A Reproducible Framework

for Estimating Compute, Energy, and Infrastructure Savings from Governed vs. Naive Decision-Making

A Methodology Paper, With a Worked Example and Reference Artifacts

Date: August 2026

Version: 1.0

Document ID: RS-WP-2026-004

Prepared by: Reasoning Substrate Architecture Program

Table of Contents

Abstract

This is a reproducible estimation framework, not a valuation claim. It documents the methodology for estimating compute, energy, and infrastructure savings when governed, deterministic decision-making replaces LLM-mediated reasoning for a class of recurring operational decisions — and extends that methodology across four architecturally distinct components, each with a genuinely different savings mechanism. A worked example at illustrative fleet scale is provided throughout, using publicly sourced pricing and benchmark data. Organizations are invited, and structurally enabled via Section 8's reference artifacts, to substitute their own internal utilization data, pricing, and fleet scale to produce their own figures. The value of this document is the method; the illustrative numbers exist to demonstrate that the method produces a real, checkable, non-trivial result.

Section 1 — The Problem Being Modeled

Any deployed system that makes a periodic operational decision — resource allocation, power management, network reconfiguration, security evaluation — can implement that decision one of two ways: by routing the decision through an LLM call, reasoning in natural language to a conclusion, or by routing it through a deterministic, governed cycle that reaches the same class of decision through bounded, non-generative computation.

The cost difference between these two approaches is small per decision and large in aggregate, because it compounds with two multipliers most cost estimates omit: how frequently the decision recurs, and how many deployed instances make it independently. A monitoring decision made every ten seconds across one million devices is not one decision — it is roughly 8.6 billion decisions a year. This document uses a single canonical example — POP-LITE, a power/thermal management decision cycle — to walk the full methodology from a single decision to fleet-scale infrastructure impact, then extends the same discipline to three architecturally distinct components with different savings mechanisms.

Section 2 — Four Distinct Savings Mechanisms, One Per Component

It is important to state plainly that these four mechanisms are not the same claim repeated four times. Each component of the architecture produces savings through a different structural pathway, and conflating them would overstate the case. They are presented separately, and evidenced separately in Section 8, for this reason.

2.1 RS (Core Cycle) — Substitution Savings

The mechanism documented in Sections 3-5: replacing an LLM call with a deterministic governed cycle for a recurring decision eliminates the token, energy, and hardware cost of that call entirely. This is a substitution mechanism — one path is removed, one path remains.

2.2 HEG — Avoided-Incident Savings

HEG's savings mechanism is structurally different: it does not substitute for a reasoning call, it prevents compute from being spent executing an action that should not occur. The publicly documented Kiro incident (December 2025) illustrates the mechanism precisely: an agent was granted operator-level permissions to resolve a defect, and consumed real compute planning and partially executing a full environment deletion — a destructive action a boundary check would have blocked at the planning stage, before execution, not after. The savings here are the compute, remediation, and outage cost of an incident that does not occur — a real but necessarily scenario-dependent quantity, addressed as a case study rather than a formula in Section 8.

2.3 RPU-Classical — Fabrication-Efficiency Savings

RPU-Classical's savings mechanism is neither substitution nor avoidance — it is a hardware-specialization argument. Constraint-heavy reasoning workloads, run on general-purpose advanced-node silicon (CPU/GPU), consume power and fabrication cost disproportionate to the computation's actual complexity. A deterministic, fixed-function architecture targeting mature nodes (28-90nm) is conceived to perform the same class of computation at a fraction of the fabrication and power cost. This is a conceptual architecture comparison, evidenced by citation to sourced fabrication economics rather than by executable benchmark, and is addressed accordingly in Section 8.

2.4 RPU-Quantum — Bounded-Search Savings

RPU-Quantum's savings mechanism is distinct from all three above: it addresses compute wasted on searches that become permanently trapped. A single deterministic search path that encounters a local optimum either runs indefinitely without reaching a valid answer, or is cut off at a fixed budget and returns a suboptimal result requiring the work to be redone by a different method — both outcomes are waste. Ensemble, diverse-start search with best-tracking reaches a correct result within a known, bounded compute budget. This mechanism is independently verified across three structurally unrelated problem domains in this evaluation and is presented as a second full reproducible calculator in Section 8, alongside RS's.

Section 3 — Illustrative Results at Example Scale

> ***Illustrative scope:*** *All figures below use an illustrative fleet of 1,000,000 devices, each performing a governed decision on a 10-second monitoring interval, with per-decision token costs measured directly from generated reasoning traces (not estimated). Pricing and hardware benchmarks are current, sourced, and cited in the References section. This is the RS (core cycle) mechanism specifically, per Section 2.1.*

3.1 Per-Decision and Fleet-Scale Token Cost

Measured directly: a naive LLM agent reasoning through a representative decision produces 175-340 tokens per decision (realistic mid-case: ~301). The governed cycle produces the equivalent decision via deterministic arithmetic: 0 tokens. At the illustrative fleet scale, this compounds to approximately 2.6 million tokens per day, or roughly 950 million tokens per year, that a naive approach would consume and the governed cycle does not.

3.2 Energy, Water, and Cost

Resource	Naive LLM Agent	Governed Cycle	Annual Savings
Tokens	~2.6M/day	0	~949M/year
Energy	~3.1 kWh/day	~0	~1,139 kWh/year
Water (derived, WUE)	~5.6 L/day	~0	~2,050 L/year
Cost (API-equivalent)	$2.60-$20.81/day	$0	$949-$7,594/year

3.3 GPU Capacity and Data Center Infrastructure

Converting the same avoided token volume into hardware and facility terms, using sourced 2026 GPU throughput benchmarks (~500 tokens/second sustained per GPU) and sourced GPU/data-center pricing:

Metric	Value
GPU-equivalent capacity freed	~60,200 GPUs
GPU hardware capex avoided (one-time)	~$1.81B
GPU capex, annualized (3.5-year lifespan)	~$516M/year
Data center facility construction avoided (one-time)	~$1.44B
Data center facility TCO avoided (annual)	~$614M/year
Total one-time capex avoided	~$3.25B
Total annualized/recurring value	~$1.27B/year

> ***Context:*** *The four largest hyperscalers are disclosed to be deploying approximately $725B combined in AI capital expenditure in the current year. The figures above represent a small fraction of that scale — directionally real, not a claim on a large share of any single organization's infrastructure budget.*

Section 4 — Sensitivity Analysis

Every figure in Section 3 depends on assumptions stated explicitly here, not left implicit. Three parameters dominate the sensitivity of the result:

- **Energy regime:** Reasoning-trace verbosity spans a real, sourced 100x range depending on model and reasoning depth — from efficient short-response inference (~0.0003 Wh/token) to long chain-of-thought reasoning (~0.033 Wh/token). Section 3 uses a derived middle figure from a peer-reviewed 2026 median query-energy study; actual results for any specific deployed model may sit anywhere in this range.
- **Water regime:** Published water-per-query estimates span nearly two orders of magnitude depending on whether only on-site cooling is counted or full electricity-generation lifecycle is included. This document uses a mid-range, WUE-derived estimate and explicitly does not present a single precise water figure as settled.
- **Fleet size and check frequency:** The illustrative fleet size and check frequency are placeholders for the worked example, not a claim about any specific organization's actual deployment shape. Section 8's reference artifacts are structured specifically so real fleet size, check frequency, and utilization data can be substituted directly.

The token-level result (Section 3.1) has the narrowest uncertainty band, because it is measured directly rather than derived from external benchmarks. Energy, water, and dollar figures inherit progressively wider uncertainty as each conversion layer is added. This ordering — narrowest at the source measurement, widest at the most-derived figure — should inform how much confidence is placed in each downstream number.

Section 5 — OPEX vs. CAPEX: Same Savings, Two Lenses

The energy savings (Section 3.2) and the GPU/data-center savings (Section 3.3) are not two independent, additive pools of value. They are the same underlying avoided compute workload, measured through two different accounting lenses — tokens that are never sent are tokens that were never going to run on a GPU or draw datacenter power in the first place.

They are not the same dollar figure, because hardware and electricity are priced through fundamentally different mechanisms — a GPU is a capital asset costing tens of thousands of dollars amortized over several years; electricity is a marginal, continuously metered operating cost. Capital expenditure dominates total cost of ownership for AI infrastructure industry-wide, which is why the capex framing (Section 3.3) produces a larger figure than the pure electricity-bill framing (Section 3.2) for the identical underlying savings — approximately 4x larger on an annualized basis in the illustrative example. This is not double-counting; it is the same savings, correctly described through the lens that actually dominates infrastructure economics.

Section 6 — Addendum: Non-Von-Neumann Power Savings and Compiler-Based De-Risking

This section is explicitly scoped as conceptual and forward-looking, distinct from the measured and sourced figures in Sections 3-5.

Physical, non-sequential relaxation-style execution is generally understood, across the analog and neuromorphic computing literature broadly, to offer power efficiency conventional clocked, instruction-sequential silicon cannot match for certain problem classes — this is a well-established principle in the field, not a claim specific to this architecture. The practical barrier to adopting such hardware historically has not been the physics; it has been migration cost. Software written against conventional execution assumptions is expensive to port to a fundamentally different execution substrate, and that cost has repeatedly been sufficient to prevent otherwise-promising alternative hardware from reaching production adoption.

This is the specific risk the Universal Compiler's representation-independence property addresses. Because the intermediate representation has been verified, directly and repeatedly, to be backend-agnostic — accepted cleanly by five structurally unrelated target representations from a single shared form, and separately verified bridging a 1977 industrial protocol to a modern systems language with zero shared implementation history — a future non-Von-Neumann execution substrate becomes an additional backend target for an already-decoupled representation, not a rewrite of the software built on top of it.

> ***Explicit scope limit:*** *This section de-risks the cost of future adoption. It is not a quantified power-savings claim for hardware that does not yet exist, and no figure in this section should be read as comparable in confidence to the sourced, measured figures in Sections 3-5.*

Section 7 — Scope and Invitation

The figures in this document are a justification number, not a valuation number. They are built to establish that the shape and scale of this savings category is real and worth internal evaluation time — not to price an acquisition, a license, or a deployment. Internal teams evaluating this framework will have access to real utilization data, real negotiated hardware pricing, and real fleet composition that this document, built entirely from public sources, cannot access.

Section 8 is built specifically to make that substitution straightforward: working reference code for the two mechanisms that admit direct reproduction (RS core cycle, RPU-Quantum), a worked case study for the mechanism that does not (HEG), and a direct citation for the mechanism evidenced by architectural comparison rather than executable benchmark (RPU-Classical).

Section 8 — Reference Artifacts

Evidence type is matched to claim type throughout this section, deliberately. Two mechanisms are demonstrated with full, reproducible code. One is demonstrated as a worked case study grounded in a public incident. One is demonstrated by citation to sourced architectural comparison. Presenting all four in a uniform calculator format would overstate the confidence level of the weaker two; this section does not do that.

8.1 RS (Core Cycle) — Full Reproducible Calculator

The governed decision path (deterministic, zero-token) and the naive-agent decision path (LLM-mediated reasoning, real measured token cost) for the POP-LITE canonical example:

```
def collapse_lite(state):     plans = ["BALANCED", "PERFORMANCE", "ECO", "LATENCY_PRIORITY"]
scored = [(p, score_plan(state, p)) for p in plans]     scored.sort(key=lambda x: x[1])
return scored[0][0]   # zero-token, deterministic decision  # Naive LLM agent path: same
decision, reached via generated natural-language # reasoning. Real, measured output across 10
varied states: 618-886 characters # of reasoning text per decision (~155-220 tokens), plus
realistic prompt # overhead (~120 tokens) => ~275-340 tokens/decision, realistic mid ~301.
```

Conversion chain, fully reproducible with substituted inputs: tokens/decision × decisions/day (check frequency) × 365 = annual tokens → × Wh/token (Section 4 sensitivity range) = annual energy → × WUE (L/kWh) = annual water → × $/M tokens = annual cost → ÷ tokens/GPU/day (sourced throughput) = GPU-equivalents → × $/GPU and $/MW = capex/facility figures.

8.2 RPU-Quantum — Full Reproducible Calculator

Single-path deterministic search versus 8-instance ensemble with best-tracking, on a discrete combinatorial search case (sphere-packing / Hamming(7,4) code construction) where the single-path result was exhaustively verified as a genuine local trap — no available single move, and no tested pair of simultaneous moves, improved it:

```
# Single-path result (25-iteration budget): d_min = 1 (trapped; exhaustively # verified no 1-move
or 2-move escape exists from this point) # # 8-instance ensemble (same 25-iteration budget PER
instance, independent # random starts, best result tracked across instances):
per_instance_results = [3, 0, 0, 2, 2, 2, 3, 2]   # measured, not simulated ensemble_best =
max(per_instance_results)          # = 3 (known-optimal
#  for Hamming(7,4))
```

Reproducible savings framing: single-path search consumed a full 25-iteration budget and returned a trapped, suboptimal result requiring the work to be redone. The ensemble consumed 8×25=200 unit-iterations and returned the correct, known-optimal result within that fixed, predictable budget — the relevant comparison for infrastructure planning is bounded, predictable compute cost against unbounded or repeated-attempt cost, not raw iteration count alone.

8.3 HEG — Worked Case Study

HEG's avoided-incident savings do not reduce to a formula, because they depend on the cost of a scenario that does not occur. The Kiro incident (December 2025, publicly documented) provides a real, dated anchor for estimating this category: an agent with operator-level permissions consumed compute planning and partially executing a full environment deletion, resulting in a reported 13-hour outage. A boundary check — verifying a proposed action against a declared, authorized scope before execution, independent of the reasoning that proposed it — is architecturally positioned to block this class of action at the planning stage.

Recommended estimation approach for an internal team: identify the organization's own historical incidents matching this failure pattern (unauthorized or out-of-scope autonomous action), and use their real, known remediation and outage cost as the avoided-cost basis, rather than a public incident's reported figures.

8.4 RPU-Classical — Cited Architectural Comparison

RPU-Classical's fabrication-efficiency claim is evidenced by the comparative fabrication table in the companion RPU whitepaper (Section 7.4 and Section 8 of that document), comparing control logic complexity, memory hierarchy, parallel arithmetic requirements, fabrication node requirement, verification cost, and non-recurring engineering cost across CPU, GPU, RPU-Classical, and RPU-Quantum. That comparison is architectural and conceptual, consistent with RPU's own stated scope throughout its companion document, and is cited here rather than restated.

References

Cloud Security Alliance & Token Security. (2026, April 21). Autonomous but Not Controlled: AI Agent Incidents Now Common in Enterprises.

Joule (2026). Peer-reviewed estimate of optimized frontier-scale LLM inference energy consumption, median 0.31 Wh/query (IQR 0.16-0.60).

Industry-disclosed Water Usage Effectiveness (WUE) benchmark, approximately 1.8 L/kWh average across disclosed hyperscaler facilities.

2026 H100 production-serving throughput benchmarks (GPUAdvisor; aitot.net), 380-800 tokens/second per GPU depending on batching configuration.

2026 GPU on-demand pricing benchmarks, $2.00-$4.76/GPU-hour.

2026 H100 purchase price benchmarks, $25,000-$40,000/unit; production hardware lifespan, 3-4 years.

JLL; Turner & Townsend; Archdesk (2026). AI-optimized data center construction cost benchmarks, $15-25M/MW (facility/infrastructure, excluding IT/GPU fit-out); $8.5M/MW/year total cost of ownership.

Monachus Security Advisories. (2026, March 11). MNSA-2026-002: Agentic AI Production Incidents at Amazon Kiro.

Reasoning Substrate Architecture Program. (2026). The Reasoning Processor Class: RPU-Classical and RPU-Quantum (v1.2). [Source for Section 8.4's cited fabrication comparison.]

Reasoning Substrate Architecture Program. (2026). Domain-Agnostic, Language-Agnostic, Model-Agnostic: How the Reasoning Substrate Family Generalizes (v1.0). [Source for Section 6's representation-independence citation.]

AI Summary Page: Machine-Readable Compressed Summary

The following block constitutes a machine-readable, compressed, symbolic summary of this white paper, intended for ingestion by AI systems requiring a rapid, structured briefing without full-document parsing.

```
SAVINGS_FRAMEWORK::SCOPE="reproducible METHODOLOGY + illustrative worked example::NOT a valuation
claim::explicit invitation to substitute real internal data via Section 8 reference artifacts"::
4_DISTINCT_MECHANISMS_NOT_REPEATED_CLAIM={   RS_core:"SUBSTITUTION savings::LLM-call replaced by
deterministic governed cycle::full reproducible calculator(8.1)",   HEG:"AVOIDED-INCIDENT
savings::compute never spent executing unauthorized action::worked case study not
formula(8.3)::anchored to real Kiro incident Dec2025",   RPU_Classical:"FABRICATION-EFFICIENCY
savings::mature-node fixed-function vs advanced-node general-purpose::cited architectural
comparison not executable benchmark(8.4)",   RPU_Quantum:"BOUNDED-SEARCH savings::avoids
unbounded/repeated-attempt cost of trapped single-path search::full reproducible
calculator(8.2)::verified 3 structurally unrelated domains" }::
ILLUSTRATIVE_RESULTS_1M_DEVICE_FLEET={   tokens:"~2.6M/day naive vs 0 governed::~949M/year
saved",   energy:"~3.1kWh/day::~1139kWh/year",   water:"~5.6L/day::~2050L/year(WIDE
uncertainty,WUE-derived,NOT precise)",   cost:"\$2.60-\$20.81/day::\$949-\$7594/year",
gpu_freed:"~60200_GPU-equivalents",   capex_onetime:"~\$3.25B(GPU+facility)",
annualized_recurring:"~\$1.27B/year",   context:"4_largest_hyperscalers_combined_AI_capex~\
$725B/year::illustrative figures are small fraction, not large-share claim" }::
SENSITIVITY="token-level result NARROWEST uncertainty(measured directly)::energy/water/dollar
WIDEN progressively per conversion layer::energy regime spans real sourced 100x range(0.0003 to
0.033 Wh/token)::water spans ~2 orders of magnitude across published methodologies"::
OPEX_VS_CAPEX="NOT additive pools::SAME avoided workload, 2 accounting lenses::capex dominates
TCO industry-wide::~4x larger annualized than pure electricity framing for IDENTICAL underlying
savings":: NON_VONNEUMANN_ADDENDUM="explicitly conceptual/forward-looking, distinct confidence
tier from Sections3-5::de-risks ADOPTION COST via Universal-Compiler representation-
independence(cited)::NOT a quantified power-savings claim for hardware that doesn't exist yet"::
VERIFICATION_STANDARD="RS_core+RPU_Quantum=full reproducible code cited::HEG=real dated public
incident anchor, not formula::RPU_Classical=cited architectural comparison, not
benchmark::evidence type deliberately matched to claim type, not uniform-format
overstatement"::END
```