WENGOLD Open-Core Strategy

How an open validity kernel becomes a business — v0.5.0

Recommendation: keep the kernel, graph and ledger open under MIT OR Apache-2.0, and charge for the shared evidence layer around them. A validity kernel is only worth anything if the team relying on it can audit how it reached its answer; a closed kernel nobody can inspect cannot be trusted with the claim that a result is reusable. The durable asset is not the algorithm, it is the accumulated, verifiable record of what was proven, against what, and when.

The problem is the rerun decision, not the tools

After a change, a verification team chooses between two bad options: re-run everything, or re-run a subset chosen by hand. The second is faster and undocumented. When a result is reused, nothing records why it was safe to reuse it, what it actually established, or what changed underneath it. That decision is the thing WENGOLD makes explicit and auditable.

The evidence that this is a real cost is structural: on OpenTitan's AES block, a change to one module re-runs 21 of 232 results (90.9% reduction) on a dependency graph that is 96.9% resolved. On CVA5, a change to the core top re-runs 11 of 124 (91.1%). Those numbers measure dependency fan-out, not wall-clock — see the caveat below.

What is open, and what is not

Open (MIT OR Apache-2.0): the graph and its front-ends, the validity kernel, the ledger, the formal loop, the tool adapters and the MCP server. Anyone can run the whole thing locally, forever, with no account.

Commercial: the hosted evidence service (shared ledger across a team, retention, cross-project search), organisation-wide waiver and acceptance policy, integrations that require licensed infrastructure (commercial simulators, Jira, Slack), and support with a response commitment.

The boundary is chosen so that the open part is genuinely useful alone. The commercial part is what a team needs when more than one person has to agree that a result may be reused.

The moat is the record, not the algorithm

The validity rules are not secret and should not be. What compounds is the ledger: a project's history of claims, evidence, waivers and the decisions that accepted them. Moving that history to a competitor means re-establishing every result, which is precisely the cost the product exists to remove. The switching cost is the accumulated evidence.

Competition, stated without decoration

Closed verification-automation startups sell a managed flow, and several open agent projects automate test and debug. WENGOLD does not compete on agent count or on tool coverage; it competes on one claim neither makes: that a stored result can be shown to still hold, or be marked unknown. Incumbents can add this feature, and an agent project can adopt the kernel. The defence is that the kernel is open, so adopting it is cheaper than rebuilding it, and the evidence it accumulates stays with the user's own record.

What is proven, and what is not

Claim	Status
The kernel classifies correctly and conservatively	Proven: 160 tests, invariant fuzzing over 400 graphs, independent reference implementation
The graph works on real RTL	Proven: OpenTitan hw/ip 3.6% unresolved, CVA5 1.7% unresolved
A verdict can come from a real tool	Proven: verilator lint, yosys synthesis, sby proof, cocotb simulation, driven end to end
Reruns are actually reduced on a real flow	NOT proven: the reduction is computed over synthetic results. Needs a design partner.
Teams will pay for shared evidence	NOT proven: no design partner, no pricing evidence.

The honest summary is that the technical thesis is demonstrated and the commercial thesis is not. The two unproven rows are the ones that decide whether this is a product.

Risks that would falsify the thesis

- **Front-end completeness is the binding constraint.** Getting the file set wrong once took OpenTitan from 3.6% to 41% unresolved and collapsed the reduction to nothing. If WENGOLD cannot take a project's own build metadata, adoption stalls at the first real repository.
- **The integration cost may exceed the saving.** If a team must rewrite its regression harness to feed WENGOLD its results, no one will do it for an unproven tool. A JUnit and log importer is a prerequisite, not a feature.
- **The saving may not be where the money is.** The expensive checks are integration tests with wide dependency closures. The 90% figure comes from a suite dominated by unit-level results.

Next steps, owned and dated

Action	Owner	By
Measure rerun reduction on one real block with real tools running	WENGOLD	Before any pricing work
Read FuseSoC .core and .f file sets so no include dirs are passed by hand	WENGOLD	Next release (v0.6)
Import an existing suite (JUnit XML, UVM logs) into results with contracts	WENGOLD	Next release (v0.6)
One design partner running it on a block they already verify	Thejas	Before the first paid tier

The test of this strategy is narrow and falsifiable: one team, one block, a measured reduction in checks re-run, with the decision to reuse each result recorded and auditable. Everything else in this document is a plan until that number exists.

Figures in this document are measured from the v0.5.0 repository or stated as unproven. No market size is claimed here, because none has been validated.