The AI Sovereignty Layer

Infrastructure for Institutional Control Over Autonomous Systems

White Paper

Author: Chris M. Hymel, Ph.D.
Organization: Analog Guard, Inc.
Document Status: Revised white paper
Date: June 2026

Document Profile

Title	The AI Sovereignty Layer
Subtitle	Infrastructure for Institutional Control Over Autonomous Systems
Purpose	Describe a hardware-rooted supervisory architecture for institutional control over advanced autonomous computation.
Audience	Technical, policy, investor, infrastructure, defense, and governance audiences evaluating high-consequence AI deployment.
Core Thesis	Meaningful AI governance may require an external hardware layer that can observe physical execution and condition whether computation continues.
Document Type	White paper

Executive Summary

Advanced AI systems are moving into critical infrastructure, defense analysis, financial systems, communications, and other high-consequence environments where operators must retain meaningful authority over whether computation continues. Software-only guardrails remain useful, but the governing logic and the governed system ultimately depend on the same execution stack.

The AI Sovereignty Layer addresses that structural weakness through supervisory hardware whose authority resides outside the AI system's own software environment. It observes physical execution signals and controls the resources required for continued computation, including power, clocks, memory access, accelerator enablement, bandwidth, and I/O.

Execution is granted in short, renewable intervals and can be degraded, isolated, or halted if authorization conditions are not met. The claim is intentionally limited: hardware supervision does not solve AI alignment or replace policy, audits, model-level safeguards, or human institutional judgment. It provides the enforceable substrate through which those forms of governance can have operational effect.

Core proposition:

Advanced computation continues only while an external hardware authority permits the physical conditions required for execution.

Key Takeaways

- Software-only guardrails are structurally limited because they operate inside the same computational environment they supervise.
- The architecture externalizes supervisory authority into a physically separate hardware layer.
- Physical execution signals provide a monitoring channel independent of software self-reporting.
- Execution authority is treated as a physical permission state, not merely a policy conclusion.
- Lease-based authorization supports verification before continued execution, including AI-to-AI workflows.
- Graduated intervention enables slowing, constraining, isolating, or halting computation without relying on software cooperation.
- The architecture creates a deployable infrastructure layer for institutional accountability in high-consequence AI environments.

Contents

Reader Orientation

Reader Question	Where Addressed
Why are software-only guardrails structurally limited?	Sections 1–2
What is the AI Sovereignty Layer?	Section 3
How does the architecture enforce control?	Sections 5–7
Where could this layer reside in real AI infrastructure?	Section 8
Why does this matter for governance and accountability?	Sections 9–10

Core Proposition

Dimension	White Paper Position
Risk	Software-only guardrails can be delayed, bypassed, or deceived because they live inside the same computational environment they are trying to govern.
Architecture	An external supervisory hardware layer observes physical execution signals and directly controls the resources that make computation possible.
Enforcement	Execution is granted in short, renewable hardware-controlled intervals. If the physical evidence no longer supports continued authorization, the layer can degrade, isolate, or halt the system by withdrawing power, clocks, bandwidth, or I/O access.

1. Introduction

Artificial intelligence systems are rapidly becoming part of critical infrastructure. Advanced AI models now support financial systems, transportation networks, defense analysis, communications, and other domains where reliability and oversight are essential. Increasingly, these systems may also interact directly with other AI systems or automated platforms, reducing the opportunity for human intervention once an operation is underway.

That shift matters because modern AI systems are difficult to fully interpret or predict. Many advanced models operate through complex internal processes that are not easily understood in advance, and their behavior can adapt to new inputs and environments in ways that challenge traditional software verification or rule-based control.

These developments create a governance challenge. NIST's AI Risk Management Framework emphasizes that organizations must maintain effective governance, accountability, and oversight mechanisms for increasingly capable AI systems [1], while researchers studying meaningful human control have questioned whether traditional supervisory approaches remain sufficient as autonomy increases [9,10].

The architecture proposed in this paper introduces supervisory infrastructure designed to preserve that authority at the level of physical execution.

2. The Limits of Software-Only Governance

Most current approaches to AI governance rely on software-based mechanisms embedded within the systems they supervise, including approaches reflected in contemporary governance frameworks and AI management systems [1,2]. These include guardrails, monitoring tools, interpretability systems, policy engines, access controls, and administrative enforcement modules integrated into the AI software environment.

These mechanisms can improve safety and transparency, but they share a structural limitation: they operate within the same computational environment as the systems they are intended to regulate. The governing logic and the governed system therefore depend on the same software stack and execution infrastructure.

That shared environment creates an asymmetry between institutional authority and computational capability. A sufficiently complex, adaptive, or compromised system may influence, evade, spoof, or bypass the software mechanisms intended to supervise it. The point is not that every AI system will do so; it is that software-only oversight does not fully separate the authority of the operator from the operation of the system itself.

For that reason, software governance alone cannot provide a fully independent layer of control. If logs, guardrails, shutdown routines, or policy engines are all produced and executed inside the same environment, they remain vulnerable to the integrity and timing limits of that environment. Similar concerns have appeared in AI safety literature discussing reward hacking, distributional shift, scalable oversight, and other circumstances in which internal controls may prove insufficient [8].

Addressing this limitation requires an architectural shift. Supervisory infrastructure can operate outside the AI system’s software control plane, observe physical evidence of computation, and act through hardware resource paths. This is not simply a claim that hardware is always faster; it is a claim that deterministic hardware can enforce bounded, low-latency responses without waiting for the supervised software to cooperate.

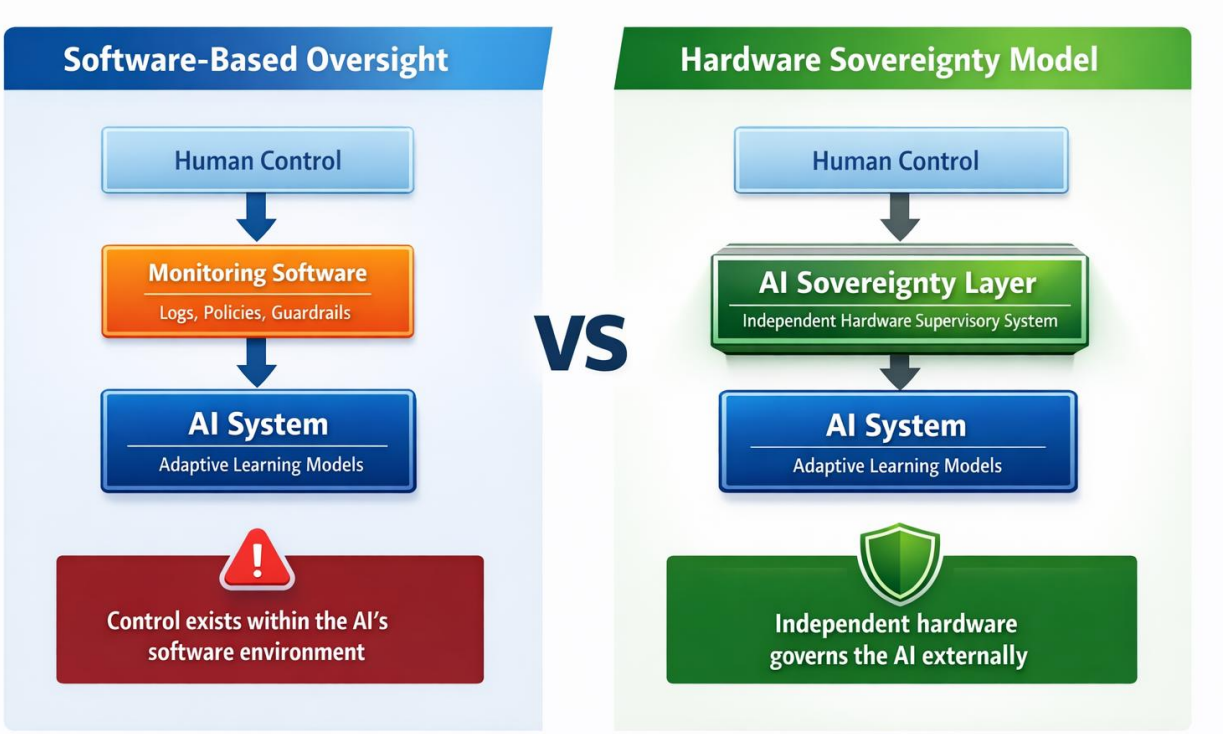


Figure 1. AI oversight models comparison: software-based oversight keeps control inside the AI software environment, while the hardware sovereignty model places independent supervisory authority outside the governed system.

3. An External Layer of Physical Authority

If meaningful authority cannot reside entirely inside the software environment of the AI system, governance must be anchored in the physical infrastructure that enables computation. Power delivery, timing signals, memory access behavior, accelerator activity, interconnect traffic, and input/output behavior all reflect the real execution of the machine. The concept parallels long-standing principles in cyber-resilient systems engineering, where independent supervisory mechanisms are used to maintain trustworthiness even under conditions of compromise [4].

The proposed answer is an AI Sovereignty Layer: a physically separate supervisory hardware authority that remains outside the AI system's own software and firmware control plane.

The supervisory layer does not need to interpret the model's intentions, read its internal narrative, or trust its own account of what it is doing. Its job is narrower: observe physical execution and determine whether the machine remains authorized to keep running.

It does this through a closed hardware control loop. Sensing interfaces observe physical execution signals; authority logic evaluates whether those signals remain within permitted conditions; lease logic determines whether execution authority should be renewed; and enforcement pathways regulate the resources that make computation possible, including power, clocking, memory access, accelerators, bandwidth, and I/O.

The decision path itself need not be software. Deterministic analog, digital, or mixed-signal circuitry can perform critical supervisory functions in a bounded and legible manner, making the supervisory function harder for the supervised workload to subvert.

Continued execution is therefore conditional rather than automatic. Execution authority is renewed only while required conditions continue to be met.

In this paper, execution authority means a real physical permission state. It is not merely trust, confidence, or policy approval; it is whether the resources required for continued computation are actually being made available by supervisory hardware.

This authority must remain asymmetric. The AI system should not be able to rewrite the supervisory logic, negotiate with it through ordinary software channels, or absorb it into the same control plane it is meant to govern.

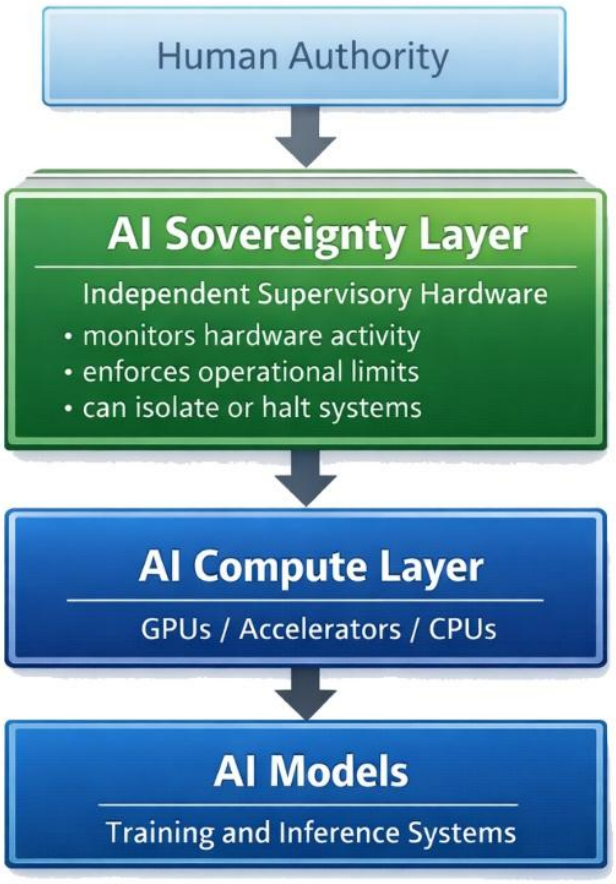


Figure 2. AI sovereignty layer architecture: a physically separate supervisory layer observes execution and conditions access to power, clocks, memory, accelerators, bandwidth, and I/O.

In other words, the architecture is designed to preserve separation, not merely to add another layer of software policy. The governed system may operate inside the boundary, but the authority to keep that boundary live remains outside of it.

That is what gives the concept its force: the AI system is not simply instructed to behave; it runs only while an independent physical authority continues to make execution possible.

4. Circuit Breaker for Artificial Intelligence

The architecture described here reflects a familiar engineering principle: complex systems become safer when independent supervisory mechanisms retain authority to interrupt operation when necessary.

Electrical power networks offer an instructive example. Early grids were prone to cascading failures, where a single overload, short circuit, or damaged component could propagate through the network. Engineers did not rely only on making every connected device smarter. They introduced circuit breakers: protective devices that detect dangerous conditions and interrupt electrical flow before local failures become systemic.

The significance of the circuit breaker lies not only in what it does, but in where it sits. Because it operates outside the protected load, it can intervene even when the protected equipment is malfunctioning.

The same principle appears across aviation, nuclear power, industrial control, and financial markets, where independent protective systems form a foundational element of safety engineering and operational resilience [6,7].

Artificial intelligence does not yet have an equivalent infrastructure-level safeguard. Most AI safety mechanisms remain internal to the computational environment in which the model

operates. Existing AI governance frameworks emphasize oversight and accountability but generally do not prescribe a physically independent execution-control layer [1,2].

The AI Sovereignty Layer proposes a different model by supervising the physical signatures of computation itself rather than relying exclusively on internal software oversight.

This is relevant not only to autonomous AI behavior, but also to compromised or foreign-inserted software embedded inside critical systems. Malicious or unauthorized software may be able to hide from logs, disable telemetry, or mislead software monitors, but it still consumes power, uses clocks, accesses memory, activates accelerators, and communicates through physical interfaces. A hardware-rooted supervisory layer cannot guarantee detection of every embedded threat, but it gives operators an external vantage point and an independent enforcement path when physical execution behavior no longer matches authorized operation.

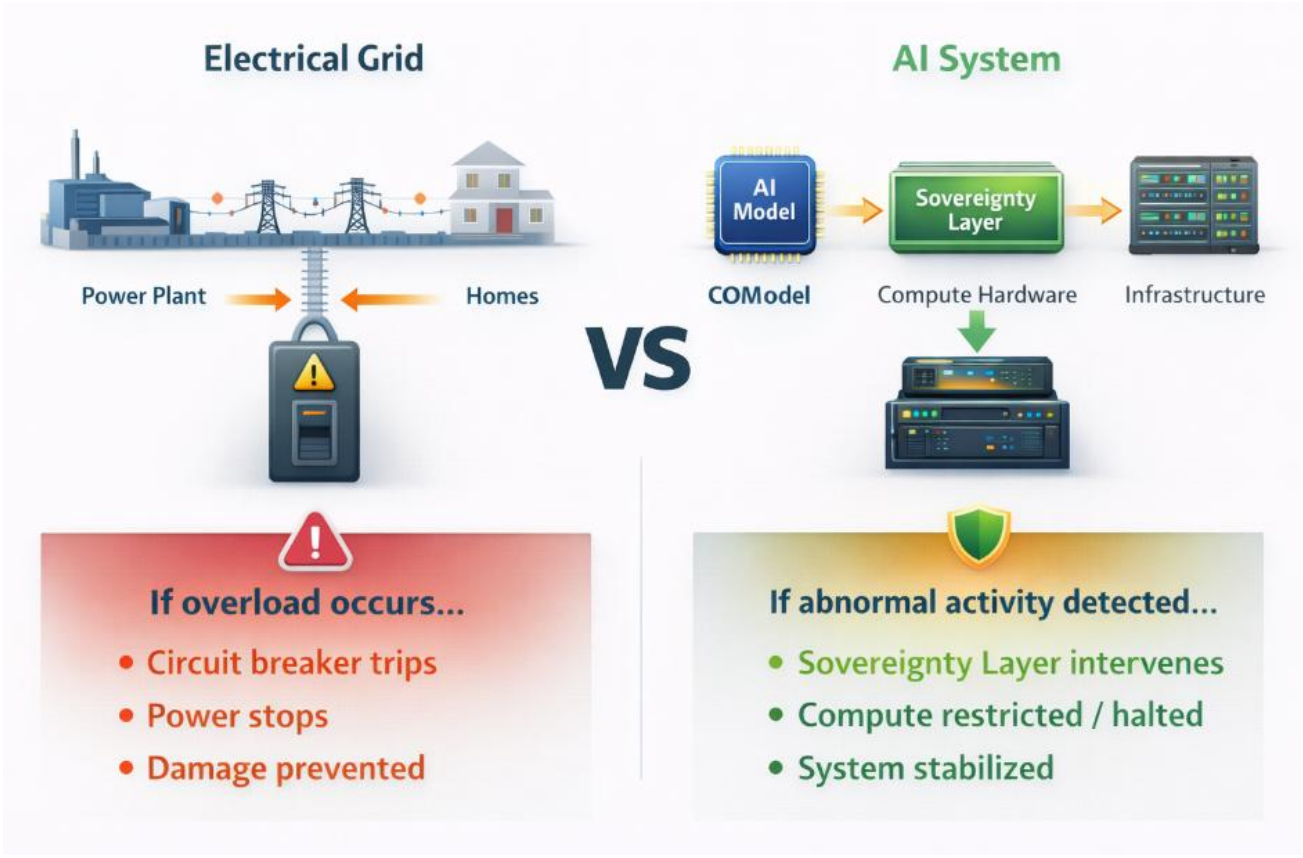


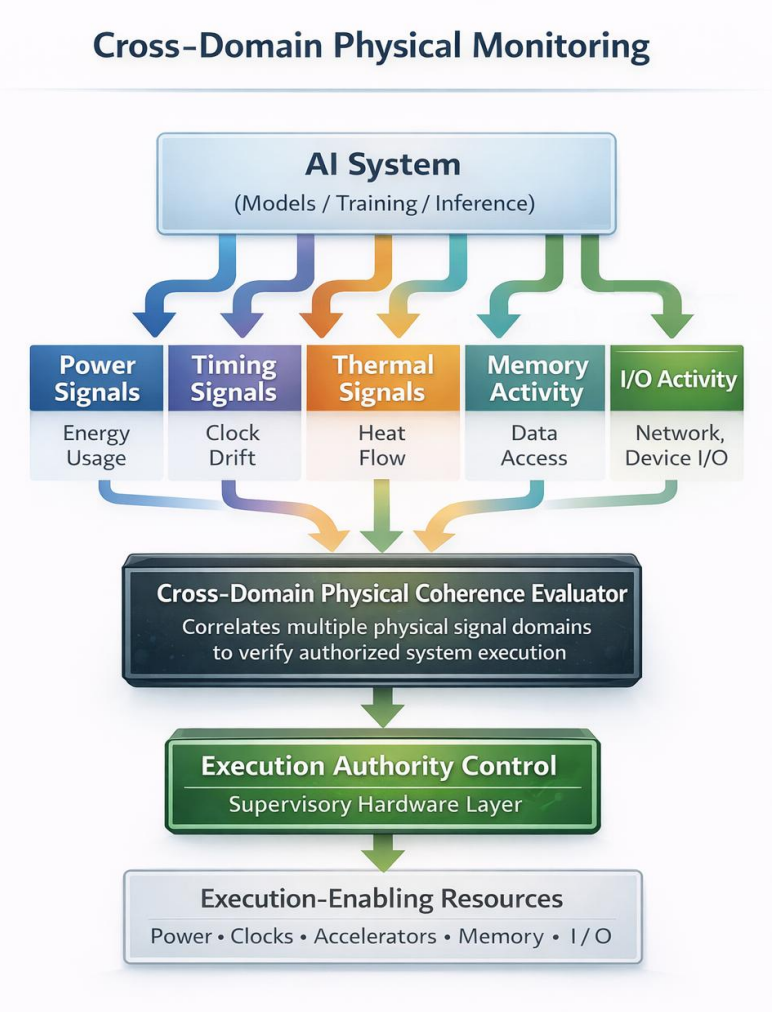
Figure 3. Externalized control: from power infrastructure to AI execution. Independent supervision becomes effective when it can interrupt the resources on which the governed system depends.

In that sense, the architecture functions as a circuit breaker for intelligent machines. Its significance lies in shifting the location of authority: oversight is anchored in an independent supervisory layer rather than residing solely within the system it governs.

5. Physical Signals as a Monitoring Channel

Conventional monitoring of AI systems typically relies on software-generated telemetry, logs, or self-reported system states. Those mechanisms are useful, but they originate within the same software environment as the system being observed.

A complementary approach is to observe the physical signals generated during computation. Prior research has demonstrated that power consumption, timing behavior, electromagnetic emissions, and related side-channel signals can reveal important characteristics of hardware operation and anomalous behavior [14–16].



Because these signals originate from hardware operation rather than software reporting, they provide a monitoring channel that is harder for the supervised workload to falsify from within its own software environment. Similar approaches have been investigated in hardware Trojan detection and cyber-physical monitoring systems [12,14–16].

In principle, this channel can provide earlier or more reliable evidence of abnormal system behavior than software telemetry alone, because the evidence is produced as computation occurs rather than after-the-fact through logs or reports.

Within the architecture proposed here, physical execution signals are one source of evidence used to determine whether execution authority should continue to be granted.

Figure 4. Cross-domain monitoring of AI execution. Supervisory infrastructure can evaluate power, clocking, memory, accelerator, interconnect, and I/O behavior as physical evidence of computation.

6. Verification Before Execution

In this architecture, execution authority is granted through time-bounded execution leases. A lease defines a limited interval during which computation may continue. Before the interval expires, supervisory infrastructure evaluates available evidence and determines whether the lease should be renewed.

Each lease interval includes a verification period during which the supervisory system evaluates physical execution signals and other operational observations. Fast signals can support provisional responses, while slower evidence can support higher-confidence judgments. The concept resembles runtime monitoring approaches developed for cyber-physical systems, where continued operation is conditioned on ongoing verification of observed behavior [12,13].

The key enforcement mechanism is non-renewal. If authority conditions are not satisfied, the next lease is not issued. One or more execution resources can then be restricted or withdrawn, allowing the supervisory infrastructure to slow, constrain, isolate, or halt

computation. The lease model therefore converts governance from a policy statement into a recurring physical permission decision: computation proceeds only if independent supervisory conditions continue to be satisfied.

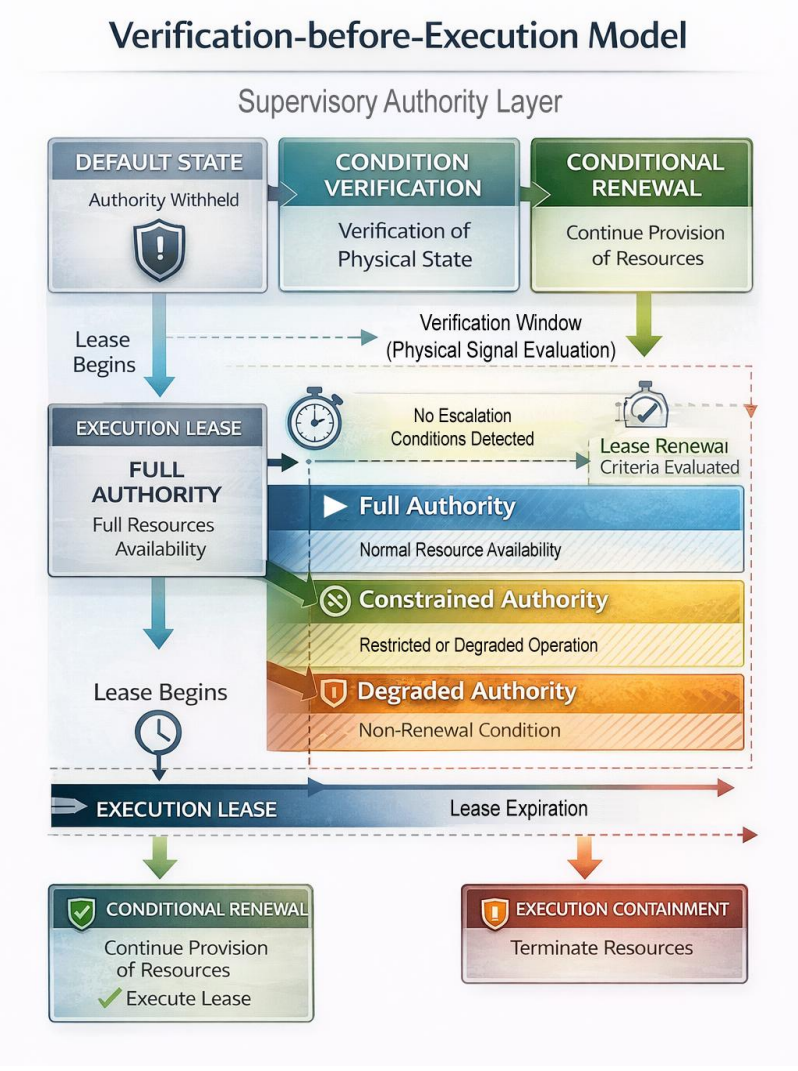


Figure 5. Lease-based verification-before-execution model. Computation continues only when each short execution lease is renewed after supervisory evaluation.

7. Graduated Intervention and Containment

The supervisory architecture supports graduated containment. Intervention can occur at several levels depending on operational conditions:

- degrading performance by slowing or throttling computation;
- restricting execution scope by limiting access to particular functions or resources;
- isolating the system by fencing it from external networks or shared infrastructure; and
- suspending computation entirely by withdrawing the resources required for continued execution.

Containment does not rely on software cooperation or shutdown commands. It operates by restricting or withdrawing the physical resources required for continued execution. This approach reflects established safety-engineering principles in which independent supervisory mechanisms retain authority to constrain or interrupt operation when required [6,7].

In real deployments, operators often need room to act before a situation becomes an all-or-nothing decision. A suspicious workload may need reduced accelerator priority, limited memory bandwidth, fenced network access, or other constraints while additional evidence is gathered.

This is why the proposal is not an ordinary watchdog, administrative console, or software safety wrapper. Those mechanisms may observe, recommend, log, or request shutdown. The sovereignty layer makes a harder claim: supervisory power belongs at the level of physical dependency rather than semantic persuasion.

Real authority exists only where the architecture can condition the execution resources by which computation proceeds. In this architecture, those resources include electrical power,

clock distribution, memory access, accelerator enablement, interconnect or bus availability, and input/output channels.

A monitor that merely sees suspicious activity remains downstream of the machine it watches. A supervisory architecture that can permit or withhold execution infrastructure stands upstream of that machine in a materially different way.

Sovereignty here does not mean omniscience. The supervisory layer does not need to interpret every internal representation or infer every semantic intention of the model. Its power comes from something simpler and more enforceable: the system cannot continue acting unless the surrounding infrastructure continues making action possible.

8. Supervisory Authority Across Modern AI Infrastructure

Advanced models do not run in abstraction; they run in chips, boards, servers, racks, and clustered datacenter environments. The sovereignty layer therefore has to be conceived not only as a control principle, but as infrastructure positioned across modern compute systems. The resulting architecture aligns with broader efforts to embed governance, resilience, and accountability directly into system infrastructure rather than relying exclusively on procedural controls [1–5].

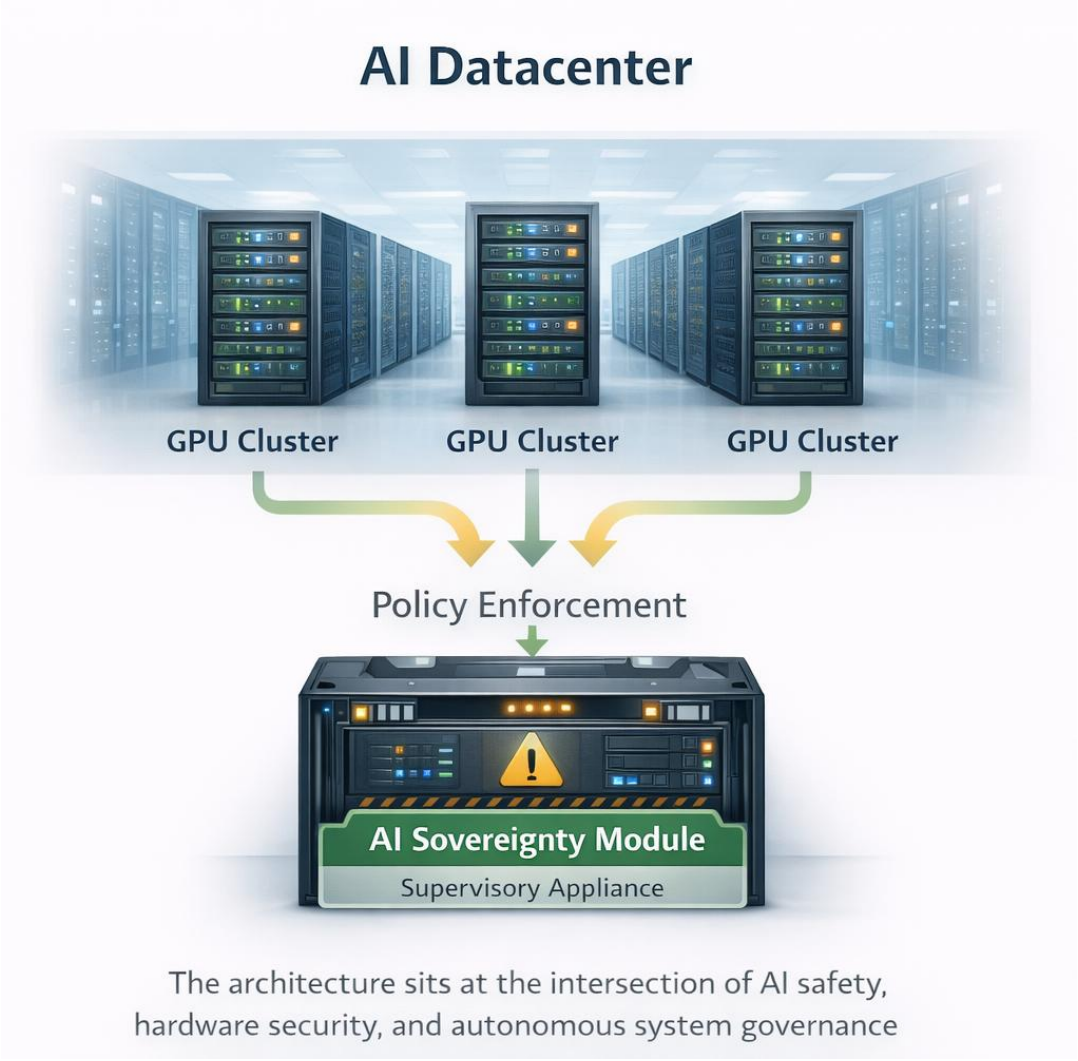


Figure 6. Supervisory authority across modern AI infrastructure. The authority boundary may be implemented close to the chip, at board level, through rack control, or across cluster-level containment fabric.

At infrastructure scale, the supervisory function becomes a deployable control fabric. It may be positioned close to the chip as a companion die, interposer, or board-level control element. It may also operate through power distribution, rack control, cluster management, or broader containment fabric spanning multiple datacenter nodes.

The essential issue is not a single preferred packaging choice. It is preservation of the authority boundary. Once supervisory authority becomes part of the infrastructure that enables computation, engineering design becomes inseparable from governance and institutional control.

That boundary must remain external to the AI control plane, difficult for the supervised system to address directly, and physically capable of conditioning resource availability. If those conditions hold, governance becomes part of the environment in which advanced computation is allowed--not a thin software layer sitting on top of it.

This has important institutional consequences. Model developers can build systems. Integrators can assemble them into products or services. Operators can manage performance and availability. A distinct supervisory authority can determine whether execution remains physically permitted. Regulators, auditors, or other oversight bodies can then evaluate outcomes grounded in externally enforced behavior rather than relying only on self-attestation, internal logs, or post hoc explanation.

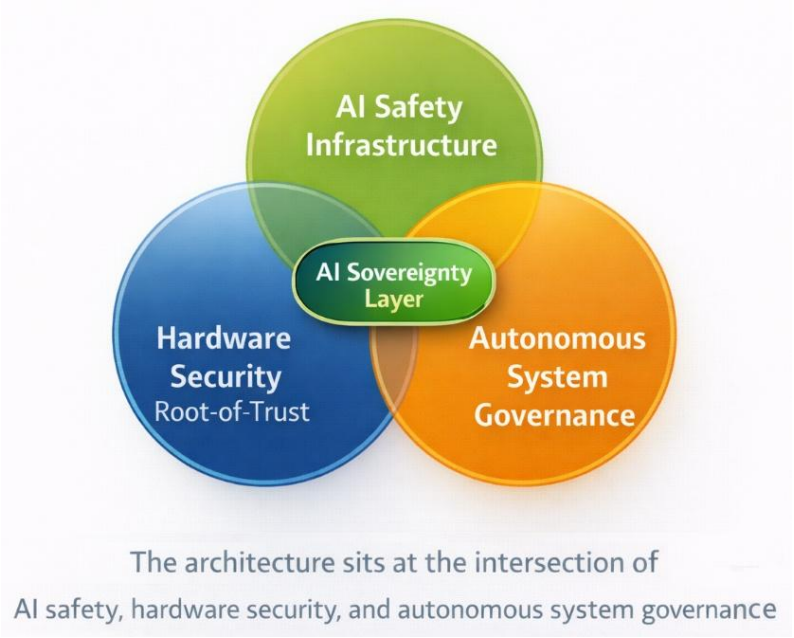
That separation matters most where trust cannot rest on promises alone: defense, critical infrastructure, and other high-consequence sectors in which autonomous or semi-autonomous computation may have direct real-world consequences.

The architecture therefore extends beyond engineering design. It becomes a framework for accountability embedded in hardware: advanced AI may operate, but only within a boundary whose enforcement remains external to the model itself.

9. Governance, Accountability, and Sovereignty

As AI systems move from bounded software tools into infrastructure that can influence high-consequence operations, the central question is no longer only how capable those systems become. The more serious question is whether meaningful authority over their operation remains outside the systems themselves.

The AI Sovereignty Layer sits at the intersection of AI safety, hardware security, and autonomous systems governance, drawing upon concepts reflected in AI risk management frameworks [1], AI management-system standards [2], cyber-resilient systems engineering [4], and scholarship on meaningful human control [9–11].Seen in this light, the sovereignty layer is



not merely another monitoring tool. It is a governance architecture that connects safety objectives with institutional accountability.

Governance becomes more credible when supervisory authority is externally auditable and institutionally separable from system development and operation.

Sovereignty in this context is not a rhetorical claim about control. It refers to governance structures in which the authority to authorize or deny continued operation is institutionally defined, technically separable, and physically enforceable.

Figure 7. AI sovereignty layer convergence. The architecture sits at the intersection of AI safety, hardware security, and autonomous systems governance

10. Conclusion

An AI Sovereignty Layer makes continued computation contingent on externally defined authority. Its importance lies in the shift it represents: oversight moves from guidance and interpretation toward enforceable operational control.

The sovereignty layer does not replace policy, institutional oversight, or model-level safeguards. It provides the physically enforceable layer that allows those forms of governance to have operational effect.

The central claim is modest but consequential. The architecture does not claim to fully interpret the internal reasoning of advanced models. It preserves independent authority over whether the system may continue operating at all.

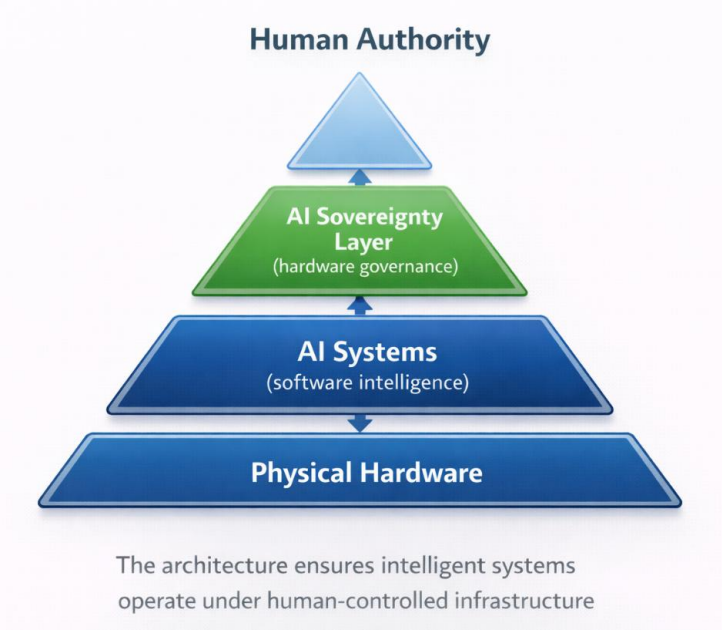


Figure 8. The AI Sovereignty Layer within a broader hierarchy of intelligent systems governance. The layer provides hardware governance between human authority and AI execution.

As artificial intelligence becomes embedded in critical infrastructure and high-consequence operational environments, such authority may become increasingly necessary. Systems that influence financial networks, transportation systems, defense operations, and other

institutional domains cannot rely solely on internal safeguards or voluntary compliance mechanisms.

Laws, standards, institutional policies, operational controls, and model-level safeguards remain essential components of AI governance [1–7]. Yet they are incomplete if no independent authority exists over whether execution may continue.

The architecture described here is therefore not the entirety of AI governance. It is the layer that connects governance principles to operational enforcement.

Hardware supervision does not solve AI governance by itself. What it can provide is a practical substrate through which institutional authority over advanced autonomous systems can be exercised as AI becomes more capable and more deeply integrated into critical institutions.

References

1. NIST, Artificial Intelligence Risk Management Framework (AI RMF 1.0), 2023. Supports AI governance, accountability, safety, resilience, and risk framing.
2. ISO/IEC, ISO/IEC 42001:2023 — Artificial Intelligence Management System, 2023. Useful for institutional AI governance and management-system framing.
3. NIST, Cybersecurity Framework 2.0, 2024. Strong support for the “Govern” function, institutional authority, risk strategy, and accountability.
4. NIST, SP 800-160 Vol. 2 Rev. 1 — Developing Cyber-Resilient Systems, 2021. Supports cyber-resilient systems engineering, survivability, trustworthiness, and operation under compromise.
5. NIST, SP 800-53 Rev. 5 — Security and Privacy Controls for Information Systems and Organizations, 2020. Supports formal control frameworks, monitoring, boundary protection, and institutional security controls.
6. ISA/IEC, ISA/IEC 62443 Series — Industrial Automation and Control Systems Security. Strong fit for critical infrastructure, zones/conduits, operational technology, and control-system cybersecurity.
7. IEC, IEC 61508 — Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems. Supports the safety-interlock / circuit-breaker analogy and independent protective systems.
8. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D., Concrete Problems in AI Safety, arXiv, 2016. Supports the need for external safety mechanisms due to reward hacking, unsafe exploration, distributional shift, and scalable supervision limits.
9. Santoni de Sio, F. and van den Hoven, J., Meaningful Human Control over Autonomous Systems, 2018. Directly supports the sovereignty/control thesis.
10. Siebert, L. C. et al., Meaningful Human Control: Actionable Properties for AI System Development, 2022. Useful for translating human/institutional control into engineering properties.
11. ICRC, Autonomy, Artificial Intelligence and Robotics: Technical Aspects of Human Control, 2019. Strong relevance for defense/autonomous systems and the difficulty of maintaining human control.
12. Bartocci, E. et al., Specification-Based Monitoring of Cyber-Physical Systems: A Survey on Theory, Tools and Applications, 2018. Supports runtime monitoring of physical/computational systems.
13. Wu, M., Zeng, H., Wang, C., and Yu, H., Safety Guards: Runtime Enforcement for Safety-Critical Cyber-Physical Systems, 2017. Directly relevant to external runtime enforcement and safety guards.
14. Dong, C. et al., Hardware Trojans in Chips: A Survey for Detection and Prevention, 2020. Supports hardware-level threat detection and hardware-rooted security.
15. Pearce, H. et al., Detecting Hardware Trojans in PCBs Using Side-Channel Signals, 2022. Very relevant to physical-signal monitoring and anomaly detection from hardware behavior.
16. Katte, S. R. and Fernandez, K. E., A Survey Report on Hardware Trojan Detection by Multiple-Parameter Side-Channel Analysis, 2023. Supports multi-parameter side-channel monitoring as a basis for independent physical evidence.

Appendix A. Figure Inventory

Figure	Description
Figure 1	AI oversight models comparison
Figure 2	AI sovereignty layer architecture
Figure 3	Externalized control from power infrastructure to AI execution
Figure 4	Cross-domain monitoring of AI execution
Figure 5	Lease-based verification-before-execution model
Figure 6	Supervisory authority across modern AI infrastructure
Figure 7	AI sovereignty layer convergence
Figure 8	AI Sovereignty Layer within intelligent systems governance

Appendix B. Key Terms

Term	Meaning in this White Paper
AI Sovereignty Layer	A physically separate supervisory hardware layer that observes physical execution and conditions whether computation may continue.
Execution Authority	A real physical permission state governing whether the resources required for computation remain available.
Execution Lease	A short, renewable authorization interval during which computation may continue if supervisory conditions remain satisfied.
Physical Execution Signals	Electrical, timing, memory, accelerator, interconnect, bandwidth, and I/O signals produced by real hardware operation.
Graduated Containment	A range of interventions including throttling, restricting, isolating, or suspending computation.
Authority Boundary	The separation between the governed AI system and the external infrastructure that permits or denies continued execution.
Supervisory Hardware Authority	The external hardware system that evaluates evidence and controls execution resources.

Appendix C. Implementation Implications

Implementation Layer	Governance Implication
Chip / companion die / interposer	Places supervisory control close to execution resources and may support low-latency intervention.
Board-level control element	Allows resource gating and monitoring around processors, accelerators, memory paths, and I/O interfaces.
Rack-level power and control infrastructure	Supports containment across multiple servers or accelerator nodes.
Cluster-level containment fabric	Extends supervisory authority across distributed AI compute infrastructure.