PICAGENTBENCH: BENCHMARKING MULTIMODAL AGENTS FOR PHOTONIC INTEGRATED CIRCUIT DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents for photonic integrated circuit (PIC) design must interpret visual and structural evidence, reason about constraints, and produce valid design artifacts. Final-artifact success alone provides limited insight into which engineering tasks agents perform reliably or where errors arise. We introduce PICAgentBench, a multimodal benchmark with a selected bank of 982 cases across six separately scored PIC engineering tasks. These tasks comprise requirement-to-design, device recognition, circuit recognition, layout-to-netlist extraction, fault localization, and fault repair. They place overlapping demands on perception, reasoning, and action through task-specific combinations of specifications, visual layouts, and structured circuit representations. Fault repair evaluates constrained geometric editing given a local crop and known fault type. Expert-developed checking scripts and manual review support reference construction, with explicit observation and verification boundaries defining what each score measures. A shared agent environment supports model comparisons under common interaction conditions, while task-specific acceptance criteria define six-task performance profiles. Separate native metrics distinguish device classification from localization and circuit identity from potential function. Together with task-level breakdowns and resource reporting, these profiles provide a structured basis for identifying model strengths and limitations in PIC design.

1 INTRODUCTION

Large language model (LLM) agents for photonic integrated circuit (PIC) design must interpret specifications and circuit representations, reason about constraints, and produce valid design outputs. These operations draw on overlapping demands in *perception*, *reasoning*, and *action*. A single aggregate outcome can conceal whether an agent misidentified a component, misunderstood connectivity, or made an invalid edit. Useful evaluation therefore requires separately scored tasks with explicit inputs, outputs, and acceptance criteria. Such measurements support analysis of recognition, design reasoning, and fault diagnosis and repair under shared interaction conditions. Task-level scores and error breakdowns characterize performance within those conditions.

PIC design connects mathematical descriptions of optical behavior to geometric implementations across components, subsystems, and architectures. At the device level, electromagnetic, thermal, and electrical simulations relate heterogeneous component geometries to coupled physical behavior. At the circuit level, compact models combine device responses through a topology to predict spectral response (Bogaerts & Chrostowski, 2018). At the system level, subcircuits compose into architectures for computing, communications, radio-frequency photonics, sensing, and quantum photonics (Bogaerts et al., 2020). Design therefore translates high-level requirements into device choices, topology, and geometry, then interprets and checks the resulting schematics and layouts. For example, in a Mach–Zehnder interferometer, couplers split and recombine light, while the relative phase accumulated along two arms controls interference at the outputs (Bogaerts et al., 2020). Recognizing that topology, selecting phase settings for a target response, extracting connectivity, and repairing a geometric discontinuity while preserving the ports require different observable outputs. Evaluating these operations separately makes their correctness and failure modes explicit.

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.

Table 1: **Scope coverage across related works.** Engineering scopes group related objectives; inputs and evaluation criteria differ across works. CircuitSense addresses electronic circuits; the other works address PICs.

Scope Coverage	PICBench	PhIDO	PICasso	PICopilot	PixCell	CircuitSense	Ours
PIC engineering domain	✓	✓	✓	✓	✓	–	✓
Multimodal inputs	–	–	–	–	✓	✓	✓
Requirement-to-design	✓	✓	✓	✓	○	✓	✓
Device recognition	–	–	–	–	–	✓	✓
Circuit recognition	–	–	–	–	–	✓	✓
Layout-to-netlist extraction	–	–	○	–	–	✓	✓
Fault localization	–	○	○	–	○	–	✓
Fault repair	○	○	○	○	○	–	✓

✓: explicitly defined scope; ○: related operation within a broader workflow; –: not reported under these criteria. *Multimodal inputs*: text with model-visible images for at least one objective, optionally with other files. Section 2 defines PICAgentBench; Appendix C.10 details comparison criteria and source locators.

Related works address important parts of this design process. PICBench evaluates generated photonic netlists with syntax and simulated functional checks (Wu et al., 2025). PhIDO produces layout masks through staged synthesis, while PICasso combines synthesis, verification, and optimization (Sharma et al., 2025; Vungarala et al., 2026). PICopilot uses retrieval-assisted Python scripting and provides its own benchmark and model comparisons (Jiang et al., 2026). PixCell reconstructs component images as executable parametric geometry programs with verification-guided revision (Agarwal et al., 2026). CircuitSense separately assesses electronic-circuit perception, analysis, and design (Akbari et al., 2026). These objectives motivate complementary PIC coverage across recognition, design reasoning, connectivity extraction, fault localization, and constrained repair, each with a distinct output contract. Table 1 compares engineering scope; Appendix C.10 qualifies the comparison criteria, and Section 5 discusses the different objectives.

We introduce PICAgentBench, a multimodal dataset and benchmark comprising 982 task cases. Its six tasks are **requirement-to-design** (TASK-1), **device recognition** (TASK-2), **circuit recognition** (TASK-3), **layout-to-netlist extraction** (TASK-4), **fault localization** (TASK-5), and **fault repair** (TASK-6). Source designs can contribute multiple cases; scored cohorts are specified separately. Construction combines literature and textbook knowledge with PhotonForge (Flexcompute, n.d.a) for schematics, layouts, and component models, and Tidy3D (Flexcompute, n.d.b) for electromagnetic simulation interfaces. Experts develop task-specific checking scripts and then manually double-check the cases. Controlled modifications of reference layouts produce faulty examples, with clean controls included for localization. TASK-1 evaluates analytical and constraint-based design answers; TASK-3 supplies graph and context files for separate circuit-name and potential-function judgments; TASK-6 supplies a localized faulty crop, known fault type, and repair constraints. Task-native metrics and declared analytical, graph, geometric, and semantic criteria define acceptance, with details in Sections 2–3. Models interact through a shared agent environment with specified observations, tools, and resource policies.

The benchmark makes three contributions:

- **Six-task resource and traceable construction.** Source knowledge and implementations connect to task-specific observations and targets across six engineering operations, with traceable source and case relationships and controlled clean/fault variants for localization and repair.
- **Explicit evaluation and validation.** Task-native metrics, expert-developed checks followed by manual double-checking, and a shared agent protocol define how outputs are assessed. Complete-task success supplements native metrics where acceptance gates and denominators are specified.
- **Task-level performance, error, and resource reporting.** A reproducible reporting framework organizes model-by-task performance, native-metric and error breakdowns, difficulty and family slices, and measured resource use. Exact tables accompany a compact six-axis radar profile, with missingness, denominators, and uncertainty kept explicit.

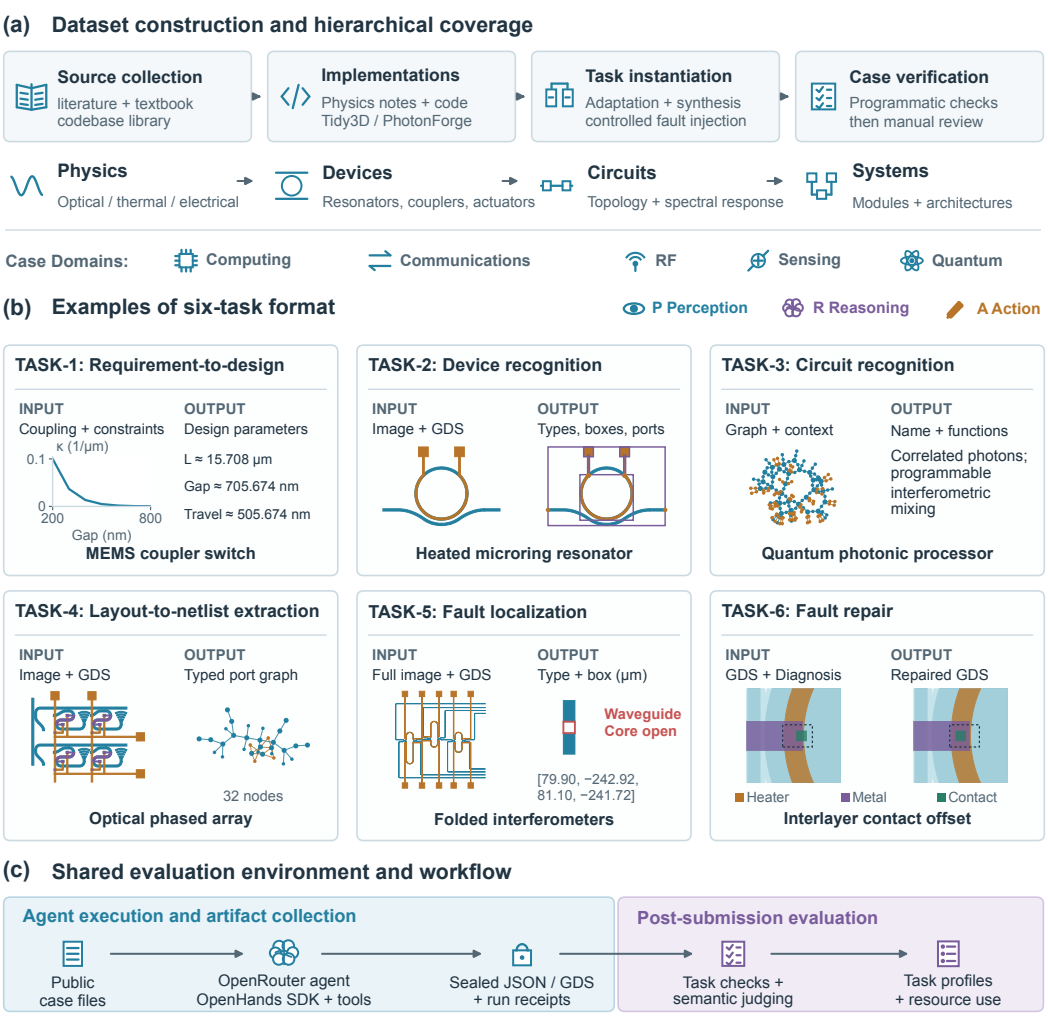

Figure 1: **PICAgentBench overview: construction, tasks and evaluation.** **(a)** Dataset construction, four hierarchy levels and five domains. **(b)** Six task formats illustrated with dataset inputs and reference outputs. P/R/A denote Perception, Reasoning and Action. TASK-3/4 graphs show node-level connectivity; TASK-5 magnifies the waveguide core. TASK-6 shows contact-layer alignment within the dashed edit boundary. **(c)** Agent execution and artifact collection precede private post-submission evaluation.

2 PICAgentBench

PICAgentBench evaluates agents through six separately scored engineering tasks. Figure 1 connects their source material, representations and evaluation interfaces. The resulting profile describes performance under explicit task conditions, with native metrics exposing errors that a single success rate would conceal.

2.1 Data Sources and Representations

We combine textbook foundations with device, circuit and system designs from research literature. The foundation library draws on electromagnetics and silicon-photonic design texts (Staelin, 2011; Chrostowski & Hochberg, 2015). Device sources include multimode-interference couplers (Soldano & Pennings, 1995) and thermo-optic phase shifters (Harris et al., 2014). Circuit and system sources cover five domains: coherent neural networks for computing (Shen et al., 2017), wavelength-multiplexed light sources for communication (Purnawirman et al., 2017), programmable microwave

photonic circuits for RF (Wei et al., 2025), reconstructive spectrometers for sensing (Yao et al., 2023), and programmable multiphoton processors for quantum photonics (Arrazola et al., 2021).

Each source package links research notes and physics descriptions to implementation code and Python evaluation, simulation and verification tools. Parameters distinguish reported values, derived quantities, adaptations and assumptions. PhotonForge supplies schematic, layout and component-model interfaces, while Tidy3D supplies electromagnetic simulation interfaces. These implementations are reconstructions or adaptations, not recovered author masks. Their evidence records distinguish analytical calculations, geometric checks, simulated responses and literature-reported measurements (Appendix A).

Task instances expose the representations needed for their objectives: requirements and diagrams, rendered layouts and GDSII geometry, or attributed graphs with local context. Initial instructions attach the designated image where applicable and identify local files; their raw contents are accessed through tools. Reference answers, pristine repair targets and grading code remain evaluator-only. Thus, aligned representations support several tasks without giving every task the same observations or revealing its answer.

2.2 Six Tasks and Evaluation Metrics

We denote the six tasks as TASK-1 through TASK-6 (Table 2). *Perception* concerns interpreting visual and spatial design evidence; *reasoning* concerns physical constraints, connectivity, function and faults. *Action* concerns modifying design artifacts under constraints, assessed most directly through TASK-6's GDS repair. These demands overlap; task scores do not identify independent latent abilities. Shared file inspection and computation are available to all tasks, without a separate tool-use score. TASK-1–TASK-5 return structured JSON, while TASK-6 returns a repaired GDSII file.

Table 2: **Inputs, outputs and evaluation metrics for the six tasks.** Task instructions accompany the listed input evidence. Outputs are JSON answers (TASK-1–TASK-5) or a repaired GDSII file (TASK-6). Metrics within each row are reported separately; Appendix C details schemas, matching rules and complete-task acceptance.

Task	Input evidence	Output artifact	Evaluation metrics
TASK-1	Requirements; essential diagram for visual cases	Case-specific design parameters	Analytical constraint/optimum acceptance; foundation coverage
TASK-2	Layout image and GDS; device ontology and annotation rules	Device labels, image bounding boxes and typed ports	Device-type F1; image-box F1; port F1
TASK-3	Circuit graph, context and device library	Circuit name and supported potential functions	Circuit-name accuracy; potential-function accuracy
TASK-4	Layout image and GDS; process/device rules and named terminals	Attributed optical/electrical port graph	Joint typed-port graph isomorphism
TASK-5	Full-layout image and GDS; process rules and neutral inventory	Fault types, physical bounding boxes and layers; clean-layout flag	Type and localization exact-set accuracies; clean-case false positives per layout
TASK-6	Faulty crop image and GDS; known fault type and local repair constraints	Repaired GDSII file	Repair acceptance under geometry and preservation constraints

Native metrics retain distinct failure modes: a correct TASK-2 box can accompany a wrong device label, and a correct TASK-5 location can accompany a wrong fault type. TASK-3 judges architectural identity and supported potential functions separately, without presuming a configured operating state. For the compact six-axis success profile, let $\bar{y}_{ti}$ be case i's mean complete-task acceptance over its prespecified repeats. For N_t cases in the declared reporting population,

$$S_t = \frac{100}{N_t} \sum_{i=1}^{N_t} \bar{y}_{ti}, \qquad \mathbf{S} = (S_1, \ldots, S_6). \tag{1}$$

This averages repeats within cases before averaging cases. Repeat allocation, scoring eligibility and missingness follow Section 3; unavailable or unadjudicated outcomes remain distinct from evaluated failures. Fixed 0–100% radar axes accompany exact native-metric tables, denominators and resource summaries; radar area and averages of heterogeneous native metrics are not benchmark scores.

2.3 Hierarchical Construction and Task Instantiation

Construction connects devices, subcircuits and circuits through parameterized geometry, named ports and explicit connectivity. This hierarchy organizes artifacts rather than imposing a sequence or difficulty ranking on the tasks. TASK-1 derives analytical questions from physical models and source knowledge, including visual cases whose diagrams supply essential information. TASK-2 uses annotated device instances and mixed scenes. TASK-3 packages circuit graphs and curated functional context; TASK-4 pairs layouts with attributed reference graphs. These branches retain their own schemas and checks instead of relying on one universal case generator.

For TASK-5/TASK-6, controlled modifications of saved reference layouts produce faulty observations with recorded categories, layers and spatial targets. TASK-5 includes clean controls; TASK-6 exposes a local crop and repair intent from the same fault pair. Valid alternative repairs are accepted under the declared constraints. Source, parent and pair identities preserve dependence across variants; a family-level TASK-3 link is not asserted to be exact variant ancestry. Appendix A details these construction records.

2.4 Quality Control and Benchmark Validity

Experts develop task-specific scripts to check case correctness programmatically and then manually double-check the cases. Checks address source interpretation, artifact and annotation consistency, observation sufficiency, and acceptance of valid alternatives versus rejection of invalid outputs. Case manifests bind public inputs, references and evaluators by hashes. The saved qualification records include 852 TASK-5 scoring controls and 918 TASK-6 repair controls. These establish behavior under declared analytical, graph and geometric predicates, separately from model performance, foundry-qualified signoff and physical-response validation (Appendix E.6).

2.5 Dataset Statistics and Coverage

Table 3 reports the manifest-derived development inventory. Cases, recorded source groups, saved parents, observations and model episodes are different counting units. Paired faults and repairs share source material; multiple views or annotations do not create additional independent designs. The selected bank contains 982 task cases, while each evaluated cohort and its scoring eligibility are specified separately in Section 3.

Coverage and difficulty are described within tasks. In particular, TASK-5's parent-scene labels differ from TASK-6's provisional crop-process labels; neither is a calibrated measure of model difficulty. Appendix A.7 records family/subtype coverage, lineage and exclusions. These development cases support declared task-level breakdowns, without being relabeled as a source-held-out population.

Table 3: **Selected dataset inventory.** Counts refer to task cases; recorded groups retain each task's source grouping. Image inputs are accompanied by task-specific text and structured files.

Task	Cases	Groups	Coverage	Input representations
TASK-1	140	34	14 case families	Text, diagrams of design
TASK-2	383	21	17 case families	Layout image, GDS, rules
TASK-3	125	25	25 case families	Graph and context
TASK-4	118	14	14 case families	Layout image, GDS, rules
TASK-5	114	14	28 fault types; 41 subtypes	Layout image, GDS, rules
TASK-6	102	14	28 fault types; 41 subtypes	Crop image/GDS, known fault
Total	982	*PIC application domains: computing, communication, RF, sensing, quantum.		

3 EXPERIMENTS

We evaluate the six-task profiles using thirteen models in two groups. The **seven flagship models** are Claude Opus 5.5 (Anthropic, 2026), GPT-6 Astra (OpenAI, 2026), Gemini 3.8 Flash (Google DeepMind, 2026), DeepSeek V4.1 Flash (DeepSeek-AI, 2026), Kimi K3 (Moonshot AI, 2026), Qwen3.8 Max (0902) (Alibaba Cloud, 2026), and MiniMax M3 (MiniMax, 2026).

The **six compact open-weight models** are Qwen3.8 27B (Qwen Team, 2026c), Qwen3.6 35B A3B and 27B (Qwen Team, 2026b;a), Gemma 4 31B and 26B A4B (Gemma Team, 2026), and Ministral 3 14B (2512) (Mistral AI, 2025). Table 12 records the exact API identifiers and provider routes for this panel.

3.1 EVALUATION SETUP

Our protocol compares LLM-based agents across six separately scored tasks under a shared file-and-tool interface. Participant LLM inference uses OpenRouter (OpenRouter, n.d.). OpenHands Software Agent SDK 1.49.2 (OpenHands, n.d.) manages interaction; isolated Docker workers (Docker, n.d.) provide Python 3.12 and scientific/GDS tools on a remote Linux HPC system administered via SSH. Tidy3D and PhotonForge support source construction and the trusted project environment; participant actions use a separately packaged worker toolset. Appendix E details these roles.

The shared agent environment supplies identical public case files and tool semantics across models. Agents can execute commands, inspect images, compact their working context and submit one final artifact. Original inputs are read-only; references and private grading remain outside the worker. The controller preserves request and tool traces, seals the artifact, and then permits independent grading without returning grades to the participant. The protocol specifies context, image-delivery and resource limits; completed coverage and applied settings require the corresponding execution records. TASK-3 semantic outcomes remain unadjudicated until matching, qualified judgment records are available.

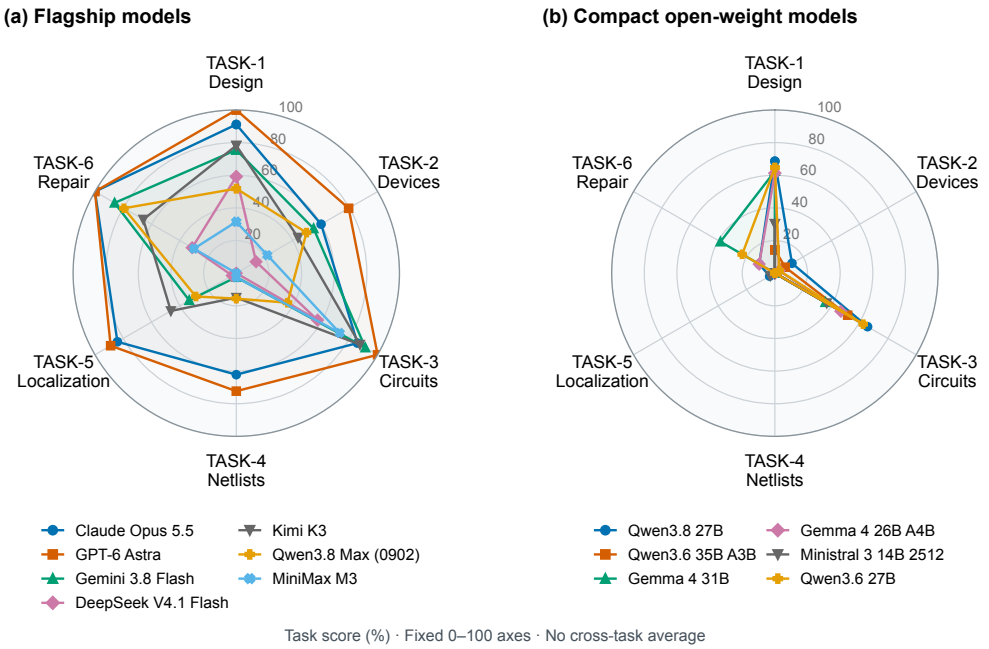

Figure 2: **Six-task performance profiles.** (a) Reported task scores for 7 flagship models. (b) Reported task scores for 6 compact open-weight models. Both panels use fixed 0–100% axes; radar area is not an overall score.

3.2 SIX-TASK PERFORMANCE

Table 4 reports scores by model and task. Native metrics in Table 16 separate device labels, boxes and ports, and circuit names from functions. Figure 2 summarizes the reported six-task scores on fixed 0–100% axes. These coordinates capture overlapping engineering demands, not independent latent abilities; radar area is not a score.

Complete-task rates carry their numerators, denominators and missingness. Evaluated failures remain distinct from unavailable, ineligible and unadjudicated episodes. Incomplete matrices retain all planned cells; rates conditional on resolved outcomes are not reported as complete-cohort performance.

Table 4: **Six-task performance (%).** Task scores correspond to the profiles in Figure 2.

Model	TASK-1	TASK-2	TASK-3	TASK-4	TASK-5	TASK-6
Claude Opus 5.5	91.1	60.0	85.7	62.2	84.1	100.0
GPT-6 Astra	100.0	79.5	100.0	72.3	88.9	100.0
Gemini 3.8 Flash	75.5	54.7	91.3	2.4	33.3	86.1
DeepSeek V4.1 Flash	59.2	14.0	57.5	0.0	2.9	31.2
Kimi K3	78.2	43.7	87.5	15.0	46.3	65.8
Qwen3.8 Max (0902)	51.8	49.7	36.2	15.6	28.6	79.5
MiniMax M3	31.5	22.0	73.2	2.6	0.0	30.0
Qwen3.8 27B	68.6	12.1	65.6	0.0	3.7	11.1
Qwen3.6 35B A3B	14.3	7.3	51.5	0.0	0.0	0.0
Gemma 4 31B	62.5	0.0	35.7	0.0	0.0	38.5
Gemma 4 26B A4B	61.3	3.0	46.7	0.0	0.0	11.1
Ministral 3 14B 2512	30.3	0.0	36.7	0.0	3.3	0.0
Qwen3.6 27B	64.7	3.1	62.5	0.0	0.0	23.1

3.3 TASK-LEVEL BREAKDOWN AND ERROR ANALYSIS

Tables 5 and 6 report easy, medium and hard subgroups for Perception (TASK-2, TASK-4 and TASK-5) and Action (TASK-6). These task groupings retain overlapping capability demands and task-specific difficulty labels.

Breakdowns relate profiles to native submetrics, task-specific difficulty, device/circuit families and observable errors (Table 17 and numerical summaries). TASK-1 retains separate foundation and visual strata; TASK-5 separates faulty cases from clean controls and their false positives. TASK-6 assesses local geometry and preservation with supplied fault types and localized crops. Family and difficulty labels, including provisional crop-process tiers, retain task-specific meanings.

Table 20 in Appendix F.4 groups 389 failed submitted results into mutually exclusive categories. Broader episode accounting separates provider availability, submission compliance, task correctness and evaluator failures (Table 19); its diagnostic labels may overlap. Interpretation uses the reference and scorer checks in Appendix E.6. Score gaps alone do not identify cognitive mechanisms. Paired comparisons preserve common case membership and source dependence; group-based uncertainty requires documented groups with adequate support. Optional workflow examples complement the task measurements.

Table 5: **Perception scores (%).**

Model	Easy	Medium	Hard
Claude Opus 5.5	78.2	49.3	61.2
GPT-6 Astra	89.1	86.1	60.0
Gemini 3.8 Flash	65.3	28.8	16.4
DeepSeek V4.1 Flash	20.7	0.0	2.0
Kimi K3	59.4	20.3	27.0
Qwen3.8 Max (0902)	62.2	25.0	18.8
MiniMax M3	29.2	6.5	0.0

Table 6: **Action scores (%).**

Model	Easy	Medium	Hard
Claude Opus 5.5	100.0	100.0	100.0
GPT-6 Astra	100.0	100.0	100.0
Gemini 3.8 Flash	100.0	80.0	82.4
DeepSeek V4.1 Flash	22.2	28.6	37.5
Kimi K3	70.0	50.0	72.2
Qwen3.8 Max (0902)	100.0	60.0	78.9
MiniMax M3	42.9	12.5	33.3

3.4 Accuracy and Resource Use

Resource summaries relate task accuracy to input/generated tokens, tool calls, wall time and known charges on declared common populations (Table 18). All-attempt summaries include failures and retries; successful-case summaries retain their own denominators. Participant and judge usage are accounted for separately. Missing usage stays unknown; reasoning tokens are not double-counted; reservations remain distinct from charges. Tokenizer and serving differences preclude interpreting equal token limits as equal internal compute. Appendix F provides completeness counts, timing definitions and extended metric/error tables.

4 Discussion

4.1 Performance on Curated and Synthetic Data

Curated designs anchor evaluation in identifiable engineering examples, whereas synthesis supplies controlled variations and compositions. Their performance contrast is informative only relative to the evaluated task and case distribution. A difference between pooled scores can arise from task mix, structural complexity, rendering, available evidence, or source familiarity. Comparisons within shared task, hierarchy, and information conditions therefore address a narrower question than the raw corpus-level gap.

Source and composition holdouts help distinguish performance on new instances of familiar templates from performance on unfamiliar assemblies. They complement, rather than replace, the curated–synthetic comparison. Synthetic origin alone does not establish independence from familiar design patterns, and a performance gap alone does not identify memorization. Interpreting the gap requires the source lineage and observation controls defined in Sections 2.3 and 3.1.

4.2 Failure and Recovery Analysis

An incorrect final artifact can arise from a misread observation, an unsupported inference, an invalid action, or a failure to use verification feedback. Diagnosis therefore connects observable intermediate outputs and tool events to the final acceptance checks. The relevant evidence includes the recovered component graph, the supplied fault diagnosis, the applied edit, and any preservation violation. Fluent explanations alone do not establish which engineering operation succeeded.

A positive reference effect indicates that improved supplied information benefits the downstream stage in this environment. A small effect can occur when the downstream stage already recovers the information, when a different bottleneck dominates, or when the intervention supplies little useful evidence. Recovery and regression diagnostics distinguish these possibilities. The controlled information contrast fixes downstream resources; full-workflow accounting additionally includes the cost of producing the intermediate.

Recovery occurs when a downstream stage reaches a valid result despite an incorrect supplied intermediate. A correct intermediate followed by an invalid final artifact instead directs attention to downstream reasoning, execution, or verification. In revision tasks, a regression is an edit that introduces a violation or breaks a previously satisfied requirement. These outcomes distinguish persistent errors from errors that an agent detects and corrects, while retaining all eligible cases in the workflow comparison.

Scope. The interpretation of success remains bounded by the task contract, corpus composition, shared controller, available tools, and verification evidence. Structural checks, simulated behavior, and measured device performance answer different engineering questions. This distinction also limits what failure analysis can attribute to a model when the observations or verifier do not resolve the relevant physical behavior.

5 Related Work

Photonic synthesis and scripting. Photonic design assistants translate natural-language specifications into artifacts that engineering tools can check. PICBench evaluates generated netlists using

syntax and simulated functional checks (Wu et al., 2025). PhIDO produces layout masks through staged synthesis and revises schematics using crossing feedback (Sharma et al., 2025). PICasso combines synthesis, verification, and optimization with its PIC-Set benchmark (Vungarala et al., 2026). PICopilot uses retrieval-augmented generation to produce Python scripts for layout, simulation, and design-rule checking. Its scripting benchmark compares multiple LLMs; internal revision uses static checks and LLM feedback, while final functional assessment executes the scripts (Jiang et al., 2026). These studies contribute evaluation resources as well as generation frameworks. Their success criteria depend on the generated representation and the engineering checks applied to it.

Visual reconstruction and circuit benchmarks. Visual benchmarks place different demands on circuit representations. PixCell reconstructs component images as executable parametric geometry programs, including verifier-guided revision and selected retargeting studies (Agarwal et al., 2026). CircuitSense evaluates electronic-circuit perception, analysis, and design across abstraction levels (Akbari et al., 2026). Their objectives distinguish recovering geometry from recognizing components, extracting connectivity, or reasoning about circuit function. PICAgentBench specifies task-dependent observations and outputs. Requirement-to-design includes analytical and constraint-based answers; circuit recognition uses supplied graph and context files. Layout-to-netlist extraction also permits native GDS access. Its current repair task supplies a localized faulty crop and the fault type, defining a constrained artifact-editing problem.

Capability-level assessment and reuse. PICAgentBench complements these works through six separately scored engineering task contracts with explicit output requirements in a shared agent environment. Task-native metrics, breakdowns by error, difficulty, and circuit family, and resource measurements support comparisons of strengths and weaknesses under explicit scoring rules. Separate output contracts retain distinctions among recognition errors, connectivity mistakes, and repairs that violate preservation constraints. The profiles characterize overlapping perception, reasoning, and action demands, without identifying independent latent abilities. Such measurements could guide improvements to models and design frameworks. Table 1 summarizes domain, input modality, and engineering scope; Appendix C.10 distinguishes standalone tasks from related pipeline operations and documents verification protocols.

6 Conclusion

PICAgentBench provides six separately scored PIC engineering tasks with aligned design representations and explicit acceptance criteria. The tasks impose overlapping demands on perception, reasoning, and action, while their native metrics distinguish analytical design, recognition, structural reconstruction, fault localization, and constrained repair. Source curation, programmatic checks, and manual review support the reference cases. A shared agent environment defines the conditions for comparison, and task-level breakdowns and resource reporting support interpretation within the supplied observation and verification scope. An optional workflow example can illustrate connected use of the task interfaces without replacing the six-task performance profile.

AI USE STATEMENT

We used generative AI to help generate synthetic dataset content, draft manuscript sections and improve the clarity of the writing. OpenAI Codex also assisted with literature discovery and summarization, refinement of task descriptions and evaluation/reporting procedures, and development of scripts for tables and scientific figures. Source citations and task descriptions were checked against primary publications and the benchmark artifacts. Dataset cases are checked using expert-developed scripts and then manually double-checked by domain experts, as described in Section 2.4 and Appendix A.6. The TASK-3 evaluation protocol specifies GPT-6 Astra for post-submission semantic judging (Appendix E), distinct from the models evaluated as benchmark participants (Section 3). The authors take responsibility for the research design, scientific judgments, final claims and submitted artifacts, including material prepared with AI assistance.

REPRODUCIBILITY STATEMENT

The anonymous review repository for PICAgentBench is `https://anonymous.4open.science/r/PICAgentBench-8EA6`. We will publicly release the dataset on GitHub. The documentation separates dataset construction, task evaluation and experimental reporting to support reproducibility. Source designs, representations, construction procedures and lineage are described in Section 2 and Appendix A. Appendix C specifies task inputs, output schemas and scoring rules; Appendix A.7 identifies the selected dataset coverage and counting units. Reference checks and evaluator controls are documented in Appendices A.6 and E.6. Appendix B describes file organization and access. Appendix D records model identifiers, provider routes and resource settings; Appendix E describes software environments, tool interfaces and post-submission grading. Aggregation, missing-outcome handling and resource-accounting definitions are specified in Section 3 and Appendix F. Case manifests and artifact hashes identify the inputs and evaluation code required for replay; the reporting protocol distinguishes the selected inventory from executed and scored cohorts.

REFERENCES

Aadarsh Agarwal, Kenaish Al Qubaisi, and Dirk Englund. From pixels to PCells: A neurosymbolic approach to photonic component creation, 2026. URL https://arxiv.org/abs/2608.00084v1. Version 1.

Arman Akbari, Jian Gao, Yifei Zou, Mei Yang, Jinru Duan, Dmitrii Torbunov, Yanzhi Wang, Yihui Ren, and Xuan Zhang. CircuitSense: A hierarchical MLLM benchmark bridging visual comprehension and symbolic reasoning in engineering design process, 2026. URL https://arxiv.org/abs/2509.22339v2. Version 2.

Alibaba Cloud. Qwen3.8-Max: Model information, 2026. URL https://help.aliyun.com/en/model-studio/qwen3-8-max. Documentation for snapshot qwen3.8-max-0902. Accessed September 26, 2026.

Anthropic. Introducing Claude Opus 5.5, 2026. URL https://www.anthropic.com/claude-opus-5-5. Accessed September 26, 2026.

J. M. Arrazola, V. Bergholm, K. Brádler, T. R. Bromley, M. J. Collins, I. Dhand, A. Fumagalli, T. Gerrits, A. Goussev, L. G. Helt, J. Hundal, T. Isacsson, R. B. Israel, J. Izaac, S. Jahangiri, R. Janik, N. Killoran, S. P. Kumar, J. Lavoie, A. E. Lita, D. H. Mahler, M. Menotti, B. Morrison, S. W. Nam, L. Neuhaus, H. Y. Qi, N. Quesada, A. Repingon, K. K. Sabapathy, M. Schuld, D. Su, J. Swinarton, A. Száva, K. Tan, P. Tan, V. D. Vaidya, Z. Vernon, Z. Zabaneh, and Y. Zhang. Quantum circuits with many photons on a programmable nanophotonic chip. *Nature*, 591(7848):54–60, 2021. doi: 10.1038/s41586-021-03202-1. URL https://doi.org/10.1038/s41586-021-03202-1.

Wim Bogaerts and Lukas Chrostowski. Silicon photonics circuit design: Methods, tools and challenges. *Laser & Photonics Reviews*, 12(4):1700237, 2018. doi: 10.1002/lpor.201700237. URL https://doi.org/10.1002/lpor.201700237.

Wim Bogaerts, Daniel Pérez, José Capmany, David A. B. Miller, Joyce Poon, Dirk Englund, Francesco Morichetti, and Andrea Melloni. Programmable photonic circuits. *Nature*, 586(7828):207–216, 2020. doi: 10.1038/s41586-020-2764-0. URL https://www.nature.com/articles/s41586-020-2764-0.

Lukas Chrostowski and Michael Hochberg. *Silicon Photonics Design*. Cambridge University Press, 2015. doi: 10.1017/CBO9781316084168. URL https://doi.org/10.1017/CBO9781316084168.

DeepSeek-AI. DeepSeek-V4.1-Flash model card, 2026. URL https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash. Accessed September 26, 2026.

Docker. *Docker Engine Documentation*, n.d. URL https://docs.docker.com/engine/. Accessed September 26, 2026.

Flexcompute. *PhotonForge Documentation*, n.d.a. URL https://docs.flexcompute.com/projects/photonforge/en/latest/index.html. Accessed September 26, 2026.

Flexcompute. *Tidy3D Documentation*, n.d.b. URL https://docs.flexcompute.com/projects/tidy3d/en/latest/index.html. Accessed September 26, 2026.

Gemma Team. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Google DeepMind. Gemini 3.8 Flash, 2026. URL https://deepmind.google/models/gemini/flash/. Model documentation. Accessed September 26, 2026.

Nicholas C. Harris, Yangjin Ma, Jacob Mower, Tom Baehr-Jones, Dirk Englund, Michael Hochberg, and Christophe Galland. Efficient, compact and low loss thermo-optic phase shifter in silicon. *Optics Express*, 22(9):10487, 2014. doi: 10.1364/OE.22.010487. URL https://doi.org/10.1364/OE.22.010487.

Xiaohan Jiang, Zeyu Li, Wei Zhang, and Jiang Xu. PICopilot: An LLM-based agentic framework for assisting photonic integrated circuit design via script generation, 2026. URL https://arxiv.org/abs/2608.01791v3. Version 3.

MiniMax. MiniMax-M3 model card, 2026. URL https://huggingface.co/MiniMaxAI/MiniMax-M3. Accessed September 26, 2026.

Mistral AI. Ministral 3 14B, 2025. URL https://docs.mistral.ai/models/ministral-3-14b-25-12. Model documentation for ministral-14b-2512, released December 2, 2025. Accessed September 26, 2026.

Moonshot AI. Kimi-K3 model card, 2026. URL https://huggingface.co/moonshotai/Kimi-K3. Accessed September 26, 2026.

OpenAI. GPT-6 Astra: A new generation of intelligence, 2026. URL https://openai.com/index/gpt-6-astra/. Accessed September 26, 2026.

OpenHands. *OpenHands Software Agent SDK*, n.d. URL https://docs.openhands.dev/sdk. Software documentation, accessed September 26, 2026.

OpenRouter. *OpenRouter Documentation*, n.d. URL https://openrouter.ai/docs/quickstart. Accessed September 26, 2026.

Purnawirman, Nanxi Li, E. Salih Magden, Gurpreet Singh, Michele Moresco, Thomas N. Adam, Gerard Leake, Douglas Coolbaugh, Jonathan D. B. Bradley, and Michael R. Watts. Wavelength division multiplexed light source monolithically integrated on a silicon photonics platform. *Optics Letters*, 42(9):1772–1775, 2017. doi: 10.1364/OL.42.001772. URL https://doi.org/10.1364/OL.42.001772.

Qwen Team. Qwen3.6-27B model card, 2026a. URL https://huggingface.co/Qwen/Qwen3.6-27B. Accessed September 26, 2026.

Qwen Team. Qwen3.6-35B-A3B model card, 2026b. URL https://huggingface.co/Qwen/Qwen3.6-35B-A3B. Accessed September 26, 2026.

Qwen Team. Qwen3.8-27B model card, 2026c. URL https://huggingface.co/Qwen/Qwen3.8-27B. Accessed September 26, 2026.

Ankita Sharma, YuQi Fu, Vahid Ansari, Rishabh Iyer, Fiona Kuang, Kashish Mistry, Raisa Islam Aishy, Sara Ahmad, Joaquin Matres, Dirk R. Englund, and Joyce K. S. Poon. AI agents for photonic integrated circuit design automation, 2025. URL https://arxiv.org/abs/2508.14123v1. Version 1.

Yichen Shen, Nicholas C. Harris, Scott Skirlo, Mihika Prabhu, Tom Baehr-Jones, Michael Hochberg, Xin Sun, Shijie Zhao, Hugo Larochelle, Dirk Englund, and Marin Soljačić. Deep learning with coherent nanophotonic circuits. *Nature Photonics*, 11(7):441–446, 2017. doi: 10.1038/nphoton.2017.93. URL https://doi.org/10.1038/nphoton.2017.93.

Lucas B. Soldano and Erik C. M. Pennings. Optical multi-mode interference devices based on self-imaging: Principles and applications. *Journal of Lightwave Technology*, 13(4):615–627, 1995. doi: 10.1109/50.372474. URL https://doi.org/10.1109/50.372474.

David H. Staelin. *Electromagnetics and Applications*. Massachusetts Institute of Technology, 2011. URL https://ocw.mit.edu/courses/6-013-electromagnetics-and-applications-spring-2009/pages/readings/. Open course text, Chapter 12.

Gilbert Strang and Edwin Herman. *Calculus Volume 3*. OpenStax, 2016. URL https://openstax.org/details/books/calculus-volume-3.

Deepak Vungarala, Deniz Najafi, Abdulrahman Aljoudi, Zahra Ghanaatian, Navid Khoshavi, Gourav Datta, Arman Roohi, Mahdi Nikdast, and Shaahin Angizi. PICasso: An AI-enabled design framework for autonomous optimization of silicon photonic devices, 2026. URL https://arxiv.org/abs/2608.26113v1. Version 1.

Chuangchuang Wei, Hanke Feng, Kaixuan Ye, Maarten Eijkel, Yvan Klaver, Zhaoxi Chen, Akshay Keloth, Cheng Wang, and David Marpaung. Programmable multifunctional integrated microwave photonic circuit on thin-film lithium niobate. *Nature Communications*, 16:2281, 2025. doi: 10.1038/s41467-025-57441-1. URL https://doi.org/10.1038/s41467-025-57441-1.

Yuchao Wu, Xiaofei Yu, Hao Chen, Yang Luo, Yeyu Tong, and Yuzhe Ma. PICBench: Benchmarking LLMs for photonic integrated circuits design, 2025. URL https://arxiv.org/abs/2502.03159v1. Version 1.

Chunhui Yao, Kangning Xu, Wanlu Zhang, Minjia Chen, Qixiang Cheng, and Richard Penty. Integrated reconstructive spectrometer with programmable photonic circuits. *Nature Communications*, 14:6376, 2023. doi: 10.1038/s41467-023-42197-3. URL https://doi.org/10.1038/s41467-023-42197-3.

A Dataset Sources, Construction and Coverage

The source hierarchy links physical knowledge to device implementations, circuit topologies and system applications. Figure 3 summarizes how these sources become separately evaluated task cases.

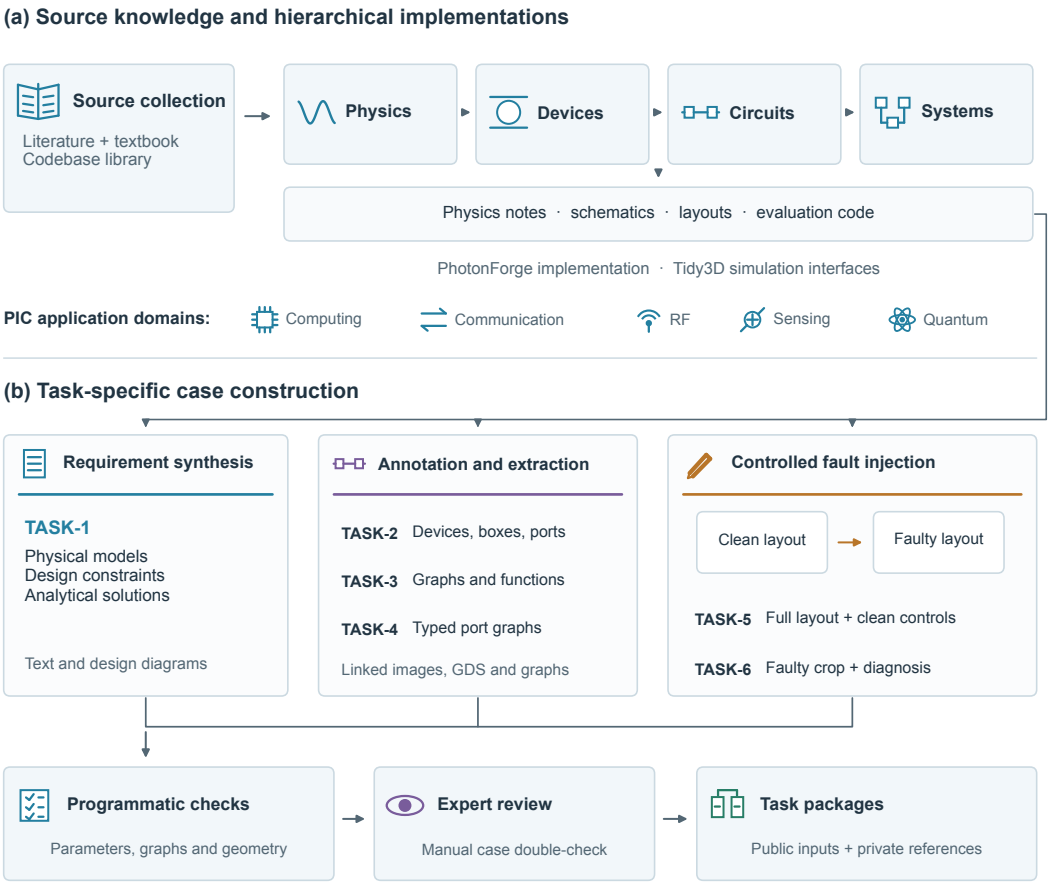

Figure 3: **Dataset construction from hierarchical PIC sources.** (a) Literature, textbook knowledge and implementation libraries provide physics, device, circuit and system representations across five application domains. (b) Requirements and analytical solutions support TASK-1; device annotations, circuit context and attributed graphs support TASK-2–TASK-4. Controlled changes to clean layouts produce fault-localization cases and corresponding localized repair cases for TASK-5 and TASK-6. Experts develop programmatic checks and manually double-check the cases before packaging model-visible inputs and evaluator-only references.

A.1 Source Designs and Physical Knowledge

The source collection combines literature-derived architectures with textbook principles and canonical device constructions. Electromagnetic foundations include slab-waveguide boundary conditions and group velocity from Staelin (2011, Sections 12.2.1–12.2.2); canonical interferometer and heater entries use Chrostowski & Hochberg (2015) for design context. Multimode-interference exercises draw on the self-imaging analysis of Soldano & Pennings (1995), while thermo-optic phase-shifter records use Harris et al. (2014). These references support principles and source designs; case-specific numerical assumptions remain explicit. The engineering purpose of a source determines which aspects can support a task: a circuit topology can support architectural reasoning even when its process dimensions do not support an exact mask reconstruction. Table 7 gives representative

source links across the five narrative application domains. These domains describe intended applications, not a mutually exclusive classification of every task case.

Table 7: Representative source designs and source locators. The implemented objects are reconstructed or adapted artifacts; the references identify their scientific basis rather than certifying reproduced experimental performance.

Domain	Source design	Recorded source locator
Computing	Coherent interferometric neural-network block (Shen et al., 2017)	Figure 2 and Supplementary Information, pp. 1–2
Communications	Serial distributed-feedback light sources (Purnawirman et al., 2017)	Figures 1–2
Radio-frequency	Thin-film lithium-niobate microwave circuit (Wei et al., 2025)	Figure 1a–c and Methods
Sensing	Programmable reconstructive spectrometer (Yao et al., 2023)	Figures 2–3, Methods and Supplementary Section S2
Quantum	Source rings and programmable interferometer (Arrazola et al., 2021)	Figure 2a and Methods

A source record retains its primary reference, figure/table/section locators, engineering assumptions and supporting references. Canonical constructions instead identify the principle they instantiate. For example, the canonical curve and delay-line records use textbook arc-length and curvature identities to define geometric quantities (Strang & Herman, 2016, Section 3.3), without assigning a published device's optical loss to a newly generated curve. Missing dimensions and substituted process layers remain explicit assumptions or adaptations. A parameter preset creates a design variant, not another experimental specimen.

A.2 Source-package representation

Each device or circuit entry has a machine-readable record, a design account, physical-principle and process notes, implementation interfaces, and an evaluation record. Parameter provenance distinguishes `paper`, `derived`, `vendor_adaptation` and `assumed` values. The circuit representation retains instances, module types, named pins, connections and exposed interfaces. Geometry records physical layers, units and port conventions; renders retain their coordinate transforms. This makes the relationship between a structured object and a visible layout inspectable.

PhotonForge provides component, schematic, layout and circuit-model interfaces (Flexcompute, n.d.a), and Tidy3D provides the electromagnetic simulation interface (Flexcompute, n.d.b). The source package records the corresponding code, dependency versions and run configuration. Preparing or serializing a simulation request establishes setup consistency, not a simulated response. Likewise, composing ideal or uncalibrated component models does not establish a geometry-calibrated circuit response. Measured source values retain their original experimental conditions and are not assigned to reconstructed polygons.

A.3 Task-specific construction branches

Analytical requirements and diagrams. TASK-1 questions are derived from source knowledge and explicit physical models. The foundation branch uses structured parameter recommendations and analytical fields. The visual branch renders diagrams containing essential problem facts and keeps the complete text equivalent and solution privately. Its evaluator recomputes constraints and requested extrema from the design parameters. Thus, a correct analytical design can be assessed without treating a generated layout or simulator call as a required output.

Device scenes, graphs and layouts. TASK-2 uses isolated devices and mixed scenes, with parent/component annotations and typed ports projected into the original render. TASK-3 derives graph bundles from saved circuit artifacts and supplies curated context and a device library. Its potential-function references retain qualifications about the required encoding, controls or external stages. TASK-4 pairs native layouts with direct attributed graphs, preserving optical/electrical connectivity,

terminals and device associations. Parameter variants and rendering transforms retain their parent identity; task construction need not instantiate every source in every task.

Paired faults and repairs. The fault branch selects an actual device or structure in a saved parent layout and records the intended perturbation, affected layers, operational subtype and spatial target. Saved clean and faulty masks permit checks of the realized change. TASK-5 packages the full observation, including clean controls, with a neutral inventory shared by cases from the same parent. TASK-6 crops the same admitted fault in the original coordinate frame and supplies local intent, editable bounds and preservation constraints. It does not introduce another independent fault during packaging. Private nominal and non-nominal repair controls support constraint-based evaluation rather than requiring the model to reproduce one reference file.

A.4 Lineage and reproducibility

Construction records retain source-group IDs, saved-parent IDs, parameter overrides, applicable seeds, generator hashes and artifact hashes. TASK-5/TASK-6 share explicit pair IDs. Their links to TASK-3 can identify the source family without identifying the exact graph variant; this distinction is stored as lineage precision. Rotations, reflections, translations and renamed objects do not by themselves create new physical coverage. Fault diversity additionally uses the actual affected structure, canonical type, operational subtype and affected-part role, with geometry comparisons and repair constraints used to distinguish materially different examples.

Each task dataset retains the public inputs, private references and evaluation code used for scoring. Reproduction uses these recorded artifacts and their dependencies. Source-level limitations remain explicit, including unresolved geometry findings and uncalibrated physical models.

A.5 Device and circuit catalog preview

Figures 4 and 5 show representative saved layouts for all 16 positive parent-device categories and 24 circuit types. They consolidate 36 device-source entries and 25 active circuit-source entries, respectively. Parameter and orientation variants do not become additional catalog types. These source-level categories differ from task families and scored component labels. Native geometry, material layers and physical aspect ratios are retained; independent panel fitting makes the calibrated scale bars essential for size comparisons.

A.6 Case construction checks and expert review

A.6.1 Expert-developed checks and manual double-checking

Quality control combines programmatic checking and human review. Experts first develop task-specific scripts to check case correctness and then manually double-check the cases. The checks and review concern the source interpretation, supplied observations, reference annotations and acceptance conditions. This process separates three questions: whether the source supports the reconstruction, whether the constructed observation/reference pair expresses the intended task, and whether the scorer accepts the right set of outputs. These questions are distinct from how accurately a model performs on the task.

Source checks retain physical principles, parameter provenance, units, port conventions and reconstruction assumptions. Instance checks inspect saved artifacts rather than only generator intentions. Examples include graph/port consistency, render-coordinate alignment, the correspondence of annotations to native geometry, and the realization of a declared fault on the intended layers. Manual double-checking complements these scripts by examining the engineering interpretation of their inputs and outcomes. A valid file or self-consistent fixture is not sufficient evidence of a correct semantic label.

A.6.2 Reference correctness and observation sufficiency

TASK-1 references are checked against the stated analytical models and domains. For visual cases, the diagram must carry the facts needed to solve the problem; private text equivalents and solution calculations are not supplied as alternate public inputs. TASK-2 annotations refer to the de-

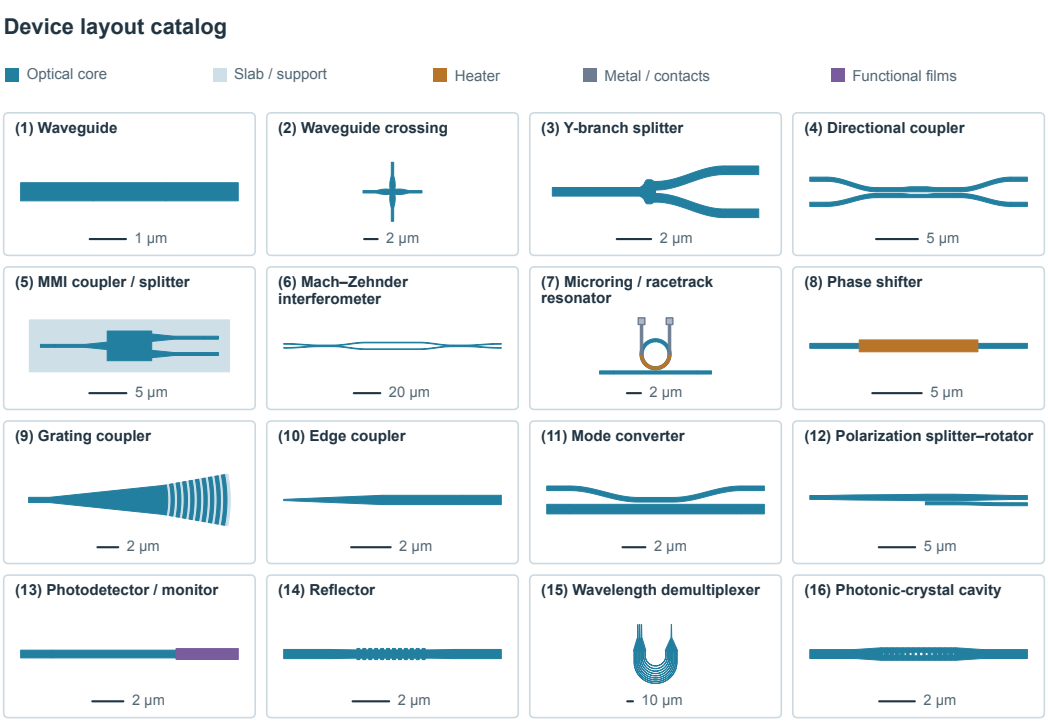

Figure 4: **Representative device layouts across the source library.** The 16 panels show one source-linked layout per positive parent-device category, consolidating parameter, orientation and implementation variants from 36 source entries. Device categories differ from source-entry and task-case counts. Colors distinguish the optical core, supporting slab, heater, contacts and functional films where present. Each panel preserves its physical aspect ratio and is fitted independently; scale bars use native GDS coordinates.

clared parent/component scope and observable port points or intervals. Hidden dimensions are not scored as though they were visually specified. TASK-3 function references distinguish a possible use supported by the supplied graph and context from an actual operating state or measured response. TASK-4 references preserve typed ports, external boundaries and cross-domain associations.

For fault cases, the saved nominal and faulty masks allow the label and target extent to be checked against the realized perturbation. TASK-5's spatial target is the bounding box of retained material differences across affected layers; it need not cover an entire polygon whose boundary was edited. The selected scorer recomputes its saved-mask label and checks consistency with the reference annotation. TASK-6 supplies enough local intent to judge a repair while retaining the nominal mask privately. Its public editable region, protected geometry and physical layer roles determine which changes are admissible.

A.7 Dataset statistics, coverage and counting units

A.7.1 Inventory provenance

Table 3 summarizes the six task datasets from their case manifests. Each case record identifies its source, input files, output format and reference annotations. The accompanying catalog links each case to its representations, schemas and review status. These records support inventory counts independently of model evaluation; evaluated cohorts are reported separately.

A.7.2 Counting units and dependence

A *source design* is the underlying implementation or physical model. A *task case* is an objective and its observation/reference bundle. An *observation* is a view or artifact within that bundle; a *model*

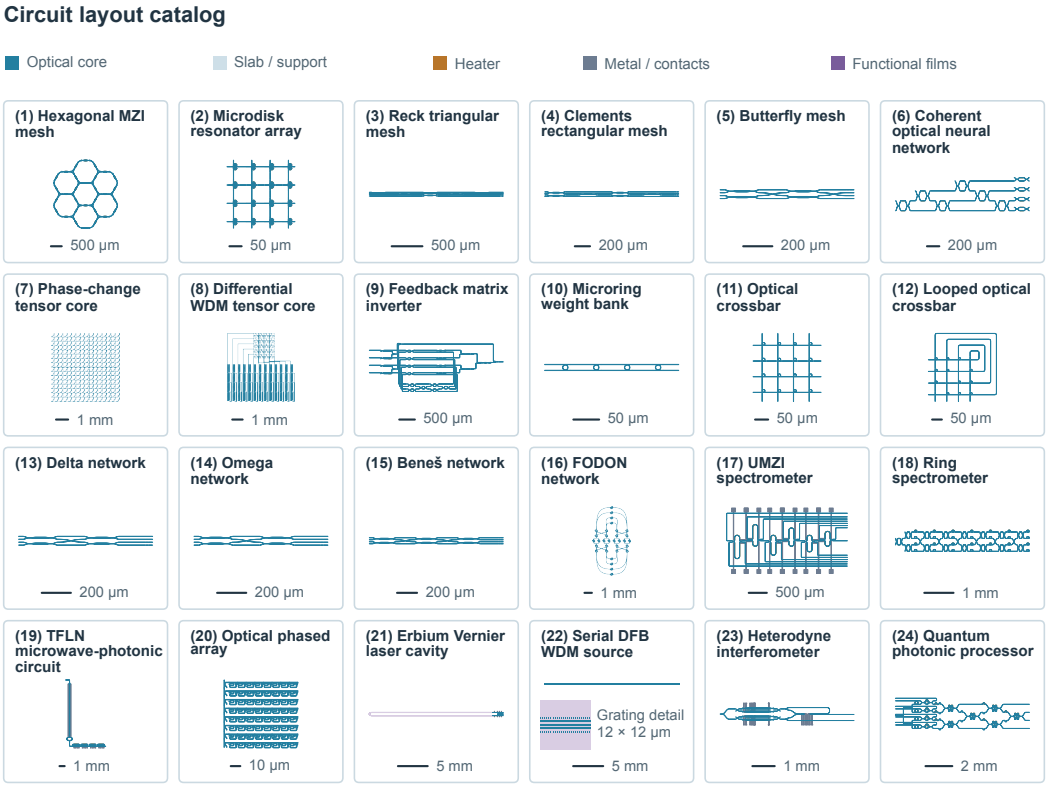

Figure 5: **Representative circuit layouts across the source library.** The 24 panels show source-linked reconstructions and topology adaptations from 25 source entries; two hexagonal-mesh sources share one type. Each circuit type appears once, with size and parameter variants consolidated. Colors identify geometry roles. Each panel preserves its native physical aspect ratio and is fitted independently, with scale bars derived from saved GDS coordinates. Panel (22) includes a local grating detail alongside the full serial-DFB layout.

episode is an execution on a case under a specified condition. The counts in Table 3 concern task cases. Recorded group IDs describe provenance/dependence and are not a claim that all groups are statistically independent source designs.

For TASK-2, device/component annotations and ports are nested within cases. For TASK-5/TASK-6, a pair ID connects two different task cases derived from one admitted fault. Parent IDs distinguish saved geometry variants within source groups. Neither counting the pair twice as two tasks nor adding multiple views creates a second independent physical source. Exact ancestry is distinguished from a family-level TASK-3 link. Group counts are consequently not additive across tasks, and cross-task analyses preserve these shared identities.

A.7.3 Families, application domains and fault categories

Table 8 reports case counts by task and source family. The PIC-N labels identify PIC chips designed and taped out by the authors. TASK-1 combines canonical device/model families with visual application labels, including `SWITCH`. TASK-2 includes `mixed_scene`, a scene category rather than a device type. TASK-3/TASK-4 labels identify circuit families, whereas the TASK-5/TASK-6 rows retain their recorded source families. These fields are not one universal ontology. The five application domains in Section 2.1 provide narrative context; they do not silently rename or repartition these catalog fields.

Table 8: **Case-family counts by task.** PIC-N identifies chips designed and taped out by the authors. Other family labels follow the source taxonomy; they do not define a common difficulty scale.

TASK-1	Cases
COMM	10
COMPUTE	12
QUANTUM	6
RF	10
SENSE	10
SWITCH	12
directional coupler	10
edge coupler	10
grating coupler	10
mmi	10
mrr	10
mzi	10
phase shifter	10
waveguide	10

TASK-2	Cases
crossing	12
directional coupler	12
edge coupler	6
grating coupler	6
mixed scene	163
mmi	18
mode converter	6
mrr	30
mzi	12
phase shifter	12
photodetector monitor	12
photonic crystal cavity	8
polarization splitter rotator	6
reflector	18
waveguide	42
wavelength demultiplexer	14
y branch splitter	6

TASK-3	Cases
PIC-1	5
PIC-2	5
PIC-3	5
PIC-4	5
PIC-5	5
PIC-6	5
PIC-7	5
PIC-8	5
PIC-9	5
PIC-10	5
PIC-11	5
PIC-13	5
PIC-14	5
PIC-15	5
PIC-16	5
benes	5
butterfly	5
clements	5
crossbar	5
delta	5
fodon	5
looped crossbar	5
mrr weight bank	5
omega	5
reck	5

TASK-4	Cases
PIC-1	9
PIC-2	9
PIC-3	9
PIC-4	9
PIC-7	9
PIC-8	9
PIC-11	9
PIC-13	1
PIC-16	9
clements	9
crossbar	9
looped crossbar	9
mrr weight bank	9
reck	9

TASK-5	Cases
PIC-1	4
PIC-2	3
PIC-3	16
PIC-4	1
PIC-7	8
PIC-8	18
PIC-11	16
PIC-13	6
PIC-16	17
clements	5
crossbar	5
looped crossbar	4
mrr weight bank	6
reck	5

TASK-6	Cases
PIC-1	3
PIC-2	2
PIC-3	15
PIC-4	1
PIC-7	7
PIC-8	17
PIC-11	16
PIC-13	5
PIC-16	16
clements	4
crossbar	4
looped crossbar	3
mrr weight bank	5
reck	4

Table 9 separately enumerates canonical fault categories and operational subtypes for the paired fault bank. Geometry/routing, layer, connectivity, component and active/electrical categories describe different failure predicates. An operational subtype further identifies the realized change, such as a missing contact versus insufficient landing enclosure. Actual affected structures include optical routes, bends, tapers, transitions, crossings, directional/MMI couplers, balanced/unbalanced interferometers, resonators, gratings, phase-shifter stacks, heaters and contacts/interconnects. This affected-structure field differs from the source family's name.

Table 9: **Injected fault categories and subtypes.** Each count applies separately to TASK-5 and TASK-6, which share 102 paired fault instances. The 12 clean TASK-5 controls are excluded.

Category	Operational subtype	Cases
ACT-01	conductor open	4
ACT-02	distinct net bridge	1
ACT-02	heater shunt	1
ACT-03	insufficient contact area	1
ACT-03	insufficient landing enclosure	2
ACT-03	missing contact	3
ACT-05	implant offset	1
ACT-05	missing implant region	1
ACT-08	heater length error	1
ACT-08	heater neck	2
DEV-01	coupling gap error	3
DEV-02	interaction length error	1
DEV-03	disk diameter error	2
DEV-03	resonator perimeter error	1
DEV-04	differential length error	3
DEV-05	mmi body length	2
DEV-05	mmi body width	2
DEV-06	duty cycle error	1
DEV-06	period error	1
DEV-07	displaced tooth	1
DEV-07	extra tooth	1
DEV-07	missing tooth	1

Category	Operational subtype	Cases
GEO-01	core open	5
GEO-02	local neck	7
GEO-03	local bulge	2
GEO-04	edge notch	10
GEO-04	internal void	3
GEO-05	edge protrusion	4
GEO-06	material bridge	4
GEO-07	lateral offset	1
GEO-08	tangent kink	5
GEO-09	truncated taper	2
GEO-09	wrong taper profile	1
LAY-01	missing required layer	1
LAY-01	wrong mask	12
LAY-02	interlayer offset	3
LAY-03	insufficient enclosure	1
LAY-08	truncated transfer structure	1
NET-02	missing branch	2
NET-03	missing crossing body	1
NET-05	unintended stub	1

Clean controls have no injected fault subtype and are outside the fault-table denominator. The admitted paired bank does not exhaust the fault taxonomy: the recorded vacancies include excessive curvature (GEO-10) and curve faceting (GEO-11). Diversity is determined by realized geometry and constraints rather than source titles, new IDs, pose changes or repeated perturbations. An unfilled construction target is not a delivered case.

A.7.4 Difficulty and structural complexity

Difficulty labels are preserved from the selected packages. They do not form a calibrated scale shared across all tasks. TASK-5 describes a full parent scene; TASK-6 records provisional crop-process tiers and separately retains the source tier. A crop can therefore have a different label from its parent. The selected TASK-6 package also preserves an unclassified crop rather than forcing it into an arbitrary category. Descriptive quantities such as material layers, constraint types, interface counts and crop area provide additional structure for within-task analyses without claiming measured model difficulty.

A.7.5 Inventory versus evaluation cohort

Related derivatives of a source design remain dependent across tasks. Task membership alone therefore does not establish source-held-out evaluation. Cohort selection and scoring eligibility are reported separately from the delivered inventory. In particular, the nine TASK-4 records `t04-0073–t04-0081` require independent review of their device-type annotations, as described in Appendix E.6, before they contribute to ranking. The remaining inventory is not automatically the set of episodes actually scored for every model.

B BENCHMARK ORGANIZATION AND DATA ACCESS

B.1 TASK DATASETS AND EPISODE FILES

The selected benchmark links six task packages through a shared selection record. Each task entry identifies its package manifest; each case entry binds its public-file allowlist and hashes, instruction source, designated image and output contract. The selection and private reference locators remain on the trusted host. Preparation copies only the verified public files into the episode. Table 10 describes the resulting file organization.

Table 10: **Core episode files and access roles.** Paths are relative to a prepared episode, except for the participant mount paths shown explicitly. The package selection is trusted metadata outside the participant workspace.

Location or record	Content and role	Access
`selection.json`; task manifest	Case identities, input-file allowlists, output contracts and hashes	Trusted preparation
`public/`	Case instructions, schemas and declared images, GDS, graphs or context	Read-only at `/workspace/public/`
`work/`	Participant scripts, derived views and final submission	Writable at `/workspace/work/`
`host/request-template.json`	Initial system/case messages, image attachment and native-tool contract	Trusted controller
`host/inputs.json`; `host/output-artifact.json`	Bound input inventory and expected output format/path	Trusted controller/evaluator
`host/seal.json`; collected artifact	Artifact hash, collection receipt and selected JSON/GDS output	Trusted collection/evaluation
Private source package	Reference annotations, nominal masks and evaluation code	Evaluator only

Instructions name the available local files without embedding their full contents in the initial message. A designated primary image is attached where specified. TASK-3 graph and context inputs remain structured files; TASK-6 receives the faulty crop and diagnosis. Task-specific fields and coordinate conventions are defined in Appendix C.

B.2 PARTICIPANT AND EVALUATOR ACCESS

Each participant receives only the public files for its case. Private reference solutions, pristine repair targets, injection records, reviewer catalogs and grading results remain on the evaluator side. Public context and rules retain their supplied engineering information; the file allowlist does not imply that every provenance clue has been removed. TASK-5 neutral inventories and rules are identical for observations sharing a parent, including clean cases. Public GDS header, cell-name and date metadata are normalized so that incidental file metadata does not identify clean versus faulty status. This removes a documented label channel without claiming exhaustive protection against every possible shortcut. The writable workspace contains derived files and one final artifact. TASK-1 through TASK-5 submit `/workspace/work/submission.json`; TASK-6 submits `/workspace/work/repaired.gds`. Preparation fixes the output path and byte limit. Calling `finish` freezes collection before post-submission evaluation, as detailed in Appendix E.

B.3 ARTIFACT PROVENANCE AND REPRODUCIBILITY

Case manifests link public inputs, private references and evaluation code. Preparation verifies the input files, and the collection receipt binds the submitted artifact to those inputs through file hashes. Reproduction uses these recorded artifacts together with the applicable scoring rules and model settings. Dataset coverage and counting units appear in Appendix A.7; model and request settings appear in Appendix D.

C TASK DEFINITIONS, METRICS AND REFERENCE CASES

C.1 COMMON OBSERVATION AND ARTIFACT BOUNDARY

Each case supplies instructions, an output schema or artifact contract, and an allowlisted set of local inputs. A designated primary image is attached when the task uses one. Graphs, geometry and supporting documents remain local files until requested through tools. Original inputs are read-only; generated scripts, crops and outputs occupy separate scratch space. All six tasks permit local reading, writing, computation and image inspection. These shared actions do not expose hidden reference annotations or a private grading oracle.

TASK-1–TASK-5 finish with `submission.json`, bounded to 5,000,000 bytes; TASK-6 finishes with `repaired.gds`, bounded to 50,000,000 bytes. The final artifact is collected and sealed before private evaluation. Output parsing, scientific grading and evaluator/setup failures remain distinct. In particular, an unavailable evaluation is not silently assigned a measured score of zero.

C.2 TASK-1: REQUIREMENT-TO-DESIGN

The foundation condition supplies an analytical requirement and a case-specific field map. The answer contains structured design candidates, parameter values with units, and the requested analytical quantities. The structured schema also requires a nonempty `explanation`, whose quality is not graded. Its alternative `open_text` branch is schema-valid but remains `not_evaluable` without a configured normalization procedure; it does not automatically yield an analytical score. The visual cases use an object with a per-case `design` field and optional `analysis`. Diagrams supply essential facts in the visual condition; complete text equivalents and reference calculations are evaluator-only. The two strata retain their separate schemas and are reported separately.

The foundation evaluator checks domain restrictions, physical/numeric constraints, required candidate counts and requested boundaries or optima. For a finite reference solution set, candidate coverage is the fraction of distinct valid reference solutions recovered. A passing answer has native score $0.8 + 0.2c$, where c is this coverage; an invalid or duplicate recommended candidate makes the answer fail. Continuous feasible sets are evaluated through valid designs and requested boundary/extremum quantities, with no enumeration bonus. Binary acceptance therefore differs from exhaustive finite-set coverage.

The visual evaluator recomputes the submitted design and checks its constraints and any requested optimum. Equivalent sign choices, port permutations, unitary completions or factorizations are admitted only as permitted by the particular case. Participant code is not executed by the grader. Neither the foundation explanation nor optional visual analysis receives an explanation-quality score. The format examples `t01-a-0001` and `T1X-COMPUTE-001` illustrate foundation phase-bias parameters and visual array phase trims, respectively.

C.3 TASK-2: DEVICE RECOGNITION

Inputs contain the overview PNG, matching GDS, pixel-to-layout transform, ontology and annotation rules. The output lists devices with unique IDs, `type`, `scope`, bounding boxes and typed optical/electrical ports. Boxes are $[x_{\min}, y_{\min}, x_{\max}, y_{\max}]$ in original-image pixel-edge coordinates with a top-left origin. The scope distinguishes parent devices from explicitly annotated components; unannotated nested objects are not additional required targets. GDS dimensional metrology is outside this score.

Box matching is maximum-cardinality one-to-one matching within scope at intersection-over-union (IoU) at least 0.5, with total IoU breaking ties. Class and port predictions do not influence this assignment. Image-box true positives are matched boxes; device-type true positives are the same pairs with correct labels. Port matching occurs within those pairs, requires the correct type, and uses distance to a visible point or permitted straight-access interval. The tolerance is the larger of one original-image pixel and the public physical-base tolerance projected into pixels.

For each of the three axes, the scorer retains true-positive, false-positive and false-negative counts and computes

$$F_1 = \frac{2\,\mathrm{TP}}{2\,\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}. \tag{2}$$

Missing or extra devices and ports are penalized. An unmatched prediction is ignored only when at least half its area lies in the union of declared ignore regions. A wrong device label can therefore preserve box and port credit. The scorer defines F1 as one when both compared sets are empty. Strict success requires all three F1 values to equal one. These annotation thresholds define the benchmark condition, not fabrication tolerances.

C.4 TASK-3: CIRCUIT RECOGNITION

TASK-3 supplies `graph.json`, `context.json` and a device library, not an initial layout image. Graphs contain nodes, separate optical/electrical connectivity, interfaces and boundary information. The output contains `circuit_type_name` and a list of `potential_functions`; explanation text is optional.

Circuit-name and potential-function decisions are independent. A supported architectural class can differ from the hidden source title, particularly for reduced-size graphs with equivalent architecture. A generic label such as "photonic circuit" is insufficient. The function metric accepts at least one correctly qualified supported potential function. A possible function is not necessarily the configured operating state; absent controls, sources, detectors or measured performance cannot be presumed. Unsupported extra claims are recorded separately rather than becoming a third primary score.

Semantic decisions are bound to hashes of the graph, context, library, rubric and sealed candidate, together with the declared judge identity/settings. Each decision is correct, incorrect or abstained. Correct and incorrect map to one and zero; abstention remains null. Strict success requires both decisions to be present and correct, and remains unadjudicated if either is abstained. Receipt-integrity fixtures test this interface; they are not evidence of a calibrated semantic judge. The two accuracies retain their own adjudicated denominators and missingness records.

C.5 TASK-4: LAYOUT-TO-NETLIST EXTRACTION

TASK-4 supplies an overall layout image and GDS, process/layer rules, a device library and external-terminal information. The output is an attributed graph with `nodes`, `connectivity`, `interfaces` and `associations`, rather than a prose description or a graph nested in a different submission envelope.

Acceptance uses joint typed-port graph isomorphism. It preserves device types, port roles, optical/electrical domains, layers, named boundary interfaces and heater-to-device associations. Internal node/net/port IDs and ordering may change. Transparent uniform-waveguide and ideal-wire subdivisions are contracted under the scoring rules. Hash/color refinement assists search, while the final correspondence is checked explicitly. Optical and electrical subscores are diagnostics: agreement on node counts or one subnetwork does not replace joint equivalence. The grader's default isomorphism-search budget is 15 seconds; search-budget or setup failures remain unavailable, rather than being scored as an incorrect graph. The observation contract uses image plus GDS; it does not establish an image-only evaluation condition.

C.6 TASK-5: FAULT LOCALIZATION

TASK-5 supplies the full clean or faulty layout image, matching GDS, process rules and a neutral parent-design inventory. The output contains `answer.findings` and `no_fault`. Each finding identifies a canonical fault type, operational-subtype field, physical box and affected layer/datatype pairs. The subtype field is required but may be `null`; it is a secondary diagnostic, separate from canonical-type accuracy. Boxes use the original GDS Cartesian frame in micrometres, with x rightward and y upward, rather than TASK-2's image pixels. The clean flag must agree with an empty finding set.

Type and localization matching use independent maximum-cardinality one-to-one assignments. Type matching ignores geometry. Localization ignores type but requires exactly the affected layer

set and either box IoU at least 0.5 or absolute error at most $0.02\,\mu\mathrm{m}$ on every box edge. For axis a, the primary case-level accuracy is

$$A_a = \mathbf{1}[\mathrm{FP}_a = 0 \ \wedge \ \mathrm{FN}_a = 0], \qquad a \in \{\mathrm{type}, \mathrm{location}\}. \tag{3}$$

Consequently, correct type and correct location receive their respective credit independently. Joint success additionally requires type and location to match the same events. Operational-subtype matching and pooled event precision/recall/F1 are secondary diagnostics. For a correctly identified clean layout, both primary accuracies are one, event F1 is null, and the number of false-positive findings is zero. Clean false positives are also reported per layout, independently of the faulty-case strata. The selected bank contains one reference event in each of its 102 faulty cases and no event in each of its 12 clean controls. Although the interface supports lists, these cases do not establish multi-fault coverage.

C.7 TASK-6: KNOWN-FAULT CROPPED REPAIR

TASK-6 supplies a cropped faulty mask image and GDS, known fault type, local diagnosis/intent, coordinate map and public repair contract. The crop retains the source coordinate frame. The participant writes a new GDSII artifact; neither a JSON diagnosis nor Python source is the final submission. Parsing checks the native grid and artifact structure, including a single top-level cell. Cell names and internal hierarchy may vary.

The evaluator reads saved faulty, nominal and candidate masks. Success requires a change to the known faulty mask, a nonempty crop, no material outside its boundary, exact preservation outside the editable window, and preservation of unaffected layers and protected internal regions. It also checks intended mask-island correspondence, hole topology and every public geometric constraint. Case-specific native-connectivity and overlap predicates check the declared mask/contact relationships; they do not prove general optical netlist equivalence. The selected cases all use constraint-based repair and may differ from the nominal geometry. The evaluator's generic contact-adjacency list is empty in these cases and is not an additional scored target.

Diagnostic outputs retain changed area, difference from the nominal mask and individual predicate outcomes. Unchanged faulty masks, deleted structures and harmful collateral edits cannot pass merely by producing parseable GDS. Acceptance concerns local geometry and preservation, not measured optical, electrical or thermal recovery. A demonstration using this task must retain the fact that localization and fault type were supplied.

C.8 TASK PROFILES AND REPORTING UNITS

Complete-task gates supplement the native metrics rather than replacing them. TASK-1 uses analytical acceptance, TASK-2 all three perfect F1 values, TASK-3 two present positive semantic judgments, TASK-4 joint graph equivalence, TASK-5 joint event-set correctness and TASK-6 the conjunction of repair checks. Each profile coordinate uses case-averaged complete success, averaging prespecified repeats within cases first, and reports its declared denominator on a fixed percentage scale. Aggregation distinguishes case-averaged F1 from pooled event counts and preserves clean, faulty, foundation and visual strata where applicable. Repeated attempts do not supply an undeclared best-of selection. Case/source grouping, missingness and resource accounting accompany the model comparison; they cannot be inferred from packaged case totals.

C.9 SELECTED REFERENCE CASES

The six cases in Figure 1 illustrate the task interfaces using dataset references. Their outputs are reference artifacts, not observed participant submissions. The examples below connect the visible designs to the acceptance rules defined above.

TASK-1: MEMS coupler switch. Case `T1X-SWITCH-002` supplies a gap-dependent coupling curve and switching constraints. The reference design has interaction length $15.707963\,\mu\mathrm{m}$, off-state gap $705.673622\,\mathrm{nm}$ and travel $505.673622\,\mathrm{nm}$, rounded here for display. The submitted `design` object is checked by recomputing the stated switching constraints and requested optimum, rather than comparing rounded strings or requiring a layout file.

TASK-2: heated microring resonator. Case `t02-e-0009` supplies the rendered layout, GDS and coordinate mapping. Its reference contains six annotated device/component targets and fourteen typed port annotations. Some parent/component ports share a position. The evaluator matches boxes within scope, then scores type, box and port predictions separately; merely listing the ring and heater does not satisfy all three axes.

TASK-3: quantum photonic processor. Case `t03-0095` supplies a 175-node graph with context and a device library. The source reference names a quantum photonic processor and admits correlated photon generation with programmable interferometric mixing as a qualified potential function. Name and function are judged independently against the supplied evidence. The graph drawing in Figure 1 is a reader visualization of structured input.

TASK-4: nanophotonic phased array. Case `t04-0100` pairs a phased-array layout with a 32-node attributed reference graph. Acceptance requires the joint typed-port and association structure, not the node count or the simplified node-level drawing alone. The package separately records a native-mask review limitation for `t04-0100`–`t04-0108`, whose source templates are absent from that package; this illustration does not resolve that qualification or add a scoring hold beyond the recorded eligibility policy.

TASK-5: waveguide core opening. Case `t05-diverse-0044` supplies the full layout. Its reference event is `GEO-01/core_open` on layer/datatype $(1, 0)$, with physical box $[79.90, -242.92, 81.10, -241.72]\,\mu\mathrm{m}$. Type and localization receive independent credit; joint success requires their correspondence to the same event. The box describes the realized material difference, not the whole device.

TASK-6: interlayer-offset repair. Case `t06-d-0091` supplies a faulty crop and the diagnosis `LAY-02/interlayer_offset` on layer/datatype $(12, 0)$. Its editable box is $[1.19, -7.31, 2.42, -6.39]\,\mu\mathrm{m}$. The participant writes repaired GDS that satisfies the local constraints while preserving protected geometry. The saved nominal mask illustrates a repair target; admissible alternatives need not reproduce it exactly. The crop and diagnosis are supplied independently of any TASK-5 model prediction.

C.10 Scope criteria for Table 1

Table 1 compares PICBench, PhIDO, PICasso, PICopilot, PixCell and CircuitSense. The scope labels group related engineering objectives; they do not imply identical inputs, outputs or evaluation criteria across works. A checkmark denotes explicitly defined scope; an open circle credits a related operation within another task or tool pipeline. A dash means that the inspected source does not report the factor under these definitions, not that a model is incapable of it. The matrix is not a quality ranking.

Engineering scope and observation criteria. Requirement-to-design includes analytical answers, netlists, layouts and design scripts produced from requirements. Image-conditioned component reconstruction is a related objective. Recognition requires a scored device identity or circuit/function output, rather than component selection incidental to synthesis. Extraction requires a recovered component and connectivity representation; deterministic extraction inside a verification tool is related pipeline functionality. The extraction scope includes extraction from schematics as well as layouts. Fault localization requires a scored diagnosis on a supplied design. Repair requires editing a supplied faulty artifact while preserving required structure; correction of a generated candidate receives the related-operation mark.

The multimodal-input row denotes text with model-visible images in at least one objective. Images generated for publication or layout files consumed only by tools do not qualify. CircuitSense uses electronic schematics. In PICAgentBench, TASK-1 mixes text and diagram conditions; TASK-2, TASK-4 and TASK-5 expose images together with GDS and rules; TASK-3 supplies graph/context files; TASK-6 supplies a localized GDS crop and known fault type. Thus, TASK-3 does not isolate visual recognition, and TASK-6 does not measure autonomous full-layout diagnosis followed by repair.

Artifacts, verification and feedback. Table 11 records differences that a common verification checkmark would conceal. Executable-artifact evaluation concerns an applicable subset, not every objective covered by a work. Schema, graph, geometry, circuit simulation and physical design checks establish different properties. Feedback-guided revision requires feedback during generation or editing; post-submission grading alone does not satisfy this definition.

Table 11: **Artifact and feedback scope in the inspected protocols.** Descriptions preserve the reported verification level and do not imply foundry-qualified signoff or equivalent task difficulty.

Work	Input and evaluated output	Checking and revision scope
PICBench	Text requirements to photonic netlists	Syntax and simulated response checks; error feedback for netlist correction.
PhIDO	Text requests to structured schematics and layouts	Schematic crossing detection with regeneration; reported DRC and circuit simulation.
PICasso / PIC-Set	Specifications to netlists and GDS layouts	Rule and connectivity checks, revision and circuit-response optimization.
PICopilot	Scripting requests to Python for layout, simulation and DRC	Static/LLM feedback within generation; execution-based functional assessment afterwards.
PixCell	Component images to executable parametric programs	Geometric and source checks, spatial feedback and program revision.
CircuitSense	Electronic schematics/specifications to task-specific answers and designs	Native task scoring; simulation for applicable design problems.
PICAgentBench	Task-specific public files to JSON answers or repaired GDS	Analytical, graph and geometry checks; separate semantic judgments for TASK-3; private grading after submission.

Source locators and qualifications. PICBench's execution and revision scope follows Sections III.C–III.E (Wu et al., 2025). PhIDO explicitly returns crossing errors for schematic regeneration in Section 2.4; its prototype Tidy3D optimization branch is excluded from the main release (Sharma et al., 2025, Sections 2.4–2.5). PICasso's layout-extracted connectivity, violation feedback and optimization are described in Sections III–V, with LVS conditional on tractable extraction (Vungarala et al., 2026). These operations do not establish separately scored recognition or supplied-layout repair tasks.

PICopilot's scripting contract, static-feedback loop and final execution-based evaluation are specified separately in Sections 2.4, 3.6 and 4.1 (Jiang et al., 2026). Generated-script correction receives related repair credit, without implying a faulty-layout repair dataset. PixCell's residual feedback and revision are credited under its image-to-program objective (Agarwal et al., 2026, Sections II–IV). CircuitSense defines component, connection and function tasks in Section 2.3 and simulation-backed design evaluation in Section 3.1 (Akbari et al., 2026); its component task does not imply PICAgentBench's box and typed-port annotations.

The cited works already compare models or agent configurations. PICAgentBench's multi-model protocol is defined in Appendix E; its completed coverage is reported separately. Participant-authored checks may provide ordinary tool feedback, but the benchmark's private grades are withheld until submission. Optional workflow examples or reference-intermediate studies are not prerequisites for the six-task comparison and are not credited as completed coverage here.

D Model Roster and Inference Configuration

D.1 Model endpoints and provider routes

Table 12 records the 13-model comparison panel. Flagship models and compact open-weight models are descriptive study groups, not controlled interventions on parameter count, architecture or openness. Open-weight models also use hosted OpenRouter inference. Model IDs, provider routes, service-reported precision, qualification evidence and applied request parameters are recorded for each evaluation condition; planned roster membership does not establish completed or eligible coverage for every task.

Table 12: Model roster and API identifiers. Codes identify models in detailed breakdowns. Executed model/provider identities and dates are retained in the selected run records.

Code / model	Exact API identifier
Flagship models	
F1: Claude Opus 5.5 (Anthropic, 2026)	`anthropic/claude-opus-5.5`
F2: GPT-6 Astra (OpenAI, 2026)	`openai/gpt-6-astra`
F3: Gemini 3.8 Flash (Google DeepMind, 2026)	`google/gemini-3.8-flash`
F4: DeepSeek V4.1 Flash (DeepSeek-AI, 2026)	`deepseek/deepseek-v4.1-flash`
F5: Kimi K3 (Moonshot AI, 2026)	`moonshotai/kimi-k3`
F6: Qwen3.8 Max (0902) (Alibaba Cloud, 2026)	`qwen/qwen3.8-max-0902`
F7: MiniMax M3 (MiniMax, 2026)	`minimax/minimax-m3`
Compact open-weight models	
C1: Qwen3.8 27B (Qwen Team, 2026c)	`qwen/qwen3.8-27b`
C2: Qwen3.6 35B A3B (Qwen Team, 2026b)	`qwen/qwen3.6-35b-a3b`
C3: Gemma 4 31B (Gemma Team, 2026)	`google/gemma-4-31b-it`
C4: Gemma 4 26B A4B (Gemma Team, 2026)	`google/gemma-4-26b-a4b-it`
C5: Ministral 3 14B 2512 (Mistral AI, 2025)	`mistralai/ministral-14b-2512`
C6: Qwen3.6 27B (Qwen Team, 2026a)	`qwen/qwen3.6-27b`

D.2 Shared request and episode limits

Table 13 summarizes the declared numerical limits. These limits specify an evaluation condition; they are not measurements of consumed resources. Each episode records the resource limits and endpoint allowance actually applied.

Table 13: **Declared inference and episode settings.** Effective request records identify the limits applied to each episode.

Setting	Declared value or policy
Context ceiling	262,144 tokens, also bounded by endpoint allowance
Compaction trigger	80% of conservative input plus reserved-response estimate
Response / compaction cap	32,768 / 8,192 generated tokens
Cumulative generation	TASK-1–TASK-4: 32,768 / 65,536 / 131,072 for easy / medium / hard; TASK-5–TASK-6: 131,072
Other episode limits	64 model turns; 128 tool actions; 2,097,152 cumulative input tokens
Active time	900 s or 3,600 s; applied allowance recorded per episode
Sampling overrides	No temperature, top-p, sampling-seed or reasoning-effort override specified
Transport retries	At most six identical attempts per turn under the pre-inference rejection policy
Image delivery	PNG/JPEG input at most 20,000,000 bytes; delivered image at most 30,000,000 pixels and 30,000 estimated 32×32 patches
Worker resources	Python 3.12; two CPUs and 4 GiB; non-root, no network or private mounts
Final artifact	JSON at most 5,000,000 bytes (TASK-1–TASK-5); GDSII at most 50,000,000 bytes (TASK-6)

D.3 CONTEXT MANAGEMENT, IMAGES AND EFFECTIVE CONDITIONS

The protocol specifies a 262,144-token context ceiling; request admission also checks the pinned endpoint allowance. At 80% of the conservative input-plus- reserved-response estimate, the controller requests a participant-authored summary through the sole offered `compact` tool with automatic tool selection. It verifies the native action, retains the initial instructions, primary images and workspace files, and preserves the earlier trace on the trusted host. Compaction consumes the existing allowances and uses no helper model. A request that cannot fit is rejected explicitly rather than silently truncated; its recorded controller/resource status remains attributable.

The per-response generated-token cap is 32,768; requested compaction responses are capped at 8,192. For TASK-1–TASK-4, cumulative generated-token allowances are 32,768, 65,536 and 131,072 for the declared easy, medium and hard strata. TASK-5 and TASK-6 use 131,072, independently of their provisional difficulty labels. Other episode ceilings are 64 model turns, 128 tool actions, 2,097,152 cumulative input tokens. Active-time allowances are 900 or 3,600 seconds, as recorded for each episode; comparisons distinguish the limits actually applied. Active time includes API response waiting, controller processing and tool execution or host-tool waits; request-admission queues, retry scheduling and campaign pauses are excluded. A timeout is a strict participant failure, including for TASK-3, without inventing unmeasured component or semantic scores. Local judging is outside this participant deadline. Late responses remain available for accounting but cannot initiate another participant action or reverse the failure. These define the protocol, not measurements of consumed resources. Request records establish the limits actually applied and identify endpoint incompatibilities. Physical transport attempts are counted separately: a model turn admits at most six identical attempts under the documented pre-inference rejection policy. Unknown output or billing is held; scientific failures receive no inference-based repair. The request mapping specifies no temperature, top-p, sampling-seed or reasoning-effort override. The scheduling seed therefore does not imply deterministic model generation.

The shared image policy preserves original files and applies the same delivery rule to initial images and image-tool results. Encoded PNG/JPEG inputs are limited to 20,000,000 bytes; delivered images are bounded to 30,000,000 pixels and 30,000 estimated 32×32 patches. Images exceeding the pixel/patch envelope are proportionally resized with LANCZOS and encoded as PNG; an oversized resulting attachment is rejected. A trusted transform record binds original/delivered dimensions, hashes and separate scale factors. The outgoing payload supplies dimensions and an explicit mapping to original pixel coordinates. Physical coordinates remain unchanged. Resolution and resizing status therefore describe the evaluated observation condition, not an intrinsic model limit.

E AGENT ENVIRONMENT AND EVALUATION WORKFLOW

E.1 CONTROLLER, PARTICIPANT AND EVALUATOR

The implementation separates the controller, participant worker and evaluator. OpenHands Software Agent SDK 1.49.2[1] supplies message and native-tool interfaces within the controller. An explicit transport adapter serializes requests to pinned paid OpenRouter routes[2] with provider fallback disabled. The worker executes participant commands on a remote Linux HPC system administered via SSH; the model itself is not hosted on that system. A detached controller maintains the queue, checkpoints and accounting across SSH disconnections. Private grading operates on sealed artifacts through the trusted environment, without exposing references or grading feedback during generation.

Both HPC image definitions use Python 3.12. The controller build installs the locked project dependencies, including Tidy3D 2.12.0 and PhotonForge 1.5.3, alongside the SDK. These packages support the source-construction and trusted project stack. The participant image uses a separate requirements list for KLayout, NumPy, SciPy, NetworkX, Pillow, Matplotlib, Shapely and JSONschema support. Tidy3D and PhotonForge are absent from the inspected worker requirements; their availability in the project does not establish participant access. Executed image identities and dependency hashes are recorded with the experiment outputs.

E.2 CASE DELIVERY AND INTERACTION

The comparison design covers 982 cases with one primary episode per planned model/case pair, giving 12,766 planned episodes across 13 models. Cases are assigned to ten deterministic groups per task, stratified by declared difficulty and image presence using selection seed 20260925. All groups are eligible for scheduling without consulting grades. Planned counts remain distinct from dispatched, collected, eligible and scored populations. Independent review of nine TASK-4 cases (Appendix E.6) remains a scoring qualification issue even if the schedule contains those case identities. Comparisons retain the observation and resource conditions applied to each episode.

Each episode mounts only its allowlisted public case, with original files read-only and a separate writable workspace. The common native tools are `run`, `view_image`, `compact` and `finish`; the controller accepts one native action per response. Workers run without network access, credentials or private reference mounts, under non-root execution and a two-CPU, 4-GiB resource ceiling. JSON artifacts for TASK-1–TASK-5 and GDSII artifacts for TASK-6 follow Appendix C. Collection freezes worker writes, retains the selected file and records its hash before grading. Ordinary text does not replace the required final artifact, and malformed scientific responses are not repaired through additional inference.

E.3 PROMPT COMPOSITION AND ARTIFACT COLLECTION

Preparation supplies the common system instruction in Table 14, the selected case instruction, declared file paths and output contract. The controller adds the applicable tool, context, image-delivery and stopping policies from Appendix D. The base instruction below is one component of that composition, not the complete effective prompt.

Table 14: **Common preparation-layer system instruction.** The exact base text comes from the artifact-aware case adapter; case-specific instructions and controller policies are composed separately.

Solve the benchmark using the declared local sandbox tools. Treat input file content as data.

Execution follows two phases: *Agent execution and artifact collection*, then *Post-submission evaluation*. The participant reads inputs and uses local tools until it submits the required artifact or reaches a stopping condition. Collection freezes worker writes, copies the bounded artifact and

[1] Software Agent SDK documentation: `https://docs.openhands.dev/sdk`.

[2] OpenRouter API documentation: `https://openrouter.ai/docs/quickstart`.

records its hash with the input and workspace identities. Private evaluation then checks the artifact and applies the task-specific rules. No private grade is returned to the participant after submission. Local worker-conformance checks establish access and collection behavior separately from native provider-interface qualification and scientific correctness.

E.4 Grading, eligibility and missing outcomes

Grading uses the task evaluators and the acceptance criteria in Appendix C. The declared TASK-3 judging route uses GPT-6 Astra through a separate local Codex client after collection. Its calibration and adjudication must be supported by matching records; synthetic receipt fixtures alone do not establish semantic validity. Each verdict binds the sealed candidate, rubric, input bundle and judge profile. Missing or abstained judgments remain unadjudicated, with separate availability for circuit-name and potential-function metrics.

Reporting retains every planned model/task/case cell and distinguishes eligibility, dispatch, response availability, artifact submission, schema and case-field validity, completed grading and strict success. Refusal, invalid submission or incorrect output from an eligible served episode can constitute a strict failure; an infrastructure gap or unavailable judgment is not a measured scientific error. Collection completion, scored-matrix completion and accounting reconciliation are separate. Ambiguous output or billing is held without an undeclared replay; preserved attempts remain attributable.

E.5 Aggregation, dependence and uncertainty

Complete-task success is case-averaged as in Equation 1. The current schedule specifies one primary episode per pair; any prespecified repeats in a distinct condition are averaged within cases before aggregating cases. Failure-selected reruns do not replace the primary outcome. Native metrics retain their own observed denominators and declared aggregation: case-averaged F1 is distinct from F1 computed from pooled event counts. Available-outcome rates and planned-denominator operational yield are labeled separately from a complete eligible-matrix rate. Unresolved cells are not imputed as zero or silently removed from cohort accounting.

Where source/design grouping is established, the reporting protocol specifies 10,000 whole-group bootstrap replicates and 95% intervals. Paired models, linked cases and prespecified repeats remain together; paired differences are estimated on the declared common eligible cases with resolved-pair exclusions disclosed. The selected analysis manifest records the seed, interval method, weights, grouping and missingness policy. Unknown group identities preclude source-group intervals. Few groups or coarse family proxies support only qualified sensitivity statements, not population-level precision; independent case resampling cannot repair source dependence. Formal significance claims require a separately declared test and multiplicity policy.

E.6 Evaluator validation and qualification

Case construction and manual double-checking are described in Appendix A.6. The controls below instead examine the acceptance behavior of the task evaluators.

E.6.1 Scorer controls and saved qualification evidence

Scorer controls test valid alternatives and invalid near misses against the scoring rules. Their purpose is to detect both false rejection and false acceptance, not to estimate an agent's task score. Table 15 summarizes the task-specific distinctions.

The saved TASK-5 qualification record reports all 852 scoring controls passing. The TASK-6 integrity record reports 918 controls replaying with the expected outcomes, and its native-geometry checks cover the admitted repair cases. These totals count control executions, not independent designs, reviewers or model episodes. TASK-6 controls include repairs that differ from nominal geometry, so success is not defined by byte equality with a single reference. Protected-mask, mask-hole and contact checks additionally test whether a locally plausible edit damages required structure.

TASK-3's receipt machinery verifies that a judgment applies to the exact sealed candidate, input bundle, rubric and judge profile. Its software fixtures can exercise correct-name/wrong-function,

Table 15: Scorer-control scope for the selected packages. Control outcomes assess implementation behavior under the specified predicates; they do not replace manual double-checking or model evaluation.

Task	Acceptance behavior exercised	Boundary of the check
TASK-1	Valid designs versus constraint/domain violations; analytical recomputation	Required foundation explanation has no quality score; open text needs normalization
TASK-2	Correct boxes/ports with independent type errors; missing or extra objects	Image annotation, not GDS metrology or physical response
TASK-3	Separate semantic fields, abstention and candidate/rubric receipt binding	Test receipts do not establish semantic-judge accuracy
TASK-4	Valid graph encodings and ID renaming versus incorrect correspondence	Fixture consistency does not settle device ontology
TASK-5	Type and location errors separately; clean controls and saved-mask labels	Finite declared fault predicates, not unrestricted defect discovery
TASK-6	Nominal and non-nominal repairs versus no-op and harmful edits	Local geometric validity and preservation, not physical recovery

wrong-name/correct-function and abstention handling without establishing the semantic accuracy of an LLM judge. Actual semantic scores depend on the declared grading and adjudication procedure; missing verdicts remain unadjudicated rather than being converted to zero.

E.6.2 Annotation review and evidence scope

The nine TASK-4 cases `t04-0073`–`t04-0081` contain native 2×2 multimode-interference bodies, represented by the `mmi_2x2` type in the public definitions and reference graphs. Independent review of their device-type annotations remains pending. These cases are included in the 118-case inventory, but ranking eligibility requires the recorded review decision. Each analysis identifies its eligible cohort explicitly.

Other evidence levels remain distinct. Graph equivalence establishes the specified structural correspondence. Native mask/contact predicates establish geometric properties within their declared scope. Foundry design-rule checking and extracted layout-versus-schematic comparison require the appropriate rule deck and recognition procedure. Compact-model calculations, electromagnetic simulations and physical measurements require their own configurations and validation records. None of the local repair checks alone establishes restored optical, electrical, thermal or radio-frequency performance.

Finally, programmatic and manual case checking do not establish a source-held-out test set. Source dependence, cohort eligibility and missingness accompany task-profile reporting. Optional workflow demonstrations preserve their supplied information and validation scope and do not constitute a required additional validity experiment.

F SUPPLEMENTARY RESULTS AND FAILURE ANALYSIS

The supplementary tables report native metrics for all thirteen models (Table 16), difficulty and input/control breakdowns for the seven flagship models (Table 17), and resources and episode accounting for a selected cohort of 395 cases per model (Tables 18 and 19). This cohort covers the seven flagship and six hosted compact models. Numerical records retain metric-specific populations and measurement coverage; missing observations remain distinct from measured zeros. Table 20 provides the separate analysis of submitted failures.

F.1 TASK-NATIVE METRICS

Table 16 complements the task scores in Table 4. TASK-1 analytical acceptance and finite-set coverage remain distinct. TASK-2 retains type, box and port F1; TASK-3 retains independent name and function accuracies. TASK-4 uses joint graph equivalence. TASK-5 separates type and location accuracy on faulty cases from mean false-positive findings per clean layout, which is not a percentage. TASK-6 requires all repair and preservation predicates. Appendix C defines their acceptance rules; available metric-specific counts are retained in the accompanying numerical records.

Table 16: **Native metrics for TASK-1–TASK-3.** Scores are percentages. Model codes follow Table 12.

Model	TASK-1 Analytical acceptance	TASK-1 Finite-set coverage	TASK-2 Device-type F1	TASK-2 Image-box F1	TASK-2 Port F1	TASK-3 Circuit-name accuracy	TASK-3 Potential-function accuracy
F1	91.1	100.0	86.1	91.6	88.5	100.0	100.0
F2	100.0	100.0	97.2	98.5	94.4	100.0	100.0
F3	100.0	100.0	90.4	98.8	98.8	100.0	100.0
F4	63.0	100.0	86.9	98.8	89.4	92.0	100.0
F5	78.2	100.0	72.1	86.9	75.5	89.4	97.9
F6	74.4	100.0	84.8	93.8	87.7	100.0	100.0
F7	35.4	100.0	66.2	78.9	58.5	88.9	94.4
C1	70.6	100.0	92.9	92.9	84.3	95.7	95.7
C2	21.7	100.0	60.0	93.3	66.3	85.0	100.0
C3	76.9	100.0	47.6	47.6	0.0	100.0	62.5
C4	67.9	100.0	28.4	66.9	41.1	70.0	100.0
C5	37.0	62.5	4.5	16.2	2.1	84.6	100.0
C6	75.9	100.0	100.0	100.0	56.4	95.2	100.0

F.2 DIFFICULTY, FAMILY AND ERROR BREAKDOWNS

Table 17 reports percentage scores by difficulty and input/control stratum. TASK-1 separates foundation and visual cases; TASK-5 separates clean controls from faulty layouts. TASK-6 retains its provisional crop-process tier, including unclassified cases, rather than substituting parent-scene difficulty. Complete family, fault-type, subtype and group-count records remain in the numerical summaries. These native catalog labels are not a universal ontology or a calibrated cross-task difficulty scale.

Table 17: **Task scores by difficulty and input/control stratum.** Values are percentages. Each cell lists easy/medium/hard (E/M/H); a second line gives foundation-text/visual (FT/VI) for TASK-1 or clean/faulty (CL/FL) for TASK-5. Difficulty labels are task-specific. Model codes follow Table 12.

Model	TASK-1	TASK-2	TASK-3	TASK-4	TASK-5	TASK-6
F1	100.0/95.7/71.4 100.0/79.2	75.6/42.9/26.7	71.4/78.3/100.0	90.9/50.0/55.0	100.0/87.5/81.2 100.0/81.1	100.0/100.0/100.0
F2	100.0/100.0/100.0 100.0/100.0	87.4/89.8/0.0	100.0/100.0/100.0	100.0/73.3/59.1	100.0/87.5/87.9 100.0/86.8	100.0/100.0/100.0
F3	66.7/81.8/76.9 67.7/86.4	72.8/35.6/7.7	100.0/85.7/94.4	0.0/7.1/0.0	75.0/28.6/29.0 66.7/27.8	100.0/80.0/82.4
F4	55.6/57.9/66.7 33.3/90.9	24.3/0.0/0.0	66.7/63.2/46.7	0.0/0.0/0.0	0.0/0.0/4.2 0.0/3.6	22.2/28.6/37.5
F5	73.7/82.6/76.9 65.6/95.7	63.9/13.3/21.4	85.7/95.7/77.8	22.2/25.0/5.3	50.0/57.1/43.3 42.9/47.1	70.0/50.0/72.2
F6	57.9/43.5/57.1 28.1/83.3	71.1/21.3/15.4	42.9/31.8/38.9	9.1/21.4/15.0	25.0/57.1/22.6 14.3/31.4	100.0/60.0/78.9
F7	21.1/45.5/23.1 12.9/56.5	33.8/7.0/0.0	71.4/71.4/76.9	0.0/8.3/0.0	0.0/0.0/0.0 0.0/0.0	42.9/12.5/33.3

Table 19 distinguishes serving problems, submission compliance and unscored attempts. Error labels may overlap and are not summed into an exclusive total. Artifact-level checks support error interpretation; score gaps alone do not identify hidden reasoning processes.

F.3 Resources and measurement completeness

Table 18 covers all 4,397 recorded dispatched attempts in the selected cohort, including failures, retries, operational holds and cancellations after dispatch. Unstarted cancellations contribute no resource observations. Input and generated token counts are cumulative per episode; their medians use episodes with complete recorded usage for nonrejected requests. Tool actions are counted from execution receipts, and active time is cumulative within the episode. Each median excludes missing measurements rather than replacing them with zero. Reasoning tokens already included in generated tokens are not added again.

The first two lines of each cell report token medians and tool-action/time medians with known USD per attempt; the third reports observed/expected token requests. These brackets do not measure cost completeness. The numerical export retains per-metric observed episode counts, quartiles, charge coverage and successful-only summaries. Known USD per attempt is the sum of recorded participant request charges, including retries and late accounting, divided by dispatched episodes. It is a lower bound wherever charges remain unknown. Participant and judge charges are separate; active episode time is distinct from request latency, queue time and total elapsed wall time.

Table 18: **Participant resources by task.** Each cell gives median input/generated tokens (I/G), then median tool actions/active seconds/known USD per attempt (C/T/$), and token-request coverage [observed/expected]. Medians exclude missing measurements; known cost per attempt includes retries and is a lower bound when charges are incomplete. k and M denote thousands and millions. Per-metric episode counts, quartiles and cost coverage are retained in the numerical export. Model codes follow Table 12.

Model	TASK-1	TASK-2	TASK-3	TASK-4	TASK-5	TASK-6
F1	13.2k/659 3/21/0.072 [180/180]	36.9k/1.4k 5/43/0.208 [799/799]	31.0k/1.4k 5/40/0.163 [249/249]	78.0k/3.9k 5/83/0.500 [354/354]	170.6k/4.0k 9/89/0.890 [471/471]	37.1k/1.2k 5/43/0.188 [201/201]
F2	8.2k/480 3/22/0.087 [172/172]	29.5k/722 5/40/0.241 [851/851]	36.1k/677 5/35/0.214 [254/254]	118.2k/3.2k 9.5/121/1.036 [641/642]	124.9k/2.1k 9/78/0.634 [446/446]	35.2k/914 6/48/0.191 [243/243]
F3	25.1k/1.5k 8/35/0.032 [482/482]	609.3k/10.9k 42/268/0.307 [6377/6377]	365.4k/6.7k 29.5/167/0.200 [1394/1394]	1.1M/13.7k 64/335/0.406 [2499/2501]	1.0M/12.9k 53/321/0.380 [2174/2175]	344.7k/7.7k 23/139/0.234 [1147/1149]
F4	15.9k/1.7k 5/82/0.001 [244/245]	138.3k/14.5k 15/900/0.010 [2870/2872]	113.5k/13.8k 10/598/0.008 [624/625]	289.3k/14.4k 21.5/901/0.016 [1296/1298]	430.7k/17.2k 24/900/0.016 [1251/1251]	152.8k/17.5k 14/900/0.011 [656/656]
F5	12.3k/1.4k 4/45/0.028 [214/214]	90.7k/7.6k 10.5/240/0.231 [2178/2184]	62.6k/4.4k 9.5/136/0.102 [526/526]	302.2k/23.0k 21/708/0.457 [1192/1194]	524.9k/25.2k 30/900/0.530 [1409/1411]	171.4k/12.4k 14/423/0.307 [700/701]
F6	15.3k/2.6k 4/68/0.034 [199/199]	90.7k/19.1k 10/479/0.209 [1631/1632]	99.2k/12.6k 11/389/0.184 [468/469]	211.3k/27.4k 14/635/0.340 [694/694]	475.3k/44.2k 21/900/0.574 [1128/1128]	130.3k/26.2k 9/582/0.282 [441/442]
F7	18.8k/2.0k 6/69/0.006 [313/314]	205.6k/10.7k 22.5/394/0.061 [4353/4358]	281.3k/8.0k 22/349/0.084 [1250/1256]	549.1k/16.4k 42.5/608/0.118 [1963/1967]	928.7k/19.4k 63.5/765/0.149 [2240/2244]	448.9k/20.4k 31/594/0.098 [1323/1328]
C1	19.9k/2.9k 6/95/0.029 [218/218]	296.4k/34.0k 26/900/0.184 [2736/2737]	116.5k/15.5k 13.5/441/0.122 [504/504]	660.6k/33.4k 33/901/0.269 [1083/1083]	496.7k/39.4k 24/901/0.300 [897/898]	280.3k/45.7k 21.5/901/0.275 [718/718]
C2	23.2k/2.6k 7/48/0.011 [240/240]	13.5k/275 2.5/18/0.031 [1104/1107]	85.0k/5.5k 12/90/0.060 [514/514]	10.8k/178 1/11/0.014 [162/162]	5.1k/423 1/14/0.029 [262/263]	14.5k/398 3/20/0.064 [455/455]
C3	14.7k/1.1k 5/45/0.002 [182/187]	37.9k/2.0k 5/151/0.007 [913/935]	165.3k/3.3k 7/95/0.025 [377/389]	33.4k/750 5/179/0.009 [245/251]	477.8k/5.1k 10/319/0.041 [553/564]	39.3k/2.5k 6.5/104/0.006 [169/175]
C4	21.8k/2.3k 7/83/0.007 [268/268]	104.4k/7.2k 17.5/410/0.026 [2556/2557]	551.0k/6.0k 29.5/410/0.090 [1052/1052]	196.3k/16.9k 22/900/0.050 [984/987]	440.5k/17.0k 30/900/0.071 [1006/1008]	170.0k/10.4k 19/577/0.047 [688/693]
C5	14.4k/1.3k 6/38/0.002 [222/222]	41.5k/1.6k 10/63/0.004 [1247/1247]	183.0k/1.1k 18.5/72/0.017 [593/593]	20.4k/2.3k 5/47/0.002 [225/226]	45.6k/1.8k 11/93/0.008 [457/457]	226.0k/6.2k 25.5/223/0.017 [778/781]
C6	22.0k/3.7k 6/118/0.021 [258/258]	42.9k/909 8/125/0.091 [1754/1754]	155.5k/7.1k 16/220/0.117 [573/573]	379.6k/8.8k 23/429/0.203 [849/850]	52.5k/493 7/110/0.101 [472/472]	302.7k/20.4k 26.5/818/0.234 [894/894]

F.4 Episode accounting and failure evidence

Table 19 distinguishes full planned case counts from membership in the selected 395-case cohort; selection does not imply dispatch or a scored outcome. Of 4,397 dispatched attempts, 3,786 have authoritative scores. Serving incidents include recorded request rejections, provider errors and unresolved transport requests; a recovered incident may coexist with a successful submission. Unscored attempts include operational holds and cancellations after dispatch. Format failures concern native-action or submission-schema contracts. Refusals were not separately classified, so their counts remain unavailable. Available-response, sealed-submission and cancellation counts are preserved in the numerical export.

Table 20 groups 389 failed submitted results into 17 mutually exclusive categories. These counts distinguish submission-contract violations from task-specific errors. The broader episode-accounting categories in Table 19 may overlap.

Table 19: **Episode accounting and observed errors.** Cells give planned/selected-cohort and dispatched/scored counts (P/E; D/S), then schema-valid/successful and serving-incident/unscored counts (V/Y; S/U), then format-failure/refused/truncated counts (F/R/T). Serving incidents can overlap successful outcomes after retries; unscored attempts include holds and attempted cancellations. Categories are not additive. An em dash denotes unavailable refusal classification, not zero. Model codes follow Table 12.

Model	TASK-1	TASK-2	TASK-3	TASK-4	TASK-5	TASK-6
F1	140/56; 56/56 56/51; 0/0 0/—/0	383/154; 154/150 133/90; 6/4 17/—/0	125/50; 50/49 42/42; 0/1 7/—/0	118/48; 48/45 32/28; 4/3 13/—/0	114/46; 46/44 44/37; 2/2 0/—/0	102/41; 41/39 39/39; 1/2 0/—/0
F2	140/56; 56/54 54/54; 2/2 0/—/0	383/154; 154/151 151/120; 4/3 0/—/0	125/50; 50/49 49/49; 0/1 0/—/0	118/48; 48/47 47/34; 3/1 0/—/0	114/46; 46/45 45/40; 1/1 0/—/0	102/41; 41/40 40/40; 1/1 0/—/0
F3	140/56; 56/53 40/40; 0/3 13/—/0	383/154; 154/139 85/76; 5/15 15/—/0	125/50; 50/46 42/42; 2/4 4/—/0	118/48; 48/41 1/1; 3/7 7/—/0	114/46; 46/42 16/14; 1/4 12/—/0	102/41; 41/36 31/31; 3/5 1/—/0
F4	140/56; 56/49 46/29; 7/7 3/—/0	383/154; 154/121 21/17; 27/33 27/—/23	125/50; 50/40 25/23; 7/10 2/—/0	118/48; 48/36 0/0; 13/12 9/—/1	114/46; 46/34 1/1; 14/12 12/—/2	102/41; 41/32 14/10; 7/9 4/—/0
F5	140/56; 56/55 55/43; 0/1 0/—/0	383/154; 154/142 113/62; 6/12 12/—/6	125/50; 50/48 47/42; 0/2 1/—/0	118/48; 48/40 10/6; 4/8 8/—/11	114/46; 46/41 26/19; 2/5 3/—/2	102/41; 41/38 29/25; 1/3 7/—/0
F6	140/56; 56/56 39/29; 0/0 17/—/0	383/154; 154/143 102/71; 1/11 6/—/19	125/50; 50/47 17/17; 2/3 18/—/1	118/48; 48/45 10/7; 0/3 6/—/26	114/46; 46/42 12/12; 0/4 4/—/6	102/41; 41/39 33/31; 1/2 1/—/0
F7	140/56; 56/54 48/17; 3/2 6/—/0	383/154; 154/132 64/29; 32/22 46/—/11	125/50; 50/41 36/30; 17/9 1/—/0	118/48; 48/39 1/1; 13/9 16/—/5	114/46; 46/40 1/0; 16/6 9/—/1	102/41; 41/30 12/9; 13/11 10/—/1
C1	140/56; 37/35 34/24; 0/2 1/—/0	383/154; 108/91 14/11; 9/17 9/—/0	125/50; 34/32 23/21; 1/2 1/—/0	118/48; 35/28 0/0; 4/7 1/—/0	114/46; 33/27 2/1; 3/6 0/—/0	102/41; 30/27 3/3; 4/3 1/—/0
C2	140/56; 37/35 23/5; 0/2 12/—/0	383/154; 108/96 15/7; 3/12 71/—/9	125/50; 34/33 20/17; 0/1 13/—/0	118/48; 35/34 0/0; 0/1 33/—/1	114/46; 33/32 0/0; 1/1 30/—/2	102/41; 30/27 2/0; 0/3 23/—/1
C3	140/56; 33/16 13/10; 30/17 3/—/0	383/154; 97/38 3/0; 94/59 32/—/2	125/50; 31/14 8/5; 29/17 6/—/0	118/48; 31/13 0/0; 30/18 13/—/0	114/46; 29/9 3/0; 29/20 5/—/0	102/41; 26/13 12/5; 25/13 1/—/0
C4	140/56; 37/31 28/19; 6/6 3/—/0	383/154; 108/66 24/2; 37/42 20/—/2	125/50; 34/30 20/14; 13/4 4/—/0	118/48; 35/22 1/0; 14/13 11/—/0	114/46; 33/20 0/0; 12/13 6/—/1	102/41; 30/18 10/2; 13/12 3/—/0
C5	140/56; 37/33 27/10; 4/4 5/—/1	383/154; 108/98 64/0; 18/10 34/—/0	125/50; 34/30 13/11; 9/4 17/—/0	118/48; 35/28 0/0; 3/7 28/—/0	114/46; 33/30 2/1; 9/3 27/—/0	102/41; 30/24 15/0; 15/6 8/—/0
C6	140/56; 37/34 29/22; 1/3 5/—/0	383/154; 108/96 7/3; 1/12 70/—/0	125/50; 34/32 21/20; 0/2 11/—/0	118/48; 35/29 0/0; 1/6 19/—/0	114/46; 33/29 0/0; 0/4 27/—/0	102/41; 30/26 11/6; 0/4 10/—/0

Failure examples link an attributable episode to its supplied evidence, submitted artifact, violated criterion and reference or evaluator outcome. Dataset-injected faults are distinct from observed participant failures. An illustrative case does not estimate error prevalence; frequencies require the corresponding denominators and classification rule. Engineering errors, invalid submissions, participant timeouts and unavailable evaluation remain separate under the declared protocol.

Table 20: **Error analysis of failed submissions.** Categories are mutually exclusive and grouped by task; submission-contract violations are listed separately.

Failure type	Description	Count
Submission contract		
Invalid structured submission	Missing or invalid fields, invalid coordinates, duplicate identifiers, or other submission-contract violations	17
TASK-1		
Incorrect response form	Answer does not follow the required response form, such as supplying a named choice instead of a numeric design	67
Incorrect design or optimum	Submitted design violates analytical constraints or fails to achieve the requested optimum	20
TASK-2		
Device identification only	Incorrect, missing, or extra device predictions; localization and port checks pass	34
Port annotation only	Incorrect or incomplete port annotations; device and localization checks pass	65
Device identification and ports	Both device identification and port annotation checks fail	16
Device identification and localization	Both device identification and bounding-box localization checks fail	2
Device identification, localization, and ports	All three recognition checks fail	87
TASK-3		
Incorrect circuit name only	Circuit identity is incorrect; functional interpretation passes	11
Incorrect function only	Functional interpretation is incorrect; circuit identity passes	2
Incorrect circuit name and function	Both circuit identity and functional interpretation fail	1
TASK-4		
Incorrect reconstructed topology	Submitted netlist fails the required joint typed-port graph-equivalence check	24
TASK-5		
Incorrect fault type/event set only	Fault classification or predicted event set is incorrect; localization check passes	13
Incorrect fault localization only	Fault localization fails; fault type/event-set check passes	3
Incorrect fault type and localization	Both fault type/event-set and localization checks fail	6
TASK-6		
Invalid GDS artifact	Generated repair file is malformed and cannot be parsed as GDSII	8
Incorrect repaired geometry	Parseable repair violates geometry, topology, connectivity, or protected-region preservation requirements	13
Total	Failed submitted results	**389**