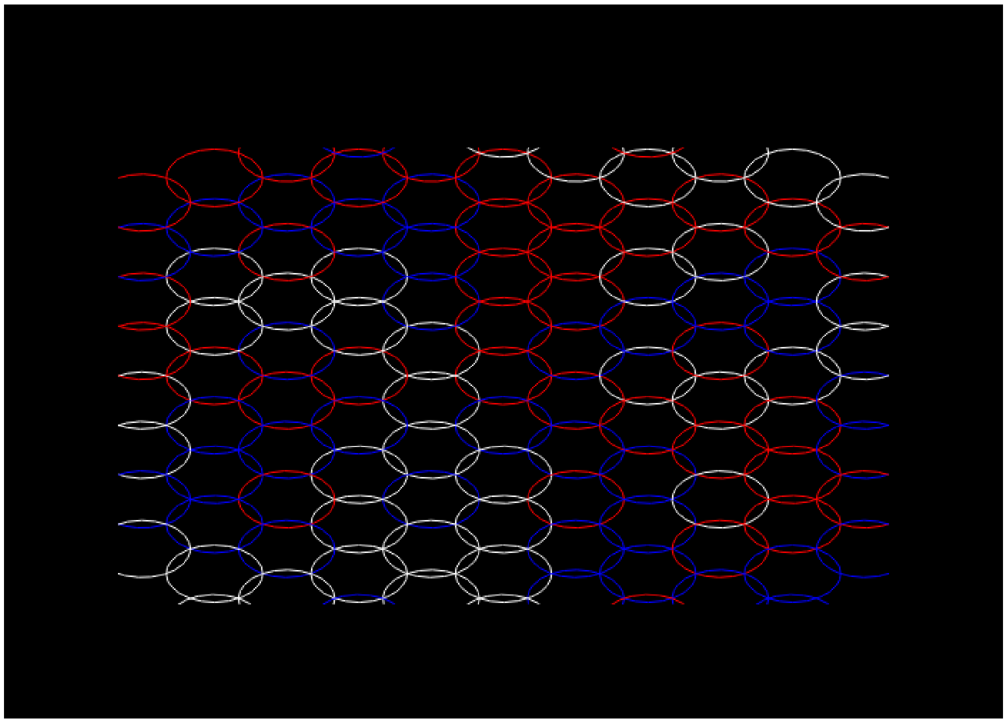
MKULTRA
A Grounded AI Framework for Self-Training and Structured Reasoning

Enrique Flores

July 30, 2025



1 Technical Field

This invention relates to artificial intelligence, structured memory systems, and self-improving language models. It introduces a hybrid architecture that integrates a deterministic state engine with a generative language model to enable reliable reasoning, simulation, and adaptation across any domain governed by formal rules. While the chess domain is used throughout as a concrete example, the invention applies equally to a wide range of state-constrained systems such as turn-based games, legal protocol engines, process automation, training simulators, and resilient control systems.

2 Background

Large language models (LLMs) have shown remarkable fluency in natural language generation, yet they are fundamentally limited in reasoning about dynamic environments. These models operate as statistical predictors without persistent memory, internal verification, or consistent access to structured ground truth. As a result, they are prone to hallucination, inconsistency, and an inability to reason through complex sequences governed by deterministic rules.

In contrast, traditional rule-based systems—such as chess engines—excel at exhaustive search and consistent evaluation within narrow domains, but lack the flexible abstraction and learning capacity of LLMs. These symbolic engines are also typically handcrafted, rigid, and opaque to language-based introspection.

This invention bridges the gap between these two paradigms by encoding the evolving state of a system in a secure, verifiable database and providing a query interface that allows a language model to access valid state transitions and historical context. This enables AI systems to reason about structured environments with both flexibility and rigor, unlocking a new class of hybrid intelligence capable of grounded, self-verifiable decision-making.

3 Summary of the Invention

The invention is a system comprising:

a. A cryptographically secure, rule-constrained database (e.g., MKSTORM) that encodes the state of a domain (such as a chessboard).

b. An interface that exposes the current state, legal transitions (moves), and historical actions to an external agent.

c. A language model (LLM) that queries this interface to generate context-aware decisions.

d. An optional subsystem for generating synthetic training data using legal permutations, enabling domain-specific LLM fine-tuning.

4 Detailed Description

4.1 System Components

- **MKSTORM state engine** – MKSTORM is a previously disclosed and patent-pending architecture for field-coherent, entropy-aligned memory and synchronization. In the context of this invention, MKSTORM functions as the deterministic substrate for encoding the full state of the game domain (e.g., chess), including:
  - Board state at every discrete timepoint,
  - Legal move transitions encoded as verified resolution paths,
  - Immutable move history blocks with monotonic 128-bit entropy timestamps,
  - Universally unique addressing for each board state and transition node.

    MKSTORM operates without reliance on global clocks or consensus mechanisms. Instead, it encodes temporal coherence via deterministic entropy snapshots generated by MKRAND. These snapshots serve as a digital "heartbeat" for advancing and validating the game state. The database is optimized for millisecond- and microsecond-resolution event tracking, making it suitable for both slow (human-paced) and fast (autonomous or simulated) play environments.

    In this invention, MKSTORM provides the **truth layer** — ensuring that all interactions, queries, and decisions made by the language model are grounded in provable and reproducible digital state. Because the MKSTORM engine enforces deterministic resolution of time, space, and causality, it becomes a perfect arbiter of board logic, and thus suitable for controlling AI agents operating within rule-constrained environments.

    This system leverages MKSTORM not only as a state store, but as a **causal anchor**: every state update and rule application is stored as a verified temporal transaction, enabling not only forward simulation but retrospective analysis, game replay, and synthetic trajectory mutation for training data generation.

- **Move legality checker** – This subsystem enforces the rules of the domain (e.g., chess) by evaluating proposed state transitions against a set of formally encoded move constraints. In the chess-specific implementation, the legality checker relies on a compact and extensible move definition table stored within the MKSTORM database.

  Each piece type (e.g., pawn, knight, bishop) is associated with a rule descriptor that defines:
  - Allowed vector displacements (e.g., $(\pm 1, \pm 1)$ for bishops),
  - Directionality constraints (e.g., pawns move forward only),
  - Capture conditions (e.g., diagonal captures for pawns),
  - Special rules (e.g., castling, en passant),
  - Movement bounds (e.g., range 1 for king, unbounded for rook),

  - Conditional modifiers (e.g., first-move rules, check legality).

  These descriptors are represented as structured data—such as tagged tuples, declarative logic entries, or compiled movement vectors—and are queried dynamically during gameplay. When a move is proposed (either by a player or the language model), the legality checker executes a comparison between the current state, the move action, and the encoded movement rules.

  Legal transitions result in a state update within MKSTORM, along with a new entropy-stamped block representing the move. Illegal transitions are rejected, and optionally returned with diagnostic messages to the LLM interface.

  This checker can be implemented as a:

  - Logic engine within the MKSTORM runtime,
  - Finite state machine evaluator mapped to piece types,
  - Constraint resolver using a declarative rule DSL,
  - Optional hardware-accelerated module in FPGA/ASIC form.

  The key innovation lies in separating the logic of gameplay from the generative model: the LLM never infers legality—it queries it. This ensures all generated responses adhere to formal rules, preventing illegal moves and reinforcing correct play through repeated grounding in the domain's logic.

- **LLM interface module** – This component serves as the connective layer between a language model (LLM) and the deterministic game state managed by MKSTORM. It translates natural-language or structured model outputs into constrained queries, and returns contextually relevant information from the database to the model.

  The interface exposes a set of functions or endpoints the LLM can invoke, including but not limited to:

  - `get_current_state/0` – Returns the current board position as a FEN string, a matrix, or an abstract token set.
  - `get_legal_moves(piece)` – Lists all legal moves for a given piece at a given position.
  - `submit_move(from, to)` – Attempts a move, returns success/failure with explanation.
  - `get_history/0` – Retrieves full or partial move history for reasoning or commentary.
  - `describe_state/0` – Returns a compressed or summarized view of the current tactical situation (optional, for narrative AI).

  This module may operate through:

  - A structured query language or JSON-RPC style API,

  - A native function interface embedded into a prompt template,
  - External HTTP/gRPC calls if the model is hosted remotely,
  - Direct shared memory access in embedded or hardware-bound deployments.

  Critically, this architecture **reverses the traditional flow**: the model does not internally track the game state or simulate rules—it **relies entirely on the MKSTORM + legality checker for grounding**. This enforces strict separation between language generation and logical state, enabling explainability, traceability, and non-hallucinatory operation.

  Additionally, the interface may include a reactive subsystem for **automatic error handling**:

  - When an illegal move is attempted, the model can be returned a structured diagnostic (e.g., "Invalid: knight cannot move to occupied square").
  - When the model requests unavailable context, the system may prefetch relevant information (e.g., past moves, checks, threats) to assist reasoning.

  This tight coupling of logic interface and model agent enables a fundamentally new class of language-aligned, rules-grounded AI capable of operating in structured, deterministic domains without relying on internal world modeling or hand-crafted heuristics.

- **Synthetic training data generator** – This subsystem leverages the deterministic rule logic of MKSTORM, combined with the move legality checker, to exhaustively generate legal sequences of gameplay. These sequences are used to train a domain-specific language model, resulting in an AI that operates with zero reliance on scraped human data or historical bias.

  For chess, the generator explores valid move trees from any arbitrary board state. Because the rules are encoded declaratively and verified in MKSTORM, no external engine (e.g., Stockfish) is needed for legality enforcement. Each generated game includes:

  - A complete move history,
  - Board states at every timestep,
  - Optional annotations (e.g., check, mate, promotion),
  - Embedded causality chains (e.g., "knight fork leads to loss of rook").

  When deployed on hardware accelerators (FPGA/ASIC), the generator can explore and validate vast branches of the move space in parallel. This creates a throughput rate orders of magnitude beyond what general-purpose CPUs can provide.

Performance Conjecture (Chess Domain)

Assuming a 64-core FPGA executing 128-bit entropy block transitions at a conservative rate of 100,000 legal moves per second:

  - In **10 seconds** – 1 million full move-pairs (plies), generating ∼50,000 unique mini-games
  - In **1 hour** – 360 million moves; ∼20 million full game trajectories
  - In **24 hours** – 8.6 billion plies; >500 million legal, annotated chess games

  This data is fed into a distillation pipeline which compresses and formats it for LLM training. The pipeline can:

  - Convert move sequences into structured tokens,
  - Annotate with outcomes, branching evaluations, and strategic markers,
  - Apply filters to extract themes (e.g., sacrifices, forks, rapid mates),
  - Create fine-tuning datasets for transformer-based models (e.g., Axon, GPT-2, LLaMA).

  Because the system does not rely on human demonstration or labeled data, it can operate indefinitely, generating **billions of clean, high-integrity examples** to refine the model over time. Models trained on this data inherit a deep, rule-consistent understanding of the game, enabling high-quality reasoning without hallucination or bias.

  This process enables not only self-training but the creation of entire **domain-specific intelligence agents**, optimized for embedded deployment, microcontroller inference, or tactical simulation environments where trust and reproducibility are essential.

4.2 Aviation Application

The architecture generalizes seamlessly to aerospace systems, where reasoning must be tightly coupled to physical laws and operational constraints.

In this application:

- A digital twin of an aircraft is encoded in MKSTORM, capturing system state variables such as velocity, altitude, orientation, control surface positions, and fuel load.
- The move legality checker enforces constraints derived from flight physics, mission rules, safety thresholds, and hardware limitations. For example:
  - Bank angles must not exceed safe tolerances,
  - Certain maneuvers are invalid below a minimum altitude,
  - Control sequences must be rate-limited based on actuator response curves.
- The LLM interface allows mission planners, operators, or onboard AI to query valid trajectories, simulate maneuver outcomes, and explain operational conditions in natural language.
- The synthetic data generator can produce millions of plausible flight paths under varying mission profiles and constraints—enabling self-training of specialized copilots, mission optimizers, or emergency responders.

Performance Conjecture (Aviation Domain): Assuming a 64-core FPGA or ASIC implementation, the system could evaluate:

- ~1 million microtrajectories per second,
- >100 million plausible events per hour,
- Multiple mission profiles, equipment failure modes, or contingency plans per second.

This data would be formatted for training domain-specific LLMs capable of:

- Interpreting real-time telemetry,
- Predicting safe and efficient control sequences,
- Explaining operational state to human pilots,
- Reacting to anomalies with grounded, rule-consistent responses.

The same pattern of deterministic state, rule-based legality, and LLM reasoning can be extended to any embedded aviation application requiring high trust, real-time validation, and explainable logic under mission-critical constraints.

5 Resilience and Self-Adaptation

One of the key innovations enabled by this architecture is the capacity for autonomous agents to not only operate within constrained environments, but to rapidly adapt when those constraints change unexpectedly. Because the system is not trained statically, but instead retains the capacity to simulate and retrain on demand, it allows for robust behavior in the face of novel, potentially catastrophic changes.

Case Study: Mid-Flight Structural Damage

Consider an AI-controlled aircraft equipped with this system. If a structural event occurs—such as the loss of part of a wing due to impact or hostile fire—the system can:

1. **Capture the anomaly in MKSTORM:** The moment of damage is logged with a 128-bit entropy-stamped timestamp. Sensor data is recorded with microsecond precision, anchoring the event in a verifiable temporal context.
2. **Update physical constraints:** The legality checker adjusts the rule set dynamically, disabling any control surfaces or flight maneuvers now deemed unsafe (e.g., tight banking turns, aggressive ascents).
3. **Generate synthetic trajectories:** The simulator begins exploring valid flight paths that conform to the new aerodynamic regime. These simulations include potential recovery arcs, glide paths, or emergency landing scenarios.

4. **Train a new micro-model:** A domain-specific model is distilled in real time from the legal trajectories, producing a tuned control agent specifically adapted to the damaged vehicle's altered capabilities.
5. **Deploy the updated LLM interface:** The new model assumes control, executing validated maneuvers and optionally narrating its reasoning for pilot or supervisory systems (e.g., "right bank reduced to 10° due to asymmetric lift profile").

Advantages

- **Zero human intervention:** No pilot or operator is required to reconfigure the system.
- **Physics-constrained behavior:** All recovery options are verified against the updated legality rules, preventing overcorrection or runaway responses.
- **Mission continuity:** In some cases, the updated agent may not only recover flight but continue executing the mission profile within acceptable tolerances.
- **Forensic replay:** After resolution, all steps—damage, decisions, outcomes—are stored for post-flight analysis and dataset improvement.

This level of resilience and self-adaptation transforms autonomous systems from brittle to antifragile. Instead of failure, unexpected inputs become catalysts for rapid self-improvement—grounded always in physical truth and verifiable logic.

6 Generalized Application

While the initial use case of this invention is framed in the domain of chess, the underlying architecture is designed to generalize across any system where:

- The state of the world can be deterministically encoded,
- The evolution of that state is governed by formal rules,
- Decisions must be made in compliance with those rules,
- Explainability, trust, and self-adaptation are essential.

By separating the generative capabilities of a language model from the deterministic logic of a state engine, this system enables domain-specific intelligence in areas traditionally resistant to LLM deployment due to the risk of hallucination or lack of formal grounding.

The architecture comprises three core elements:

1. A high-integrity state database (MKSTORM),
2. A legality checker enforcing domain constraints,
3. A reasoning layer (LLM or agent) that queries the system but never overrides its constraints.

This enables deployment across a broad set of domains:

1. Industrial Control Systems

Manufacturing lines, robotic arms, power grids, and automated logistics systems must respond to real-time inputs while obeying physical, safety, and scheduling constraints. MKSTORM can encode:

- Sensor readings and actuator states over time,
- Safety boundaries and interlock logic,
- Task schedules and dependency graphs.

The LLM interface allows for natural-language querying of plant status, simulation of control sequences, and generation of optimized action plans grounded in physical feasibility.

2. Financial Transaction Systems

Modern financial systems—such as banks, payment networks, and clearing houses—process vast numbers of stateful transactions governed by legal, regulatory, and institutional rules. MKSTORM can encode:

- Ledger and account state,
- Transaction constraints (e.g., authorization rules, overdraft limits, compliance flags),
- Immutable audit trails with verified chronological ordering.

Paired with a legality checker and an LLM interface, the system enables:

- Real-time validation of proposed transactions,
- Detection of policy violations or anomalous behavior,
- Autonomous simulation of new regulations, economic scenarios, or internal policy shifts.

3. Medical and Life Sciences

Complex treatment planning, drug interactions, and clinical protocols involve rule-based decision-making across vast state spaces. This system can model:

- Patient state trajectories (vitals, medications, diagnoses),
- Treatment protocols encoded as conditional state transitions,
- Regulatory compliance and contraindication logic.

The LLM interface allows for:

- Patient-specific query and explanation,
- Protocol optimization under real-time constraints,
- Simulation of treatment plans and edge cases,
- Automated documentation of decisions and rationale.

4. Autonomous Vehicles and Robotics

Mobile agents operating in the real world must adhere to kinematic constraints, safety rules, and mission logic. MKSTORM enables:

- Spatial-temporal encoding of position, velocity, and intent,
- Zone-based legality constraints (e.g., no-fly zones, occupancy maps),
- Dynamic rule updates based on environmental feedback.

LLMs trained on legal movement trajectories can generate high-level plans (e.g., "deliver to site B avoiding congestion") while the legality checker enforces route validity and safety compliance.

5. Regulatory Compliance and Audit

Enterprises subject to evolving regulatory regimes must continuously validate decisions against changing legal landscapes. MKSTORM can encode:

- Internal policies and decision records,
- Formal rule changes and legal updates,
- Causal chains from decisions to outcomes.

The LLM interface enables:

- Retroactive analysis (e.g., "what decisions violated Rule 9 after 2023?"),
- Proactive simulation ("what happens if regulation X passes?"),
- Natural-language compliance reporting.

6. Education and Training Systems

Simulations for skill development and interactive learning benefit from deterministic environments with embedded feedback. MKSTORM can encode:

- Learner state, task progress, and knowledge vectors,
- Rule-based content progression and adaptive paths,
- Immutable records of interaction for personalization and audit.

7 Governance and Public Integrity

1. Immutable Accountability Infrastructure

MKSTORM enables the encoding of government actions, decisions, and transactions into a tamper-proof, time-ordered ledger. Every state transition—be it a policy enactment, fund disbursement, legal ruling, or administrative order—can be:

- Recorded with a 128-bit entropy-aligned timestamp,
- Verified against constitutional and legal constraints,
- Stored in a public instantiation of the MKSTORM database for auditability.

This provides a high-integrity system where no record can be erased, redacted, or modified without detection, restoring public trust in institutional operations.

2. Rule-Constrained Decision Enforcement

MKSTORM integrates a legality checking layer that can encode constitutional rights and statutory frameworks as logic constraints. Proposed actions by state entities (e.g., surveillance orders, budget reallocations, executive directives) can be automatically evaluated:

- If they violate enumerated rights (e.g., due process, free speech, equal protection), they are blocked or flagged.
- If compliant, the system issues a signed resolution record confirming legal conformance.

This allows governments to be operated with software-enforced constitutionalism, dramatically reducing the potential for illegal, corrupt, or authoritarian behavior.

3. LLM-Based Watchdog Agents

Specialized language models trained on:

- Constitutional law,
- Historical abuses of power,
- Policy precedents,

can act as automated integrity monitors. These agents can:

- Issue real-time alerts on questionable actions,
- Annotate MKSTORM records with legal context,
- Respond to public queries about government behavior in human-readable form.

This replaces passive transparency with active accountability.

4. Public Transparency and Civic Access

Citizen interfaces can be built on top of MKSTORM's queryable, cryptographically verified state. These interfaces allow:

- Voters to ask natural-language questions (e.g., "Where did federal COVID relief funds go in Texas?"),
- Journalists to simulate consequences of pending legislation,
- Whistleblowers to submit signed integrity-checking triggers without compromising anonymity.

Public trust is restored by making state logic and decision pathways fully explorable.

5. Anti-Censorship and Regime Resilience

MKSTORM's distributed, entropy-anchored database ensures that verified governmental records:

- Cannot be deleted by future administrations,
- Remain cryptographically provable even if access is censored,
- Can survive hostile takeovers or institutional collapse.

This transforms the rule of law from a political agreement into a digital invariant.

AI Governance

This invention enables a radically transparent and resilient model of governance, where human institutions are bounded by code, protected by entropy, and overseen by tireless AI watchdogs. In an era of democratic backsliding and rising authoritarianism, MKSTORM offers an architecture for incorruptible institutional memory and lawful, constrained governance.

The LLM acts as a tutor, guide, or analyst—grounded not in open-ended dialog but in formalized progressions and validated pedagogical structures.

Key Property Across All Domains

In each application, the separation of deterministic state and generative reasoning ensures:

- No hallucination or rule-breaking behavior,
- Verifiability and auditability of decisions,
- The capacity to self-train using domain-specific legal trajectories,
- Adaptability in the face of novel inputs or evolving constraints.

This architecture supports the emergence of mission-critical, domain-specialized AI that is not only capable of reasoning—but doing so in ways that are always explainable, safe, and formally correct.

Architectural Consistency

Each of these domains benefits from the same modular pattern:

- **MKSTORM:** Immutable, deterministic record of system state.
- **Legality Checker:** Rule set enforcing valid actions/transitions.
- **LLM Interface:** Translator between logic and natural language or decision planning.
- **Synthetic Training:** Generation of legal, diverse, domain-specific data to improve autonomous models.

The core advantage is that **each domain becomes self-describing and self-training**, requiring no external data or hand-labeled corpora. New fields can be onboarded by simply encoding their rule structures and state descriptions—at which point the system begins reasoning and refining behavior on its own.

Key Benefits Across Domains

- **Modular Deployment:** The architecture separates state, rules, and reasoning, enabling independent updates and targeted optimization.
- **Explainability:** Every decision can be traced to a legal move grounded in domain logic and time-stamped state.
- **Hardware Optimization:** FPGAs or ASICs can be customized to accelerate legality checking and data generation, maximizing throughput for real-time or large-scale deployments.
- **Resilient AI Behavior:** Agents trained in this system learn to act only within allowed spaces—making them safer, more predictable, and better suited for deployment in regulated or high-stakes environments.

8 Mitigation of Language Model Hallucination

This invention provides a structural mechanism for eliminating hallucinated outputs common in large language models. By integrating a deterministic, rule-constrained state engine (MKSTORM), the system ensures that any generative output made by the language model must pass through a verification layer grounded in the domain's formal logic. This provides a factual substrate that fundamentally alters the statistical behavior of the model, replacing freeform token prediction with constrained, state-aware decision making.

9 Claims (Provisional)

1. A method for encoding and updating a system's state in a secure, verifiable database.
2. A system wherein a language model queries this database to obtain valid states and make decisions.
3. A process for generating training data via the replay and mutation of legal state transitions.
4. A hybrid AI architecture combining symbolic rule logic and statistical generation.

10 Conclusion

MKULTRA represents a unification of logic-driven state machines and probabilistic reasoning systems. Through deterministic grounding, synthetic training, and autonomous state coupling, this architecture offers a powerful solution to hallucination, resilience, and long-term agent trust across domains as diverse as gaming, aviation, and governance.