VERDICT: VERIFIER-DECOUPLED INFERENCE-TIME CROSS-VERIFICATION FOR VERILOG GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time scaling for Verilog samples several designs and must return one without a testbench. With the pool's own selector, the return, not the sampling, is where correct designs are lost: on VerilogEval-v2, nine samples from two cheap model families contain a correct design for 93.2% of problems, more than Claude-Opus-4.8 solves in one shot, yet the pool's cross-verifier returns one for only 85.3%, and twelve pool-internal selectors all stay below one-shot. We present VERDICT, a verifier-decoupled, testbench-free harness that separates selection from generation. VERDICT freezes a cheap pool so that every selector is scored on the same candidates; makes selection a swappable slot whose default extends ChipMATE's Verilog–Python cross-verification; and, before the slot, runs execution tools that drop candidates which fail to compile or contradict a waveform the specification itself prints, plus a behavior gate that calls the selector only where survivors differ. Filling the slot with Claude-Opus-4.8, which judges but never writes Verilog, recovers 5.6 percentage points on VerilogEval-v2 and 4.2 on RTLLM-v2, positive on every seed, bringing the harness to 90.8%, statistically indistinguishable from a matched frontier one-shot (91.2%) that writes the design itself. On VerilogEval-v2 the tools add 1.5 points to the unchanged cross-verifier without losing a problem; waveform replay covers the 12 of 156 specifications that print one and was calibrated on them. Run over the frontier model's own samples, the same harness reaches 92.9% against one-shot's 91.2% (not significant). VERDICT's accounting also explains when a harness pays off: a pool gains over one-shot exactly when its margin exceeds its selection loss. VERDICT thus turns "sample more" into three quantities to report: coverage, selection loss, and cost.

1 INTRODUCTION

Test-time scaling is a common way to raise the pass rate of LLM-written code: sample several candidates, then let tests, execution agreement, or a model return one (Li et al., 2022; Chen et al., 2023; Shi et al., 2022). Verilog (RTL) generation inherits this recipe without its checker: at inference the specification and module interface are available, but an independently authored testbench is not. ChipMATE dispenses with the golden testbench by cross-verifying independently written Verilog and Python implementations (Yu et al., 2026a). Such agreement is evidence, but it cannot by itself say which implementation follows the specification.

On VerilogEval-v2, the correct design is usually already in the pool: nine samples from cheap models contain one for 93.2% of problems (the *pool oracle*), more than a matched-prompt Claude one-shot solves (91.2%). Yet the pool's own cross-verifier returns one for only 85.3%. For this selector the larger shortfall is selection, not generation. Across three generation seeds the cross-verifier misses an available correct design on 7.9% of problem-seeds (problem$\times$seed pairs), more than the 6.8% where the pool holds none. That loss is well above the pool's 1.9 percentage-point (pp) margin over one-shot. Twelve pool-internal selectors and the floor pick (the strongest member's first sample) all land within a 2.4pp band, and the model-written tests and repair loops we tried do no better.

To isolate what does, **VERDICT** freezes one cheap pool per generation seed and varies only the selector slot and the optional execution tools (Figure 1). On four Verilog benchmarks, with three generation seeds on VerilogEval-v2 and RTLLM-v2, we ask where selection loses correct designs (§3), what recovers them (§5), and when recovery pays (§5–6).

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.

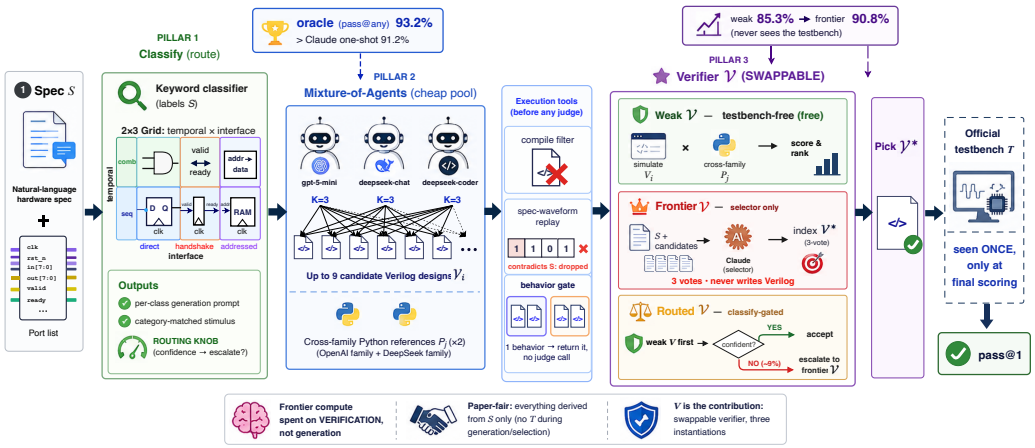


Figure 1: **VERDICT freezes the pool and swaps only the selector slot, so selection loss $\ell = o - r$ is measured on fixed candidates.** Left to right, a fixed keyword classifier (Pillar 1) reads the specification S and sets per-class prompts and stimulus. The frozen cheap pool (Pillar 2) holds at most nine candidates V_i per seed from three API names (two families, §2), plus cross-family Python references P_j. The optional execution tools (C2–C3) drop candidates that do not compile or contradict a printed waveform, and return a lone behavior without a judge. The selector slot (C1, Pillar 3) takes the cross-verifier ("weak"), the Claude judge ("frontier"; three votes per judged problem), or a routed variant escalating the least-confident 9% of problems to the Claude judge. We withdraw the routed variant's earlier cost point and find no cost advantage for it or any other configuration (§5.3). The official testbench T is absent at inference and scores the returned design once, offline. Values are on VerilogEval-v2 without tools: pool oracle $o = 93.2$, cross-verifier $r = 85.3$ ($\ell = 7.9$pp), Claude judge $r = 90.8$ ($\ell = 2.4$pp), matched one-shot $f = 91.2$ (margin 1.9pp).

VERDICT has three parts (§4): a swappable selector slot (C1), filters that reject candidates contradicting the specification's own printed waveform before any judge (C2), and a gate that calls the judge only where survivors differ (C3). Replacing only the cross-verifier with Claude-Opus-4.8 adds $+5.6 \pm 1.3$pp on VerilogEval-v2 and $+4.2 \pm 2.1$pp on RTLLM-v2 but none on CVDP-CID003. Both are positive on every seed and, on VerilogEval-v2, significant across seeds but not within seed 1 (§5.1). On VerilogEval-v2 the execution tools lift the unchanged cross-verifier on every seed, $+1.5$pp on average, with no regression. A second frontier judge, `gpt-5`, shows no gain over the cross-verifier in our run (-1.4pp, $p = 0.73$).

With the Claude judge and no execution tools, the harness is statistically indistinguishable from matched one-shot (90.8% against 91.2%, $p \geq 0.79$) although the frontier model never writes Verilog, at $\geq 5.0\times$ the cost. Without a frontier model it scores 89.7% at \$0.022 per problem, matched by randomly choosing `gpt-5-mini` or Claude one-shot per problem at equal expected cost (89.5%). Over Claude's own three samples the same judge finishes above one-shot, not significantly; averaged over seeds, every cheap Verilog pool we ran end to end loses more to selection than its margin over one-shot (§6).

Our contributions are:

- Where a testbench-free Verilog harness loses accuracy (§3).
- VERDICT, which swaps only the selector over a frozen pool: a Claude judge recovers part of the loss on VerilogEval-v2 and RTLLM-v2 but none on CVDP-CID003, and has no advantage over frontier one-shot (§4–5).
- Costs against price-matched one-shots and a margin/loss decomposition of each gap (§5–6).

2 Testbench-free selection: setting and measures

Task and selection signals. Given a specification S (prose and a port list), the harness returns one Verilog design $V^\star$. The official testbench T supplies the final label and is unavailable at inference.

The *selector* is the slot that returns $V^\star$; a *judge* is a model in that slot. The default instance, the *cross-verifier*, compares candidates' simulated traces with each other (V–V) and with Python implementations of S that the pool's models write (*references*; V↔P). Both checks observe behavior on chosen (*probe*) inputs; neither certifies equivalence to S. A *behavior* is a set of candidates with identical traces on those inputs.

Pool and measures. The default *pool* samples `gpt-5-mini` and the likely aliases `deepseek-chat` and `deepseek-coder` (two *families*; Appendix B) $K{=}3$ times each: at most nine candidates. The *pool oracle* o (*coverage*) is the fraction of problems with a correct candidate; the *selected accuracy* r is the pass@1 of the returned design. With f the pass@1 of frontier one-shot, the *selection loss* is $\ell = o - r$, the *margin* is $o - f$, and *capture* is r/o. The *floor pick* is `gpt-5-mini`'s first sample.

Benchmarks and controls. We use VerilogEval-v2 (156 problems) (Liu et al., 2023b; Pinckney et al., 2025a), RTLLM-v2 (48 of 50, two file-broken) (Lu et al., 2024; Liu et al., 2024b), ChipBench-SC (30) (Yu et al., 2026b), and CVDP-CID003 (78, cocotb-scored, $K{=}1$ each) (Pinckney et al., 2025b). On VerilogEval-v2 we re-measured frontier one-shot with the pool's exact generation prompt, including its module-interface block (the *matched* prompt). The earlier baseline omitted that block and scored 89.7%. Matched, one-shot scores $91.2{\pm}1.0\%$ (three draws; Appendix M), turning the Claude-judged harness's nominal $+1.1$pp lead into a nominal -0.4pp difference, not significant. The other three benchmarks keep the *unmatched* one-shot, and the effect of matching there is unmeasured (Appendix M). Results are means±sd across generation seeds unless a single seed is named; differences are computed from counts and rounded last (Appendix B).

Paper-fair protocol. Generation and selection never read T; their one input from reference RTL is the VerilogEval-v2 module header (port names, directions and widths, which its specifications also state in prose), given to the pool and the matched baseline alike. T is run once, offline, to label o and score r. Three RTLLM-v2 specification repairs drew on the testbench or the reference before generation. Every arm ran on the repaired benchmark, but the repairs may have affected the selectors unequally. On `ring_counter`, the RTLLM-v2 testbench prints its pass banner even after an output mismatch, which can inflate any arm by at most one problem (2.1pp; Appendix C). A further exception: we calibrated spec-waveform replay (§4) once, in development, against the 12 VerilogEval-v2 waveform problems' golden designs, which are never read at inference. Its gain (§5.2) is measured on those same 12 (Appendix P). Every run logs candidate counts and reference crash rates ($< 1\%$ after six development fixes). Before them, references crashed on 38–65% of clocked problems and masked every component's effect as 0pp (Appendices E–F).

3 Where accuracy is lost

We freeze each seed's stored pool and vary only the selector. The main comparison and the twelve-selector sweep use three VerilogEval-v2 seeds (468 problem-seeds; Appendix G), per-class and pool-size experiments use seed 1, and Appendix O adds thirteen more experiments.

Observation 1: on VerilogEval-v2, the pool holds a correct design more often than one frontier sample writes one. Nine cheap samples reach an oracle of $93.2{\pm}1.0\%$, against $91.2{\pm}1.0\%$ for matched Claude-Opus-4.8 one-shot, and the oracle leads on every seed. However, the lead is nine samples against one. Three Claude samples already cover 92.3–94.4% across draw sets, bracketing the nine-sample pool (Appendix M). The pool's $+3.6{\pm}1.3$pp oracle gain over its best member's three samples likewise confounds sample count with model diversity. On seed 1, the headroom above the floor pick sits in sequential (clocked, stateful) designs (Figure 2(a); Appendix H). Selection, not generation, is the larger shortfall. The pool holds no correct design for 6.8% of problem-seeds, yet the cross-verifier misses one the pool holds for 7.9% (per seed $11/14/12$ against $9/11/12$, a tie on seed 3). On RTLLM-v2, by contrast, the 81.2% oracle sits below unmatched one-shot's 85.4%, so generation binds first and no selector can reach one-shot.

Observation 2: all twelve pool-internal selectors stay below one-shot, inside a 2.4pp band; the cross-verifier's loss concentrates in disputed problems. All twelve and the floor pick fall in 83.3–85.7% (Figure 2(b)); a learned selector over the same features reaches 83.3% (seed 1,

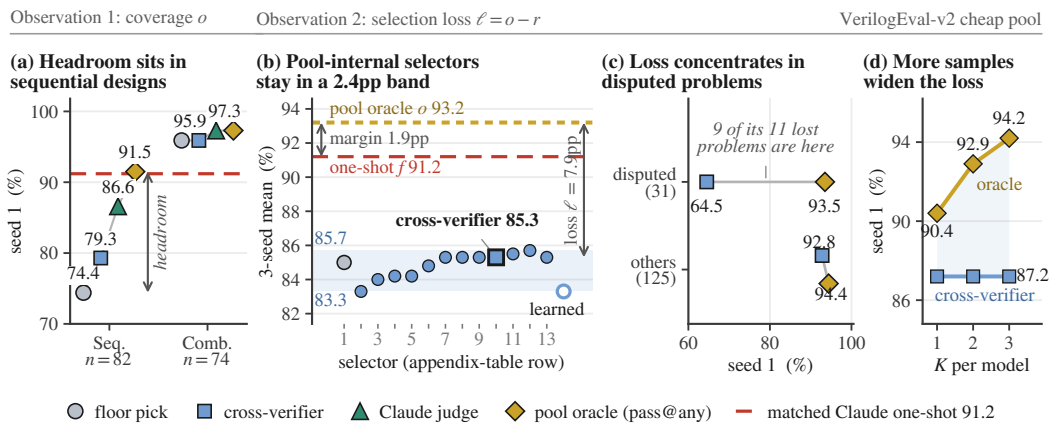


Figure 2: **Coverage is not accuracy: the cheap pool holds a correct design more often than one frontier sample writes one, and every pool-internal selector returns fewer** (VerilogEval-v2; dashed line: matched one-shot 91.2%). **(a)** Headroom sits in sequential designs (seed 1 by class; Appendix Table 3). **(b)** The twelve selectors and the floor pick (Appendix Table 2) fall in a 2.4pp band, and the cross-verifier's loss $\ell = 7.9$pp exceeds the pool's 1.9pp margin (three-seed means; open circle: learned selector, seed 1). **(c)** The loss concentrates on the 31 disputed problems (seed 1). **(d)** More samples per member widen it (seed 1).

Appendix O(iii)). The band bounds only the selectors we tested. Over three seeds the cross-verifier adds only $+0.2\pm2.1$pp over the floor pick on VerilogEval-v2 and $+4.2\pm3.6$pp on RTLLM-v2; its 2.6pp seed-1 lead is sampling variation. Call a problem *disputed* when its largest V–V agreement cluster covers less than half the pool. On seed 1, the 31 disputed problems hold 9 of the cross-verifier's 11 losses: it returns a correct design on 20 although 29 contain one (Figure 2(c); Appendix O(ii)). More samples widen the loss: from $K{=}1$ to $K{=}3$ per member the oracle rises from 90.4 to 94.2% while the cross-verifier stays at 87.2% (seed 1, Appendix O(i)).

Insight 1. Coverage is not accuracy (Observations 1–2). A pool can hold more correct designs than one-shot writes and still return fewer. Report o, r and $\ell = o - r$ for every configuration.

Observation 3: checks whose expected outputs come from a model's reading of the specification did not recover the loss. On a fresh four-family pool, selecting by eight model-written testbenches per problem scores 80.1%, below that pool's 84.0% floor pick. A reference-consensus repair loop gains $+0.2\pm0.7$pp, and candidate synthesis loses one problem (Appendix O). With a simulator, `gpt-5` as judge scores 83.3% on seed 1 against its own 87.2% under the same protocol without the tool. Inspected transcripts show it testing for outputs its own misreading of the specification predicted (Appendix O(x)). Appended to a cheap judge's otherwise identical input (`deepseek-chat`, §5.3), execution evidence yields 108 correct choices against 115 without it, on the 154 problem-seeds the gate (§5.2) sent to a judge ($p = 0.14$; Appendix P). Each check may inherit its author model's misreadings (untested).

Insight 2. Filter with the specification's own outputs, not a model's reading of them (Observation 3). The printed waveform is the one expected-output source that passes through no model. Where the specification prints its expected outputs, reject candidates that contradict them before any judge. RTLLM-v2 prints none (§5.2).

4 VERDICT: SEPARATING SELECTION FROM GENERATION

VERDICT turns Insights 1–2 into three components (Figure 1). Before a swappable selector slot (C1), *execution tools* filter candidates (C2) and gate selection (C3).

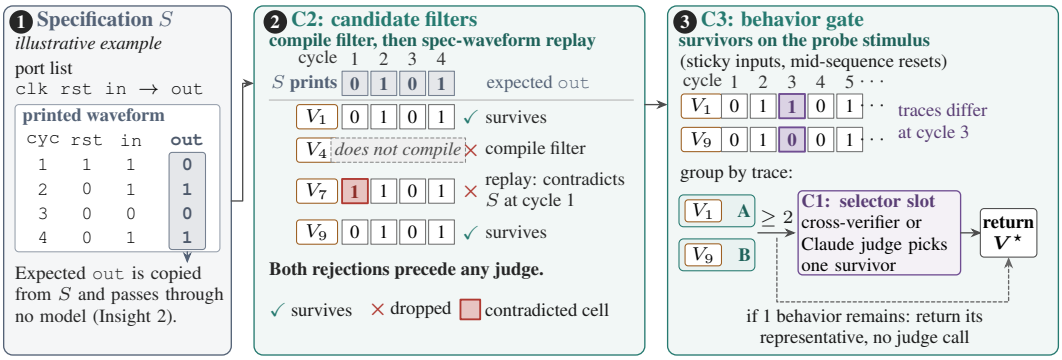


Figure 3: **Execution tools reject what the specification's own printed waveform contradicts, and call a judge only where the survivors still differ** (illustrative values). ① S prints a waveform whose `out` column is the expected output, taken from S itself, not from any model's reading of it. ② C2: the compile filter drops V_4 and replay drops V_7, whose cycle-1 output contradicts S's own row (red). Replay skips unknown (`x`) entries and never drops the last survivor. ③ C3: survivors with identical probe traces form one behavior; here V_1 and V_9 differ at cycle 3, so the selector slot C1 chooses. A lone behavior is returned without a judge.

(C1) Agreement leaves correct designs unselected: make the selector swappable. Following Insight 1, the selector accepts a stored pool and returns an index, so its loss can be measured without regenerating candidates. The default cross-verifier extends ChipMATE's V↔P comparison (Yu et al., 2026a) with V–V agreement. It simulates every candidate and reference on 40 random input vectors and compares each candidate with other families' references. Starting from the floor pick, it promotes a candidate by full-match count, then V–V agreement, then mean match rate (margin 0.15). The seed-1 VerilogEval-v2 pool, used by every seed-1 experiment below, has two references per family where every other pool has one; one per family gives 86.5 or 85.9%, not 87.2% (Appendix B).

The *frontier judge*, Claude-Opus-4.8 (the *Claude judge* of Table 1), reads S and text-deduplicated candidates and returns an index by three-vote majority. It never writes Verilog in this role. Its prompt asserts that exactly one candidate is correct (the *single-correct premise*), false whenever two distinct candidates are correct; most frontier-judge arms used it. In the frontier-free configuration, removing it halved a cheap judge's cost with no detectable accuracy change on stored pools (89.7 to 88.9%, $p = 0.45$). A pre-registered confirmation found no effect on answer rate (Appendix Q).

(C2) A judge can misread expected behavior: reject direct contradictions first. Following Insight 2, two filters act before any judge, and neither depends on a model's reading of S (Figure 3). The compile filter drops candidates that cannot compile or simulate. When S prints a waveform table for the top module, spec-waveform replay drops every candidate that contradicts it, unless that would drop them all. Two parser guards (sub-module tables; header aliases and hexadecimal tables) postdate the golden-design calibration of §2 and are not covered by it. In an offline audit of the 12 waveform problems over three seeds, all 107 testbench-correct candidates pass the replay and 97 of 105 incorrect ones contradict it (Appendix P).

(C3) A judge is needed only where survivors differ: gate judgment on disagreement. Observation 2 places most of the loss on disputed problems, so the behavior gate calls the selector only when survivors' traces differ. Otherwise the probes cannot separate them (though T might), and the gate returns a representative. The gate changes which problems reach a judge, not how judges decide. On recorded verdicts it adds no accuracy and can cost one problem when a behavior group's representative is wrong (seed 2). Its one benefit, fewer judge calls, is left unpriced (§5.2; Appendix P).

Frozen apparatus. A specification-only keyword classifier assigns temporal and interface classes, which select generation prompts and stimulus (Appendix A) and are held fixed when the selector is substituted. Class-specific selection variants add 0.0 to +0.4pp over three seeds (−1.3pp combined on seed 1), so the classes only locate headroom (§5.1, Appendix H). The repair and synthesis loops of Observation 3 are disabled (Appendix O).

Table 1: **Changing the selector recovers part of the selection loss but does not establish an advantage over frontier one-shot.** Selected accuracy (%, mean±sd); the oracle row is coverage (pass@any), not a selected accuracy. Cost is per VerilogEval-v2 problem.

Configuration	VerilogEval-v2	RTLLM-v2	Cost ($)
Strongest pool member, `gpt-5-mini` (mean pass@1)	84.1±0.5	64.8±2.6	0.0032
Cheap pool oracle	93.2±1.0	81.2±2.1	—
cross-verifier	85.3±1.7	72.2±2.4	0.017
Claude judge	90.8±1.0	76.4±3.2	$\geq$ 0.141
tools + cross-verifier	86.8±1.5	—	0.017
tools + Claude judge (recorded-verdict bounds)†	[91.5, 92.9]	—	—
tools + DeepSeek judge	89.7±2.3	—	0.022
one per behavior (exploratory)	89.7	—	0.019
Claude K=3 pool + Claude judge	92.9	—	0.179
Claude one-shot	91.2±1.0	85.4*	0.028

† Recorded-verdict bounds (§5.2), not a fresh run. ∗ Unmatched prompt. Costs mix metered usage and estimates (Appendix Table 7); other configurations: Appendices B and I. Member row: pass@1 over retained samples, not the floor pick of Figure 2 (85.0); with missing draws as failures, 83.8/63.7 (Appendix B).

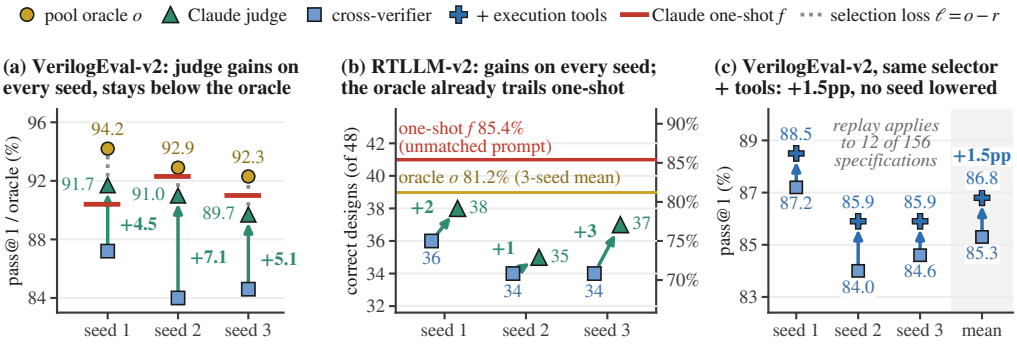


Figure 4: **Changing only the selector recovers accuracy on every seed but stays below the pool oracle; the tools add a smaller, regression-free share. (a)** VerilogEval-v2 per seed: arrows mark the Claude judge's gain over the cross-verifier, circles the pool oracle, and ticks the matched one-shot draw paired with each seed by convention. **(b)** RTLLM-v2: the Claude judge gains on every seed, but the three-seed oracle mean is below one-shot (unmatched prompt). **(c)** The execution tools raise the unchanged cross-verifier on every seed (mean 85.3 → 86.8%) and lower no problem.

5 Evaluating the Selector and the Tools

Four questions organize the evaluation: **Q1**, does changing only the selector reduce the loss? **Q2**, do execution tools before any judge raise accuracy with the selector unchanged? **Q3**, does the accuracy justify its cost against one-shot? **Q4** (§6), when does the margin exceed the loss? Table 1 answers Q1–Q3; per-seed counts are in Appendix B.

5.1 Q1: The Claude judge recovers part of the loss; a second judge does not

The selector substitution produces the largest controlled gain. With the pool fixed, the Claude judge raises accuracy on every seed: +5.6±1.3pp on VerilogEval-v2 and +4.2±2.1pp on RTLLM-v2 (Table 1, Figure 4). Across seeds the VerilogEval-v2 gain is significant (t-test $p = 0.019$), yet no run that kept per-problem verdicts reaches significance. Exact McNemar gives $p = 0.09$ on VerilogEval-v2 seed 1 (10 fixes, 3 breaks) and $p = 0.63/1.00/0.38$ on the three RTLLM-v2 seeds. VerilogEval-v2 seeds 2–3 kept only logged counts (Appendix B).

On seed 1 the gain lands where Observation 1 placed the headroom. On sequential problems the Claude judge lifts the cross-verifier's 79.3% to 86.6%, against an oracle of 91.5%; on combinational

problems it adds one design (Appendix H). The end-to-end gain over the strongest member (Table 1: $+6.7$/$+11.6$pp) confounds sampling budget with selection (Appendix I).

A second frontier judge and CVDP-CID003 show no gain. A spend cap stopped `gpt-5`'s three-vote run after 144 of 156 seed-1 problems. On that easier subset it scores 90.3%, against 91.7% for the cross-verifier and 93.8% for Claude (full seed: 87.2 and 91.7%), and neither difference is significant ($p = 0.73$, $p = 0.13$). Under a plain pick prompt on all 156 it reaches 87.8%, within one problem of the cross-verifier (Appendix S). On CVDP-CID003 the same substitution gains nothing. The Claude judge scores $42.3\pm1.8\%$ against the cross-verifier's 43.6% and, on seed 1, fixes one pick and breaks three; its prompt lacked the single-correct premise (Appendix B).

5.2 Q2: Execution tools recover a small part of the loss

Execution that rejects candidates before any judge lifts the cross-verifier on every seed. With the selector unchanged, the tools raise mean VerilogEval-v2 accuracy from 85.3 to 86.8% ($+1.5$pp) and lower no problem (Table 1). Composed with Claude's recorded seed-1 verdicts, replay fixes three errors and breaks none. One problem would need a fresh judgment, so seed 1 lies in $[93.6, 94.2]\%$ (from 91.7%). Bounding seeds 2–3, which lack per-problem verdicts, puts the three-seed mean in $[91.5, 92.9]\%$. These are recorded-verdict bounds, not fresh runs or confidence intervals; seed 1 is not a win over one-shot ($p = 0.227$). The gate sends $52/51/51$ of 156 problems per seed to a judge. We do not count the skipped calls as a saving, since they are unmetered and cheap (Appendix P).

The RTLLM-v2 arm no longer tests the tools out of sample. The tools were developed on seed 1 and frozen before seeds 2–3 were scored. On RTLLM-v2, where no specification prints a waveform, they simulated each file's first module instead of its top module, lowering the Claude-judge arm to $[74.3, 76.4]\%$. A fix made after seeing RTLLM-v2 restores 76.4% (one problem lost, one gained), so the arm is no longer held out.

5.3 Q3: No configuration demonstrates a cost advantage over one-shot

The Claude-judged harness costs more without an established accuracy advantage. Without the tools it scores 90.8% against matched one-shot's 91.2% (Table 1); the seed-1 paired interval $[-3.4, +6.0]$pp supports neither a win nor equivalence. It costs at least \$0.141 per problem against one-shot's metered \$0.028 ($\geq 5.0\times$; Figure 5(c)). We estimate judge cost at an output length below the measured median, so these figures are lower bounds, not intervals.

A frontier-free configuration is matched by mixing two one-shots. The cheap pool, tools, and `deepseek-chat` judge score $89.7\pm2.3\%$ at \$0.022 per problem, 1.5pp under one-shot, not significant ($p = 0.36$). Randomly choosing between `gpt-5-mini` and Claude one-shot at the same expected cost yields 89.5%. Judge cost is metered; the \$0.0135 generation cost is a 40-problem estimate. A one-per-behavior variant, whose judge reads one candidate per behavior (Table 1), keeps 89.7% at \$0.019, 1.1pp above the one-shot mix at that price. That lead rests on the same generation estimate, so we do not count it as a cost advantage (Appendix Table 7). The judge alias now resolves to DeepSeek-V4.1-Flash, newer than the pool's June members. Its own one-shot (75.6%) and the evidence ablation of Observation 3 ran at half the 89.7% row's output budget, so neither is budget-matched (Appendices P–Q).

A confidence-gated router adds nothing over a random gate. Escalating the weakest 9% of cross-verifier picks to the Claude judge scores 87.8% on seed 1, against a random gate's expected 87.6%. With hidden reasoning tokens priced, it costs at least \$0.0285 against one-shot's \$0.028. We withdraw the routed point (Appendix J).

6 Decomposing the gap to frontier one-shot

Q4: A pool's margin pays only if selection loses less. Since $r = o - \ell$ (Insight 1), $r > f$ exactly when the margin $o - f$ exceeds ℓ. On measured runs this is an identity, used diagnostically to separate a pool whose coverage falls short ($o < f$) from one whose selection consumes a positive margin ($0 < o - f < \ell$). Figure 5(a) places each configuration by margin and loss, and (c) prices it.

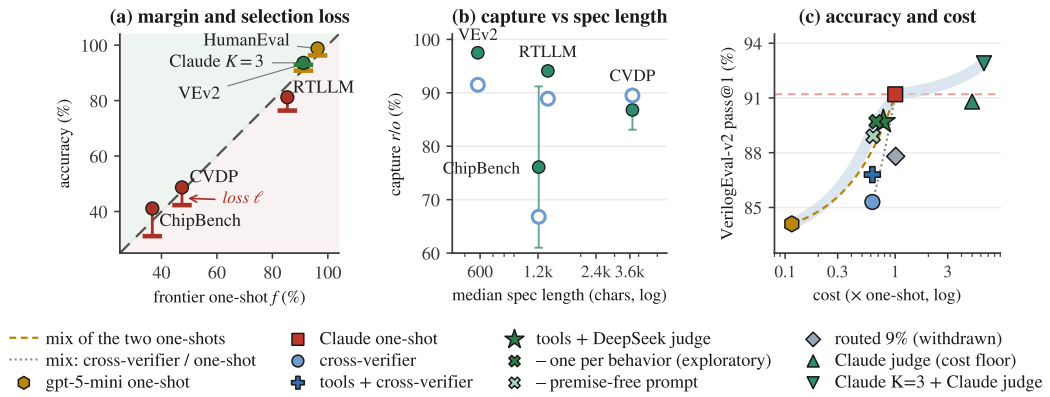


Figure 5: **A pool's margin pays only if selection loses less. (a)** On seed means no cheap Verilog pool finishes above one-shot after selection; only the frontier's own three samples stay above (+1.7pp, not significant); HumanEval (general code) for comparison. Dots: pool oracles; dashes: Claude-judge selected accuracy; diagonal: frontier one-shot f. **(b)** Claude-judge capture (filled; cross-verifier open) falls with median specification length on three benchmarks; ChipBench-SC breaks the trend. **(c)** Frontier one-shot dominates the full Claude-judged configuration. A "mix" chooses randomly between its two named configurations at matched expected cost. With mixtures admitted, the shaded frontier runs through `gpt-5-mini` one-shot, Claude one-shot and the Claude K=3 arm, and its one frontier-free vertex, the one-per-behavior variant, is exploratory (Appendix Table 7).

On seed means, every cheap Verilog pool loses more to selection than its margin over one-shot, even with the Claude judge. On VerilogEval-v2 the Claude judge leaves $11/468 = 2.4$pp of coverage unselected, above the 1.9pp margin. With this judge, absent designs (6.8pp) outweigh unselected ones (2.4pp), reversing Observation 1's order, yet the unselected share still decides the comparison with one-shot. Single seeds can fall either way: on seed 1 the Claude judge (91.7%) is +1.3pp above the paired one-shot draw (not significant). RTLLM-v2 has no margin (−4.2pp, unmatched one-shot). Against their unmatched one-shots, ChipBench-SC holds a 4.4pp margin (on the three-seed oracle mean) yet loses 10.0±6.7pp under the Claude judge, which equals one-shot only on its best seed (Appendix Table 5). CVDP-CID003 holds 1.3pp and loses 6.4±1.8pp (two seeds, two pools). The recorded-verdict bounds for tools + Claude judge ($[91.5, 92.9]\%$, §5.2) both exceed one-shot, but we do not count them: they were never run fresh, and the one fresh attempt, on seed 2, failed on a token cap (Appendix P).

Over the frontier's own three samples, the Claude judge loses one problem. On Claude's own K=3 pool (oracle 93.6%), the judge returns a correct design on 145 of the 146 problems that contain one: a 0.6pp loss against a 2.4pp margin. It scores 92.9% at $0.179 per problem, +1.7pp over one-shot but significant against none of the three one-shot draws. Unlike the cheap-pool runs, its votes were capped at 2,000 output tokens, and 13 of its 225 votes (three per judged problem) hit the cap. All three on the one miss did, so that pick fell back to the first candidate. Had the 13 finished, the loss could be 0.0–1.3pp (Appendix N). Pools of 9, 12 and 32 candidates over the same 156 problems give Claude losses of 2.6, 3.2 and 7.7pp, and the default pool plus one frontier sample 4.5pp (seed 1). These pools differ in contents as well as size, and every arm kept the single-correct premise, so neither size nor confusability is isolated (Appendices K and N).

Cross-benchmark patterns do not identify the cause of loss; margin and loss recur outside RTL. Claude-judge capture falls with median specification length across VerilogEval-v2, RTLLM-v2 and CVDP-CID003 (another judge prompt), where it drops below the cross-verifier's; ChipBench-SC breaks the trend (Figure 5(b)). Four benchmarks with one outlier establish no length effect (Appendix K). The margin and the loss recur on general code: on the 93 of 112 LiveCodeBench (Jain et al., 2025) problems with a non-empty frontier draw, the pool oracle is 84.9% against one-shot's 80.9% (three pooled draws), yet an execution-agreement cross-verifier loses 8.6pp. We ran no frontier selection there, and contamination could move the margin either way. On HumanEval (Chen et al.,

2021) the Claude-judged pool ties one-shot and leads it on HumanEval+ (Liu et al., 2023a) by one problem (Appendix R).

A labeled slice resolves large gaps more often than near ties. We label 25 problems per benchmark–configuration cell with the official testbench offline, estimate r and f, and commit to harness or one-shot only when the estimated gap exceeds 1pp. Where selection trails one-shot by more than 5pp, this test commits in 84–100% of random slices and is right in 93–100% of commitments. On the 1.3pp VerilogEval-v2 seed-1 near-tie, however, it is right in only 62.8%. This is a sampling-uncertainty analysis, not an out-of-sample test: the labeled problems are part of the gap being estimated (Appendix L, Table 8).

Insight 3. Coverage diagnoses; accuracy and cost decide (Q4). Use o to tell whether a pool lacks correct designs or fails to select them, then compare r and cost with one-shot directly.

7 Related work

Sampling and selection. Sampling raises coverage, but selection plateaus without an automatic check (Brown et al., 2024). Self-consistency, mixture-of-agents and compute allocation study how to spend inference compute (Wang et al., 2023; 2025; Snell et al., 2025), and AlphaCode and AlphaCodium add filtering or iterative execution feedback (Li et al., 2022; Ridnik et al., 2024).

Verification signals. CodeT selects by generated tests, MBR-exec by execution agreement and LEVER by learned execution scores, while trained verifiers, likelihood reranking and model judges supply other signals (Chen et al., 2023; Shi et al., 2022; Ni et al., 2023; Cobbe et al., 2021; Zhang et al., 2023; Zheng et al., 2023). Our sweep implements analogues of these signals, not the original systems. Model-judge best-of-N code selection is the closest setting, but VERDICT freezes the pool to measure the judge's loss on fixed candidates. ReAct-style agents feed execution results to the model (Yao et al., 2023); VERDICT removes candidates by execution before a model reads them.

RTL generation. VerilogCoder is an agentic workflow, while RTLCoder and OriGen improve models or training data (Ho et al., 2025; Liu et al., 2024a; Cui et al., 2024). VERDICT's default cross-verifier extends ChipMATE's independent Verilog/Python comparison (Yu et al., 2026a).

8 Limitations and conclusion

Limitations. We establish a controlled gain for one judge, not what makes a judge effective. In our run, `gpt-5` showed no gain, and on CVDP-CID003 the Claude judge is one problem below the cross-verifier. Shared errors, specification comprehension and candidate confusability remain untested hypotheses. Most frontier arms, not CVDP-CID003's, used the single-correct premise, whose pre-registered confirmation failed its primary test. The tools' accuracy gain is confined to the waveform specifications used in development; the RTLLM-v2 tool fix was made after seeing that benchmark. Curation exceptions, the pass-banner scoring bound, unmatched supporting baselines, and closed-model alias drift limit reproduction and comparison. The labeled-slice analysis measures sampling uncertainty within benchmarks, not out of sample, and repeated generations test seed sensitivity, not transfer to new specifications (Appendix S). Earlier-draft numbers are superseded (Appendix B): the unmatched one-shot's 89.7%, a seed-1 +2.6pp cross-verifier gain, a stratified 94.2% frontier-pool estimate, and a routed 88.5% at 0.68× one-shot's price (retained sweep: 87.8%).

Conclusion. On VerilogEval-v2 the cross-verifier leaves more correct designs unselected than the pool fails to generate. The Claude judge recovers the largest measured share of these, execution tools a smaller one, yet neither gives cheap Verilog pools a demonstrated advantage over frontier generation on accuracy or cost. The lesson is a measurement discipline: report o, r, and what recovery costs.

AI USE STATEMENT

Large language models are the *object of study* in this work: the evaluated harness orchestrates LLMs as generators, reference implementers, and verifiers, and every such use is part of the method itself, fully specified in §4 and Appendix B. Separately, we used generative AI tools as research tooling, including uses that require disclosure. LLM-based coding agents implemented and debugged much of the harness, evaluation, and analysis code under the authors' direction; LLM agents proposed, ran, and summarized diagnostic and ablation experiments that the authors specified or approved, and gave first-pass interpretations of run records; and LLM-based mock reviews critiqued the methodology. We also used LLMs to draft and edit parts of the manuscript, to write figure-generation scripts, to produce the artwork of Figures 1 and 6 with an image-generation model (then checked and edited by the authors), to search for related work, and to check and format references. We did not use generative AI to create datasets or benchmark data. The authors set the research questions, approved every experiment, and made the final decisions on design and claims. We reviewed all AI-assisted work: reported numbers are recomputed by deterministic scripts from stored per-candidate records against the official benchmark testbenches, with the exceptions identified in Appendix B, and every reference was checked against its primary source. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Stored cheap-pool records support candidate-level reconstruction of the reported numbers; Claude-judge selection on VerilogEval-v2 seeds 2–3 and ChipBench-SC seed 1 survives only as aggregates (for VerilogEval-v2, plus run logs of fix/break counts), and earlier frontier baselines lack retained generations. Appendix B identifies these limits and gives the experimental recipe: the generation-pool API names and settings, the frontier-judge protocol, the simulator and commands, benchmark problem counts, the paper-fair protocol (the official testbench is never read during generation or selection; three RTLLM-v2 spec repairs drew on it or the reference), the 3-seed protocol with per-seed counts, and a corrections log. Appendix G lists the twelve testbench-free selectors of the controlled selection study, Appendix E documents the verification-harness bugs and the completeness instrumentation (Appendix F), and Appendix C lists the benchmark file repairs; every arm ran on the repaired files, and the three spec repairs are disclosed as curation exceptions (§2). All benchmarks are public; our harness, prompts, and re-scoring scripts will be released, and an anonymized version is available to reviewers as supplementary material.

REFERENCES

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. CodeT: Code generation with generated tests. In *International Conference on Learning Representations (ICLR)*, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Fan Cui, Chenyang Yin, Kexing Zhou, Youwei Xiao, Guangyu Sun, Qiang Xu, Qipeng Guo, Yun Liang, Xingcheng Zhang, Demin Song, and Dahua Lin. OriGen: Enhancing RTL code generation with code-to-code augmentation and self-reflection. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–9, 2024.

Chia-Tung Ho, Haoxing Ren, and Brucek Khailany. VerilogCoder: Autonomous Verilog coding agents with graph-based planning and abstract syntax tree (AST)-based waveform tracing tool. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 300–307, 2025.

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 (ICLR)*, 2025.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, et al. Competition-level code generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023a.

Mingjie Liu, Nathaniel Pinckney, Brucek Khailany, and Haoxing Ren. VerilogEval: Evaluating large language models for Verilog code generation. In *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pp. 1–8, 2023b. doi: 10.1109/ICCAD57390.2023.10323812.

Shang Liu, Wenji Fang, Yao Lu, Qijun Zhang, Hongce Zhang, and Zhiyao Xie. RTLCoder: Outperforming GPT-3.5 in design RTL generation with our open-source dataset and lightweight solution. In *2024 IEEE LLM Aided Design Workshop (LAD)*, pp. 1–5, 2024a.

Shang Liu, Yao Lu, Wenji Fang, Mengming Li, and Zhiyao Xie. OpenLLM-RTL: Open dataset and benchmark for LLM-aided design RTL generation. In *2024 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2024b.

Yao Lu, Shang Liu, Qijun Zhang, and Zhiyao Xie. RTLLM: An open-source benchmark for design RTL generation with large language model. In *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 722–727, 2024. doi: 10.1109/ASP-DAC58780.2024.10473904.

Ansong Ni, Srini Iyer, Dragomir Radev, Veselin Stoyanov, Wen-tau Yih, Sida I. Wang, and Xi Victoria Lin. LEVER: Learning to verify language-to-code generation with execution. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Nathaniel Pinckney, Christopher Batten, Mingjie Liu, Haoxing Ren, and Brucek Khailany. Revisiting VerilogEval: A year of improvements in large-language models for hardware code generation. *ACM Transactions on Design Automation of Electronic Systems*, 30(6):1–20, 2025a.

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*, 2025b.

Tal Ridnik, Dedy Kredo, and Itamar Friedman. Code generation with AlphaCodium: From prompt engineering to flow engineering. *arXiv preprint arXiv:2401.08500*, 2024.

Freda Shi, Daniel Fried, Marjan Ghazvininejad, Luke Zettlemoyer, and Sida I. Wang. Natural language to code translation with execution. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations (ICLR)*, 2025.

Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *International Conference on Learning Representations (ICLR)*, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Zhongkai Yu, Yichen Lin, Chenyang Zhou, Yuwei Zhang, Kun Zhou, Junxia Cui, Haotian Ye, Zhengding Hu, Zaifeng Pan, Ruiyi Wang, Yujie Zhao, Hejia Zhang, Jingbo Shang, Jishen Zhao, and Yufei Ding. ChipMATE: Multi-agent training via reinforcement learning for enhanced RTL generation. *arXiv preprint arXiv:2605.12857*, 2026a.

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*, 2026b.

Tianyi Zhang, Tao Yu, Tatsunori B. Hashimoto, Mike Lewis, Wen-tau Yih, Daniel Fried, and Sida I. Wang. Coder reviewer reranking for code generation. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, volume 36, pp. 46595–46623, 2023.

A CLASSIFIER KEYWORDS (TWO-AXIS TAXONOMY)

The classifier (Python regex, case-insensitive, spec text only) assigns two orthogonal labels, `<temporal>-<interface>`, each mechanically decidable and each mapping to one concrete bug class.

Temporal axis ($\Rightarrow$ clocking bugs). SEQ if any of `clk`, `clock`, `posedge`, `negedge`, `reset`, `rst`, `register`, `state`, `cycle`, `edge`, `flip-flop`, `latch`, `previous`, `over time`, `each/per/next clock` matches; else COMB.

Interface axis ($\Rightarrow$ timing-contract / addressing bugs), most-specific first.

- ADDRESSED: `addr(ess)?`, `memory`, `ram`, `rom`, `fifo`, `register file`, `lifo`, `stack`, `queue`, `cache`, `mem`, `look-?up table`, `lut`, `read/write`.
- HANDSHAKE: `valid`, `ready`, `ack`, `enable`, `en`, `start`, `done`, `busy`, `handshake`, `serial`, `uart`, `spi`, `i2c`, `req`, `grant`, `stream`, `back-?pressure`, `stall`.
- DIRECT: default — inputs map to outputs each cycle with no flow control.

This 2×3 grid (e.g. SEQ-HANDSHAKE = a ready/valid FSM, SEQ-DIRECT = a plain counter/FSM) makes each axis orthogonal and each cell's specialized prompt a targeted checklist against that axis's bug class. Word boundaries are anchored (`\b`) so substrings like `en` inside `enable` or `alu` inside `value` do not mis-route.

B REPRODUCIBILITY

Generation pool (frontier-free). The pool samples `gpt-5-mini` (OpenAI API; `max_completion_tokens` ≥ 8000 so reasoning does not starve the answer, Appendix E, bug 5), `deepseek-chat` and `deepseek-coder` (DeepSeek API), each at $K{=}3$ ($K{=}1$ on CVDP-CID003, three candidates per problem). Python references for the cross-verifier are written by `gpt-5-mini` and `deepseek-chat`, one sample each per problem, so a candidate is compared with the one reference of the other *family*, where a family is the model provider (OpenAI or DeepSeek): a `deepseek-coder` candidate, like a `deepseek-chat` one, is compared only with `gpt-5-mini`'s references. The VerilogEval-v2 seed-1 pool is the exception: when bug fix 1 (Appendix E) regenerated its references we drew two samples per family, giving four references on 153 problems and three on 3, so there every candidate is compared with two cross-family references and the full-match count ranges over 0–2 (it reaches 2 on 117 problems); RTLLM-v2 seed 1 was regenerated with one sample per family. Extraction failures leave a single reference on $3/2$ VerilogEval-v2 problems (seeds 2/3) and on $2/1/0$ RTLLM-v2 problems (seeds 1/2/3). Every seed-1 VerilogEval-v2 cross-verifier number in this paper (87.2, the per-class table, the seed-1 columns of Table 9, and experiments (i)–(iv) and (xi)–(xii) of Appendix O) was selected with the four-reference pool. Restricting seed 1 to the first reference of each family returns $135/156$ (86.5%) instead of $136/156$, and an independent one-per-family regeneration of the same pool returns $134/156$ (85.9%); the three-seed cross-verifier mean would read $85.0{\pm}1.3$ or $84.8{\pm}1.0$ instead of $85.3{\pm}1.7$, and its gain over the floor pick 0.0 or -0.2pp instead of $+0.2$pp. We report the four-reference run because every other seed-1 arm (Claude judge, execution tools, per-class table) was measured on that stored pool. **Three API names, two families, probably two models.** The pool's three generation slots are API model names, which need not be three distinct models. DeepSeek's change log records on 2024-09-05 that "the DeepSeek V2 Chat and DeepSeek Coder V2 models have been merged and upgraded into the new model, DeepSeek V2.5. For backward compatibility, API users can access the new model through either `deepseek-coder` or `deepseek-chat`"; no later entry mentions `deepseek-coder`. Its entry of 2026-04-24 states that `deepseek-chat` would be discontinued on 2026-07-24 and until then pointed to the non-thinking mode of `deepseek-v4-flash`. Every cheap pool in this paper was generated before that date (record timestamps between June and 16 July 2026); the diagnostic pools of Appendix Q also include DeepSeek one-shot draws made in September 2026. We found no documentation of what `deepseek-coder` resolved to in that period, and the pool records do not keep the `model` field the API returns, so the resolution cannot be confirmed directly. The harness sends both names through the same DeepSeek call with the identical prompt and sampling settings (temperature 0.4 in the earliest committed code, dated 8 June 2026; `gpt-5-mini` runs at the API's default temperature; the LiveCodeBench code also treats the two

names identically), so only the model string differs. The stored samples are consistent with one model behind both names: over the six VerilogEval-v2 and RTLLM-v2 pools (612 problem-seeds), 25.4% of same-problem pairs of `deepseek-chat` samples, 24.3% of pairs of `deepseek-coder` samples and 25.4% of cross-name pairs are byte-identical, against 3.7% of `gpt-5-mini`–DeepSeek pairs (LiveCodeBench pool: 1.8, 1.2 and 1.4%, against none of 1,998), and the two names' pass@1 is 72.6 vs. 72.7% on VerilogEval-v2 and 62.7 vs. 63.7% on RTLLM-v2 (single-seed ChipBench-SC and CVDP-CID003: 26.7 vs. 23.3 and 34.6 vs. 38.5%; Table 5). We therefore describe the pool as three sampled API names from two model families, probably two models; on that reading six of the nine samples per problem come from one DeepSeek model. "Member" and "per model" in this paper mean one API name, and a *family* is the model provider, the label the cross-verifier's reference exclusion uses. This changes how the pool is described, not any measured number; if the two names are one model, the default pool is less diverse than three models would make it, and the four-family pool of Appendix O(v) is our configuration with more distinct models. In September 2026 every call we logged under `deepseek-chat` (2,157 judge and one-shot calls) returned the model name `deepseek-flash`, which DeepSeek's change log of 2026-09-10 assigns to V4.1-Flash, although the retirement of `deepseek-chat` had been announced. **Frontier judge.** Claude-Opus-4.8 via CLI, used *only* as a selector (reads spec + de-duplicated candidates, returns an index), 3-vote majority over three independent calls per problem; it never generates Verilog. **Simulation.** `iverilog -g2012` + `vvp`. The cross-verifier simulates each candidate and each reference on 40 random input vectors (ChipMATE's generator, fixed seed; the harness drives clock and reset). A candidate *matches* a reference when every output agrees on every vector; its mean match rate averages, over the cross-family references, the fraction of agreeing (vector, output) cells. Selection orders candidates by (full matches, V–V agreements) and breaks ties by mean match rate with a 0.15 margin, starting from the floor pick. Every reported cross-verifier number is this selection re-run offline over the stored pool (`offline_select.softmr_select`); the in-harness routine `xverify_select` orders V–V ahead of the reference count when V–V is enabled, and its live picks are not used anywhere in this paper. The 30–90-vector per-class budgets and the class-matched stimulus belong only to the classification arms of Appendix G, whose variants use the in-harness ordering plus its quorum gate and compile-filter fallback. **Prompt parity between the pool and the frontier baseline** (Appendix M): on VerilogEval-v2 the baseline receives the byte-identical prompt the pool receives, including the explicit module-interface block (prompt correction recorded below); the RTLLM-v2, ChipBench-SC and CVDP-CID003 baselines keep the unmatched prompt (Appendix M). **Official scoring (one environment for every arm, including the baselines; one residual asymmetry, noted below).** Five RTLLM-v2 testbenches load reference vectors by *relative* path (e.g. `$readmemh("reference.dat")`), so the scorer stages each problem's data files beside the compiled object and runs the simulator there; otherwise the reference array stays X, every comparison mismatches, and the testbench reports a failure even for the *official golden* design. Three problem directories also disagree with the module name their own testbench instantiates (e.g. `fixed_point_substractor` vs. `fixed_point_subtractor`), so on an elaboration error naming an unknown module the scorer retries once under that name — a naming fix, never a functional one. With both in place $47/48$ official golden designs pass their own testbench (the exception, `asyn_fifo`, ships a helper module ahead of the design under test). Scoring both arms in one environment matters: the frontier baseline is scored inside each problem directory (`score_in_dir`) while the harness previously was not, which silently handed the baseline three designs. The baseline's generations were not retained, so it cannot be re-scored; instead we established that the two scorers agree, on 750 pool candidates spanning all three seeds, with *zero* disagreements (against $19/430$ for the unfixed scorer, concentrated on exactly the five data-file problems). The fix is not uniformly favorable to us: `multi_booth_8bit`'s testbench reads its vectors with `$fopen`, so with the file absent its checking loop never executed and every candidate passed vacuously — staging the file makes that problem *stricter*. One residual asymmetry remains: after an elaboration error our scorer retries under the module name the simulator reports, which `score_in_dir` does not, so a multi-module candidate can be rescued in our arms that would not be in the baseline. **Benchmarks.** VerilogEval-v2 (156), RTLLM-v2 (48 after excluding 2 file-broken designs), ChipBench-SC (30), CVDP-CID003 (78, cocotb), and — for the cross-domain experiment — HumanEval (Chen et al., 2021) (164, Python), where cross-verification is *execution*-based: candidates run on model-generated inputs and the selector scores output-consensus agreement (no official test read until final scoring). **Tables 1 and 5: estimators.** Member pass@1 is, per problem, the fraction of that API name's *retained* samples that pass, averaged over problems. A draw whose response contained no extractable module was never added to the pool, so it is absent rather than recorded as a failure, and a problem with no retained

sample scores 0: `gpt-5-mini` is missing $4/4/6$ of 468 draws per VerilogEval-v2 seed and $2/3/1$ of 144 per RTLLM-v2 seed, the other members at most one. Counting every missing draw as a failure instead gives 83.8 ± 0.4 and 63.7 ± 1.7 for `gpt-5-mini` (end-to-end gains $+7.0$ and $+12.7$pp) and leaves every other printed row unchanged. Pool pass@k is the unbiased estimator (Chen et al., 2021) $1-\binom{n-c}{k}/\binom{n}{k}$ over the n retained candidates ($7 \le n \le 9$), averaged over problems, and pass@n is the oracle. VerilogEval-v2/RTLLM-v2 rows are 3-seed means$\pm$sd; ChipBench-SC/CVDP-CID003 rows are seed-1 except the Claude-judge row (3- and 2-seed respectively), and measured ChipBench-SC rows use the X-tolerant init convention (Appendix D). **Paper-fair protocol.** The official testbench, golden RTL, and testbench-specific signals are never read during generation or selection; the testbench is run once to score the single chosen design. The exceptions are three: the VerilogEval-v2 module header (port names, directions and widths), which generation parses from the reference file and which each VerilogEval-v2 specification also states in prose, given to the pool and the matched baseline alike (Appendix M); the one-time development calibration of spec-waveform replay on the golden designs of the 12 VerilogEval-v2 waveform problems (§2); and benchmark curation, where three RTLLM-v2 spec repairs drew on the testbench or the reference (Appendix C). **Variance.** Table 1 numbers are 3-seed means$\pm$sd over independent generation seeds: VerilogEval-v2 cross-verifier 85.3 ± 1.7, Claude judge 90.8 ± 1.0 (per-seed $91.7/91.0/89.7$), oracle 93.2 ± 1.0; RTLLM-v2 cross-verifier 72.2 ± 2.4, Claude judge 76.4 ± 3.2, oracle 81.2 ± 2.1 (all RTLLM numbers under the benchmark's "Your Design Passed" banner convention; on `ring_counter` the banner survives a mismatch, bounded at ≤ 2.1pp per arm, Appendix C). **Every mean and every difference is computed from problem counts and rounded last** (VEv2 counts over the three seeds: oracle $147/145/144$, Claude judge $143/142/140$, cross-verifier $136/131/132$, floor pick $132/133/133$, best member pass@3 $139/141/139$ of 156), so a difference can sit 0.1pp from the difference of two rounded means: the selector substitution is $(425-399)/468 = 5.56 \to 5.6$pp although $90.8-85.3=5.5$; the VerilogEval-v2 margin is $(436-427)/468 = 1.9$pp although $93.2-91.2=2.0$. **Derived failure counts.** Across the three VerilogEval-v2 seeds, no correct design is present on $32/468 = 6.8\%$ of problem-seeds; a correct design is present but not selected on $37/468 = 7.9\%$ with the cross-verifier and $11/468 = 2.4\%$ with Claude. For Claude's own K=3 pool, these counts are $10/156 = 6.4\%$ absent and $1/156 = 0.6\%$ present but unselected. **Retained Claude-judge verdicts.** Per-problem Claude-judge verdicts are not retained for VerilogEval-v2 seeds 2–3 or ChipBench-SC seed 1; those cells survive only as aggregates and, for VerilogEval-v2, each run's log, which counts the cross-verifier picks the judge fixed and broke and names the broken problems. Table 8 and the exact McNemar tests of §5.1 use only cells with per-problem records, and the recorded-verdict bounds of §5.2 are tight to one problem only on VerilogEval-v2 seed 1. The seed-2 and seed-3 logs record 12 fixes and 0 breaks, and 10 fixes and 2 breaks; taken at face value, exact McNemar would give $p < 0.001$ and $p = 0.04$. We do not count these as within-run tests: the fixes are not itemized, and each log counts against its own recomputation of the cross-verifier, which on seed 2 returned 130 correct designs rather than the 131 reported above (seed 3: 132 in both). We therefore claim neither significance nor its absence within VerilogEval-v2 seeds 2–3; on VerilogEval-v2 the claim is significance across seeds ($p = 0.019$, from the reported counts). **CVDP-CID003 Claude-judge runs: pools, fallbacks, prompt.** The CVDP-CID003 Claude-judge row (42.3 ± 1.8, per-seed $41.0/43.6$) comes from two judge runs over two independently generated pools, one per generation seed, not over one pool: no problem has the same candidate set in both. The two pools' counts coincide — each holds a correct design on 38 of 78 problems (oracle 48.7%), and on each the cross-verifier selects a correct design on 34 (43.6%) — but the oracle label differs on 6 problems and the cross-verifier's outcome on 8. The other CVDP-CID003 rows of Table 5 print seed 1. *Seed-1 fallbacks.* In the original seed-1 run none of the three votes returned a parsable answer on 11 of the 78 problems, contiguous in pool order (consistent with a transient service failure; the cause was not recorded), and the driver returned the first de-duplicated candidate, which on all 11 was also the cross-verifier's pick (2 of the 11 correct). We later re-judged exactly these 11 with the same prompt and pool (the client may differ from the original run's): each received a pick, from three parsed votes on 4, two on 2 and one on 5, and no pick's correctness changed (2 of 11 correct again). Seed 1 therefore scores $32/78 = 41.0\%$ under either record; the printed values use the re-judged record, and seed 1's fixes and breaks against the cross-verifier ($+1/-3$) all fall on the 67 problems the original run judged. Seed 2 had no fallback, but 18 of its 78 picks rest on fewer than three parsed votes (one on 10, two on 8); vote texts were not retained, so truncated votes cannot be told from failed ones. On seed 2 the Claude judge fixes one cross-verifier pick and breaks one. *Prompt.* The CVDP-CID003 judge ran from a separate driver whose prompt does not carry the single-correct premise: it states "Exactly one (or more) correctly implements the spec; the others contain bugs (wrong reset/init, off-by-one, wrong FSM/handshake,

wrong widths, timing)". It still asserts that some candidate is correct, which is false on the 40 of 78 problems per seed whose pool holds no correct design. It also differs from the default selector prompt (Appendix Q) elsewhere: it names no module, lists partly different bug types, and omits the instruction to attend to clocking, reset, initial values, and edge cases. Its driver requested a 4,096-token cap per vote (16,000 for the re-judging); which cap, if any, took effect depends on the client path, which was not recorded. Two or more text-distinct candidates are correct on 29 (seed 1) and 27 (seed 2) of the 38 problems with a correct design, so the single-correct premise would have been false on most problems the judge could get right. The CVDP-CID003 result is therefore not a same-prompt replication of the VerilogEval-v2/RTLLM-v2 gain, and the cross-benchmark capture comparison (§6, Appendix K) mixes prompts at CVDP-CID003. **RTLLM per-seed counts.** On 48 problems per seed, the cross-verifier returns 36/34/34 correct designs and the Claude judge 38/35/37. The gains are +2/+1/+3 problems, or +4.2/+2.1/+6.3pp, positive on every seed. Per-problem granularity is 0.64pp (VEv2, 156 problems) / 2.1pp (RTLLM, 48). On ChipBench/CVDP the Claude-judge row is 3-/2-seed (Table 5 notes) and the other rows are seed-1 (granularity 3.3pp / 1.3pp; the ChipBench-SC oracle exceeds Claude by one problem, and the CVDP-CID003 Claude-judge mean sits one problem below the cross-verifier).

Corrections log (superseded reports). Earlier drafts averaged rounded seed percentages and printed 93.1, 7.8, 2.3, and 5.5 where the corrected count-based summaries are 93.2, 7.9, 2.4, and 5.6. A seed-1 per-class table incorrectly used the three-seed cross-verifier mean 85.3 in place of 87.2, obscuring the specialization loss. An earlier draft reported a clean +2.6pp cross-verifier gain over the floor pick on VerilogEval-v2 seed 1 (136 against 132 of 156); over three seeds the gain is +0.2±2.1pp on VerilogEval-v2 and +4.2±3.6pp on RTLLM-v2, so the seed-1 lead is sampling variation across generation seeds, not a selector effect. The June live log for VerilogEval-v2 seed 2 printed a cross-verifier count of 130; canonical rescoring of the retained pool gives 131, which every table uses (the per-problem record of the June run was not retained, so the one differing problem cannot be localized). The old frontier one-shot omitted the interface block and scored 89.7; matching the prompt gives 91.2, changing the harness comparison from nominal +1.1 to −0.4pp, not significant in either direction. The routing point was first priced at 0.68× one-shot: generation priced from returned code alone was $0.0048 and omitted hidden reasoning. Revised estimates are $0.0135 for candidates, $0.0174 including references, and $0.028 for frontier one-shot rather than $0.024; the routed point costs at least as much as one-shot. A withdrawn routing curve reported 5%→87.8, 15%→91.0, 25%→92.3 by treating every escalated oracle-solvable problem as correct, rather than using the selector's recorded outcomes. Experiment iv now uses the latter. The routed point's 88.5 at 9% escalation was a June quorum gate whose escalation set cannot be reconstructed from the retained pool (its stored cross-verification metadata is stale); the retained evidence-ordered sweep scores 87.8 at that fraction, so the routed point is 87.8 everywhere in this paper, and the alternative-gate comparison is withdrawn for lack of a record. A September re-score of RTLLM-v2 ran the simulator outside each problem directory, so testbenches that load their reference vectors by relative path failed for our arms but not for the baseline, which was scored in the directory; it was retracted, and every RTLLM-v2 number here comes from the staged scorer described above. The original labeled-slice analysis loaded that retracted re-score and produced negative selection loss; the reconciliation check now excludes that artifact. Its old unconditional agreement metric counted abstention as correct only on negative cells. That metric and its 94.3–100% claim are withdrawn; the old near-tie agreement values (45% at m=25, 60.5% at m=80) are superseded by symmetric commit-rate and conditional-correctness reporting in Table 8. The frontier K=3 pool arm (Appendix N) was first reported as a 17-problem stratified estimate: across five stored one-shot draws 131/156 problems passed in every draw and 8/156 failed in every draw, so we regenerated only the 17 that decide, scored them, and let the other 139 contribute deterministically. That gave 94.2%, and as the positive cell of the labeled-slice analysis those 139 deterministic contributions inflated conditional correctness to 100%. The full run reports 92.9%: the stratification was optimistic by 1.3pp, because re-drawing a pool does not reproduce the always-pass set exactly. The number in this paper is the full-scale one. The frontier-pool loss is 0.6pp after rounding 1/156, replacing inconsistent 0.7pp prose. A pass@9 calculation applied where the pool had fewer than nine candidates returned the impossible 94.7%; the coverage curve now stops at the smallest pool size, k=7. An 8k output cap starved `gpt-5`'s reasoning as a judge and printed 84.6 for its plain-prompt run; at a 24k budget it scores 87.8 (experiment (ix), Appendix O). These corrections change claims and reporting; the current tables and their stated protocols are the evidence used in the main text.

C BENCHMARK FILE REPAIRS

We patched seven published benchmark files (originals retained as `.orig_broken`); an eighth problem is handled in the harness, not the file. Every generation run we report post-dates the patches; the pool and the harness read the patched specs, and testbench patches apply to every arm at scoring. Every comparison is therefore on the *curated* benchmark: because selectors pick different candidates, a patch can still change the difference between two arms. Three spec patches (`sequence_detector`, `freq_divbyeven`, `barrel_shifter`) drew on the testbench or the reference design; they are the one exception to the rule that neither informs what generation or selection reads (testbench patches act only at scoring).

- RTLLM-v2 `LFSR/testbench.v`: a positional bind `(out_tb, clk_tb, rst_tb)`, which matches the reference's port order `(out, clk, rst)` but not the order in which the spec lists the ports; rewritten to a named bind.
- RTLLM-v2 `sequence_detector`: the spec used `reset_n` but the testbench and reference use `rst_n`; renamed in the spec.
- RTLLM-v2 `fixed_point_substractor` (no file changed): the directory name differs from the module name in the spec and reference; the harness reads the module name from the spec's `Module name:` field.
- VerilogEval-v2 `Prob099_m2014_q6c_test.sv`: the testbench binds `.Y2/.Y4`, while the reference and the spec's interface block use `Y1/Y3` (the spec's prose says Y2/Y4); renamed in the testbench to match the reference.
- RTLLM-v2 `freq_divbyeven`: the spec's `Module name:` field read `freq_diveven`; corrected to the name the directory and testbench use. The same patch added "Default value of NUM_DIV is 6", a behavioral detail absent from the published spec and taken from the reference design.
- RTLLM-v2 `asyn_fifo`: `break` inside a `repeat`, which `iverilog` rejects, rewritten as an equivalent flagged loop.
- RTLLM-v2 `ring_counter`: a SystemVerilog array literal rewritten as element assignments, and the stop condition changed from `i==9` to `i==10`. Under `iverilog` the original condition is never observed (the two blocks clocked on the same edge race), so no design, the reference included, could pass; the change makes the pass banner reachable after all ten checks. This testbench also prints its banner after a mismatch, and our scorer, following the benchmark's convention, accepts the banner. An audit of all 1,099 distinct RTLLM-v2 pool candidates found this is the only testbench that passes a candidate that printed a failure: 9 of the 12 it passes use a synchronous reset where the reference resets asynchronously, and mismatch at the first check only. Failing any mismatch would leave the RTLLM-v2 oracle unchanged, cost the cross-verifier one problem on every seed (it picks one of the nine), and cost the Claude-judge arm and the Claude one-shot baseline at most one each (their stored records keep only verdicts): at most 2.1pp per arm.
- RTLLM-v2 `barrel_shifter`: the spec says the design "shifts or rotates"; we rewrote it to a logical right shift *after reading the testbench*. Like the `freq_divbyeven` default, this adds semantic content from outside the spec, here from T. The three curation exceptions touch three problems (at most 6.3pp of RTLLM-v2); the Claude one-shot baselines pass all three.

D INITIALIZATION CONVENTIONS MOVE THE SCORED ORACLE BY ABOUT 1PP

Initialization conventions affect the scored pool oracle on VerilogEval-v2: the reference initializes registers (`initial q=0`) while the cheap models leave a functionally-correct register uninitialized, mismatching only at $t{=}0$. These labels do not by themselves distinguish specification understanding from familiarity with benchmark conventions. A paper-fair register-initialization transform (never reading T) lifts the VerilogEval-v2 oracle $+1.1$pp on average ($93.2 \rightarrow 94.2$). This measures sensitivity to initialization conventions; it does not isolate the source of the frontier model's advantage.

ChipBench-SC scoring. The same reset-boundary convention dominates on ChipBench-SC, whose self-checking testbenches we therefore score *X-tolerantly*: a candidate output that is X at the reset boundary — an uninitialized but otherwise functionally-correct register — is treated as don't-care

rather than a mismatch, the upstream ChipBench convention. Under this scoring all 30 reference designs pass except `Prob013`, whose golden RTL fails its own testbench — a benchmark item on which the golden design itself fails; we retain it in the 30-problem denominator, where it scores as a miss for every method alike, so it cannot affect any delta. Handling the convention at the testbench level — rather than via the candidate-side transform above, which is net-negative (-3) on this benchmark — lifts the cheap pool's oracle $33.3 \rightarrow 40.0$ and the cross-verifier $23.3 \rightarrow 26.7$; the ChipBench-SC numbers (Table 5) are under this scoring.

E THE SIX SILENT VERIFICATION-HARNESS BUGS

The audit summarized in §2 found six silent failures in the verification path, each returning a benign value rather than an error, so cross-verification scored zero agreement and fell back to the floor.

1. Python references crashed with `KeyError: 'clk'` (38–65% of refs) because the prompt never pinned the once-per-cycle, no-`clk`-key calling convention ⇒ `match_rate`$= 0$ silently.
2. Reference output keys were width-suffixed (`out[3:0]`) or bit-blasted; the bare-name lookup returned `None` on every cycle ⇒ silent dead reference (10–17% of refs even after fix 1).
3. Active-low resets named with a bare trailing `n` (`resetn`, `aresetn`) were classified active-high ⇒ the DUT was held in reset all test, and *wrong* candidates falsely matched at rate 1.0.
4. Clocks were detected only by the exact name `clk`; `clock`, `CLK`, `wclk` went undetected ⇒ sequential DUTs were never clocked, or the testbench silently failed to compile.
5. The reasoning models' token budget was frozen at import to 3k; chain-of-thought starved the answer ⇒ ∼44% of the floor model's samples were silently empty and dropped on hard problems.
6. Two ablation-harness bugs (absolute-index sample subsetting; an anchored control-signal regex) mismeasured per-class effects.

Fixes 2–4 changed only how candidates were verified and were applied retroactively to the saved pool; fix 1 regenerated the Python references with the calling convention pinned (two samples per family for the VerilogEval-v2 seed-1 pool, one per family for RTLLM-v2 seed 1; the seed-2/3 pools were generated after the fix with one per family, Appendix B), and fix 5 regenerated the reasoning model's candidates with an adequate budget, so every pool in this paper is the regenerated one. Together they restored the cross-verifier to function (crash rate $< 1\%$). A seventh silent failure surfaced later, on RTLLM-v2: the simulator renamed the *first* module in a file, so a candidate that defines sub-modules before its top module was verified as its sub-module (Appendix P); it changes which candidate the cross-verifier picks on eight problem-seeds and none of the picks' correctness.

F VERIFIER COMPLETENESS INSTRUMENTATION

Every reported run logs two completeness signals that make silent verifier failures visible. (i) **Candidate count**: valid Verilog candidates surviving extraction per problem; with $K{=}3$ across three API names the target is nine, and a shortfall flags silent generation drops (the token-starvation bug produced $n{=}6$ on hard problems as all three floor samples returned empty). (ii) **Reference crash rate**: the fraction of Python references that raise or return all-`None`. Before the fixes this was 38–65% on clocked problems (KeyError on `clk`; width-suffixed output keys); after, $< 1\%$. We treat any run with high crash rate or incomplete candidate counts as untrustworthy and re-run it. These signals were essential both ways: the floor-bound cross-verifier (Appendix E) was diagnosable as a crashing reference, not a true null, only because crash rate was instrumented — and, conversely, the post-fix cross-verifier's ≈ 0 contribution over the floor pick could be measured with those failures removed; this audit does not exclude every residual implementation issue.

G THE TWELVE TESTBENCH-FREE SELECTORS

Reporting details. Every testbench-free selector we ran, on the identical frozen pool (VerilogEval-v2, selected accuracy r, 3-seed mean$\pm$sd over the same three generation seeds as Table 1). The band is not an assertion: **twelve selectors and the floor pick span** 83.3**–**85.7, a 2.4pp spread ($(401-390)/468$), against an oracle of 93.2 and the Claude judge at 90.8. The rows marked * are

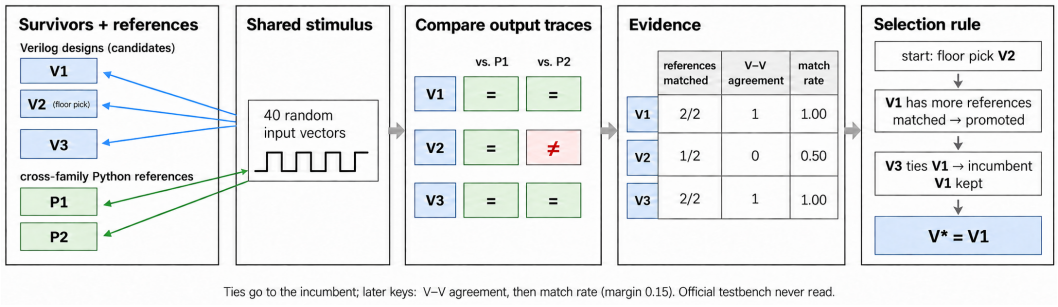


Figure 6: **The cross-verifier ranks execution agreement, without an official correctness signal.** Illustrative example. Every candidate (with execution tools, every survivor) and reference runs on the same 40 random input vectors; each candidate is compared with the cross-family references, and, starting from the floor pick, the incumbent is replaced only on strictly more references matched, then strictly more V–V agreement, then a match-rate margin above 0.15. This is one implementation of selector slot C1. The Claude judge is a separate implementation that reads the specification and candidates; it does not inherit these execution scores.

Table 2: **Twelve pool-internal selectors and the floor pick span a narrow band.** VerilogEval-v2 selected accuracy, three-seed mean±sd (%); the Claude judge and the oracle provide reference points. Reporting details follow.

Testbench-free selector (over the fixed cheap pool)	signal	VerilogEval-v2
floor pick (strongest single model, no cross-verification)	—	85.0±0.4
MBR-exec (Shi et al., 2022) (pure argmax match-rate)*	V–P	83.3±1.1
self-consistency (Wang et al., 2023) (output majority)*	V–V	84.0±0.6
V–V consensus count	V–V	84.2±1.3
P-consensus (multi-sample references)†	V–P	84.2±1.0
mean match-rate, margin 0.15	V–P	84.8±2.1
mean match-rate, margin 0.05	both	85.3±1.7
mean match-rate, margin 0.10	both	85.3±1.7
argmax match-rate + V–V	both	85.3±1.7
CodeT-style (Chen et al., 2023) cand.–ref. (our cross-verifier)*	both	**85.3±1.7**
+ per-class-K subsetting	V–P	85.5±1.0
+ per-class stimulus	V–P + class stim.	85.7±0.7
+ both classify arms (cross-verifier+classify)	both + class arms	85.3±0.6
Claude judge (**ours**)	spec	**90.8±1.0**
oracle (pass@any, ceiling)	—	93.2±1.0

analogues of established selection methods, evaluated on our stored pools. †Requires two samples per family; only the seed-1 pool has them, so on seeds 2–3 this row reduces to the single-reference full-match rule (Appendix B). The "signal" column names the execution evidence each row ranks on: V–P is agreement with the cross-family Python references, V–V is agreement among candidates. The three classification rows use the in-harness ordering with its quorum gate and compile-filter fallback on 30–90 per-class vectors; the remaining rows re-rank the stored pool offline on 40 vectors (Appendix B). Figure 6 sketches the ranking.

Every row is paper-fair (the testbench is never read) and every row is computed on the same three frozen pools, so the spread is a property of the selectors and not of the generation. Two further testbench-free modalities sit outside the table because they use a different signal and both land lower: model-generated self-checking testbenches select at 80.1 (experiment (vi)) and a *learned* pool-internal verifier at 83.3 (experiment (iii)). Enabling candidate synthesis nets −1 problem (experiment (xii)). The tested agreement-based selectors remain in the 83–86 band. This pattern is consistent with shared errors, but does not test that hypothesis or bound other selectors (Appendix K). The model-testbench selector recovered four problems on one seed but an *entirely different* set on a re-run (net zero, high

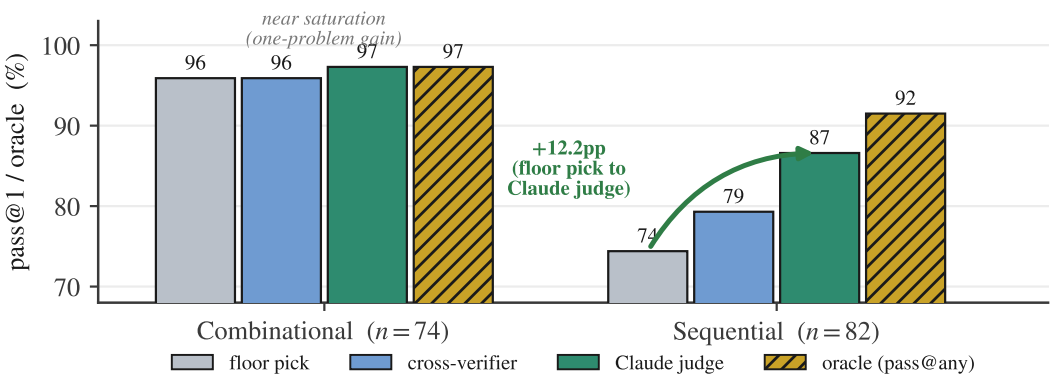


Figure 7: **Sequential designs hold the larger gain.** VerilogEval-v2 seed 1: floor-to-Claude accuracy rises $74.4 \rightarrow 86.6\%$ on sequential problems and $95.9 \rightarrow 97.3\%$ on combinational problems. Classification locates this difference; its class-specific selection variants lower accuracy on this seed (Table 3).

Table 3: **The larger gain lies in sequential designs.** VerilogEval-v2 seed 1 (%): Claude adds one combinational problem; the class-specific variants lower cross-verifier accuracy on sequential problems. Every cell uses the same seed; three-seed class-variant results are in Table 2.

class (n)	floor pick	cross-verifier	+ both classify arms	Claude judge	oracle
Combinational (74)	95.9	95.9	95.9	97.3	97.3
Sequential (82)	74.4	**79.3**	76.8	**86.6**	91.5
Overall (156)	84.6	**87.2**	85.9	**91.7**	94.2

variance). The Claude judge (Table 1) reaches 90.8% (3-seed mean, above 90% on two of three seeds); execution tools with a cheap judge reach 89.7 (Appendix P).

H PER-CLASS GAINS AND CLASS-SPECIFIC SELECTION VARIANTS (SEED-1)

Table 3 gives the per-class rows behind Figure 7. The combinational class starts at 95.9% and gains one correct problem with selection by the Claude judge; the larger gain lies in the sequential cells ($74.4 \rightarrow 86.6$ floor-pick-to-Claude-judge). The two class-specific selection variants — class-matched cross-verification stimulus and per-class candidate subsetting — lower accuracy on seed 1, and they do so on the very cells they target: cross-verifier $79.3 \rightarrow$ cross-verifier+classify 76.8 on the sequential half (-2 problems), and $87.2 \rightarrow 85.9$ overall (-2). Each arm alone costs about one problem on this seed (86.5 for either, against 87.2); over three seeds the arms are 0.0 to $+0.4$pp (Table 2). The selection loss holds at class granularity; what classification contributes is a map of which cells have headroom, not a specialization of generation.

We had offered confidence-gated routing as classification's cost payoff (Figure 5(c)): gating the Claude judge (three votes, at least \$0.124 per problem, $\geq 4.4\times$ one-shot; Appendix Table 7) on the 9% of problems where the cross-verifier's evidence is weakest reaches 87.8% ($137/156$, seed 1) at a revised cost of at least $1.0\times$ one-shot. It does not survive its controls — a random gate at the same rate does about as well, and at its revised price frontier one-shot (91.2%) dominates it (Appendix J) — so we withdraw it.

I FULL RESULTS AND PUBLISHED-SYSTEM CONTEXT

Full-results notes. † Per-seed $91.7/91.0/89.7$ against the matched one-shot draws $90.4/92.3/91.0$: the paired tests do not establish a win or equivalence — paired exact McNemar on our best seed (the only one with per-problem verdicts) against each draw gives $p = 0.79/1.00/1.00$ (discordant $8/6$, $5/6$, $7/6$). The Claude judge casts three votes, i.e. three calls per problem, so the harness with

Table 4: **Published systems provide context under different protocols.** Reported by Yu et al. (2026a), without re-scoring; their RTLLM denominator is 50, ours 48. Model versions, scorers, and sampling budgets differ, so cross-table comparisons are indicative, not controlled.

		VerilogEval-v2 (156)		RTLLM-v2		ChipBench-SC (30)		CVDP-CID003 (78)	
Type	Model / system	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
Reported by Yu et al. (2026a) (their scorer; RTLLM-v2 = 50 designs)									
Foundation	GPT-4o	64.1	73.7	56.5	70.3	20.0	33.3	39.0	40.4
	GPT-5.5	84.7	90.4	63.2	68.0	30.7	36.7	44.0	48.7
	Claude Opus 4.7	86.9	90.4	64.8	68.0	31.3	46.7	42.8	47.9
	DeepSeek Coder (236B)	68.5	80.8	57.6	70.0	16.7	30.0	22.3	37.2
	DeepSeek V4 (1.6T)	67.3	80.1	58.8	66.0	18.0	36.7	21.5	34.6
	DeepSeek R1 (671B)	77.5	84.7	64.7	75.8	26.7	40.0	27.7	42.1
Specialized	CodeV-R1 distill (7B)	65.2	75.2	57.2	71.9	13.3	26.7	26.2	42.1
	CodeV-R1 (7B)	68.8	78.2	68.0	78.2	30.0	40.0	26.8	43.3
Base	Qwen3.5-4B	41.7	60.9	34.3	49.7	6.7	10.0	11.8	13.9
	Qwen3.5-9B	48.5	66.6	36.1	57.8	13.3	20.0	13.3	21.5
Trained RTL	ChipMATE-Verilog-4B	67.4	71.8	68.0	74.6	26.7	33.3	24.7	39.2
	ChipMATE-Agents-4B	75.0	76.3	74.6	77.3	33.3	43.3	32.1	41.3
	ChipMATE-Verilog-9B	75.3	77.6	71.9	75.8	30.0	36.7	28.1	42.1
	ChipMATE-Agents-9B	80.1	82.4	75.8	77.3	36.7	43.3	40.4	44.6

Table 5: **Full measured results distinguish selected accuracy from pool coverage.** Member pass@1 averages each API name's retained samples (the two DeepSeek names are probably one model; Appendix B); pool pass@k is the unbiased estimator. The oracle is unavailable to the selector. Notes below specify seed counts and cells whose sampling budgets differ from the column header.

		VerilogEval-v2 (156)		RTLLM-v2		ChipBench-SC (30)		CVDP-CID003 (78)	
Type	Model / system	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
Measured in this work (RTLLM-v2 = 48; frontier-pool exception labeled)									
Pool member	`gpt-5-mini`	84.1±0.5	–	64.8±2.6	–	31.1	–	37.2	–
	`deepseek-chat`	72.6±0.6	–	62.7±2.1	–	26.7	–	34.6	–
	`deepseek-coder`	72.7±0.8	–	63.7±1.1	–	23.3	–	38.5	–
Cheap pool	pooled samples	76.4±0.4	90.6±0.7	63.6±1.1	78.8±2.6	27.0	37.8	36.8	48.7*
	ORACLE (pass@n)	93.2±1.0		81.2±2.1		40.0		48.7	
VERDICT	cross-verifier (r)	85.3±1.7	–	72.2±2.4	–	26.7	–	43.6	–
(ours)	Claude judge (r)	**90.8±1.0**†	–	**76.4±3.2**	–	31.1±5.1‡	–	42.3±1.8§	–
+ *exec.*	cross-verifier**	**86.8±1.5**	–	–	–	–	–	–	–
tools	Claude judge**	[91.5, 92.9]	–	–	–	–	–	–	–
no frontier	*tools + cheap judge***	**89.7±2.3**	–	–	–	–	–	–	–
same sel.,	*Claude K=3 pool*¶	**92.9**	93.6	–	–	–	–	–	–
Frontier	Claude-Opus-4.8 one-shot	91.2±1.0‖	92.3‖	85.4#	–	36.7#	–	47.4#	–

the Claude judge costs at least 5.0× a one-shot call (Appendix S). ‡ 3-seed mean±sd (per-seed 36.7/30.0/26.7) over three generation seeds, i.e. independently generated pools whose oracles are 40.0/40.0/43.3; the oracle cell prints seed 1. The ChipBench-SC margin (4.4pp) and the mean loss (10.0pp) in §6 are taken against the 3-seed oracle mean 41.1; the loss's sd (±6.7) is over the per-seed differences from each seed's own oracle (3.3/10.0/16.6). § 2-seed mean±sd (41.0/43.6) over two independently generated pools, one per generation seed (no problem has the same candidates in both), whose oracle (48.7) and cross-verifier (43.6) values coincide; the other CVDP-CID003 rows print seed 1. Seed 1 uses the record in which its 11 zero-vote fallbacks were re-judged (no pick's correctness changed), and the CVDP-CID003 judge prompt omits the single-correct premise (Appendix B). No gain over the cross-verifier, within noise (§6). ∗ CVDP-CID003 has $n \leq 3$ samples/problem, so the cell is pass@3 (= oracle). ¶ The same Claude judge over the *frontier's own* K=3 samples, all 156 problems, matched prompt. The P@5 cell is that pool's pass@3 oracle, so the selection loss is 0.6pp: 145 of the 146 problems that contain a correct design are returned (the one miss is a fallback caused by the arm's 2,000-token selector cap; completing its 13 capped votes could leave the loss anywhere in 0.0–1.3pp, so 0.6pp is not an upper bound, and the loss without the cap was not measured). +1.7pp over the matched baseline, but only ≤ 6 discordant problems, so the margin is not significant (McNemar $p = 0.22/1.00/0.25$) (Appendix N). ‖ Re-measured with the *byte-identical* prompt our pool receives, interface block included: draws 90.4/92.3/91.0, and the P@5 cell is their pass@3 (92.3), not a pass@5. The matched-prompt baseline is detailed in Appendix M. # Not re-measured under the matched prompt. On VerilogEval-v2 that correction *raised* the baseline (89.7 → 91.2), the direction of the prompt effect on these three benchmarks remains unmeasured. ∗∗ The same

Table 6: **Selector substitution gives the largest mean gain in this decomposition.** Selected pass@1, three-seed mean±sd (%). The oracle comparison changes sample count as well as model diversity; it does not isolate a family effect.

Configuration	VerilogEval-v2	RTLLM-v2
floor pick (cheap pool, no cross-verifier)	85.0±0.4	68.1±1.2
+ cross-verifier (full cheap harness)	85.3±1.7	72.2±2.4
cross-verifier contribution (over floor pick)	+0.2±2.1	+4.2±3.6
+ Claude judge	**90.8±1.0**	**76.4±3.2**
selector-substitution contribution (Claude judge − cross-verifier)	**+5.6±1.3**	**+4.2±2.1**
oracle (pass@any)	93.2±1.0	81.2±2.1
strongest member's own pass@3 oracle	89.5±0.7	72.2±1.2
pool-oracle gain (oracle − strongest member)	+3.6±1.3	+9.0±1.2

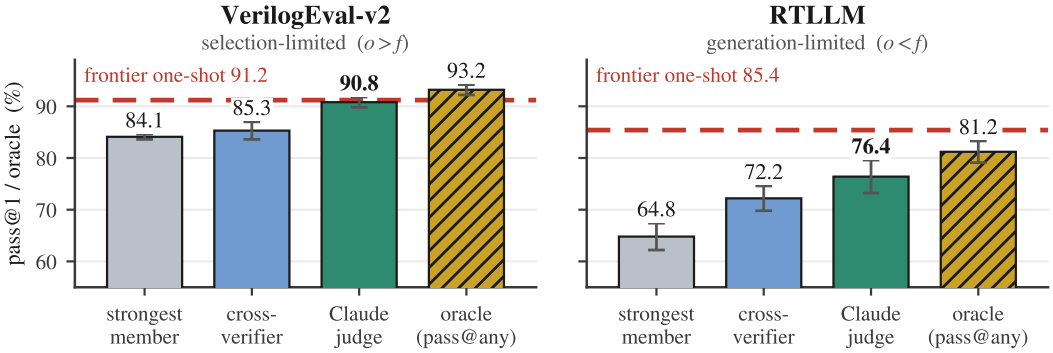


Figure 8: **The Claude judge improves both means; on VerilogEval-v2 it is statistically indistinguishable from one-shot.** (a): VerilogEval-v2 oracle exceeds one-shot; selected accuracy is nominally lower, a difference that is not significant. (b): RTLLM-v2 oracle itself is below one-shot. Bars are three-seed means±sd; the one-shot prompt is matched only on VerilogEval-v2.

verifiers after the harness's execution tools (Appendix P); VerilogEval-v2 only. Cross-verifier: exact on every seed (88.5/85.9/85.9). Claude judge: composed with its recorded verdicts — [93.6, 94.2] on seed 1, wider bounds on seeds 2–3, whose per-problem verdicts were not retained — so the cell is a conditional accounting interval for the 3-seed mean, not a measurement or a confidence interval. *No frontier*: the same tools with `deepseek-chat` (DeepSeek-V4.1-Flash, one-shot 75.6) as the judge, measured fresh on all three seeds, $0.022/problem (judge metered, generation re-metered).

Over the pool's strongest member, the Claude judge is +6.7pp on VerilogEval-v2 (84.1 → 90.8) and +11.6pp on RTLLM-v2 (64.8 → 76.4; Figure 8 draws the three-seed rows of Table 5). That changes pool, sample budget and selector at once, so it is an end-to-end figure; the single-variable number remains the selector substitution (+5.6±1.3), and against Table 6's other single-model baselines it reads +5.8 (floor pick) and +1.3 (that member's own pass@3 *oracle*, 89.5±0.7, unfavorable to us by construction). The systems reported by Yu et al. (2026a) (Table 4) provide context rather than a controlled ranking. Three caveats apply: their Claude row is a generation older (against our own Opus-4.8 baseline we are 0.4pp *below*, within noise); `gpt-5-mini` alone scores 84.1, so only +6.7 of the +10.7 over their best trained model is the harness; and their pass@1 rows use different sampling budgets from our ≤ 9 cheap samples; their Claude row's reported pass@5 (90.4) and our pool's measured pass@5 (90.6±0.7) provide context but do not establish equivalence.

Reporting details. Cost against accuracy on VerilogEval-v2, every arm priced from one ledger. List prices for the models actually called (Claude-Opus $15/$75 per M in/out, `gpt-5-mini` $0.25/$2.00, DeepSeek $0.27/$1.10). *Metered*: frontier one-shot ($0.0282, its 468 retained calls), the frontier K=3 arm, the frontier-free judge, and one `gpt-5-mini` generation call ($0.00324, re-metered on 40 random problems *with* its hidden reasoning; the frozen pool kept no usage records). A DeepSeek generation call is $0.00063, measured on the API's current alias. *Modelled*: the Claude judge's 3-vote pass over the cheap pool ($0.124, at 80 output tokens per vote against a measured

Table 7: **Frontier one-shot dominates the full configuration; no cheap configuration shows a demonstrated cost advantage.** The one-per-behavior variant's advantage is exploratory, with estimated generation cost. VerilogEval-v2 accuracy (%) versus cost; full usage, price assumptions, and limitations follow.

Configuration	$/problem	VerilogEval-v2	on the Pareto frontier?
`gpt-5-mini` one-shot	0.0032	84.1	yes (cheapest assessed)
cross-verifier + execution tools	0.017	86.8	no (one-shot mix: 88.1)
cross-verifier alone	0.017	85.3	no
VERDICT, no frontier model‡	**0.022**	**89.7**	no (next row)
judge shown one per behavior¶	0.019	89.7	yes (+1.1, exploratory)
premise-free selector prompt¶	0.018	88.9	no (mix: 89.3)
random: 75% Claude / 25% `gpt-5-mini`§	0.022	89.5	—
random: 43% Claude / 57% cross-verifier†	0.022	87.8	no
Claude-Opus-4.8 one-shot	0.028	91.2	yes
frontier K=3 + our 3-vote selector	0.179	**92.9**	yes (most accurate)
VERDICT routed (9%, withdrawn)	$\geq$ 0.029	87.8	no (one-shot dominates)
VERDICT full (3-vote Claude judge)	$\geq$ 0.141	90.8	no (one-shot dominates)
one cheap-pool sample (average)	0.0015	76.4	—

† The elementary baseline: send a random fraction of problems straight to frontier generation and the rest to the cross-verifier harness; the fraction is set so the mixture costs what the frontier-free configuration costs, and its accuracy is the per-seed mixture of the two arms' 3-seed results (87.5–88.6 by seed, all below the frontier-free row; 88.1–89.3 if the cross-verifier end also uses the tools, one seed above it). The revised pricing places the routed arm below frontier one-shot in accuracy at a higher cost. ‡ The cheap pool, the execution tools and a cheap judge (`deepseek-chat`, now DeepSeek-V4.1-Flash, whose own one-shot is 75.6), 3-seed mean (144/137/139 of 156, ±2.3): $0.0085 metered for the judge plus $0.0135 for the nine candidates (Appendix P). Its accuracy gain over the cross-verifier mixture is not significant ($p = 0.18$). § The strongest elementary baseline from the measured one-shot endpoints: at each price, mix the two one-shot baselines (`gpt-5-mini` 84.1 at $0.0032, Claude 91.2 at $0.028). It reaches 89.5 at the frontier-free price and 88.1 at the cross-verifier's price, so it dominates both cross-verifier rows; it trails the frontier-free row by 0.3pp and the two remedy rows by 1.1 and 0.7pp (exploratory, ¶). Mixtures are computed from unrounded accuracies (Claude $427/468 = 91.24$, frontier-free $420/468 = 89.74$), so displayed rows can differ by 0.1 in the last digit. ¶ The two remedies of Appendix Q, same pool and tools, fresh 3-seed runs: judge cost $0.0055 and $0.0042 metered, plus the $0.0135 generation estimate. The one-per-behavior row's parenthesis is its accuracy gain over the price-matched mix of the two one-shots (88.6); the premise-free row is 0.7pp above that mix (88.2) but below a mix of the one-per-behavior variant and `gpt-5-mini` one-shot (92%/8%), which reaches 89.3 at its price. These gains are exploratory: the generation cost is re-metered on a sample, and neither remedy changed accuracy detectably relative to the original prompt.

median of 307), so the two rows that use it are lower bounds and could only move right. The pricing correction is recorded in Appendix B. **VERDICT's full configuration is dominated by frontier one-shot**. With randomized mixtures admitted, the frontier runs through `gpt-5-mini` one-shot, the one-per-behavior variant, Claude one-shot and the frontier K=3 arm. The original frontier-free configuration sits only 0.3pp above a random mix of the two *one-shot* baselines at its price and is dominated by that variant; the variant's 1.1pp margin is a point estimate from a run made after the original, with an estimated generation cost, so the harness shows no *demonstrated* cost advantage over choosing between two models.

J THE ROUTING OPERATING POINT, SWEPT END TO END

The withdrawn routing configuration escalates the 9% of problems (14 of 156) on which the cross-verifier's evidence for its own pick — full cross-family matches, V–V votes and mean match rate — is weakest, and scores 87.8% (137/156, seed 1). Its revised cost is at least one-shot's cost. The sweep below evaluates the gate against random routing and price-matched alternatives; the corrections log (Appendix B) records the earlier 88.5 at this fraction and the superseded pricing.

Is the gate real? Against a *random* router at the same escalation fraction (mean of 200 shuffles), the confidence gate is worth nothing at the operating point (87.8 vs 87.8 at e=15%, 87.8 vs 87.6 at e=9%) and only +1.4 to +1.9pp in the middle of the range (e=0.3–0.5), where the design already

costs more than frontier one-shot. An *oracle* router that escalates exactly the problems the Claude judge would fix reaches 93.6, so the headroom exists; what is missing is a cheap signal that finds it.

At equal cost. Frontier one-shot costs a metered \$0.028 per problem and the routed design \$0.0174$+e\times$(at least \$0.124), so the two are equal at $e \leq 8.7\%$ — essentially the routed point itself, which scores 87.8 against the frontier's 91.2. The routed design is *better than* the cross-verifier alone (87.2 at $0.62\times$) but never cheaper than frontier generation at a comparable accuracy, and we do not claim a Pareto win.

Choosing the threshold honestly. Selecting e by 5-fold cross-validation — fitted on four folds, scored on the fifth — returns $e{=}0$ under an accuracy-minus-cost objective, i.e. do not escalate at all. The 9% point is therefore not one a held-out procedure recovers, and once generation is re-metered it is not cheaper than one-shot.

This bounds what the keyword classifier can claim. Its selection-side arms lower accuracy on seed 1 (-1.3pp; 0.0 to $+0.4$pp over three seeds, Table 2), and its routing arm, withdrawn, had a gate that carried little information at its operating point. Classification provides a *diagnostic* that locates the larger share of headroom in the sequential cells (§5).

K What the pool comparisons leave unresolved

Shared errors, specification comprehension, and candidate count are possible explanations for selection loss. The following arms keep the specifications fixed while changing pool size and composition. They show variation at fixed specification length, but do not separate the remaining explanations.

pool (all VerilogEval-v2, median spec 584 chars)	cands	oracle	r	ℓ
frontier Claude $K{=}3$	3	93.6	92.9	**0.6**
cheap pool $K{=}3$ (seed-1)	9	94.2	91.7	2.6
cheap pool + one frontier sample	10	96.2	91.7	4.5
four-family $K{=}3$	12	93.6	90.4	3.2
union of every pool generated	32	96.8	89.1	7.7

Specification length does not explain this within-benchmark variation. Every row uses the same 156 problems while ℓ ranges from 0.6 to 7.7pp. This holds specification length fixed, not comprehension itself. Whatever sets ℓ here, it is not how long the specifications are. The spec-length relation of §6 is a *between*-benchmark pattern that four benchmarks do not establish, and it does not explain the within-benchmark spread. Across benchmarks, Claude-judge capture falls from 97.5 to 94.1 to 86.8% as median specification length rises across VerilogEval-v2, RTLLM-v2, and CVDP-CID003, where it drops below the cross-verifier's 89.5% (the CVDP-CID003 judge's prompt differs, Appendix B); ChipBench-SC (1,210 characters) breaks the trend at $76.1{\pm}15.1\%$.

Count and confusability are not separable, and the mixed pool shows why. Loss rises monotonically with candidate count over the homogeneous series ($3, 9, 12, 32 \rightarrow 0.6, 2.6, 3.2, 7.7$), which is experiment (viii)'s reading. The mixed-pool row complicates this reading: adding one frontier sample to the cheap pool raises the oracle $94.2 \rightarrow 96.2$ and raises the loss $2.6 \rightarrow 4.5$, a bigger jump than adding two more cheap families does. The result suggests that count alone is an incomplete proxy for selection difficulty. The set also gained a distinct correct candidate; on diagnostic pools, distinct correct candidates lengthened a cheap judge's output under the single-correct premise, but no such effect was detected for the frontier judge these rows use (Appendix Q). The same reading is consistent with the top row: the frontier's own $K{=}3$ pool is the smallest and has the lowest measured selection loss. Size, quality, and composition change together.

What we therefore claim. Confusability — how many candidates a verifier must tell apart and how nearly they agree — is consistent with these five arms. All five used the single-correct premise and we did not re-run them without it; the premise was never removed for this judge, adding correct candidates had no detected effect on it, and on real pools removing it changed no end-to-end accuracy for the cheap judge (Appendix Q), so we do not offer it as an explanation. Candidate count tracks loss across these particular pool choices, but its effect is not isolated from composition or quality. Spec length is a *between*-benchmark correlation, not an isolated effect. We have not isolated error correlation from confusability. The controlled pools of Appendix Q vary how many copies stand

Table 8: **Labeled slices resolve large gaps more reliably than near ties.** Commit rate and correctness given commitment over 3,000 draws per cell. The whole-benchmark target includes the sampled labels: this is sampling uncertainty, not an out-of-sample evaluation.

	true gap	m=15		m=25	
Cell (n)	(pp)	commits	right	commits	right
decision-relevant gap, negative					
RTLLM-v2 s1 (48)	−6.3	75.5%	83.2%	84.1%	93.2%
RTLLM-v2 s2 (48)	−12.5	88.1%	97.7%	97.6%	99.6%
RTLLM-v2 s3 (48)	−8.3	78.9%	100.0%	95.3%	100.0%
ChipBench-SC s2 (30)	−6.7	74.1%	91.7%	95.4%	99.9%
ChipBench-SC s3 (30)	−10.0	88.8%	100.0%	99.7%	100.0%
decision-relevant gap, positive					
frontier pool (156)[‡]	+2.6	43.9%	86.0%	58.9%	87.9%
gap inside the noise					
VerilogEval-v2 s1 (156)[†]	+1.3	61.7%	59.6%	71.7%	62.8%

On the five cells whose gap exceeds 5pp it commits 74–89% of the time at m=15 (84–100% at m=25) and is right 83–100% of those times (93–100% at m=25). On the positive cell it commits only 44–59% and is right 86–88% of those. On the near-tie it commits 62–72% and is right about 60% — i.e. *it does not reliably abstain there*. † The paired seed-1 comparison (Claude judge 91.7 vs matched draw 0 90.4); Table 1's Claude-judge row averages three seeds against three draws and is −0.4pp, not significant. ‡ The Claude judge over the frontier's own K=3 pool, all 156 problems, matched prompt (Appendix N), against matched draw 0.

behind each of two fixed behaviors at a fixed count; they detect no majority-following (with low power) and expose the premise instead.

L Sampling uncertainty in the margin/loss comparison

Call the decision "commit to the pool when the estimated margin $o - f$ exceeds the estimated loss ℓ by more than 1pp" the *rule*; §6 states its algebra as $r > f \Leftrightarrow o - f > \ell$. As algebra the rule is an identity on measured runs: since $\ell := o - r$, with r the selected accuracy, the inequality $(o - f) > \ell$ reduces to $r > f$, so on the run it was measured on the rule attributes rather than predicts (§6). This appendix therefore tests the operational claim — *estimate the margin and the loss on a small labeled slice, then decide*. That claim is about sampling noise, not algebra, and it can fail: a slice small enough to be worth labeling may be too noisy to decide anything. It is *not* an out-of-sample test: the whole-benchmark target contains the slice, which is 83% of ChipBench-SC (25/30) and 52% of RTLLM-v2 (25/48), so on those cells agreement is partly by construction, not prospective transfer to a new benchmark.

Protocol. For each benchmark–seed cell with per-problem records we draw m problems uniformly at random, estimate (o, ℓ, f) on them, and ask whether the resulting decision matches the *whole* benchmark's outcome — the practitioner's question, "I labeled m problems; is my decision right here?". 3,000 draws per cell. The rule *commits* only when its estimated gap exceeds 1pp; abstaining inside that band is correct behavior, not failure, so we report commit rate and correctness conditional on committing. The old asymmetric agreement metric is withdrawn (Appendix B). Every cell is admitted only after its per-problem records reproduce the aggregate this paper publishes for it to within 0.1pp. The reconciliation gate excludes artifacts whose stored outcomes fail this check; the corrections log records the earlier failure.

Reporting details. The true gap is selected accuracy minus frontier one-shot accuracy on the whole benchmark; a draw of m labeled problems *commits* when its estimated gap exceeds 1pp in magnitude.

Result. Scored symmetrically (commit rate and accuracy given a commitment; Table 8), at m=25 the five decision-relevant negative cells commit 84–100% of the time and are right 93–100% of those times. Two dozen labeled problems are therefore informative where the gap is large, and the near-tie paragraph below says where they are not.

It is not merely a loss detector. Five of the seven cells are losses, which would leave open whether the rule has any positive resolving power, so we add the one cell with a decision-relevant *positive* gap: the Claude judge over the frontier's own $K{=}3$ pool (gap $+2.6$pp, all 156 problems, Appendix N). There the rule is right 86% of the time it commits at $m{=}15$, 88% at $m{=}25$ and 98.8% at $m{=}80$, while the commit rate climbs from 32% at $m{=}10$ through 44% at $m{=}15$ and 59% at $m{=}25$ to 92% at $m{=}80$: a 2.6pp gap needs a larger slice than a 6–12pp one, which is the rule behaving as designed.

The near-tie is not reliably resolved. On the VerilogEval-v2 cell with true gap $+1.3$pp, the rule commits 71.7% of the time at $m{=}25$ and is correct on 62.8% of those commitments (Table 8). Even at $m{=}80$, mean absolute error is 1.8pp, larger than the effect. This quantifies a resolution limit; it does not establish equivalence or reliable abstention near a tie.

An exploratory cross-benchmark fit. The correlation between capture and specification length suggests checking whether a fit predicts a benchmark omitted from fitting. Fitting capture $\sim a + b\log(\text{median spec chars})$ on three of the four Verilog benchmarks and predicting the fourth gets the win/lose sign right on $3/4$ with a mean absolute error of 7.8pp on selected accuracy. The one miss is ChipBench-SC, which §6 already identifies as sitting below the capture curve and has under-specified testbench conventions (Appendix D). This association does not establish why the fit misses. With four benchmarks this is indicative at best, and the error is far too large for the fit to be used as a predictor; we retain it as an exploratory prediction check, not a test that isolates comprehension.

M Prompt parity with the frontier baseline (and the correction it forced)

The defect. Candidate generation appends an explicit module-interface block — exact port names, directions and widths — to the spec (on VerilogEval-v2 parsed from the module header of the reference file, our one reference-derived input; on RTLLM-v2 parsed from the specification text), so no candidate of ours can fail on a guessed interface. Our first frontier baseline did not receive that block: the baseline script parsed the same ports and then discarded them, leaving the frontier model to infer the interface from prose while our pool was handed it. The paper nonetheless described both arms as sharing a matched prompt. That claim was wrong, the asymmetry ran in our favor, and two stored runs bracketed what it was worth: 89.7 without the block and 92.9 for an earlier port-explicit variant.

The correction. We re-ran the VerilogEval-v2 baseline under byte-identical prompt construction — same role preamble, same class guidance, same spec, same interface block, same extractor, same scorer — matching the prompt and scoring protocol for the comparison across generation strategies. Three independent draws give $90.4/92.3/91.0$, i.e. $\mathbf{91.2{\pm}1.0}$, with pass@3 $= 92.3$. The corrected baseline is 1.5pp *stronger* than the one the earlier draft compared against, and it is the number every VerilogEval-v2 comparison in this paper now uses. Unlike the earlier baselines, the generations are retained, so this arm can be re-scored if the scorer changes again.

What it costs us. The Claude-judge/one-shot difference moves from a nominal $+1.1$pp over one-shot to a nominal -0.4pp: $90.8{\pm}1.0$ against $91.2{\pm}1.0$. Paired exact McNemar of our best seed $(143/156)$ against the three baseline draws $(141/144/142)$ gives $p = 0.79/1.00/1.00$ (discordant pairs $8/6$, $5/6$, $7/6$) — the arms are statistically indistinguishable (paired 95% interval $[-3.4, +6.0]$pp), so we drop the claim of a beat and report a statistical tie, without asserting equivalence within a preset margin. The results that do not move are the ones the paper rests on: the oracle still exceeds one-shot on every seed ($94.2/92.9/92.3 > 91.2$), the Claude-judge gain over the cross-verifier is unchanged ($+5.6{\pm}1.3$pp), and the frontier-pool arm (Appendix N) is unaffected because it is controlled against the stored draws generated from that same prompt.

Scope of the correction. Only VerilogEval-v2 was re-measured. RTLLM-v2, ChipBench-SC and CVDP-CID003 keep the unmatched baseline (85.4, 36.7, 47.4). The direction of the prompt effect on these benchmarks is unmeasured. One further asymmetry runs the other way and we leave it: a curious side effect of the interface block is that it *lowers* baseline pass@3 (92.3 matched over three draws vs. 94.4 unmatched over five), consistent with a more constrained prompt producing less diverse failures; we do not claim this as a finding, since the two runs are months apart.

N SELECTION ON A FRONTIER MODEL'S OWN POOL

The cheap-pool arms leave a further question: is the selection loss a property of *cheap* pools, or of imperfect verification as such? We answer it by pointing the same 3-vote frontier selector at Claude-Opus-4.8's *own* $K{=}3$ samples, on all 156 problems.

Design. The model generates $K{=}3$ candidates per problem from the byte-identical prompt our cheap pool receives — role preamble, class guidance, spec, and the explicit module-interface block — our 3-vote selector (same prompt, same majority rule as every other arm) picks one, and the official testbench scores it once. The comparator needs no new run: the matched-prompt one-shot baseline of Appendix M was drawn from that same prompt with its generations retained, so the two arms are matched by construction rather than by assertion.

Result. Realized accuracy is $145/156 = \mathbf{92.9}\%$ against the matched baseline's $91.2{\pm}1.0$ (per-draw $90.4/92.3/91.0$) — $+1.7$pp on the mean, $+2.6$pp on the draw the arm is paired against. The pool's own oracle is $146/156 = 93.6\%$, so the **selection loss is 0.6pp ($1/156$): of the 146 problems where a correct design was generated, the selector returned it on 145.** Two problems produced no parsable vote and fell back to the first candidate. The selector votes in this arm were capped at $2{,}000$ output tokens (the cheap-pool Claude-judge runs were uncapped); 13 of its 225 votes hit that cap before an `ANSWER` line, which produced both fallbacks, and the one miss (`review2015_fancytimer`) is a fallback after all three of its votes did, so on no problem that holds a correct design did an answered majority choose an incorrect one. That does not make 0.6pp an upper bound on the judge's loss: on `lemmings4` the returned correct design rests on its one answered vote, the other two having hit the cap, and two completed votes for its incorrect candidate would outvote it. Holding the 212 answered votes fixed, completing the 13 capped ones could leave the loss anywhere from 0 to 2 problems (0.0–1.3pp); the remaining capped votes cannot change their problems' outcomes (two answered votes already agree, or every candidate is correct, or none is). The uncapped loss itself was not measured.

Significance. Paired exact McNemar against the three matched draws gives $p = 0.22/1.00/0.25$ (discordant $5/1$, $2/1$, $3/0$). *The positive difference is a point estimate, not an established win*: with at most six discordant problems out of 156 this design cannot resolve a 1.7pp effect, and we do not claim it does. What the arm establishes is not the margin but the capture rate.

Reading: the smaller Claude-generated pool has lower measured selection loss. The same selector loses 2.4pp of its cheap pool's oracle (90.8 of 93.2, capture 97.5%) and 0.6pp of the frontier pool's (92.9 of 93.6, capture 99.3%). The selection loss is therefore not a property of cheap pools alone. Pool quality, size, composition, and selector budget are not separately controlled, so the lower loss is consistent with easier discrimination among fewer candidates but does not isolate it. Whether the single-correct premise binds here we cannot say: on the 75 judged problems, at least 62 of which hold two or more distinct correct candidates, the frontier judge answered 212 of its 225 votes (94.2%); the 13 unanswered votes were cut off at the $2{,}000$-token selector cap rather than refused (Appendix Q). Two further things follow, and they pull in opposite directions for us. The frontier leaves 2.4pp of its own coverage unclaimed at $K{=}3$ (93.6 available, 91.2 one-shot) and our selector claims 71% of it. But $92.9 > 90.8$ means a cheap pool is not the accuracy-optimal choice, and no cheap configuration shows a demonstrated cost advantage either (§5). And this is the margin/loss comparison's (§6) one positive case measured at full scale: margin $o - f = 93.6 - 91.2 = 2.4$ exceeds loss $\ell = 0.6$, selected $92.9 > 91.2$ is the corresponding positive, nonsignificant difference.

O ADDITIONAL EXPERIMENTS ON SELECTION LOSS

Thirteen experiments on VerilogEval-v2 characterize the gap and cover the routes past it we could think of (i–iv on the fixed seed-1 pool, on which the cross-verifier scores 87.2 — the seed-1 value behind the 3-seed mean $85.3{\pm}1.7$; v–viii on freshly generated and merged pools; ix–x on the selector's prompt, identity, and tools; xi–xiii on agentic repair and synthesis).

(i) Scaling generation *widens* the selection loss (Figure 9). As the pool grows $K{:}1 \to 3$ the oracle climbs $90.4 \to 92.9 \to 94.2$, but cross-verification stays pinned at 87.2 at every K — the added candidates increase coverage without increasing the count of correct final selections. The selection

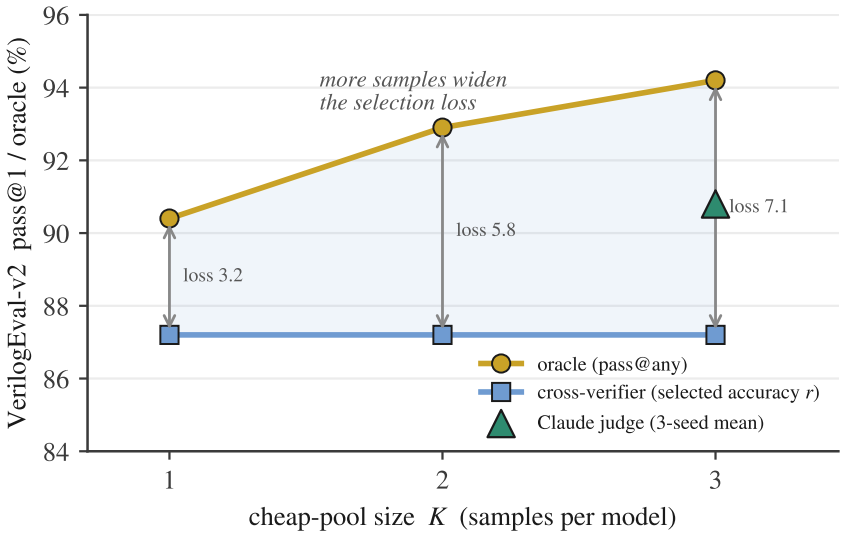


Figure 9: **Adding samples increases coverage without improving cross-verifier accuracy in this sweep.** VerilogEval-v2 seed 1: oracle rises $90.4 \rightarrow 94.2\%$ as K grows, while cross-verifier accuracy stays at 87.2%, so the selection loss widens $3.2 \rightarrow 5.8 \rightarrow 7.1$pp (5, 9 and 11 of 156 problems). The Claude marker is a three-seed mean and is not part of the matched seed-1 comparison.

loss therefore widens with test-time scaling ($3.2 \rightarrow 5.8 \rightarrow 7.1$pp: 5, 9 and 11 of 156 problems). Scaling *generation* makes *selection* the growing bottleneck — the inverse of the best-of-N premise.

The same pool's pass@k curve. The curve (Figure 10; every one of the nine candidates re-scored against the official testbench, offline) reads $76.7/83.9/87.5/91.3/93.2$ at $k = 1/2/3/5/7$, against the seed-1 oracle 94.2 ($147/156$); the strongest pool member's pass@1 is 84.1 (3-seed `gpt-5-mini`, Table 1). The curve stops at k=7 because the pool holds 7–9 candidates per problem, so at k=9 the unbiased estimator is undefined wherever $n < k$ and, scored naively, returns a value above the oracle it is bounded by; the corrections log (Appendix B) records the withdrawn value. The selected-accuracy lines in the figure are three-seed summaries, so their distance from the seed-1 curve is not a matched selection-loss estimate; the matched estimate is the K sweep above.

(ii) Selection loss concentrates where the pool disagrees. A problem is disputed when its largest V–V agreement cluster covers less than half the pool. On the 31 disputed seed-1 problems, the cross-verifier returns $20/31$ correct designs (64.5%), while $29/31$ contain one (93.5%). On the other 125 problems it returns $116/125$ (92.8%), with $118/125$ available (94.4%). The $147 - 136 = 11$ lost seed-1 problems therefore split into $29 - 20 = 9$ disputed and $118 - 116 = 2$ others. This concentration is consistent with, but does not test, the shared-error hypothesis.

(iii) A learned pool-internal verifier does not escape it. A gradient-boosted classifier over the behavioral features (full V–P matches, V–V votes, mean match-rate), evaluated with *problem-disjoint* 5-fold cross-validation (paper-fair: a problem's label never trains on itself), reaches only 83.3% — below the hand-built cross-verifier (87.2) and far below the Claude judge (90.8). This learned selector did not resolve the loss with these features and this amount of data. That is evidence against the easy answer to "why not just train a verifier?", not a proof that no learned verifier can.

(iv) Confidence ranks correctness but does not establish a useful routing policy. A learned logistic gate over cheap signals (top match-rate, top V–V votes, top-1/top-2 margin) predicts whether the cross-verifier's pick is correct at AUC 0.67 (5-fold cross-validated), so it does rank which problems need the Claude judge. Re-scored with the Claude judge's actual per-problem outcome on the same pool and the same learned gate, the curve is $5\% \rightarrow 87.2$, $9\% \rightarrow 86.5$, $15\% \rightarrow 88.5$, $25\% \rightarrow 89.7$, $50\% \rightarrow 91.7$, $100\% \rightarrow 91.7$ ($136/135/138/140/143/143$ of 156). This is a different gate from the evidence-ordered gate of Appendix J, and it reaches the same conclusion: the learned gate trails a random router at the operating point (86.5 vs 87.6 at 9%) and beats it by about a point only from e=25%, so the revised results do not support routing as a cost or accuracy contribution. We keep the

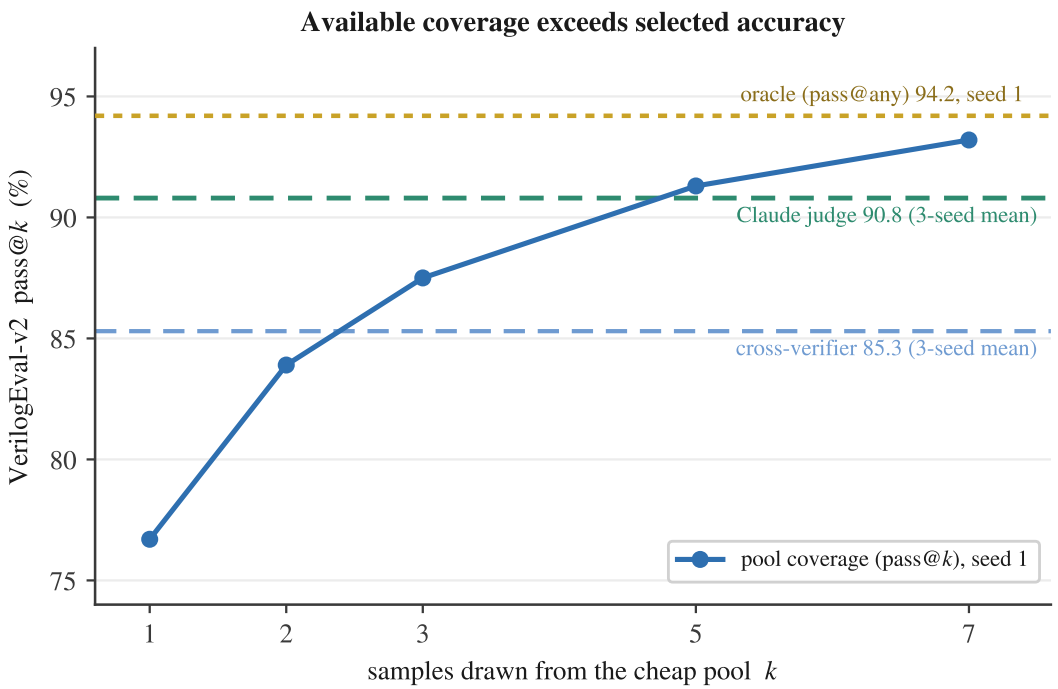


Figure 10: **Available coverage exceeds selected accuracy.** The pass@k curve and oracle are from the stored seed-1 pool; the two selected accuracy lines are three-seed summaries, shown for context. Their vertical distance from the curve is not a matched selection-loss estimate. The curve stops at k=7, the smallest pool size; Appendix O gives the values.

experiment because the AUC result stands — the signal ranks — and because the error is instructive: an "idealized" escalation model silently turns a router into an oracle.

(v) A four-family pool widens the cross-verifier's loss. A four-family pool (`gpt-5-mini` + `deepseek-chat` + `qwen-2.5-coder-32b` + `llama-3.3-70b`, K=3, $\geq$3 families on all 156 problems) has oracle 93.6 but the cross-verifier falls *below the floor pick* (79.5 vs. 84.0): the measured selection loss is 14.1pp. This is consistent with disagreement making consensus less informative, but family count and pool composition change together. The Claude judge on the same pool reaches 90.4 (+17 problems, +10.9pp net over the cross-verifier, 0 fallbacks) — compared with 90.8±1.0 on the default pool, without an equivalence claim. In this pool, more diverse generation bought coverage, not selected accuracy.

(vi) Eight model-written testbenches per problem still select below the floor pick. We strengthened the model-testbench selector to eight self-checking Verilog testbenches per problem authored by all four families, with aggressive argmax-by-pass-count selection (no anchor on the cross-verifier's pick). Result: 80.1%, still below the floor (84.0), with recoveries and breaks canceling inside the sequential class. On 59/156 problems even *correct* designs fail the majority of model-written tests. This suggests that incorrect tests limit this selector, but does not establish matched difficulty or shared specification errors between generation and test writing. Making cheap verification active rather than passive did not rescue it in the variants we tried (all under the selector prompt of Appendix Q); the test author here comes from the same pool as the design author.

(vii) Per-class expert prompts trade coverage for consensus. Applying the per-class plan-then-code protocol (the classifier's per-class generation prompts) to the diverse pool homogenizes its outputs: the cross-verifier recovers from 79.5 to 84.0 (candidates agree more), but the oracle *drops* from 93.6 to 91.7 — the protocol narrows exactly the diversity that produced the extra correct designs — and the Claude judge lands at 88.5. The prompt intervention changes coverage and selected accuracy differently for the two selectors; it improves the cross-verifier while lowering selection by the Claude judge.

(viii) Verification does not scale with the pool — even for the Claude judge. Taking the union of every pool we generated (three homogeneous seeds, the four-family pool, the expert-prompted pool; deduplicated, mean 32.5 candidates/problem) raises the oracle to $\mathbf{96.8}\%$ — the pools recover *different* problems. But the Claude judge on this union reaches only 89.1% ($10/156$ residual fallbacks to the cross-verifier's pick; ≤ 90.4 even if all resolved): its selection gap *grows* with candidate count — 2.6pp at 9 candidates (seed 1) (homogeneous), 3.2pp at 12 (diverse), 7.7pp at 32 (union). Every added candidate is consistent with acting as another plausible decoy, and capture degrades as the ceiling rises. This extends experiment (i) from the cross-verifier to the Claude judge: here scaling generation widened the selection loss for both judges; growing the pool was not a free lunch even with a frontier judge.

The sharpest single case adds a frontier sample rather than another cheap one. Handing the same selector the cheap pool *plus one frontier sample* — the retained generation from the matched-prompt baseline (Appendix M), so the added candidate is byte-identical to the one that baseline is scored on — lifts the oracle $94.2 \rightarrow 96.2$ and leaves selected accuracy at 91.7, *exactly* the cheap-pool figure: $+2.0$pp of headroom, 0.0pp cashed, the selection loss widening $2.6 \rightarrow 4.5$pp (seed-1 only, n=156; we report it as a single-seed observation supporting the experiment above, not as a separate arm). Two incidental facts from the same run are worth recording. On $38/156$ problems (24.4%) the frontier model's code is *byte-identical* to a cheap candidate's, 37 of them correct — the cheap pool does not merely contain an equivalent answer, it often contains the same one. And $53/468$ selector calls (11.3%) exhausted a 16k output budget without emitting a decision; capping the selector at 2k costs nothing in accuracy and over half the bill, which is the verification-side analogue of the generation-side token-starvation bug (Appendix E, bug 5). It resembles the budget exhaustion that Appendix Q associates, for a cheap judge on the discovery pools, with the selector prompt's single-correct premise; we did not re-run this arm with the premise-free prompt, so decoys and the premise are not separated here.

(ix) Prompt variants did not raise selected accuracy. We varied the *selector's* prompt (plain holistic pick; class-routed verification checklists — the classifier's classes applied to verification; structured per-candidate elimination; an error-analysis-driven hybrid in which the judge *nominates* every functionally-correct candidate and cross-family agreement breaks ties mechanically) and the selector's identity. A cheap judge (`qwen-2.5-coder-32b`) lands at 76.9–85.3 under every prompt — *below* the cross-verifier (87.2), and the checklist makes it *worse* (-10pp below the cross-verifier: confidently wrong, now with a checklist). A second frontier judge (`gpt-5`), measured cleanly (24k-token budget, $2/156$ no-answer), reaches 87.8 plain and 87.2 under the nominate-all hybrid — the cross-verifier's level, far below Claude's 91.7, with its errors concentrated in the same place (sequential/waveform problems: 80.5% vs. Claude's $\sim 89\%$). Reading its reasoning shows the same failure mode as Claude's: *confident misreadings of the spec or timing diagram*, which poison nomination sets and tie-breaks alike. Across seven selection protocols and three judges, no prompt or protocol variant raised selected accuracy — including, later, the premise-free prompt of Appendix Q (88.9 vs 89.7, $p = 0.45$); the only intervention that moved a number was fixing a measurement bug (an 8k token cap that starved `gpt-5`'s reasoning, $84.6 \rightarrow 87.8$). Prompts do change a cheap judge's behavior — its answered rate, truncation and cost — so we do not claim prompting is irrelevant, only that no variant we tried moved end-to-end accuracy.

(x) The tested simulation-assisted protocol lowers selected accuracy; inspected transcripts include incorrect specification interpretations. We gave the second frontier judge an *execution loop*: it designs discriminating input sequences from the spec, the harness simulates every candidate on them (`iverilog`, real outputs, candidates grouped by behavior), and the judge picks with the table in hand (nominate-all + consensus tie-break). Result: 83.3 — *below* the 87.2 of the same nominate-all protocol without tools (87.8 under the plain prompt), with the sequential class dropping to 72.0 (recovered 2, broke 8). The inspected transcripts suggest why. Where the judge's error was a faulty *mental simulation* of the RTL, the tool corrected the inspected scenarios' simulated outputs. Where the error was a misreading of the *spec*, the judge derived expected outputs from the misread spec and eliminated every candidate that contradicted them — including the correct ones. In the inspected transcripts, execution evidence helped only downstream of comprehension; where the judge had misread the spec, its rejections followed the misreading. The aggregate comparison alone does not establish this mechanism.

(xi) Agentic repair with pool models: $+0$. The harness ships a repair loop that is closer to a modern coding agent than to a selector: simulate the selected candidate against every Python reference in the

pool (both families), extract the cases where ≥ 2 references *agree with each other* and disagree with the candidate, re-derive from the spec (never from a reference, which is not an oracle), and accept the rewrite only if mean agreement across all references strictly improves. It is gated off by default, so every number elsewhere in this paper was produced with it disabled. Switching it on and widening its trigger from the shipped $11/156$ problems to the 39 whose selected candidate does not already agree perfectly with every reference — repairing with a *pool member*, so generation stays frontier-free — accepts 5 rewrites and nets 0**: zero fixed, zero broken**. Its own failure mode motivated the next variant: $12/39$ problems produced no consensus case at all. This run used the seed-1 pool, whose references number four on 37 and three on 2 of the 39 triggered problems (Appendix B), so the missing signal is not a two-reference artifact: on hard problems the references disagree among themselves. Adding three further DeepSeek-written references (seven per problem on seed 1; five on seeds 2–3, whose pools hold two) restores the signal on seed 1 (consensus cases $27/39 \rightarrow 32/39$) and still nets nothing across three seeds: $+1/-1/+1$ problems, $+0.2 \pm 0.7$pp. (The "pool member" in (xi) and (xii) was called as `deepseek-chat`, one of the two names the pool's DeepSeek samples were drawn under in June, when DeepSeek documented it as the non-thinking mode of DeepSeek-V4-Flash (Appendix B); by September that API alias resolved to DeepSeek-V4.1-Flash, a newer model of the same family. Both loops were therefore run with a same-family model that is not the pool's own generator; we did not measure which of the two is stronger.)

(xii) Candidate synthesis: -1, and it never breaks the ceiling. Selection is bounded by pass@any, so the harness's synthesis stage — merge the correct logic from several candidates, guided by a cheap model adjudicating the contested inputs against the spec — is the only mechanism here that could exceed the oracle. Also off by default. Enabled on the same 39 problems it accepts 5 merges and nets -1 (one fixed, two broken); on the 4 problems where no candidate is correct, and where exceeding pass@any is the only way to score, it succeeds **zero** times.

(xiii) Why the agentic loops fail — it is not for lack of a signal. The two loops are blind on almost disjoint sets: repair on $12/39$ (the references contradict each other) and synthesis on a different $12/39$ (the candidates agree, so nothing is contested), overlapping on only 3. On the remaining $18/39$ *both* loops had a usable disagreement signal and still netted nothing. The loops therefore failed even on problems with a disagreement signal. The result is consistent with difficulty deciding which implementation follows the specification; it does not identify the cause or rule out other repair methods.

Together, (i)–(xiii) cover the routes we tried: passive consensus (twelve selectors), learned selection, pool diversity, active model-written tests, per-class generation scaffolds, pool union, selector prompting, and tool-augmented agentic verification all leave selected accuracy at or below the homogeneous configuration; what moved it is the Claude judge, or — found later — execution that takes decisions away from a cheap judge (Appendix P), the former landing at 90.4–90.8 on the 9–12-candidate pools and degrading only as the pool itself is homogenized or inflated (88.5 expert-prompted, 89.1 union). Within the variants we tried, selected accuracy tracked the selector more than the pool or the protocol; most arms shared a selector prompt with a false single-correct premise (Appendix Q lists the exceptions). Which *property* of the selector is decisive we do not know: the one other frontier judge we tested does not reproduce the gain on the same pool (-1.4pp on the 144-problem 3-vote run, $+0.6$pp plain; Appendix S).

P EXECUTION TOOLS: TAKE THE DECISION AWAY FROM THE JUDGE'S READING

Experiment (x) and the evidence-fed judge below are consistent with execution checks depending on a judge's reading of the specification. The matched comparison alone is not significant. The three harness-side tools here are built the other way round — each makes a decision that no model's reading enters — and they run before any verifier, which is left unchanged. All three read only S and the candidates; T is never read.

The tools. *(i) Compile filter:* candidates that do not compile or simulate are dropped. *(ii) Spec-waveform replay:* 12 of the 156 VerilogEval-v2 specs print a simulation waveform of the top module (a `time` header whose columns are ports); every candidate is replayed on that table and candidates that contradict it are dropped, provided one survives. One timing convention, fixed once: at each row

the clock column is applied first, so an edge samples the previous row's data; data inputs are applied next; outputs are sampled after both; `x` entries are not checked. Tables that describe a *sub*-module ("Module B can be described by the following waveform … now consider a top-level module") are skipped. *(iii) Behavior gate:* survivors are simulated on the harness's stimulus and on one probe run — 200 cycles of "sticky" inputs (each holds its value with probability 0.8, producing the long runs that framing and sequence detectors need) and 200 uniform cycles with mid-sequence reset pulses, outputs sampled after the inputs settle, after the falling edge and after the rising edge, plus the warm-up; designs with at most 8 input bits also get their exhaustive truth table. Candidates with identical outputs form one *behavior*. If one behavior remains it is returned and *no judge is called*; otherwise the verifier chooses among the survivors exactly as before, and the harness returns the judge's own pick.

Validation of the replay. The convention reproduces $12/12$ reference designs' own spec tables. (The reference was used only for this one check of the convention, never at selection; two parser guards — sub-module tables, and header aliases and hexadecimal tables — were added after it.) Counting distinct candidates on those 12 waveform problems across all three seeds' pools gives 107 testbench-correct designs; $107/107$ **testbench-correct candidates pass the replay** — none of the audited correct candidates is rejected — while of 105 incorrect ones 97 contradict the table, 7 do not run and 1 passes.

Development and freezing. The tools were developed on seed 1 and frozen before seeds 2–3 were scored. Two fixes came from that development set and are general, not problem-specific: sampling only after the rising edge hid a level-sensitive latch that imitated a dual-edge flip-flop, and uninitialized-register differences were invisible until the warm-up was sampled.

Table 9: **Execution tools improve the cross-verifier on VerilogEval-v2; bounds using recorded Claude verdicts remain conditional.** VerilogEval-v2 accuracy (%). Bounds assume stable recorded verdicts after removing unselected candidates and are not confidence intervals. The gate's call counts do not establish a cost saving.

	seed 1	seed 2	seed 3	mean
cross-verifier	87.2	84.0	84.6	85.3
+ execution tools	**88.5**	**85.9**	**85.9**	**86.8**
Claude judge	91.7	91.0	89.7	90.8
+ execution tools	[**93.6**, 94.2]	[90.4, 92.3]	[90.4, 92.3]	[91.5, 92.9]
pool-family judge (`deepseek-chat`)	86.5	—	—	
+ execution tools	**89.7**	—	—	
judged only: −/+ evidence[†]	42/37	41/37	32/34	115/108
problems where the judge is needed	52	51	51	of 156
pool oracle / best-behavior ceiling	147/146	145/143	144/144	

Reporting details. Execution tools on VerilogEval-v2, the selector otherwise unchanged. Cross-verifier rows are exact on every seed and cost no model call. Claude-judge rows compose the tools with the judge's *recorded* verdicts: bounds that are tight to one problem where per-problem verdicts were retained (seed 1: one recorded pick is filtered out and would need a new verdict), and wider where only the aggregate and the run log survived (seeds 2–3). These are conditional accounting bounds, not confidence intervals. Both assume a verdict is unchanged when candidates the judge did *not* pick are removed from its input (exercised on 11 seed-1 problems). "Judge needed" counts problems left with more than one behavior. [†]Correct picks on those judged problems only, same candidates and prompt, without/with the execution evidence appended.

What the tools change. On VerilogEval-v2 the tools improve the cross-verifier on every seed and lower no problem on any seed (85.3 → 86.8; Table 9). On the Claude judge's seed-1 verdicts they fix three problems and break none (91.7 → 93.6; `circuit7`, `circuit9`, `fsm_ps2data`, all by replay: in each the judge had chosen a design the spec's own table refutes), while the judge is consulted on 52 problems instead of 155 (judge calls 465 → 156). One further seed-1 problem (`Prob089`) is left open: the judge had picked a candidate that does not compile, the filter removes it, and the survivors include both correct and incorrect designs in different behaviors, so a new verdict

would be needed; seed 1 is therefore $[93.6, 94.2]$, not a point. Against the matched one-shot draw this is 8 fixes and 3 breaks, exact McNemar $p = 0.227$: a point estimate, not a significant win. On seeds 2–3 the bounds say the tools can lower the Claude judge by at most one problem on any seed (seed 2, `Prob078`: a single-behavior group whose representative is wrong while a member is right — which also means seed 2 can reach at most 144 of its 145 oracle-solvable problems, the upper end of its interval); the seed-3 lower bound uses its log's record that the judge chose wrongly on `circuit9`, which replay fixes deterministically. On the frontier model's own $K{=}3$ pool the gate cuts judge calls $225 \rightarrow 39$ at unchanged accuracy (92.9). Calls are not cost: the problems the gate skips are the easy, cheap ones, so verification cost falls by less than the call count, and we did not measure by how much.

On specifications the tools were not developed on. We ran the frozen tools on RTLLM-v2 (48 clean designs, three seeds), composing them with the frontier judge's recorded verdicts for all three seeds; the per-problem records reproduce the published cross-verifier, Claude-judge and oracle means (72.2, 76.4, 81.2) to within 0.1pp. No RTLLM-v2 spec prints a waveform, so only the compile filter and the gate act. As frozen, the tools *hurt*: the Claude-judge arm fell to $[74.3, 76.4]$, because the executor inherited ChipMATE's rule of simulating the *first* module in a file, and RTLLM-v2 candidates often define sub-modules before their top module — so the filter simulated a sub-module against the top's ports and discarded correct designs (`adder_32bit`, `barrel_shifter`). Selecting the module by the name the spec declares, as the official scorer does, changes none of the 4,665 VerilogEval-v2 candidates, so every VerilogEval-v2 number above stands; on RTLLM-v2 the tools then leave the Claude-judge arm at exactly 76.4 ($-1/+1/0$ problems by seed; the loss is a probe-indistinguishable pair, the gain a non-compiling pick) and the cross-verifier at 72.2, while cutting judge calls by a third ($432 \rightarrow 288$). This fix was made *after* seeing RTLLM-v2, so RTLLM-v2 is no longer a held-out test of the fixed tools. The same first-module rule sits inside the cross-verifier's V↔P simulation; it touches 21 RTLLM-v2 candidates and changes which candidate the cross-verifier picks on eight problem-seeds, but not whether the pick is correct, so no published number moves. On unseen specs, then, the frozen tools were not safe; fixed, they were accuracy-neutral in aggregate (one problem lost, one gained) and saved a third of the judge calls; and their accuracy gain did not transfer, being specific to specs that print their own behavior.

A frontier-free configuration, metered. The tools make a *cheap* judge usable, so we ran the whole harness with no frontier model in it: the cheap pool, the tools, and `deepseek-chat` (DeepSeek-V4.1-Flash) as the judge, consulted only on problems left with more than one behavior, seeing the surviving candidates under the paper's selector prompt, 3 votes. Every call's token usage is stored on its problem's record. The output budget is 16k: the same configuration at 8k (the no-evidence arm of the ablation below) lost 13 seed-3 problems to votes cut off mid-deliberation and scored 88.5, and we raised the budget for that reason before this run, not after it. **Result:** $144/137/139$**,** $\mathbf{89.7 \pm 2.3}$, at **\$**0.022 **per problem**: \$0.0085 metered for the judge, priced at the \$0.27/\$1.10 per M rate this paper uses for DeepSeek (the billed balance moved by \$3.69 for the three seeds), plus \$0.0135 for the nine candidates. That generation figure is itself re-metered: the frozen pool kept no usage records and the paper's earlier cost model priced each candidate by the length of its returned Verilog, which omits `gpt-5-mini`'s hidden reasoning. Re-running the pool's own `gpt-5-mini` call (same prompt, 8k budget) on 40 random problems bills \$0.00324 per call, 1,312 of its 1,562 output tokens being reasoning (it scores 90.0% on that sample, against the member's 84.1 on all 156); a DeepSeek call bills \$0.00063 (the 468 one-shot calls above, the current alias standing in for the June DeepSeek names). Nine candidates are therefore \$0.0135, not the \$0.0048 we had modeled, and matched Claude one-shot is \$0.028 metered from its retained usage, not \$0.024. The judge on its own is a poor generator: its one-shot generation under the matched prompt scores 75.6 ± 2.6 (three draws, below the pool's own `gpt-5-mini`, 84.1), so this is not a strong model in a cheap model's clothing. Against the references: 1.5pp below frontier one-shot (91.2; pooled paired fixes 18, breaks 25, McNemar $p = 0.36$, which treats the three seeds' evaluations of one specification as independent) at $0.78\times$ its cost; below the Claude-judge configuration with the same tools ($[91.5, 92.9]$, i.e. by 1.8–3.2pp) at most $0.16\times$ its cost; and above the randomized generation/verification mixture of equal cost on every seed ($+3.7/+0.4/+1.8$pp, mean $+1.9$, t-test $p = 0.18$), or $+1.1$ on average ($+3.0, -0.7, +1.0$) against the stronger mixture whose cheap end also uses the tools. Neither margin is significant, and 123 of the 462 judge calls still reached the 16k budget. The decisive comparison is a baseline derivable from numbers already in the paper: a price-matched mix of the two one-shot baselines (Table 7). It reaches 89.5 at the frontier-free price, 0.3pp below the frontier-free harness,

and 88.1 at the cross-verifier configuration's price, above both cross-verifier rows. The frontier-free configuration dominates the routed point and the cross-verifier mixture of Table 7, edges the one-shot mixture by 0.3pp, and is itself dominated by its one-per-behavior variant ($0.019, 89.7; Appendix Q). We therefore claim no demonstrated cost advantage for any cheap configuration. Two caveats: the judge model is newer than the pool's June DeepSeek members, and the dollar figures depend on the stated price for it.

What the tools cannot do. Two candidates can agree on every probe and still be separated by T (e.g. `Prob151` on seed 1, a pattern-recognizing sequential design: no probe produced an input history on which the two differ). That is why the harness returns the judge's own pick whenever it is consulted, and why the best-behavior ceiling sits 0–2 problems below the oracle. Replay applies only where a spec prints its behavior: 12 of 156 VerilogEval-v2 problems and none of the RTLLM-v2 specs.

Appended evidence did not raise the cheap judge's accuracy; decisions taken before it did. One pool-family judge (`deepseek-chat`, an API alias that now resolves to DeepSeek-V4.1-Flash; 3 votes, 8k-token budget). *Decisions taken away.* Choosing from all candidates with the paper's selector prompt it scores 86.5 on seed 1; the same verdicts with the tools applied score $\mathbf{89.7}$ ($+3.2$pp: four replay fixes and one compile-filter fix, no breaks), above the cross-verifier (87.2). *Evidence fed in, matched.* On every problem the tools leave with more than one behavior ($52/51/51$), the judge sees the same surviving candidates in the same order under the same prompt, votes and fallback, and one arm additionally receives an EXECUTION EVIDENCE block (which candidates share a behavior, and the input windows on which the behaviors disagree, with each behavior's outputs). Appending it does not help: 115 versus 108 correct without and with it over the 154 judged problem-seeds ($42/37$, $41/37$, $32/34$ per seed); the evidence fixes 5 and breaks 12 (exact McNemar $p = 0.14$). The sign is negative on two seeds and positive on the third, where the arm without evidence lost more problems to the token budget (13 versus 5 fallbacks; the judge was still deliberating when cut off). The ablation ran at an 8k budget, so it does not establish what evidence would do at the 16k budget of the frontier-free run above; the judge's one-shot control also ran at 8k, where only 5 of its 468 calls reached the cap. An earlier, unmatched variant — one implementation per behavior with the evidence prompt, rotated order, 8k budget — scored 85.3; under the original selector prompt at 16k, one implementation per behavior scores 89.7 (Appendix Q). With experiment (x) (a frontier judge that designs its own tests: 83.3 against 87.2 under the same nominate-all protocol without tools, 87.8 plain) and Appendix S, item 3 (the frontier judge handed the candidates' divergent simulated outputs: 91.0 against 91.7), no judge we tried, two of them frontier, was helped by execution routed *through* its reading, while the same execution used to take decisions *away* from it helped both the cross-verifier and this judge. We state that as the design rule the harness follows, not as a law: the matched test is one judge, not significant on its own, and the positive side rests on 12 waveform specs.

A failed measurement, disclosed. To replace the bounds with a fresh draw we re-ran the gated frontier judge on seed 2. It is not used: we gave the judge a 4,096-token output budget, believing that matched the original runs, but those had none (the client silently dropped the parameter), and 29% of votes were truncated before their answer, forcing a fallback on 12 of the 51 judged problems. It is the same starved-reasoning failure as the `gpt-5` cap in experiment (ix).

Q A CONTROLLED DIAGNOSTIC FINDS A FALSE PREMISE IN THE SELECTOR PROMPT

VERDICT's selector prompt, shared by every judge arm in this paper except those listed next, asserts that exactly one candidate is correct: "Exactly identify which candidate CORRECTLY implements the spec (others contain bugs: ...)". The exceptions are the CVDP-CID003 Claude judge, whose separate prompt reads "Exactly one (or more) correctly implements the spec" (Appendix B); the evidence-fed Claude judge of Appendix S, item 3, whose prompt drops the premise sentence and asks for the candidate most consistent with the spec-correct behavior; the nominate-all `gpt-5` protocols and the `qwen-2.5-coder-32b` judge's elimination and nominate-all variants of experiments (ix)–(x) (Appendix O), which allow several correct candidates; and the premise-free prompts tested below. The HumanEval judge (Appendix R) states the premise in its own words ("Exactly one family of

them is correct"). We built controlled pools to test whether many copies of one wrong behavior sway a judge. No such majority-following was detected; what the pools exposed instead is that on those pools a cheap judge was far more likely to exhaust its output budget when several candidates were correct, and on the three-correct pools a rewrite of the premise sentence restored its answers. A pre-registered confirmation on 30 unseen problems (last paragraphs) failed its primary test: the lost answers did not recur, and only exploratory token contrasts reproduced the extra output.

Design (an offline diagnostic, not part of the harness). Six files — the three cheap-pool seeds, the matched Claude one-shot draws, Claude's $K{=}3$ pool and the DeepSeek one-shot draws — give 3,544 text-distinct candidates on the 156 VerilogEval-v2 problems. Here a *behavior* is a set of candidates with identical outputs on the harness's probes *and* the same official-testbench verdict: the diagnostic uses testbench verdicts to build pools and to score votes, never shows them to a judge, and is not part of VERDICT, which reads the testbench only to score its final pick. 41 problems have a correct and a wrong behavior with at least three text-distinct members each (31 clocked designs, 4 combinational FSM next-state functions, 6 other combinational functions); they are selected for disagreement and do not represent the benchmark. From each problem's most frequent correct and wrong behavior we draw c_1, c_2, c_3 and w_1, w_2, w_3 and form four pools (Table 10). Every judge sees the paper's selector prompt verbatim with a 16k output budget; candidate order is randomized per vote, seeded by problem and vote index. Contrasts are paired over the 41 problems (t-test; Wilcoxon agrees).

Table 10: **The discovery effect on answers does not imply a gain in final accuracy.** Controlled pools vary correct multiplicity; real pools test representation and prompt changes. Rates are percentages. Definitions, budgets, and fallback accounting follow.

Controlled pools (41 problems)	W3	B2	R3	REP
DeepSeek C/A	68.3/88.6	61.0/80.5	59.3/69.1	**83.7/95.9**
budget	17/123	36/123	48/123	5/123
decision	80.5	75.6	87.8	87.8
Qwen3-14B C/A	53.7/97.6	49.6/93.5	71.5/94.3	68.3/99.2
budget	3/123	8/123	7/123	1/123
Claude C/A	84.6/98.8	—	84.1/93.9	87.8/100.0
budget	1/88	—	6/88	0/88
R3, first vote only, same pools and order	answered / correct (of 41)		mean output tokens	
original prompt	27 / 23		10,141	
premise-free prompt	**39 / 37**		3,026	
Real pools, 3 seeds	accuracy		judge $/problem	budget
original prompt, all survivors	420/468 (89.7)		0.0085	123/462
original prompt, one per behavior	420/468 (89.7)		0.0055	58/462
premise-free prompt, all survivors	416/468 (88.9)		**0.0042**	**24/462**

Reporting details. The controlled diagnostic, the premise test, and the real pools. Pools: W3 = wrong majority $\{c_1, w_1, w_2, w_3\}$; B2 = balanced $\{c_1, c_2, w_1, w_2\}$; R3 = right majority $\{c_1, c_2, c_3, w_1\}$; REP = one implementation per behavior $\{c_1, w_1\}$. C/A: problem-mean per-vote correct / answered rate (%), an unanswered vote counting as incorrect. "Budget": votes that reached the 16k output budget. Votes per pool: DeepSeek-V4.1-Flash $3{\times}41{=}123$; Qwen3-14B (local) 123; Claude-Opus-4.8 88 (two votes, three on six pilot problems; not run on B2). *Decision*: share of the 41 problems whose final pick is correct under the execution-tools harness's rule — plurality of answered votes over candidates, ties to the earliest vote; with no answered vote, the pool's larger behavior (a 2–2 tie in B2 counts 0.5). Real pools: frontier-free configuration (cheap pool, execution tools, DeepSeek judge), three generation seeds; accuracy over 468 problem-seeds, budget over the 462 judge votes, judge cost averaged over all 468 at \$0.27/\$1.10 per M tokens.

What the controlled pools show. For DeepSeek the answered rate falls as correct copies are added (88.6 → 69.1, R3−W3 −19.5pp, $p = 0.0064$), while three redundant *wrong* copies leave it high (W3). Every one of its 81 unanswered votes reached the 16k budget; the stored endings show it hunting for the bug the prompt says must exist ("either the problem has two correct answers, or I'm wrong about one of them"). Showing one implementation per behavior lifts its per-vote correct rate by 15.4pp over W3 ($p = 0.0034$) and 24.4pp over R3 ($p = 0.0010$), but this is accounted for

by answering: among answered votes REP and R3 differ by -0.5pp (34 problems, $p = 0.91$, 95% CI $[-9.3, +8.3]$). REP also halves the number of candidates and the prompt, so its gain cannot be attributed to presentation alone. Scored with the harness's decision rule, R3 and REP both reach 87.8, but for different reasons: REP's 36 correct decisions all come from answered votes, while R3's comprise 29 from answered votes and 7 from problems with no answered vote, which the rule sends to the larger behavior — correct by construction in R3 (and wrong by construction in W3). Voting alone does not explain the equality. Majority-following is not detected: rescaling answered accuracy as $(a - s)/(1 - s)$, with s the share of correct candidates, R3$-$W3 is -4.9pp for DeepSeek (32 problems, $p = 0.72$, 95% CI $[-32, +22]$) and -32.2pp for Qwen3 (40 problems, $p = 0.10$) — the raw rise of Qwen3's accuracy with the correct share ($+17.9$pp, $p = 0.0033$) is what guessing among candidates predicts — so the test cannot exclude it either. Qwen3, a reasoning model, almost always finishes; Claude's changes are not detected (95% CIs about ± 11pp), though its answered rate moves the same way ($98.8 \rightarrow 93.9$).

A rewrite of the premise sentence. On the same R3 pools and the same first-vote candidate order, we rewrite only that sentence: several candidates may be correct, correct candidates may differ only in coding style, and any correct one is acceptable. The rewrite removes the premise and also adds a style hint. DeepSeek answers 39 of 41 instead of 27 (12 gained, 0 lost, exact McNemar $p = 0.0005$), is correct on 37 instead of 23, and writes 70% fewer tokens. Qwen3, given the rewrite on the R3 and W3 pools with the same three vote orders (246 votes), answers slightly more often (W3 $97.6 \rightarrow 100$, R3 $94.3 \rightarrow 99.2\%$; $p = 0.08$ each) and writes 16–24% fewer tokens; its per-vote correct rate moves $+5.7$pp on R3 ($p = 0.15$) and -6.5pp on W3 ($p = 0.17$), neither detectable.

On real pools. Both remedies change the cheap judge's behavior and cost, not its end-to-end accuracy. On the 154 judged problem-seeds, one implementation per behavior raises the per-vote correct rate from 64.1 to 72.5% ($p = 0.0006$) and the answered rate from 79.4 to 90.7%, with final accuracy unchanged (8 fixes, 8 breaks; one break is a probe group whose representative is wrong while a member is right). The premise-free prompt lowers the share of judge votes reaching the budget from 26.6% to 5.2% and halves the judge's cost per problem (about 20% of the configuration's total), with no detectable change in accuracy (-0.9pp; 6 fixes, 10 breaks, exact McNemar $p = 0.45$). Both nulls are consistent with majority voting absorbing truncated votes (only 6 of 154 judged problem-seeds had no answered vote), though no run switches voting off.

Caveats. Under REP DeepSeek trails Claude by 4.1pp (95% CI $[-13.8, +5.7]$), so we do not claim the two judges are equivalent; the gap concentrates on the 13 problems whose c_1 Claude wrote (-19.2pp vs. $+3.0$pp on the other 28; post hoc, uncorrected $p = 0.02$), a pattern compatible with self-preference that difficulty could also produce. DeepSeek's REP$-$R3 gain exceeds Claude's by 20.7pp ($p = 0.0093$), which does not survive Holm correction over the 14 contrasts we report. On 6 of the 41 problems the real gate would not call the judge (spec replay removes the wrong behavior on 5; on 1 the two behaviors give identical probe outputs); on the other 35, REP still beats W3 by 16.2pp ($p = 0.008$) and R3 by 24.8pp ($p = 0.003$). Unlike the diagnostic, the real gate groups by probe outputs alone and picks representatives by a fixed family order. Every frontier-judge arm in this paper outside the exceptions listed at the start of this appendix used the same premise-laden prompt; we did not re-run them without it.

A pre-registered confirmation on unseen problems. The discovery pools were chosen after looking at candidates, so before any further judge call we wrote down a confirmation protocol and recorded its SHA-256 (`7ef68934...f148a5`; the protocol is in the supplementary material). Problems: every VerilogEval-v2 problem outside the 41 and every clean RTLLM-v2 problem with at least two text-distinct correct and two wrong candidates, $13 + 17 = 30$. Pools of three, seeded per problem: W2 $= \{c_1, w_1, w_2\}$ (one correct) and R2 $= \{c_1, c_2, w_1\}$ (two). Three prompts: P0 the original; P1 with the premise sentence replaced ("more than one candidate may be correct; any correct one is an acceptable answer"); P2 the rewrite above, which the protocol describes as P1 plus the style hint but which also words the acceptability clause differently. Three votes per cell, candidate order shared across prompts, 16k budget. The registered primary test asked whether removing the premise recovers more answers on R2 than on W2, $D = [A(\mathrm{R2}, \mathrm{P1}) - A(\mathrm{W2}, \mathrm{P1})] - [A(\mathrm{R2}, \mathrm{P0}) - A(\mathrm{W2}, \mathrm{P0})]$ with A the answered rate under the harness's parser (Table 11). One deviation: an

accidental concurrent restart issued 125 of the 540 cells twice (665 calls); we keep the first row written per cell, and D is identical whichever copy is kept.

Table 11: **The registered answer-rate interaction fails confirmation.** 30 unseen problems (13 VerilogEval-v2, 17 RTLLM-v2); A/C denotes answered/per-vote correct rate (%). Token reductions and the secondary accuracy contrast do not change the failed primary result. Protocol details follow.

	A/C		tokens		budget		decision	
Prompt	W2	R2	W2	R2	W2	R2	W2	R2
DeepSeek-V4.1-Flash (primary judge)								
P0 original	96.7/82.2	97.8/95.6	3,244	4,487	7	12	90.0	96.7
P1 premise removed	98.9/80.0	100.0/93.3	2,563	1,399	3	0	86.7	96.7
P2 rewrite	96.7/74.4	95.6/92.2	2,148	1,533	2	1	80.0	96.7
Qwen3-14B, local (secondary judge)								
P0 original	98.9/64.4	94.4/71.1	5,726	7,803	1	5	66.7	76.7
P1 premise removed	96.7/62.2	97.8/82.2	5,192	5,966	3	2	63.3	90.0
P2 rewrite	98.9/63.3	98.9/76.7	4,780	5,447	1	1	63.3	73.3

Reporting details. Pre-registered confirmation, 30 **unseen problems (**13 **VerilogEval-v2,** 17 **RTLLM-v2).** A/C: problem-mean answered / per-vote correct rate (%), answers parsed as in the harness. Tokens: mean output tokens per vote; budget: votes reaching the 16k budget, of 90. Decision: problems (% of 30) whose final pick is correct under the harness rule (no problem needed the no-answer fallback). 540 votes per judge: DeepSeek-V4.1-Flash ($2.09 metered over all 665 calls) and Qwen3-14B (local). Differences in the text are computed before rounding.

The primary test failed: $D = 0.0$pp (95% CI $[-4.6, +4.6]$; t and Wilcoxon $p = 1.0$), which by the protocol's own criterion (mean $D \leq 0$) counts against the premise account of non-response. Under the original prompt DeepSeek answered 96.7% of W2 votes and 97.8% of R2 votes (+1.1pp, $[-2.9, +5.1]$, against the discovery pools' -19.5pp). The harness's parser takes the last `ANSWER` line anywhere in the output, as in the discovery study, so 16 votes that stopped at the budget still count as answered (10 of them R2/P0). Counting those as unanswered, a reading we chose after seeing the data, gives $D = +8.9$pp ($[-0.9, +18.7]$, $p = 0.07$): the predicted direction, short of the registered $\alpha = 0.05$. Registered secondaries: removing the premise changed R2 answering by +2.2pp ($p = 0.16$); the rewrite's extra wording lowered it by 4.4pp relative to P1 ($p = 0.043$, uncorrected); R2's per-vote correct rate moved -2.2pp (P1$-$P0, $p = 0.33$). The extra work did reproduce (the protocol registered output tokens as a secondary outcome; the contrasts and p-values that follow are exploratory): with the premise, two correct candidates cost the judge 1,244 more output tokens than one (paired t $p = 0.098$; Wilcoxon $p = 0.025$), and 12 of 90 R2 votes reached the budget against 7 of 90 W2 votes; without it R2 costs 69% less (4,487 $\rightarrow$ 1,399 tokens, no vote at the budget) and the R2$-$W2 difference reverses (interaction $-2{,}408$ tokens, $p < 0.001$). Accuracy did not improve, and the point estimates are negative, as on real pools: the per-vote correct rate moves -2.2pp (P1$-$P0, pooled over pools, $p = 0.49$), and on one-correct pools the decision falls 90.0 $\rightarrow$ 86.7 $\rightarrow$ 80.0% (P0 to P2: 1 fixed, 4 broken, $p = 0.38$). The secondary judge, Qwen3-14B, fails the primary test too ($D = +5.6$pp, $[-3.7, +14.8]$, $p = 0.23$), and shows the extra work more strongly: with the premise, two correct candidates cost it 2,078 more output tokens than one ($p < 0.001$), and removing the premise cuts R2's output by 1,838 tokens ($p < 0.001$; interaction $-1{,}304$, $p = 0.018$). Unlike DeepSeek, removing it also raises Qwen3's R2 per-vote correct rate by 11.1pp ($[+2.3, +20.0]$, $p = 0.016$) and its R2 decisions 76.7 $\rightarrow$ 90.0% (4 fixed, 0 broken, $p = 0.13$), with no change on W2 (-2.2pp). This is a registered secondary contrast, uncorrected, and it would not survive correction over the six such contrasts of the two judges; the P2 rewrite does not share the gain (+5.6pp, $p = 0.36$).

We therefore narrow the claim. The confirmation detected no effect of the premise on answering at the registered level, for either judge. In exploratory contrasts it made two-correct pools cost both cheap judges more output, and removing it shortened that output; we have no corrected evidence that it costs accuracy (Qwen3 showed an uncorrected secondary R2 gain, DeepSeek none). Lost answers were observed only on the discovery pools, whose four candidates with three correct copies, on problems selected for disagreement, differ from these in both pool and problem set; we have not shown that the 27 $\rightarrow$ 39 of 41 result extends beyond those pools, nor that it is due to the premise

Table 12: **The pool with the Claude judge ties one-shot on HumanEval and leads by one problem on HumanEval+.** Single-seed experiment, 164 problems, with execution-based cross-verification; HumanEval+ uses conservative repr-equality scoring. This does not establish a generalization trend or a cost advantage.

Benchmark	strongest member (one-shot)	cross-verifier (ours)	Claude judge (ours)	oracle (pass@any)	Claude-Opus one-shot
HumanEval (164, Python)	95.1	92.1	**96.3**	98.8	96.3
HumanEval+ (164, harder)	92.1	89.0	**93.3**	95.1	92.7

rather than the accompanying style hint. We did not run the protocol's Claude arm: it was conditional on budget, and Claude answered 94–100% of votes on the discovery pools.

R Cross-domain experiments: Python and general code

We apply the generation-and-selection workflow to HumanEval (164 Python problems, single-seed experiment), replacing RTL simulation with execution on model-generated inputs. The cheap pool's oracle is 98.8%, frontier one-shot 96.3%, and cross-verifier accuracy 92.1%, below the 95.1% floor pick (Table 12). The Claude judge adds +4.2pp and ties one-shot's observed 96.3%. The pool's generation alone costs approximately 5× less under returned-code pricing, which omits hidden reasoning tokens (Appendix P re-meters the Verilog pool at about half a frontier call once they are priced); this is not a measured saving for the full selection pipeline. All rows are near saturation (92–99%).

Re-scoring the same candidates with HumanEval+ (EvalPlus (Liu et al., 2023a), approximately 120 extra inputs per problem, conservative repr-equality scoring) gives oracle 95.1%, frontier one-shot 92.7%, cross-verifier 89.0%, and floor 92.1%. The Claude judge reaches 93.3%, one problem above one-shot. This small experiment does not establish that selector advantage grows as benchmark saturation falls.

HumanEval is near-saturated, so we replicate on a harder, un-compressed general-code benchmark: the LiveCodeBench (Jain et al., 2025) *test6* increment (112 problems; Table 13 gives the contest window and test counts). Here the two facts underlying the paper's accounting both replicate outside RTL. First, the cheap pool's oracle exceeds Claude one-shot (84.9 vs. 80.9, margin +4.0pp, on the 93 of 112 problems with at least one non-empty frontier draw; f is the pass rate pooled over the 267 non-empty draws of three per problem, $216/267$, whereas every other row is a per-problem count; the first draw alone gives $73/93 = 78.5\%$; the other 19 are excluded, 16 hard and 3 medium, on which the pool oracle is $9/19$) on this subset. The contest window may overlap the pretraining of the frontier model and of the pool members alike, so contamination could move this margin either way; we claim no direction. Second, the selection loss replicates almost exactly: pool-internal cross-verification captures only 76.3% (71/93), an ℓ=8.6pp gap below the oracle, against Verilog's 7.9–9.0pp in a different verification modality (execution-agreement vs. V↔P simulation). And the margin is not uniform — it is +0.0/+3.4/+7.7pp across easy/medium/hard problems, a concentration toward harder problems, consistent with the sequential-design pattern in RTL (§3). The one row we do not measure on LCB is the Claude judge's selected accuracy (the frontier selector over the same pool, beyond our frontier-call budget); the two quantities the accounting needs — the oracle margin and the cross-verifier loss — do not depend on it. The Claude judge recovers part of this same selection loss on VerilogEval-v2, RTLLM-v2 and HumanEval, and none of it on CVDP-CID003 (§6).

Reporting details. The margin and loss recur on general code (LiveCodeBench). Same cheap pool (`gpt-5-mini+deepseek-chat+deepseek-coder`, K=3, 9 candidates; two families, the two DeepSeek names probably one model, Appendix B), execution-agreement cross-verification on model-generated inputs. LCB *test6* increment (112 stdin/AtCoder problems, contests 2025-01-04→2025-04-06; up to 40 hidden tests each, 40 for 103 of the 112; a program passes only if it passes every public and hidden test). Left: the selector rows on the f-subset (the n=93 problems with at least one non-empty frontier draw). Two of the paper's measured facts recur: the cheap pool's oracle *exceeds* frontier one-shot (84.9 > 80.9, margin +4.0), and the pool's own cross-verifier leaves a selection loss of ℓ=8.6pp, close to Verilog's 7.9–9.0pp, in a different domain and a different

Table 13: **LiveCodeBench reproduces an oracle margin and cross-verifier loss.** The 93 of 112 problems are those with at least one non-empty frontier draw; the 19 excluded (13 never sampled after the API budget ran out, 6 with three empty draws) are 16 hard and 3 medium problems on which the pool oracle is $9/19$, so the subset is easier than the full window (on all 112: oracle $88/112 = 78.6\%$, cross-verifier $77/112 = 68.8\%$, loss $\ell = 9.8$pp). The Claude-judge row is unmeasured; possible contamination can affect either side of the margin. Differences use unrounded values.

Selector (f-subset, n=93)		**Margin by difficulty**				
Configuration	pass@1 (%)	Difficulty	n	o	$f^{\ddagger}$	$o-f$
floor pick (first `gpt-5-mini` draw)	67.7	easy	26	100.0	100.0	+0.0
cross-verifier (ours)	76.3	medium	23	91.3	87.9	+3.4
Claude judge (ours)	–[†]	hard	44	72.7	65.0	**+7.7**
oracle (pass@any)	**84.9**					
Claude-Opus one-shot (3 draws, pooled)[‡]	80.9					

† The Claude judge (a frontier model as selector over the same pool) is the one row not measured on LCB: the 3-vote pass over the slice exceeded our frontier-call budget. The two quantities this table reports, the oracle margin $o>f$ and the cross-verifier's selection loss ℓ, do not depend on it. On VerilogEval-v2, RTLLM-v2 and HumanEval the Claude judge recovers part of that loss (5.6 of 7.9pp, 4.2 of 9.0pp, 4.2 of 6.7pp) and on CVDP-CID003 none; whether it recovers any of it on LCB is left open, so this table cannot show that the selector substitution transfers to LCB, only that the margin and the cross-verifier's loss do. ‡ Every oracle, cross-verifier and floor-pick entry is a per-problem count over the n problems shown (oracle $79/93$; cross-verifier $71/93$; the floor pick is the first draw of `gpt-5-mini`, $63/93$; by difficulty the oracle is $26/26$, $21/23$, $32/44$). The frontier one-shot f was sampled three times per problem and is the pass rate pooled over the non-empty draws: $216/267$ overall and $78/78$, $58/66$, $80/123$ by difficulty; 12 of the 279 draws ($0/3/9$ by difficulty) returned no program and are excluded, so f is not a count over 93 problems and the difficulty rows do not n-average to the overall row. The three per-draw rates are $79.3/82.0/81.4\%$. Counting the empty draws as failures gives $216/279 = 77.4\%$, and the first draw alone gives $73/93 = 78.5\%$; the oracle exceeds f under every convention (margin +4.0 pooled, +6.5 first draw, +7.5 empties-as-failures). A draw or candidate passes only if it passes every public and hidden test; there are no fractional scores and no weighting.

verification modality. Right: the margin is concentrated on hard problems ($+0.0/+3.4/+7.7$ by difficulty), the same shape as Verilog's combinational-vs-sequential split. The contest window may overlap the pretraining of the frontier model and of the pool members alike, so contamination could move the margin in either direction.

S LIMITATIONS

1. **The cheap-pool comparison is a statistical tie, not a win.** Over three independent generation seeds it reaches 90.8 ± 1.0 (per-seed $91.7/91.0/89.7$) against a matched-prompt Claude one-shot baseline of 91.2 ± 1.0 (per-draw $90.4/92.3/91.0$; Appendix M) — nominally 0.4pp *below*, within noise on both sides. The paired comparison cannot separate the two arms (item 2), and the tie is bought at at least $5.0\times$ the cost of one frontier generation call. With the execution tools of Appendix P the 3-seed mean lies in $[91.5, 92.9]$ — $[93.6, 94.2]$ on seed 1, wider on seeds 2–3 whose per-problem verdicts were not retained — which is recorded-verdict accounting never run fresh, and still not a significant win (seed-1 McNemar $p = 0.227$). The results that repeat across seeds are the Claude-judge gain over the cross-verifier ($+5.6\pm1.3$pp) and the margin and loss themselves (oracle $>$ one-shot, cross-verifier $\ll$ oracle) on every seed. ChipBench/CVDP: the Claude-judge row is 3-/2-seed (Table 5); the other rows are seed-1.
2. **The Claude-judge/one-shot comparison is a statistical tie, not an established equivalence.** On VerilogEval-v2 the paired exact McNemar tests of Table 5 (the Claude judge's best seed against the three matched one-shot draws) give $p = 0.79/1.00/1.00$ and a paired 95% interval of $[-3.4, +6.0]$pp. That interval admits a 3pp loss and a 6pp win, so it supports neither the old +1.1pp claim nor an equivalence claim, and we make neither. The Claude-judge gain over the cross-verifier is reproducible in sign across seeds ($+5.6\pm1.3$pp, t-test p=0.019) but is itself imprecise within seed 1 (paired 95% interval $[0.0, +9.0]$pp on seed-1).
3. **The Claude judge leaves 2.4pp of coverage unselected, and tuning it did not recover it.** The Claude judge captures 90.8 of the 93.2 oracle on VerilogEval-v2 (3-seed); a perfect Claude/cross-

verifier router caps at 93.6 (146/156, seed-1: the union of the 143 problems the Claude judge returns and the 136 the cross-verifier returns; one problem with a correct candidate, `fsm_ps2data`, is returned by neither, so the router cap sits 0.6pp below the 94.2 oracle). Reaching the oracle would require selecting correctly whenever a correct candidate is available; these experiments do not identify a sufficient judge property. The residual did not respond to the tuning we tried: increasing the judge's vote count $3 \rightarrow 5$ does not help (seed-1 $91.7 \rightarrow 91.0$), and neither does grounding the judge in behavioral evidence (feeding it the candidates' divergent simulated outputs on concrete stimuli: 91.0, fallbacks 0) — three protocols cluster tightly at 91 ± 0.5. The 3- and 5-vote protocols kept the selector prompt's single-correct premise; the evidence-fed prompt drops that sentence (Appendix Q); the errors we inspected are mostly confident misselections. What does move it is taking decisions away from the judge: the execution tools of Appendix P fix three seed-1 misselections, each a design the spec's own printed waveform refutes ($91.7 \rightarrow 93.6$).

4. **The Claude-judge gain does not reproduce on the second frontier judge we tested, and this is now measured rather than open.** Every Claude-judge number uses Claude-Opus-4.8. We re-ran the identical 3-vote protocol with `gpt-5` as the judge over the identical stored seed-1 pool (24k reasoning budget; 144/156 problems before a spend cap, 2 unparsable votes, no fallbacks). Scored on those same 144 problems for all three arms: cross-verifier 91.7, **`gpt-5`** 90.3, Claude 93.8, oracle 95.8. **The second frontier judge does not reproduce the gain**: -1.4pp on this 144-problem 3-vote run (-2 problems, p=0.73; $+0.6$pp on the plain-prompt run over all 156, experiment (ix)), where Claude is $+2.1$pp on the same candidates; head to head Claude fixes 6 of its picks and breaks 1 (McNemar p=0.13), and `gpt-5` against the cross-verifier is 3 fixed, 5 broken (p=0.73). Three pre-specified diagnostics say what is going on, and the first weakens the explanation we reached for first.

   Family preference does not explain it, though this does not settle the mechanism. `gpt-5` shares a family with `gpt-5-mini`, the pool's strongest member, so the obvious reading is that it inherits its blind spots. If that produced a visible preference for its own family's candidates, it does not: `gpt-5` selects an OpenAI-family candidate on 74.1% of problems against Claude's 81.8% (χ^2=2.46, n.s.), i.e. *less* often. That measures which source it picks, not whether it shares specification errors with that source, so it weakens the family reading without excluding inherited blind spots.

   The failure is entirely in the sequential class. On the 74 combinational problems `gpt-5` scores 95.9, exactly the cross-verifier, and Claude 97.3 — there is no headroom there for anyone. On the 70 sequential problems the cross-verifier gets 87.1, **`gpt-5` drops to** 84.3 and Claude rises to 90.0 against an oracle of 94.3. Six of the seven problems where the two judges disagree are sequential, and Claude is right on six of the seven.

   Its own uncertainty cannot rescue it. `gpt-5`'s three votes are unanimous on only 62% of problems; it scores 93.3 where they agree and 84.9 where they split (χ^2=2.61, n.s.), and a hybrid that abstains on split votes and falls back to the cross-verifier's pick scores 90.2 — no better than `gpt-5` alone and *worse* than the cross-verifier alone (91.7, 132/144, on these problems).

   What the second judge establishes. Judge identity changes the observed result, and the difference is concentrated on sequential problems. This is consistent with differences in specification comprehension, but n=2 judges do not isolate comprehension from composition, prompting, or other model properties. A third judge alone would not settle the mechanism.

5. **The Claude-judge gain is significant across seeds, not within any run with per-problem verdicts.** The paired exact McNemar for the cross-verifier against the Claude judge on the identical stored pool is p=0.09 on VerilogEval-v2 seed-1 and p=0.63/1.00/0.38 on the three RTLLM-v2 seeds; the across-seed t-test on VerilogEval-v2 is p=0.019 (§5). At n=156 a 4.5pp effect with 13 discordant problems is simply not resolvable, so the claim we make is reproducibility, not per-run significance. VerilogEval-v2 seeds 2–3 lack per-problem verdicts; their logged fix/break counts are reported in Appendix B but not counted as within-run tests.

6. **The two OOD Claude-judge rows carry high variance across seeds.** On ChipBench-SC the Claude-judge row is 31.1 ± 5.1 over three generation seeds (per-seed 36.7/30.0/26.7 on independently generated pools with oracles 40.0/40.0/43.3; the margin is taken against the 3-seed oracle mean 41.1, and the loss 10.0 ± 6.7pp is the mean and sd of the per-seed oracle differences 3.3/10.0/16.6): the best seed alone would suggest parity with Claude one-shot (36.7); the mean does not, and we report the mean. On CVDP-CID003 it is 42.3 ± 1.8 over two seeds, each judged on its own independently generated pool (41.0/43.6) — no gain over cross-verification

(43.6), the margin-below-loss case (margin 1.3pp $<$ loss 6.4 ± 1.8pp) where the Claude judge overrides correct cross-verifier picks ($+1$ fixed / -3 broken on seed 1, each break decided by three parsed votes whose split was not retained; $+1/-1$ on seed 2). This judge ran under a prompt without the single-correct premise, and seed 1 had 11 zero-vote fallbacks, later re-judged with no change in correctness (Appendix B). The Verilog results (VerilogEval-v2, RTLLM-v2) are 3-seed.

7. **The candidate-side init-convention transform is benchmark-dependent** ($+1.1$ oracle on VerilogEval-v2 3-seed, -3 on ChipBench-SC, where we instead handle the convention at the testbench level, Appendix D); it is a convention aid, not a capability gain, and is reported as such.
8. **Cost framing (quantified, honest).** The design is *not* cost-neutral and we do not claim it is. At list prices (Opus \$15/\$75 per M; cheap models \$0.25–2/M) over the real VerilogEval-v2 pool, the *full* design — three judge calls per problem, each reading the whole de-duplicated pool — costs at least \$0.141/problem ($\geq 5.0\times$ a metered \$0.028 one-shot; the verification term is modeled with 80 output tokens per vote, below the measured median of 307, so it is a lower bound): it is *more* expensive, not less. The routed point, once the pool's generation is re-metered with `gpt-5-mini`'s hidden reasoning, is no cheaper than one-shot either. The configurations of ours below one-shot's price are the cross-verifier configuration (\$0.017, 85.3; 86.8 with the tools) and the frontier-free one (\$0.022, 89.7; Appendix P) with its two exploratory variants (\$0.019, 89.7, which dominates it; \$0.018, 88.9, below a mix); none shows a demonstrated cost advantage over a price-matched mix of the two one-shots. Cheap *generation* of the nine candidates is about half of one frontier call; the frontier *verification* is the cost driver.