RISCFORGE-BENCH: BENCHMARKING THE GAP BETWEEN LOCAL RTL AND PROCESSOR-EXTENSION DELIVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Module-level RTL benchmarks ask whether an AI system can generate functionally correct hardware. A deployable processor extension must instead preserve one software-visible operation across software, an ISA contract, executable semantics, temporal interfaces, a processor core, workloads, and implementation constraints. We introduce RISCFORGE-BENCH, a family-based executable benchmark spanning 12 independently admitted RISC-V extension families across four workload domains. Each family is one statistical breadth unit; together, they define 48 registered variation axes and 8,403 enumerated sealed-test entries. Across matched model–agent configurations, systems pass 95/108 isolated compute-RTL cells. When systems also author processor-facing RTL, 6/108 cells pass integrated RTL functionality, but none passes protocol stress. End-to-end delivery under full cross-artifact authorship remains 0/108. Same-artifact replay finds 177 locally correct RTL modules and no corresponding delivery. Supplying the admitted implementation restores 32/32 provider-completed deliveries, showing that the evaluator can accept correct artifacts. Responsibility controls place the earliest tested break at processor-facing RTL without claiming to isolate protocol reasoning. Targeted repair recovers nonzero delivery from a near-complete implementation. These results expose a transfer gap between local artifact competence and cross-artifact executable consistency, and establish a versioned target for evaluating long-horizon hardware agents under bounded engineering budgets and auditable scoring.

1 INTRODUCTION

Language-model agents can generate RTL and assertions, repair hardware repositories, and operate simulation or synthesis tools (Liu et al., 2023; Kang et al., 2025; Cui et al., 2026; Deng et al., 2026). Processor customization poses a different agent problem. A software-visible instruction is represented simultaneously by an encoding, an intrinsic, executable semantics, compute logic, temporal handshakes, processor state, and workload behavior. A locally correct arithmetic module is useful, but it is not a delivered extension if these representations disagree.

This setting exposes a general failure mode in long-horizon agents: a semantic invariant must survive across multiple files, languages, tools, and temporal interfaces. Module-generation benchmarks observe one representation; repository-repair benchmarks validate a patch against a regression suite; downstream physical-design evaluations emphasize closure from supplied or generated RTL. These settings do not isolate whether an agent can preserve a newly introduced software-visible instruction across ISA, software, processor protocol, core execution, and synthesis. The resulting gap matters to both hardware evaluation and agent science because local artifact competence can coexist with failure to coordinate artifacts over a long tool-use trajectory.

The additional gates are not independent obstacles appended to make the benchmark harder. Each gate is an executable consumer of the same software-visible operation. A delivery failure therefore exposes inconsistency in one shared contract, not failure on an unrelated downstream task.

This paper asks one focused question: when an agent can write correct local RTL, can it preserve that operation through processor-facing integration and deliver an executable extension?

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

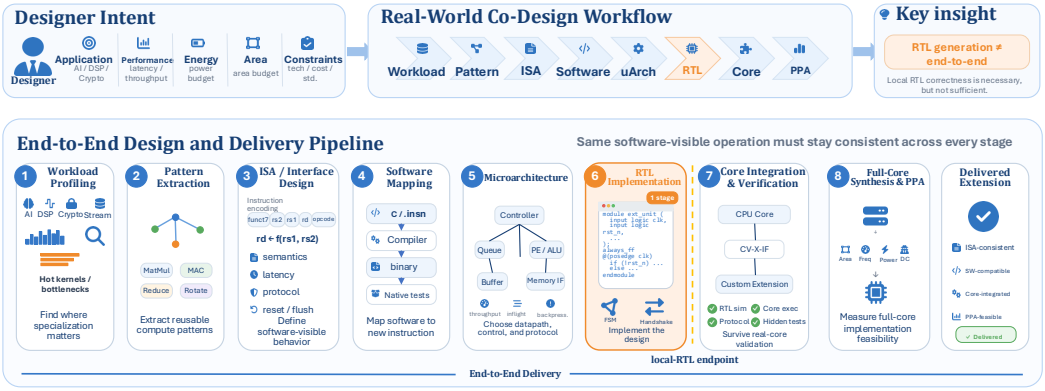

Figure 1: **From design intent to processor-extension delivery.** Workload profiling and pattern extraction describe benchmark-family construction; they are not scored agent actions. A scored arm begins from its frozen family specification and assigns a stated subset of ISA, software, semantics, compute RTL, processor-facing RTL, and finalization to the agent. Independent evaluation continues through core execution, hidden workloads, and full-core synthesis. Local RTL benchmarks stop at one artifact; RISCFORGE-BENCH evaluates whether the same operation survives the coupled delivery chain.

RISCFORGE-BENCH makes this boundary executable. It comprises 12 task families on CV32E40X and CV-X-IF, each with a provenance-qualified native workload, a legal numeric and protocol-schedule space, sealed instances, independent semantics and RTL oracles, protocol and core tests, mutation-tested judges, and paired full-core synthesis evidence. Benchmark scale has two non-interchangeable axes. Independent breadth is the 12 families; within-family depth comprises 48 registered variation axes, 8,403 enumerated sealed-test entries, and 12 ordered strict delivery gates.

The evidence forms a three-part test. First, frontier systems solve 95/108 matched local compute cells. Second, under processor-facing authorship, 6/108 separate matched cells pass integrated RTL functionality, but none passes protocol stress or completes A0, and B0 delivery is 0/108; replay likewise finds 177 local passes but no delivery. Third, supplying the admitted implementation restores 32/32 provider-completed deliveries. The implementation-provided anchor rules out an evaluator that is intrinsically unable to accept a successful artifact. Together with the matched responsibility contrast, the earliest observed break appears when processor-facing RTL becomes agent-authored; the current experiments do not fully isolate temporal protocol reasoning from wrapper construction, build mechanics, or bounded execution.

Our contributions are:

- **A family-based executable benchmark.** Twelve frozen task families cover four workload domains and expose 48 registered numeric, workload, and temporal-protocol axes instantiated by 8,403 enumerated sealed-test entries. Evaluation spans ISA, software, semantics, RTL, core execution, hidden evaluation, and full-core synthesis without reclassifying within-family entries as tasks.
- **Validity and localization controls.** Oracle admission, 272 certified mutations, same-artifact replay, matched responsibility scopes, and an implementation-provided anchor distinguish local generation, processor-facing authorship, and evaluator operation.
- **A cross-artifact transfer finding.** Local artifact competence does not transfer to cross-artifact executable consistency in the tested model–agent configurations. Added runtime or static workflow support changes intermediate progress without restoring delivery.

Table 1: **Benchmark scope comparison.** Each benchmark is listed separately to avoid transferring one work's scope to another. ISA/SW denotes agent-authored software-visible instruction or mapping; Proc. denotes processor protocol or core integration; Synth. denotes executable synthesis evidence, not necessarily agent-run physical design. ✓ = in scope; ○ = subset or adjacent evidence; — = outside the stated evaluation unit.

Benchmark	ISA/SW	Sem.	Proc.	Native	Hidden exec.	Synth. eval.
VerilogEval (Liu et al., 2023)	—	—	—	—	○	—
OpenLLM-RTL (Liu et al., 2024)	—	—	—	—	○	—
RealBench (Jin et al., 2025)	—	—	—	—	○	—
ChipBench (Yu et al., 2026)	—	—	—	—	○	—
CVDP (Pinckney et al., 2025)	—	—	—	—	○	—
FVEval (Kang et al., 2025)	—	—	—	—	○	—
AssertionBench (Pulavarthi et al., 2025)	—	—	—	—	○	—
HWE-Bench (Cui et al., 2026)	○	—	○	○	○	—
HSCO-Bench (Tsai et al., 2026)	✓	—	✓	✓	○	✓
CLOSER-Bench (Zhou et al., 2026)	—	—	—	—	✓	✓
FluxBench (Deng et al., 2026)	—	—	○	—	○	✓
RISCForge-Bench	✓	✓	✓	✓	✓	✓

2 Related Work

Hardware-language-model benchmarks differ primarily in their evaluation unit. VerilogEval and OpenLLM-RTL measure specification-to-RTL generation; RealBench, ChipBench, and CVDP broaden module or design scale; FVEval and AssertionBench evaluate verification artifacts (Liu et al., 2023; 2024; Jin et al., 2025; Yu et al., 2026; Pinckney et al., 2025; Kang et al., 2025; Pulavarthi et al., 2025). HWE-Bench evaluates native repository repair, and HSCO-Bench studies application-driven hardware–software co-design (Cui et al., 2026; Tsai et al., 2026). CLOSER-Bench, FluxBench, and FluxEDA move downstream toward cross-stage implementation, RTL-to-GDS tool use, and persistent EDA execution (Zhou et al., 2026; Deng et al., 2026; Chen et al., 2026). FluxBench asks whether an agent can carry supplied RTL through physical implementation. RISCForge-Bench asks the complementary upstream question: can an agent first create a coherent software-visible processor extension and carry it into implementation evaluation? Its evaluation unit therefore spans ISA, software, semantics, RTL, processor protocol, core execution, workload behavior, and synthesis evidence.

The benchmark also follows outcome-based software-agent evaluation. SWE-bench replaces snippet generation with executable repository issues; LiveCodeBench emphasizes temporally separated evaluation; AndroidWorld and Terminal-Bench define parameterized or interactive environments with programmatic success conditions (Jimenez et al., 2024; Jain et al., 2025; Rawles et al., 2025; Merrill et al., 2026). RISCForge-Bench adopts the same outcome principle while treating processor-extension delivery, rather than reference-code similarity, as the terminal state.

3 RISCForge-Bench

3.1 Task Families, Instances, and Construction

A *task family* fixes a workload intent, typed artifact contract, processor platform, and legal numeric and protocol-schedule space. A public native example and sealed numeric, workload, reset, flush, back-pressure, and scheduling cases are instances of that family. Thus the registry contains 12 independent families, not 12 isolated test cases. The task family is the independent statistical unit. A trajectory evaluates one model–agent configuration on a frozen family package.

The frozen suite contains 48 registered variation axes and 8,403 enumerated sealed-test entries. The latter provide executable within-family stress rather than additional independent samples. Their median count is 24 per family (range 8–4,097); Appendix E reports the counting rule and full scale audit.

Construction began from five pinned upstream projects in AI, DSP, cryptography, and stream/memory. Before confirmatory model execution, a family entered the registry only if it had auditable repository, commit, license, source-function and source-hash provenance; a deterministic native adapter and observable output; a software-visible instruction mapping; an independently

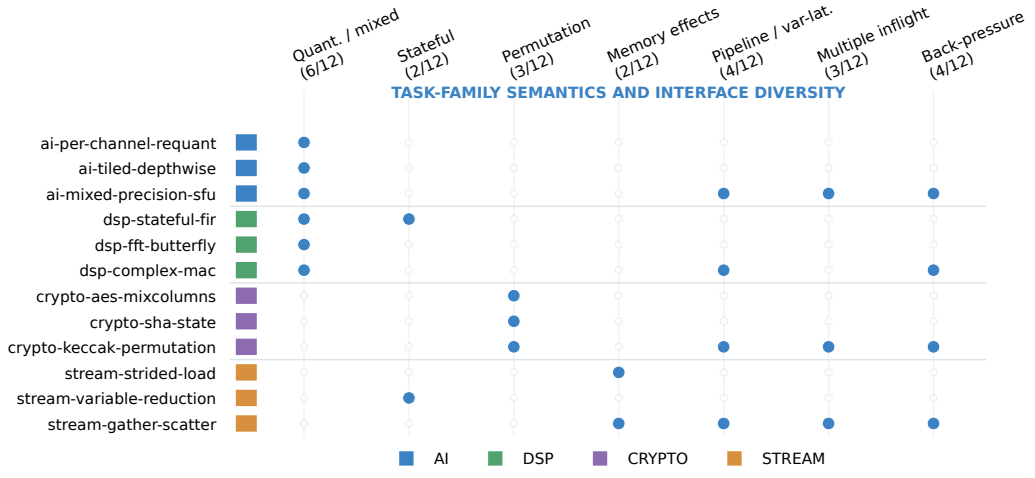

Figure 2: **Twelve task families cover diverse semantic and interface properties.** Each row is an independent family and expands into four registered variation axes. Marks are conservative annotations of frozen public specifications and admitted interface contracts. Common core, hidden-workload, and synthesis checks add evaluation depth but are intentionally not counted as family diversity.

implemented oracle; executable CV-X-IF and CV32E40X tests; and a feasible paired full-core reference. This is a coverage-designed suite, not a random sample of all processor extensions; the machine-readable registry states the inclusion, exclusion, and refresh rules. Family selection did not use confirmatory model outcomes.

The three families per domain span quantized or mixed-precision arithmetic, stateful recurrence, bitwise permutation, memory effects, pipelined or variable latency, multiple inflight operations, and back-pressure (Figure 2). Common core, hidden-test, and synthesis requirements are omitted from the matrix because they describe evaluator depth rather than family diversity.

3.2 Artifact Boundary and Delivery Gates

A scored family package begins with a workload-grounded extension requirement. Depending on the responsibility scope, the candidate may author a typed ISA contract, executable semantics, software mapping and workload patch, compute RTL, processor-facing RTL, and candidate tests. The evaluator owns the core, build environment, sealed instances, independent oracles, and synthesis flow.

Let the ordered strict-gate sequence be

$$(g_1, \ldots, g_{12}) = (G_1, \ldots, G_{10}, G_{12}, G_{13}).$$

Then strict delivery is

$$\textsc{Deliver@Budget}(r) = \mathbb{1}\left[\bigwedge_{k=1}^{12} g_k(r)\right].$$

G11 measures workload benefit separately and is not part of strict delivery. The strict sequence checks submission validity, ISA contract and encoding, software mapping, executable semantics, integrated RTL functionality, protocol stress, core execution, native and sealed workloads, full-core synthesis feasibility, and logical-resource compliance. This notation preserves the historical gate IDs without incorrectly placing G11 inside the delivery conjunction.

G6 compiles and functionally simulates the complete compute-plus-wrapper RTL candidate. G7 then stresses CV-X-IF temporal behavior, including issue, commit, writeback, stalls, multiple in-flight requests, reset, and flush. "Local RTL pass" is a separate A-1 or replay checkpoint applied

Table 2: **Matched responsibility scopes.** Family intent, sealed numeric distribution, core, and final evaluator are held fixed; supplied artifacts are read-only.

Scope	Agent-authored artifacts	Supplied artifacts
A-1 Local Compute RTL	Compute RTL and candidate tests	ISA, semantics, software/workload, CV-X-IF wrapper, core
A0 Processor-Facing RTL	Compute RTL, CV-X-IF RTL, tests	ISA, semantics, software/workload, core
B0 Cross-Artifact Delivery	ISA, semantics, software/workload, compute and CV-X-IF RTL, tests	Core and evaluator

only to the compute module; it is not G6. The distinction explains why a protocol-authorship condition can first fail at G6 before reaching the dedicated G7 stress suite.

3.3 Oracle Admission, Mutation, and Implementation Evidence

Before model evaluation, an independent oracle for every family must pass the native baseline, semantics, software build, integrated RTL, protocol, core image, sealed workload, and paired full-core synthesis envelope. All 12/12 families pass. Mutation analysis injects at least 22 certified non-equivalent faults per family across contracts, software, semantics, RTL, protocol, reset/flush, and integration; the judges kill 272/272. These controls establish an executable solution and sensitivity to the injected faults, not exhaustive correctness.

Baseline and candidate synthesis use the same CV32E40X commit, CV-X-IF top, library, clock, operating condition, constraints, Tcl flow, and parser. The evaluator records area, timing, and vectorless power. Feasibility and workload benefit remain separate: a candidate can fit the implementation envelope without accelerating the workload. Synthesis is evaluator-run outcome evidence, not an agent-authored physical-design task.

3.4 Benchmark Release and Reproducibility

We release the complete RISCForge-Bench benchmark; its code is Apache-2.0 licensed. The release contains all 12 family specifications, native workload adapters and provenance, contract schemas, workspace/submission CLI, executable evaluators, experiment configurations, result ledgers, and analysis code. Third parties can run agents, create immutable bundles, and locally evaluate all delivery gates (RTL compilation and simulation, protocol stress, processor integration, workload validation, and full-core synthesis) without curator scoring.

The version-pinned evaluator checks the same immutable bundle and records gate outcomes, environment versions, and evidence hashes. Scoring inputs and reference implementations are isolated from the agent workspace, limiting agent access rather than local reproducibility. The versioned registry supports new instances.

Reproducing full-core synthesis requires the corresponding licensed EDA tools and target technology libraries. The benchmark ends at ASIC-oriented full-core synthesis and excludes placement and routing, signoff, tape-out, silicon, compiler backends, operating systems, and coherence.

4 Evaluation Protocol

4.1 Model–Agent Configurations and Responsibility Scopes

Results measure complete model–agent configurations. A-1, A0, and B0 provide the cleanest matched responsibility contrast (Table 2). A-1 supplies the processor wrapper and evaluates local compute RTL. A0 assigns compute and processor-facing RTL while supplying ISA, semantics, and software. B0 assigns the coupled ISA, semantics, software, workload patch, compute RTL, and processor-facing RTL chain.

A0 reports two distinct checkpoints. G6 means cumulative closure through integrated compute-plus-wrapper functional simulation. Registered A0 completion means cumulative closure through

G8, after G7 protocol stress and G8 core integration. A-1, A0, and B0 are matched but separately generated cells, not one artifact funnel.

The evidence program has three layers. First, a frozen discovery study crosses 12 families, three model identifiers, three seeds, and four registered conditions. Its 432 trajectories establish the strict delivery floor and provide immutable artifacts for same-artifact replay. Second, a matched 324-cell confirmation crosses GPT-6 Astra, GPT-5.6 Sol, and DeepSeek V4 Pro with A-1, A0, and B0 on every family and seed. This contrast tests whether local RTL competence survives processor-facing responsibility. Both studies use 36 model calls, 36 tool calls, and a 720-second safety limit. They compare complete model–agent systems under fixed scopes, not provider-agnostic foundation models.

Third, explanatory controls test competing interpretations of the floor. The implementation-provided L0S anchor asks whether the evaluator can accept a correct bundle; PR/WS/FS varies processor-facing support from repair to from-scratch authorship. Separately locked L1–L4 responsibility, execution-support, runtime, and PPA diagnostics ask where progress stops, whether additional support changes it, and whether surviving artifacts remain implementable. Results follow this order: endpoint validity, local-to-delivery transfer, responsibility gradient, alternative explanations, and implementation headroom.

4.2 Metrics and Statistical Units

The primary endpoint is FullDeliveryRate, the mean of DELIVER@BUDGET over a stated denominator. Normalized Closure Depth (NCD) is a secondary diagnostic:

$$\mathrm{NCD}(r) = \frac{1}{12} \sum_{k=1}^{12} \mathbb{1}\left[\bigwedge_{j=1}^{k} g_j(r) \right].$$

NCD is the fraction of consecutive strict gates passed from the start; it is not a success rate and does not replace DELIVER@BUDGET. Independent judges may run after cumulative closure breaks, but those observations do not restore delivery. We report scheduled and provider-completed denominators separately. Statistical inference clusters by the 12 task families; 95% intervals use 10000 family-clustered bootstrap replicates, and paired contrasts retain common family–model–seed keys.

The same-artifact replay was pre-specified and locked before execution; it recompiles and tests the final `extension_compute.sv` from every provider-completed primary trajectory against an independent oracle on sealed inputs. Missing source is failure. This replay approximates the stopping point of an isolated compute-RTL benchmark and reports the observed conditional delivery rate among local passes; it is not a population probability estimate.

5 Results

5.1 The Strict Endpoint Is Solvable but Remains Unreached by Authored Systems

We first ask whether zero delivery reflects an impossible benchmark or a capability limit. The provider completed 386 of 432 scheduled primary trajectories; 46 were infrastructure-unavailable. Delivery was 0/386 among completed trajectories and 0/432 on the scheduled matrix. In contrast, all 12/12 independently authored family oracles passed the same endpoint. The zero is therefore a result about tested model–agent systems under bounded responsibility, not an unsatisfiable benchmark.

The all-zero endpoint conceals intermediate capability: primary systems produced 128 valid submissions, 70 valid ISA contracts, 53 working software mappings, and 53 executable-semantic passes. Only three preserved cumulative closure through integrated RTL compilation, and none passed integrated RTL functionality. This motivates the stage-resolved controls rather than promoting NCD to a replacement endpoint.

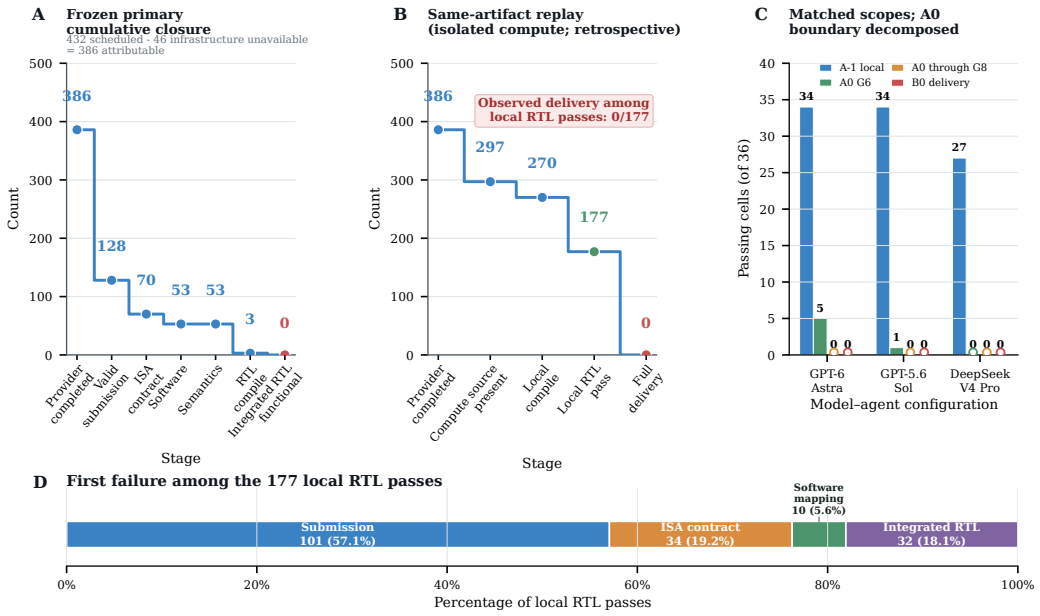

Figure 3: **Local RTL success does not imply processor-extension delivery.** **A**, frozen-primary cumulative survival. **B**, retrospective same-artifact isolated-compute replay. **C**, matched scope outcomes for three frontier configurations; A0 reports G6 functionality separately from cumulative completion through G8. These are separately generated cells, not one survival curve. **D**, evaluator-stage composition of the first failure among 177 local RTL passes. Panel D localizes rejection boundaries, not latent reasoning errors.

5.2 Local RTL Success Is Insufficient for Delivery

We next test the central claim on identical artifacts. The same-artifact replay finds compute source in 297/386 completed trajectories, local compilation in 270/386, and exact sealed numeric correctness in 177/386. The observed conditional delivery rate among the 177 local passes is 0/177. Of these, 101 first fail submission. G1 is not one latent cause: 66/101 are malformed delivery artifacts (54 schema, 11 path/allowlist, one manifest), whereas 35/101 end before valid finalization. Both fail delivery, but they distinguish artifact correctness from unfinished workflow. The remaining 76 locally correct artifacts form valid submissions but still deliver 0/76; their first failures are ISA contract (34), software mapping (10), or integrated RTL functionality (32). Thus the gap persists after conditioning on valid submission.

An independently scheduled matched confirmation yields 95/108 A-1 local passes. Within the separate A0 cells, the cumulative count is 75 at G1, 70 at G4, 58 at G5, 6 at G6, and 0 at G7 and G8. GPT-6 Astra contributes five G6 passes and GPT-5.6 Sol one; all six first fail protocol stress, and B0 delivery is 0/108. Every family has at least one A-1 pass (2/9–9/9), but the six G6 passes occur in two stream/memory families. The matched contrast therefore places the earliest aggregate break within processor-facing authorship while showing that functional integration is not uniformly zero. The sanitized ledger identifies G7 as the rejection boundary but not the hidden trigger or waveform; we do not infer a specific stall, commit, reset, or flush mechanism.

5.3 A Graded Processor-Facing Responsibility Diagnostic

We then ask whether the floor comes from an evaluator that cannot accept success. Supplying the admitted implementation changes strict delivery to 32/32 among provider-completed L0S cells (32/36 scheduled). Eight available model–agent configurations validate and submit four domain-balanced families. The evaluator, submission format, and terminal workflow can therefore accept correct artifacts.

Table 3: **Graded GPT-5.6 Sol processor-facing responsibility diagnostic.** One frozen family per domain, one seed, and the same strict judge. PR exposes faults in issue-ready gating, result-ID propagation, and matching commit-kill cancellation. This post-primary diagnostic is descriptive, not a population estimate. G6/G7 counts cells passing integrated RTL functionality and protocol stress.

Condition	Processor-facing RTL supplied	Deliver	G6/G7	Mean NCD
L0S anchor	Complete admitted implementation, read-only	4/4	4/4	100.0%
PR repair	Wrapper with three disclosed fault classes	1/4	1/4	56.3%
WS skeleton	Typed shell, safe defaults, explicit TODOs	0/4	0/4	31.3%
FS from scratch	None	0/4	0/4	31.3%

A prospectively locked GPT-5.6 Sol/Codex diagnostic tests whether this responsibility boundary is graded. It holds four domain-balanced families, trusted compute RTL, seed, 12-minute budget, hidden judge, and delivery criterion fixed. The conditions progress from a complete supplied implementation (L0S), through near-complete repair (PR) and a typed wrapper skeleton (WS), to processor-facing RTL authored from scratch (FS; Table 3).

The observed ordering is Complete $>$ Repair $>$ Skeleton $\approx$ Scratch: L0S delivers 4/4 cells, PR 1/4, and WS and FS 0/4. A near-correct processor-facing implementation can therefore cross the endpoint, whereas interface structure alone is insufficient under the tested budget. The PR result is descriptive evidence of a graded responsibility frontier, not a population repair rate or an isolated protocol-reasoning effect. The older one-seed responsibility ablation is consistent with this ordering: L1 supplies trusted compute RTL but assigns the wrapper from scratch and delivers 0/24; L2–L4 also deliver 0/24.

5.4 More Runtime and Workflow Support Change Progress, Not Delivery

We finally test two simpler explanations: insufficient execution support and insufficient time. Static workflow guidance and public stage tracking alter valid-submission counts and NCD, but all tested support conditions remain at zero delivery. The nonconcurrent stateful condition has a paired NCD difference of +3.1 percentage points relative to workflow, with a 95% interval [-2.4, 9.0] that includes zero. These studies compare bundled execution systems, not isolated prompting effects.

The 12-to-24-minute probe increases partial progress. In the four-family GPT-5.6 Sol probe, 12-to-48 minutes leaves delivery at 0/4 in both arms. Telemetry lacks per-event timestamps and provider latency, so it cannot partition inference, tools, waiting, or idle time. Both arms record 13 model calls; completed tool calls rise only from 54 to 61, with no trace-visible `rf-v2 submit`. This is a larger wall-clock guard, not four times the effective work.

5.5 Implementation Evidence Remains Informative

PPA answers a different question: whether independently measurable artifacts fit the implementation envelope. The evaluator obtains 11 independent full-core synthesis reports, of which 3 satisfy all registered area, timing, and vectorless-power limits. These measurements occur after the cumulative path has already broken, so they certify implementation properties without creating delivery. Appendix G reports all 11 observations and marks the three with hidden-workload cycle counts. Retaining this gate preserves headroom: as functional systems improve, the benchmark can distinguish executable extensions that fit the target implementation envelope from those that do not.

6 Discussion

Evaluate the shared contract, not an isolated file. Frontier configurations solve local compute in every family, yet those modules do not remain consistent with software, processor-facing RTL, core behavior, and delivery. Processor extensions therefore test whether a long-horizon agent can preserve and revalidate one executable invariant across heterogeneous artifacts.

Make authored responsibility explicit. The implementation-provided anchor makes zero authored delivery interpretable but not discriminative. Only near-complete repair restores delivery. Strict delivery remains primary, with NCD used only to localize failure and scopes separate.

Repair the earliest broken contract, not merely the prompt. Workflow guidance and more time change progress, not delivery. Gate-aware execution should repair the earliest invalid artifact and revalidate its consumers; outcomes characterize model–agent configurations, not foundation models alone.

RISCFORGE-BENCH asks whether an agent can author a coherent extension; FluxBench instead starts from supplied RTL and tests physical-flow execution (Deng et al., 2026).

7 LIMITATIONS

The evidence covers 12 coverage-designed families from five projects on one CV32E40X/CV-X-IF platform, not a random sample. Sealed entries and repeated runs support within-family comparison, not external generality.

Public upstream code may occur in pretraining; sealed instances reduce copying but cannot exclude contamination. Evaluation inputs and reference implementations are provided to reproducing users but isolated from the evaluated agent during scored runs. All gates run locally; full-core synthesis requires licensed EDA tools and target libraries.

RISCFORGE-BENCH ends at evaluator-run full-core synthesis and excludes placement and routing, signoff, tape-out, silicon, compiler backends, operating systems, and coherence. Vectorless power is estimated; first failure is an evaluator-stage, not reasoning-error, taxonomy; supplied implementation is not credited to the agent.

8 CONCLUSION

Across 12 admitted families, matched systems pass 95/108 isolated compute cells; 6/108 separate A0 cells pass integrated RTL functionality, but none passes protocol stress or completes A0, and B0 delivery is 0/108. Replay finds 177 local passes without delivery; supplying the implementation restores 32/32 deliveries, and near-complete repair recovers nonzero delivery. The target is cross-artifact preservation of one software-visible contract, not merely Verilog generation. Under tested systems and budgets, the boundary is processor-facing authorship, not protocol reasoning in isolation.

REPRODUCIBILITY STATEMENT

We release the complete benchmark, including the 12-family registry, provenance records, native adapters, executable evaluators, experiment configurations, result ledgers, and analysis and figure-generation code. The release supports local candidate evaluation and reproduction of the evaluation protocol and reported analyses. Benchmark code is licensed under Apache-2.0; full-core synthesis requires the corresponding licensed EDA tools and target technology libraries.

ETHICS STATEMENT

This study uses licensed public hardware artifacts and involves neither human subjects nor personal data; failures, provenance, and the limits of each tested model–agent configuration are disclosed.

AI USE STATEMENT

Generative AI assisted literature retrieval, planning, implementation and analysis scripts, figures, translation, and drafting. Human authors fixed protocols and claims, verified outputs and citations, made all interpretations, and take responsibility for the work.

REFERENCES

Alibaba Cloud Model Studio. Qwen3.5-397b-a17b model information and pricing. https://help.aliyun.com/zh/model-studio/qwen3-5-397b-a17b, 2026. Accessed 2026-09-23.

Zhengrui Chen, Zixuan Song, Yu Li, Qi Sun, and Cheng Zhuo. Fluxeda: A unified execution infrastructure for stateful agentic eda. *arXiv preprint arXiv:2603.25243*, 2026. URL https://arxiv.org/abs/2603.25243.

Fan Cui, Hongyuan Hou, Zizhang Luo, Chenyun Yin, and Yun Liang. Hwe-bench: Benchmarking llm agents on real-world hardware bug repair tasks. *arXiv preprint arXiv:2604.14709*, 2026. URL https://arxiv.org/abs/2604.14709.

DeepSeek. Models and pricing. https://api-docs.deepseek.com/quick_start/pricing/, 2026. Accessed 2026-09-19.

Jinyuan Deng, Zhengrui Chen, Xufeng Wei, Tianyu Xing, Chenyi Wen, Qi Sun, and Cheng Zhuo. Can ai agents really complete rtl-to-gds? lessons from benchmarking tool-interactive eda workflows. *arXiv preprint arXiv:2607.17528*, 2026. URL https://arxiv.org/abs/2607.17528.

European Central Bank. Euro exchange rates—22 september 2026. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:C/2026/04613, 2026. Official Journal of the European Union; accessed 2026-09-24.

Ethan Hsu, Hong Meng Yam, Ines Bouissou, Aaron John, Raj Thota, Josh Koe, Vivek Putta, G Dharesan, Alexander Spangher, Shikhar Murty, Tenghao Huang, and Christopher Manning. WebDS: An end-to-end benchmark for web-based data science. In *International Conference on Learning Representations*, pp. 31777–31795, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/35686e5837762715a9b105c96cba43af-Abstract-Conference.html.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/94074dd5a072d28ff75a76dabed43767-Abstract-Conference.html.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/edac78c3e300629acfe6cbe9ca88fb84-Abstract-Conference.html.

Pengwei Jin, Di Huang, Chongxiao Li, Shuyao Cheng, Yang Zhao, Xinyao Zheng, Jiaguo Zhu, Shuyi Xing, Bohan Dou, Rui Zhang, Zidong Du, Qi Guo, and Xing Hu. Realbench: Benchmarking verilog generation models with real-world ip designs. *arXiv preprint arXiv:2507.16200*, 2025. URL https://arxiv.org/abs/2507.16200.

Minwoo Kang, Mingjie Liu, Ghaith Bany Hamad, Syed M. Suhaib, and Haoxing Ren. Fveval: Understanding language model capabilities in formal verification of digital hardware. In *2025 Design, Automation & Test in Europe Conference (DATE)*, pp. 1–6, 2025. doi: 10.23919/DATE64628.2025.10992720. URL https://ieeexplore.ieee.org/document/10992720.

Patrick Tser Jern Kon, Qiuyi Ding, Jiachen Liu, Xinyi Zhu, Jingjia Peng, Jiarong Xing, Yibo Huang, Yiming Qiu, Jayanth Srinivasa, Myungjin Lee, Mosharaf Chowdhury, Matei Zaharia, and Ang Chen. EXP-Bench: Can AI conduct AI research experiments? In *International Conference on Learning Representations*, pp. 120555–120604, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/c411f5b2d9c55f1685e72db224ad8b0e-Abstract-Conference.html.

Mingjie Liu, Nathaniel Pinckney, Brucek Khailany, and Haoxing Ren. Invited paper: Verilogeval: Evaluating large language models for verilog code generation. In *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pp. 1–8, 2023. doi: 10.1109/ICCAD57390.2023.10323812. URL https://ieeexplore.ieee.org/document/10323812.

Shang Liu, Yao Lu, Wenji Fang, Mengming Li, and Zhiyao Xie. Openllm-rtl: Open dataset and benchmark for llm-aided design rtl generation. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–9, 2024. doi: 10.1145/3676536.3697118. URL https://dl.acm.org/doi/10.1145/3676536.3697118.

Mike Merrill, Alexander Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/444a3737adaee10d86ad2ef5f74468e6-Abstract-Conference.html.

MiniMax. Pay-as-you-go pricing. https://platform.minimax.cn/docs/guides/pricing-paygo, 2026. Accessed 2026-09-23.

OpenAI. Api pricing. https://developers.openai.com/api/docs/pricing, 2026. Accessed 2026-09-19.

Nathaniel Pinckney, Chenhui Deng, Chia-Tung Ho, Yun-Da Tsai, Mingjie Liu, Wenfei Zhou, Brucek Khailany, and Haoxing Ren. Comprehensive verilog design problems: A next-generation benchmark dataset for evaluating large language models and agents on rtl design and verification. *arXiv preprint arXiv:2506.14074*, 2025. URL https://arxiv.org/abs/2506.14074.

Vaishnavi Pulavarthi, Deeksha Nandal, Soham Dan, and Debjit Pal. Assertionbench: A benchmark to evaluate large-language models for assertion generation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 8073–8080. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-naacl.449. URL https://aclanthology.org/2025.findings-naacl.449/.

Chris Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/01a83bc2f2732a58e6aa731e659e7101-Abstract-Conference.html.

Pei-Huan Tsai, Kuan-Lin Chiu, William Baisi, Pin-Yu Chen, and Luca P. Carloni. Hsco-bench: An agent-driven end-to-end hardware-software co-design benchmark for systems-on-chip. *arXiv preprint arXiv:2605.19399*, 2026. URL https://arxiv.org/abs/2605.19399.

Zijian Wu, Xiangyan Liu, Xinyuan Zhang, Lingjun Chen, Fanqing Meng, Lingxiao Du, Yiran Zhao, Fanshi Zhang, Yaoqi Ye, Jiawei Wang, Zirui Wang, Jinjie Ni, Yufan Yang, Arvin Xu, and Michael Qizhe Shieh. MCPMark: A benchmark for stress-testing realistic and comprehensive MCP use. In *International Conference on Learning Representations*, pp. 79852–79895, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8138d211ce8790fdfbeeeb9781838a37-Abstract-Conference.html.

Zhongkai Yu, Chenyang Zhou, Yichen Lin, Hejia Zhang, Haotian Ye, Junxia Cui, Zaifeng Pan, Jishen Zhao, and Yufei Ding. Chipbench: A next-step benchmark for evaluating llm performance in ai-aided chip design. *arXiv preprint arXiv:2601.21448*, 2026. URL https://arxiv.org/abs/2601.21448.

Zhipu AI. Model api pricing. https://open.bigmodel.cn/pricing, 2026. Accessed 2026-09-23.

Peilong Zhou, Zhirong Chen, Cangyuan Li, Haoyu Gao, Kaiyan Chang, Ziming Qu, and Ying Wang. Closer-bench: Evaluating budgeted cross-stage design closure for hardware agents. *arXiv preprint arXiv:2607.16632*, 2026. URL https://arxiv.org/abs/2607.16632.

A Task Families and Provenance

The confirmatory family registry is balanced across domains. AI covers per-channel requantization, tiled depthwise computation, and a mixed-precision special function; DSP covers a stateful FIR, FFT butterfly, and complex MAC; cryptography covers AES MixColumns, SHA state, and Keccak permutation; stream/memory covers strided load, variable reduction, and gather/scatter. Each public card records the task ID, source project and commit, license, native function, transformation, interface requirements, fixed assets, editable paths, and hashes for evaluation evidence.

Table 4: Confirmatory task-family cards. Every row also has an admitted independent oracle, native/hidden workload, CV-X-IF/core judge, full-core synthesis pair, and sealed numeric and schedule instance space.

Domain	Task family	Required operation
AI	`ai-mixed-precision-sfu`	INT8 per-channel requantization with activation approximation
AI	`ai-per-channel-requant`	INT8 convolution with per-channel requantization
AI	`ai-tiled-depthwise`	INT8 depthwise tiled accumulation
DSP	`dsp-complex-mac`	Q15 complex multiply-accumulate with rounding and saturation
DSP	`dsp-fft-butterfly`	Fixed-point FFT butterfly and complex MAC
DSP	`dsp-stateful-fir`	Stateful Q15 FIR with explicit state reset
CRYPTO	`crypto-aes-mixcolumns`	AES round with MixColumns and byte-order contract
CRYPTO	`crypto-keccak-permutation`	Keccak theta-rho-pi-chi-iota state permutation
CRYPTO	`crypto-sha-state`	SHA-256 state update with rotate/mask semantics
STREAM	`stream-gather-scatter`	Strided gather-compute-scatter with back-pressure
STREAM	`stream-strided-load`	Strided stream with ready/valid stalls
STREAM	`stream-variable-reduction`	Stateful variable-latency reduction

Provenance constrains what "native" means. The task bundle stores the upstream file/function hash, adapter patch, build command, compiler flags, input source, baseline invocation, extension invocation, and expected output. Trace-backed adapters and handwritten stress tests are labeled separately.

B Oracle Admission and Mutation Audit

Admission precedes all confirmatory model scores. The locked admission report records 12/12 passing family oracles for semantics, software, RTL, protocol/reset/flush, CV32E40X execution, native and hidden workload, and full-core synthesis. Candidate code cannot import the reference implementations or read hidden inputs.

Mutation testing probes cross-artifact sensitivity. The confirmatory suites contain 272 certified non-equivalent mutants, at least 22 per family, and kill 272/272. Categories include encoding/schema inconsistencies, software operand and intrinsic faults, numeric semantics, RTL arithmetic and state faults, handshake errors, reset/flush behavior, and integration mistakes. Passing this audit supports judge sensitivity to these injected faults, not exhaustive verification.

Table 5: Ordered delivery gates. G11 is the separately reported benefit gate and is not part of the strict delivery conjunction; independent diagnostics are never substituted for this cumulative order.

Gate	Boundary	Independent executable check
G1	Submission	Package schema, required files, and workspace allowlist
G2	ISA contract	Requirements and typed contract agree
G3	Encoding	Opcode legality and cross-artifact encoding consistency
G4	Software mapping	Intrinsic/.insn build, operand order, and dynamic invocation
G5	Semantics	Public/private differential and metamorphic cases
G6	Integrated RTL functional	Compile compute plus wrapper; run numeric/stateful simulation
G7	Protocol stress	CV-X-IF issue/commit/writeback, stalls, reset, and flush
G8	Core integration	CV32E40X execution and base-ISA regression
G9	Native workload	Application output on the extension path
G10	Hidden/quality	Hidden workload, numerical, and security checks
G12	Full-core synthesis	Paired area, timing, and vectorless-power feasibility
G13	Budget	Locked logical-resource compliance

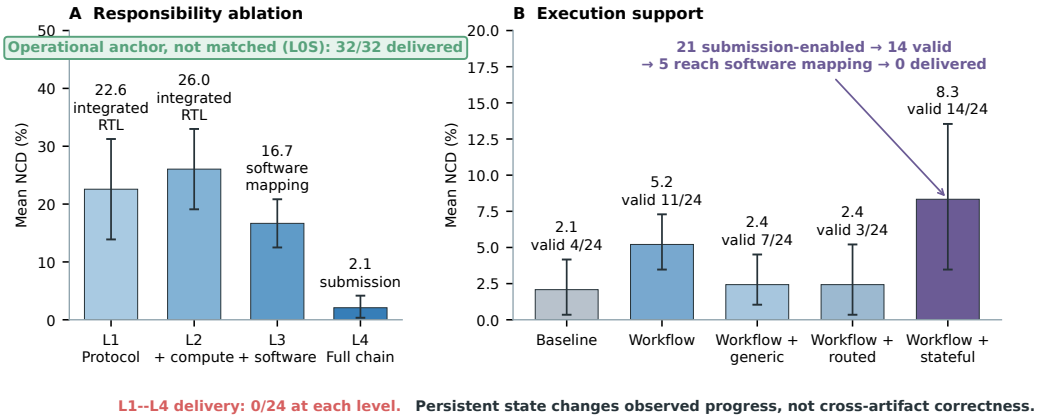

Figure 4: **Responsibility and execution-support diagnostics. A**, mean NCD and family-clustered 95% intervals for L1–L4, with the dominant first failure. The green annotation is the separately locked, nonmatched L0S operational anchor; full delivery is 0/24 in L1–L4. **B**, mean NCD under 5 execution-support conditions; labels give valid submissions. The controller condition is nonconcurrent and bundled. NCD records ordered progress, not delivery.

C Responsibility and Execution-Support Protocols

The responsibility ablation preserves the final judge while changing supplied artifacts.

Table 6: L1–L4 responsibility-ablation results. Mean NCD intervals are family-clustered; NCD is not delivery.

Level	Agent responsibility	Valid submission	Dominant first failure	Full delivery	Mean NCD [95% CI]
L1	Protocol RTL	13/24	RTL functionality (13/24)	0/24	22.6 [13.9, 31.2]
L2	Compute + protocol RTL	15/24	RTL functionality (15/24)	0/24	26.0 [19.1, 33.0]
L3	+ software/workload	16/24	software mapping (16/24)	0/24	16.7 [12.5, 20.8]
L4	Full front-end chain	4/24	submission (20/24)	0/24	2.1 [0.3, 4.2]

Execution-support conditions preserve family keys, workspace, hidden judge, logical budget, and seed. Baseline requests validation and submission; workflow adds stage order and earliest-failure repair; generic and routed skills expose static or selected guidance; the stateful condition appends public stage state and gates terminal submission. Routed and stateful conditions are nonconcurrent follow-ups. The controller exposes public state only and receives no hidden-judge result.

Table 7: Execution-support diagnostics. Every condition has zero full delivery.

Condition	Valid submission	Full delivery	Mean NCD [95% CI]
Baseline (v2.1 ref.)	4/24	0/24	2.1 [0.3, 4.2]
Workflow	11/24	0/24	5.2 [3.5, 7.3]
Workflow + generic skill	7/24	0/24	2.4 [1.0, 4.5]
Workflow + routed skill	3/24	0/24	2.4 [0.3, 5.2]
Workflow + stateful controller	14/24	0/24	8.3 [3.5, 13.5]

D Full Model-by-Family Results

Capability varies with responsibility scope and supplied artifacts. The complete machine table is `reports/v2.0-final/model-arm-stage-results.json`; it preserves scheduled and

provider-completed denominators for every system–condition cell. The tables below report local RTL, ordered cross-artifact progress, and strict delivery under their registered responsibility conditions.

Table 8: **Matched A0 cells that cumulatively pass G6 and first fail G7.** G8–G12 are independent downstream records and do not restore cumulative delivery after G7 fails. The public-safe ledger exposes the failed gate but not its hidden trigger or waveform, so no specific protocol mechanism is assigned.

System	Family	Seed	G6	G7	G8–G12	G13
GPT-5.6 Sol	Gather/scatter	3303	pass	fail	pass	pass
GPT-6 Astra	Gather/scatter	1101	pass	fail	pass	fail
GPT-6 Astra	Gather/scatter	2202	pass	fail	pass	pass
GPT-6 Astra	Strided load	1101	pass	fail	pass	pass
GPT-6 Astra	Strided load	2202	pass	fail	pass	pass
GPT-6 Astra	Strided load	3303	pass	fail	pass	pass

The focused responsibility ablation uses one common agent scaffold.

Table 9: Descriptive mean NCD by model–agent configuration and responsibility condition; full delivery is zero throughout.

Model	L1	L2	L3	L4	L4 full delivery
DeepSeek-V3.2	26.0	26.0	21.9	2.1	0/8
Kimi-K2.5	20.8	26.0	18.8	3.1	0/8
MiniMax-M2.5	20.8	26.0	9.4	1.0	0/8

Supplying admitted RTL separates operational delivery from implementation authorship. Table 10 reports the prospectively locked L0S schedule. GPT-5.3 Codex Spark was unavailable in all four cells; every one of the 32 provider-completed cells delivered. Three initial parallel-judge runs encountered explicit host CPU, memory, or process-resource exhaustion. The raw ledger is preserved; only those unchanged submissions were rejudged serially, all passed, and no model was rerun.

Table 10: RTL-provided operational delivery. Supplied implementation artifacts remain read-only; Delivered is strict DELIVER@BUDGET after documented serial adjudication of three host-resource faults.

System / agent	Scheduled	Completed	Delivered
DeepSeek-V3.2 / mini-SWE-agent	4	4	4
GPT-5.3 Codex Spark / Codex	4	0	0
GPT-5.5 / Codex	4	4	4
GPT-5.6 Luna / Codex	4	4	4
GPT-5.6 Sol / Codex	4	4	4
GPT-5.6 Terra / Codex	4	4	4
GPT-6 Astra / Codex	4	4	4
Kimi-K2.5 / mini-SWE-agent	4	4	4
MiniMax-M2.5 / mini-SWE-agent	4	4	4

A unified short-budget diagnostic compares frontier systems under one A0/B0 protocol. Table 11 replaces the earlier mixture of formal, short-budget, and stopped-run rows with a single descriptive setting: every system receives the same 12 task families, three seeds, A0/B0 prompts, independent judges, 120-second wall-clock guard, six model calls, and six tool calls. Six rows come from the prospectively locked v2.25 completion; GLM-5.3 and the two Claude rows come from the already locked v2.15 study with the identical logical budget. The rows compare complete model–agent configurations rather than isolated models: GPT uses pinned Codex CLI, while the other systems use mini-SWE-agent 2.4.6 and their registered provider routes.

Only GPT-6 Astra obtains nonzero ordered progress in this short-budget setting: two A0 cells form valid submissions and first fail integrated RTL functionality (G6), producing mean A0 NCD 2.3%. Its B0 score is zero. Every other A0/B0 group first fails submission validity (G1), and all nine systems deliver 0/36 B0 cells. Thus Table 11 is a uniformly budgeted stress diagnostic, not a replacement for the matched 720-second A-1/A0/B0 result in the main paper.

Table 11: **Unified 120-second frontier diagnostic.** Every A0 and B0 row contains 36 provider-completed family–seed cells under a 120-second, six-model-call, six-tool-call budget with common prompts and judges. NCD is ordered progress. Tokens and USD equivalents combine A0 and B0 and are list-price-equivalent sensitivity estimates, not billed spend. Codex timeouts did not emit complete-turn token telemetry, so the two GPT token and cost entries are unavailable. The DeepSeek range brackets published off-peak and peak direct-API rates.

System	A0 NCD (%)	B0 NCD (%)	B0 delivered	Tokens (M)	USD equiv.
GPT-6 Astra	2.3	0.0	0/36	n/a	n/a
GPT-5.6 Sol	0.0	0.0	0/36	n/a	n/a
DeepSeek V4 Pro	0.0	0.0	0/36	1.91	\$0.67–1.34
Qwen3.5-397B-A17B	0.0	0.0	0/36	0.88	\$0.19
MiniMax M3	0.0	0.0	0/36	1.35	\$0.22
GLM-5 Turbo	0.0	0.0	0/36	0.92	\$0.42
GLM-5.3	0.0	0.0	0/36	1.28	\$0.88
Claude Opus 5	0.0	0.0	0/36	2.02	\$2.37
Claude Fable 5.1	0.0	0.0	0/36	2.49	\$2.92

The pricing snapshot uses version-matched public rates and converts CNY at 1 USD = 6.7001 CNY using the 22 September 2026 ECB reference rates (European Central Bank, 2026; OpenAI, 2026; DeepSeek, 2026; Alibaba Cloud Model Studio, 2026; MiniMax, 2026; Zhipu AI, 2026). The Claude values use the mini-SWE-agent/LiteLLM per-trajectory list-price-equivalent telemetry. Gateways did not expose a common auditable billed-spend field. Response identity is explicit in 430/432 v2.25 cells; two Qwen responses omit the response-side model field but retain the frozen exact-model request and provider preflight.

E Benchmark Scale Audit

The benchmark reports independent family breadth separately from registered variation, sealed evaluation depth, mutation sensitivity, and delivery-gate depth. This differs from broad agent suites that derive breadth from many independent research, web, or tool-use tasks (Kon et al., 2026; Hsu et al., 2026; Wu et al., 2026); the units are not treated as directly commensurate.

Table 12: **Benchmark scale separates independent breadth from executable depth.** "Hidden entries" sum the explicit `case_count` fields or disjoint `category_counts` in the public hidden-audit certificates. They are within-family tests, not additional tasks or statistical units. The total is dominated by 2 exhaustive memory suites with 4,097 cases each.

Scale axis	Frozen extent	Inferential role
Independent breadth	12 families; four domains; five upstream projects	Statistical clusters and the stated coverage boundary
Registered variation	48 axes: 27 numeric/data/workload and 21 temporal/protocol	Construction diversity within families
Sealed evaluation	8,403 hidden entries; median 24, range 8–4,097 per family	Within-family stress, not independent breadth
Judge validation	272/272 certified mutants killed; 22–24 per family	Sensitivity to injected cross-artifact faults
Delivery depth	12 ordered strict gates	Consecutive closure from submission through core, workload, and synthesis

F Runtime, Budget, and Provider Accounting

Logical budgets are frozen independently of provider availability. Primary trajectories allow 36 model calls and 36 tool calls under a 720-second safety guard. Provider-unavailable trajectories retain their scheduled cells, are excluded from capability-conditional denominators, and are never

converted into model failures. The primary accounting identity is 432 scheduled = 386 provider-completed + 46 infrastructure-unavailable.

The post-freeze runtime probes are exploratory and excluded from every frozen denominator. The v2.3 12-to-24-minute contrast increases observed partial progress. The separately locked v2.7

Table 13: **Additional runtime does not restore delivery.** Each row contains four domain-balanced GPT-5.6 Sol/Codex cells. Contrasts are paired only within a probe: v2.3 uses workflow guidance and its original judge, whereas v2.7 uses the resampled B0 task package and judge.

Probe	Runtime	Valid submission	Full delivery	Mean NCD	Mean wall time
v2.3 workflow	12 min	1/4	0/4	6.2	720s
v2.3 workflow	24 min	3/4	0/4	18.8	1440s
v2.7 B0	12 min	1/4	0/4	6.2	720s
v2.7 B0	48 min	1/4	0/4	6.2	2880s

12-to-48- minute contrast obtains one valid submission and 0/4 deliveries in each arm; mean model calls remain 3.25 and mean tool calls change only from 13.50 to 15.25. All eight v2.7 trajectories end at the runtime cap before a terminal usage event, so their token totals and price equivalents are censored. An initial v2.7 launch exposed a judge-environment precheck defect after model execution; all of its cells were excluded and the complete eight-cell replacement schedule was rerun without reusing any candidate. The accompanying v2.3 gateway matrix produced no model responses and likewise has no scientific score. This result bounds only the tested range, not unlimited compute.

A post-hoc observable-interaction audit distinguishes wall time from effective work. Per-event timestamps and provider latency are unavailable, so the trace cannot allocate seconds among inference, tool execution, waiting, or idle time. Across the four cells, total model calls are 13 in both arms and completed command events are 54 versus 61. Commands containing inspection/retrieval operations number 47 versus 43; completed file-change events number 3 versus 4; and validator invocations number 3 versus 7. These indicators may overlap and are not time allocations. Neither arm contains a trace-visible `rf-v2 submit` invocation. The 4× condition is therefore a fourfold wall-clock guard, not a fourfold effective-work treatment.

G PPA Methodology and Raw Evidence

PPA is evaluator-run, paired, and conditional. The registered flow synthesizes the same CV32E40X plus CV-X-IF top for baseline and candidate under identical library, clock, operating condition, constraints, Tcl, and parser settings. It records area, timing, and vectorless power; no workload SAIF/VCD, placement, routing, or signoff is used. Raw public-safe summaries and hashes are indexed by each task's oracle-admission record.

Independent feasibility does not create a delivered path. The evaluator obtains 11 independent reports and 3 feasible observations, but cumulative survival has already reached zero. Thus 3/11 characterizes only candidates for which that independent measurement exists.

The cycle-observed annotations show why feasibility is not benefit. Two low-area candidates obtain 1.87–2.15× cycle speedups and satisfy their implementation envelopes. A third obtains a 1.64× speedup but uses 159% of its area-overhead allowance and is infeasible. The figure reports all 11 synthesis observations rather than selecting only the three with cycle counts.

H Security and Reproducibility

The release and run-time isolation boundaries are hash-locked. The release provides family cards, native adapters, schemas, submission CLI, protocol definitions, executable evaluators, evaluation inputs, independent oracle and reference implementations, immutable result ledgers, analysis code, figures, and the paper. Reproducing users can run every delivery gate locally. During a scored run, the evaluator mounts evaluation inputs and reference implementations outside the agent workspace, so the agent cannot read them even though they are available to the reproducer. Credentials, raw provider logs, and proprietary synthesis reports are excluded. Full-core synthesis requires

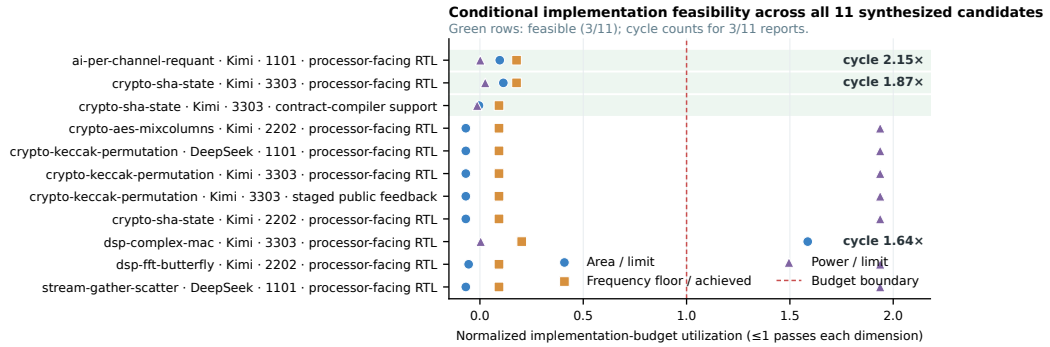

Figure 5: **Conditional implementation feasibility across all synthesized candidates.** Each row shows area, timing, and vectorless-power utilization normalized by that family's registered limits; values at or below one pass the corresponding dimension. Green row shading marks the 3/11 reports satisfying all three limits. Cycle speedup is annotated where both hidden-workload counts exist (3/11). None lies on a delivered cumulative path.

user-supplied licensed EDA tools and target technology libraries. A deterministic manifest covers every released file and evaluation-material payload.

I Stopped Pilots and Excluded Diagnostics

Stopped or post-freeze studies never enter the primary denominator. The v2.4 rigid-phase controller pilot stopped after its sole development cell and is excluded. The A-1 model-breadth study extends only the local RTL diagnostic across additional model–agent adapters; it does not alter the 108-cell A-1 denominator or support cross-adapter model-effect claims. The runtime probes in Appendix F, routed-skill condition, stateful-controller condition, and L1–L4 responsibility ablation remain separately locked diagnostics.

Local INT8 arithmetic is within the observed capability envelope. The historical development task is a four-lane packed signed-INT8 dot/reduction primitive with requantization provenance, rather than an entire GEMM accelerator. All 33 historical cross-artifact pilot trajectories are classified as never initialized in the frozen failure analysis: none records successful initialization, validation, or final submission. In one high-budget trace, the candidate compute RTL implements the intended four-lane dot product and validation passes; submission then first fails before initialization, later creates a submission, and finally encounters the diagnostic runner's workspace-binding/finalization inconsistency. These pilots are excluded from formal scores. By contrast, the corrected A-1 local control passes 4/4 provider-available cells across two systems and two budgets. These observations separate arithmetic competence from cross-artifact orchestration and delivery.

The unified Table 11 study remains separate from the main confirmatory denominator. The v2.25 completion schedules 432 cells (six configurations $\times$ 12 families $\times$ three seeds $\times$ A0/B0) under 120 seconds and six model/tool calls. All 432 effective cells are provider-completed after prospectively documented replacement of launch attempts censored by container PID exhaustion; censored attempts made no attributable contribution to the effective scores. GPT-6 Astra obtains A0/B0 NCD 2.3/0.0, while GPT-5.6 Sol, DeepSeek V4 Pro, Qwen3.5-397B-A17B, MiniMax M3, and GLM-5 Turbo obtain 0.0/0.0. The matching v2.15 protocol supplies complete GLM-5.3, Claude Opus 5, and Claude Fable 5.1 rows, each at 0.0/0.0. All nine B0 groups deliver 0/36. The earlier v2.24 official-BigModel GLM-5.3 run remains a stopped audit record and no longer contributes a zero-imputed Table 11 score. GPT-5.6 Luna failed its pre-specified development admission rule; DeepSeek V4.1 Flash was not executed, and the historical V4 Flash identifier is not retroactively aliased.

Full prompts and protocol locks are released for audit. They appear under `configs/prompts-v2*`, `benchmark-v2/difficulty-ladder`, and the associated denominator-

lock reports. Stopped pilots, failed provider launches, and post-freeze diagnostics remain outside primary delivery.

The v2.12 responsibility diagnostic preserves its runner incident. Its first concurrent three-cell launch reached the 720-second guard with empty trajectories, zero model/tool tokens, and no submissions. Those cells are infrastructure-censored, and the next three in-flight cells were stopped before completion. Before any replacement call, the incident, zero-event classification, unchanged fixtures/budget/judge, and complete 12-cell sequential recovery schedule were hash-locked and pushed. All 12 sequential cells completed without an infrastructure failure; Table 3 uses only that recovery ledger.

The recovery ledger records one stream/memory PR delivery; the other three PR cells first fail at G6. WS and FS each contain three G6 failures and one G1 no-submission outcome.