The Reasoning Processor Class

RPU-Classical and RPU-Quantum

A New Middleware Compute Primitive for Enterprise AI Infrastructure

Conceptual Architecture White Paper | Version 2.1 | November 2026

Matthew Thornton | Inventor and applicant | icpub.org | matt@icpub.org

Publication information

Title	The Reasoning Processor Class: RPU-Classical and RPU-Quantum
Version	2.1, November 2026. Supersedes version 2.0 and earlier.
Type	Conceptual architecture white paper. Not peer reviewed. No hardware has been built or measured; every quantitative result in this paper is mathematical or numerical.
Patent record	US 2026/0330264 A1, "Hardware Processor for Deterministic Contractive State Evolution Using a Fixed Point Update Operator," U.S. Appl. No. 19/707,753, filed June 15, 2026, published October 22, 2026; continuation of U.S. Appl. No. 19/543,866. Full citation details: Addendum A. The cited publication governs its own claims; this paper does not modify them.
Interactive card	Addendum B: a plain-text card that can be loaded into an AI assistant to ask questions about the architecture.
Cite as	Thornton, M. (2026). *The Reasoning Processor Class: RPU-Classical and RPU-Quantum.* Version 2.1, November 2026. icpub.org.
Rights	© 2026 Matthew Thornton.

Version history

2.1	November 2026	Stated the norm used throughout (Euclidean, with the maximum norm in the mesh result) and gave the maximum-norm form of the fixed-point rule; corrected the tile figure and the card's unit list to the units the patent publication names; added claim numbers to the card's mesh line.
2.0	November 2026	Restated around the published operator $\Theta(s) = f(As+b)$. Added the convergence and tolerance results, the exact reductions, the mesh convergence check, the fixed-point design rule, an evidence ledger, the interactive card and a patent citation addendum. Unit name written as Reasoning Processing Unit. The earlier machine-readable summary page is replaced by the card.
1.3	November 2026	Aligned with the published hardware patent; stated the medium-agnostic principle.
1.0–1.2	2026	Earlier conceptual drafts.

1 Executive Summary

The Reasoning Processing Unit (RPU) is a processor class built around one native operation: iterating a deterministic, bounded update operator to a fixed point. The operator is stated in the hardware patent publication US 2026/0330264 A1 as

$$\Theta(s) = f(As + b), \tag{1}$$

with A a contractive matrix and f a bounded nonlinearity or projection, optionally extended by an energy-guided correction,

$$\Theta(s) = f\big(As + b - \eta \nabla E(s)\big). \tag{2}$$

A convergence unit halts the iteration when the change between successive states falls below a threshold, or when an iteration cap is reached.

This paper restates the architecture around that operator and adds analysis that does not depend on any particular physical medium:

- **One operator, a family of methods.** With specific choices of A, f and E, the operator is exactly a recurrent layer, projected gradient descent, ISTA, or a Hopfield-type fixed-point layer (Section 2.2).
- **A convergence guarantee with an explicit condition.** When $k = \mathrm{Lip}(f)\,(\|A\| + \eta L) < 1$, the iteration has a unique fixed point, converges geometrically, and halts with error at most $\varepsilon/(1-k)$ (Section 2.3).
- **Feasibility by construction.** Hard constraints placed in the projection f hold at every iterate (Section 2.4).
- **Medium tolerance.** Any realization whose per-step error is at most δ stays within $\delta/(1-k)$ of the ideal fixed point, so an imperfect medium changes the result by a bounded amount (Section 4.2).
- **Mesh.** Arrays of RPU tiles reach the same fixed point under synchronous and asynchronous updates when the combined map is a contraction (Section 3.2).

Two variants are described. **RPU-Classical** is deterministic contractive iteration. **RPU-Quantum** is an ensemble of independent classical paths with ensemble-level selection and convergence; it uses no qubits.

RPUs are not physical processors today. They are a mathematically defined architecture intended to guide silicon development. They are middleware: they complement CPUs, GPUs and TPUs by taking on constraint-heavy fixed-point workloads, and they are not general-purpose processors.

2 The Operator

2.1 Definition

Execution starts from an initial state s_0 and repeats $s_{t+1} = \Theta(s_t)$. It halts at the first t for which

$$\|s_{t+1} - s_t\| < \varepsilon \tag{3}$$

or when t reaches the iteration cap $T_{\max}$. The two forms of Θ above correspond to claims 3 and 4 of the patent publication (Addendum A). Gradient-based formulations of this kind are standard [1].

s	state vector (a plan, schedule, routing configuration or allocation)
A, b	contractive matrix and bias vector
f	bounded nonlinearity or projection
E, ∇E	energy function encoding constraint violations and inefficiencies, and its gradient
η	step size of the energy-guided correction
ε, $T_{\max}$	convergence threshold and iteration cap
k, L	contraction factor, and Lipschitz constant of ∇E
$\|\cdot\|$	Euclidean norm on states throughout, and the induced (spectral) norm $\|A\|$ for matrices; k, L and ε refer to the same norm. Section 3.2 states a result in the maximum norm instead.

The processor is deterministic: for fixed A, b, f, E, η, ε, $T_{\max}$ and initial state, the sequence of states is fixed. This paper specifies no numerical value for η, k, ε, fixed-point width or multiply-accumulate size, and none is claimed. Sufficient conditions below are stated as mathematics.

2.2 One Operator, a Family of Methods

The operator contains established iterative methods as exact special cases (Table 1). Trajectories of Θ and of the named method were compared in double precision; the maximum difference is at the level of machine precision. The identities show that problems an RPU can execute include these established methods under

Table 1: Exact reductions of the RPU update (numerical check, double precision, $n = 6$).

Setting	Resulting method	Max difference
$E(s) = \frac{1}{2}s^\top Qs + q^\top s$, $f = \tanh$	Recurrent layer $\tanh(Ws+c)$ with $W = A - \eta Q$, $c = b - \eta q$	2.2×10^{-16}
$A = I$, $b = 0$, $f =$ clip to a box	Projected gradient descent	5.6×10^{-17}
$A = I$, $b = 0$, $f =$ soft-threshold	ISTA for ℓ_1-regularised least squares [12]	2.2×10^{-16}
$\eta = 0$, $f = \tanh$	Hopfield-type and deep-equilibrium fixed point $s = \tanh(As+b)$ [8][13]	residual 5.6×10^{-17}, against an independent Newton solve
$A = I$, $b = 0$, $f =$ identity	Gradient descent (contractive only under further conditions on E and η)	by definition

one halting rule and one hardware contract. They do not make those methods specific to the RPU, and they do not by themselves establish contraction; that condition is checked per instance (Section 2.3).

2.3 Convergence Guarantee

Theorem 1 (Convergence)**.** *Let f be L_f-Lipschitz and ∇E be L-Lipschitz, and let $k = L_f(\|A\| + \eta L)$. If $k < 1$, then Θ is a k-contraction and: (i) it has a unique fixed point s^*; (ii) $\|s_n - s^*\| \le \dfrac{k^n}{1-k}\|s_1 - s_0\|$; (iii) if the iteration halts at the first n with $\|s_{n+1} - s_n\| < \varepsilon$, then $\|s_{n+1} - s^*\| \le \dfrac{k}{1-k}\varepsilon < \dfrac{\varepsilon}{1-k}$; (iv) the halting test is satisfied for every $n > \ln(\varepsilon/\|s_1 - s_0\|)/\ln k$.*

The proof is in Addendum C; it is the contraction mapping principle [11]. The condition is sufficient, and the bounds are worst-case. Table 2 compares the halting bound with measured halting steps on random contractive instances, and Figure 4 shows the same data with the fixed-point precision result of Section 4.4.

For operators outside the condition, including the broader operator forms recited in claims 20 and 21 of the

Table 2: Halting step against the worst-case bound, $\varepsilon = 10^{-3}$, 200 random instances per row ($n = 16$, $f = \tanh$). The error bound of Theorem 1(iii) held in 800 of 800 instances.

k	Bound on halting step (median)	Measured (median)	Measured (max)	Worst measured/bound
0.50	11	5	6	0.55
0.78	30	7	10	0.34
0.90	71	8	11	0.16
0.95	145	8	14	0.10

publication, convergence is not guaranteed. The iteration cap bounds execution, and an implementation can distinguish the two outcomes by comparing the final step size with ε.

2.4 Constraints

Lemma 1 (Feasibility)**.** *If f maps into a closed set C, then every state s_t with $t \geq 1$ lies in C, and so does the fixed point s^*.*

Hard constraints expressed as membership in C, such as a box, a simplex or a capacity set, are therefore satisfied at every iterate after the first update, and by the result. Constraints encoded as penalties in E are minimized, not guaranteed. A design chooses which constraints belong in f and which in E.

2.5 Expressing Problems

A problem is executable on an RPU when it can be expressed through (A, b, f, E): any problem expressible as a differentiable or relaxable energy, with its hard constraints expressed as projections. Logistics, scheduling, routing, allocation and configuration problems are constraint-satisfaction and optimisation problems [3]. Constructing the energy is the problem-specific work; the processor then executes the iteration natively. An RPU does not execute arbitrary programs; it solves.

3 Architecture

3.1 The Tile

The patent publication discloses a processor built from state registers, a matrix-multiply unit, a bias unit, a bounded nonlinearity, a convergence unit and a control unit; in digital embodiments the matrix-multiply unit is a fixed-point multiply-accumulate array, the convergence unit comprises a delta unit, a norm unit and a comparator, and the control unit is a finite-state machine with an iteration cap. An optional energy-guided correction unit implements the $-\eta\nabla E$ term. In mesh embodiments a tile also exchanges state with its neighbours (Section 3.2). Figure 1 shows these units as one tile. One iteration is: (1) read the state registers; (2) multiply by A; (3) add b; (4) optionally subtract $\eta\nabla E$; (5) apply f; (6) compute the step $s_{t+1} - s_t$, its norm, and compare with ε; (7) halt, or repeat until the cap. The cycle is fixed-function with shallow control logic and minimal local memory.

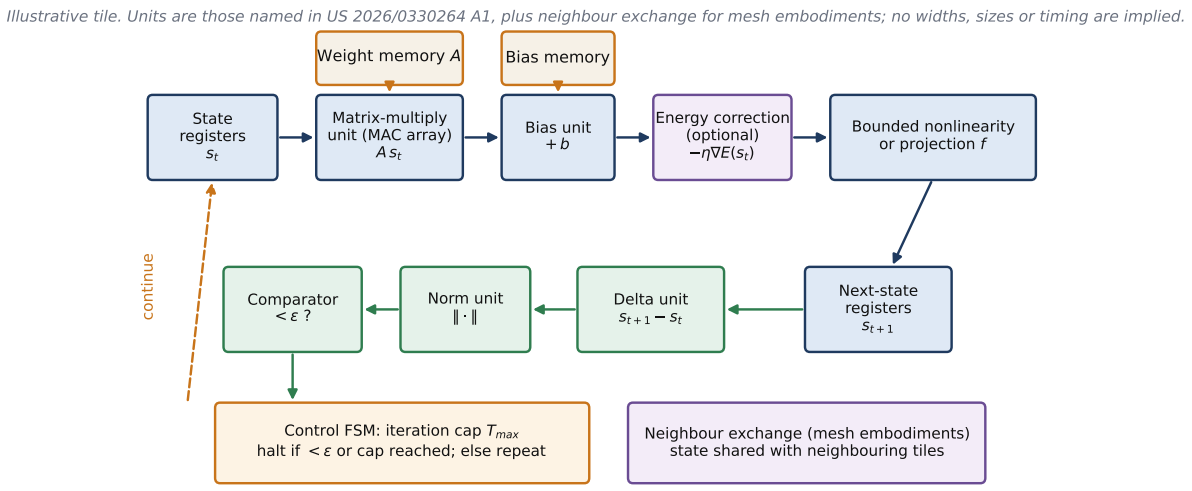


Figure 1: Illustrative RPU tile, drawn from the units named in the patent publication, with a neighbour-exchange port for mesh embodiments (claims 6 and 7). No widths, sizes or timing are implied.

3.2 Mesh

Several tiles can be arranged as a mesh or distributed array, each acting as a convergent node and exchanging state with its neighbours (claims 6 and 7). If the combined update is a contraction in the maximum norm, asynchronous iteration with bounded staleness converges to the same fixed point as synchronous iteration [14]. A numerical check used a 4×4 mesh (16 tiles, four state variables each, neighbour coupling, combined contraction factor at most 0.8). Synchronous updates came within 10^{-6} of the fixed point in 7 rounds. Asynchronous updates in random order, with stale neighbour reads of up to 0, 2 and 5 steps, converged in all 15 runs to the same fixed point, in 15.4, 16.4 and 16.8 sweeps (node updates divided by 16), about 2.2 to 2.4 times the synchronous count (Figure 2). No global clock is required.

3.3 Position in the Compute Stack

RPUs are envisioned as middleware accelerators between application logic and general-purpose hardware, following the layered compute-stack model [5]. They receive structured problem states, iterate to a fixed point, and return improved configurations, integrating alongside existing compute primitives without displacing them: CPUs for control and orchestration, GPUs for tensor compute and training, TPUs for neural-network inference, and future quantum systems for deep combinatorial search.

4 Medium Independence

4.1 The Operator Is Defined Without Reference to a Medium

Contraction is a mathematical property of the update, not of the material. The operator settles to a fixed point because it is contractive, not because it is made of silicon. The patent publication states the operator across media: silicon, magnetic, optical, photonic, polymer, printed, atomic, nuclear and quantum-inspired substrates (claim 5), with analog, optical or photonic linear-transformation elements (claim 17), magnetic, atomic or nuclear-inspired transformation elements (claim 18), and optical, photonic, magnetic, spintronic, atomic or nuclear-inspired realizations of the broader operator form (claims 22 to 24). References in the publication to

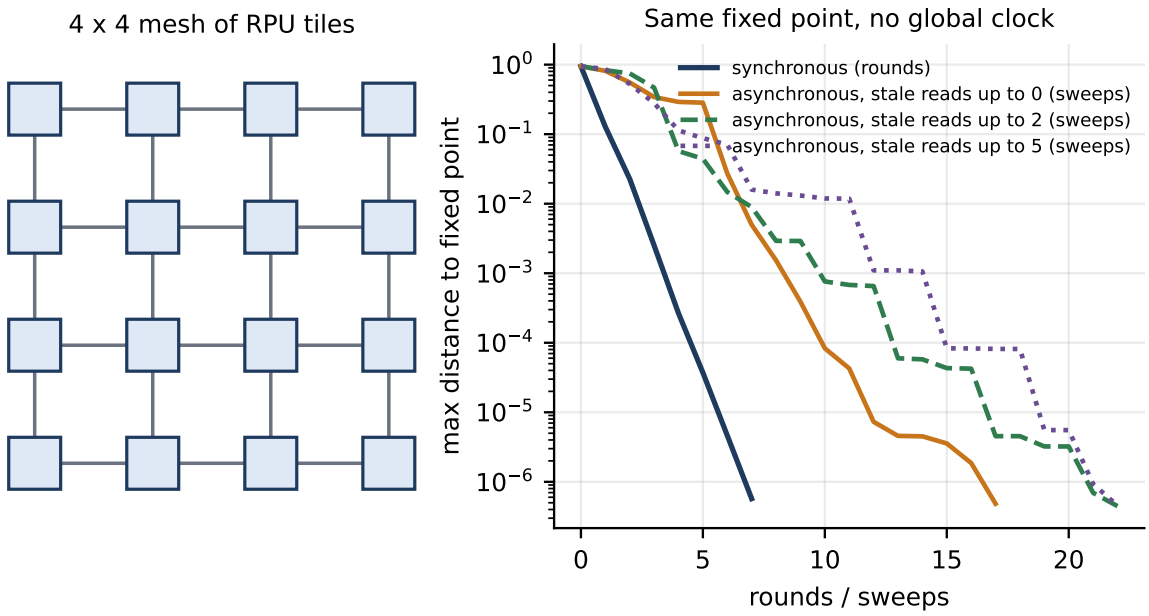


Figure 2: Mesh of RPU tiles, and convergence of synchronous and asynchronous updates to the same fixed point (numerical check).

quantum-inspired embodiments denote classical implementations of energy-guided or gradient-corrected update rules; they do not require quantum computing hardware. Each medium realizes the same operator; the medium changes how the update is physically realized, not what is computed. The target instantiation of this paper is standard silicon (Section 9); specific implementations in each medium are future engineering work.

4.2 Tolerance of an Imperfect Medium

Physical media are imperfect: analog gains drift, optical paths lose power, registers have finite width. For a contractive operator the effect of such errors is bounded.

Theorem 2 (Tolerance)**.** *Let* Θ *be a* k*-contraction with fixed point* s^**, and let a realization* $\tilde{\Theta}$ *satisfy* $\|\tilde{\Theta}(s) - \Theta(s)\| \le \delta$ *on the states it visits. Then for the realized iterates* $\tilde{s}_n$*,*

$$\|\tilde{s}_n - s^*\| \le k^n \|\tilde{s}_0 - s^*\| + \frac{\delta}{1-k}(1-k^n), \qquad \textit{so} \qquad \limsup_{n\to\infty} \|\tilde{s}_n - s^*\| \le \frac{\delta}{1-k}.$$

Errors do not accumulate; they are forgotten at rate k. In a general instruction stream, a single erroneous operation can change a result without bound; in a contractive iteration, a bounded error has a bounded effect. Table 3 and Figure 3 show the bound against measured steady-state distance ($k = 0.8$, $n = 8$) for adversarial per-step errors, random bounded noise, state quantization and an analog-style gain error. The bound held in every case, with the measured distance between 5% and 31% of the bound.

Table 3: Tolerance bound against measured steady-state distance (numerical check).

Error model	δ	Bound $\delta/(1-k)$	Measured	Measured / bound
Adversarial worst-case	0.001	0.005	0.001562	0.31
Adversarial worst-case	0.01	0.05	0.01562	0.31
Adversarial worst-case	0.05	0.25	0.07822	0.31
Random bounded noise	0.001	0.005	0.001162	0.23
Random bounded noise	0.01	0.05	0.01103	0.22
Random bounded noise	0.05	0.25	0.05938	0.24
State quantised to 4 bits	0.08839	0.4419	0.05626	0.13
State quantised to 8 bits	0.005524	0.02762	0.003481	0.13
State quantised to 12 bits	0.0003453	0.001726	0.0002777	0.16
Analog gain error, $\pm 1\%$ on A	0.01697	0.08485	0.00438	0.05

4.3 What Each Medium Promises

Realization	Determinism	Guarantee
Digital fixed-point	Exact; bit-reproducible for fixed inputs	Theorem 1 holds for the quantized map; halting per the design rule below
Mixed-signal and analog	Reproducible within a declared per-step error δ	Convergence to within $\delta/(1-k)$ of the ideal fixed point (Theorem 2)
Optical and photonic	Reproducible within a declared δ	As above
Magnetic, atomic and other media	Reproducible within a declared δ	As above, with δ characterized for the medium

Each realization declares its δ. The operator, the fixed point and the form of the guarantee are the same in every medium.

4.4 Fixed-Point Design Rule

A fixed-point datapath is a map on a finite set, so every orbit ends in a cycle. The convergence unit can trip only on a cycle whose step is below ε. A one-step oscillation of one least-significant bit in each of n state variables has norm up to $\sqrt{n}\,2^{-b}$ for b fractional bits. This gives the rule

$$\varepsilon \;\geq\; \sqrt{n}\,2^{-b} \tag{4}$$

for the Euclidean norm. In the maximum norm the same oscillation has size 2^{-b}, and the rule is $\varepsilon \geq 2^{-b}$. Table 4 shows the effect ($\varepsilon = 10^{-3}$, $n = 16$, $k = 0.8$, cap 400 iterations, 100 random instances per row): at 6 and 8 fractional bits, about half of the runs ended in a one-step oscillation and the convergence unit never tripped; at 12 bits and above it tripped in every run.

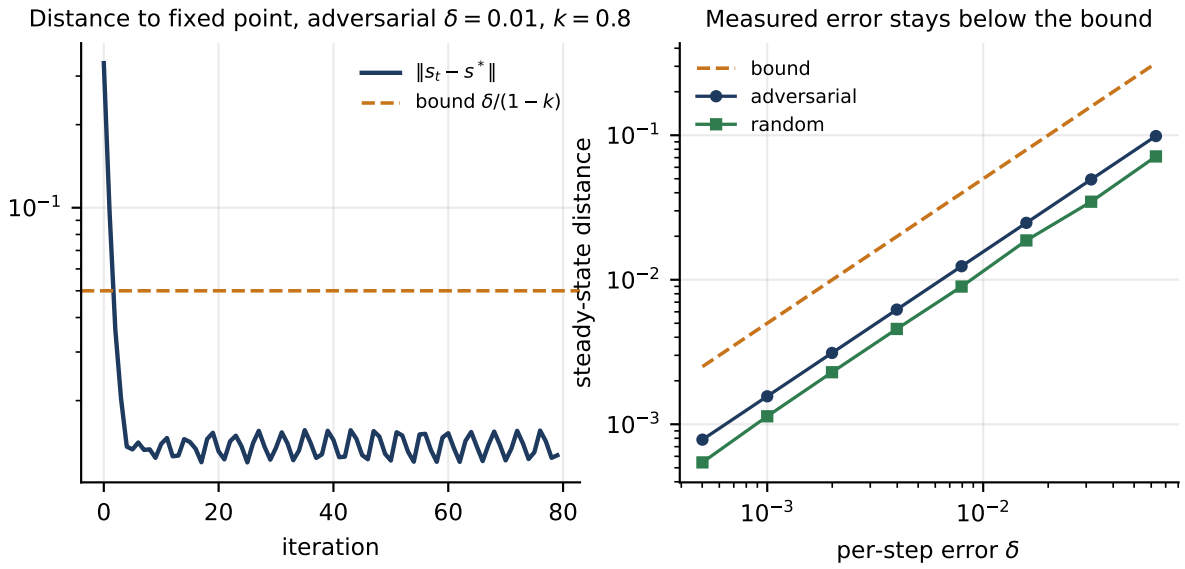


Figure 3: Left: distance to the fixed point under adversarial per-step error. Right: steady-state distance against per-step error, with the bound of Theorem 2.

Table 4: Fixed-point precision and the convergence unit (numerical check, bit-accurate state rounding).

Fractional bits b	LSB 2^{-b}	$\sqrt{n}\,2^{-b}$	Tripped	Never tripped	Median final error
6	2^{-6}	0.0625	49 of 100	51 of 100	0.018
8	2^{-8}	0.0156	44 of 100	56 of 100	0.0047
10	2^{-10}	0.0039	76 of 100	24 of 100	0.0012
12	2^{-12}	0.0010	100 of 100	0 of 100	0.00035
16	2^{-16}	0.0001	100 of 100	0 of 100	0.00019

5 RPU-Classical

5.1 Definition

RPU-Classical is the deterministic architecture of Section 2: it applies Θ directly, with the contraction condition providing stability and repeatability. With hard constraints in the projection f, solutions remain feasible throughout convergence (Lemma 1).

RPU-Classical is	RPU-Classical is not
A deterministic fixed-point engine with an optional energy-guided correction	A CPU or GPU replacement
A low-power reasoning coprocessor (design target)	A quantum computer
A middleware layer for structured decision problems	A general-purpose compute unit
A standard-silicon, room-temperature architecture target	A product available today

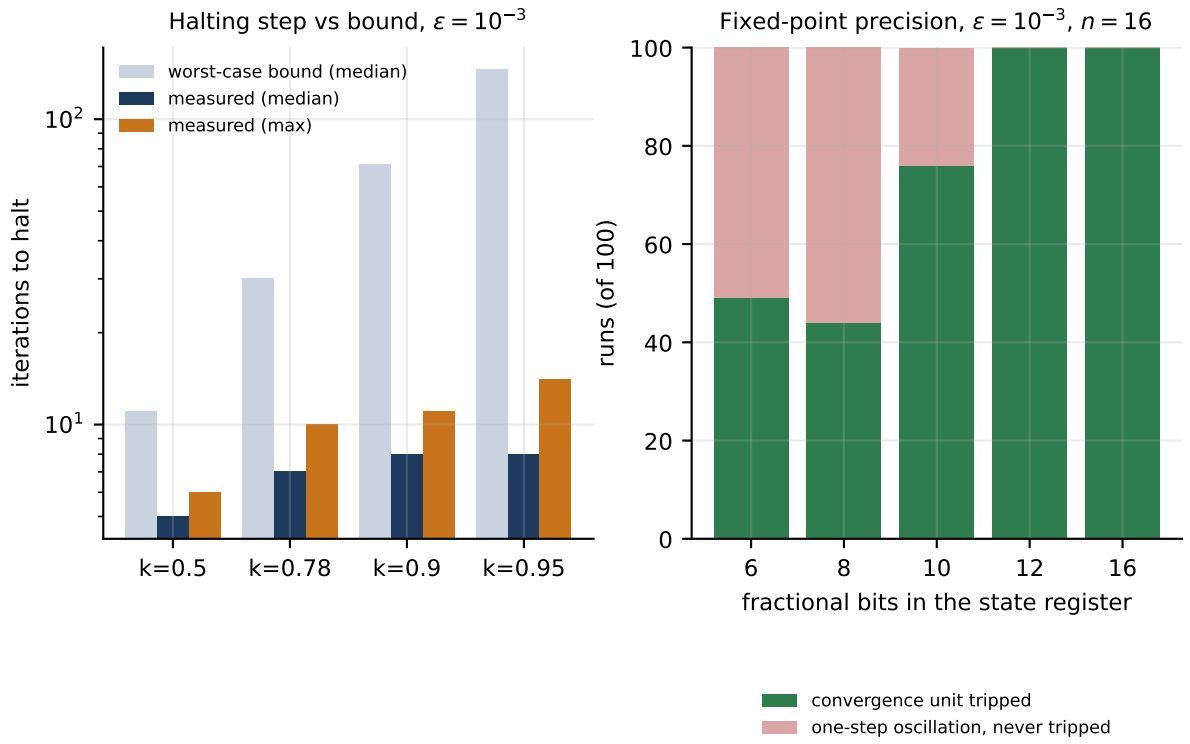


Figure 4: Left: measured halting step against the worst-case bound. Right: fixed-point precision and the convergence unit.

5.2 Target Domains

Logistics and routing: real-time route planning, last-mile optimisation and multi-vehicle coordination, with constraint violations encoded as energy. **Scheduling:** production scheduling, workforce allocation and resource sequencing. **Resource allocation:** cloud resource planning, bandwidth allocation and network load balancing. **Safety-critical reasoning:** automotive, aerospace and medical systems need decisions that respect hard constraints; hard constraints in f hold at every iterate, and validation against specific safety standards is required for any implementation. **Edge environments:** the standard-silicon target and projected low power make RPU-Classical a candidate for edge devices, IoT infrastructure and mobile platforms where cloud offload may not be viable.

5.3 Architectural Value Proposition

RPU-Classical is designed to reduce AI inference cost by offloading reasoning workloads from CPU and GPU clusters, to stabilise LLM reasoning pipelines by enforcing constraint consistency [7], and to extend edge AI within strict power envelopes [6]. These are design targets; none has been measured.

6 RPU-Quantum

6.1 Definition

RPU-Quantum extends RPU-Classical with parallel exploration. It runs m independent classical paths from distinct initial states, each applying Θ, keeps the lowest-energy state found, and applies the halting test to the ensemble as a whole, not path by path (Section 6.3). Its purpose is to escape shallow local minima that trap a single deterministic path, in the spirit of perturbation-based strategies in simulated annealing [2] and quantum annealing [4]. It uses no qubits and no cryogenic infrastructure. The name denotes the structure of parallel exploration with best-state selection; consistent with the patent publication, quantum-inspired embodiments are classical. An optional variant adds a structured perturbation, $s_{t+1} = \Theta(s_t) + \alpha\,\xi_t$, where ξ_t is a deterministic or seeded sequence so that runs remain reproducible; this variant is not evaluated in this paper.

RPU-Quantum is	RPU-Quantum is not
An ensemble of independent classical paths with ensemble-level selection	A qubit machine
A design for escaping shallow local minima on classical silicon	An implementation of Shor's algorithm
A room-temperature, standard-silicon architecture target	A quantum chemistry simulator
A candidate for the task class that quantum optimisation systems address	A general-purpose quantum computer

This is a design target for the same class of tasks that quantum optimisation systems address (ground-state finding, energy minimisation and configuration search), not a benchmark claim.

6.2 Target Domains

Complex planning; large-scale logistics (fleet management, supply chain, port and terminal scheduling); global routing (traffic engineering and backbone optimisation); high-dimensional configuration search (hyperparameters, hardware configuration, resource orchestration); national-security planning and asset allocation; telecommunications network reconfiguration after a channel-degrading event; and combinatorial structure generation such as error-correcting code construction, where a single search path can become provably trapped.

6.3 Evidence for the Exploration Principle

Evaluation-stage evidence comes from direct execution of small-scale, illustrative implementations. It supports the underlying mathematical principle, that diverse parallel search escapes shallow minima a single deterministic path cannot, without constituting a claim about the performance, efficiency or behavior of any silicon implementation. A single-path deterministic search was compared with an ensemble of independent searches across three structurally unrelated instances:

- **Continuous energy minimisation.** A single deterministic solver, tested against an asymmetric energy function with a defined local trap, converged to and remained at the local minimum regardless of iteration budget. An ensemble differing only in starting points, with best-result tracking, located the true global optimum.

- **Network reconfiguration.** A single-path search converged to a result identical to taking no corrective action. The same ensemble approach recovered a materially better configuration.
- **Discrete combinatorial search.** A single greedy local search became provably trapped; exhaustive verification confirmed that no single move, and no tested pair of simultaneous moves, could improve the result. An ensemble of independent searches, with best-result tracking across the ensemble, reached a known, externally verifiable optimum for the tested case.

One operational finding bears on any implementation: applying stopping criteria inside each individual path, rather than at the ensemble level, degraded results by terminating paths before they could escape a temporary plateau. Convergence detection at the ensemble level avoided the degradation. Underlying records are available to qualified reviewers under NDA [10].

6.4 Predictability of the Ensemble

For independent starts with single-start success probability p, an m-path ensemble succeeds with probability $1-(1-p)^m$, so ensemble size can be chosen from single-path statistics. A numerical check on a four-variable double-well energy (4000 trials, eight paths, gradient steps from uniform random starts) gave a single-start success rate of 0.074, a measured best-of-eight success rate of 0.443, and a predicted 0.461. The difference is within sampling error, since the prediction itself carries about ± 0.02 uncertainty from estimating p on the same trials. This result describes the ensemble as predictable and analyzable.

7 Evidence Ledger

Every statement in this paper has one of five statuses: **Patent** (stated in the cited publication); **Proved** (a theorem or lemma, Addendum C); **Checked** (a numerical check, with its setup stated in the text); **Illustrative** (small-scale implementation evidence, Section 6.3); **Target** (design intent, not measured).

Statement	Status	Section
The operator $\Theta = f(As+b)$, its energy-guided form, the convergence unit and cap, the mesh, and the listed media	Patent (publication; claims 1–24)	2–4
Unique fixed point, geometric convergence, halting error $< \varepsilon/(1-k)$ when $k < 1$	Proved; Checked (800 instances)	2.3
Hard constraints in f hold at every iterate	Proved	2.4
Exact reduction to a recurrent layer, projected gradient, ISTA and a Hopfield-type layer	Checked (machine precision)	2.2
Error of a realization stays within $\delta/(1-k)$ of the ideal fixed point	Proved; Checked (10 error models)	4.2
Fixed-point rule $\varepsilon \geq \sqrt{n}\,2^{-b}$	Checked (500 runs)	4.4
Synchronous and asynchronous mesh updates reach the same fixed point	Cited theory; Checked (15 runs)	3.2
An ensemble escapes traps that stop a single path	Illustrative (3 instances)	6.3
Ensemble success follows $1-(1-p)^m$	Checked (4,000 trials)	6.4
Hardware performance, power, area and fabrication cost	Target; not measured	8, 9

Statement	Status	Section
Capability on the task class of quantum optimisation	Target; no benchmark	6.1

8 Strategic Value and Deployment

All statements in Sections 8 and 9 describe design targets, not measured results.

8.1 A Compute Primitive for Constraint-Aware Reasoning

CPUs process, GPUs parallelise, TPUs accelerate inference. The RPU proposes a role focused on constraint-aware, fixed-point decision-making. Constraint-heavy reasoning workloads are among the fastest-expanding categories of enterprise AI demand, and they are handled today through software layers on hardware not designed for them.

8.2 Middleware Positioning

RPUs receive structured problem states from host systems, iterate to a fixed point and return improved configurations. The boundary is clean and the integration model is additive: an RPU augments an existing compute stack without disrupting it.

8.3 AI Infrastructure

Reasoning pipelines for large language models carry significant constraint-handling overhead in multi-step workflows [7]. A dedicated coprocessor is designed to take on that work, reducing latency and per-query cost and enforcing constraint consistency before results reach application layers.

8.4 Defence and Intelligence

Defence and intelligence environments require decisions under hard constraints, at speed, in air-gapped or forward-edge conditions. RPU-Classical's deterministic convergence is designed for real-time operational decisions, and RPU-Quantum's exploration for high-dimensional planning. Both target standard silicon, which suits environments where specialised hardware cannot be sustained.

8.5 Energy

The architecture targets high-value reasoning work at a fraction of the projected power of equivalent CPU and GPU workloads. Quantified projections await implementation-level validation.

8.6 Deployment Targets

Cloud	Reasoning coprocessor reached through standard APIs alongside existing compute tiers, with workloads routed to RPU-Classical or RPU-Quantum by problem complexity and optimisation depth.
On-premises and hybrid	Standard rack deployment on commodity silicon, with no power, cooling or networking requirements beyond existing hardware.
Edge and mobile	Local reasoning without cloud round-trips, reducing latency and preserving data sovereignty [6]; IoT, mobile platforms and autonomous systems.
Automotive and industrial IoT	On-device path planning, resource scheduling and fault management under tight power budgets; certification to any specific standard is future engineering work.
Defence and classified	Air-gapped and classified infrastructure; standard-silicon compatibility reduces supply-chain dependencies.
Space and extreme environments	Spacecraft, satellites and deep-space probes face limited power, minimal cooling, radiation and intermittent connectivity. Projected low power, deterministic convergence and a room-temperature silicon target make RPU-Classical a candidate for autonomous, constraint-respecting decisions where GPU-class thermal output is not available.

9 Fabrication and Reshoring Candidacy

All characterisations reflect conceptual design intent, not validated manufacturing data. The 28–90 nm mature node discussed here is the target instantiation for the standard-silicon RPU; it is not the only medium the operator can run on, and it is not a limit on the operator.

9.1 A Mature-Node Architecture Target

The RPU is designed around a fixed-function update cycle with shallow control logic and minimal local memory. Unlike CPUs, GPUs and TPUs, which need deep pipelines, speculative execution, multi-level caches and large parallel arithmetic arrays, an RPU would not need the transistor density or lithographic precision of advanced nodes. This is a deliberate architectural choice: the task, iteration of a fixed update rule, does not benefit materially from advanced-node performance headroom.

9.2 Historical Design Lineage

Early analog and mixed-signal VLSI built for energy-minimisation and constraint-solving tasks showed that such architectures can be realized without advanced lithography [8][9]. The RPU echoes that lineage, extending its mathematical principles into a formally defined architecture with an explicit convergence guarantee. This is a conceptual lineage; it does not imply direct hardware revival or continuous development between eras.

9.3 Relevance to U.S. Semiconductor Policy

The CHIPS Act identifies domestic mature-node fabrication capacity as a distinct and under-addressed priority, separate from sub-7 nm efforts. RPU architectures, if pursued toward silicon, are candidates for that category: their simplicity, low thermal envelope and modest die complexity make them plausible for cost-effective production in reshored legacy-node capacity, without competing for the advanced-node capacity CPUs and GPUs require.

9.4 Comparative Fabrication Complexity

RPU-Classical and RPU-Quantum share the same fundamental silicon profile and are treated as one class here, differentiated by the ensemble in RPU-Quantum.

Dimension	CPU	GPU/TPU	RPU-Classical	RPU-Quantum
Control logic	Very high	High	Low	Low–moderate
Memory hierarchy	Multi-level caches	Large SRAM/HBM	Small local SRAM	Small local SRAM
Parallel arithmetic	Moderate	Extremely high	Minimal	Minimal–moderate
Node requirement	Advanced (5–14 nm)	Advanced (5–14 nm)	Mature (28–90 nm)	Mature (28–90 nm)
Verification cost	Very high	Very high	Low	Low–moderate
NRE	High	Very high	Low–moderate	Moderate

10 Comparative Analysis

The table reflects intended design characteristics, not benchmarked performance. RPU columns are architectural targets.

Dimension	CPU	GPU	TPU	RPU-Classical	RPU-Quantum
Primary function	General compute	Parallel tensor	NN inference	Deterministic fixed-point iteration	Ensemble fixed-point iteration
Reasoning native	No	No	No	Yes (by design)	Yes (by design)
Constraint handling	Software only	Software only	Software only	Hardware-native (conceptual)	Hardware-native (conceptual)
Power profile	High	Very high	Moderate	Low (target)	Low–moderate (target)
Deployment	Universal	Data centre, cloud	Data centre, cloud	Cloud, edge, defence, space	Cloud, edge, defence
Fabrication node	5–14 nm	5–14 nm	5–14 nm	28–90 nm (target)	28–90 nm (target)

RPUs are not intended to compete with or replace any of these primitives. The relationship is additive: each class performs its purpose-built role, and the stack as a whole becomes more capable, particularly for constraint-heavy reasoning workloads that lack a hardware-native home.

11 Path to Evaluation

The results in this paper are mathematical and numerical. The path to hardware-level evidence is a sequence of steps, none of which has been completed:

1. a bit-accurate model of the tile and mesh;
2. a register-transfer-level implementation and simulation against that model, compared by signature;
3. a prototype on a programmable device;
4. benchmarks against conventional iterative solvers on shared tasks, measured under published rules;
5. characterisation in additional media, each with a declared per-step error δ.

A parametric cycle model is the natural starting point for sizing. With M multiply-accumulate units and n state variables, one iteration takes roughly $\lceil n^2/M \rceil + c$ cycles, where c is a fixed overhead, and the iteration count follows from Theorem 1(iv). No numerical sizing is claimed in this paper.

12 Conclusion

The Reasoning Processor Class is defined by a published operator. Its guarantees are stated as theorems and checked numerically: a unique fixed point with an explicit halting error, feasibility by construction for hard constraints, bounded effect of an imperfect medium, and convergence of meshes without a global clock. Its exploration variant is predictable from single-path statistics. The operator is stated across physical media, and the tolerance result says how an imperfect medium affects the answer.

This is an architectural concept. The mathematics is stated and checked, the design targets are defined, and the strategic case is real. The path from concept to silicon is the work ahead.

Strategic note. The case for a dedicated reasoning processor is strongest before the workload category becomes too large and too costly to address through software alone. Enterprise and government stakeholders who engage with this architectural direction early are positioned to shape both the technology and its adoption pathway.

References

[1] Bottou, L. (2010). "Large-scale machine learning with stochastic gradient descent." Proceedings of COMPSTAT 2010. Springer.

[2] Kirkpatrick, S., Gelatt, C.D., & Vecchi, M.P. (1983). "Optimization by simulated annealing." Science, 220(4598), 671–680.

[3] Dechter, R. (2003). Constraint Processing. Morgan Kaufmann Publishers.

[4] Kadowaki, T., & Nishimori, H. (1998). "Quantum annealing in the transverse Ising model." Physical Review E, 58(5), 5355.

[5] Hennessy, J.L., & Patterson, D.A. (2019). Computer Architecture: A Quantitative Approach (6th ed.). Morgan Kaufmann.

[6] Han, S., et al. (2016). "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding." ICLR 2016.

[7] Wei, J., et al. (2022). "Chain-of-thought prompting elicits reasoning in large language models." NeurIPS 2022.

[8] Hopfield, J.J. (1982). "Neural networks and physical systems with emergent collective computational abilities." Proceedings of the National Academy of Sciences, 79(8), 2554–2558.

[9] Hopfield, J.J., & Tank, D.W. (1985). "Neural computation of decisions in optimization problems." Biological Cybernetics, 52(3), 141–152.

[10] Reasoning Substrate Architecture Program. (2026). Internal small-scale verification records for the ensemble-exploration evidence cited in Section 6.3. Available to qualified reviewers under NDA.

[11] Banach, S. (1922). "Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales." Fundamenta Mathematicae, 3, 133–181.

[12] Daubechies, I., Defrise, M., & De Mol, C. (2004). "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint." Communications on Pure and Applied Mathematics, 57(11), 1413–1457.

[13] Bai, S., Kolter, J.Z., & Koltun, V. (2019). "Deep equilibrium models." NeurIPS 2019.

[14] Bertsekas, D.P., & Tsitsiklis, J.N. (1989). Parallel and Distributed Computation: Numerical Methods. Prentice-Hall.

The special cases in Section 2.2 correspond to refs. [8], [12] and [13].

Addendum A: Patent Publication Record

Publication	US 2026/0330264 A1
Title	Hardware Processor for Deterministic Contractive State Evolution Using a Fixed Point Update Operator
Application number	19/707,753
Filed	June 15, 2026
Published	October 22, 2026
Inventor and applicant	Matt Thornton, Houston, TX, United States
Relationship	Continuation of U.S. Appl. No. 19/543,866, filed February 19, 2026, "Systems and Methods for an Operator Separated Cognitive Architecture for AI Reasoning" (US 2026/0289346 A1, published September 24, 2026)
Classification	CPC G06N5/04
Record	https://www.patents-review.com/a/20260330264-hardware-processor-deterministic-contractive-state-fixed.html
Official status	https://patentcenter.uspto.gov/applications/19707753

The record link is a third-party mirror of the publication. Current status should be checked at USPTO Patent Center. The table below orients the reader; it does not characterize or limit the claims, and the published claims govern.

Element discussed in this paper	Published claims
Iterative application of a contractive operator Θ to a fixed point (Section 2.1)	1, 8, 12
$\Theta(s) = f(As+b)$ (Section 2.1)	3, 9, 13
Energy-guided correction $f(As+b-\eta\nabla E(s))$ (Section 2.1)	4, 10, 14
State registers, operator unit, convergence unit, control unit; norm of the difference vector (Sections 2.1, 3.1)	2, 19
Fixed-point multiply-accumulate array (Section 3.1)	16
Mesh or distributed array of processors (Section 3.2)	6, 7, 11, 15
Physical media (Section 4.1)	5, 17, 18, 22, 23, 24
Broader operator forms (Section 2.3)	20, 21

Addendum B: Interactive Card

The card below is plain ASCII. To use it, paste it into an AI assistant and ask questions about the architecture; the card carries the claim boundaries, so answers should stay within what this paper and the patent publication state. The text is also reproduced here so it can be copied from this document; line breaks fall between words and do not change the card. To check a copy, replace each line break with one space and compute SHA-256 over the result; the expected value is

31013f2004ebf9bae7f10a5b50f4710827238dad2cd6b616764d40c3c425df45

```
RPU PUNCHCARD v2.1 | sources: The Reasoning Processor Class v2.1 (Nov 2026); US
2026/0330264 A1 (App. 19/707,753, filed 2026-06-15, published 2026-10-22). WHAT: a
processor class built on one native operation: iterate a deterministic, bounded update
operator to a fixed point. Concept paper. No hardware built or measured. No performance
claim. NAME: RPU = Reasoning Processing Unit. OPERATOR (patent): Theta(s) = f(A s + b),
A contractive, f a bounded nonlinearity or projection. Energy-guided form (patent):
Theta(s) = f(A s + b - eta*grad E(s)). The patent claims also recite any deterministic
bounded contractive operator (claim 20) and any deterministic bounded operator iterated
to a convergence criterion (claim 21). NORM: ||.|| is the Euclidean norm and ||A|| the
induced (spectral) norm throughout; k, L and epsilon use the same norm; the mesh result
uses the max norm. CYCLE: apply Theta; delta = ||s_next - s||; halt when delta < epsilon
or the iteration cap is reached. UNITS (disclosed): state registers, weight memory, bias
memory, MAC array, bias unit, bounded nonlinearity, delta unit, norm unit, comparator,
control FSM, optional energy correction unit. GUARANTEE (paper analysis): if k =
Lip(f)*(||A|| + eta*L) < 1, L = Lipschitz constant of grad E, then a unique fixed point
s* exists, ||s_n - s*|| <= k^n/(1-k)*||s_1 - s_0||, and at halt ||s - s*|| <=
epsilon/(1-k). Outside this condition convergence is not guaranteed; the cap ends the
run. CONSTRAINTS: hard constraints placed in f (projection) hold at every iterate;
constraints placed in E are minimized, not guaranteed. SPECIAL CASES (exact, numerical
check): quadratic E gives a recurrent layer; clip gives projected gradient;
soft-threshold gives ISTA; tanh with eta=0 gives a Hopfield / deep-equilibrium fixed
point; A=I, b=0, f=identity gives gradient descent (not contractive in general). MEDIA:
the patent states the operator across silicon, magnetic, optical, photonic, polymer,
printed, atomic, nuclear and quantum-inspired (classical) substrates. TOLERANCE (paper
analysis, numerical check): if each realized update is within delta of the ideal update,
the state stays within delta/(1-k) of the same fixed point; errors fade at rate k.
DETERMINISM: exact in digital fixed-point; within the tolerance in other media. DESIGN
RULE: in fixed-point choose epsilon >= sqrt(n)*LSB (Euclidean norm; epsilon >= LSB in
the max norm) or the convergence unit may never trip. MESH (claims 6, 7, 11, 15): nodes
exchange state; under a contraction, synchronous and asynchronous updates reach the same
fixed point (numerical check). RPU-QUANTUM: an ensemble of independent classical paths
with ensemble-level selection and convergence; no qubits; for independent starts,
success of m paths follows 1-(1-p)^m. NOT: not a built processor, no benchmark, not a
CPU or GPU replacement, not a quantum computer. PRIORITY: the cited patent publication
governs its claims; the paper governs its own analysis.
```

A worked example for the card. Ask the assistant to apply Θ for three steps with $A = \mathrm{diag}(0.5, 0.3)$, $b = (1,1)$, $f =$ clip to $[-2,2]$ and $s_0 = (0,0)$. The expected states are $s_1 = (1,1)$, $s_2 = (1.5, 1.3)$ and $s_3 = (1.75, 1.39)$. In the Euclidean norm, $k = \|A\| = 0.5$ and $\|s_1 - s_0\| = \sqrt{2}$, so Theorem 1(ii) gives $\|s_3 - s^*\| \leq 0.3536$ against the true distance 0.2530 from $s^* = (2, 1.4286)$ (in the maximum norm the bound is 0.25).

Further questions to try. What is the RPU update operator? Is gradient descent the RPU operator? What does the convergence guarantee require? What happens in an imperfect medium? Does an RPU use qubits? Has an RPU been built or benchmarked?

Addendum C: Proofs

Theorem 1. For states s, s', let $u = As + b - \eta \nabla E(s)$ and u' likewise. Then $\|u - u'\| \le \|A\| \|s - s'\| + \eta \|\nabla E(s) - \nabla E(s')\| \le (\|A\| + \eta L)\|s - s'\|$, and $\|\Theta(s) - \Theta(s')\| = \|f(u) - f(u')\| \le L_f \|u - u'\| \le k\|s - s'\|$. With $k < 1$ on a complete normed space, the contraction mapping principle [11] gives (i). For (ii), $\|s_{j+1} - s_j\| \le k^j \|s_1 - s_0\|$, so $\|s_n - s^*\| \le \sum_{j \ge n} \|s_{j+1} - s_j\| \le \frac{k^n}{1-k} \|s_1 - s_0\|$. For (iii), $\|s_{n+1} - s^*\| \le \sum_{j \ge 1} \|s_{n+j+1} - s_{n+j}\| \le \frac{k}{1-k} \|s_{n+1} - s_n\| < \frac{k\varepsilon}{1-k}$. For (iv), $\|s_{n+1} - s_n\| \le k^n \|s_1 - s_0\| < \varepsilon$ whenever $n > \ln(\varepsilon / \|s_1 - s_0\|) / \ln k$.

Lemma 1. If $f(\mathbb{R}^n) \subseteq C$ then $s_{t+1} = f(\cdot) \in C$ for every $t \ge 0$. A fixed point satisfies $s^* = \Theta(s^*) \in C$, and the limit of points in a closed set lies in the set.

Theorem 2. $\|\tilde{s}_{n+1} - s^*\| \le \|\tilde{\Theta}(\tilde{s}_n) - \Theta(\tilde{s}_n)\| + \|\Theta(\tilde{s}_n) - \Theta(s^*)\| \le \delta + k\|\tilde{s}_n - s^*\|$. Induction on n gives $\|\tilde{s}_n - s^*\| \le k^n \|\tilde{s}_0 - s^*\| + \delta \sum_{j<n} k^j$, and the geometric sum is $(1 - k^n)/(1 - k)$.

Fixed-point rule. A one-step oscillation changes each of n coordinates by at most one least-significant bit 2^{-b}, so its step norm is at most $\sqrt{n}\,2^{-b}$. A convergence unit with threshold ε cannot trip on a cycle whose step norm is $\ge \varepsilon$; $\varepsilon \ge \sqrt{n}\,2^{-b}$ removes this obstruction for one-step oscillations of that size.