FERMIONS ASI CORP.
AI-powered intelligence for digital chip design
www.fermions.co

W H I T E P A P E R

SAiGE: Closed-Loop Agentic Bug-Fixing on a Production EDA Toolchain

A Triaged, Confidence-Tiered Fix Campaign on OpenROAD

June 2026

Table of Contents

SAiGE: Closed-Loop Agentic Bug-Fixing on a Production EDA Toolchain

Abstract

A Fermions technical campaign, executed with SAiGE — Fermions' autonomous RTL/EDA debugging agent (fermions.co). Every fix was independently verified and reviewed/DCO-signed by a human submitter (@saurav-fermions).

We report a SAiGE-driven autonomous, closed-loop bug-fixing campaign on **OpenROAD**, the open-source RTL-to-GDSII physical-design flow (`The-OpenROAD-Project/OpenROAD`). Working from open GitHub issues, **SAiGE** — Fermions' autonomous RTL/EDA debugging agent — triaged **24 issues** across eleven OpenROAD subsystems: global routing (`grt`), resizing (`rsz`), OpenDB (`odb`), detailed placement (`dpl`), power distribution (`pdn`), clock-tree synthesis (`cts`), detailed routing (`drt`), power intent (`upf`), timing (`dbSta`), the core (`ord`), macro placement (`mpl`), and the build system.

Of these 24 issues, **17** received independently rebuilt, test-verified fixes with real fail-before/pass-after evidence, including bounding a global-route runtime blow-up that had turned a ~3-minute route into ~20 hours. **4** were confirmed already fixed on the base commit, including 2 with new regression tests and 2 report-only findings. **2** are strictly safe defensive guards whose original crashes could not be reproduced, and **1** is an explicitly opt-in scaffold.

Each code fix was delivered as a single git format-patch on one pinned OpenROAD base commit, with a per-issue report explaining the root cause, the files changed, the verification performed, and any gaps that could not be fully verified. The two confirmed-already-fixed report-only items carry no patch. A key feature of this SAiGE campaign is honest confidence tiering: each result is labeled as HIGH, ALREADY-FIXED, DEFENSIVE-UNREPRODUCED, or SCAFFOLD. In simple terms, this means SAiGE separates fully verified fixes from issues that were already fixed, safe-but-unproven defensive changes, and unfinished scaffolds. As a result, only independently validated fixes are presented as ready, while uncertain or incomplete work is clearly labeled and never over-claimed.

1. Introduction

OpenROAD is a large, actively maintained C++ EDA codebase implementing the full digital place-and-route flow. Unlike RTL design generation, which is the usual target for agentic EDA benchmarks, fixing OpenROAD is a software-engineering-in-a-domain problem: defects manifest as crashes, runtime blow-ups, wrong geometry, or wrong timing decisions deep inside numerical and graph algorithms such as maze routing, Steiner trees, legalization, and rebuffering. The oracle is the tool's own module test suites plus physical-design sanity, not a golden ISA model.

This regime stresses an agent differently from RTL repair:

- **No single golden reference:** Correctness is judged per subsystem by ctest suites, fault injection, scale measurement, and physical-design invariants — not one simulator.
- **Reproduction is sometimes impossible:** Some issues' original failing designs are proprietary or external and were unavailable, so SAiGE must distinguish "I fixed and proved it" from "I wrote a safe guard but could not reproduce the original crash" — and say so.
- **Blast radius matters:** A change to global routing or legalization that subtly alters QoR is worse than no change. Fixes are scoped to be byte-for-byte identical on the already-working path wherever possible.

This paper presents how SAiGE applied a closed-loop, evidence-first debugging methodology to a production-scale EDA toolchain, moving beyond isolated RTL generation into real-world software debugging for physical-design infrastructure.

1.1 Contributions

- **17 independently verified fixes** across `grt`, `rsz`, `odb`, `pdn`, `cts`, `drt`, `upf`, `dbSta`, `ord`, and the build system, each rebuilt from committed source and validated by the affected module's full ctest suite with real fail/pass or measured before/after evidence.
- **A confidence-tiering discipline** - HIGH, ALREADY-FIXED, DEFENSIVE-UNREPRODUCED, and SCAFFOLD - that makes the campaign's epistemic state explicit and prevents over-claiming.

- **A self-contained, reproducible fix pack**: one pinned base commit, one `git format-patch` per issue, a per-issue report, an `apply_all.sh` that recreates one branch per fix, and an authoritative `VERIFICATION.md`.
- **Three deep case studies** - a global-route runtime blow-up fix, a fatal maze-routing abort turned into graceful edge recovery, and 45° LEF-polygon decomposition - illustrating root-cause-first debugging on real EDA algorithms.

2. Target and Method

Agent: The campaign was carried out by **SAiGE**, Fermions' autonomous RTL/EDA debugging agent (fermions.co). SAiGE triaged the issues, localized and root-caused each defect, wrote the fixes and reproducers, and ran the verification. A human submitter (@saurav-fermions) reviewed and DCO-signed every patch before upstreaming.

Target: OpenROAD at pinned base commit `4db8ad9fda299ac0142ab20da6180df1869b4422` (`The-OpenROAD-Project/OpenROAD`).

Fix unit: Each fix is a `git format-patch` applied with `git am` onto the base, producing a branch `fix/<issue>`. No full source is vendored; the base repo is cloned fresh and patched.

Verification protocol per fix:

1. **Independent rebuild** — the fix branch is rebuilt from its committed source, rather than trusting the agent's prior claim.
2. **Module suite** — the affected module's full ctest suite is run using `ctest -R '^<module>\.'`; pass counts are recorded.
3. **Real fail/pass or scale** — wherever possible, verification uses a test that fails on the unfixed base and passes after the fix, a measured before/after at scale, or fault injection.
4. **Explicit gaps** — each report states what could not be verified, such as cases where the original crash was not reproducible.

Confidence tiers:

- **HIGH** — independently rebuilt, module suite passed, and real fail/pass or measured before/after evidence available.
- **ALREADY-FIXED** — the defect is already absent on the base commit; confirmed, and where a regression test was missing, one was added. No new code patch, or test-only.
- **DEFENSIVE-UNREPRODUCED** — sound root cause and a strictly safe guard, but the original crash could not be reproduced with available in-repo data. This is not treated as closing the issue.
- **SCAFFOLD** — opt-in, default-off no-op; an extension point, not a finished feature.

3. Results

3.1 Tier distribution

Tier	Count	Issues
HIGH — independently verified	17	#5617, #8013, #8075, #8466, #8941, #9563, #9995, #10082, #10256, #10273, #10350, #10414, #10474, #10490, #10602, #10622, #10668
ALREADY-FIXED — confirmed on base	4	#6704, #9070 regression test added; #9724, #4252 report-only
DEFENSIVE-UNREPRODUCED — safe guard	2	#8351, #10677
SCAFFOLD — opt-in, default-off	1	#8871
Total triaged	**24**	22 carry patches; #9724 and #4252 are report-only

3.2 By subsystem (PR-ready set: 17 verified + 2 test-adding already-fixed)

Module	Issues	Representative independent check
`grt` — global route	#8466, #8941, #10273, #10474	131/131–132/132 grt; #8466 scale 19.1s → 4.4s on 150 NDR nets
`rsz` — resizer	#8013, #8075, #10622	238/238–239/239 rsz; cross-clock hold detection real fail/pass
`odb` — OpenDB	#10082, #10256, #10668	82/82–92/92 odb; 45° polygon decomposition, numeric EDGETYPE
`dpl` — placement	#6704 already-fixed + test	118/118 dpl
`pdn` — power grid	#10490	151/151 pdn
cts — clock tree	#9070 already-fixed + test, #10602 doc	Regression test added; doc cross-checked vs TritonCTS code

drt — detailed route	#9995	20/20 drt; per-net auto-taper, default unchanged
`upf`	#5617	5/5 upf
`dbSta`	#10414	71/71 dbSta; supply ports restored in write_verilog
`ord` / build	#9563, #10350	Symbol-archive link fix; default threads → nCPU

4. Case Studies

4.1 #8466 — A default flip that turned a 3-minute route into ~20 hours (grt)

A CTS change flipped the default NDR strategy from `NONE` to `HALF`, attaching a non-default routing rule to **every** clock net. Global routing then tries to honor each NDR, fails for capacity, disables them one at a time with `GRT-0273`, and restarts its entire congestion/overflow iteration loop after each disable. With many clock-NDR nets, this becomes O(N) full overflow loops: a ~3-minute global route became ~20 hours.

SAiGE fix: SAiGE bounded the number of soft-NDR-triggered loop restarts while leaving the CTS feature intact, rather than reverting the feature. Measured on a Nangate45 stress design with wide NDR on 150 nets, runtime improved from 19.1s to 4.4s and restarts dropped from 1001 to 487. The fix passed 132/132 `grt` tests.

4.2 #10273 — Fatal abort turned into graceful recovery (grt)

After antenna repair, incremental global routing could hit `[ERROR GRT-0183] heap underflow during 3D maze routing`, a [[`noreturn`]] logger error that aborts the whole flow, followed by a read of `src_heap_3D_[0]` on an empty vector. The empty-queue state is simply "no path found for this edge within the constrained reroute region," not a fatal condition.

SAiGE fix. SAiGE changed the behavior to treat heap underflow as "no path for this edge": recover the edge and continue instead of aborting. The fix was verified by fault injection, with `GRT-0183` before and clean recovery after. The fix passed `132/132 grt` tests.

4.3 #10256 — 45° LEF polygons mangled, rectilinear path untouched (odb)

The reported symptom — "POLYGON statements dropped from LEF" — was only partly accurate. Rectilinear polygons were already ingested, decomposed to `dbBox`, and propagated correctly. The remaining defect was **non-rectilinear 45° or sloped polygons**, such as octagonal IO pads, being corrupted during decomposition.

SAiGE fix. SAiGE scoped the change exactly to the sloped-polygon case while leaving the already-working rectilinear decomposition byte-for-byte identical. The fix passed 82/82 `odb` tests. This case illustrates SAiGE's discipline of narrowing the fix to the genuinely broken path and not "fixing" what already works.

5. Honest Limitations

The campaign's verification record explicitly separates proven fixes from safe but unproven guards.

- **#10677 and #8351 — DEFENSIVE-UNREPRODUCED:** The original crashes could not be reproduced with in-repo data. Issue #8351's resizer was re-architected roughly 8500 commits later, and #10677 requires the original `.odb`. The fixes are strictly safe guards, with full module suites passing — 118/118 `dpl` for #10677, and #8351 additionally ASan-clean before/after — but they are **not** proven against the exact reported failure and must not close their issues without the original designs.
- **#8871 — SCAFFOLD:** The opt-in timing-driven macro-placement cost term works mechanically, with default `0.0` meaning no-op and no golden change, but it uses a connectivity proxy for criticality rather than real STA slack. It is an extension point, not a finished feature.
- **No original-customer-design validation:** Those test cases live on external or PII servers and were unavailable. Verification used in-repo designs, synthetic reproductions, fault injection, and scale stress.
- **Generic patch authorship:** Patches were generated under a generic `Ubuntu` git identity and must be re-attributed to the real contributor identity before upstreaming.

6. Recommended Upstreaming Posture

OpenROAD is a major, CI-gated project with an active maintainer team. The campaign therefore proposes a **tier-gated, staged** submission rather than a bulk dump.

- **Submit as PR-ready:** the 17 HIGH-tier fixes and the 2 test-adding ALREADY-FIXED items, #6704 and #9070, as focused single-issue PRs. Each PR should link its issue and carry the report's verification evidence. The recommended starting point is a few self-contained, high-impact fixes such as #8466 scale improvement, #10273 crash recovery, and #10082 crash-hole handling, to calibrate maintainer response before opening the rest.
- **Hold or file as discussion, not as a fix:** #8351 and #10677. These defensive-unreproduced guards should be offered upstream only with the explicit caveat that the original crash is unreproduced, while asking maintainers for the original design.
- **Do not submit as a fix:** #8871. At most, it should be submitted as an RFC or draft describing the extension point.
- **Report-only, no PR:** #9724 and #4252, which were already fixed on base. The report is the deliverable, optionally accompanied by a regression test.

7. Relationship to the Ara Campaign

This OpenROAD campaign is the EDA-tool analogue of Fermions' PULP Ara RVV correctness campaign. The same SAiGE methodology — closed-loop execution, root-cause-first debugging, evidence before claims, and human-reviewed/DCO-signed submission — is applied to a different layer of the semiconductor stack: physical-design software rather than vector RTL.

The key methodological addition in the OpenROAD campaign is **explicit confidence tiering**. This is necessary because some OpenROAD defects are filed against unreproducible external designs, so "verified fix" and "safe-but-unproven guard" must be kept rigorously distinct.

8. Conclusion

SAiGE, Fermions' autonomous RTL/EDA debugging agent, triaged 24 OpenROAD issues and produced 17 independently verified fixes across nine subsystems plus the build system. The campaign includes measured before/after evidence on a global-route runtime blow-up and fault-injected proof on a fatal maze-routing abort, while explicitly tiering confidence and refusing to over-claim the 3 unreproduced or unfinished items.

The deliverable is a reproducible, single-base-commit fix pack with per-issue reports and an authoritative verification record, structured for staged, tier-gated upstreaming. This campaign demonstrates SAiGE's ability to move beyond RTL generation into production-scale EDA debugging, where correctness requires root-cause analysis, disciplined verification, careful blast-radius control, and honest confidence reporting.

References:

[1] T. Ajayi et al., "*OpenROAD: Toward a Self-Driving, Open-Source Digital Layout Implementation Tool Chain,*" in Proceedings of the Design Automation Conference, 2019. Available: https://par.nsf.gov/servlets/purl/10171024

[2] The OpenROAD Project, "*OpenROAD: Foundations and Realization of Open, Accessible Design*," project website. Available: https://theopenroadproject.org/

[3] The OpenROAD Project, "*OpenROAD,*" GitHub repository. Available: https://github.com/The-OpenROAD-Project/OpenROAD

[4] The OpenROAD Project, "*OpenROAD-flow-scripts*," GitHub repository. Available: https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts

[5] The OpenROAD Project, "*OpenROAD Documentation*," documentation website. Available: https://openroad.readthedocs.io/

[6] The OpenROAD Project, "*OpenDB,*" OpenROAD documentation. Available: https://openroad.readthedocs.io/en/latest/main/src/odb/README.html

[7] The OpenROAD Project, "*Parallax Static Timing Analyzer / OpenSTA*," OpenROAD documentation. Available: https://openroad.readthedocs.io/en/latest/main/src/sta/README.html

[8] The OpenROAD Project, "*Detailed Routing,*" OpenROAD documentation. Available: https://openroad.readthedocs.io/en/latest/main/src/drt/README.html

[9] A. B. Kahng, L. Wang, and B. Xu, "*TritonRoute: The Open-Source Detailed Router*," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 40, no. 3, pp. 547–559, 2021. Available: https://ieeexplore.ieee.org/document/9120211

[10] C.-K. Cheng, A. B. Kahng, I. Kang, and L. Wang, "*RePlAce: Advancing Solution Quality and Routability Validation in Global Placement*," IEEE Transactions on Computer-Aided Design of

Integrated Circuits and Systems, vol. 38, no. 9, pp. 1717–1730, 2019. Available: https://ieeexplore.ieee.org/document/8418790

[11] M. Pan and C. Chu, "*FastRoute: A Step to Integrate Global Routing into Placement*," in Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, 2006. Available: https://dl.acm.org/doi/10.1145/1233501.1233596

[12] N. Pinckney, C. Deng, C.-T. Ho, Y.-D. Tsai, M. Liu, W. Zhou, B. Khailany, and H. Ren, "*Comprehensive Verilog Design Problems: A Next-Generation Benchmark Dataset for Evaluating Large Language Models and Agents on RTL Design and Verification*," arXiv:2506.14074, 2025. Available: https://arxiv.org/abs/2506.14074

[13] Y. Lu, S. Liu, Q. Zhang, and Z. Xie, "*RTLLM: An Open-Source Benchmark for Design RTL Generation with Large Language Models*," arXiv:2308.05345, 2023. Available: https://arxiv.org/abs/2308.05345

[14] S. Liu, Y. Lu, W. Fang, M. Li, and Z. Xie, "*OpenLLM-RTL: Open Dataset and Benchmark for LLM-Aided Design RTL Generation*," arXiv:2503.15112, 2025. Available: https://arxiv.org/abs/2503.15112

Notice

This document is provided for informational purposes only and shall not be regarded as a warranty of any functionality, condition, or quality of any product or system. Fermions ASI Corp. (“Fermions”) makes no representations or warranties, expressed or implied, as to the accuracy, completeness, or reliability of the information contained in this document and assumes no responsibility for any errors or omissions.

This document does not constitute a commitment to develop, release, or deliver any product, feature, code, or functionality described herein. Fermions reserves the right to make changes, corrections, enhancements, or modifications to the content of this document at any time without notice.

The information presented is intended for general guidance only. Users are responsible for independently evaluating the applicability of this information for their specific use cases and for performing all necessary validation, testing, and verification.

Fermions shall not be liable for any direct, indirect, incidental, special, or consequential damages arising from the use of, or reliance on, this document or the information contained herein.

No license, express or implied, is granted under any intellectual property rights of Fermions. References to third-party products, services, or technologies do not constitute endorsement or recommendation and may require separate permissions or licenses from their respective owners.

Reproduction, distribution, or use of this document, in whole or in part, is permitted only with prior written consent from Fermions and must be reproduced without modification and with all notices included.

Important Disclaimer

THIS DOCUMENT IS PROVIDED “AS IS.” FERMIONS MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

Copyright

Fermions — AI-powered intelligence for digital chip design

contact@fermions.co | *www.fermions.co*