

THE MAGLEV RAILS PATENT FORTRESS

Publication-Ready Patent Specification Docket: PROV-013

Filing Date: September 25, 2026

Type: Provisional Application for Patent

Status: Confidential Defensive Work Product — For Inventor and Counsel Eyes Only

TITLE OF INVENTION

Zero-Serialization Authority-Attenuated Resident-State Architectures for Deterministic Media, Tool, and Interface Execution

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a standalone U.S. Provisional Patent Application (Docket PROV-013 / Provisional 6). Pursuant to 35 U.S.C. § 111(b)(7) and MPEP 201.08, this provisional application establishes an independent priority date as of its filing date of September 25, 2026.

Benefit under 35 U.S.C. § 119(e) to this application and the following earlier-filed co-pending U.S. Provisional Patent Applications will be claimed in a subsequent consolidated Nonprovisional Utility Application filed under 35 U.S.C. § 111(a) within twelve (12) months of September 22, 2026:

Provisional Priority Anchor	Application No.	Filing Date	Subject Matter
P1	64/159,586	Sept 22, 2026	ATESO Core Substrate & Capability Addressing
P2	64/161,921	Sept 23, 2026	Resident State & Planner Admission
P3	64/161,944	Sept 23, 2026	Multi-Agent Bus & Relocation Tables
P4	64/161,949	Sept 23, 2026	Real-Time Telemetry & Fluid Continua
P5	64/162,043	Sept 24, 2026	Enterprise Memory & Hardware Admission
P6	PROV-013	Sept 25, 2026	The Rooftop & The Ladder (Maglev Fortress)

The complete disclosures of each of the above-listed provisional applications, as well as the specifications titled **ROOFTOP-AND-LADDER-SPEC.md** and **MAGLEV-PATENT-FORTRESS.md**, are hereby incorporated by reference in their entireties.

TECHNICAL FIELD

The present invention relates generally to deterministic execution systems, and more particularly to zero-serialization binary direct-memory command pipelines, authority-attenuated neural network decoding, closed-form timeline reconstruction, resident-state memory arenas, and hardware-neutral key-value cache admission.

BACKGROUND OF THE INVENTION & PRIOR ART INEFFICIENCIES

Existing machine-learning tool invocation systems rely on textual serialization formats such as JSON or JSON-RPC. Textual tool-invocation paths may construct intermediate string representations and perform parsing, validation, and object materialization. The disclosed embodiments instead couple constrained descriptor generation to a binary command representation and an authority- and revision-gated execution path. Performance depends on the workload, representation, validation obligations, runtime, and hardware. No numerical latency comparison is asserted here for a configuration that has not been measured. Furthermore, under representative managed-runtime memory accounting (such as a 224-byte live-object baseline with an assumed 25% garbage-collection headroom), managed heap allocations consume approximately $4.14\times$ the memory capacity of fixed-stride structures (or $3.11\times$ without headroom). The disclosed memory embodiments combine an invariant 64-byte payload granule with structured memory accounting—comprising a 64-byte payload, an 8-byte directory entry (12.5% structural overhead), and hierarchical Merkle tree verification metadata—to achieve zero-heap execution with bounded memory overhead.

Media pipelines may preserve encoded packets through streamcopy or container-level operations when the selected operation and formats permit. Operations requiring decoded-frame transformations may require decoding and re-encoding. The disclosed media embodiments address the coordination of transcript-linked edit boundaries, payload-preserving segment references, and direct-time state evaluation under declared codec, container, and synchronization constraints.

Existing memory systems use heap allocation during state mutation, dynamic directory metadata, and double buffering without cryptographic revision binding. PostgreSQL TOAST and SQLite overflow pages address variable-width storage but do not provide an invariant 64-byte granule with radix-offset extension arithmetic, dual-epoch density-based shadow swapping, and decoupled indexing structures.

Existing capability systems such as Macaroons or MCP use textual RPC and static grammars. They do not dynamically recompile token logit masks from a delegation lattice at every decoder step.

Existing UI animation systems use `requestAnimationFrame`, CSS transitions, or numerical integration. They do not provide bit-identical temporal reconstruction under a declared IEEE-754 profile across non-coordinated cross-window runtimes.

Existing attention KV-cache systems, such as PagedAttention and GPUDirect RDMA, do not require cryptographic block admission through an epoch-published descriptor with a hardware-verified Merkle leaf tag and density-triggered peer replication.

Thus, there is a need for a unified, hardware-rooted, deterministic, and auditable pipeline as described herein.

SUMMARY OF THE INVENTION

The present invention provides a six-domain architecture:

1. **Zero-Serialization Binary Direct Memory** — A neural decoder emits binary opcode tokens into a memory-mapped SPSC ring buffer with a revision counter and authority-lattice descriptor, without JSON or string intermediates.
2. **Closed Edit-Decision Container & Invariant Timeline** — Media tracks are stored as closed-form analytical state functions with container truncation, boundary-evaluated Kuramoto phase synchronization, and semantic lattice affinities.
3. **Resident-State & Epoch-Bound Radix-Offset Arena** — A 64-byte granule arena uses radix-offset extension arithmetic, dual-epoch density-triggered shadow swapping, and isolated side-table metadata.
4. **Sovereign Capability Tool Broker** — A hardware-rooted broker dynamically masks decoder logits from a delegation lattice, speculatively executes read-only tools, and commits only after non-amplification verification.
5. **Closed-Form Harmonic State $\Pi(\text{record}, t)$ & Entanglement** — Bit-identical spatial state reconstruction using the damped harmonic oscillator under a declared IEEE-754 profile, cross-window SharedArrayBuffer transport, and native fluid displacement shaders.
6. **Hardware-Neutral KV-Block Sovereign Admission** — A 64-byte packed descriptor, hardware-verified Merkle leaf tag, epoch-published block admission, and density-triggered peer replication.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

I. Definitions and Conventions

As used herein:

- **Binary opcode vocabulary** means a finite set of tokens, each token mapping to a canonical byte encoding of an opcode or operand defined by an opcode schema.
- **Authority-lattice descriptor** means a data structure, such as a packed bit vector, that represents permissions in a meet-semilattice.
- **Non-amplification rule** means that a requested permission set satisfies $(\text{granted} \cap \text{required}) == \text{required}$, and where delegation is present, the granted authority is a strict descendant of a parent authority in a declared lattice.
- **Meet-semilattice** means a partially ordered set in which every nonempty finite subset has a greatest lower bound, used herein for authority composition.
- **Closed-form analytical state function** means a function of continuous time that yields playback position, clip identity, and transform coordinates without cumulative numerical integration state.
- **Bit-identical** means byte-for-byte identical output under a declared IEEE-754 precision and rounding mode.
- **Granule** means a 64-byte aligned memory region.

- **Epoch** means a monotonic generation identifier bound to a revision condition.
- **Mutation density** means $\rho_t = k_t / N$, where k_t is the number of mutated 64-byte granules in an epoch interval, and N is the total number of active granules in an arena.

II. Domain 1 — Zero-Serialization Binary Direct Memory

A neural decoder emits a token sequence over a binary opcode vocabulary. A per-position logit mask is generated from an opcode schema Σ . The mask restricts token sampling strictly to valid opcode-to-operand transitions.

The sampled token is decoded into its canonical byte encoding and written directly into a memory-mapped SPSC ring buffer slot. No intermediate JSON, string, or text allocation occurs.

The slot header includes:

```

struct spine_slot_header {
    uint64_t magic;                // 0x5350494E4550494E ("SPINESPINE")
    uint32_t revision;             // Monotonic state revision counter
    uint16_t authority_mask;       // Packed 16-bit authority lattice descriptor
    uint16_t opcode_count;         // Number of contiguous binary opcodes
    uint64_t struct_length;        // Total byte length of payload
    uint64_t relocation_table_off; // Relative offset to relocation table
    uint64_t data_off;             // Relative offset to binary payload
    uint8_t reserved[24];          // Explicit cache-line pad; total header = 64 bytes
} __attribute__((packed));

struct spine_reloc_entry {
    uint32_t relative_offset;      // Stride relative to slot base
    uint16_t granule_index;        // 64-byte granule target index
    uint16_t field_width;          // Field width in bytes (1, 2, 4, 8)
} __attribute__((packed));

```

The relocation table resolves each relative offset against the current slot base address. The host execution agent reads the relocatable binary struct directly from the slot and executes the corresponding tool only after verifying:

1. `revision > previous_revision_for_slot`; and
2. The authority-lattice descriptor satisfies the non-amplification rule.

The SPSC ring buffer is preferably backed by a persistent file mapping having a `.spine` extension. The same physical pages are mapped into both the decoder virtual address space and the executor virtual address space, enabling zero-copy transfer by virtual memory page aliasing.

III. Domain 2 — Closed Edit-Decision Container and Invariant Timeline

Media tracks are represented as closed-form analytical state functions $x_i(t)$. These functions yield, for arbitrary continuous time t , playback position, clip identity, and transform coordinates with zero cumulative

frame-update state.

A transcript edit is admitted as an authenticated mutation descriptor targeting a verified token span. The token span maps to a media interval `[t_start, t_end]`.

Container truncation modifies container boundary index tables only. Encoded frame payloads of retained segments remain bit-identical, with no re-render or re-encode.

Phase synchronization across independent media tracks is performed according to the Kuramoto phase equation:

$$\frac{d\theta_i}{dt} = \omega_i + \sum_j K_{ij} \sin(\theta_j - \theta_i)$$

The coupling gains K_{ij} are evaluated exclusively at segment boundaries. During each open interval between segment boundaries, the phase offsets are held fixed. This mathematical restriction removes per-frame numerical integration drift while preserving exact phase alignment at edit boundaries.

Semantic clustering arranges transcript segments in a multi-dimensional spatial lattice using Mycelium-Agar affinity dynamics. Semantic affinity creates attractive spring forces. Semantic dissimilarity creates repulsive diffusion forces. An elastic edge is formed when semantic embedding similarity exceeds a predetermined threshold. The spring constant is proportional to token overlap. Lattice positions are integrated at a coarse interval τ decoupled from video display frame rates.

IV. Domain 3 — Resident-State and Epoch-Bound Radix-Offset Arena

Storage for variable-width attributes is preallocated in a contiguous `.span` arena before command admission. Dynamic heap allocations are prohibited.

The base 64-byte granule maintains an invariant array stride. The granule includes an identity field, a revision field, a 16-bit packed extension bitmask, and a relative 32-bit overflow offset.

The radix-offset bit-shift formula for extension attribute addressing is:

$$\text{attr_addr} = \text{granule_base} + \text{primary_stride} + (\text{extension_index} \ll \text{shift})$$

where:

- `granule_base` is the base address of the 64-byte granule;
- `primary_stride` is a fixed byte offset from the granule base to the primary attributes;
- `extension_index` is a 0-based index into the extension region;
- `shift` is a predefined shift based on field width, for example `shift = log2(field_width)`.

A SIMD stride engine accesses fixed primary granule attributes at $\mathcal{O}(1)$ stride and resolves variable-width attributes in $\mathcal{O}(1)$ with the above formula.

A dual-epoch slab manager maintains an active resident arena and an inactive shadow arena. It monitors real-time mutation density:

$$\rho_t = \frac{k_t}{N}$$

where k_t is the number of mutated 64-byte granules in the current epoch interval and N is the total number of active granules.

When $\rho_t \leq 0.15$, the system performs in-place sparse delta updates. When $\rho_t > 0.15$, the system performs direct-memory bulk streaming into the shadow arena followed by an atomic root pointer swap. The epoch identifier is bound to the revision condition κ , so any sparse mutation minted for epoch e cannot land in epoch $e+1$.

An isolated virtual-memory side table stores entity directory indices, revision state, and authority permissions separately from the primary granules.

In one embodiment, fully occupied 2 MiB data blocks are authenticated using an unpadded binary Merkle tree storing one 32-byte digest for each leaf and each internal node. For L data blocks, digest storage is $32(2L - 1)$ bytes and the ratio of digest storage to represented block bytes is less than $64/2,097,152$, or approximately 0.003052%. This ratio excludes directory entries, allocation padding, separately stored proofs, signatures, and additional arena or tree copies. Total memory accounting includes those components separately. The occupancy condition is required: a 32-byte digest for 64 occupied bytes is 50% overhead relative to occupied data, even when the containing allocation has a nominal 2 MiB capacity.

V. Domain 4 — Sovereign Capability Tool Broker

Tool invocation commands are words δ within a strictly declared binary schema Σ . Authority is a tuple:

$$a = (\text{Ops}, \text{Obj}, \text{Val}, \text{Win}, \text{Del})$$

Admission computes the meet operation down the delegation chain, enforcing:

$$\alpha(\delta) \sqsubseteq a_j$$

across all ancestors.

At each autoregressive decoder step q , the dynamic vocabulary masking engine recompiles a token logit mask from the active delegation lattice. The permitted token set is strictly the set of tokens forming a prefix of a valid descriptor δ satisfying:

$$\alpha(\delta) \sqsubseteq a_{\text{leaf}}$$

Static JSON grammars cannot perform this because the mask depends on the active authority lattice at decoder state q .

Speculative read-only tool execution runs ahead of token-stream commitment. If the speculative operation attempts a store or state mutation outside the designated read-only buffer, the execution engine aborts and reverts.

Tool responses are written directly into resident memory slabs at preallocated offsets. The generative model re-ingests the response directly from the invariant base address, without JSON-RPC translation. A prompt-injection isolation barrier validates the cryptographic provenance of the tool response against the active capability token before making the response visible to the decoder.

Commit occurs only when the final emitted token sequence satisfies:

$$\text{granted} \cap \text{required} = \text{required}$$

against all ancestral authorities.

VI. Domain 5 — Teeming Closed-Form Harmonic State $\Pi(\text{record}, t)$ and Entanglement

Widget spatial state is stored as an immutable admitted record containing a continuous start time t_0 , target coordinates, and underdamped harmonic oscillator parameters (ω, ζ) with $0 < \zeta < 1$.

The spatial state vector $\Pi(\text{record}, t)$ is evaluated directly at continuous time t according to:

$$s(t) = 1 - e^{-\zeta\omega(t-t_0)} \left(\cos(\omega_d(t-t_0)) + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin(\omega_d(t-t_0)) \right)$$

where:

$$\omega_d = \omega\sqrt{1-\zeta^2}$$

This equation is evaluated under a declared IEEE-754 precision profile. Scrubbing to arbitrary continuous time t produces bit-identical output with zero Euler integration drift and zero per-frame accumulation.

Cross-window runtime entanglement propagates binary records through `SharedArrayBuffer` memory blocks using `Atomics.wait` and `Atomics.notify`. This eliminates `postMessage` JSON serialization.

Procedural fluid displacement is rendered natively on WebGL2 or WebGPU. The shader is bound to daylight paper-tone tokens including `#F4F4F2` ground and `#1D4AFF` cobalt mark. The fluid displacement continuum evaluates Navier-Stokes velocity advection and vorticity curl noise on GPU compute or fragment shaders. When a `prefers-reduced-motion` media query evaluates true, the motion is clamped to zero.

VII. Domain 6 — Hardware-Neutral KV-Block Sovereign Admission

The KV-block descriptor is a 64-byte packed structure:

```
struct kv_block_descriptor {
    uint32_t block_id;           // Physical block identifier
    uint32_t epoch;              // Monotonic state epoch
    uint64_t phys_addr;          // Physical memory address (HBM/DRAM)
    uint32_t layer_idx;          // Attention layer index
    uint32_t head_idx;           // Attention head index
    uint32_t token_start;        // Starting sequence token offset
    uint32_t token_count;        // Number of tokens in block
    uint8_t merkle_tag[32];      // Cryptographic leaf integrity tag
} __attribute__((packed));
```

A direct-memory mapping layer maps resident-state granules directly into an attention KV-cache block buffer via a direct memory access (DMA) engine. A hardware cryptographic engine verifies the 32-byte Merkle leaf integrity tag over the mapped attention block at wire speed.

The block becomes visible to a neural network decode kernel only upon atomic publication of the epoch counter. The decode kernel unconditionally drops or masks any KV block whose epoch does not match the active

generation epoch.

An execution router replicates admitted KV blocks across peer memory nodes via high-speed interconnect when:

$$\rho_t > \tau$$

where τ is a predetermined density threshold.

GPUDirect RDMA, BlueField DPUs, NVLink, and CXL memory controllers are recited strictly as dependent species of “a direct memory access engine” and are not required in the broadest independent embodiment.

VIII. Electronic Record Integrity and Evidentiary Support

Each commit, revision, authorization decision, speculative execution event, and state mutation generates an append-only electronic record. The record includes at least:

- a monotonic epoch identifier;
- a revision condition value;
- an authority-lattice descriptor;
- a cryptographic integrity tag;
- a timestamp; and
- an operator or model identifier.

Electronic records and copied data may be authenticated through the certification procedures in Rules 902(13) and 902(14), where applicable. Those procedures do not themselves establish a hearsay exception. Where a record qualifies as a record of a regularly conducted activity, a custodian or other qualified person may provide a Rule 902(11) certification addressing Rule 803(6)(A)–(C), subject to applicable notice, inspection, trustworthiness, and other evidentiary requirements. Cryptographic verification does not independently establish the truth of the recorded assertions, and admissibility remains subject to judicial determination.

This evidentiary capability provides an auditable, verifiable record-keeping foundation, but is distinguished from the technical mechanisms of the claimed invention.

IX. EU Data Act, GDPR Art. 20, and Machine-Readable Portability Architecture

The resident-state arena, `.spine` bus, and side-table metadata store maintain data in a structured, commonly used, machine-readable format. A user or controller may request direct client-side export.

The system is architected to facilitate technical compliance with data portability frameworks such as **GDPR Article 20** and data-sharing interfaces under the **EU Data Act Regulation (EU) 2023/2854** by providing an authenticated, structured export path that does not depend on intermediate textual serialization.

However, statutory data portability rights are subject to statutory conditions and exceptions (including the protection of trade secrets, intellectual property, and third-party rights), and the EU Data Act expressly prohibits using coercive means or abusing vulnerabilities in protective infrastructure to obtain data. Providing client-side export capability is an architectural implementation design; it does not confer blanket statutory immunity under

18 U.S.C. § 1030 (Computer Fraud and Abuse Act) or other computer access statutes, nor does it override contractual terms, common-law duties, or legal obligations governing authorized access to external computer systems.

FORMAL PATENT CLAIMS

What is claimed is:

Claim 1. A computing system for zero-serialization invocation of execution tools from a machine-learning decoder, comprising:

- a. a machine-learning decoder that emits a token sequence over a binary opcode vocabulary;
- b. an opcode schema coupled to said decoder defining a set of valid opcode-to-operand transitions, wherein the decoder applies a per-position logit mask generated from said opcode schema such that each emitted token strictly encodes a valid opcode or operand of a canonical binary tool-call structure;
- c. a memory-mapped single-producer single-consumer (SPSC) ring buffer comprising a plurality of slots, each slot having a header and a payload, the header including a monotonic revision counter and an authority-lattice descriptor, the payload including a relocatable binary struct;
- d. a relocation table stored in association with the ring buffer, the relocation table containing relative offsets resolved against a current slot base address; and
- e. a host execution agent configured to read the relocatable binary struct directly from said slot and execute a corresponding tool, wherein the host agent executes said tool only after verifying (i) that the monotonic revision counter is greater than a previously observed counter for said slot and (ii) that a permission check using the authority-lattice descriptor satisfies a meet-semilattice non-amplification rule.

Claim 2. A method for zero-serialization command transfer from a neural network decoder to an execution runtime, comprising:

generating unnormalized token logits at a neural network decoding step;

applying a dynamic binary mask to said logits, the mask derived from an opcode schema specifying valid binary command encodings;

sampling a binary token based on the masked logits;

decoding the sampled binary token into a canonical byte encoding and writing the canonical byte encoding directly into a memory-mapped single-producer single-consumer (SPSC) ring buffer slot as a relocatable binary struct, without generating intermediate textual, string, or JSON representations;

storing in a slot header associated with said slot a monotonic revision counter and an authority-lattice descriptor;

relocating the binary struct against a current buffer base address using a relative offset table; and

executing the binary struct only upon verifying that (i) the monotonic revision counter indicates a newer revision and (ii) the authority-lattice descriptor satisfies a non-amplifying subset check for the targeted tool.

Claim 3. The system of claim 1, wherein the memory-mapped SPSC ring buffer is backed by a persistent file mapping having a `.spine` extension mapped into both the decoder virtual address space and the executor virtual address space, enabling zero-copy command transfer via virtual memory page aliasing.

Claim 4. The system of claim 1, wherein the authority-lattice descriptor comprises a packed bit vector, and the non-amplifying permission check evaluates `granted & required = required`, and wherein if a delegation bit is set, the granted authority is verified to be a strict descendant of a parent authority within a declared lattice.

Claim 5. The method of claim 2, wherein the relocatable binary struct is aligned to a 64-byte granule stride, and each entry of the relative offset table comprises `{reloff, granule_index, width}` resolved via a single radix-offset shift operation according to `attr_addr = granule_base + primary_stride + (extension_index << shift)`.

Claim 6. A media editing system for deterministic timeline reconstruction, comprising:

- a. a track store representing each media track as a closed-form analytical state function $x_i(t)$ that deterministically yields, for arbitrary continuous time t , playback position, clip identity, and transform coordinates with zero cumulative frame-update state;
- b. a transcript editor configured to receive a deletion event targeting a token span and map the token span to a media interval $[t_{start}, t_{end}]$ using an authenticated token timestamp index;
- c. a container rewriter configured to perform a deterministic container truncation on a media container segment associated with $[t_{start}, t_{end}]$ by modifying container boundary index tables while retaining encoded frame payloads of retained segments bit-identical without re-rendering or re-encoding;
- d. a phase synchronization engine configured to synchronize at least two independent media tracks by evaluating coupled Kuramoto phase updates $\frac{d\theta_i}{dt} = \omega_i + \sum_j K_{ij} \sin(\theta_j - \theta_i)$, wherein coupling gains K_{ij} are evaluated exclusively at segment boundaries; and
- e. a semantic clustering engine configured to arrange transcript segments in a multi-dimensional spatial lattice using Mycelium-Agar affinity dynamics wherein semantic affinity creates attractive spring forces and semantic dissimilarity creates repulsive diffusion forces.

Claim 7. A method of editing media timelines via immutable token boundaries, comprising: storing, for each of a plurality of media tracks, an analytical state function $x_i(t)$ having zero frame-delta accumulators; receiving a token deletion event targeting a token span from a transcript interface; mapping the deleted token span to a bounded media interval $[t_{start}, t_{end}]$; truncating a media container at boundaries derived from $[t_{start}, t_{end}]$ by updating a segment reference table while preserving bit-identical frame payloads for unedited segments; adjusting a phase synchronization variable between tracks using a Kuramoto phase equation evaluated at boundary transitions; and updating a multi-dimensional semantic lattice by applying attractive and repulsive affinity forces between transcript nodes.

Claim 8. The system of claim 6, wherein container truncation modifies container duration headers and index chunk offsets without executing a video decompression or recompression cycle.

Claim 9. The method of claim 7, wherein the Kuramoto phase update is evaluated strictly at segment transition boundaries and is inhibited during steady-state playback, ensuring zero frame drift during continuous timeline scrubbing.

Claim 10. The system of claim 6, wherein the Mycelium-Agar clustering engine forms an elastic edge between two transcript nodes when semantic embedding similarity exceeds a predetermined threshold, said edge having a spring constant proportional to token overlap, and integrates lattice positions at a coarse interval τ decoupled from video display frame rates.

Claim 11. A resident-state memory architecture for zero-allocation state mutation, comprising: (a) a primary contiguous memory arena partitioned into fixed 64-byte granules aligned to cache lines, each granule having an invariant array stride and comprising an identity field, a revision field, a 16-bit packed extension bitmask, and a relative 32-bit overflow offset; (b) an auxiliary preallocated overflow arena (`.span`) configured to store variable-width attributes at relative offsets addressed by said relative 32-bit overflow offset without initiating dynamic heap allocation; (c) a SIMD stride engine configured to access fixed primary granule attributes at $O(1)$ stride and resolve variable-width attributes via said relative 32-bit overflow offset in $O(1)$ time; (d) a dual-epoch slab manager configured to maintain an active resident arena and an inactive shadow arena, monitor a real-time mutation density $\rho_t = k_t/N$, execute in-place delta updates when $\rho_t \leq \tau$, and execute direct-memory bulk streaming into the shadow arena followed by an atomic root pointer swap when $\rho_t > \tau$; and (e) an isolated virtual-memory side table storing entity directory indices, revision state, and authority permissions separately from the primary granules, and a hierarchical cryptographic integrity tree comprising leaf digests corresponding to arena data blocks and internal-node digests derived from child digests.

Claim 12. An integrated circuit memory controller for hardware-accelerated capability verification, comprising: a command ingress interface coupled to a system memory bus; a capability verification pipeline configured to extract a capability descriptor, evaluate an authority bitmask against a permitted interval, and compare an expected revision value with a stored monotonic revision register; an SRAM execution unit configured, upon successful verification, to apply a binary mutation directly to an SRAM-backed memory slab; and a hardware control path configured to execute the capability check, revision comparison, and SRAM write within 4 to 8 clock cycles without asserting a host processor interrupt.

Claim 13. The system of claim 11, wherein the 16-bit packed extension bitmask assigns distinct bit positions to declared attribute widths, and the relative address of an extension attribute is computed via direct bit-shift arithmetic according to $\text{attr_addr} = \text{granule_base} + \text{primary_stride} + (\text{extension_index} \ll \text{shift})$.

Claim 14. The hardware memory controller of claim 12, further comprising a hardware revocation engine that invalidates active capabilities by incrementing an epoch counter register, causing the verification pipeline to reject access requests carrying mismatched epoch values.

Claim 15. The system of claim 11, wherein the mutation density threshold τ equals 0.15, and wherein the atomic root pointer swap binds an epoch identifier to the revision condition κ , rejecting any sparse mutation minted under an earlier epoch.

Claim 16. A secure tool invocation architecture for machine-learning inference engines, comprising: (a) a hardware-rooted capability broker holding an authority meet-semilattice and a monotonic revision register, configured to issue capability tokens enforcing a non-amplification rule across delegation chains; (b) a dynamic

vocabulary masking engine configured to compute, at each step of an autoregressive model decoder, a logit mask restricting token sampling strictly to valid prefixes of binary mutation descriptors permitted by the active capability token; (c) a speculative execution engine configured to execute, during decoder generation and prior to token-sequence commitment, a read-only tool operation matching a cryptographically signed read-only schema; and (d) an execution barrier configured to commit state mutations resulting from generated binary descriptors only upon verifying that the monotonic revision register matches a target state revision and that the capability token satisfies an authority verification check.

Claim 17. A method for sovereign tool invocation from generative models, comprising: maintaining an authority meet-semilattice and a monotonic revision counter within a trusted execution domain; dynamically compiling a token logit mask during model decoding, wherein the mask restricts token candidates strictly to valid prefixes of binary descriptors permitted by an active authority descriptor; speculatively executing a read-only tool operation prior to final sequence generation; writing a zero-copy tool response directly into a preallocated resident memory slab; permitting the generative model to read the tool response directly from the resident memory slab; and committing the state mutation only if the generated descriptor satisfies a non-amplification check granted $\cap$ required = required against all ancestral authorities.

Claim 18. The system of claim 16, wherein the speculative execution engine aborts and reverts state if the speculative operation attempts a memory store or state mutation outside the designated read-only buffer.

Claim 19. The method of claim 17, wherein the zero-copy tool response is written into a `.span` resident arena granule having a 16-bit packed extension mask, and the generative model ingests the response by referencing the granule's invariant base address without invoking a JSON-RPC serialization bridge.

Claim 20. The system of claim 16, further comprising a prompt-injection isolation barrier that validates the cryptographic provenance of the tool response against the active capability token before making the response visible to the decoder, discarding any payload containing unverified authority directives.

Claim 21. A spatial user interface system for bit-identical temporal reconstruction across asynchronous display runtimes, comprising: (a) a spatial state store recording widget motion parameters as an admitted record comprising an absolute time datum t_0 , target coordinates, and harmonic oscillator parameters (ω, ζ) ; (b) a closed-form analytical motion engine configured to evaluate a spatial state vector $\Pi(\text{record}, t)$ directly at continuous time t according to:

$$s(t) = 1 - e^{-\zeta\omega(t-t_0)} \left(\cos(\omega_d(t-t_0)) + \frac{\zeta}{\sqrt{1-\zeta^2}} \sin(\omega_d(t-t_0)) \right)$$

where $\omega_d = \omega\sqrt{1-\zeta^2}$, wherein scrubbing to arbitrary continuous time t evaluates without cumulative frame integration drift; (c) a cross-window spine bus configured to stream zero-copy binary state between independent browser windows via `SharedArrayBuffer` memory blocks and atomic synchronization primitives; and (d) a native procedural fluid displacement shader bound to daylight tokens including `#F4F4F2` ground and `#1D4AFF` cobalt mark, executable across WebGL2 and WebGPU without experimental browser flags.

Claim 22. A method for zero-drift spatial user interface rendering, comprising: storing a widget motion specification as an immutable record including continuous start time t_0 and harmonic parameters; receiving an asynchronous scrub request targeting continuous time t ; evaluating the spatial state vector $\Pi(\text{record}, t)$ directly

from the closed-form analytical equation under a declared IEEE-754 floating-point profile with zero per-frame numerical accumulation; writing the evaluated spatial state vector directly into a `SharedArrayBuffer` using atomic memory operations; and rendering a procedural fluid displacement continuum on a native canvas surface bound to daylight paper-tone tokens.

Claim 23. The system of claim 21, wherein the cross-window spine bus synchronizes state between an editing master window and a preview window via `SharedArrayBuffer` atomics without invoking `DOM.postMessage` string serialization.

Claim 24. The system of claim 21, wherein the procedural fluid displacement continuum evaluates Navier-Stokes velocity advection and vorticity curl noise on GPU compute or fragment shaders, and clamps motion to zero when a prefers-reduced-motion media query evaluates true.

Claim 25. The method of claim 22, wherein evaluating $\Pi(\text{record}, t)$ produces bit-identical pixel coordinates across independent computing devices executing identical IEEE-754 rounding modes.

Claim 26. An enterprise hardware memory admission system for language model attention state, comprising: (a) a direct-memory mapping layer configured to map resident state granules directly into an attention key-value (KV) cache block buffer via a direct memory access (DMA) engine; (b) a hardware cryptographic engine configured to verify a 32-byte Merkle leaf integrity tag over said mapped attention block at wire speed; (c) a block admission controller configured to publish a 64-byte packed KV-block descriptor comprising a block identifier, an epoch counter, a physical address, token sequence indices, and said 32-byte Merkle leaf integrity tag, wherein said block becomes visible to a neural network decode kernel only upon atomic publication of said epoch counter; and (d) an execution router configured to replicate admitted KV blocks across peer memory nodes via high-speed interconnect when a mutation density ρ_t exceeds a predetermined threshold.

Claim 27. A method for hardware-moated attention cache admission, comprising: mapping a resident-state `.span` arena directly into an attention KV-cache block buffer via direct memory access; verifying an integrity tag over the mapped attention block using a hardware cryptographic verification unit; publishing a packed block descriptor comprising a physical address, token offsets, an epoch counter, and the integrity tag; and restricting access by an attention decode kernel exclusively to blocks whose epoch counter matches an active generation epoch, and discarding mismatched blocks without interrupting a host processor.

Claim 28. The system of claim 26, wherein the direct memory access engine is a GPUDirect RDMA subsystem, and the hardware cryptographic engine is a Data Processing Unit (DPU) verifying Merkle leaf integrity at line rates of at least 400 Gbps.

Claim 29. The system of claim 26, wherein peer memory replication executes over an NVLink interconnect at bandwidths of at least 900 GB/s per direction, bypassing host CPU DRAM.

Claim 30. The method of claim 27, wherein the attention KV-cache buffer is updated via direct hardware memory writes without executing an intermediate text token serialization or host operating system context switch.