Machine-Verified Viability Threshold Certificates for Coherence-Controlled Fusion

A Lean 4 Certificate from Explicit Power-Balance Models

IndisputableMonolith Project

2026-01-24

Abstract

We formalize and certify a concrete *viability* criterion for fusion power balance in the Recognition Science (RS) coherence-controlled fusion program. To avoid hidden assumptions, we commit to explicit, auditable proxy models for (i) thermally averaged reactivity and (ii) bremsstrahlung + transport losses, together with a deposition fraction. Within this model layer, we derive two *solvable thresholds*: (1) a temperature threshold T^* computed directly from the closed-form Gamow-exponent proxy and fuel parameters, and (2) an enhancement threshold E^* obtained by algebraically solving the dominance inequality required for net-positive deposited heating. We then prove in Lean 4 that if $T \geq T^*$ and $E \geq E^*$ (with explicit side conditions on density and coefficients), the viability inequality $L_{\text{total}}(T) < E \cdot P_{\text{dep}}(T)$ holds. The paper is structured to function as a human-readable *certificate excerpt*: it records the exact definitions, theorem names, and a reproducibility command that replays the proof. As with any certified model, this work does not claim that nature must satisfy the proxy equations; instead it cleanly separates what is machine-derived from what remains an empirical seam.

1 Introduction and scope

1.1 Problem: viability is an inequality, not a slogan

In fusion engineering discourse, statements such as "viability" or "net positive power" are often discussed informally. For deployment-grade rigor, however, viability must be expressed as an *inequality* between explicit, auditable functions of operating conditions. In this paper we treat *viability* as a concrete power-balance statement of the form

$$L_{\text{total}}(T, n, Z_{\text{eff}}, \ldots) \;<\; E \cdot P_{\text{dep}}(T, n, \ldots), \tag{1}$$

where L_{total} is a chosen loss model (here: bremsstrahlung + transport), P_{dep} is a deposited fusion-heating proxy, and E is an abstract *enhancement factor* representing any mechanism that increases effective reactivity relative to a classical baseline.

1.2 Goal: a Lean-checked implication

The central technical goal is to make the viability claim *checkable* and *machine-verifiable*. Concretely, we prove a Lean-checked implication of the form

$$(T \geq T^*) \;\wedge\; (E \geq E^*) \;\Longrightarrow\; \text{viable}, \tag{2}$$

where the thresholds T^* and E^* are not opaque tuning knobs, but are derived by explicit algebra from the committed proxy models.

At the Lean level, the "viable" conclusion is a precise predicate over the concrete loss and heating functions; the final statement is proved in `IndisputableMonolith/Fusion/ViabilityThresholds.lean` (see `viable_of_T_ge_T_star_and_E_ge_E_star`). This paper is designed so that a reader can: (i) understand the definitions used, and (ii) replay the proof via a single build command, without trusting any informal algebra performed outside the proof assistant.

1.3 Relationship to `RS_Coherence_Controlled_Fusion.tex`

The companion document `papers/tex/RS_Coherence_Controlled_Fusion.tex` develops the broader Recognition Science (RS) coherence-controlled fusion program: it defines the RS coherence variables, the barrier scale, and the operational protocol for computing an enhancement effect from diagnostic and scheduling measurements.

This paper is deliberately narrower. It serves as the *certificate/derivation layer for power balance*: given any proposed or measured enhancement factor E (whether computed from RS coherence metrics, inferred empirically, or supplied as a conservative bound), we certify that viability follows once the explicit thresholds T^* and E^* are met under the committed loss and deposition models. In this way, the two papers separate concerns:

- **RS coherence paper:** how to obtain or justify an enhancement factor E from RS control.
- **This certificate paper:** how to convert E into a formal, replayable viability guarantee.

1.4 Scope and non-scope

In scope:

- Explicit proxy definitions for $\langle\sigma v\rangle(T)$, L_{total}, and P_{dep} used in the certificate.
- Closed-form, solvable thresholds T^* (from the Gamow proxy) and E^* (from solving the dominance inequality) and a Lean proof of (2).
- A reproducibility path that replays the proof in the Lean kernel.

Out of scope (future strengthening):

- Facility-specific calibration of coefficients and unit conversions (kept as explicit parameters).
- Bosch–Hale or tabulated reactivity fits, richer transport/radiation models, and detailed energy-partition/neutronics (these can be swapped in as future model modules while keeping the certificate interface stable).
- "Ignition is trivial" style claims beyond the certified model implications; this paper only certifies the stated inequality under explicit assumptions.

2 Seam policy and unit conventions (RS-native vs CODATA-quarantined)

2.1 Certified surface vs. empirical seam

This paper is written to support a national-lab audit posture: the reader must be able to separate what is *kernel-checked* from what is *externally asserted*. We therefore adopt the repository-wide cal-

ibration seam policy (see `IndisputableMonolith/CALIBRATION_SEAM_POLICY.md`), which enforces a strict separation between:

- **Certified surface (cost-first core + explicit model layer):** definitions and theorems proved in Lean, including the statement and proof of the implication (2) once a model is committed.
- **Empirical seam (external anchors and calibration inputs):** any mapping to SI units, CODATA numerals, facility calibrations, and any measurement-derived quantities not proved from the model assumptions.

The key point is not to eliminate seams, but to *label them mechanically* and to ensure they are the only inputs that can affect numerical conclusions. The Lean proof itself is invariant under facility choices: once the definitions are fixed, only the hypotheses determine applicability.

2.2 Unit conventions: RS-native core and CODATA-quarantined adapters

The certified core is expressed in an RS-native unit system with $\tau_0 = 1$ tick and $c = 1$ (voxel/tick), so that theorems do not depend on any empirical values of c, $\hbar$, G, etc. This is formalized in `IndisputableMonolith/Constants/RSNativeUnits.lean`.

When SI reporting is required, it is routed through an explicit adapter layer that takes an *externally supplied* calibration record (e.g. seconds per tick, joules per coherence quantum). This adapter lives in `IndisputableMonolith/Measurement/RSNative/Calibration/SI.lean` and is intended to be the only place where RS-native quantities are converted for external reporting.

Conversely, SI/CODATA constants are quarantined into explicit anchor modules (e.g. `IndisputableMonolith/Constants/Codata.lean` and related anchors), which are *not* imported by the certified fusion viability proof modules. This ensures that replaying the viability theorem does not depend on any mutable empirical database of constants.

2.3 Facility-supplied inputs (explicit parameters)

The viability certificate proved here is conditional on explicit parameters that must be supplied by a facility (or a simulator configured for a facility). Concretely, the power-balance model is parameterized by the coefficients in `IndisputableMonolith/Fusion/PowerBalance.lean`:

- **Loss coefficients:** k_{brem} and k_{tr} (with nonnegativity assumptions), encoding the selected bremsstrahlung and transport-loss proxy magnitudes in the facility's chosen units.
- **Heating scale:** k_{fus} (with positivity/nonnegativity assumptions), absorbing Q-value, geometry, and any baseline scaling choices in the committed proxy.
- **Deposition fraction:** $f_{\mathrm{dep}} \in [0, 1]$ (with positivity used for the threshold theorem), representing the fraction of fusion power that deposits into the plasma.
- **Operating point / regime constraints:** density n (assumed $n > 0$ in the threshold theorem) and effective charge Z_{eff} (assumed $Z_{\mathrm{eff}} \geq 0$).

These are *not* hidden: the proof exposes them as hypotheses, and the derived enhancement threshold E^* is an explicit function of these facility-supplied parameters. In addition, the enhancement factor E itself is treated abstractly: it may be supplied conservatively, inferred empirically, or computed from RS coherence metrics as described in the companion paper.

2.4 Audit posture for national-lab runs

We recommend treating every viability claim as a two-layer artifact:

1. **Kernel-checked certificate core:** the Lean theorem `viable_of_T_ge_T_star_and_E_ge_E_star` together with its import closure (the exact definitions of T^*, E^*, L_{total}, and P_{dep}).
2. **Externally asserted instantiation record:** the concrete numeric values for $(k_{\text{brem}}, k_{\text{tr}}, k_{\text{fus}}, f_{\text{dep}}, n, Z_{\text{eff}}, T, E)$, plus provenance for how they were obtained and (optionally) a separate checkable computation trace showing the inequalities $T \geq T^*$ and $E \geq E^*$ hold for those values.

In other words:

- **Lean kernel checks:** the algebraic implication "assumptions $\Rightarrow$ viability" for the committed model layer (no shortcuts, no informal steps).
- **The facility asserts:** the numerical instantiation and calibration choices (including any SI mapping), and the measurement-derived claim that a given shot or design point satisfies the hypotheses.

This separation is not a weakness; it is the audit interface. It ensures that disagreements can be localized: either one disputes the model (by proposing a different $\langle \sigma v \rangle$, transport model, or deposition model), or one disputes the instantiation (the facility-supplied parameters and their measurement provenance), without ever needing to dispute the correctness of the proof itself.

3 Formal definitions (Lean-aligned)

This section fixes notation and records the Lean-aligned definitions used throughout the certificate. The guiding rule is: every symbol in the certificate must have a corresponding Lean definition, and every nontrivial fact used in the certificate must be either a Lean theorem or an explicitly stated hypothesis (seam input).

3.1 Viability predicate (net-positive deposited heating)

We treat viability as a strict power-balance predicate at a given operating point. Let $P_{\text{fus}}(T)$ denote a deposited fusion-heating proxy and let $P_{\text{loss}}(T)$ denote a loss proxy. Then the viability condition is:

$$\text{viable}(T) \;:=\; P_{\text{loss}}(T) \;<\; P_{\text{fus}}(T). \tag{3}$$

This aligns exactly with the Lean definition `Ignition.viable (P_fus P_loss : ℝ → ℝ) (T : ℝ) : Prop := P_loss T < P_fus T` in `IndisputableMonolith/Fusion/Ignition.lean`.

In our certificate instantiation, $P_{\text{loss}}(T)$ is the concrete loss proxy L_{total} from `IndisputableMonolith/Fusion/PowerBalance.lean`, while $P_{\text{fus}}(T)$ is an enhanced deposited-heating proxy defined as a baseline deposited proxy multiplied by a dimensionless enhancement factor E (defined next).

3.2 Enhancement factor E: meaning and (optional) RS coherence linkage

The enhancement factor E is a *dimensionless multiplicative lower bound* on how much larger the deposited-heating proxy is compared to a classical baseline proxy at the same operating point. Concretely, we model:

$$P_{\text{fus}}(T) \;:=\; E \cdot P_{\text{dep},0}(T), \tag{4}$$

where $P_{\mathrm{dep},0}(T)$ is a baseline deposited fusion-heating proxy fixed by the committed model layer. In the Lean development, $P_{\mathrm{dep},0}$ corresponds to `PowerBalanceBounds.Pdep_proxy`.

Optional direct RS linkage. This certificate paper treats E abstractly on purpose: it may come from RS coherence control, from conservative bounds, or from empirical inference. If (and when) we choose to link E directly to the RS coherence/barrier-scaling mechanism, there is a canonical Lean definition available:

$$E_{\mathrm{RS}}(T) \;:=\; \frac{\exp(-\eta_{\mathrm{RS}}(T))}{\exp(-\eta(T))}, \tag{5}$$

i.e. the ratio of the RS tunneling proxy to the classical tunneling proxy at temperature T for a chosen fuel pair. In the Lean code this ratio is defined as `PowerBalance.enhancement` in `IndisputableMonolith/Fusion/PowerBalance.lean`, and it is proved to satisfy $E_{\mathrm{RS}}(T) \geq 1$ (theorem `enhancement_ge_one`) because the RS barrier scale satisfies $0 < S \leq 1$ and therefore never increases the exponent.

Operationally, the companion coherence paper provides the mechanism to compute the barrier scale S from RS control metrics and thereby justify a lower bound on E; this certificate then converts that E into a replayable viability guarantee.

3.3 Parameter bundle and side conditions

The certificate is parameterized by an explicit, audited parameter bundle (Lean structure `PowerBalance.Params` in `IndisputableMonolith/Fusion/PowerBalance.lean`) containing:

- **Loss coefficients:** $k_{\mathrm{brem}} \geq 0$ and $k_{\mathrm{tr}} \geq 0$.
- **Fusion heating scale:** $k_{\mathrm{fus}} \geq 0$.
- **Deposition fraction:** $f_{\mathrm{dep}} \in [0, 1]$.

In addition to these built-in side conditions, the *threshold theorem* that yields explicit T^* and E^* requires the following regime assumptions at the operating point:

- **Positivity for divisions:** $n > 0$, $f_{\mathrm{dep}} > 0$, and $k_{\mathrm{fus}} > 0$ (so that the derived E^* is well-defined and physically meaningful).
- **Effective charge nonnegativity:** $Z_{\mathrm{eff}} \geq 0$.
- **Temperature positivity:** $T > 0$ (encoded in Lean via `GamowParams`).

These conditions are explicit inputs in the theorem statement, not implicit assumptions.

3.4 Notation-to-Lean map (certificate surface)

Paper symbol	Meaning	Lean definition (file)
viable	net-positive power predicate	`Ignition.viable` (`Fusion/Ignition.lean`)
L_{total}	loss proxy (brems + transport)	`PowerBalance.L_total` (`Fusion/PowerBalance.lean`)
$P_{\mathrm{dep},0}$	baseline deposited heating proxy	`PowerBalanceBounds.Pdep_proxy` (`Fusion/PowerBalanc`
E	enhancement factor (dimensionless)	external hypothesis / optional `PowerBalance.enhancemer`
T^*	temperature threshold	`ViabilityThresholds.T_star` (`Fusion/ViabilityThres`
E^*	enhancement threshold	`ViabilityThresholds.E_star` (`Fusion/ViabilityThres`

4 Committed physics proxies

This certificate is only meaningful after we commit to explicit proxy models for reactivity, losses, and deposited heating. The purpose of this section is to state those commitments transparently and to map them to the Lean definitions that the kernel checks.

4.1 Reactivity proxy

We define a concrete proxy for thermally averaged reactivity:

$$\langle \sigma v \rangle(T) \;:=\; T \,\exp\big(-\eta(T)\big). \tag{6}$$

Here $\eta(T)$ is the Gamow-exponent proxy used in the fusion rate layer. In Lean, the corresponding definitions are:

- `ReactivityProxy.sigmaV` in `IndisputableMonolith/Fusion/ReactivityProxy.lean` (defined as `g.temperature * classicalTunneling g cfgA cfgB`).
- `classicalTunneling` and `gamowExponent` in `IndisputableMonolith/Fusion/ReactionNetworkRates.lean`.

4.2 Loss model (bremsstrahlung + transport)

We commit to a two-term loss proxy consisting of bremsstrahlung radiation and a simplified transport loss:

$$L_{\mathrm{brem}}(T, n, Z_{\mathrm{eff}}) := k_{\mathrm{brem}}\, Z_{\mathrm{eff}}\, n^2 \sqrt{T}, \tag{7}$$

$$L_{\mathrm{tr}}(T, n) := k_{\mathrm{tr}}\, n\, T, \tag{8}$$

$$L_{\mathrm{total}}(T, n, Z_{\mathrm{eff}}) := L_{\mathrm{brem}}(T, n, Z_{\mathrm{eff}}) + L_{\mathrm{tr}}(T, n). \tag{9}$$

These correspond to `PowerBalance.L_brem`, `PowerBalance.L_tr`, and `PowerBalance.L_total` in `IndisputableMonolith/Fusion/PowerBalance.lean`.

4.3 Deposited heating proxy

We define deposited heating as a deposition fraction times a baseline fusion heating proxy built from density and reactivity:

$$P_{\mathrm{dep},0}(T, n) \;:=\; f_{\mathrm{dep}}\, k_{\mathrm{fus}}\, n^2\, \langle \sigma v \rangle(T), \tag{10}$$

and the enhanced deposited heating used in the viability inequality as:

$$P_{\mathrm{dep}}(T, n) \;:=\; E \cdot P_{\mathrm{dep},0}(T, n). \tag{11}$$

In Lean, the baseline deposited proxy is implemented as `PowerBalanceBounds.Pdep_proxy` in `IndisputableMonolith/Fusion/PowerBalanceBounds.lean`, with the same parameter bundle `PowerBalance.Params`.

4.4 Model commitments, not nature-claims

These proxy equations are *model commitments* introduced to make the certificate mathematically explicit and auditable. They are not claims that nature must obey the proxies exactly. The certificate therefore has the following semantics:

> If the facility (or simulator configuration) commits to (6)–(11) as its model layer, and if the facility-supplied instantiation satisfies the theorem hypotheses, then the Lean kernel certifies the resulting viability implication.

If a different reactivity fit, transport model, radiation model, or deposition model is desired, the correct procedure is to replace the model layer and re-prove the corresponding threshold theorem, preserving the same audit interface.

5 Main theorem (the certificate theorem)

5.1 Statement (Lean-certified)

Fix a fuel pair (nuclear configurations) and a facility parameter bundle $(k_{\text{brem}}, k_{\text{tr}}, k_{\text{fus}}, f_{\text{dep}})$ satisfying the side conditions in `PowerBalance.Params` (including $k_{\text{brem}} \geq 0$, $k_{\text{tr}} \geq 0$, $k_{\text{fus}} \geq 0$, and $f_{\text{dep}} \in [0, 1]$). Let n be the operating density and Z_{eff} the effective charge.

Define the explicit thresholds T^* and E^* as in `IndisputableMonolith/Fusion/ViabilityThresholds.lean`:

$$T^* := \max\bigl(1,\ \text{gamowCoeff}^2\bigr), \quad \text{gamowCoeff} := 31.3\, Z_1 Z_2 \sqrt{\mu}, \tag{12}$$

$$E^* := \frac{k_{\text{brem}} Z_{\text{eff}} + k_{\text{tr}}/n}{f_{\text{dep}}\, k_{\text{fus}}\, e^{-1}} + 1. \tag{13}$$

Here Z_1, Z_2 and μ (the reduced mass proxy) are determined by the fuel pair in the nuclear configuration layer.

Certificate theorem (informal math statement). Assume the explicit side conditions

$$n > 0, \quad Z_{\text{eff}} \geq 0, \quad f_{\text{dep}} > 0, \quad k_{\text{fus}} > 0. \tag{14}$$

If

$$T \geq T^* \quad \text{and} \quad E \geq E^*, \tag{15}$$

then the viability inequality holds for the committed proxy model:

$$L_{\text{total}}(T, n, Z_{\text{eff}}) \ < \ E \cdot P_{\text{dep},0}(T, n). \tag{16}$$

Equivalently, in the Lean-aligned predicate form (Section 3), the operating point is viable.

Lean theorem. This result is proved by the Lean theorem `viable_of_T_ge_T_star_and_E_ge_E_star` in `IndisputableMonolith/Fusion/ViabilityThresholds.lean`.

5.2 Short proof roadmap (mirrors the Lean structure)

The proof is intentionally structured as a small number of auditable lemmas, each corresponding to a named Lean theorem:

1. **Derive the "good exponent" regime from $T \geq T^*$.** Using the closed form $\eta(T) = \text{gamowCoeff}/\sqrt{T}$ in the Gamow proxy, Lean proves that $T \geq T^*$ implies:

- $T \geq 1$ (so $\sqrt{T} \leq T$), and
- $\eta(T) \leq 1$ (so $\exp(-\eta(T)) \geq \exp(-1)$).

This is discharged in Lean by `gamowExponent_le_one_of_T_ge_T_star`.

2. **Solve the dominance inequality to obtain E^*.** From $E \geq E^*$ and the positivity assumptions (14), Lean proves a strict "margin" inequality ensuring enhanced deposited heating dominates the loss terms. This is the lemma `margin_of_E_ge_E_star`.
3. **Apply the bound-to-viability lemma.** With (i) the regime facts from Step 1 and (ii) the margin inequality from Step 2, Lean applies the core inequality lemma `PowerBalanceBounds.L_total_lt_E_Pdep_proxy`, which combines: $\sqrt{T} \leq T$ and $\exp(-\eta(T)) \geq \exp(-1)$ with the explicit margin to conclude (16).
4. **Conclude the viability predicate.** Finally, Lean unfolds the definition of `Ignition.viable` and rewrites the inequality into the predicate form (3).

6 Certificate excerpt: human-readable mirror of the Lean artifact

This section is intentionally written in the style of a "certificate excerpt": it records the exact Lean artifacts that constitute the certified surface of the viability claim. A lab can attach this section as an appendix to an experimental report, together with an instantiation record of numerical parameters.

6.1 Lean sources (paths)

Primary sources for this certificate:

- `IndisputableMonolith/Fusion/ViabilityThresholds.lean`
- `IndisputableMonolith/Fusion/PowerBalanceBounds.lean`
- `IndisputableMonolith/Fusion/PowerBalance.lean`
- `IndisputableMonolith/Fusion/ReactivityProxy.lean`
- `IndisputableMonolith/Fusion/Ignition.lean`
- `IndisputableMonolith/Fusion/ReactionNetworkRates.lean`

6.2 Certified theorem name

- **Theorem:** `viable_of_T_ge_T_star_and_E_ge_E_star`
- **File:** `IndisputableMonolith/Fusion/ViabilityThresholds.lean`

6.3 Key definitions referenced by the theorem

- **Viability predicate:** `Ignition.viable` (`Fusion/Ignition.lean`)
- **Loss model:** `PowerBalance.L_total`, `L_brem`, `L_tr` (`Fusion/PowerBalance.lean`)
- **Deposited heating proxy:** `PowerBalanceBounds.Pdep_proxy` (`Fusion/PowerBalanceBounds.lean`)
- **Reactivity proxy:** `ReactivityProxy.sigmaV` (`Fusion/ReactivityProxy.lean`)
- **Temperature threshold:** `ViabilityThresholds.T_star` (`Fusion/ViabilityThresholds.lean`)
- **Enhancement threshold:** `ViabilityThresholds.E_star` (`Fusion/ViabilityThresholds.lean`)

6.4 Replay command (reproducibility)

To replay the proof from scratch using the project toolchain:

```
lake build IndisputableMonolith.Fusion.ViabilityThresholds
```

6.5 Optional: one-page certificate block (lab appendix)

RS Fusion Viability Threshold Certificate (Lean 4)

Claim certified (model-layer):
If $T \geq T^*$ and $E \geq E^*$ (with $n > 0$, $Z_{\text{eff}} \geq 0$, $f_{\text{dep}} > 0$, $k_{\text{fus}} > 0$), then $L_{\text{total}}(T, n, Z_{\text{eff}}) < E \cdot P_{\text{dep},0}(T, n)$, i.e. the operating point is viable.

Lean theorem: `viable_of_T_ge_T_star_and_E_ge_E_star`
Lean file: `IndisputableMonolith/Fusion/ViabilityThresholds.lean`
Replay: `lake build IndisputableMonolith.Fusion.ViabilityThresholds`

Facility instantiation record (to attach):
$\{k_{\text{brem}}, k_{\text{tr}}, k_{\text{fus}}, f_{\text{dep}}, n, Z_{\text{eff}}, T, E\}$ with provenance and a check that $T \geq T^*$ and $E \geq E^*$.

7 Integration into the RS Native Fusion Simulator (certificate bundles)

This section describes how the RS Native Fusion Simulator operationalizes the Lean certificate as a machine-readable artifact. The guiding design constraint is the national-lab audit posture articulated in Section 2: the simulator must emit outputs that (i) are traceable to a specific Lean theorem and (ii) contain enough information for an independent reviewer to check that the theorem hypotheses were satisfied for a given shot or design point.

7.1 Computing T^* (fuel-dependent)

The temperature threshold T^* depends only on the chosen fuel pair (nuclear configurations) through the Gamow exponent proxy. In the Lean development, T^* is `ViabilityThresholds.T_star`, defined as $\max(1, \text{gamowCoeff}^2)$ where $\text{gamowCoeff} = 31.3\, Z_1 Z_2 \sqrt{\mu}$ and $\mu = \frac{A_1 A_2}{A_1 + A_2}$ is the reduced-mass proxy.

Simulator implementation. Given a reaction selection (e.g. D–T, p–B11), the simulator extracts Z_1, Z_2, A_1, A_2 from its internal nuclear configuration objects, computes μ, then computes gamowCoeff and finally T^*. The critical engineering rule is that this computation must follow the Lean definition *exactly* (same constant, same reduced-mass formula) so that the emitted T^* is the Lean T^*, not a nearby approximation.

7.2 Computing E^* (facility- and regime-dependent)

The enhancement threshold E^* is the minimal enhancement sufficient to overcome the committed loss model at the given operating point. In Lean, E^* is `ViabilityThresholds.E_star` and (for $n > 0$, $f_{\text{dep}} > 0$, $k_{\text{fus}} > 0$) has the closed form (13):

$$E^* = \frac{k_{\text{brem}} Z_{\text{eff}} + k_{\text{tr}}/n}{f_{\text{dep}}\, k_{\text{fus}}\, e^{-1}} + 1.$$

Simulator implementation. The simulator computes E^* from the facility-supplied instantiation record: $\{k_{\text{brem}}, k_{\text{tr}}, k_{\text{fus}}, f_{\text{dep}}, n, Z_{\text{eff}}\}$. These are the explicit seam inputs described in Section 2; the simulator treats them as configuration parameters with provenance (calibration source, uncertainty model, timestamp, etc.).

7.3 Obtaining the enhancement factor E

The certificate layer treats E abstractly, but the simulator must produce (or accept) a concrete value or lower bound. There are two supported modes:

- **RS-derived mode:** compute E from RS coherence/barrier scaling by evaluating the ratio of tunneling proxies (or reactivity proxies) under RS scaling versus classical scaling at the operating point. This is consistent with the Lean definition `PowerBalance.enhancement` (ratio of RS tunneling to classical tunneling).
- **External-bound mode:** accept a conservative E supplied by the facility (e.g. from an empirical bound, a diagnostic inference, or a worst-case analysis), and treat it as an explicit hypothesis.

In both cases, the certificate emitted by the simulator must record which mode was used and the provenance of the bound.

7.4 Emitting a viability certificate bundle

The simulator emits a *certificate bundle* (JSON) whose core purpose is to tie a particular run to: (i) the Lean theorem reference and replay command, and (ii) the concrete hypothesis-check that makes the theorem applicable.

Suggested certificate fields. The existing simulator already implements auditable certificate bundles (see `rs_fusion_simulator/fusion/certificate.py` and the Lean interface `IndisputableMonolith/Fusion/Executable/Interfaces.lean`). For viability, a certificate bundle should minimally include:

- **Theorem reference:** `IndisputableMonolith.Fusion.ViabilityThresholds.viable_of_T_ge_T_star_an`
- **Replay command:** `lake build IndisputableMonolith.Fusion.ViabilityThresholds`
- **Inputs (hashed + optionally embedded):** fuel pair identifiers $(Z_1, A_1), (Z_2, A_2)$, T, n, Z_{eff}, E, and $\{k_{\text{brem}}, k_{\text{tr}}, k_{\text{fus}}, f_{\text{dep}}\}$
- **Derived values:** T^*, E^*, and boolean checks of $T \geq T^*$ and $E \geq E^*$
- **Status:** `passed = true` iff the side conditions and inequalities required by the theorem are satisfied

Certificate semantics. When `passed = true`, the certificate asserts: "the hypotheses of the Lean theorem are satisfied for these recorded inputs." The Lean kernel provides the implication from those hypotheses to the viability predicate; the simulator provides the instantiation record. When `passed = false`, the simulator must record which hypothesis failed (e.g. $E < E^*$ or $T < T^*$), so the failure is diagnostic rather than ambiguous.

National-lab audit workflow. For a live run, the lab can archive: (i) the certificate JSON, (ii) the raw diagnostic data used to justify E (and the calibration record for coefficients), and (iii) the exact repository/toolchain state needed to replay `lake build`. This yields a full chain: *measurement provenance ⇒ hypothesis check ⇒ kernel-checked implication.*

8 Limitations and next strengthening steps

The certificate proved in this paper is intentionally conservative and model-layer in nature. Its strength is auditability: every dependency is explicit and every inference step is kernel-checked. Its limitation is also explicit: the certificate is only as realistic as the committed proxy models. This section records the immediate strengthening path while preserving the same audit interface.

8.1 Upgrade the reactivity proxy to Bosch–Hale (import or formalize)

The current certificate commits to the analytic proxy $\langle\sigma v\rangle(T) = T\exp(-\eta(T))$, which captures the dominant exponential barrier structure but does not reproduce high-fidelity, channel-specific reactivity curves.

The next strengthening step is to replace the proxy with a Bosch–Hale type parameterization (or tabulated fit) for $\langle\sigma v\rangle(T)$ for each fuel cycle of interest (D–T, D–^{3}He, p–B^{11}, etc.). There are two acceptable routes:

- **Import route:** ingest a published Bosch–Hale fit into a quarantined "external fit" module, with explicit provenance and (optionally) interval bounds, then prove the required inequality lemmas against those bounds.
- **Formalize route:** encode the Bosch–Hale formula directly in Lean and prove the monotonicity/bounding lemmas needed by the threshold proof.

In both cases, the goal is not to "trust Bosch–Hale" but to make the dependence auditable and to enable certified, replayable bounds at the temperatures relevant to a facility's operating envelope.

8.2 Richer radiation, transport, and deposition physics

The present loss/heating layer is intentionally minimal: bremsstrahlung scales as $Z_{\text{eff}}n^2\sqrt{T}$ and transport scales as nT. This can be strengthened along three axes:

- **Radiation:** add line radiation, synchrotron, opacity/self-absorption, and nonequilibrium effects, with explicit regime hypotheses (e.g. optically thin vs thick).
- **Transport:** replace the single $k_{\text{tr}}nT$ term with a confinement-time model (or a family of models), including temperature and density dependence, geometry scalings, and turbulence proxies where appropriate.
- **Deposition:** replace constant f_{dep} with a physically grounded deposition model depending on areal density, stopping powers, alpha ranges, and (for aneutronic fuels) charged-particle energy partition.

Each strengthening should be introduced as a new Lean "model module" that makes all new assumptions explicit.

8.3 Preserve the certificate interface (only the model module changes)

Crucially, none of the strengthening steps require changing the *certificate interface*: we still want a theorem of the same audit shape,

$$(T \geq T^*) \ \wedge \ (E \geq E^*) \implies \text{viable}, \quad \text{with explicit side conditions.}$$

What changes is the definition of $\langle \sigma v \rangle$, L_{total}, and P_{dep}, and therefore the algebra that produces the thresholds T^* and E^*.

Operationally, this means the simulator can keep the same "viability certificate" data model (Section 7) while updating only:

- the model module that computes the proxies and thresholds, and
- the Lean theorem reference it cites (a new theorem proved for the upgraded model).

This preserves audit continuity: reviewers can compare certificates across model versions by comparing the referenced theorem and its import closure, rather than reverse-engineering ad hoc code changes.

A Lean excerpts (key definitions and theorem statement)

This appendix provides a compact, human-readable mirror of the key Lean artifacts. These excerpts are included for audit convenience; the authoritative sources are the files listed in Section 6. Proof bodies are omitted where lengthy; the intent is to show the exact definitions and the theorem statement that the kernel checks.

A.1 Viability predicate (`Ignition.viable`)

File: `IndisputableMonolith/Fusion/Ignition.lean`

```
def viable (P_fus P_loss : ℝ → ℝ) (T : ℝ) : Prop :=
  P_loss T < P_fus T
```

A.2 Reactivity proxy (`ReactivityProxy.sigmaV`)

File: `IndisputableMonolith/Fusion/ReactivityProxy.lean`

```
def sigmaV (g : GamowParams) (cfgA cfgB : NuclearConfig) : ℝ :=
  g.temperature * classicalTunneling g cfgA cfgB
```

A.3 Loss model (`PowerBalance.L_total`)

File: `IndisputableMonolith/Fusion/PowerBalance.lean`

```
def L_brem (P : Params) (T n Zeff : ℝ) : ℝ :=
  P.k_brem * Zeff * (n ^ 2) * Real.sqrt T

def L_tr (P : Params) (T n : ℝ) : ℝ :=
  P.k_tr * n * T

def L_total (P : Params) (T n Zeff : ℝ) : ℝ :=
  L_brem P T n Zeff + L_tr P T n
```

A.4 Deposited heating proxy (PowerBalanceBounds.Pdep_proxy)

File: IndisputableMonolith/Fusion/PowerBalanceBounds.lean

```
def Pdep_proxy (P : PowerBalance.Params) (g : GamowParams)
    (cfgA cfgB : NuclearConfig) (n : ℝ) : ℝ :=
  P.f_dep * (P.k_fus * (n ^ 2) * ReactivityProxy.sigmaV g cfgA cfgB)
```

A.5 Threshold definitions (T_star and E_star)

File: IndisputableMonolith/Fusion/ViabilityThresholds.lean

```
def T_star (cfgA cfgB : NuclearConfig) : ℝ :=
  max 1 ((gamowCoeff cfgA cfgB) ^ 2)

def E_star (P : PowerBalance.Params) (n Zeff : ℝ) : ℝ :=
  ((P.k_brem * Zeff) + (P.k_tr / n)) / A P + 1
```

A.6 Certificate theorem (Lean statement)

File: IndisputableMonolith/Fusion/ViabilityThresholds.lean

```
theorem viable_of_T_ge_T_star_and_E_ge_E_star
    (P : PowerBalance.Params)
    (cfgA cfgB : NuclearConfig)
    (T n Zeff E : ℝ)
    (hT : T_star cfgA cfgB ≤ T)
    (hn : 0 < n) (hZ : 0 ≤ Zeff)
    (h_fdep : 0 < P.f_dep) (h_kfus : 0 < P.k_fus)
    (hE : E_star P n Zeff ≤ E) :
    viable (fun _ => E * PowerBalanceBounds.Pdep_proxy P
        ⟨T, T_pos_of_T_ge_T_star cfgA cfgB T hT⟩ cfgA cfgB n)
      (fun _ => PowerBalance.L_total P T n Zeff) T := by
  -- proof in Lean file (omitted here)
```

B Reproducibility checklist (toolchain, commands, file paths)

This checklist records the minimum information needed to replay the certificate theorem in a clean environment.

Toolchain and packages

- **Lean toolchain:** `leanprover/lean4:v4.27.0-rc1` (from the repository file `lean-toolchain`).
- **Dependencies:** pinned by `lake-manifest.json` (including Mathlib).

Commands to replay the theorem

1. (Recommended) Fetch the Mathlib cache once:

   ```
   lake exe cache get
   ```

2. Replay the certificate theorem build target:

   ```
   lake build IndisputableMonolith.Fusion.ViabilityThresholds
   ```

3. (Optional) Replay the bound lemma module explicitly:

```
lake build IndisputableMonolith.Fusion.PowerBalanceBounds
```

File paths (certificate import closure)

- `IndisputableMonolith/Fusion/ViabilityThresholds.lean`
- `IndisputableMonolith/Fusion/PowerBalanceBounds.lean`
- `IndisputableMonolith/Fusion/PowerBalance.lean`
- `IndisputableMonolith/Fusion/ReactivityProxy.lean`
- `IndisputableMonolith/Fusion/Ignition.lean`
- `IndisputableMonolith/Fusion/ReactionNetworkRates.lean`

Recommended audit metadata to archive with lab runs

- Repository revision identifier (git commit hash) and any local patch set.
- The emitted simulator certificate JSON (Section 7), including input hash and theorem reference.
- The facility instantiation record and provenance for E, coefficients, and diagnostics.