MEASURED · not asserted

THE LAB · A RESEARCH LANE

From the silicon up: a language that proves itself

QUANTA is a language an AI writes and a machine proves. It self-hosts byte-identical across two independent compilers, targets four backends gate-clean, and now lexes and parses itself — in itself. Here is what is measured, and what is still open.

A measured research lane

Shipped — measured, gate-clean

COVER: A MEASURED OSCILLOSCOPE WAVEFORM ON A LAB GRATICULE

Vocabotics

A VOCABOTICS LAB NOTE

From the silicon up: a language that proves itself

QUANTA is a language an AI writes and a machine proves. It self-hosts byte-identical across two independent compilers, targets four backends gate-clean, and now lexes and parses itself — in itself. Here is what is measured, and what is still open.

PREPARED

2 July 2026

VERIFIED BY

Jon Ross — Founder, Vocabotics — 15 years building safety-critical systems

HOW TO READ IT

Drafted with AI, verified by a named human. Sources at the back.

01 The claim, in brief

> ***Compile it twice, with two different compilers, and get identical bytes. The agreement is the proof there is no backdoor.***
>
> The costly signal

The measured result

Every number carries its own honesty tier — the tiering is the credibility.

53/53
diverse-double-compile, byte-identical
🟢 SHIPPED

4
backends gate-clean
🟢 SHIPPED

4.2×
faster regalloc, 3× smaller
🟢 SHIPPED

~271 / 600
chapters gate-green
🔬 RESEARCH

Most languages are written for people and then, with effort, made fast for machines. QUANTA is the other way round: a language an AI writes, and a machine proves correct, from the silicon up. It is not a wrapper over LLVM or a DSL that lowers to someone else's runtime. It is a compiler, four backends, and a growing standard library, built so that the *proof* travels with the code.

This page is the deep, honest account of what is measured today — and, in the same breath, what is still open. That second half is not a disclaimer. It is the point: a claim you can check is worth more than one you can't.

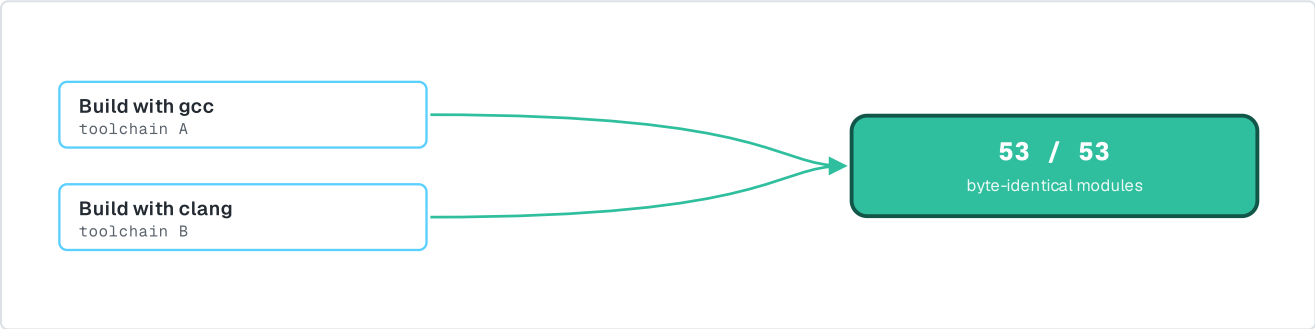
02 The self-hosting proof: byte-identical, twice

🟢 SHIPPED — MEASURED, GATE-CLEAN

The headline result is a costly signal a competitor cannot fake on a slide. `qc2`, the QUANTA compiler, compiles itself — and we verify the output with **diverse double-compiling**: build the compiler with `gcc`, build it again with `clang`, and compare the bytes. Across **53 of 53** modules the output is **byte-identical.** 🟢

Why this matters: identical bytes from two independent toolchains is evidence that no compiler has smuggled anything into the binary — the classic "trusting-trust" attack. The agreement *is* the proof. It is the same discipline a safety assessor applies to a signalling system: don't trust the tool, verify the artefact.

The front-end now clears an even higher bar. `qcc11` — a code generator **written in QUANTA** — compiles the full `qlexer.q` and `qparser.q` (a real five-parameter lexer and a recursive-descent parser with Pratt precedence), and the native output matches the interpreter exactly. QUANTA now lexes and parses QUANTA, in QUANTA. We are roughly one or two language features (`use` plus closures, both landed) from the complete fixpoint.



Diverse double-compiling: build the compiler with gcc, build it again with clang, compare the bytes. Across 53 of 53 modules the output is identical — the agreement is the proof there is no backdoor.

03 Four backends, one gate

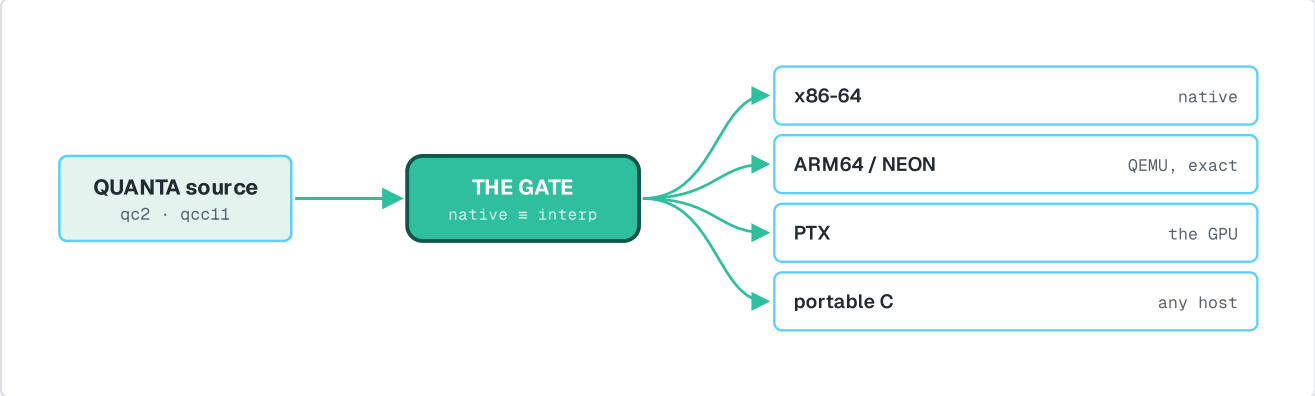
🟢 SHIPPED — MEASURED, GATE-CLEAN

The same source compiles gate-clean to **four backends**: x86-64, ARM64/NEON (run under QEMU, exact), PTX for the GPU, and portable C. 🟢 Every one is held to the same rule — the native result must equal the interpreter's — so "runs on any chip" is not a marketing line, it is a passing test on four targets.

Two capabilities make this more than a toy compiler:

Cost-as-type. Accuracy and cost are carried *in the type system*. A cost-model picks the strategy, which is how the register allocator went from 25.8 to 9.6 cycles per call — **4.2× faster and 3× smaller at once** 🟢 — and how a matmul reaches 0.8 TFLOP/s with no BLAS library underneath.

Real cryptography against published vectors. SHA-2/3, BLAKE3, ChaCha20, HMAC (RFC 4231) and AES-128 (FIPS-197) all validate against their official test vectors. 🟢 These are the tests you cannot bluff.



One source, one gate, four backends. Every target is held to the same rule — the native result must equal the interpreter's — so 'runs on any chip' is a passing test, not a slogan.

25.8	4.2× faster, 3× smaller	9.6
cycles / call		cycles / call

Cost-as-type in action: the register allocator, picked by a cost model carried in the type system.

04 Languages in, languages out

🟢 SHIPPED — MEASURED, GATE-CLEAN

QUANTA imports and exports, gate-verified in both directions. `c2q` proves a C function equivalent to its QUANTA translation per-function (`sum` , `dot` , `count_pos` all verified equivalent); `py2q` and a Rust path do the same coming in. Going out, QUANTA emits portable C plus the three native backends — a four-target Rosetta stone for code.

The honest edge on import

Coverage is a real subset — loops, arrays, floats, conditionals — not arbitrary-program-complete. Translating a whole third-party library is future work, not a shipped claim. We show the boundary because the boundary is where the next result comes from.

05 What is still open

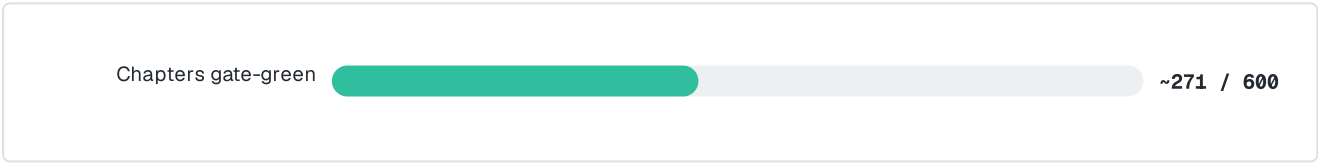
🔬 RESEARCH — MEASURED, PARTIAL

Kept visible, because the tiering is the credibility:

Strings, IO and FFI still go through C. 🔭 The runtime membrane is not closed. QUANTA computes and proves; it does not yet own its own edges.

The memory model is undecided. 🔭 This is a design decision we have deliberately not rushed — get it wrong and every later proof inherits the mistake.

~271 of 600 chapters are gate-green. 🔬 Real oracle asserts, stubs rejected; roughly 321 remain, including 256-bit elliptic curves and the zk/STARK tier. The gate harness itself is now memory-flat (1.5 GB → 3.4 MB), which is the efficiency thesis embodied in the tooling.



The frontier, owned: roughly 271 of 600 chapters are gate-green (real oracle asserts, stubs rejected) — about 321 remain, including 256-bit curves and the zk/STARK tier.

06 Why it ladders back

A language whose output you can verify byte-for-byte, on any chip, is how correct software gets cheap enough for everyone. That is the abundance argument, made concrete: not "trust us", but "compile it yourself, twice, and check."

What is still open — kept visible

The honest edges of this lane, shown next to the wins. None of these is a footnote; each is a claim we have not yet earned — and marking it is what earns the right to be believed on the rest.

- [ ] Strings, IO and FFI still route through C — the runtime edge is not closed.
- [ ] The memory model is undecided; it is a deliberate open design question, not a solved one.
- [ ] c2q / py2q import covers a real subset (loops, arrays, floats, conditionals) — not arbitrary-program-complete.
- [ ] ~271 of 600 chapters are gate-green; roughly 321 remain.

07 Sources, glossary & the ledger

Where every claim comes from — and what the plain words mean.

Sources

01 **Diverse double-compiling as a defence against trusting-trust attacks** — D. A. Wheeler (the DDC method QUANTA's self-host test is built on) (as of 2009)

02 **Vocabotics Dashboard — Organ 1 (QUANTA / The Forge), measured 2026-07-02** — Vocabotics internal record (as of July 2026)

Vocabotics

THE HONESTY LEDGER

🟢 Shipped — measured, gate-clean.

🔬 Research — measured, partial; the gap is shown.

🔭 Vision — designed, honest; not yet built.

The bigger claim goes up a tier, never down. We show the eval beside the demo — the tiering is the credibility.

See how the proof becomes the product

QUANTA is only half the story. The gate — native ≡ interpreter on every compile — is why the agreement can be trusted. Read the moat.

/research/the-gate

Nothing faked.

Measured, not asserted. Every claim carries its own honesty tier and the open edges are shown next to the wins — the tiering is the credibility.

Vocabotics · calm, clear, candid · no client names · global rail & metro heritage