INVESTOR BRIEF · CONFIDENTIAL & PROPRIETARY

# CratonOS™

## We make LLMs trustworthy for chip design.

This packet, page by page:



NEEDS

### The problem we solve

GenAI writes RTL in seconds — but **SystemVerilog captures behavior, not intent**, so **functionally incorrect RTL looks identical to correct RTL**. Models generate hardware they can't prove, can't cover, and forget every project. With first-silicon failure at **~$50M** / 2–3 years and **~$30B/yr** in re-spins, unverified AI multiplies the risk. Teams need AI they can **trust to sign off on silicon**.

APPROACH

### How we solve it

A chip-design **operating system** beneath the models, giving them what they lack — **knowledge, verification, memory, grid-scale execution** — under one brain. One intent is realized **concurrently in independent representations that must agree**, and an **independent model signs** it: every concern resolves to **GO / FIX / BLOCK**. Not an EDA tool, not an LLM wrapper — the layer that closes prompt → GDSII.

BENEFIT

### What the customer gets

Teams get AI output they can **actually ship** — correct-by-construction, verification-first, not code they prove by hand. It decides *during* the run like a senior engineer; its **memory compounds every tape-out** so each design starts ahead of the last; and IP, data & learnings **stay in the customer's network** (on-prem / air-gapped). Thesis: fewer escapes, a step-change in verification efficiency, **5–10×** on derivatives.

COMPETITION

### Competition & our superiority

**Category creation, not feature competition.** Like Windows, iOS & Android, durable value lives in **owning the OS layer** beneath the apps — chip design today is all apps (tools, LLMs, agents) with no OS underneath. Point tools, single-purpose AI-RTL assistants, and model companies each miss the full loop; CratonOS makes **any** model trustworthy and is **used across all, displaced by none**. Patent-pending.

**VISION** Fully autonomous chip-design intelligence — compounding every tape-out, until *future chips design themselves.*

THE OS · THE ECONOMICS

# What the OS is — and how it improves cost, quality & time-to-market.

**CratonOS is a self-learning operating system for chip design** — the ground-truth layer beneath every model, tool and engineer. It gives GenAI the **design knowledge, verification, memory and grid-scale execution** it lacks, as **four cognitive layers under one orchestration brain** (left). Bolting more tools and headcount onto fragmented flows only adds cost; an OS that **encapsulates the complexity** is the only thing that scales (right).


↻ **Knowledge → Action → Correctness → Memory → back.**
Each layer feeds the others; every tape-out makes the OS smarter.


### ▸ The gap

More tools, agents & headcount **multiply cost & invocations** — with no unified intent, proof or memory, output can't be trusted or reused.

### ▸ The opportunity

Not another tool or more people — a **self-learning, LLM-native OS** that absorbs the complexity: **50%+ efficiency & autonomy.**

### ▸ The solution

**Vertical integration** — the architect sets intent; the OS authors, verifies & closes RTL, TB & coverage, **compounding every tape-out.**

CratonOS **encapsulates the complexity** — EDA tools · LLMs · agents · compute grid — in **Guide, Peer or Autonomous** modes; the architect works only on intent. Goal: **50–70% lower cost**, higher quality and faster time-to-market — across design, verification, execution & learning, compounding every tape-out *(directional)*.

THE CHIP-DESIGN OPERATING SYSTEM FOR THE GENAI ERA

# CratonOS™

## We make LLMs trustworthy for chip design.

A self-learning, agentic OS that gives reasoning models like **Claude & Gemini** the design knowledge, verification, memory, and grid execution they lack on their own — turning English intent into **verified, signed-off** RTL, from prompt to GDSII. Not another EDA tool. Not an LLM wrapper.

**CATEGORY**
**Category creation**

The **first chip-design operating system** — four cognitive layers under one orchestration brain that compound with every tape-out. A category, not a competitive tool.

**FOUNDERS**
**Both halves of the chip**

The **only AI-chip founding team** spanning the full RTL-to-GDSII flow — **95+ years**, **75+ tape-outs**: front-end + verification **and** back-end + implementation.

**PROOF**
**Provable by design**

One intent → **SystemVerilog · C · SystemC** that must agree, then an **independent model signs it**. A **peer verdict bus** runs every concern — coverage-driven DV, synthesis/PPA, lint, CDC, connectivity — into one **GO / FIX / BLOCK**.

**MOAT**
**Compounding memory**

**Two memories compound into intelligence no single engineer can hold** — decades of expertise encoded in its tools, plus a process memory whose **self-learning engine builds intelligence dynamically**: RTL and verification generation rules · coverage · waivers · bugs · synthesis · implementation — growing every tape-out. Each derived SoC starts ahead of the last. **Patent-pending** across multiple engines.

**STATUS**
**Benchmarked · in stealth**

**Benchmarked across 109 production IP designs** — spanning CPU, GPGPU, crypto, security, memory & peripherals. In early discussions with selected customers. Currently building in **stealth**.

**$30B/yr**
lost to silicon re-spins
Bloomberg Intelligence, 2025

**$50M**
per first-silicon failure
Mordor Intelligence, 2026

**2–3 yr**
delay from an A0 failure
industry consensus

**LLMs write RTL they can't prove.**
CratonOS makes AI trustworthy for silicon.

# The substance behind the OS

**CratonOS is a self-learning, agentic operating system for chip design.** It augments reasoning LLMs (Claude, Gemini) with the design knowledge, persistent memory, verification, and autonomous execution they lack on their own — closing the loop from natural-language intent to **verified, signed-off** design, RTL to GDSII. The problem: LLMs write RTL in seconds — but functionally incorrect RTL looks identical to correct RTL. They can't prove it, can't close coverage, and forget every project; generation without verification just multiplies the ~$30B/yr re-spin spend.

## ▸ What it delivers

- ✓ **Verification-first, correct-by-construction** — verifies intent before execution (no EDA runtime or token spend on unverified intent); zero silicon escapes, **50%+** verification efficiency, **5–10×** productivity.
- ✓ **Decides during the run, not after** — the OS rules on every sign-off concern *while the run is live*, like a human engineer — not once the regression has finished.
- ✓ **Compounding memory** — two memories: expertise **encoded into the tools**, plus a **process memory** the databases learn across the whole design-and-verification flow and compound every tape-out — up to **50–70%** autonomy on derivatives.
- ✓ **IP & data protection** — runs on-prem / air-gapped or with local LLMs; design data, IP & learnings stay inside the customer network.

## ▸ Why it's defensible

- ✓ **A systems invention, not a toolchain** — DSLs + compilers + self-learning DBs + 450+ encoded rules as one closed loop; can't be assembled from existing point tools.
- ✓ **95+ years, rebuilt LLM-native** — decades of tape-out expertise (which interactions must happen, and in what order) encoded as native intelligence; the four layers required concurrent, multi-year discovery.
- ✓ **GenAI companies need it, can't build it** — reasoning-engine companies lack chip-design failure diagnosis; a structural partnership, not competition. **Patent-pending** across multiple engines.
- ✓ **Enterprise agents at grid scale** — thousands of agents on the real compute grid; native LLM stacks (Claude/Gemini skill or MD files) can't — cost, latency, context bloat and hallucination cap them at a few threads.

HOW IT'S ORGANIZED — FOUR COGNITIVE LAYERS UNDER ONE ORCHESTRATION BRAIN (ARCHITECTURE & METHODS UNDER NDA)

**Knowledge** encode intent → **Action** autonomous execution → **Correctness** verify & close → **Memory** compounds every tape-out

VISION

**Full autonomy**

Fully autonomous chip-design intelligence — a self-learning system that **compounds every tape-out**, until **future chips design themselves**. Currently in **stealth**.

WHAT COMPETING SOLUTIONS LACK

# Seven things no point tool, LLM or agent does.

Each line below is a **working engine** — built, released and **ready for production engagement**, not a roadmap item.

### The OS composes itself, per task

`the composition kernel`

Not a fixed toolchain. For each intent the kernel assembles a different pipeline of languages, compilers, agents and models **in the order a chip flow actually requires**, and re-composes as it learns. Building an agent is easy; **encoding the ordering** is the invention.

### Design-configuration equivalence

`design–configuration engine`

SystemVerilog has **no construct for design configuration** — the industry never built one, and no EDA vendor ships the check. CratonOS declares it once, block → IP → SoC, and proves every tool downstream builds that same design. **A pre-requisite for trusting an LLM with silicon**: a model can emit perfect RTL and still be building the wrong variant.

### Prompt-to-intent extraction & RTL generation

*Intent closes before a token is spent; the generator never grades its own work*

`intent extraction · peer verdict bus`

**Design intent** and — separately — **verification intent** are made precise before anything generates. The rules are **compiler artifacts** — versioned, inspectable, applied with no model in the loop — which is how **proprietary knowledge no model has ever been trained on** becomes machine-applicable. An engineer adds their own; a third engine distils shipped RTL back into intent to **learn the rules a company already follows**.

Every generated artifact is then fanned out to specialist verifiers — lint, CDC, connectivity, configuration, coverage, PPA — and aggregated into a single **signed verdict**. The generator never grades its own work.

### Dynamic coverage-driven verification

*Coverage decided during the run — not after it*

`stimulus generator + coverage engine`

Stimulus is synthesized against a **live** coverage monitor while the simulation runs; conventional coverage-driven verification can only decide once a regression has finished. Coverage is **simulator-agnostic** — closure is never gated on which licence a team holds.

### Enterprise agentic execution

`enterprise agent engine`

An engineer's whole analysis — logs, spec, coverage, memory, RTL, synthesis, waivers — becomes a **programmable agent** that decides **while the tool is still running**. Authored by prompting a model, then dispatched **across the grid with no model present**. A skill file needs one every single time, at token cost.

### One source of truth — design, configuration, execution

`design & tool knowledge layer`

An LLM sees **source text, not an elaborated design**. This is the compiled, unified representation — design, configuration and tools in one — that grounds every model and drives every EDA tool, and it lets a run be **watched while it runs**.

### Self-learning & memory

*It compounds every tape-out — and some findings can never be waived*

`memory vault · waiver governance`

Intent, coverage, waivers, configuration and every fix are inherited by the next derivative. Debug is keyed **by the shared IP a bug travels along, not by project**, so one engineer's fix unblocks every design instantiating it. It never leaves the customer's network and **trains no one's model**.

A **unified waiver learning DB** spans every sign-off concern, with a **risk tier that makes silicon-safety findings permanently un-waivable** — and approval that is never automatic. Point tools waive locally, with no one holding the invariant.

THE HALF NO ONE ELSE IS LEARNING

Prompt to RTL, testbench generation, RTL to GDSII — **every transformation in the flow is structural**: the answer is already determined and only the execution is hard. That is the half every tool, flow and AI entrant is automating, and in time they will. The **creative** half sits above all of them — how an architect decomposes the problem, what they choose to check, which trade-off they take. It has never been captured: it lives in engineers' heads and differs at every company. CratonOS learns **both**, and the creative half stays **the customer's own IP** — inside their network, inherited by their next derivative, training no one's model. **An engineer tames the flow with their own proprietary intelligence.**

*Cost & efficiency figures are directional (a value thesis, proven keyless on open designs). Patent-pending · shared in confidence.*

## Introducing CratonOS — the first chip-design OS.

The only team that spans both halves of the flow — **95+ years and 75+ production tape-outs**, SoC/IP design & verification through back-end implementation to GDSII.

**CratonOS™ Inc.**
info@cratonos.ai
cratonos.ai