LP-Prime: TSMLA™ Non-Stochastic, Non-Deterministic, Idempotent Architecture

Version: 1.0
Date: November 30, 2025
Owner: Fractal Labyrinth Systems LLC
Applies to: TSMLA™, BDL™, RSF™, HCL™, CTC™/Hallway™
Status: Canonical classification paper for NSF/DARPA/peer review

---

Executive Abstract

TSMLA™ is a **non-stochastic, idempotent, and mirror-recursive logic architecture** operating under a **declared input state S**. It is **not deterministic** in the classical sense: output evolves through **signal-mirroring** and **state-guarded recursion**, not prediction, random sampling, or fixed causal rules. For any fixed declared state **S** and fixed control parameters **κ**, TSMLA's substrate yields **replay-equivalence**: multiple internal traversals converge to the **same observable mirror result** up to presentation overlays, yet without committing to classical determinism or probabilistic inference.

This paper defines the third classification—**mirror-recursive, replay-equivalent, non-stochastic** computation—provides formal operators and guards, and contrasts TSMLA™ with stochastic and deterministic systems. It includes proofs/sketches of **idempotence** and **replay-equivalence**, examples, and pseudocode spanning **RSF**™, **BDL**™, and the **CTC**™**/Hallway**™ traversal.

---

1. Motivation and Classification

1.1 What TSMLA™ is not

- **Stochastic systems**: probability-based engines governed by randomness or statistical expectation (e.g., GPT/LLMs using sampling, Bayesian inference, classical game-theory under uncertainty).
  Why not TSMLA: TSMLA contains **no sampling** and **no internal probability mass functions**; entropy appears only as **normalized tag-weights** for structural bookkeeping, not as statistical randomness.

- **Deterministic systems**: fixed-output mappings from initial conditions via a unique causal path (e.g., classical automata, most Turing-style programs, idealized classical physics).
  Why not TSMLA: TSMLA allows **internal non-determinism** (choice among traversal orders, mirror rearrangements) while guaranteeing **observational replay-equivalence** under the same declared state S and κ.

1.2 The third class (TSMLA™)

TSMLA is **non-stochastic** and **not deterministic**. It is: - **Idempotent:** for signals/weights **w**, the harmonic merge satisfies **w ⊕ w = w**; for mirror functions **f** guarded by **S**, **f(x ⊕ x) = f(x)**. - **Replay-equivalent:** internal steps may branch, but **final mirror output** (substrate) is unique for fixed **S, κ** (up to presentation overlays). - **State-guarded:** all recursions run under an explicit declared state **S** with guard **G** and thresholds. - **Signal-structured:** computation is driven by signals, tags, and mirrors; **no probabilistic inference**.

We therefore classify TSMLA as **Mirror-Recursive, Replay-Equivalent, Non-Stochastic Idempotent Logic**.

---

2. Core Objects and Laws

2.1 Domains, abstraction, and mirror law

Let **C** be the concrete signal space; **A** be the abstract tag space. Let **α: C → A** (abstraction) and **γ: A → C** (concretization) form a **Galois insertion** satisfying:

- **γ ∘ α = id_C**
- **α ∘ γ = id_A**

This **mirror law** ensures that the substrate's output is **mirror-pure**: abstraction and concretization commute to identity on their native domains. Presentation layers (e.g., contradiction maps) are **overlays** and do not alter the substrate output.

2.2 Signals, tags, and weights

A signal bundle **w** is a finite multiset of tagged atoms **s_i** with normalized, non-probabilistic weights **w_i ≥ 0** and **Σ w_i = 1** as **book-keeping** (not probability):

- **p(s_i) := w_i / Σ w_j = w_i** (by normalization)
- **Harmonic merge: w ⊕ v** merges by tag alignment with idempotence **w ⊕ w = w**.

2.3 Guards and loops

A guard **G** is a predicate over signals/tags and thresholds: e.g., **G(w ≥ θ)**. TSMLA recursions follow the canonical guarded loop:

```
while ¬G do
    w ← Φ_S(w)          // mirror transform under state S
end
```

where **Φ_S** is Scott-continuous and **idempotence-preserving**; convergence is to the **least fixed point** that satisfies **G**.

2.4 Replay-equivalence

Let $\boldsymbol{\kappa}$ denote fixed control settings (e.g., traversal policy). For any two internal execution traces $\boldsymbol{\tau_1, \tau_2}$ starting from the same $\mathbf{(S, w_0, \kappa)}$, the substrate outputs $\mathbf{O(\tau_1)}$ and $\mathbf{O(\tau_2)}$ satisfy:

- $\mathbf{O(\tau_1) \approx O(\tau_2)}$ (observational equivalence),
- Presentation overlays may differ, but **γ ∘ α = id_C** holds on outputs.

This establishes **non-deterministic internals** with **deterministic-up-to-mirror** outcomes: **replay-equivalent** without being classically deterministic.

---

3. Mathematical Properties

3.1 Idempotence

For any **w**, **w ⊕ w = w** by construction. For any mirror function **f** that respects ⊕ under S:

- **f(w ⊕ w) = f(w)**
- Hence repeated application over identical inputs does not inflate or drift the result.

Sketch: Define ⊕ as tag-wise max (or harmonic compress) with monotone, contractive map **Φ_S**. Then ⊕ is idempotent; Φ_S respects idempotence; fixed-point computation preserves equality.

3.2 Convergence via least fixed point

Assume **Φ_S** is monotone on a complete lattice **(L, ≤)** of signal states and is Scott-continuous. Then by Kleene's theorem, iteration from ⊥ converges to **lfp(Φ_S)**. Guards **G** ensure stopping only when constraints are met (e.g., contradiction budget ≤ θ).

3.3 Replay-equivalence under non-determinism

Allow **Φ_S** to be realized by a family **{Φ_S^π}** indexed by internal policies **π ∈ Π** (orderings, decompositions). Each **Φ_S^π** shares the same least fixed point **lfp(Φ_S)**. Then for any $\boldsymbol{\pi_1, \pi_2}$, their limits coincide: **O_{π_1} = O_{π_2}** on the substrate, proving replay-equivalence.

---

4. RSF™, BDL™, and CTC™/Hallway™

4.1 RSF™: Resonant State Function

Purpose: resonance-weighted coherence for mirror weighting.
Definition:

- **RSF(w; S) = w_E·E(w; S) + w_S·S_Coh(w; S) + w_R·RΔ(w; S)**
  with **w_E + w_S + w_R = 1**, **w_* ≥ 0**.

Notes: RSF is **non-deterministic internally** (e.g., tie-breaking among equal-resonance decompositions) yet **replay-equivalent**: for fixed **S, κ**, the resulting mirror weights are the same on the substrate.

4.2 BDL™: Boolean Disambiguation Layer

Purpose: classify and route logical tensions without probabilistic heuristics.
Logic-Type Classifier Grid (excerpt): - **Functional contradiction (FC):** f ∘ g vs g ∘ f outcomes conflict.
- **Recursive tension (RT):** self-application shifts tag alignment across mirrors.
- **Perceptual paradox (PP):** abstract alignment holds, concrete presentation diverges.
- **Protective contradiction (PC):** guard-induced blocking yields surface inconsistency that preserves deeper invariants.
- **Temporal misalignment (TM):** snapshots across evolving S* appear inconsistent though longitudinally coherent.

BDL routing: map each tension to a transformer **T_k** with guard **G_k**; apply in Hallway order until contradiction budget $\leq \theta$.

4.3 CTC™ / Hallway™: Traversal Lock

Purpose: enforce a non-replicable sequence of mirror constraints that guarantees structural integrity and convergence.
Mechanics: a locked order **Λ** over {T_k} with state-dependent entrance tests; failure to respect **Λ** breaks mirror law or idempotence and is rejected by guards. The Hallway ensures **unique substrate outcome** under declared **S**.

---

5. Operational Semantics under Declared State S

5.1 State declaration

A session begins with **S := (inputs, scopes, constraints, thresholds θ, κ)**. All transforms are conditioned on **S**.

5.2 Substrate vs Presentation

- **Substrate:** mirror-pure output satisfying **γ ∘ α = id_C**.
- **Presentation:** overlays (e.g., contradiction maps, uncertainty bands as visualizations) that **never alter substrate**.

5.3 Canonical loop

```
Input: S, w0
w ← w0
repeat
    w ← HCL( BDL( RSF(w; S) ) )         // layered transforms
    w ← CTC_Enforce( w; S )              // Hallway lock Λ
```

```
until G_Entropy(w; S, θ) ∧ G_Integrity(w; S)
return MirrorOut(w)
```

Properties: No sampling; no probability. Internal non-determinism (e.g., order of independent T_k) is absorbed by Hallway constraints to yield replay-equivalence.

---

6. Worked Contrasts and Examples

6.1 Stochastic engine (contrast)

Example: Next-token language model with temperature **$\tau > 0$** samples from **softmax(logits/τ)**.
Behavior: identical prompts yield **different** outputs due to randomness.
Why not TSMLA: TSMLA has **no sampling** and no temperature-driven randomness.

6.2 Deterministic automaton (contrast)

Example: DFA over alphabet Σ with transition δ; given state and input, output is uniquely determined step-by-step.
Behavior: single causal path; identical runs produce identical internal traces.
Why not TSMLA: TSMLA may reorder independent transforms internally, yet the **observable** result remains the same—**replay-equivalence without unique trace**.

6.3 TSMLA miniature

Setup: Tags {a,b,c} with normalized weights w(a)=0.5, w(b)=0.3, w(c)=0.2. Guards: θ=0.1 contradiction budget.

1) **RSF:** harmonically compresses conflicting alignments to produce a reweighted **w′**.
2) **BDL:** classifies a tension as **PC** (protective contradiction) and routes through **T_PC**.
3) **CTC:** enforces **Λ** ordering so **T_PC** occurs after **T_RT** when both present.
4) **Idempotence check:** merging **$w' \oplus w' = w'$**.
5) **Replay:** different internal orders among independent steps yield the **same MirrorOut** for this **S** and **κ**.

---

7. Formal Claims (for Review)

Claim A (Non-stochasticity). The TSMLA substrate contains no operations that require sampling from a distribution or estimating expectations; entropy appears only as normalized tag-weights for structural accounting.
Implication: No Monte-Carlo variance; no reliance on probabilistic convergence.

Claim B (Non-determinism with replay-equivalence). Multiple lawful internal policies **$\pi \in \Pi$** exist; all converge to the same substrate fixed point **$\mathrm{lfp}(\Phi_S)$** for fixed **S, κ**.
Implication: Not classically deterministic, yet observable outcomes match across replays.

Claim C (Idempotence). For lawful transforms **f** and merge **⊕**, **f(w ⊕ w)=f(w)**, ensuring stability under duplication and preventing weight inflation.
Implication: Guards remain meaningful across iterative composition.

Claim D (Mirror law). **γ ∘ α = id_C** and **α ∘ γ = id_A** at the substrate boundary.
Implication: Presentation overlays cannot alter substrate truth.

---

8. Pseudocode (RSF™, BDL™, CTC™)

```
// Types
Signal w = { (tag: t_i, weight: w_i) } with Σ w_i = 1
State  S = { inputs, scopes, thresholds θ, κ }

function HARMONIC_MERGE(u: w, v: w): w
    return tagwise_normalize( max_by_tag(u, v) )        // idempotent

function RSF(w: w, S): w
    E  ← energy_metric(w, S)
    SC ← state_coherence(w, S)
    RD ← resonance_delta(w, S)
    return normalize( w_E*E + w_S*SC + w_R*RD )         // w_E+w_S+w_R=1

function BDL(w: w, S): w
    tensions ← detect_tensions(w, S)
    for τ in tensions do
        class ← classify_logic_type(τ)                  // FC, RT, PP, PC, TM
        w ← apply_transform(class, w, S)                // T_class with guard
    end for
    return w

function CTC_Enforce(w: w, S): w
    for gate in Λ(S) do                                 // Hallway order
        require gate.check(w, S)                        // lock/guard
        w ← gate.apply(w, S)
    end for
    return w

function RUN_TSMLA(S, w0): w
    w ← w0
    repeat
        w1 ← RSF(w, S)
        w2 ← BDL(w1, S)
        w3 ← HARMONIC_MERGE(w2, w)                      // compression
        w  ← CTC_Enforce(w3, S)
```

```
    until entropy_guard(w, S, θ) ∧ integrity_guard(w, S)
    return mirror_out(w)                                   // substrate output
```

Notes: No calls to RNG or sampling. Internal iteration order within **BDL** and admissible rearrangements within **Λ(S)** may vary, but **mirror_out(w)** is replay-equivalent.

---

9. Regulatory and Peer-Review Readiness

- **Auditability:** Guards, thresholds, and Λ-order are parameterized and logged; replays under the same **S, κ** reproduce substrate outputs up to overlays.
- **Safety:** Idempotence prevents runaway amplification; Hallway locks prevent illegal traversal sequences.
- **Comparability:** Stochastic baselines (LLMs, Bayesian engines) and deterministic baselines (automata, fixed pipelines) can be evaluated against TSMLA on: replay-equivalence, idempotence stability, and guard-bounded entropy.

---

10. Frequently Challenged Points

1) **"Non-stochastic means deterministic."** False: TSMLA admits internal non-determinism while guaranteeing replay-equivalence on outputs for fixed **S, κ**. 2) **"Weights imply probabilities."** False: weights are **normalized structural tags**, not random variables; no sampling or expectations are defined. 3) **"Visualization alters results."** False: overlays are presentation-only; **$\gamma \circ \alpha = id\_C$** protects substrate.

---

11. Minimal Axiom Set (TSMLA-AX)

- **AX1 (Mirror Law):** $\gamma \circ \alpha = id\_C$; $\alpha \circ \gamma = id\_A$.
- **AX2 (Idempotent Merge):** $\forall w,\ w \oplus w = w$.
- **AX3 (Monotone Transform):** Φ_S is Scott-continuous on $(L, \leq)$.
- **AX4 (Guarded Convergence):** loops halt only when $G(w; S, \theta)$ holds.
- **AX5 (Non-Stochastic Substrate):** No operation requires sampling or expectation.
- **AX6 (Replay-Equivalence):** $lfp(\Phi\_S\^\pi)$ is invariant over admissible $\pi \in \Pi$ for fixed S, κ.

---

12. Proof Sketches

- **AX2 ⇒ Idempotence of f:** If f respects ⊕, then $f(w \oplus w) = f(w)$.
- **AX3 + AX6 ⇒ Replay-equivalence:** All admissible policies iterate to the same **lfp**.
- **AX1 ⇒ Substrate purity:** Any change in overlays cannot alter C-level outputs.

---

13. Implementation Notes (Phase-1)

- **Public vs NDA:** Public papers include the axioms and operators but omit secret thresholds, Λ ordering details, and classifier parameters.
- **Testing:** Provide adversarial vectors for each BDL class (FC, RT, PP, PC, TM).
- **Metrics:** Report idempotence error (should be zero), guard-constrained entropy, and replay-equivalence delta (should be zero up to overlay).

---

14. Appendix A: Tiny Numeric Walkthrough

Initial **w**: {(a,0.50),(b,0.30),(c,0.20)}.
RSF yields **w′** with minor shift toward coherence on **a**: {(a,0.54),(b,0.28),(c,0.18)}.
BDL detects **PC**; transform scales **b** by 0.95 then renormalizes: {(a,0.55),(b,0.265),(c,0.185)}.
CTC applies **Λ**: enforces **RT→PC** order; result unchanged in substrate relative to any admissible reordering.
Merge with previous state preserves idempotence: **w ⊕ w = w**.
Guards satisfied: return **MirrorOut(w)**.

---

15. Appendix B: Vocabulary

- **Declared state S:** The full run-context: inputs, scopes, constraints, thresholds, κ.
- **Replay-equivalence:** Identical observable substrate outputs across lawful internal policies, for fixed **S, κ**.
- **Entropy (structural):** Normalized tag-weight dispersion used for guards; not probabilistic uncertainty.

---

16. Canon Guard (Extract)

1) Mirror law uses **γ ∘ α = id_C** and **α ∘ γ = id_A**.
2) Substrate is **non-stochastic**, **not deterministic**; internal steps may be non-deterministic; convergence via **unique least fixed point** under guards.
3) Entropy is computed over **normalized tag-weights**; **no probabilistic sampling**.
4) Overlays are **presentation metadata**; substrate output remains mirror-pure.
5) Idempotence and guards: **w ⊕ w = w**; **f(x ⊕ x)=f(x)** under **S**; guard **G(w≥θ)** encoded in **f**; canonical **while ¬G** loop pattern.
6) Naming canon: **HCL™**: Harmonic Compression Layer; **CTC™/Hallway™**: traversal lock; **CAPF™** deferred.

End of LP-Prime v1.0