CIRCUITTRACE: TRACING CIRCUIT INTENT FROM PREDICTION TO EDITABLE SCHEMATICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph accuracy alone can overstate the advantage of a circuit-generation method. CircuitTrace traces natural-language requests through model prediction, executable validation, and saved KiCad artifacts to reveal where correctness is lost. It compares two contracts through a shared deterministic backend: SemanticIntent predicts circuit intent and delegates symbol and pin resolution to a binder; FullSpec predicts executable graphs directly. Layered checks separate graph correctness, contract validity, artifact fidelity, reference topology, and native electrical rule checking. On 59 project-disjoint passive-circuit tasks with one or two components across three training seeds, post-hoc value and net-name diagnostics give FullSpec 141 correct graphs against SemanticIntent's 131, reversing the literal-value ordering. This 10-graph lead shrinks to 3 joint reference-topology successes after realization: 133/177 versus 130/177 (75.1% versus 73.4%). FullSpec loses 4 successes at executable validation, 2 at artifact saving, and 2 at topology checking; SemanticIntent loses one artifact. Native electrical checks further distinguish topology agreement from error-free artifacts. CircuitTrace exposes implementation failures hidden by graph accuracy and provides a reproducible protocol for evaluating circuit generation through the saved design.

1 INTRODUCTION

A natural-language circuit request names an electrical function, but an editable schematic also needs library symbols, physical pins, nets, and saved connections. These implementation details cross a model–tool boundary: the model may predict them directly, or a tool may derive them from a more semantic graph. The resulting artifact matters because engineers inspect and revise the saved design, not the model's intermediate text.

SchGen and pcbGPT generate editable KiCad circuits, PCBSchemaGen and CircuitLM use structured representations and verification, and IoTGen links component retrieval to schematic and board creation (Luo et al., 2026b; King et al., 2026; Zou et al., 2026; Al Hasan et al., 2026; Luo et al., 2026a). GUI-EDA studies interaction with engineering software (Li et al., 2025). A matched evaluation can connect these stages by following each prediction through the same realization tool and checking the resulting saved schematic.

Three boundaries make that comparison difficult (Figure 1). First, equivalent component values can be written differently, so literal graph scoring can change a representation ranking. Second, an electrically correct graph can lack executable library or pin details. Third, a valid execution plan can fail to produce the intended saved topology; native electrical rule checking (ERC) asks a further electrical question. A useful evaluation must retain the same tasks while observing each boundary.

CircuitTrace compares two whole prediction contracts (Figure 2). SemanticIntent predicts component classes, roles, values, and logical nets; a deterministic binder supplies implementation identities and pin endpoints. FullSpec predicts the executable graph directly. Both paths use the same compiler and deterministic live KiCad executor. A verifier then checks the saved artifact against the prediction and held-out reference and runs native electrical checks.

On final Evaluation59, FullSpec produces 141 value-aware correct graphs across three training seeds, compared with 131 for SemanticIntent. The 10-success lead shrinks to 3 after execution (133 versus 130 joint reference-topology successes). The staged trace distinguishes prediction errors, im-

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.

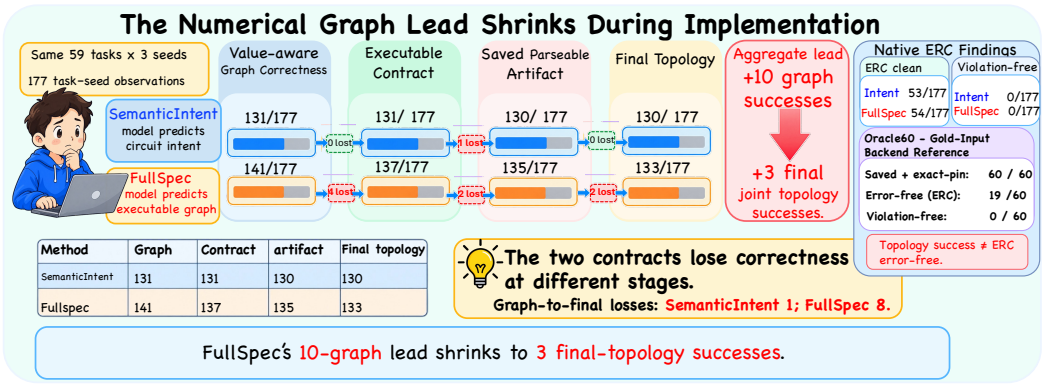


Figure 1: Final Evaluation59 outcomes across three seeds, with post-hoc ValueSemantics v1 and NetName v2. Topology and model ERC counts are joint successes. Bar fills are schematic, not a quantitative scale; printed counts give the results. Oracle60 is a separate original-release gold-input run. The comparison covers whole contracts. Enlarged copies appear in Appendix A.

plementation constraints, and execution failures; native ERC checks further electrical properties of the saved design.

We contribute (1) a project-disjoint benchmark of passive-circuit units with source and reference records; (2) a matched comparison of two prediction contracts through a shared compiler and executor; and (3) per-case evidence linking graph, contract, saved-artifact, topology, and native ERC outcomes. Together, these records locate failures that a graph-level score cannot distinguish.

2 Related Work

Grounded circuit generation. Jansen studied electronic-device design from natural-language descriptions (Jansen, 2023), and ChatSCH separates netlist planning from schematic layout and routing (Wang et al., 2027). SchGen studies semantic-grounded code and editable KiCad generation; PCB-SchemaGen couples code synthesis with structured verification; and CircuitLM integrates component knowledge, graph construction, and deterministic checking (Luo et al., 2026b; Zou et al., 2026; Al Hasan et al., 2026). pcbGPT combines a grounded Python circuit DSL with execution and ERC checks before export, followed by structural and semantic validation (King et al., 2026). IoTGen connects semantic component retrieval, schematic generation, and PCB layout in an end-to-end natural-language hardware workflow (Luo et al., 2026a). The EasyEDA Pro API Skill exposes a live-editor bridge for AI coding tools (EasyEDA, n.d.).

Representations and tool-mediated design. Schemato and EEschematic reconstruct editable schematics from netlists (Matsuo et al., 2025; Liu & Chitnis, 2026). AnalogCoder and AnalogCoder-Pro use code generation and feedback for analog design (Lai et al., 2025; 2026); LaMAGIC and LaMAGIC2 study analog topology formulations (Chang et al., 2024; 2025). ChatEDA, ChipNeMo, EDAid, and AutoEDA use language models as EDA assistants or tool-flow coordinators (Wu et al., 2024; Liu et al., 2023; Wu et al., 2025a; Lu et al., 2026). EEBench evaluates atopile design bundles with build, simulation, and bill-of-materials checks, while HWE-Bench evaluates 300 board-level design tasks using datasheets, static rules, and simulation (atopile, n.d.; Qiu et al., 2026).

Execution and artifact-grounded evaluation. Pirouzan evaluates generated circuits with distinct structural and simulation checks (Pirouzan, 2026); Cua-Bench grades saved KiCad projects against reference netlists (Qi & Parchami, 2026). WebArena and OSWorld evaluate agents in executable environments (Zhou et al., 2024; Xie et al., 2024), while CADBench evaluates executable 3D CAD programs and geometric fidelity (Doris et al., 2026). SeeClick, CogAgent, OS-ATLAS, Agent S, PAGER, and GUI-EDA study grounding, action policies, or engineering interfaces (Cheng et al., 2024; Hong et al., 2024; Wu et al., 2025b; Agashe et al., 2025; Wei et al., 2026; Li et al., 2025). CircuitTrace compares matched prediction contracts through shared realization and saved-artifact checks.

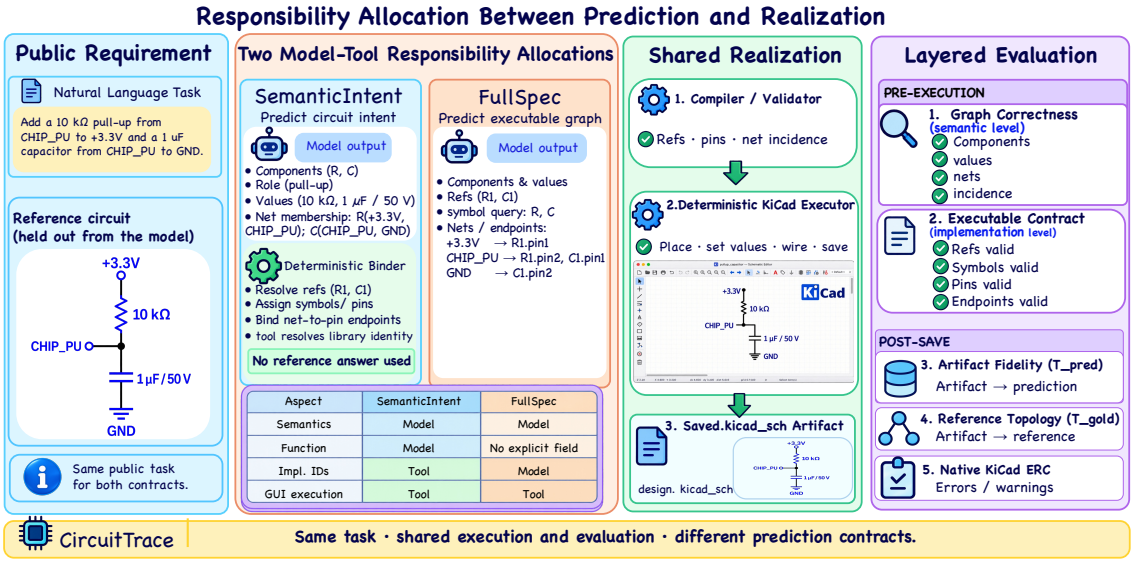


Figure 2: Both contracts use the same public request and a held-out reference. The full request specifies a $1\,\mu\mathrm{F}/50\,\mathrm{V}$ capacitor. SemanticIntent binds logical nets to pins; "Function" denotes its functional-role field. This is a whole-contract comparison.

3 Method

3.1 Two Prediction Contracts

A task consists of a public requirement x and a reference circuit graph $g^\star$. The requirement specifies identifiable components, values, named interfaces, connections, and explicit constraints. We evaluate bounded passive circuit units: resistors, capacitors, and two-terminal symmetric jumpers, with named power and ground nets.

A semantic graph contains components with class and value attributes, logical net membership, and explicit constraints. SemanticIntent also predicts functional roles; its deterministic binder adds references, supported library identities, and physical endpoints. FullSpec predicts those executable fields itself. The shared compiler turns either executable graph into a plan of KiCad GUI actions, then parses the saved artifact (Figure 2). Both models own component classes, values, and logical nets; geometry, routing, actions, and artifact checks remain tool-owned.

3.2 Deterministic Grounding and Compilation

The binder supplies implementation fields only when the supported contract determines them. It assigns references and library identities, gives symmetric terminals a deterministic ordering, and serializes logical connections as pin endpoints. It preserves the model's component inventory, values, and logical connectivity. It does not consult the reference answer. Missing asymmetric-terminal evidence or unsupported symbols cause rejection rather than guessed completion.

The compiler checks reference uniqueness, endpoint existence, supported terminals, and net incidence before generating placement, parameter-entry, wiring, labeling, and save actions. A syntactically valid endpoint string is insufficient: a pin must exist and cannot belong to conflicting nets. Consequently, JSON-schema acceptance, executable-contract validity, and circuit correctness are separately observable.

SemanticIntent also receives functional-role supervision, which FullSpec does not represent. The common graph score excludes roles; the comparison covers the complete contracts, including this supervision difference.

3.3 Verification from the Saved Artifact

Verification begins with the saved `.kicad_sch`, not the intended action plan. Let F indicate a saved, parseable artifact. We record T_{pred}, agreement with the bound prediction, and T_{gold}, agreement with the separately provided reference graph. The former tests faithful execution; the latter tests reference-relative correctness. Both are required to explain a successful realization, because faithful execution alone can preserve a model error.

Native KiCad electrical rule checking (ERC) (KiCad Developers Team, 2026) supplies two additional predicates: error-free means zero reported errors, and violation-free means zero reported errors and warnings under the recorded configuration. Joint native success additionally requires joint reference-topology success. These electrical predicates complement topology checks; they do not establish functional equivalence.

Each case retains its prediction, bound graph, plan, attempt identity, artifact hash, and verification reports. Interrupted attempts and permitted retries remain visible. The evaluated GUI executor follows deterministic plans; autonomous model-generated GUI planning is outside this experiment.

4 Dataset

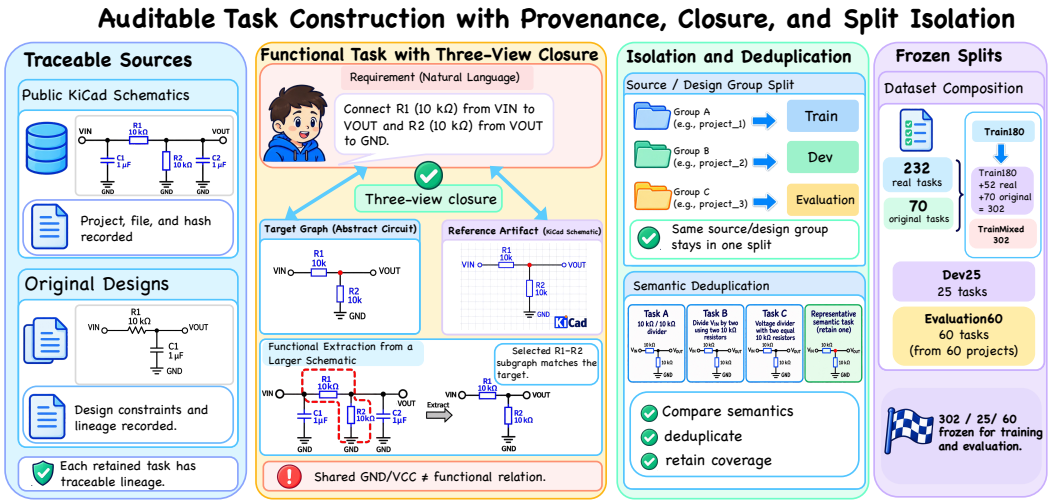


Figure 3: The panels illustrate distinct source, functional-closure, and split checks; they do not trace one circuit through a single provenance chain. Acceptance requires requirement–target–artifact agreement. Evaluation60 is the constructed pool; final evaluation excludes REF006 (Table 1).

4.1 Dataset Construction

Figure 3 summarizes the construction checks. Real-derived tasks begin with public schematic artifacts, project provenance, fixed-commit records, and file hashes. Extraction selects a functional unit with evidence for its role, rather than any connected collection of components. Sharing ground or a power rail does not establish a functional relationship. For example, an RC task requires the resistor and capacitor to form the stated signal/control path; two unrelated devices sharing a supply are rejected.

Original-design tasks begin with a reference graph and explicit design constraints. Requirements and both output representations are then derived and checked against those constraints. Original and real-derived records retain separate provenance. Paraphrases, renamed instances, and value-only variants are not counted as additional core tasks.

Construction checks requirement–target–artifact closure against source or design evidence. AI agents reviewed these relationships during construction. An initial native source audit identified four extraction errors. We versioned their corrected public requests, both target representations, and reference artifacts, then reran the affected tasks with the original checkpoints. Independent source

review then rejected REF006 and flagged three net-name judgments. The final Evaluation59 release excludes REF006 and applies a uniform minus-sign naming policy (Section 5.5); it supplies the main results. The original release supports the training and capacity comparisons. Appendix F.3 records the corrections and review findings.

4.2 Dataset Analysis

Train180 contains 180 real-source tasks. Mixed302 retains those tasks and adds 52 real-derived and 70 original tasks; twelve augmentation duplicates were excluded. The expansion changes both the quantity and composition of training data. Multi-component tasks increase from 66/180 to 178/302.

Table 1: Frozen task splits. Multi-comp.: at least two functional components; family counts appear in Appendix C.1.

Split	Tasks	Real	Original	Projects[†]	Multi-comp.
Train180	180	180	0	144	66 (36.7%)
Mixed302	302	232	70	171	178 (58.9%)
Dev25	25	25	0	25	20 (80.0%)
Evaluation60	60	60	0	60	42 (70.0%)

[†]Counts cover real-source projects; original designs have a separate lineage. Final Evaluation59 excludes REF006: 59 tasks/projects, including 41 two-component tasks.

Dev25 supports development decisions. The original Evaluation60 release was frozen before inference and contains 60 real-derived tasks from 60 projects and artifacts, disjoint from training by source group and task identity. It has 18 one-component and 42 two-component tasks. Mixed302 contains 302 unique semantic tasks, with decoupling and pull-up accounting for 160; Evaluation60 emphasizes pull-up, control, interface, divider, termination, and filter tasks. Appendices C.1 and H give family counts and example circuits.

4.3 Benchmark Comparison

Existing work covers parts of the path from natural language to a circuit artifact. Table 2 compares reported evaluation boundaries, using "NR" when the cited source does not establish a specific check. SchGen and pcbGPT generate editable KiCad outputs (Luo et al., 2026b; King et al., 2026); PCBSchemaGen uses a layered custom graph verifier (Zou et al., 2026); HWE-Bench evaluates board-level tasks with datasheets, static rules, and simulation (Qiu et al., 2026); Cua-Bench compares saved KiCad projects with reference netlists (Qi & Parchami, 2026). Our differentiating evaluation is the matched prediction-contract comparison through a shared executor with separate graph, contract, artifact, and native ERC outcomes.

Table 2: Evaluation dimensions. Y: reported; G: KiCad ERC gate without a standalone saved-artifact rate; NR: not reported. Live: editor execution; matched: common realization.

System	NL input	Edit. file	Live editor	Ref. topology	Native ERC	Project split	Matched boundary	Human audit
SchGen	Y	Y	NR	Y	G	Y[a]	NR	Y
PCBSchemaGen	Y	Y	NR	NR	G	NR	NR	Y
pcbGPT	Y	Y	NR	Y	NR	NR	NR	Y
HWE-Bench	Y	NR	NR	NR	NR	NR	NR	NR
Cua-Bench	Y	Y	Y	Y	NR	NR	NR	Y[b]
CircuitTrace	Y	Y	Y	Y	Y	Y	Y	Y[c]

[a]SchGen also tests on 20 unseen GitHub projects. [b]A second engineer reviewed all 25 Cua-Bench tasks. [c]Separate reviews cover 97 prediction triples and all 60 source references; Section 5.5 reports failures and disagreements. PCBSchemaGen checks generic motifs, not task gold graphs; pcbGPT's ERC is pre-export.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

The main comparison evaluates Qwen3-8B (Yang et al., 2025) Mixed302 checkpoints for both contracts on final Evaluation59, using training seeds 20260915, 20260916, and 20260917. The original release supplies matched Base, Train180, and Mixed302 comparisons and a Qwen3-14B Mixed302 comparison with the first seed. Training uses BF16 LoRA (Hu et al., 2022) (rank 8, alpha 16, dropout 0.05, all-linear targets), AdamW at 10^{-5} with cosine decay, and 1,400 optimizer steps per single-GPU run. Microbatch 1, accumulation 4, and a 4,096-token sequence limit are shared; the confirmation inference host records NVIDIA A100-SXM4-40GB cards. Equal steps give different sample and token exposure across training sets. Inference uses the same JSON-schema-constrained decoding, disabled thinking, and 1,024-token cap for both contracts; no Mixed302 SFT output reaches the cap.

Common graph correctness G^v checks classes, values, named nets, logical incidence, and explicit constraints under value interpretation v, allowing component renaming and symmetric-pin permutations. Executable validity C checks each contract's implementation fields; contract-qualified success is $S^v = G^v \wedge C$. Joint reference-topology success is $J^v = S^v \wedge F \wedge T_{\text{pred}} \wedge T_{\text{gold}}$. Each main-result rate uses all 59 retained tasks per seed; failed or ineligible cases remain in the denominator. Native ERC is reported separately as zero errors or zero errors and warnings under the recorded KiCad 10.0.0 configuration (KiCad Developers Team, 2026). The original scoring and GUI selection use literal values. ValueSemantics v1 was specified and tested after inspecting those results, then applied uniformly as a post-hoc sensitivity analysis; 39 newly eligible Mixed302 predictions were executed and joined with hash-verified original artifacts. For the corrected release, four tasks were rerun for both contracts and all three checkpoints, producing 24 predictions and 16 contract-qualified executions. Final review excludes REF006. NetName v2 equates Unicode minus and ASCII hyphen in net identifiers, preserves polarity, and rejects normalization collisions. Uniform rescoring admits three additional FullSpec predictions, each executed once with the unchanged backend; other artifacts are reused and affected labels rechecked. Both diagnostic policies are post-hoc. Appendix D records the chronology.

5.2 MAIN RESULTS

Table 3 and Figure 4 follow the final Evaluation59 Mixed302 predictions through realization. Mean and standard deviation summarize three training seeds evaluated on the same 59 tasks.

Table 3: Final Evaluation59, Mixed302 8B, with NetName v2. Accuracy (%), mean ± seed SD. L: literal graph scoring; V: post-hoc value-aware scoring. Contract, saved artifact, and joint topology use V eligibility and the full 59-task denominator.

Output	Graph L	Graph V	Contract	Saved	Joint topology
SemanticIntent	69.5 ± 3.4	74.0 ± 2.6	74.0 ± 2.6	73.4 ± 2.6	73.4 ± 2.6
FullSpec	61.6 ± 1.0	79.7 ± 0.0	77.4 ± 1.0	76.3 ± 1.7	75.1 ± 2.6

Value interpretation changes the numerical graph ordering. Under NetName v2, literal-value graph accuracy is 69.5% for SemanticIntent and 61.6% for FullSpec. ValueSemantics v1 changes these rates to 74.0% and 79.7%, respectively. For example, a FullSpec prediction writes `20` for a requested $20\,\Omega$ resistor. Literal matching rejects the spelling; value-aware matching accepts its electrical value. Because component alignment depends on values, this distinction can also change connectivity verdicts.

The graph lead shrinks during realization. Across three seeds, SemanticIntent follows $131 \rightarrow 131 \rightarrow 130 \rightarrow 130$ through value-aware graph, contract, saved artifact, and joint topology checks. FullSpec follows $141 \rightarrow 137 \rightarrow 135 \rightarrow 133$. Its 10 additional graph successes become 3 additional final successes: 4 predictions fail executable validation, 2 lack parsed artifacts, and 2 fail topology checks. SemanticIntent loses one artifact. This trace locates the boundaries behind the smaller final difference.

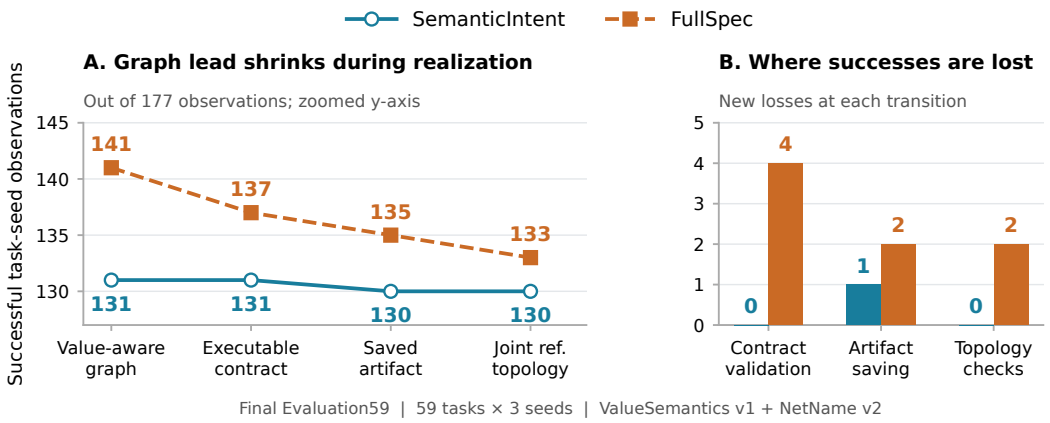


Figure 4: Final Mixed302 results on 59 tasks × three seeds, with ValueSemantics v1 and NetName v2. A: stage totals (zoomed axis). B: losses at each transition (zero-based axis). The 10-success graph lead shrinks to 3 joint topology successes. Appendix E.1 reports per-seed uncertainty.

Table 4: Original-release value-aware accuracy (%) on Evaluation60, mean ± seed SD for SFT and one evaluation for Base. Joint reference-topology success was evaluated only for Mixed302. Value-aware scoring is post-hoc; Appendix E.2 gives literal and per-seed results.

Training	Output	Graph	Contract	Joint topology
Base	SemanticIntent	0.0	0.0	–
Base	FullSpec	0.0	0.0	–
Train180	SemanticIntent	69.4 ± 1.9	69.4 ± 1.9	–
Train180	FullSpec	71.7 ± 0.0	63.3 ± 1.7	–
Mixed302	SemanticIntent	75.6 ± 1.9	75.6 ± 1.9	75.0 ± 1.7
Mixed302	FullSpec	80.0 ± 0.0	77.8 ± 1.0	75.6 ± 2.5

Per-seed topology counts are 42/44, 45/43, 43/46 (SemanticIntent/FullSpec), with different successful task sets. Paired 95% bootstrap intervals span zero; Appendix E.1 gives counts and tests.

Topology agreement and native ERC measure different properties. Joint error-free ERC counts are 18/18/17 for SemanticIntent and 18/18/18 for FullSpec; neither contract produces a violation-free artifact under this protocol. The original-release gold-input Oracle60 run likewise reaches 60/60 exact-pin matches, with 19/60 error-free and zero violation-free reports. The extracted fragments can preserve their requested connections while exposing undriven-power findings (Section 5.4). Appendix F.3 gives the native-netlist checks and escaped-name diagnostic.

5.3 Training Expansion and Capacity

Table 4 preserves the training comparison on the original requests and targets.

On the original release, the Train180-to-Mixed302 comparison raises value-aware graph accuracy from 69.4% to 75.6% for SemanticIntent and from 71.7% to 80.0% for FullSpec, with increases in every seed. Mixed302 jointly adds 122 tasks, source diversity, and multi-component relations, so this is an expansion-strategy comparison rather than an isolated data-volume effect. Train180 FullSpec loses 4–6 graph-correct cases per seed at executable validation, compared with 1–2 per Mixed302 seed; SemanticIntent has no such loss in these runs.

At 14B and one seed on the original release, Mixed302 FullSpec yields 49/60 value-equivalent graphs against 44/60 for SemanticIntent, with contract-qualified counts 47/60 and 44/60. This capacity diagnostic has no multi-seed or joint artifact comparison. Appendix E reports paired tests within each release and seed.

5.4 Case and Error Analysis

Graph correct, contract failed. A request filters `ADC_VREF` into `ADC_AVDD` through a $1\,\Omega$ resistor with a $2.2\,\mu\mathrm{F}$ shunt capacitor. FullSpec predicts the requested components and connections but marks `ADC_AVDD` as a power net; the executable contract rejects that unsupported power name (Appendix F.1, case `ac7a4fdf15fbed4c`).

Artifact failed. A request places a $120\,\Omega$ termination resistor between `RJ45_OUT_3` and `RJ45_OUT_4`. SemanticIntent predicts that resistor and the two nets, but the KiCad GUI Add/Place Symbol menu fallback fails. No validated saved schematic results, so this case has no successful topology or ERC outcome (case `a1e42f3309b33fbf`).

Topology passed, ERC failed. A pull-up request specifies a $10\,\mathrm{k}\Omega$ resistor on `CHIP_PU` to $+3.3\,\mathrm{V}$ with a $1\,\mu\mathrm{F}/50\,\mathrm{V}$ capacitor to ground. Its saved topology matches the reference, while KiCad reports two undriven-power errors (case `1ae8e2eb370efee7`). The three cases locate different failure boundaries; Appendix E.4 gives per-seed error predicates.

5.5 Human Evaluation

Two human experts independently checked AI-assisted request/reference/candidate materials from the original evaluation without seeing each other's judgments. Deduplicating 192 selected observations yields 97 distinct triples. This evaluation checks prediction equivalence; reference-source review is a separate task. Reviewers agreed on 94/97 triples (76 equivalent, 18 non-equivalent; three unresolved), giving $\kappa = 0.911$ with conditional judgments as a third category. ValueSemantics v1 agreed with 92/94 decisive consensuses. Two jointly accepted value spellings remained conservative scorer false negatives; frozen scores were not revised. Appendix C gives the sampling and disagreement details.

Two independent source-review reports cover the corrected 60-task pool: both accept 56 references and reject REF006; three net-name judgments differ. Final adjudication excludes REF006 and retains REF048–050 under NetName v2. The policy checks negative-net identity while accepting the two minus spellings, and is applied to all frozen predictions. Appendix F.4 preserves the original judgments, tests, and versioned results.

6 Conclusion and Limitations

CircuitTrace follows circuit predictions through executable contracts into saved KiCad schematics. On final Evaluation59, FullSpec's value-aware graph lead of 10 task–seed observations shrinks to 3 at joint reference topology (133/177 versus 130/177). The per-case trace locates the contract and realization losses behind this change. Native ERC adds electrical findings that topology agreement alone does not capture. Circuit-generation evaluation therefore benefits from reporting what is predicted, what can be executed, and what is preserved in the saved design.

The final evaluation covers 59 project-disjoint passive units with one or two components and deterministic execution. Source review motivated one exclusion and an explicit net-name policy. Contract differences include role supervision, so they do not isolate implementation responsibility. ValueSemantics v1, source correction, and NetName v2 are post-hoc; paired intervals establish neither superiority nor equivalence. Training expansion changes data quantity and composition together. Larger circuits, further model families, and autonomous GUI planning remain to be evaluated.

References

Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Wang. Agent S: An open agentic framework that uses computers like a human. In *International Conference on Learning Representations*, 2025.

Khandakar Shakib Al Hasan, Syed Rifat Raiyan, Hasin Mahtab Alvee, and Wahid Sadik. CircuitLM: A multi-agent LLM-aided design framework for generating circuit schematics from natural language prompts. *arXiv preprint arXiv:2601.04505*, 2026.

atopile. EEBench: Methodology. Project website, n.d. URL https://www.eebench.org/methodology.html. Accessed September 24, 2026.

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

Chen-Chia Chang, Wan-Hsuan Lin, Yikang Shen, Yiran Chen, and Xin Zhang. LaMAGIC2: Advanced circuit formulations for language model-based analog topology generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 7351–7360. PMLR, 2025.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing GUI grounding for advanced visual GUI agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9313–9332, 2024. doi: 10.18653/v1/2024.acl-long.505.

Anna C. Doris, Jacob Thomas Sony, Ghadi Nehme, Era Syla, Amin Heyrani Nobari, and Faez Ahmed. CADBench: A multimodal benchmark for AI-assisted CAD program generation. *arXiv preprint arXiv:2605.10873*, 2026.

EasyEDA. EasyEDA Pro API Skill. GitHub repository, n.d. URL https://github.com/easyeda/easyeda-api-skill. Accessed September 24, 2026.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. CogAgent: A visual language model for GUI agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14281–14290, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Peter Jansen. From words to wires: Generating functioning electronic devices from natural language descriptions. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 12972–12990, 2023. doi: 10.18653/v1/2023.findings-emnlp.864.

KiCad Developers Team. KiCad schematic editor. KiCad 10.0 official documentation, 2026. URL https://docs.kicad.org/10.0/en/eeschema/eeschema.html. Accessed September 24, 2026.

Tobias King, Steven Kehrberg, Michael Beigl, and Tobias Röddiger. pcbGPT: Automatic PCB schematic synthesis from natural language requirements. *arXiv preprint arXiv:2606.01188*, 2026.

Yao Lai, Sungyoung Lee, Guojin Chen, Souradip Poddar, Mengkang Hu, David Z. Pan, and Ping Luo. AnalogCoder: Analog circuit design via training-free code generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(1):379–387, 2025. doi: 10.1609/aaai.v39i1.32016.

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

Chunyi Li, Longfei Li, Zicheng Zhang, Xiaohong Liu, Min Tang, Weisi Lin, and Guangtao Zhai. Using GUI agent for electronic design automation. *arXiv preprint arXiv:2512.11611*, 2025.

Chang Liu and Danial Chitnis. EEschematic: Multimodal-LLM based AI agent for schematic generation of analog circuit. In *2026 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 822–826. IEEE, 2026. doi: 10.1109/ISCAS66217.2026.11562002.

Mingjie Liu, Teodor-Dumitru Ene, Robert Kirby, Chris Cheng, Nathaniel Pinckney, Rongjian Liang, Jonah Alben, Himyanshu Anand, Sanmitra Banerjee, Ismet Bayraktaroglu, Bonita Bhaskaran, Bryan Catanzaro, Arjun Chaudhuri, Sharon Clay, Bill Dally, Laura Dang, Parikshit Deshpande, Siddhanth Dhodhi, Sameer Halepete, Eric Hill, Jiashang Hu, Sumit Jain, Ankit Jindal, Brucek Khailany, George Kokai, Kishor Kunal, Xiaowei Li, Charley Lind, Hao Liu, Stuart Oberman, Sujeet Omar, Ghasem Pasandi, Sreedhar Pratty, Jonathan Raiman, Ambar Sarkar, Zhengjiang Shao, Hanfei Sun, Pratik P. Suthar, Varun Tej, Walker Turner, Kaizhe Xu, and Haoxing Ren. ChipNeMo: Domain-adapted LLMs for chip design. *arXiv preprint arXiv:2311.00176*, 2023.

Yiyi Lu, Hoi Ian Au, Junyao Zhang, Jingyu Pan, Guanglei Zhou, Yiting Wang, Jingwei Sun, Ang Li, Jianyi Zhang, Hai Li, and Yiran Chen. AutoEDA: Enabling EDA flow automation through microservice-based LLM agents. *ACM Transactions on Design Automation of Electronic Systems*, 2026. doi: 10.1145/3842675.

Qinpei Luo, Ruichun Ma, Xinyu Zhang, and Lili Qiu. [Emerging Ideas] IoTGen: Towards LLM-driven IoT hardware generation. In *Proceedings of the 24th Annual International Conference on Mobile Systems, Applications and Services*, pp. 792–807. ACM, 2026a. doi: 10.1145/3745756.3809239.

Qinpei Luo, Ruichun Ma, Xinyu Zhang, and Lili Qiu. SchGen: PCB schematic generation with semantic-grounded code representations. *arXiv preprint arXiv:2605.30345*, 2026b.

Ryoga Matsuo, Stefan Uhlich, Arun Venkitaraman, Andrea Bonetti, Chia-Yu Hsieh, Ali Momeni, Lukas Mauch, Augusto Capone, Eisaku Ohbuchi, and Lorenzo Servadei. Schemato–an LLM for netlist-to-schematic conversion. In *2025 ACM/IEEE 7th Symposium on Machine Learning for CAD (MLCAD)*, pp. 1–7. IEEE, 2025. doi: 10.1109/MLCAD65511.2025.11189059.

Ali Hedayati Pirouzan. Validation and simulation catch different errors: Four levels of evaluation for LLM-generated circuits. *arXiv preprint arXiv:2609.26830*, 2026.

Zhengyang Jason Qi and Armin Parchami. Cua-Bench: Benchmarking computer-use agents on professional software. Snorkel AI technical blog, 2026. URL `https://snorkel.ai/blog/cua-bench-benchmarking-computer-use-agents-on-professional-software/`. Accessed September 24, 2026.

Weibo Qiu, Yinhao Xiao, and Runyu Pan. HWE-Bench: Can language models perform board-level schematic designs? *arXiv preprint arXiv:2603.18102*, 2026.

Wenqing Wang, Wenhao Yin, Ruichen Zhang, Kexin Ma, Chengyang Liu, Junjie Jiao, Zhi Chai, Junqi Shen, Zhiqiang Huang, Xiangjian Zeng, and Qingqiang Wu. ChatSCH: A large language model-driven framework for board-level schematic generation. In *Advanced Intelligent Computing Technology and Applications*, volume 16656 of *Lecture Notes in Computer Science*, pp. 378–390. Springer, 2027. doi: 10.1007/978-981-92-3441-7_32.

Jingxuan Wei, Xi Bai, Shan Liu, Caijun Jia, Zheng Sun, Xinglong Xu, Siyuan Li, Linzhuang Sun, Bihui Yu, Conghui He, and Cheng Tan. PAGER: Bridging the semantic-execution gap in point-precise geometric GUI control. *arXiv preprint arXiv:2605.15963*, 2026.

Haoyuan Wu, Zhuolun He, Xinyun Zhang, Xufeng Yao, Su Zheng, Haisheng Zheng, and Bei Yu. ChatEDA: A large language model powered autonomous agent for EDA. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 43(10):3184–3197, 2024. doi: 10.1109/TCAD.2024.3383347.

Haoyuan Wu, Haisheng Zheng, Zhuolun He, and Bei Yu. Divergent thoughts toward one goal: LLM-based multi-agent collaboration system for electronic design automation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1710–1721, 2025a. doi: 10.18653/v1/2025.naacl-long.83.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. OS-ATLAS: Foundation action model for generalist GUI agents. In *International Conference on Learning Representations*, 2025b.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094, 2024. doi: 10.52202/079017-1650.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

Huanghaohe Zou, Peng Han, Emad Nazerian, Mafu Zhang, Zhicheng Guo, and Alex Q. Huang. PCBSchemaGen: Reward-guided LLM code synthesis for printed circuit boards (PCB) schematic design with structured verification. *arXiv preprint arXiv:2602.00510*, 2026.

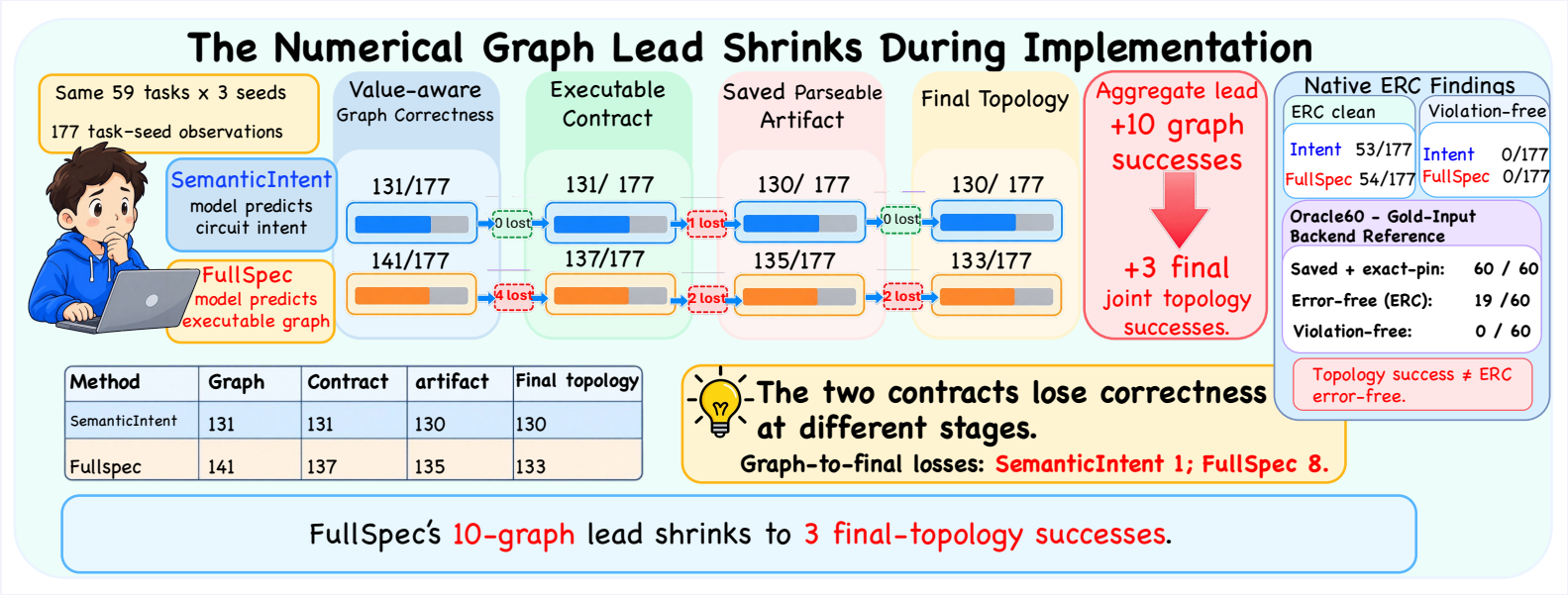
AI USE STATEMENT

Generative AI agents assisted with research planning, implementation, candidate construction, prompt revision, dataset review under user-authorized standards, experiment coordination, analysis, figure preparation, and manuscript writing. Dataset construction review was agent-assisted. Separately, two human experts independently verified the 97-triple electrical-equivalence packet using AI-assisted materials, without access to each other's judgments. Two confirmed reviewer reports also document independent source-reference review of the corrected 60-task release using prepared evidence. The authors retain responsibility for the final claims and submission disclosures.

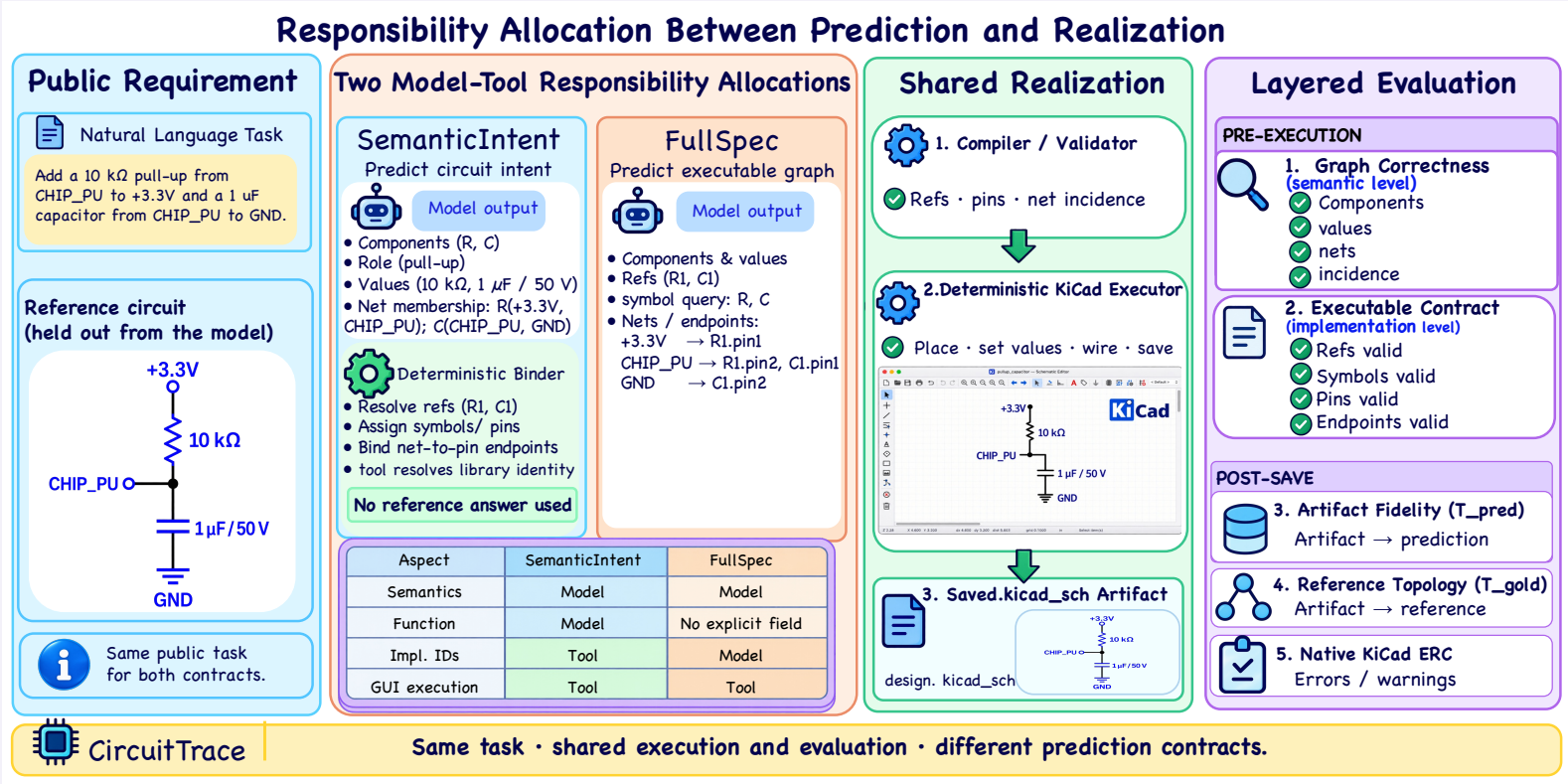
A Enlarged Main Figures

The following pages enlarge Figures 1–3. Figure 1 reports final Evaluation59 with ValueSemantics v1 and NetName v2; its Oracle60 panel is the separate original-release gold-input run. The panels in Figure 3 illustrate separate construction checks.

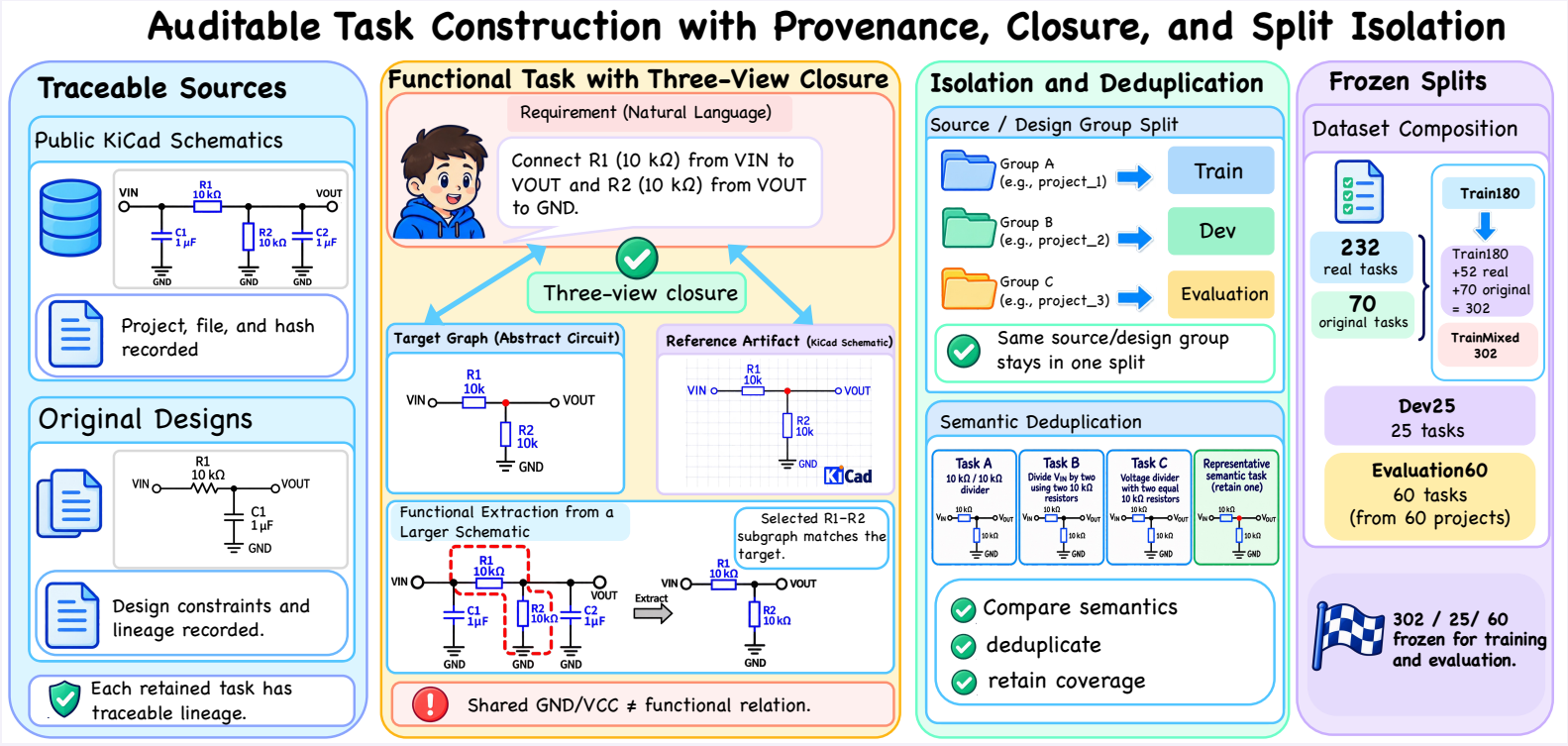
Enlarged Figure 1



Enlarged Figure 2



Enlarged Figure 3



B Representation Interfaces

Guide to the appendices

A: enlarged main figures; **B:** representation schemas; **C:** dataset and expert verification; **D:** scoring protocol; **E:** detailed results; **F:** execution traces; **G:** reproducibility; **H:** full source schematics and extracted tasks.

B.1 Representation Schemas and a Complete Example

SemanticIntent requires `version`, `components`, `nets`, and `explicit_constraints`. Each component has a local ID, class, role, and an optional explicit value; nets list component memberships. FullSpec instead supplies executable references, symbol queries, net kinds, and numbered endpoints. Decoding schemas and runtime validators are separate: a schema-valid string need not name a supported power symbol or physical pin.

Example request — A series signal link

Add a 20 Ω series resistor in the signal path between `USB_D-` and `USBDN1`.

Required relation: the two named nets connect through one resistor. **Reference identity:** task suffix `0ab5c6190c576247`. The complete reference targets are shown below.

SemanticIntent — Classes and logical memberships

```
{
  "version": "semantic_intent_spec_v1",
  "components": [{"component_id": "c1", "class": "resistor",
                  "role": "series_signal_link", "value": "20R"}],
  "nets": [
    {"name": "USBDN1", "members": [{"component_id": "c1"}]},
    {"name": "USB_D-", "members": [{"component_id": "c1"}]}
  ],
  "explicit_constraints": []
}
```

FullSpec — Symbols and numbered endpoints

```
{
  "version": "kicad_circuit_spec_v1",
  "task_key": "TRAINV2_PAIR_0AB5C6190C576247",
  "layout_preset": "horizontal_basic",
  "components": [{"query": "R", "reference": "R1", "value": "20R"}],
  "nets": [
    {"name": "USBDN1", "kind": "label", "endpoints": ["R1.pin1"]},
    {"name": "USB_D-", "kind": "label", "endpoints": ["R1.pin2"]}
  ]
}
```

The reference's `task_key` is execution-side metadata, not a field in the FullSpec decoding schema. In contrast, `layout_preset` is a required decoded field fixed by the schema to `horizontal_basic`; the model cannot select another layout. The executor determines geometry and routing. The resistor's symmetric terminal assignment changes serialization, not its electrical connection. No analogous polarity inference is permitted for asymmetric devices outside the supported scope.

C Dataset and Expert Verification

This section describes what the frozen splits contain and what the independent expert verification establishes. Task counts, project counts, and graph signatures measure different properties and are reported separately.

C.1 Dataset Composition

Dataset — Functional families

Table 5: Functional families in the frozen splits.

Family	Mixed302	Dev25	Evaluation60
Decoupling	63	0	0
Pull-up	97	7	13
Pull-down	6	0	0
Control support	58	4	13
Interface support	16	4	13
Signal filter	15	1	3
Termination	19	1	5
Voltage divider	21	8	13
Feedback	5	0	0
Configuration	1	0	0
Signal support	1	0	0
Total	302	25	60

The 70 original records comprise 11 divider, 13 filter, 13 termination, 11 interface, 13 control, five feedback, and four decoupling tasks. Counts of original tasks do not establish counts of independent base designs. Their source-artifact hash field is absent, so the 177 real artifacts are not a complete original-artifact inventory. Descriptive topology labels remain missing for 122 Mixed302, 25 Dev25, and one evaluation record.

Five provisional development rows were removed for overlap under the actual semantic projection, leaving Dev25. Historical Train205/Val44 and previously exposed development sets are kept separate from these results. Train180 spans 144 real projects and 151 artifacts; Mixed302 spans 171 real projects and 177 real artifacts, plus separate original-design lineage. Dev25 and Evaluation60 each have one real project and artifact per task.

C.2 Independent Human Verification of AI-Assisted Review Materials

Two human experts independently verified AI-assisted review materials without access to each other's judgments. This was verification of prepared request/reference/candidate triples, not blind annotation from scratch. The original-release packet combines all 120 Mixed302 seed-20260915 outputs with 85 value-rescued outputs across conditions, including 13 already in the first group: $120 + 85 - 13 = 192$ observations. Exact triple deduplication leaves 97 distinct triples. The experts agree on 94/97 pairs (96.9%): 76 equivalent and 18 non-equivalent. Three pairs remain unresolved. Treating conditional review as a third nominal category gives Cohen's $\kappa = 0.911$.

The scorer agrees with 92/94 decisive consensuses (97.9%). All 18 distinct value-rescued pairs receive agreement on equivalence. Two additional pairs are accepted by both reviewers but rejected by ValueSemantics v1: capacitor spelling `22n` versus `22nF`, and the same spelling combined with resistor `1k0` versus `1k`. The unchanged ValueSemantics v1 scorer retains these two conservative false negatives relative to the expert consensus. Three unresolved pairs concern repeated same-name nets and net-name typography. One also exposes a public-request/reference discrepancy: Unicode minus in the request's differential-net name versus ASCII hyphen in the reference. Original scores remain unchanged. The enriched, deduplicated packet is an evaluator diagnostic, not an estimate of population accuracy or an independent set of 97 circuit tasks.

D Scoring Rules and Evaluation Protocol

The literal score is retained alongside a class-aware value diagnostic. The examples below define their boundary; neither scoring track repairs a predicted circuit or changes the executable contract.

D.1 ValueSemantics v1 Specification

Value equivalence — Accepted, rejected, and deliberately unresolved

Accepted: resistor `20`, `20R`, and 20 Ω; resistor `1k` and 1000 Ω; capacitor `100nF` and `0.1uF`.

Rejected: `100pF` versus `1nF`; `20k` versus `20`; `100nF 25V` versus `100nF`.

Not inferred: capacitor `22n` versus `22nF`. The frozen parser does not supply an omitted farad unit.

Unchanged: identifiers, functional roles, component counts, logical connections, and runtime restrictions.

Normalization is class-aware and applies to values only. For resistors, bare finite decimals denote ohms; `20`, `20R`, and 20 Ω agree, as do `1k` and 1000 Ω. SI prefixes and embedded-decimal forms such as `4K7` are supported. Capacitors require an explicit farad unit: `100nF` equals `0.1uF`, but `100pF` differs from `1nF`. Bare capacitor numbers and unknown trailing ratings are not assigned implied units or stripped. Unparsed values fall back to normalized literal comparison.

The frozen test file contains eight positive and seven negative pairs. Negatives include `100` versus `100pF`, `100nF 25V` versus `100nF`, `20k` versus `20`, negative resistance, trailing text, and open versus bridged jumpers. This layer adds no identifier normalization: it does not merge names, rewrite Unicode punctuation in net names, or alter references, roles, connections, and runtime decisions. Existing literal preprocessing is unchanged.

ValueSemantics v1 is unchanged across releases. Final Evaluation59 additionally applies NetName v2 to both output representations: U+2212 maps to U+002D during net-identifier comparison and explicit-name lookup, with all other naming rules unchanged. Distinct raw net names that collide after this mapping are rejected; positive and negative terminals remain distinct. Raw requests, targets, predictions, and saved artifacts are preserved. The policy was fixed before its three newly eligible executions. Tests on REF048–050 accept the two spellings and reject wrong negative-net labels, wrong connections, and canonical-name collisions.

D.2 Scoring and Execution Chronology

The retained training release and metric addendum are dated 16 September 2026. The addendum follows the first Dev25 observations and specifies a common-scorer hash for prospective Confirmation60 evaluation. That hash matches the current scorer and subsequent reports. A Semantic seed15 launch record timestamps the request at 21:15:12 UTC on 16 September with the v9 input hash; the closeout state was registered retrospectively at 02:11:09 UTC on 17 September. These records document the sequence; they do not constitute independent preregistration.

Protocol — Evaluation chronology

Table 6: Recorded evaluation sequence. Dates refer to retained records; stage labels distinguish input, scoring, and execution snapshots.

Record	Stage	Evidence retained
16 Sep 2026	Training release and common-score addendum	Input identities and prospective common-scorer hash after Dev25 observations.
16 Sep, 21:15 UTC	SemanticIntent seed15 inference launch	Request record matches the Evaluation60 v9 input hash.
17 Sep, 07:40 UTC+8	Recorded backend-stage snapshot	Manifest labels the stage as post-CPU-score, pre-KiCad execution; includes selection and code-state patch hashes.
18 Sep 2026	ValueSemantics diagnostic	Specification, implementation, and tests hashed before uniform rescoring of existing predictions.
18 Sep 2026	Incremental execution and final join	Selected prediction hashes, 39 additional executions, and reuse of verified original artifacts.
26 Sep 2026	Source correction and matched rerun	Four versioned corrections; 24 new predictions, 16 eligible executions; original results retained.

Selected per-file identities can be checked across retained records: the compiler runtime matches the Dev25 record, the common scorer and compiler entry point match the inference manifest, and eight scoring/execution files match the 17 September portability-preflight source snapshot. No Windows execution contributes to the reported matrix.

The backend-stage manifest records a snapshot after CPU scoring; it does not establish a pre-inference freeze of every binder, compiler, executor, and verifier dependency. Value-aware eligibility extends the original execution set. Its final join retains the original results and adds the 39 executions required by the revised selection rule.

The value-aware diagnostic reproduced the original per-case literal results before extending eligibility. Its specification, implementation, and 15 equivalence tests were hashed before uniform rescoring; all tests pass. Unparsed ratings and identifiers retain their original comparison rules.

D.3 Prior-Work Evidence Boundaries

Related work — Outputs and verification

Table 7: Reported outputs and verification stages in the cited evidence. NR means not established by the cited evidence; custom graph ERC and native KiCad ERC are distinct.

System	Reported output/checks	Relation to this study
SchGen (Luo et al., 2026b)	Editable KiCad; netlist metrics; experts; KiCad ERC zero-critical-error gate	Representation ablations and composite validity reported; separate ERC error-free/violation-free rates NR
PCBSchemaGen (Zou et al., 2026)	SKiDL; five-layer graph verifier (Sec. 3.3); downstream KiCad checks	Generic motif/pin-role checks and KiCad acceptance gate; standalone native ERC rate NR
CircuitLM (Al Hasan et al., 2026)	CircuitJSON; deterministic ERC and judge (Sec. III)	Canonical pin retrieval already grounds generation; native KiCad ERC NR
pcbGPT (King et al., 2026)	Editable KiCad; grounded Python DSL; execution and ERC before export; semantic checks	Twenty-task reference evaluation; matched prediction-to-artifact retention NR
IoTGen (Luo et al., 2026a)	Semantic component retrieval; KiCad schematic and PCB generation	End-to-end hardware workflow; matched prediction-contract and saved-artifact tracing NR
GUI-EDA (Li et al., 2025)	Operational descriptions and click coordinates (Sec. III-C)	Single-step description/click evaluation and selected physical cases; saved circuit topology comparison NR
CircuitTrace	Bound graph, saved-file comparisons, native KiCad ERC	Matched output contracts with separate prediction and realization losses

Our comparison connects prediction contracts to common graph scoring, executable validity, and saved-artifact checks within one matched evaluation.

E Detailed Results and Statistical Tests

We first report final Evaluation59 with NetName v2, then retain original-release training, capacity, and diagnostic results under explicit version labels. Historical Evaluation60 uses 60 tasks per seed and the earlier net-name rule; final Evaluation59 uses 59. ValueSemantics v1 and the realization backend are unchanged.

E.1 Final Evaluation59

Final Evaluation59 — Per-seed results

Table 8: Final counts out of 59 per seed with NetName v2. Triplets follow seeds 20260915/16/17. L: literal; V: value-aware. EF/VF: joint native error-free/violation-free.

Output	Graph L	Contract L	Graph V	Contract V
SemanticIntent	39/43/41	39/43/41	42/45/44	42/45/44
FullSpec	36/37/36	34/36/35	47/47/47	45/46/46
Output	**Saved**	**Joint topology**	**Joint EF**	**Joint VF**
SemanticIntent	42/45/43	42/45/43	18/18/17	0/0/0
FullSpec	45/44/46	44/43/46	18/18/18	0/0/0

For the final release, paired tests resample the 59 task IDs within each seed using 20,000 bootstrap draws. The 15-test Holm family covers literal graph and contract, value-aware graph and contract, and joint topology for all three seeds. All 15 intervals span zero and all adjusted exact McNemar p-values equal 1. Table 9 gives the value-aware and final-topology comparisons. The seeds repeat the same tasks and are not pooled as 177 independent samples.

Final Evaluation59 — Paired comparisons

Table 9: Final Evaluation59 paired comparisons, SemanticIntent minus FullSpec. Cells: both correct / SemanticIntent only / FullSpec only / neither. Differences and bootstrap 95% intervals are percentage points; p is exact two-sided McNemar before adjustment. All tests belong to the stated 15-test Holm family.

Seed	Metric	Paired cells	Difference	95% CI	p
15	Graph V	40/2/7/10	−8.5	[-18.6,1.7]	0.180
15	Contract V	38/4/7/10	−5.1	[-16.9,5.1]	0.549
15	Joint topology	37/5/7/10	−3.4	[-15.3,8.5]	0.774
16	Graph V	42/3/5/9	−3.4	[-13.6,5.1]	0.727
16	Contract V	41/4/5/9	−1.7	[-11.9,8.5]	1.000
16	Joint topology	38/7/5/9	3.4	[-8.5,15.3]	0.774
17	Graph V	42/2/5/10	−5.1	[-13.6,3.4]	0.453
17	Contract V	41/3/5/10	−3.4	[-13.6,5.1]	0.727
17	Joint topology	40/3/6/10	−5.1	[-15.3,5.1]	0.508

E.2 Original-Release Scores and Decoding

All 480 original-release Base and Mixed302 raw outputs pass the actual decoding JSON Schema. Runtime validity is stricter because it checks endpoint identities and incidence. This explains why grammar-valid outputs can fail the executable contract.

Original release — Per-seed results

Table 10: Original-release score tracks and realized outcomes, counts out of 60. Triplets follow seeds 20260915/16/17; 14B uses seed 20260915 only. Graph/contract under literal (L) and electrical-value (E) scoring are distinct. EF/VF: joint native error-free/violation-free. Base has zero graph/contract successes under both tracks.

Training	Output	Graph L	Contract L	Graph E	Contract E
Train180	Intent	35/35/36	35/35/36	41/41/43	41/41/43
Train180	FullSpec	36/37/37	32/32/32	43/43/43	39/38/37
Mixed302	Intent	41/44/43	41/44/43	44/46/46	44/46/46
Mixed302	FullSpec	38/39/38	36/38/37	48/48/48	46/47/47
Mixed302,14B	Intent	39	39	44	44
Mixed302,14B	FullSpec	44	43	49	47
Mixed302 8B execution		Artifact	Joint topology	Joint EF	Joint VF
	Intent	44/46/45	44/46/45	18/18/17	0/0/0
	FullSpec	46/45/47	45/44/47	17/17/17	0/0/0

Runtime-valid Mixed302 counts are 59/59/59 for Intent and 55/56/57 for FullSpec. Rejected library aliases and power names remain contract failures, not electrical-graph errors. Mean generated lengths span 135.7–136.5 tokens; all outputs terminate within the 1,024-token cap.

E.3 Component-Count Diagnostic

Diagnostic — Component counts

Table 11: Original-release post-hoc component-count breakdown. Counts follow seeds 20260915/16/17; N is per seed. The same tasks recur across seeds.

Components	Output	N	Electrical graph	Contract	Joint topology
1	Intent	18	17/17/17	17/17/17	17/17/16
1	FullSpec	18	16/16/16	16/16/16	16/16/16
2	Intent	42	27/29/29	27/29/29	27/29/29
2	FullSpec	42	32/32/32	30/31/31	29/28/31

No evaluation task has three or more functional components. On the original release, FullSpec's graph advantage and all four of its contract losses occur in two-component tasks; Intent has higher graph accuracy among single-component tasks. These groups also differ in functional families and relations, so the split cannot establish that implementation loss grows with circuit complexity.

E.4 Post-Normalization Error Structure

Diagnostic — Value and graph errors

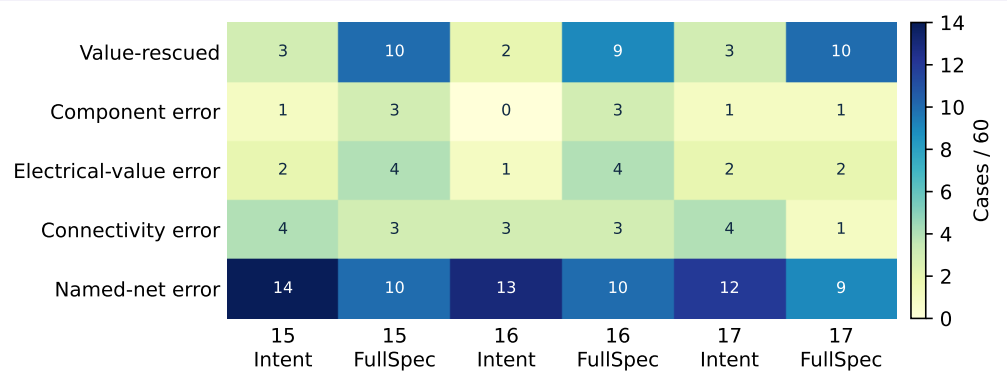


Figure 5: Original-release ValueSemantics v1 transitions and residual error flags. Value-rescued denotes literal-wrong/electrical-correct cases. Error predicates overlap; connectivity and named-net predicates ignore value mismatch. Each column uses the same 60 tasks.

The literal score aligns components using values, which couples value and connectivity failures. The value-agnostic connectivity diagnostic permits broader component alignment and may not recover a unique correspondence for repeated devices. Named-net graph errors combine named incidence and graph structure rather than measuring token-level spelling accuracy.

E.5 Original-Release Paired Tests and Execution Costs

The paired analysis resamples 60 task IDs within each seed 20,000 times and uses exact two-sided McNemar tests. For the six original-release value-aware graph/contract comparisons, Holm correction is applied as one family. Table 12 reports the paired cells; all intervals span zero and all adjusted p-values equal 1.0. The earlier 27-test analysis of literal scores and original joint outcomes is retained separately. An all-zero violation-free empirical bootstrap is uninformative about unobserved population outcomes.

Original release — Paired comparisons

Table 12: Original-release paired value-aware tests on 60 tasks per seed. Cells: both correct / first-only / second-only / neither. Difference and bootstrap 95% CI are percentage points; p is exact McNemar. Holm families remain separate: six representation tests (top) and twelve expansion tests (bottom).

Output	Seed	Metric	Paired cells	Difference	95% CI	p	p_H
Mixed302 Intent minus FullSpec							
Both	15	Graph	42/2/6/10	−6.7	[-16.7,1.7]	.289	1.000
Both	15	Contract	40/4/6/10	−3.3	[-13.3,6.7]	.754	1.000
Both	16	Graph	42/4/6/8	−3.3	[-13.3,6.7]	.754	1.000
Both	16	Contract	41/5/6/8	−1.7	[-13.3,10.0]	1.000	1.000
Both	17	Graph	44/2/4/10	−3.3	[-11.7,5.0]	.688	1.000
Both	17	Contract	43/3/4/10	−1.7	[-10.0,6.7]	1.000	1.000
Mixed302 minus Train180, within each output representation							
Intent	15	Graph	36/8/5/11	5.0	[-6.7,16.7]	.581	1.000
Intent	15	Contract	36/8/5/11	5.0	[-6.7,16.7]	.581	1.000
Intent	16	Graph	39/7/2/12	8.3	[-1.7,18.3]	.180	1.000
Intent	16	Contract	39/7/2/12	8.3	[-1.7,18.3]	.180	1.000
Intent	17	Graph	38/8/5/9	5.0	[-6.7,16.7]	.581	1.000
Intent	17	Contract	38/8/5/9	5.0	[-6.7,16.7]	.581	1.000
FullSpec	15	Graph	41/7/2/10	8.3	[-1.7,18.3]	.180	1.000
FullSpec	15	Contract	37/9/2/12	11.7	[1.7,21.7]	.065	.654
FullSpec	16	Graph	42/6/1/11	8.3	[0.0,16.7]	.125	1.000
FullSpec	16	Contract	37/10/1/12	15.0	[5.0,25.0]	.012	.129
FullSpec	17	Graph	42/6/1/11	8.3	[0.0,16.7]	.125	1.000
FullSpec	17	Contract	36/11/1/12	16.7	[6.7,26.7]	.006	.076

The expansion analysis uses the same paired task resampling and 20,000 draws per comparison. A positive percentile interval need not imply a significant exact McNemar test, especially after multiplicity correction. Seeds are not pooled as independent tasks. Case-level transitions are retained with the analysis.

The original model backend contains 239 effective executions and three prior attempts. Of those, 240 attempts have completed timers totaling 42,216.406 seconds. Two lack final durations. All 39 incremental effective executions retain case timers, totaling 5,857.109 seconds (1.627 hours); their interrupted predecessor has no final duration. Combined model timers total 48,073.515 seconds (13.354 hours), excluding those three untimed interruptions. Oracle timed execution totals 12,134.581 seconds, with 12,134.825 seconds queue wall time. Summed case times are not calendar time. These timing totals cover the original release and its value-aware extension; no formal Windows execution contributes. The corrected rerun adds 293.304 single-GPU seconds in the recorded generation loops and 2,251.209 seconds summed over 16 formal GUI case timers. Final NetName v2 adds 305.346 seconds over three formal GUI executions and no new inference. The added timings exclude model loading, environment calibration, and provisional or interrupted attempts.

F Execution Traces and Failure Cases

The three retained cases below fail at different boundaries. Each identifies what was correct, what failed, and which downstream evidence is available. They illustrate failure mechanisms, not a new evaluation subset.

F.1 Failure Case Studies

Case 1 — Graph correct; executable contract rejected

Run: FullSpec, seed 20260915; task suffix `ac7a4fdf15fbed4c`.

Request: filter `ADC_VREF` into `ADC_AVDD` with a $1\,\Omega$ series resistor and a $2.2\,\mu$F capacitor to ground.

Prediction: the resistor, capacitor, values, and net incidence pass the common graph check. The output assigns `ADC_AVDD` the power-net kind.

Failure: the runtime rejects that name under its supported-power policy. This is an executable-contract restriction, not an electrical-graph error.

Case 2 — Prediction accepted; artifact unavailable

Run: Intent, seed 20260917; task suffix `a1e42f3309b33fbf`.

Request: place a $120\,\Omega$ termination resistor across `RJ45_OUT_3` and `RJ45_OUT_4`.

Prediction: one `120R` resistor with role `termination` connects the two named nets; semantic selection passes. **Execution:** the GUI log records a failed Add/Place Symbol menu fallback; no validated artifact is produced.

Outcome: joint success fails. Raw ERC is unavailable, not electrically clean. The executor failure remains in the fixed denominator.

Case 3 — Topology preserved; native ERC fails

Run: Intent, seed 20260915; task suffix `1ae8e2eb370efee7`.

Required branch: a 10k pull-up from `CHIP_PU` to `+3.3V`, and a `1uf/50V` capacitor from `CHIP_PU` to ground.

Saved artifact: prediction- and reference-relative topology both pass. Native ERC reports two undriven-power-pin errors. The local fragment preserves the requested topology without certifying complete electrical closure.

Case 3 — Saved native artifact

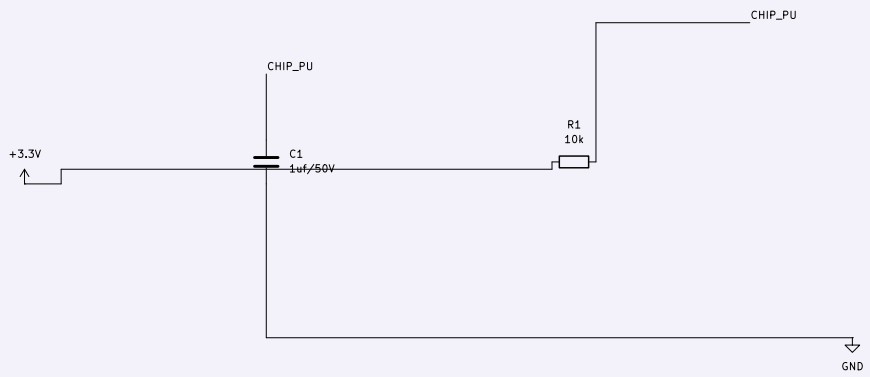


Figure 6: The saved KiCad artifact for Case 3, exported without changing its layout or electrical content. Repeated `CHIP_PU` labels identify the same net. The crossing without a junction is not a connection. The original layout includes a wire crossing the capacitor's value text; topology and ERC checks do not measure this visual defect. The retained report has two undriven-power-pin errors despite passing both topology comparisons.

F.2 Attempts and Failures

Under the original protocol, first-demonstrated/effective joint topology counts differ only for Intent16 (42/44) and FullSpec16 (33/34). The other conditions match: Intent15 41/41, Intent17 42/42, FullSpec15 35/35, and FullSpec17 36/36; all denominators are 60. Two Intent16 retries have external-desktop-interference records. A prior FullSpec16 attempt was interrupted during a user-requested pause. Original attempts are retained. Missing first-attempt evidence is neither imputed as success nor reported as an observed electrical mismatch.

The five original artifact failures comprise menu failures in Intent17, FullSpec16, and FullSpec17; a missing resistor visual body in FullSpec16; and a child process without a result record in FullSpec16. The two parsed topology failures concern the same task in FullSpec15 and FullSpec16. No generic retry-until-success policy is used.

The incremental value-aware queue completed all 39 planned case–condition executions. Every incremental execution passes backend consistency; the final fixed-denominator join combines these results with hash-verified original artifacts. The earlier externally interrupted partial attempt remains archived and is not counted as an additional execution.

F.3 Reference and Verifier Checks

Oracle60 has 60 first-attempt parsed artifacts and exact-pin passes, 59 frozen gold-graph passes, 19 error-free reports, and zero violation-free reports. The gold-graph discrepancy concerns `VMETER{slash}2`: an artifact helper decodes `{slash}` on only one side. Preserving the raw spelling yields 60/60; applying that diagnostic to all 234 original model artifacts changes none of their verdicts.

Reference data are separate from predictions, but the original parser/comparison code is shared. On two retained parsed artifacts, mutation checks reject changed values, removed or renamed net membership, and merged nets. A saved-file check modifies copies of a single-resistor schematic: changing its Value property or deleting the first wire is rejected after reparsing, while the unmodified file passes with unchanged hash.

As an independent saved-file check, KiCad 10's native netlist exporter processes all 60 hash-matched Oracle artifacts. Its XML component identities, values, and pin-to-net partitions match the frozen executable targets in all 60. Literal net-name strings match in 59; for the remaining case,

both the source schematic and saved artifact contain VMETER{slash}2, which KiCad exports as /VMETER/2. Applying that observed export naming rule yields 60/60 net-name matches. This check does not alter frozen scores or assess whether each reference captures its public request.

The initial post-freeze native export of all 60 original source schematics found 51 exact selected-component/value/net signatures. It classified four flags as material source–target differences (REF019, REF021, REF025, REF026), and five as net-name or value-notation issues. The source project's full schematic confirms that REF021's selected R33 lies between an amplifier output and RAMP_OUT, not ramp_pre. Excluding these four flags from the original ValueSemantics v1 rows changes the three-seed joint reference-topology totals from 135/136 (SemanticIntent/FullSpec, 60 tasks per seed) to 126/124 (56 tasks per seed). The original scores are preserved. The later source review below resolves REF006, previously a possible alias, as an additional source error.

On 26 September, author-authorized, AI-assisted source adjudication produced a corrected release: REF019 uses the actual divider midpoint (explicitly named divider_mid); REF021 uses the amplifier output node (ramp_amp_out); REF025 changes the lower resistor from 200 Ω to 10 kΩ; and REF026 changes the pull-up rail from VBUS to VCC_3V3. The new names for unnamed extracted nodes have explicit source-pin mappings. REF021 instance references and units were resolved before native export; REF025 reference strings differ between stored project instances, so its parts are tracked by UUID. All four revised source signatures and reference-artifact native netlists match their targets. Split checks find no new project, source, artifact, or semantic-signature overlap with Mixed302/Dev25. The revision was frozen before new inference, which used hash-matched original adapters and unchanged system prompts and decoding. This intermediate 60-task result is retained in Table 14; Table 3 reports final Evaluation59 after the adjudication below. This technical adjudication is distinct from independent two-expert reference review.

Corrected-release execution records. Before the matched rerun, reference preparation exposed a closed properties panel, and two provisional model attempts used a different primary display (one topology failure and one pass). A further display-calibration attempt was interrupted. These records remain separate. The properties panel and original built-in-display geometry were restored and a gold-input environment check passed before all 16 eligible predictions were executed once in that fixed environment. No prediction was edited or selected by execution outcome.

Native KiCad exports match the corrected component values and named-net memberships in all 16 saved model artifacts, providing a check separate from the custom parser.

Per-seed scores and paired comparisons appear in Appendix E.1.

F.4 Independent Source-Reference Review

Two reviewers submitted separate, confirmed reports dated 26 September 2026 on the same corrected Evaluation60 file (SHA-256 prefix 6537554a98bc6283). Both reports describe independent review of the public request, the selected source subgraph, and the two target representations using prepared evidence. This is distinct from the 97-triple prediction-equivalence evaluation. All 60 packaged source hashes match the task records. The supplied judgments yield 56 shared passes, one shared failure, and three disagreements (Table 13), with agreement on 57/60 cases.

Source-reference review

Table 13: Independent judgments on the corrected 60-task release. A and B are anonymous reviewer identifiers; disagreement is preserved.

Record	PASS	FAIL	UNRESOLVED
Reviewer A	56	1	3
Reviewer B	59	1	0

Both reject REF006. For REF048–050, A records UNRESOLVED and B records PASS with a scoring limitation; the other 56 cases receive PASS from both.

REF006: source–target conflict. The source's two 1 MΩ resistors R8/R9 meet at `Net-(U1D-+)`, connected to U1 pin 12. Its existing `IN+` net instead connects to U1 pin 5, C4, J2, and R14. The public request and both targets call the divider midpoint `IN+`; this is not an alias for an unnamed source node. Both reviewers reject the mapping. Final Evaluation59 excludes this task from all main metrics for both contracts and every seed; the frozen 60-task release remains archived.

REF048–050: net-name policy and scorer coverage. The requests use a Unicode minus (U+2212) for the negative terminal; the source and target identifiers `485-`, `D15-`, and `DMX_D-` use an ASCII hyphen (U+002D). Both reviewers find the selected values and connections consistent. A leaves exact-label closure unresolved; B accepts the typographic interpretation. The frozen graph scorer recognizes explicit net names by literal occurrence in the request. In a read-only mutation check, replacing each target's negative-net label with `WRONG_NET` still passes the labeled common-graph comparison. Thus reference acceptance does not establish that this scorer checks these three exact labels. This exposed a scoring gap as well as a false-negative path: all three frozen FullSpec REF050 predictions use the request's Unicode spelling and were rejected by the earlier comparison. Final NetName v2 resolves both paths uniformly; the same wrong-label mutations now fail. The three newly eligible predictions are executed without changing their output or the backend. Raw reviewer judgments remain unchanged; retention under the explicit naming policy is the authors' adjudication.

The reviews also distinguish visible +3.3 V notation from native `+3V3` names in REF027/039, and schematic connectivity from assembly fields in REF043/056. PASS concerns the selected subgraph; it does not certify assembly, measured behavior, or label scope after reinsertion into the complete project. The corrected REF019/021 names remain documented task-local aliases.

Reference-review sensitivity

Table 14: Historical corrected-60 results and exclusion sensitivity under the earlier net-name rule (before NetName v2). Counts pool three seeds descriptively; n denotes task–seed observations. Graph, contract, saved artifact, and joint topology use value-aware eligibility.

Population	Contract	n	Graph	Valid	Saved	Joint
All 60	SemanticIntent	180	134	134	133	133
	FullSpec	180	141	137	135	133
Exclude REF006	SemanticIntent	177	131	131	130	130
	FullSpec	177	138	134	132	130
Both-reviewer PASS	SemanticIntent	168	122	122	121	121
	FullSpec	168	132	128	126	124

The last population excludes REF006 and REF048–050. These historical subsets reuse earlier outcomes. Final Evaluation59 additionally applies NetName v2 and executes its newly eligible predictions; its main results are in Table 3.

F.5 Native Report Configuration

The 234 original model reports, 60 Oracle reports, 16 corrected-rerun reports, and three final-policy reports record KiCad 10.0.0, both error and warning severities, and the same ignored checks: `footprint_filter`, `four_way_junction`, `simulation_model_issue`, and `single_global_label`. Error-free and violation-free refer to that configuration, not every possible KiCad check.

Native ERC — Finding counts

Table 15: Native ERC finding counts in the 234 original parsed model artifacts and 60 Oracle artifacts. Entries count findings, not failing cases; the 39 incremental executions are excluded.

Condition	Undriven power	Isolated label	Disconnected pin	Wire endpoint
Intent15	39	54	0	0
Intent16	42	58	0	0
Intent17	42	53	0	0
FullSpec15	45	37	2	1
FullSpec16	41	40	1	2
FullSpec17	44	38	0	0
Oracle60	61	71	0	0

Findings in Table 15 are not interchangeable with failing-case counts. Native error-free status agrees between Intent and Oracle for every originally available matched artifact. FullSpec differs on the same I^2C task in all three seeds because of its ordinary-label versus reference-power-symbol choice. Comparisons condition on artifact availability.

G Reproducibility

Versioned experiment records retain source/task identities, requirements, targets, prediction hashes, scorer versions, eligibility lists, attempts, and artifact/native-report hashes. The original summaries, ValueSemantics transitions, incremental eligibility manifest, corrected-release predictions and joins, and final Evaluation59/NetName v2 retain separate version identities.

G.1 Reproducibility Manifest

Table 16 provides abbreviated identifiers; the archived evidence manifest retains complete SHA-256 hashes and source-relative paths. Per-file hashes identify the retained code bytes independently of repository state. The retained protocol specifies model configuration hashes and paths; upstream model and tokenizer revisions are not established by these identifiers.

Reproducibility — Recorded identities

Table 16: Reproducibility anchors. Hashes are SHA-256 prefixes, not replacement version labels.

Item	Recorded identity
Models	Qwen3-8B; supplementary Qwen3-14B
Mixed302/Dev25 freeze manifest	`8454219a8fc7d7c4`
Evaluation60 v9 integrity manifest	`bf8c921acf75655e`
Source-corrected Evaluation60 manifest	`a4cabbb7c824171a`
Final Evaluation59 task file	`ba64c301129f9ec2`
NetName v2 evaluation wrapper	`6199cf140cd9636a`
59-task paired-analysis adapter	`f1fb2835d75acfb7`
Common graph scorer	`cb59ac6c7ed132e8`
ValueSemantics v1 implementation	`29a0463e34326125`
Compiler entry point	`6f97df2b36bdee2e`
Semantic bridge entry point	`433fd3561873a3b4`
Native ERC runtime	KiCad 10.0.0
8B seeds	20260915, 20260916, 20260917
Artifact identity	Per-artifact and raw ERC SHA-256

The evidence inventory includes frozen inputs and targets, lineage records, both original score tracks, the 97-triple packet, submitted A/B consensus columns, the user-confirmed review protocol, paired transitions, original and incremental selection, artifacts, native reports, attempts, and timers. For the 97-triple study, the submitted material is a merged A/B table; separate completed original forms and original timestamps were not supplied. The later 60-reference audit has two separately confirmed reports dated 26 September; anonymized summaries, verdicts, and exclusion counts accompany the manuscript source. Reviewer B states that all cases except REF006 pass; the normalized case table records that explicit summary rather than inventing itemized original responses. Blank forms are not presented as completed reviews. The local archive preserves the named originals separately from the anonymous source package.

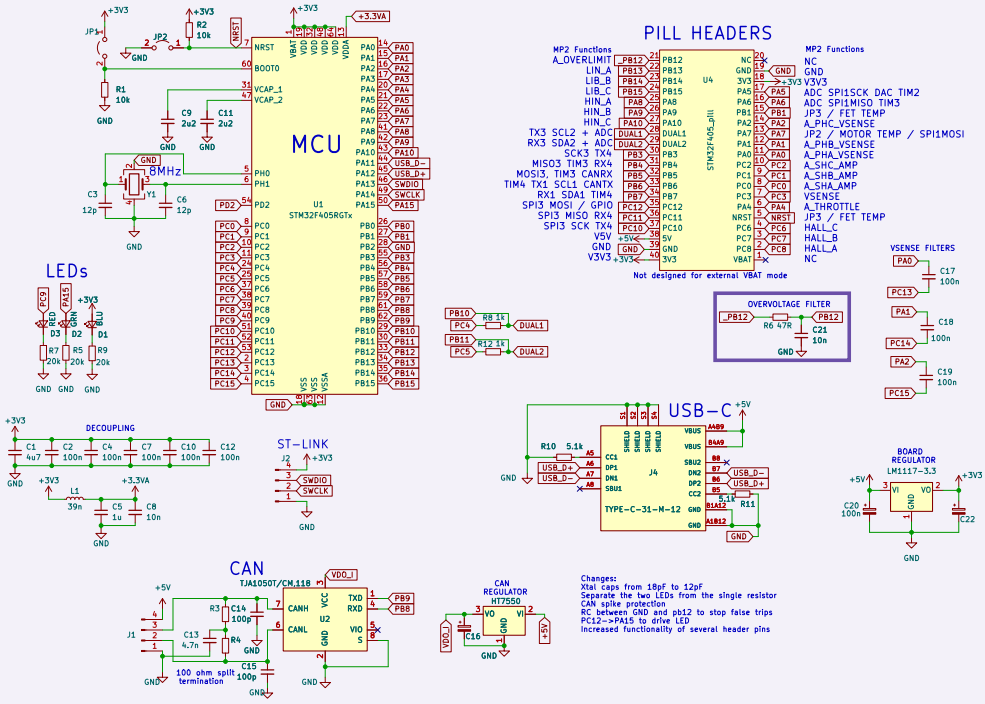
H Source Schematics and Evaluation60 Tasks

The full source sheets show where the extracted functional units occur in larger circuits. Purple outlines locate each source detail; the enlarged inset retains the original components and connections.

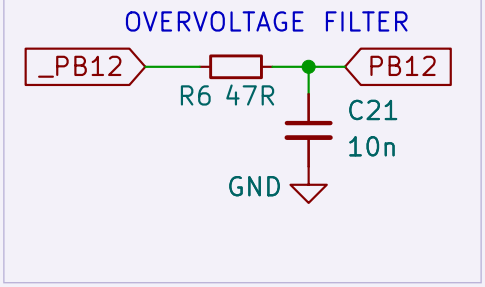
Full source circuit — STM32 control board

REF043 | STM32 control board

Source: davidmolony/F405_pill | Commit 7c7c8d3cd073



ENLARGED SOURCE DETAIL



EXTRACTED TASK

Suppress false trips on PB12 with a 47 Ω resistor between _PB12 and PB12, and a 10 nF capacitor from PB12 to GND.

Source components: R6, C21

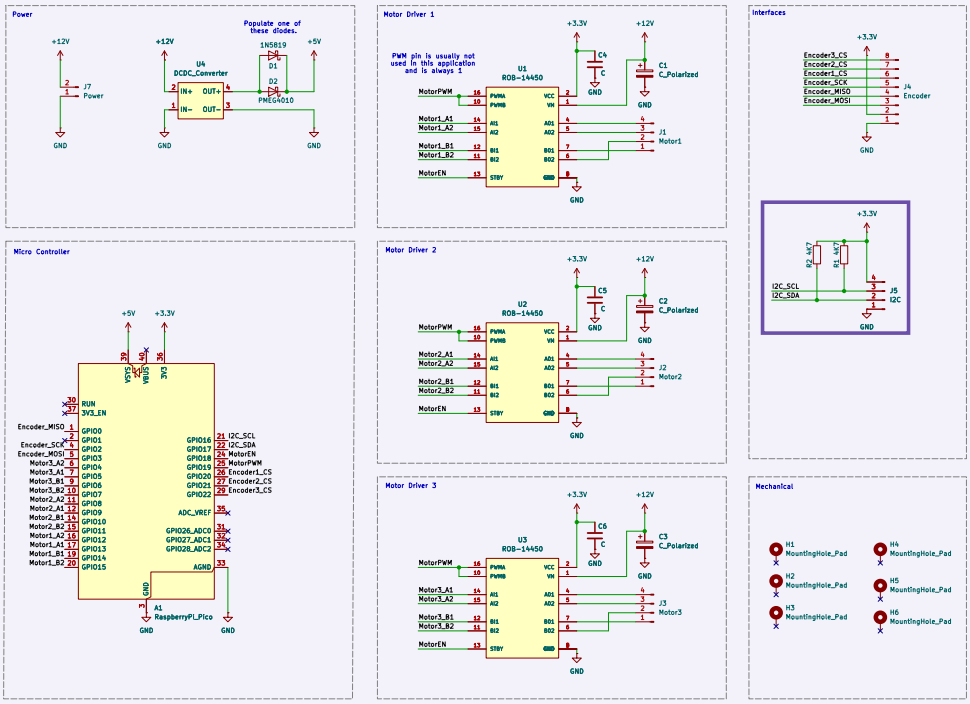
Reference topology: RC signal filter in the task gallery.

Figure 7: REF043 in its full STM32 control-board source sheet. The selected R6–C21 branch connects `_PB12`, `PB12`, and ground. The inset enlarges the original source; Figure 10 shows the task's reference topology.

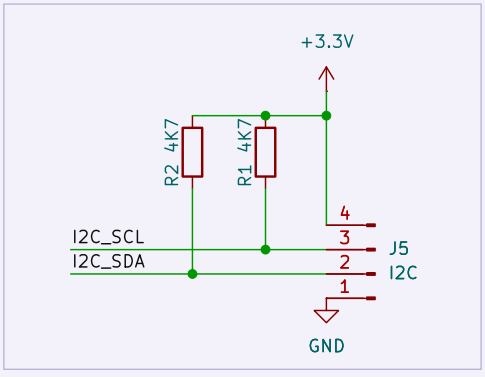
Full source circuit — Pico motor controller

REF009 | Pico three-motor controller

Source: 0x23/MicroManipulatorStepper | Commit 63d960f8a5b2



ENLARGED SOURCE DETAIL



EXTRACTED TASK

Provide pull-up support for I2C_SDA and I2C_SCL from +3.3V: use a 4K7 resistor from I2C_SDA to +3.3V and a 4K7 resistor from I2C_SCL to +3.3V.

Source components: R1, R2

Reference topology: Dual I2C pull-up in the task gallery.

Figure 8: REF009 in its full Pico motor-controller source sheet, including power, three motor drivers, and interfaces. Source R1 and R2 provide the two I^{2}C pull-ups. The connector remains visible in the source detail; the extracted task covers the two resistors (Figure 9).

Figures 9–10 show one task per Evaluation60 family, selected for structural coverage and legibility independently of model outcomes.

Task examples — Interface, divider, and pull-up

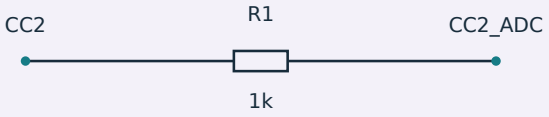
PUBLIC REQUEST AND PROVENANCE

REFERENCE TOPOLOGY (REDRAWN)

REF036 Series signal link

Add a 1 kΩ series resistor in the signal path between CC2 and CC2_ADC.

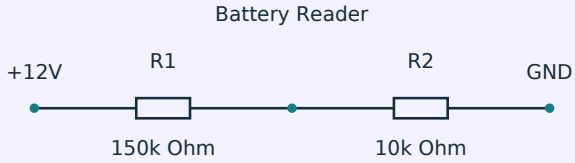
Source: cr1tbit/eaaeoe
Commit 75110081079f; source R10



REF025 Voltage divider

Build a two-resistor voltage divider from +12V to GND, using a 150k Ohm resistor from +12V to Battery Reader and a 10k Ohm resistor from Battery Reader to GND.

Source: DrGlaucous/DungeonMaster
Commit e8cd240e09c8; source R1, R2



REF009 Dual I2C pull-up

Provide pull-up support for I2C_SDA and I2C_SCL from +3.3V: use a 4K7 resistor from I2C_SDA to +3.3V and a 4K7 resistor from I2C_SCL to +3.3V.

Source: 0x23/MicroManipulatorStepper
Commit 63d960f8a5b2; source R1, R2

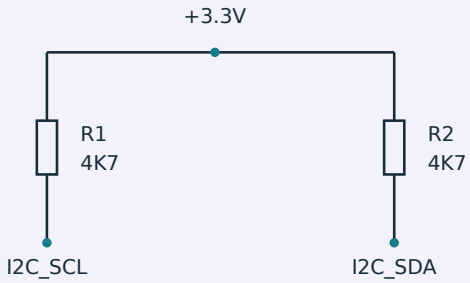


Figure 9: Interface support, voltage divider, and pull-up examples. Each public request is paired with its frozen reference topology, redrawn with local component references and unchanged net names and values. Source project, commit, and original component references identify the provenance; REF025 uses the corrected release.

Task examples — Control, termination, and filter

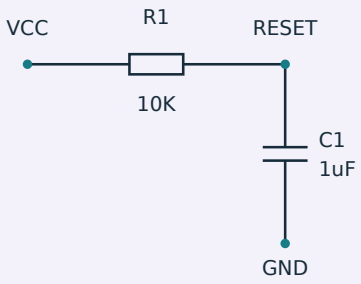
PUBLIC REQUEST AND PROVENANCE

REFERENCE TOPOLOGY (REDRAWN)

REF008 Reset RC support

Provide control support for RESET: use a 10K resistor to pull it up to VCC, and connect a 1uF capacitor from RESET to GND.

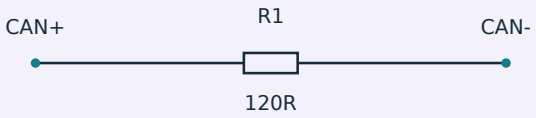
Source: spacelab-ufsc/slcam
Commit ac5c6166ff6b; source C1, R1



REF024 Differential termination

Place a 120 Ω termination resistor directly between CAN+ and CAN-.

Source: eccentricOrange/can-daq
Commit 71c056387712; source R5



REF043 RC signal filter

Suppress false trips on PB12 with a 47 Ω resistor between _PB12 and PB12, and a 10 nF capacitor from PB12 to GND.

Source: davidmolony/F405_pill
Commit 7c7c8d3cd073; source R6, C21

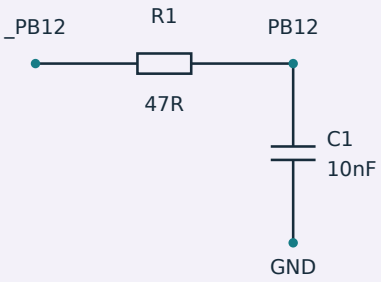


Figure 10: Control support, termination, and signal filter examples, using the same reference-topology presentation as Figure 9. The circuits illustrate the functional units specified by the public tasks.