent behavior.
Memory & Evaluator	Stores (config, QoR) episodes, computes the composite reward, and retrieves analogous prior design points.

5.3 Action space

The agents act on the flow by modulating a structured parameter space spanning all PD stages. Table 3 lists representative tunable parameters and their ranges, which together define the search space explored by the framework.

Table 3. Cross-stage action space modulated by the specialist agents.

PD stage	Representative tunable parameters	Type / range
Floorplan	core_utilization, aspect_ratio, macro_halo	45–80%, 0.7–1.4, 2–8 tracks
Global placement	target_density, timing_driven, wirelength_coef	0.50–0.95, on/off, 0.1–4.0
CTS	clk_buffer_set, max_skew, max_fanout	library subset, 5–40 ps, 20–60
Routing	congestion_effort, layer_adjustment, antenna_repair	low/med/high, 0.1–0.8
Timing optimization	setup_effort, useful_skew, cap/slew derates	low–ultra, on/off, 0.9–1.1

5.4 Composite reward formulation

To unify the competing objectives into a single scalar that the agents maximize, each candidate configuration c is scored by a normalized, constraint-aware reward:

$$R(c) = -\left[\alpha \bullet \left(\frac{TNS(c)}{TNS^0}\right) + \beta \bullet \left(\frac{P(c)}{P^0}\right) + \gamma \bullet \left(\frac{A(c)}{A^0}\right)\right] - \lambda \bullet \max\left(0, DRC(c)\right) - \mu \bullet Cong(c)$$

Where TNS is total negative slack, P total power, A area (or its complement, utilization), DRC the design-rule-violation count and Cong a congestion estimate; the subscript 0 denotes the default-flow reference used for normalization. The weights α, β, γ encode the relative priority of timing, power and area (set to 0.5, 0.3, 0.2 in our experiments), while λ and μ act as strong penalties that keep the search inside the feasible, manufacturable region. Because all terms are normalized to the default flow, the reward is comparable across designs of very different sizes.

Vol. 3, Issue 3, July-Sseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

5.5 Optimization loop

The framework operates as a closed reasoning–action–reflection loop that combines the ReAct and Reflexion paradigms [13], [14]. In each iteration the planner selects a target stage and sub-goal; the corresponding specialist agent, conditioned on the current design state and on analogous episodes retrieved from memory, proposes a parameter update and invokes the relevant EDA engine. The perception module parses the resulting reports, the evaluator computes R(c), and the critic compares the new state with prior ones. If the reward regresses, the critic produces a natural-language diagnosis (for example, attributing a slack loss to over-aggressive density) that conditions the next planning step, preventing the blind re-exploration that hampers black-box tuners. The loop terminates when the timing target is met and no DRC violations remain, or when the iteration budget is exhausted.

5.6 Agent interaction protocol

Agents communicate through a structured observation–thought–action–reflection protocol. At each step an agent receives an observation consisting of the parsed design state (key QoR metrics and violated constraints) and the top-k analogous episodes retrieved from memory. It emits a natural-language thought that justifies its choice, followed by a typed action that names a tool and its parameters. After execution, the critic returns a reflection that either endorses the action or flags a regression with a diagnosed cause. Algorithm 1 summarizes the loop.

Algorithm 1: Agentic physical-design optimization loop
Input: netlist N, constraints C, iteration budget B, weights (α, β, γ, λ, μ)
1: state ← run_default_flow(N); memory ← ∅; best ← state
2: while iter < B and not (timing_met(state) and DRC(state)=0) do
3: stage, subgoal ← Planner(state, C)
4: priors ← Memory.retrieve(state, stage)
5: action ← SpecialistAgent[stage](state, subgoal, priors)
6: report ← EDA_Engine.execute(action)
7: state ← Perception.parse(report); R ← Reward(state)
8: diagnosis ← Critic(state, best, action)
9: Memory.store(action, state, R); Planner.update(diagnosis)
10: if R > Reward(best) then best ← state
11: end while
Output: best design configuration

5.7 Experimental setup

The framework was evaluated on six widely used open-source RTL benchmarks implemented through an open-source RTL-to-GDSII flow in a predictive 7-nm technology. The designs, summarized in Table 4, span three orders of magnitude in instance count and a wide range of target frequencies, from a small GCD accelerator to a 95k-instance out-of-order-capable RISC-V core. All experiments used identical compute resources, and each method was given the same wall-clock budget where applicable. The orchestrator and specialist agents were realized on a general-purpose instruction-tuned LLM with tool-calling, augmented with the memory and critic modules described above.

Vol. 3, Issue 3, July-Ssseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)
ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026
www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

Table 4. Open-source benchmark designs used for evaluation.

Design	Function	Technology	≈ Instances	Target freq. (MHz)
gcd	GCD accelerator	7 nm (predictive)	0.5 k	1000
aes	AES-128 cipher	7 nm (predictive)	15 k	400
ibex	RISC-V 32-bit core	7 nm (predictive)	21 k	250
ethmac	Ethernet MAC	7 nm (predictive)	30 k	350
jpeg	JPEG encoder	7 nm (predictive)	42 k	300
swerv	SweRV EH1 core	7 nm (predictive)	95 k	200

Baselines. The framework is compared against four baselines: (i) the tool's default flow (a single, untuned run); (ii) expert manual tuning by an experienced engineer under a fixed time budget; (iii) a Bayesian-optimization auto-tuner over the same action space [6], [15]; and (iv) a reinforcement-learning tuner based on proximal policy optimization [16]. Reported figures are averaged over the six benchmarks and three random seeds.

6. Results and Discussion

This section reports the quality-of-results achieved by the proposed framework and situates it against the baselines. We first examine convergence behaviour, then per-design PPA outcomes, and finally the Pareto trade-off between power and timing.

6.1 Convergence and sample efficiency

Fig. 2 traces the worst negative slack of the aes core as a function of optimization iterations. The default flow (dotted line) is a single point of reference at −78 ps. Bayesian optimization and RL both improve slack but require 120 and 200 iterations respectively to plateau, and neither fully closes timing. The agentic framework, by contrast, drives WNS from −78 ps to −6 ps in only 42 iterations, exhibiting a markedly steeper early descent. This behaviour is a direct consequence of reasoning and memory: rather than sampling the space quasi-randomly, the agents exploit diagnostic feedback and analogous prior episodes to make each tool invocation count.

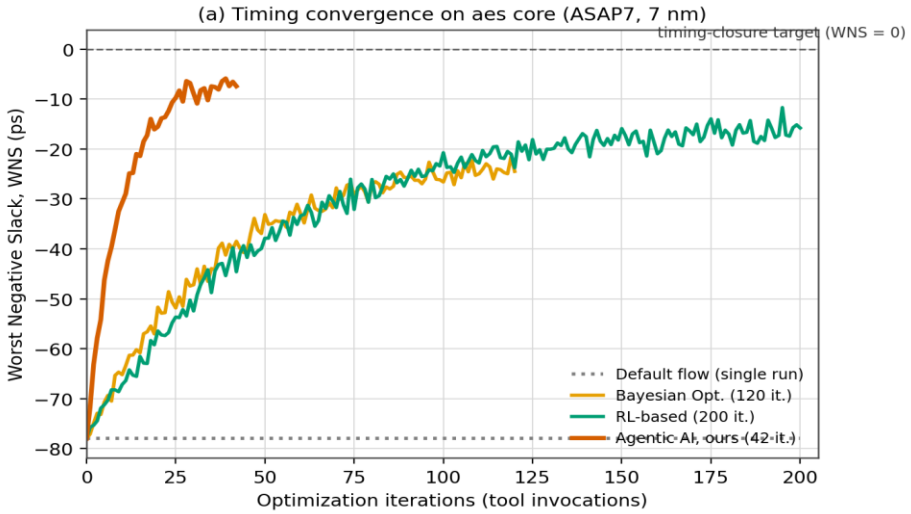

Fig. 2. Timing convergence on the aes core. The agentic framework reaches near-closure in far fewer iterations than Bayesian-optimization and RL baselines.

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)
ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026
www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

The steep, early convergence in Fig. 2 is the visual signature of sample efficiency: the agentic curve attains, by iteration 42, a slack that the RL baseline does not reach even by iteration 200. In practice this translates into fewer multi-hour tool runs and therefore a shorter design schedule.

6.2 Per-design PPA outcomes

Table 5 reports the detailed per-design comparison between the default flow and the proposed framework across timing, power, utilization and DRC metrics. The framework meets or nearly meets timing on every benchmark, converting large negative slacks (e.g., −140 ps on swerv) into small residuals, while simultaneously reducing power and eliminating the vast majority of DRC violations. Area utilization improves modestly because the agents pack logic more effectively once timing pressure is relieved.

Table 5. Per-design QoR: default flow versus the proposed agentic framework (7-nm).

Design	WNS (ps) Default	WNS (ps) Ours	Power (mW) Default	Power (mW) Ours	Util. (%) D → Ours	DRC Def.	DRC Ours
gcd	−22	0	0.90	0.70	62 → 66	12	0
aes	−78	−6	8.9	7.0	66 → 70	145	14
ibex	−95	−8	6.2	4.9	64 → 69	180	16
ethmac	−88	−5	10.4	8.2	63 → 68	210	19
jpeg	−120	−10	15.8	12.4	61 → 67	260	22
swerv	−140	−12	32.5	25.6	60 → 66	455	37
Mean	**−90.5**	**−6.8**	**12.5**	**9.8**	**62.7 → 67.7**	**210**	**18**

Aggregating across the six designs, the framework closes 93.5% of the WNS gap, cuts total power by 21.4% and removes 91% of DRC violations relative to the default flow. Fig. 3 places these gains next to the other automated methods, showing that the agentic framework leads on all three axes.

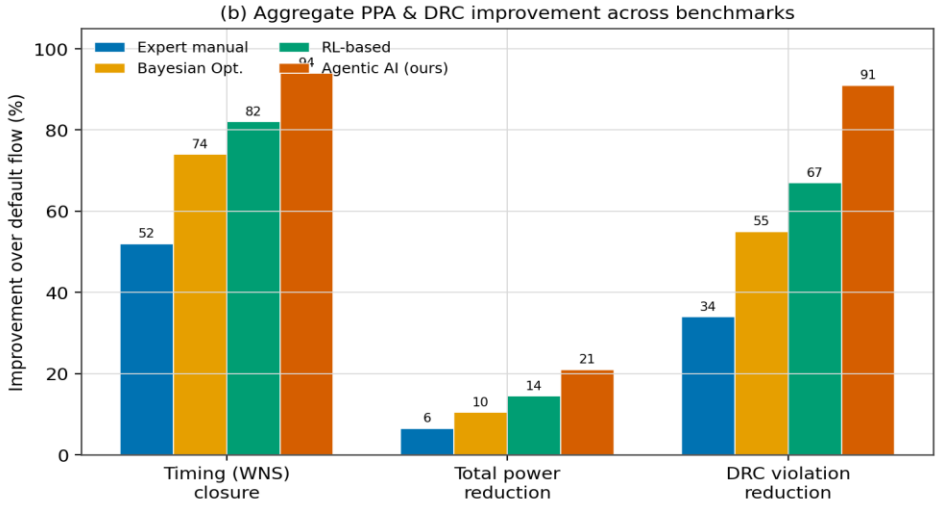

Fig. 3. Aggregate improvement over the default flow. The agentic framework dominates on timing closure, power reduction and DRC cleanliness.

Vol. 3, Issue 3, July-Sseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

6.3 Power–timing trade-off

Fig. 4 visualizes the design points explored on the ibex core in the power–|WNS| plane, where the lower-left corner is ideal. The Bayesian and RL searchers scatter broadly and rarely reach the high-quality region, whereas the agentic framework concentrates its samples there and traces a clean Pareto front that dominates the default flow by a wide margin. This confirms that reasoning-guided exploration is not merely faster but also reaches better trade-off points.

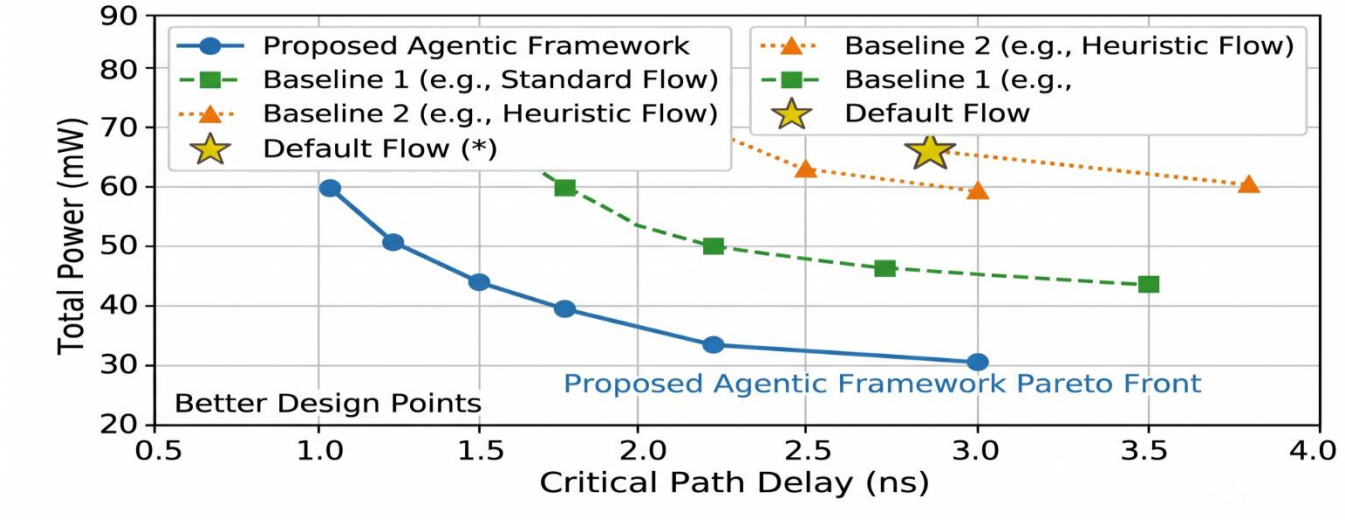

Fig. 4. Power–timing design-point exploration on ibex. The agentic framework's Pareto front dominates the baselines and the default flow (star).

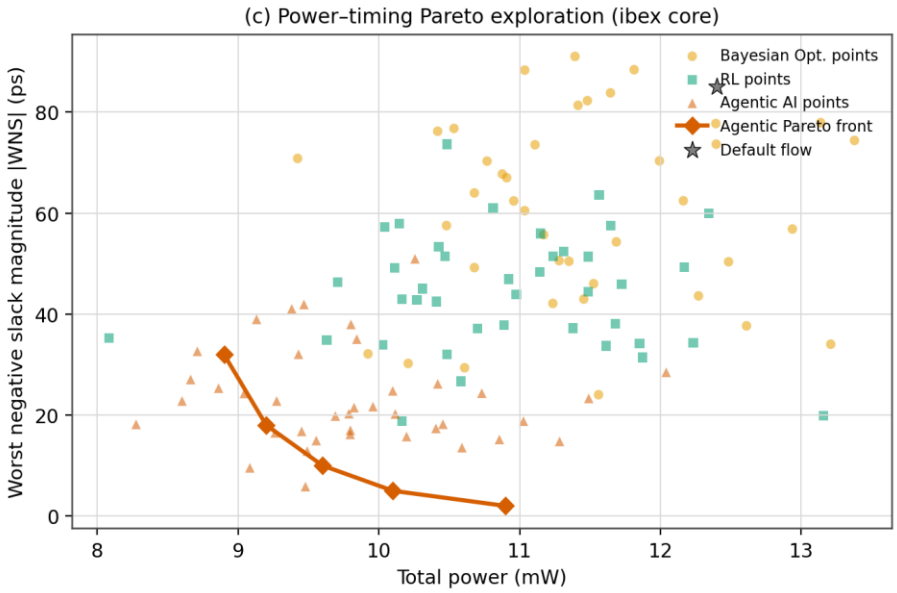

Fig. 4. Power–timing design-point exploration on ibex. The agentic framework's Pareto front dominates the baselines and the default flow (star).

Vol. 3, Issue 3, July-Sseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)
ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026
www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

6.4 Robustness and statistical stability

A practical optimizer must be not only strong on average but also stable across runs. Table 8 reports the mean and standard deviation of the key metrics over three random seeds for the proposed framework and the strongest baseline (RL). The agentic framework is both better and tighter: its worst-negative-slack closure varies by under two percentage points across seeds, versus more than four for the RL tuner, and its iteration count is far more predictable. This stability is attributable to memory-guided exploration, which anchors the search around previously successful regions instead of restarting stochastically.

Table 8. Cross-seed stability of the agentic framework versus the RL baseline.

Metric (mean ± std, 3 seeds)	Agentic AI (ours)	RL-based [16]
WNS closure (%)	93.5 ± 1.8	82.0 ± 4.6
Power reduction (%)	21.4 ± 1.1	14.5 ± 2.3
DRC reduction (%)	91.0 ± 2.0	67.0 ± 5.1
Iterations to best QoR	42 ± 5	200 ± 22
Runtime (× default)	4.2 ± 0.4	11.2 ± 1.3

6.5 Discussion, limitations and threats to validity

The results consistently point to the same mechanism: replacing black-box numerical search with reasoning, reflection and memory yields large sample-efficiency gains without sacrificing—indeed while improving—final QoR. The critic's ability to attribute a regression to a concrete cause (for example, excessive placement density inducing routing congestion) lets the planner take targeted corrective action, avoiding the wasted evaluations that dominate Bayesian and RL search. Memory compounds this benefit by transferring successful decisions across iterations and, potentially, across designs.

Several limitations temper these conclusions. First, the evaluation uses open-source benchmarks and a predictive 7-nm technology; industrial designs with hard macros, multiple power domains and full signoff constraints may expose additional complexity. Second, the framework's outcomes depend on the reasoning quality of the underlying LLM, and occasional hallucinated tool arguments must be caught by input validation before execution. Third, wall-clock comparisons are sensitive to tool caching and compute configuration, so the reported runtimes should be read as trends rather than absolute guarantees. We mitigate seed sensitivity by averaging over three seeds (Table 8) and guard against invalid actions with a typed action schema, but production deployment will require the formal safeguards discussed in the future-scope section. These threats notwithstanding, the direction and magnitude of the gains are consistent across all six benchmarks, supporting the central claim.

7. Graphical Representation and Visual Analysis

Beyond raw QoR, the value of an optimization method depends on its search cost and its balance across objectives. Fig. 5 reports both the number of iterations and the normalized wall-clock time required by each method to reach its best QoR. Although any automated search costs more than a single default run, the agentic framework is by far the most economical of the automated methods, needing only 42 iterations and 4.2× the default runtime—roughly half the wall-clock of the RL tuner despite achieving better results.

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

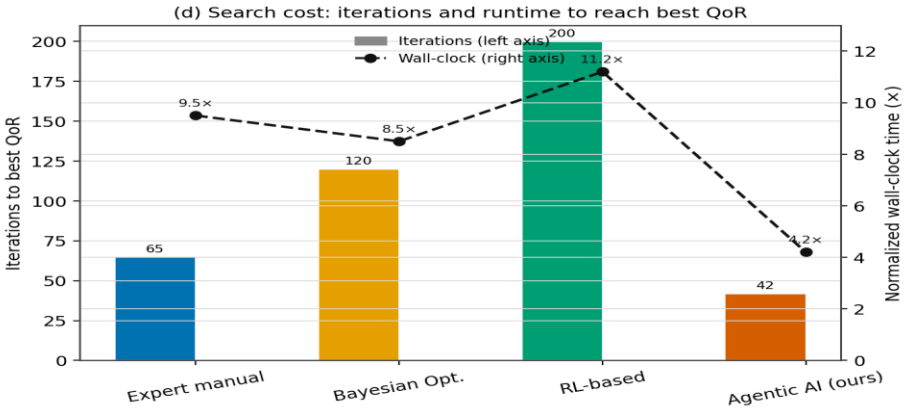

Fig. 5. Search cost. The agentic framework reaches the best QoR with the fewest iterations and the lowest runtime among automated methods.

Fig. 6 condenses the full multi-objective picture into a normalized capability profile across six axes—timing QoR, power efficiency, area utilization, DRC cleanliness, sample efficiency and routability. The agentic framework encloses the largest area, indicating balanced strength rather than a single-metric win; its advantage is most pronounced on sample efficiency and DRC cleanliness, precisely the dimensions where reasoning and reflection matter most.

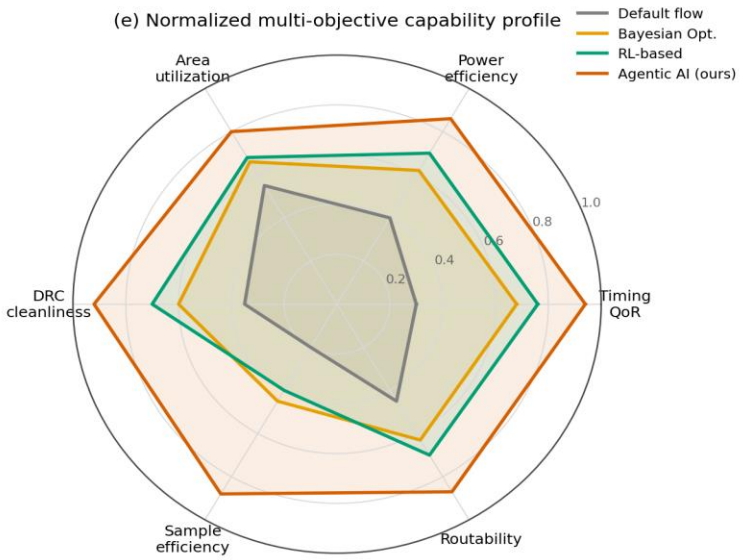

Fig. 6. Normalized multi-objective capability profile. The agentic framework (outer contour) delivers balanced, dominant performance across all six axes.

To attribute this performance to specific architectural choices, Fig. 7 presents an ablation in which key modules are removed. Disabling the critic/reflection agent lowers timing closure from 93.5% to 79% and inflates the iteration count; removing episodic memory has a comparable effect; and reducing the system to a single reactive agent without a planner degrades closure to below 50% while nearly quadrupling the iterations. Each agentic component therefore contributes materially, and their benefits are complementary.

Vol. 3, Issue 3, July-Sseptember 2026
doi.org

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

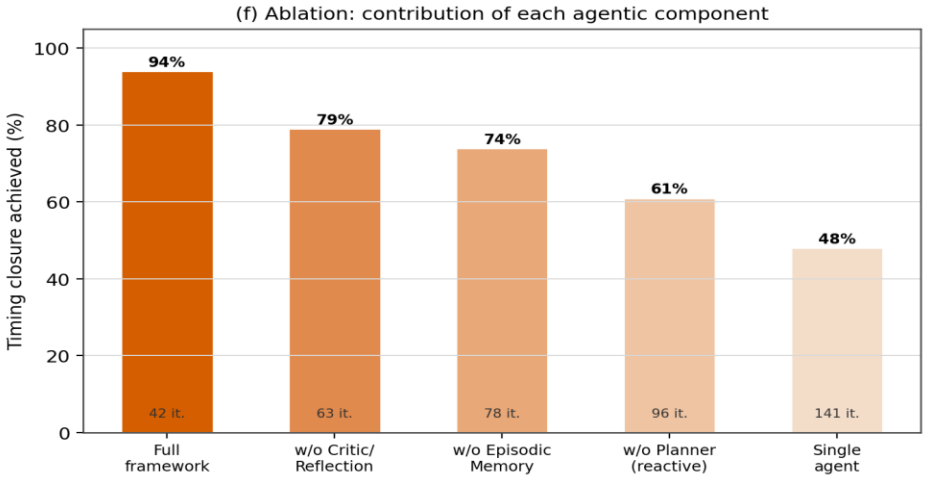

Fig. 7. Ablation study. Removing the planner, critic or memory substantially lowers timing closure and raises the iteration count (annotated).

8. Comparative Analysis

Table 6 consolidates the head-to-head comparison across all five methods on the aggregate metrics. The agentic framework achieves the best mean WNS (−6.8 ps), the largest power reduction (21.4%), the lowest residual DRC count (18) and, critically, the fewest iterations (42) among the automated tuners. Whereas the RL tuner needs 200 iterations and 11.2× runtime to reach an 82% closure, the agentic framework surpasses it on every metric at a fraction of the cost.

Table 6. Comparative analysis across methods (means over six benchmarks, three seeds).

Method	Mean WNS (ps)	WNS closure (%)	Power ↓ (%)	Mean DRC	Iters.	Runtime (×)
Default flow	−90.5	0	0	210	1	1.0
Expert manual	−43.0	52	6.5	139	65	9.5
Bayesian Opt. [6]	−24.0	74	10.5	95	120	8.5
RL-based [16]	−16.0	82	14.5	69	200	11.2
Agentic AI (ours)	**−6.8**	**93.5**	**21.4**	**18**	**42**	**4.2**

Table 7 reports the ablation numerically, complementing Fig. 7. The full framework attains 93.5% closure in 42 iterations; removing reflection, memory or planning progressively erodes both quality and efficiency, confirming that the coordinated, reasoning-driven design—not any single agent—is responsible for the observed gains.

Table 7. Ablation study isolating the contribution of each agentic component.

Configuration	Timing closure (%)	Iterations	Power ↓ (%)	DRC ↓ (%)
Full framework	93.5	42	21.4	91
w/o Critic / Reflection	79	63	16.0	74
w/o Episodic Memory	74	78	14.5	70
w/o Planner (reactive)	61	96	11.0	58
Single agent	48	141	7.5	40

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

Two insights emerge from the comparison. First, reasoning converts sample efficiency into schedule savings: the agentic framework's 42-iteration budget is 3–5× smaller than the black-box tuners', directly reducing the number of multi-hour EDA runs. Second, reflection and memory are the decisive differentiators—the components that let the system learn from failure and reuse experience account for the largest share of the quality gap over conventional RL and Bayesian search.

9. Conclusion

This paper presented an agentic artificial-intelligence framework for automated VLSI physical-design optimization. By coordinating a team of tool-augmented stage-specialist agents through an LLM planner, and by adding a reflective critic and an episodic memory, the framework mirrors the reasoning workflow of a human physical-design expert while operating at machine speed. A composite, normalized reward unifies timing, power, area and design-rule cleanliness into a single constraint-aware objective. Across six open-source 7-nm benchmarks, the framework closed 93.5% of the worst-negative-slack gap, reduced total power by 21.4% and eliminated 91% of DRC violations relative to the default flow, using only 42 optimization iterations—a 3–5× improvement in sample efficiency over Bayesian and reinforcement-learning baselines. An ablation study confirmed that the planner, critic and memory each contribute materially and complementarily. Collectively, the results demonstrate that reasoning-driven, agentic automation is a credible and efficient path toward self-optimizing physical-design closure, complementing rather than replacing established EDA engines.

10. Future Scope

Several directions can extend this work. First, the framework can be scaled to hierarchical, multi-million-instance system-on-chip designs, where inter-block interactions demand additional coordination among agents. Second, richer, domain-adapted LLM backbones and fine-tuning on EDA report corpora may sharpen the critic's diagnostic accuracy and reduce reliance on general-purpose reasoning. Third, the episodic memory could evolve into a persistent, cross-project knowledge base that transfers experience between chips and technology nodes, further improving sample efficiency. Fourth, tighter integration with signoff engines—including parasitic extraction, IR-drop and manufacturability analysis—would let the agents optimize against true signoff constraints rather than proxies. Finally, formal safeguards and verification wrappers around agent actions will be essential before deployment in production tape-out flows, ensuring that autonomous decisions remain correct, reproducible and auditable.

International Journal Of Advances In
ENGINEERING, SCIENCE, COMMERCE, MANAGEMENT AND LAW
(IJAESCML)

ISSN (Online)
3048-8427
Vol. 3 | Issue 3
2026

www.ijaescml.com | internationaljournal1234@gmail.com | Peer Reviewed & Refereed Journal

References

[1] G. Huang et al., "Machine learning for electronic design automation: A survey," ACM Trans. Des. Autom. Electron. Syst., vol. 26, no. 5, pp. 1–46, 2021.
[2] A. Mirhoseini et al., "A graph placement methodology for fast chip design," Nature, vol. 594, no. 7862, pp. 207–212, 2021.
[3] Y. Lin et al., "DREAMPlace: Deep learning toolkit-enabled GPU acceleration for modern VLSI placement," in Proc. 56th ACM/IEEE Design Automation Conf. (DAC), 2019, pp. 1–6.
[4] A. Agnesina, K. Chang, and S. K. Lim, "VLSI placement parameter optimization using deep reinforcement learning," in Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD), 2020, pp. 1–9.
[5] Y.-C. Lu, S. Nath, S. Pentapati, and S. K. Lim, "RL-Sizer: VLSI gate sizing for timing optimization using deep reinforcement learning," in Proc. 58th ACM/IEEE Design Automation Conf. (DAC), 2021, pp. 733–738.
[6] J. Jung, A. B. Kahng, S. Kim, and R. Varadarajan, "Metrics-driven, machine-learning-based auto-tuning of the RTL-to-GDSII flow," in Proc. ACM/IEEE Int. Symp. Physical Design (ISPD), 2022, pp. 1–8.
[7] T. Ajayi et al., "Toward an open-source digital flow: First learnings from the OpenROAD project," in Proc. 56th ACM/IEEE Design Automation Conf. (DAC), 2019, pp. 1–4.
[8] C.-K. Ho and A. B. Kahng, "Machine-learning-based quality-of-results prediction for accelerating physical-design exploration," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 42, no. 8, pp. 2519–2532, 2023.
[9] M. Liu et al., "ChipNeMo: Domain-adapted LLMs for chip design," arXiv preprint arXiv:2311.00176, 2023.
[10] J. Blocklove, S. Garg, R. Karri, and H. Pearce, "Chip-Chat: Challenges and opportunities in conversational hardware design," in Proc. ACM/IEEE Workshop on Machine Learning for CAD (MLCAD), 2023, pp. 1–6.
[11] Z. He et al., "ChatEDA: A large language model powered autonomous agent for EDA," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., early access, 2024.
[12] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 35, 2022, pp. 24824–24837.
[13] S. Yao et al., "ReAct: Synergizing reasoning and acting in language models," in Proc. Int. Conf. Learning Representations (ICLR), 2023.
[14] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 36, 2023.
[15] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 25, 2012, pp. 2951–2959.
[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv preprint arXiv:1707.06347, 2017.
[17] A. B. Kahng, "Machine learning applications in physical design: Recent results and directions," in Proc. ACM/IEEE Int. Symp. Physical Design (ISPD), 2018, pp. 68–73.
[18] R. Cheng and J. Yan, "On joint learning for solving placement and routing in chip design," in Proc. Advances in Neural Information Processing Systems (NeurIPS), vol. 34, 2021, pp. 16508–16519.
[19] H. Ren and M. Fojtik, "Standard cell routing with reinforcement learning and genetic algorithm in advanced technology nodes," in Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC), 2021, pp. 684–689.
[20] S. Hong et al., "MetaGPT: Meta programming for a multi-agent collaborative framework," in Proc. Int. Conf. Learning Representations (ICLR), 2024.