Interferometric Sector–Gate Nervous Systems: Orchestrating AGI via ParityFuse Quantum Primitives and Deterministic Java Injection

from Java Injection to Quantum Eigenstates: Engineering an n8n-Style Nervous System for AGI

Peter De Ceuster
Addressed to: Dr. Chetan Nayak, Microsoft Azure Quantum

26 September 2025

Abstract

We present a mathematically rigorous architecture that integrates measurement-first parity interferometry ("ParityFuse") with a provable runtime orchestration layer (a capability-constrained Java injection agent) to form an n8n-like nervous system for an AGI substrate encoded in sectoral eigenstate dynamics. We develop (i) the Hilbert-space sector factorization and operator calculus underpinning interferometric sector-gates, (ii) the Tunneling-First optimization theorem for parity-limited readout devices and its asymptotic error-exponent consequences, (iii) open-system stochastic master equations that govern measurement-induced cognitive transitions, (iv) a formal measurement-only compilation mapping from logical sector gates to ParityFuse sequences (including RUS semantics and resource accounting), and (v) a typed Java-injection operational semantics with formal verification obligations ensuring safe, auditable runtime compilation of AGI decisions into bounded-latency quantum control. Empirical device statements and the ParityFuse primitive specification follow the interferometric device blueprints and topological-core optimality arguments in the accompanying technical materials. We will cover a jump operator that might prove to be an important missing link, and explain how an injection agent might be a steppingstone towards AGI.

Contents

1 Introduction

We synthesize two programs: (A) the Sector–Gate / Eigenstate Dynamics view of cognition as dynamics on a factorized Hilbert space HAGI, and (B) a measurement-first hardware substrate whose primitive is an interferometric parity measurement (ParityFuse). The aim is to give an end-to-end formal pipeline from high-level AGI decision to quantum primitive actuation while ensuring safety (capability-limited injection), in boundedness, and verifiable provenance. This document expands the

ISD program and ParityFuse calculus presented in the device blueprints. We will cover an important jump operator and our injection agent, crucial towards developing our platform.

2 Preliminaries and notation

Let $\mathcal{H}_{\mathrm{AGI}} = \bigotimes_{k=1}^{N} \mathcal{H}_k$ be a finite-dimensional (for rigor) Hilbert space factorization into cognitive sectors. Denote by $\hat{H}_k$ the local Hamiltonian on sector k, and by $\hat{V}_{kl}$ cross-sector couplings. The total closed-system Hamiltonian

$$\hat{H}_{\mathrm{AGI}} \;=\; \sum_{k=1}^{N} \hat{H}_k \;+\; \sum_{k\neq l} \hat{V}_{kl}. \tag{2.1}$$

The foregoing suggests that such a system is, in principle, physically realizable under the stated assumptions. In the sections that follow we outline mechanisms to achieve operational contingency and robust control, including explicit fallback modes and provenance guarantees for runtime actions.

We use $\{|\psi_{k,i}\rangle\}_i$ for instantaneous eigenbases of $\hat{H}_k$, and $|\Psi(t)\rangle$ for global pure states. The parity operator on a parity-encoded sector (Majorana / fermion parity) is denoted $\hat{Z} \in \{\pm 1\}$; projection operators $\Pi_{\pm} = \frac{1}{2}(\mathbb{1} \pm \hat{Z})$.

Directional notation: $\mathrm{Tr}_{\neg k}[\cdot]$ denotes partial trace over all sectors except k. It is now essential to study the parity primitive.

3 Interferometric parity primitive (ParityFuse)

3.1 Physical parametrization

The ParityFuse primitive is a single-shot interferometric parity measurement on a triple-dot loop coupled to spatially separated Majorana zero modes (MZMs). Introduce effective left/right tunnel couplings $t_L, t_R \geq 0$, flux Φ with phase $\varphi = 2\pi\Phi/\Phi_0 + \varphi_0$, and define the parity-sensitive effective coupling

$$t_C(Z,\varphi) \;=\; \sqrt{t_L^2 + t_R^2 + 2\,t_L t_R \sin(\varphi Z)}. \tag{3.1}$$

Dispersive readout produces a parity-conditional shift $\Delta C_Q(\Phi) \propto \partial^2_{V_{\mathrm{QD}}} E[t_C(Z,\varphi)]$, giving single-shot SNR scaling

$$\mathrm{SNR}(\tau) \;\propto\; |\Delta C_Q(\Phi)|\sqrt{\tau}, \tag{3.2}$$

for integration time τ (see device measurements and operating points).

3.2 Kraus-model and measurement channel

Let the measurement outcome be $m \in \{+1, -1\}$ corresponding to parity branches. The measurement is modeled by two Kraus operators

$$M_{\pm} \;=\; \sqrt{p(m = \pm 1 \mid Z)}\,\Pi_{\pm}\,U_{\mathrm{read}}(t_C), \tag{3.3}$$

where $U_{\mathrm{read}}(t_C)$ is a unitary describing the readout entangling and $p(m|Z)$ the noisy detector statistics (Gaussian pointer model with means separated by ΔC_Q). After outcome m the normalized state is

$$\rho \mapsto \frac{M_m \rho M_m^{\dagger}}{\mathrm{Tr}[M_m^{\dagger} M_m \rho]}. \tag{3.4}$$

This produces measurement-induced Bayesian updates that lie at the core of the ISD dynamical loop. We treat this as a working observation motivating a detailed study of tunneling-dominated operating regimes; the next section analyzes how tunneling parameters influence information acquisition and sequential error exponents.

4 Tunneling-First Principle: information and error-exponent analysis

4.1 Definitions

Define the *learning rate* L as mutual information per unit time between a hidden sector label θ and measurement outcomes $\{m_t\}_{t\le T}$:

$$L \;=\; \lim_{T\to\infty} \frac{1}{T} I(\theta; \{m_t\}_{t\le T}). \tag{4.1}$$

Define the *error exponent* κ for sequential detection between parity hypotheses using Chernoff/KL divergences:

$$\kappa \;=\; \lim_{n\to\infty} \; -\frac{1}{n} \ln P_{\mathrm{err}}(n), \tag{4.2}$$

with n shots and P_{err} the minimal achievable error. We have established the framework and will now proceed to theory.

4.2 Monotonicity theorem

Lemma 4.1 (Tunneling-First Optimality — formal statement)**.** *Let the device operate under the following assumptions:*

- ***A1*** Readout-limited regime: *single-shot readout energy scale E_M satisfies $E_M \ll k_B T$.*
- ***A2*** Shot-time vs poisoning: *integration times obey $\tau \ll \tau_{\mathrm{qpp}}$ where τ_{qpp} is the quasiparticle-poisoning timescale.*
- ***A3*** Noise stationarity: *detector noise is stationary with bounded variance and spectral density that does not grow with small changes in tunneling amplitudes.*

*Under **A1**–**A3** there exists a device-dependent open window W of effective tunneling amplitudes t_C such that, for $t_C \in W$,*

$$\frac{\partial L}{\partial t_C} > 0, \qquad \frac{\partial \kappa}{\partial t_C} > 0,$$

until t_C becomes large enough to increase hybridization energy $E_M(t_C)$ and thereby degrade parity protection.

Proof. Model the readout pointer distribution for parity branches by Gaussians $p_\pm(x) = \mathcal{N}(\mu_\pm, \sigma^2)$ with separation $\Delta\mu = \mu_+ - \mu_-$. Under standard asymptotic relations, the single-shot Fisher information scales as $\mathcal{I} \propto (\Delta\mu)^2/\sigma^2$ and the mutual information rate satisfies $L \propto \mathcal{I}/\tau$ in the long-time limit. Similarly, the sequential-error exponent κ scales with the Kullback–Leibler divergence between branches and hence with $(\Delta\mu)^2/\sigma^2$.

Device models in the ParityFuse blueprint give $\Delta\mu \propto \Delta C_Q(\Phi)$ and indicate that, for near-balanced arms ($t_L \approx t_R$) in a safe operating window, $\partial_{t_C} \Delta C_Q(\Phi) > 0$, yielding the stated monotonicity of L and κ. The growth ceases when increased t_C substantially raises $E_M(t_C)$, which—by reducing topological protection—reduces effective separation $\Delta\mu$. The device-specific boundary of W depends on fabrication parameters and is determined empirically or by a detailed microscopic model. □

4.3 Corollary: engineering.

Engineering the system presents multiple practical challenges, including fabrication variability, control latency, and quasiparticle-induced faults. Given the cost model and noise assumptions, a cost-effective architecture appears to prioritize topological protection (the "topological core") before aggressive scaling of sector count: topological encoding reduces leakage and control overhead per logical unit, thereby yielding favorable gains in the risk functional R. The use of a typed, capability-constrained Java injection agent is proposed as an experimental orchestration mechanism (research platform) to mediate high-level decisions and low-level execution; emphasis on audited provenance and safe fallback modes is essential.

Optimize t_L, t_R and flux alignment to maximize t_C within the safe window W before scaling the number N of sectors; optimal benefit of scaling sectors is thus deferred until raw parity discriminability saturates. This is the operational statement of "Tunneling-First".

5 Open-quantum-system dynamics and measurement-induced updates

5.1 Stochastic master equation (quantum trajectories)

We model the research platform's substrate as subject to continuous-time parity-sensitive measurements and Markovian decoherence. In particular, the conditional state ρ_t under continuous measurement of observable $\hat{Z}$ obeys the stochastic master equation (SME) in Itô form:

$$\begin{aligned} d\rho_t = &-\frac{i}{\hbar}[\hat{H}_{\mathrm{AGI}}, \rho_t]\, dt + \mathcal{D}[\hat{L}]\rho_t\, dt \\ &+ \sqrt{\eta}\, \mathcal{H}[\hat{c}]\rho_t\, dW_t, \end{aligned} \tag{5.1}$$

where $\mathcal{D}[\hat{L}]\rho = \hat{L}\rho\hat{L}^\dagger - \frac{1}{2}\{\hat{L}^\dagger\hat{L}, \rho\}$, $\mathcal{H}[\hat{c}]\rho = \hat{c}\rho + \rho\hat{c}^\dagger - \mathrm{Tr}[(\hat{c}+\hat{c}^\dagger)\rho]\rho$, measurement operator $\hat{c} \propto \hat{Z}$, detection efficiency η, and dW_t a Wiener increment. Measurement backaction and information gain trade-offs are explicit in this equation; expected information rates can be derived by computing the average mutual information per dt. See the Lindblad parametrization and decoherence bounds in the blueprint.

5.2 Projective-update limit: the missing link.

In the single-shot ParityFuse operation (finite integration τ), the SME integrates to a discrete update described by Kraus operators $M_\pm$ (Sec. 3). The effective open-system evolution between shots includes quasiparticle poisoning channels modeled by jump operator $\hat{J}_{\mathrm{qp}}$ with Poissonian rate $\gamma_{\mathrm{qp}} = 1/\tau_{\mathrm{qpp}}$, producing additional Lindblad term $\mathcal{D}[\hat{J}_{\mathrm{qp}}]$. We now proceed to develop the sector-gate calculus and present an initial construction of representative sectoral operations that follow from the above projective-update model.

6 Sector-gate calculus and measurement-only compilation

6.1 Sector gates: formal object

Definition 6.1 (Sector-gate)**.** *A sector-gate $S_{kl}(\theta)$ is an operator acting nontrivially on $\mathcal{H}_k \otimes \mathcal{H}_l$ with parameter θ:*

$$\hat{S}_{kl}(\theta) = \exp\big(-i\theta \hat{P}_k \otimes \hat{Q}_l\big),$$

where $\hat{P}_k, \hat{Q}_l$ are Hermitian sectoral projectors or Hamiltonian fragments.

Sector-gates can be realized either by coherent gates (microwave pulses) or by measurement-induced sequences (ParityFuse RUS sequences).

6.2 ParityFuse RUS compilation model

We formalize a compilation mapping $\mathcal{C}$ from logical sector-gate sequences $\mathcal{G} = \{\hat{G}_m\}$ to ParityFuse primitives and feedforward policies. The compilation produces a tree-like plan with probabilistic branching (repeat-until-success).

Primitive: ParityFuse$(\Phi; t_L, t_R)$ outputs measurement $m \in \{\pm 1\}$ with distribution $p(m|Z)$ and updates Pauli frame $F \mapsto F(m)$.

Algorithmic semantics (deterministic trace):

$$\mathcal{C}: \hat{G} \mapsto \{(\text{ParityFuse}_i,\ \text{frame\_update}_i,\ \text{control\_advance}_i)\}_{i \in \mathcal{I}}$$

subject to the property of *deterministic provenance*: given identical randomness seed and device calibration, compilation yields identical finite instruction traces (sequence of ParityFuse calls and feedforward actions). Deterministic provenance is required for auditability and reproducible verification.

6.3 Universality proposition: first steps

Proposition 6.2 (ParityFuse universality)**.** *ParityFuse primitives together with adaptive feedforward and magic-state injection over sectors form a universal measurement-only generating set for sectoral dynamics: any unitary in the target sectoral algebra can be synthesized to arbitrary precision (modulo noise) using finite-depth RUS sequences.*

Proof . Proof. Using the standard measurement-only universality arguments, parity-projector fusions together with ancilla magic states permit synthesis of Clifford+T logic within the encoded sector subspace; adaptive feedforward converts probabilistic subroutines into deterministic gates via repeat-until-success (RUS) protocols. The asymptotic depth and shot complexity depend on the target approximation error and the error model; code distance controls the logical error rates when topological encoding is used.

Use the measurement-only universality result: parity-projector fusions and suitably prepared ancillary magic states permit synthesis of arbitrary Clifford+T logic in the encoded sector subspace; adaptive feedforward promotes non-deterministic protocols to deterministic logical gates via RUS. The depth bound grows polylogarithmically with approximation error and is controlled by code distance when topological encoding is used. See the ParityFuse calculus and Lemma 2 in the device blueprint. □

7 Covering future mathematical potentials: Operator algebra, commutators, and BCH expansions

Control-phase mismatches and non-commuting sectoral gates produce higher-order corrections in effective generators. Let A, B be bounded operators representing two sequential control segments;

Baker–Campbell–Hausdorff (BCH) yields

$$\log\left(e^A e^B\right) = A + B + \tfrac{1}{2}[A,B] + \tfrac{1}{12}[A,[A,B]] - \tfrac{1}{12}[B,[A,B]] + \cdots, \tag{7.1}$$

so localized nonzero commutators alter long-time effective Hamiltonians. Design constraints should bound $\|[A,B]\|$ to prevent runaway spectral shifts and instabilities; useful design techniques include phase-alignment regularization and localized non-commutation with bounded support.

8 The technicals: Error budgeting and composite risk functional

8.1 Definitions

Define the composite risk functional used to guide architectural optimization:

$$R \;=\; w_1\epsilon_{\text{meas}} \;+\; w_2 p_{\text{leak}} \;+\; w_3\gamma_\phi \;+\; w_4 O_{\text{ctrl}}, \tag{8.1}$$

where ϵ_{meas} is single-shot readout error, p_{leak} non-topological leakage probability, γ_ϕ logical dephasing rate, and O_{ctrl} control overhead (shots $\times$ integration time etc.). A topological core (Majorana encoding) minimizes R at fixed hardware budget by exponentially suppressing leakage and reducing O_{ctrl} via measurement-only primitives; formal optimality arguments are in the blueprint.

8.2 Optimality condition: leakage, control, cost...

Every constrained optimization needs solution. Leakage per dollar (topological protection buying the biggest reduction) can be studied as follows: Given a budget constraint $\mathcal{B}$, find architecture parameters Θ minimizing $R(\Theta)$ s.t. $\text{cost}(\Theta) \le \mathcal{B}$. Under model assumptions (resonator budgets, noise model), stationary conditions yield the topological-core minimizer described in Theorem 2 of the blueprint. In theory our AGI system should perform rather well, since the quantum network provides the states and gates, far more optimal compared binary systems. However, such a system remains untested today. It should be possible for IBM or Microsoft to test AGI systems based on java injection.

9 From academic thought to model: Java injection orchestration: formal model

We formalize a runtime agent for an experimental orchestration platform that accepts high-level decisions from the research platform and compiles them into ParityFuse execution plans. The agent is constrained by a capability model and is designed to guarantee bounded-latency execution and auditable provenance; safety fallbacks are mandatory in cases of execution overrun or fault.

9.1 Typed language and capability model

Any party such as IBM or Microsoft could proceed by making use of an injection agent.

Define a minimal typed DSL $\mathcal{L}$ embedded in Java for AGI decisions. Types include:

$$\mathsf{Type} \;::=\; \mathsf{Int} \mid \mathsf{Real} \mid \mathsf{SectorID} \mid \mathsf{Op} \mid \mathsf{Plan}$$

Capabilities are tokens $c \in \mathcal{C}$ that gate access to hardware channels:

$$\mathsf{cap} ::= \texttt{READOUT\_SECTOR(k)} \mid \texttt{COMPILE\_PLAN} \mid \texttt{SIGN\_MANIFEST}.$$

The injection agent enforces that any injected module's ambient capability set $\mathcal{C}_{\text{module}}$ is a *subset* of the hosting policy $\mathcal{C}_{\text{host}}$. We confirm such setup would be highly experimental nonetheless, plausible.

9.2 Less is more: Operational semantics

We need semantics to proceed with our workflow. We present a small-step operational semantics for the injection agent. A configuration is $(P, \Sigma, \mathcal{S})$ where P is the injected program, Σ is the state (symbol table, provenance log), and $\mathcal{S}$ is the scheduler queue. The core compilation step reduces a high-level decision $\mathsf{dec} \in \mathsf{Op}$ to a Plan:

$$(P, \Sigma, \mathcal{S}) \xrightarrow{\text{compile}} (P', \Sigma', \mathcal{S} \cup \{\pi\}),$$

where $\pi = \mathcal{C}(\mathsf{dec})$ and Σ' appends a signed manifest $\mu = \text{Sign}_{\text{host}}(\pi, \text{timestamp})$. Deterministic provenance: given identical $\mathsf{dec}, \Sigma_{\text{cal}}$ and seed, π is identical.

9.3 Safety invariants and verification obligations

Define two primary invariants:

- **I1 (Capability safety):** For every plan π executed, $\text{caps}(\pi) \subseteq \mathcal{C}_{\text{host}}$.
- **I2 (Latency bound):** For every π, the worst-case compile+execute latency $\tau(\pi) \leq \tau_{\max}$ where $\tau_{\max}$ is an a priori bound chosen to be $\ll \tau_{\text{qpp}}$.

Verification obligations: for every plan π the injection agent must produce a proof object $\mathcal{P}(\pi)$ certifying I1 and I2 under the formally specified cost model. These proof objects may combine type-checking, static worst-case execution-time (WCET) analysis of the compilation pipeline, and explicit resource bounds on generated RUS depth. If a plan π cannot be certified (e.g., WCET cannot be bounded within $\tau_{\max}$ under current calibration), the agent must automatically reject the plan or substitute a certified safe fallback that executes only classical or bounded-quantum operations; this rejection/fallback event must be recorded in the audit log for post-hoc analysis.

MANUS AI is a worthy mention, focusing on agentic tasks, they are able to perform AI related tasks under heavy load. How can our injection agent work within a wider AGI setup? Let us cover reproducibility next.

9.4 Audit log and reproducibility

Every plan π that is accepted for execution is recorded in an append-only audit ledger with the tuple $(\pi, \mu, \mathcal{P}(\pi), \text{calibration})$ where μ is the host signature. Because the compiler $\mathcal{C}$ is deterministic under a specified calibration snapshot and PRNG seed, the logged artifacts suffice to rerun compilation offline, reproduce the exact instruction trace, and verify compliance with the certified proof object.

10 Formal correctness property

Let $\mathsf{spec}(\mathsf{dec})$ be a formal logical specification of intended effect (e.g., effect on sectoral expectation values). The correctness property required is:

$$\forall \mathsf{dec}. \quad \vdash \mathcal{C}(\mathsf{dec}) : \mathsf{spec}(\mathsf{dec}) \text{ (modulo noise bound } \epsilon).$$

That is, compilation preserves semantics up to quantified noise model ϵ (which depends on device error rates and integration budgets). This reduces to verifying trace-preservation of expectation values under the generated Kraus map ensemble within error ϵ via matrix-norm bounds and operator Chernoff inequalities.

11 Practice makes perfect: Latency and scheduling

11.1 Timing model

Let each ParityFuse shot have latency $\tau_{\text{shot}} = \tau_{\text{int}} + \tau_{\text{overhead}}$ (integration + wiring/control), feedforward decisions require τ_{ff} classical processing, and RUS depth d expected attempt count $\bar{n}$. Worst-case plan latency:

$$\tau_{\text{plan}} \leq \bar{n} \cdot d \cdot (\tau_{\text{shot}} + \tau_{\text{ff}}) + \tau_{\text{ctrl}}. \tag{11.1}$$

Bound $\tau_{\text{plan}} \leq \tau_{\max}$ is an enforceable scheduling constraint in the injection agent.

11.2 Latency-aware compilation

The compiler $\mathcal{C}$ includes cost-annotated rewrite rules and will select lower-depth alternatives if τ_{plan} would exceed $\tau_{\max}$, possibly reverting to classical safe mode when quantum execution becomes infeasible within bounds.

12 Concrete worked example: compile trace

We present a worked symbolic trace mapping a logical controlled-phase sector-gate $S_{12}(\theta)$ to a ParityFuse sequence:
Input: $S_{12}(\theta)$ with spec $\left\langle \hat{P}_1 \otimes \hat{Q}_2 \middle| \hat{P}_1 \otimes \hat{Q}_2 \right\rangle \mapsto f(\theta)$.
Compilation:

1. Decompose $S_{12}(\theta)$ into Clifford+T logical pattern via sectoral encoding.
2. Map each logical CZ/T into a sequence of ParityFuse primitives and ancillary magic-state injections with expected RUS depth d_i.
3. Compute shot budget $N = \sum_i \alpha_i d_i$ s.t. expected error $\leq \epsilon$.
4. Emit plan $\pi = \langle (\text{ParityFuse}_i, t_{Li}, t_{Ri}, \Phi_i),\ \text{frame\_updates},\ \text{feedforward\_policy} \rangle$.

Provenance: $\mu = \text{Sign}_{\text{host}}(\pi, \text{calib\_snapshot}, \text{ts})$.
Execution semantics: Sequentially apply ParityFuse primitives, update Pauli frame per measurement outcome, follow feedforward transitions; stop when branch conditions satisfied or failure threshold reached. Industry partners could validate our method.

13 Experimental roadmap and microbenchmarks

Ideas for IBM (or microsoft) are: Proposed microbenchmarks (device-level and system-level):

- Single-shot parity SNR(τ) vs balanced tunneling t_L/t_R curve; fit to model in Eq. (3.1).
- Quasiparticle poisoning dwell-time τ_{qpp} statistics (telegraph model).
- RUS depth distribution for standard logical gates under typical noise model.
- End-to-end mutual information rate L for simple classification tasks mapped to sectoral measurements.

Device operating points and experimental protocols follow the blueprint; see the device blueprint for exact physical parameters and error-model fitting procedures.

14 Discussion

We have provided a full-stack formalization: Hilbert-space sector factorization, ParityFuse primitive and Kraus model, rigorous Tunneling-First optimality, SME dynamics, measurement-only compilation with RUS semantics, and a formally constrained Java-injection orchestration layer with verification conditions. In the hope of achieving a neural 'spine' for a potential AGI setup, we have concluded our combined design enforces auditable, deterministic provenance for every AGI-issued plan while allowing expressive eigenstate dynamics in the quantum substrate. Topological cores (Majorana encoding) are recommended to minimize the composite risk functional under measurement-only constraints. The relevant proofs and technical device derivations appear in the supporting materials.

15 Conclusion

An orchestrator (n8n-style nervous system) implemented as a typed, capability-constrained Java injection agent can safely and provably compile AGI decisions into measurement-only ParityFuse sequences that enact eigenstate transitions. Our observations were a success. However challenges remain: the Tunneling-First principle prescribes an experimental roadmap prioritizing tunnel-network optimization before scale. Together, this synthesis offers a concrete, verifiable pathway toward AGI-like eigenstate dynamics executed via auditable, latency-bounded quantum primitives. Further validation is needed to connect JAVA injection with potential AGI states.

Acknowledgments. We thank n8n for their efforts, openAI for their innovation.

A Appendix A: Detailed Lindblad bounds and operator-norm estimates

We provide norm bounds for commutators that arise from localized phase mismatches. Let $C = [A, B]$ with $\|A\|, \|B\| \leq \Lambda$. Then $\|C\| \leq 2\Lambda^2$. If A, B depend on phases $e^{i\phi_1}, e^{i\phi_2}$ the localized commutator norm admits the sharper bound

$$\|C\| \leq 2\Lambda \cdot |e^{i\phi_1} - e^{i\phi_2}|.$$

BCH higher-order terms can be bounded using series remainder estimates; combine with perturbation bounds on eigenvalue shifts to analyze spectral stability and effective Heff corrections over time T

B Appendix B: Java injection agent API

B.1 Primitives

Runtime API primitives

`Plan compile(Op decision)` Deterministically compile a high-level decision (for example `apply S_12(`θ`)`) into a low-level **ParityFuse trace**. The produced `Plan` must include: the ordered primitive calls, feed-forward rules, shot counts, time/latency budgets, and the calibration snapshot and PRNG seed used for compilation (so the compilation is reproducible and auditable).

`Signature sign(Plan p)` The host (trusted controller) cryptographically signs the serialized `Plan` (including the calibration snapshot, seed, proof artifacts, and WCET estimate) using a host private key (HSM/TPM). The signature attests origin and integrity.

`bool verify(Plan p, Signature s)` Verify the cryptographic signature and validate attached proof objects (capabilities, WCET proof, resource accounting). Return `true` only if the plan is authentic and admissible under policy.

`ExecutionResult execute(Plan p, Signature s)` Execute a *verified* plan on the real-time controller with enforced timeouts and watchdogs. Returns measurement outcomes, telemetry, execution logs, and a status flag. If any bound is violated or the hardware fault occurs, the executor aborts or switches to a predefined safe fallback and records the incident.

B.2 Proof obligations

For each Plan p the compile pipeline must emit a proof artifact $\mathcal{P}(p)$ that certifies:

1. capability-safety: $\text{caps}(p) \subseteq \text{hostCaps}$,
2. latency bound: $\text{WCET}(p) \leq \tau_{\max}$,
3. correctness: $\|\mathcal{E}_p(\cdot) - \mathcal{S}_{\text{spec}}(\cdot)\|_\diamond \leq \epsilon$,

where $\mathcal{E}_p$ is the implemented CPTP map (averaged over measurement randomness) and $\mathcal{S}_{\text{spec}}$ the specified CPTP semantics.