INSTITUTE FOR PHYSICAL AI @ BMI
The Charlot Lab

Technical Report TR-2026-23
Engineering Report · Preprint v2
23 July 2026 · rev. 24 July 2026

ONE DIGEST

One Digest: Bit-Identical Neural Computation Across Heterogeneous Compute Fabrics

Making a transformer forward pass compute the same bits in a browser tab, on Apple silicon, and on an NVIDIA GPU — the mechanisms that break it, the pins that hold it, and the measured price.

David Jean Charlot, PhD

Dean of Physical AI · The Charlot Lab, Institute for Physical AI @ BMI

Correspondence: contact@physicalai-bmi.org · physicalai-bmi.org
Reproduction kit: github.com/dcharlot-physicalai-bmi/ferric · scripts/fabric-ci.sh · applied in Ferralloy (ferralloy.physicalai-bmi.org)

Abstract. The same shader source, run on two GPUs, does not compute the same bits — and on neural workloads the differences compound into different behavior. We report an engineering campaign that made a complete transformer kernel set (matmul, RMSNorm, LayerNorm, softmax, RoPE, causal attention, and a full multi-layer forward pass) produce *bit-identical* output digests on six substrates: Apple Metal, NVIDIA Vulkan, and Chrome's WebGPU — including each one's subgroup (warp) hardware — plus ARM, x86, and WebAssembly CPUs. We identify the divergence mechanisms layer by layer (fast-math shader compilation, fma contraction at three independent levels, register promotion of private and workgroup memory, host libm variance, vendor transcendentals) and the corresponding pins, of which only device-visible storage memory is universally reliable against compilers we do not control. For accelerators behind opaque compilers (Apple's Neural Engine via CoreML) we measure and formalize a second-tier contract: per-family *oracle digests*, shown viable by run- and process-stable measurements in which the ANE demonstrably engaged. Determinism's measured cost is +15–53% per kernel against the pre-campaign fast path on an idle machine. A later result (§3, addendum) shows the toolchain patches are in fact redundant: the same bit-identity holds on *stock, unpatched* wgpu — cross-fabric determinism is a property of the shader source alone, and the deterministic engine ships on the public crate registry unmodified. All results are dated, reproducible with one command from a public repository, and free to use; the failures and one retracted prediction are reported alongside the successes.

1. Why bits, not tolerances

Numerical work usually accepts small cross-platform differences and validates within a tolerance. Three uses make that insufficient for us. **Verification:** if a robot policy's behavior is to be checked

cryptographically — sign the expected outputs, let the device re-run and byte-compare[7] — then "close" is not checkable; only equality is. **Education:** a student's browser experiment should be exactly reproducible on the instructor's machine, or the discrepancy itself becomes an ungraded mystery. **Heterogeneous scheduling:** when every substrate computes the same bits, a scheduler may place work on whatever is free — CPU cores, a GPU, a browser tab — without changing any result. Bit-identity converts placement from a correctness question into a pure performance question, which is precisely what makes dynamic heterogeneous compute trustworthy. It is also, we found, indifferent to load: our cross-substrate gate passed unchanged while the same GPU ran a heavy concurrent decoding workload.

2. Where the bits diverge

We probed with a fixed instrument: deterministic integer-generated inputs (xorshift with mantissa bitcast — no library math anywhere), each kernel's raw output bits hashed with FNV-1a, and, as ground truth, plain-IEEE CPU replicas of every operation sequence in Rust, whose `f32` arithmetic is round-to-nearest with no fast-math by construction. Divergences localized to five mechanisms:

(i) Fast-math shader compilation. Metal compiles shaders with fast math unless told otherwise; wgpu never told it otherwise. A barriered Newton–Raphson sequence deviated from the IEEE replica on 183 of 768 inputs (1 ulp) on Metal and on 0 of 768 on NVIDIA Vulkan — the forensic that first separated compiler behavior from hardware behavior. The pin: `MTLMathModeSafe`.

(ii) fma contraction, three independent levels. SPIR-V permits drivers to fuse `a*b+c` unless each operation carries a `NoContraction` decoration; NVIDIA fuses. Metal contracts even under safe math — floating-point contraction is a separate, default-on setting (`FP_CONTRACT`). And a browser's WGSL compiler (Tint) contracts on its own schedule. Two compilers that both fuse can agree by coincidence; we observed exactly that, and observed the agreement dissolve the moment one side was pinned.

(iii) Register promotion. Source-level barriers (bit-casting through an opaque runtime value) hold only until the compiler promotes the surrounding memory into registers and re-applies (i) and (ii). We measured, in one browser compiler: private arrays promoted whenever indices were provably thread-private; barrier-free workgroup memory promoted in straight-line code but not in dynamic-trip loops; the same function fused in one inlining context and not in another. The behavior is real, reproducible — and unreliable, which for a guarantee is the same as absent.

(iv) Host library math. `sinf` differs by ulps between macOS and glibc; model weights generated through host libm made "the same model" differ across machines before a single shader ran.

(v) Vendor transcendentals. GPU built-in `exp/sin/cos/sqrt` — and division — have implementation-defined precision and differ across vendors regardless of every setting above.

3. The pins, in order of strength

Each mechanism has a countermeasure; the campaign's central finding is their strict ordering.

LEVEL	PIN	HOLDS AGAINST	FAILS AGAINST

toolchain	`MTLMathModeSafe · SPIR-V NoContraction · FP_CONTRACT OFF`	compilers we build (native)	compilers we do not (browsers)
source: algorithm	deterministic transcendentals from correctly-rounded ops only; integer-generated constants; fixed reduction shapes	vendor libm/intrinsics everywhere	—
source: registers	`bitcast-XOR with runtime-opaque value`	some contexts	promotion + re-fusion (iii)
source: workgroup	per-thread slots, phase-entangled (writer ≠ reader across each barrier)	dynamic-trip loops	straight-line promotion
source: storage	**device-visible buffers, unique slot per step**	**every compiler measured**	**nothing observed**

Table 1. The pin hierarchy. Storage memory is host-visible, so no conforming compiler may elide, reorder, fuse, or promote across it — the one pin that held unconditionally, including in the browser.

> **Addendum, 24 July 2026 — the toolchain tier is redundant.** The compiler pins (row 1) were built first and mattered before the storage discipline was complete; we later tested whether they are still necessary. Rebuilding the entire probe against **stock, unpatched wgpu 30 and naga 30** — no `MTLMathModeSafe`, no SPIR-V `NoContraction`, no MSL `FP_CONTRACT OFF` — the core kernel set is *still* bit-identical on Metal and Vulkan, matching the patched digests exactly (matmul `2a5bf373…`, the full transformer `1935293a…`, 9/9 rows). Since the browser never had the patches and already matched, all three fabrics are deterministic on an unpatched toolchain. The corrected, stronger claim: **cross-fabric bit-identity is a property of the shader source alone** — it requires no fork of the GPU toolchain. (The lone patch-dependent artifact is the subgroup-butterfly probe kernel, which stock naga does not yet compile; it is a demonstration, not part of the core determinism.) The practical consequence is that the deterministic engine now builds and publishes against the public registry unmodified.

Parallel reductions fit the same discipline with fixed shapes: 64 strided partial classes and a six-level add tree through barrier-separated phases, with each phase's reader assigned a *different* thread's stride class than its writer — a rotation that defeats privatization while provably preserving values (the tree pairing commutes with class rotation). Warp-level reduction joins the contract by replacing the driver's `subgroupAdd` (whose order is its own) with five explicit `subgroupShuffleXor` pairwise adds and a canonical lane broadcast, the subgroup width and lane mapping declared as part of the algorithm's shape.

4. Results: six substrates, one digest

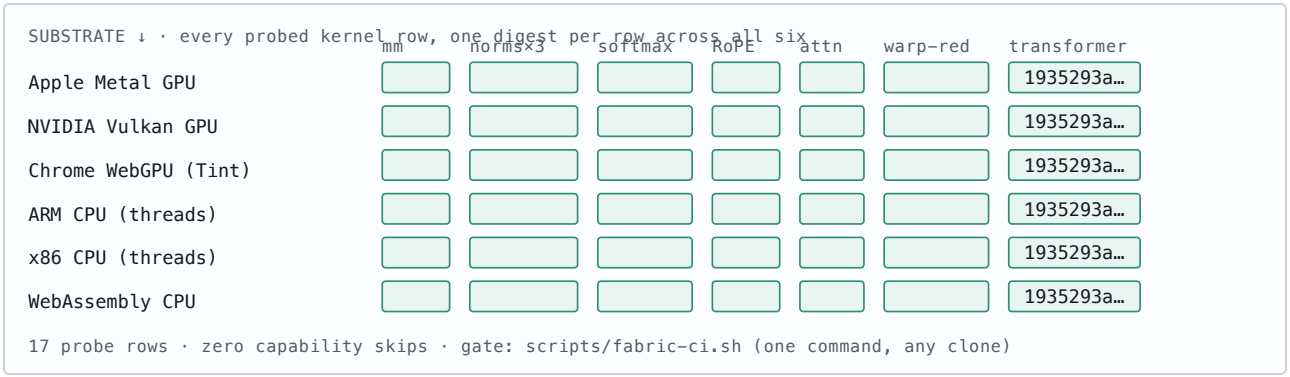

Figure 1. The substrate matrix. Every probed kernel row — shown here by class, with the full-transformer digest spelled out — is bit-identical on all six substrates. The three warp-hardware implementations (Apple simdgroups, NVIDIA warps, Chrome subgroups) and three CPU lane simulations agree on the butterfly's digest as well.

PROBE ROW	DIGEST (IDENTICAL ON ALL SIX SUBSTRATES)	NOTES
matmul	`2a5bf3731421828e`	barriered MAC loop
RMSNorm / LayerNorm / softmax	`7448dc28… · 6fc6af46… · 0025cd63…`	storage-chain scalar tails
sqrt	`c7d7cc0539d1f015`	0/768 vs plain-IEEE CPU on *all* fabrics
RoPE · causal attention	`62dcba6d… · 6c6beb0f…`	deterministic sin/cos/exp
parallel tree norms ×3	`ddeeb58c… · 11430c38… · f72aee8e…`	64-thread/row + CPU twins
subgroup butterfly	`38e0cb5c61db3037`	warps on all three GPUs + 3 lane sims
3-layer transformer forward	**`1935293abd55f5e5`**	attention, softmax, RoPE, norms, SwiGLU

Table 2. Selected rows of the 17-row probe (23 July 2026). A policy trained in a browser carries a behavioral signature that any Metal or Vulkan device can verify bit-exactly — the digest, not a tolerance, is the contract.

5. Opaque accelerators: the family-oracle contract

Neural engines sit behind compilers that are not ours to pin, so the one-digest contract cannot reach them; pretending otherwise would be the assumption this program exists to eliminate. The honest second tier: **measure** what a device family produces for a fixed (model, precision, compute-units, inputs) tuple, and pin *that* as the family's oracle. We established the contract's necessary conditions on Apple's Neural Engine (M5 Max, CoreML):

PROGRAM	UNITS	DIGEST	LATENCY	STABILITY
fp16	`CPU_ONLY`	`323687e4…`	0.69 ms	run + process stable
fp16	`CPU_AND_GPU`	`cb412e4f…`	0.66 ms	run + process stable
fp16	**`CPU_AND_NE`**	**`8a7ca6f8…`**	**0.30 ms**	run + process stable
fp32	`CPU_AND_NE`	`= CPU digest`	= CPU time	ANE silently not engaged

Table 3. The ANE grid (4×(matmul 1024^2 + relu) on 256×1024). Engagement is twice-witnessed — 2.3× speedup *and* a distinct digest. Three deterministic execution families coexist on one chip; each can carry an oracle.

The grid also surfaced the traps a naïve integration ships: fp32 programs silently never reach the ANE (identical digest and time to CPU — one would believe oneself NPU-verified and be wrong); small models never leave the CPU at all; and the `ALL` scheduler setting picks different units for different programs. Oracle contracts therefore pin explicit compute units and precision, and version their oracle store by (family, OS, compiler) until cross-device and cross-update stability are measured — which requires hardware we do not yet have, and is stated as such.

6. The measured price

KERNEL (512-CLASS SHAPES)	FAST BASELINE	BEST DETERMINISTIC	COST
matmul 512^3	0.41 ms	0.47 ms	+15%
softmax 512×4096	0.62 ms	0.72 ms (sequential)	+16%
LayerNorm 512×4096	0.62 ms	0.77 ms (tree)	+24%
causal attention 256·8·64	1.39 ms	1.78 ms	+28%
RMSNorm 512×4096	0.47 ms	0.72 ms (sequential)	+53%

Table 4. Idle-machine, batched-dispatch, minimum of five runs per generation (Metal, M5 Max). Full cross-substrate determinism costs +15–53% per kernel against the pre-campaign fast path.

Two methodological corrections stand in the record. Early per-op measurements included an 8 MB readback and overstated the cost by up to 6× — those numbers measured the bus. And our prediction that the parallel tree kernels would beat the fast baseline was wrong: on an idle GPU at these shapes, one-thread-per-row already saturates the machine, sequential deterministic kernels edge the trees for two of three norms, and only LayerNorm's tree wins decisively. Tree-versus-sequential is shape-dependent; both remain available under the same digest contract, so the choice is purely a performance one.

7. What this enables, and what it does not

The two-tier contract is now load-bearing in an open toolchain: signed policy packs whose eval vectors are recorded on one substrate and verified bit-exactly on another before an update goes live[7]; classroom experiments whose browser results match the reference machine to the bit; and free placement of deterministic kernels across any pinned substrate. It does not give cross-family identity for opaque accelerators (by design), does not yet cover tensor-unit matrix paths (their accumulation order is not pinnable today; they stay off the contract until it is), and depends on compilers we do not control continuing to respect storage semantics — which conformance requires, and which the one-command gate re-verifies on every run rather than assumes. Everything here — kernels, probes, forensic method, and this report — is published under MIT/Apache for anyone to use, check, teach from, or improve, and the deterministic engine and the fleet tools that consume it (`cargo install ferralloy`) are on the public crate registry, built on stock wgpu with no toolchain fork.

References

1. W3C. *WebGPU Shading Language* (WGSL), W3C Working Draft, 2026 — floating-point accuracy and subgroup built-ins.
2. Khronos Group. *SPIR-V Specification* — the NoContraction decoration and permitted operation contraction.
3. Apple. *Metal Shading Language Specification* — math modes and floating-point contraction semantics.
4. D. Goldberg. "What Every Computer Scientist Should Know About Floating-Point Arithmetic." *ACM Computing Surveys* 23(1), 1991.
5. IEEE. *IEEE 754-2019 Standard for Floating-Point Arithmetic*.
6. gfx-rs contributors. *wgpu* and *naga* — the portable GPU stack this work builds on, maintained here as patchable forks.
7. Institute for Physical AI @ BMI. *Ferralloy: signed, verified-behavior packs for physical-AI fleets.* ferralloy.physicalai-bmi.org, 2026.
8. Apple. *Core ML Tools* (coremltools 9) — MIL programs and compute-unit selection.
9. Institute for Physical AI @ BMI. *Cross-fabric determinism: campaign log and forensic method.* github.com/dcharlot-physicalai-bmi/ferric, docs/determinism/, 2026.

Reproduction: `git clone github.com/dcharlot-physicalai-bmi/ferric && scripts/fabric-ci.sh`
NPU grid: `experiments/npu-oracle/oracle.py --big` · price table: `examples/det_perf.rs`

Preparation: drafted with AI assistance and reviewed by the author. All measurements are real, dated, and reproducible from the public repository; failures and one retracted prediction are reported in §6 and in the campaign log.