KTLean

Formal Spine, Module Roadmap, and Research Program

A public guide to the machine-checked formalization of Kosmoplex Theory

Repository: https://github.com/KosmoNexus/KTLean
Project website: https://theKTProject.org

Christian Macedonia

Draft 1 - July 2026

Verified repository checkpoint. At the time of this release, KTLean contains 1,992 theorem and lemma declarations, completes a full `lake build`, contains no occurrences of `sorry` or `sorryAx`, has undergone a `#print axioms` audit of principal capstones, and has passed a fresh-clone build on an independent physical drive.

Abstract

KTLean is a machine-checked formalization program for the mathematical and computational structure of Kosmoplex Theory. Its current corpus establishes a substantial formal spine: triadic closure, the canonical 42-glyph spectrum, spinorial double-cover structure, visible projection with recoverable escrow, four directed views per glyph, 168 monads, and an exact structural inverse-alpha invariant of 137. Conditional physical modules construct Planck-scale quantities, define four candidate projection-state grammars, expose the canonical-state selection problem as an explicit formal obligation, and provide an exact interval-certification architecture for the projected value of alpha.

This document serves two purposes. First, it gives scientists, mathematicians, and formal methods researchers a compact public entry point into the project. Second, it acts as the canonical explanatory source for the repository README, module roadmap, glossary, key theorem tour, open-problems program, and dependency diagram.

Contents

1 README: What KTLean Is

1.1 Purpose

KTLean is a formal mathematics repository written in Lean 4. It encodes a growing portion of Kosmoplex Theory as explicit definitions, structures, theorems, computational certificates, and verified dependency chains.

The purpose of the repository is not merely to translate mathematical prose into formal notation. Its purpose is to force every claimed transition to appear as one of the following:

1. a definition;
2. a theorem proved from earlier definitions and theorems;
3. an explicitly stated hypothesis;
4. an explicitly visible computational certificate; or
5. an explicitly named open obligation.

This separation is central to the project. A formal corpus becomes scientifically useful when a reader can distinguish what has been constructed, what has been proved, what has been computed, what has been assumed, and what remains open.

1.2 Repository and cloning instructions

The public repository is located at:

github.com/KosmoNexus/KTLean

To clone and build the project on macOS or Linux:

```
git clone https://github.com/KosmoNexus/KTLean.git
cd KTLean
lake build
```

A first build may download Mathlib and related dependencies. This can take time depending on network speed and storage performance. Later builds are normally much faster because the dependency cache is already present.

To confirm that the clone is clean and synchronized:

```
git status
```

To check that the source tree contains no unfinished proof placeholders:

```
grep -RniE '\bsorry\b|sorryAx' KTLean --include='*.lean'
```

To count theorem and lemma declarations:

```
grep -RhoE \
  '^[[:space:]]*(theorem|lemma)[[:space:]]+[A-Za-z0-9_]+' \
  KTLean --include='*.lean' | wc -l
```

A clean reproducibility check is stronger than a normal local build. Clone the repository into a new directory or independent drive and run `lake build` there. A successful result demonstrates that the project does not depend on hidden local files from the working development directory.

1.3 Current verified checkpoint

At the present checkpoint:

- the repository contains 1,992 theorem and lemma declarations;
- `lake build` completes successfully;
- the source tree contains no `sorry` or `sorryAx`;
- the main branch is synchronized with the public repository;
- the principal capstone theorems have been examined with `#print axioms`;
- the audit reports standard Lean foundations such as `propext`, `Classical.choice`, and `Quot.sound`, together with explicitly visible `native_decide` certificates for selected finite computations;
- no empirical constant, canonical projection count, or decimal alpha value is concealed as an undeclared project-specific axiom;
- a fresh clone on an independent USB drive downloaded pinned dependencies and built successfully.

1.4 The formal spine

The present structural theorem chain can be summarized as

triadic closure $\longrightarrow$ 42 glyphs $\longrightarrow$ spinorial boundary $\longrightarrow$ visible projection plus escrow $\longrightarrow$ 4 directed views $\longrightarrow$ 16

A second, conditional physical chain extends this structure:

$(\hbar, c, G) \longrightarrow (m_P, \ell_P, t_P) \longrightarrow$ four candidate projection depths $\longrightarrow$ canonical-state obligation $\longrightarrow$ certified projecte

The distinction between these two chains matters. The first is the unconditional finite structural spine collected in

`MainTheoremChain.ktlean_structural_spine`.

The second is conditional on explicit positive physical normalizations and, at the canonical-state frontier, on an explicit selection criterion and uniqueness certificate.

1.5 What the repository does not yet claim

The present corpus does not claim:

- a completed derivation of every Standard Model parameter;
- a proved unique physical criterion selecting the canonical projection-state count;
- a kernel-only replacement for every use of `native_decide`;
- a complete formal reconstruction of Bell's theorem or the Tsirelson bound within the KT framework;
- a complete Hurwitz classification theorem for the octonionic boundary;
- a completed derivation of the measured decimal value of α^{-1} without an explicit interval certificate.

These are not hidden gaps. They are identified research boundaries and are stated explicitly so they can become the subjects of future formal work.

2 Module Roadmap

The repository contains many modules. The useful way to enter it is by architectural layer rather than alphabetically.

2.1 Foundations and closure

`Axioms.lean`	Defines the foundational closure framework and proves that canonical triadic completion satisfies Axiom 6 through `completion_satisfi`
`PrimitiveRecurrence.lean`	Separates primitive recurrence from ordinary arithmetic primality and identifies the double-cover boundary.
`SpinorIrreducibility.lean`	Introduces the additional indecomposability condition required to pass from primitive recurrence to a prime-mode conclusion.

2.2 Glyph construction and the canonical spectrum

`GlyphSpectrum.lean`	Provides the canonical symbolic registry of 42 spectral values.
`GlyphSpectralFibonacci.lean`	Reconstructs the golden-ratio pair from the additive triadic stream.
`GlyphSpectralExponentialLimit.lean`	Reconstructs the Euler pair through the factorial/exponential stream.
`GlyphSpectralWallisLimit.lean`	Reconstructs the circle pair through the Wallis stream.
`GlyphSpectralTranscendentalGenerators.lean`	Assembles the six-value transcendental-generator family.
`GlyphSpectralLogarithmicBasin.lean`	Registers the logarithmic basin.
`GlyphSpectralTrigonometricOperations.lean`	Registers the trigonometric operation block.
`GlyphSpectralSpecialFunctionOperations.lean`	Registers the special-function operation block.
`GlyphSpectralSpinorialBoundary.lean`	Registers the spinorial boundary family and its two-sheet recurrence behavior.
`GlyphSpectralCompleteEmergence.lean`	Assembles the seven six-element families into the exact canonical 42-glyph spectrum.
`GlyphSpectralAlphaBoundary.lean`	Counts the usable projection channels and proves the exact structural value 137.

2.3 Fano, quaternionic, and routed algebra

`BraidedQuaternion.lean`	Formalizes braided quaternion multiplication and its relation to routed Cayley–Dickson multiplication.
`CayleyDicksonQuaternion.lean`	Connects routed multiplication with the Cayley–Dickson construction.
`YangBaxterRouting.lean`	Proves the Yang–Baxter relation for routed exchange.
`FanoRecovery.lean`	Distinguishes exact recovery of Fano incidence from residual path information not fixed by the recovered exterior structure.
`OperationNormalForm.lean`	Classifies lawful framed operations by a unique normal form associated with the 42 coordinate triples.
`HurwitzBoundary.lean`	Certifies dimensions and the quaternion-doubling boundary of the constructed carrier; it does not claim the full Hurwitz classification theorem.

2.4 Projection, directed events, and monads

`OMBTProjectionInterface.lean`	Defines visible projection, escrow, decomposition, and reconstruction.
`OMBTLocalityGeneration.lean`	Formalizes locality as a non-injective visible quotient whose hidden information remains recoverable through escrow.
`OMBTDirectedEvent.lean`	Defines the four directed information views.
`OMBTMonadEmergence.lean`	Combines 42 glyphs with four directed views to obtain exactly 168 monads.
`OMBTMonadDynamics.lean`	Adds transformations and dynamical structure on the monad space.
`MonadOrbit.lean` and `MonadProjection.lean`	Formalize orbit factorization and scalar projections constant on orbit classes.
`RoutedResidueSensitive.lean`	Formalizes quasi-stochastic visible behavior arising from deterministic complete dynamics with hidden route residue.

2.5 Memory escrow and reversible tokenization

`MemoryEscrow.lean`	Proves exact reconstruction of initial states from retained transition history.
`MemoryEscrowRouted.lean`	Proves that a complete routed state is exactly its visible state together with escrow.
`RoutedTokenization.lean`	Defines a reversible encoding between complete routed states and token states and proves recovery through token history.

2.6 Dimensional physical tokens

PhysicalTokenInterface.lean Defines dimensional signatures for action, causal propagation, gravity, and dimensionless interaction.

PhysicalActionToken.lean Defines the action token and $\hbar = h/(2\pi)$ under explicit positive normalization.

PhysicalCausalToken.lean Defines the causal token $c = \Delta x/\Delta t$ under positive interval assumptions.

PhysicalGravityToken.lean Defines the gravity token $G = \Gamma/\Sigma$ under positive scale assumptions.

PhysicalPlanckScale.lean Constructs

$$m_P = \sqrt{\frac{\hbar c}{G}}, \qquad \ell_P = \sqrt{\frac{\hbar G}{c^3}}, \qquad t_P = \sqrt{\frac{\hbar G}{c^5}},$$

proves positivity and exact square identities, and preserves token provenance.

2.7 Alpha and projection-state certification

PhysicalAlphaInvariant.lean Proves the exact structural relation

$$\binom{8}{4} = 70, \qquad 2 \cdot 70 = 140, \qquad 140 - 3 = 137,$$

and hence $\alpha_{8D}^{-1} = 137$.

PhysicalAlphaProjectionCorrection.lean Defines the exact symbolic projected inverse-alpha correction using canonical spectral terms.

PhysicalProjectionState.lean Defines four positive dimensionless candidate depth grammars: length, area, volume, and action depth.

PhysicalCanonicalStateCount.lean Separates candidate generation from canonical selection. It formalizes a criterion, a uniqueness certificate, and the open `CanonicalStateCountObligation`.

PhysicalAlphaCertifiedValue.lean Defines exact alpha and inverse-alpha expressions and an interval-certificate architecture without asserting an uncertified decimal.

2.8 Audit and navigation

MainTheoremChain.lean Collects the principal stages into named capstones and distinguishes the unconditional structural spine from conditional physical stages.

AxiomAudit.lean Uses `#print axioms` on selected load-bearing and terminal theorems. It is the canonical record of logical dependencies.

3 Glossary

Term	Meaning in KTLean
Axiom 6	Triadic closure: the canonical completion relation closing a compatible pair into its third state.
Balanced ternary	The three-state alphabet $\{-1, 0, +1\}$ underlying the closure language and directed-state interpretation.
Canonical spectrum	The ordered registry of 42 symbolic spectral values represented by `GlyphSpectrum.values`.
Canonical-state obligation	The explicit unclosed proposition asserting that some criterion uniquely selects one of the four candidate projection-state grammars with count greater than one.
Certificate	A data structure carrying a proposition together with the evidence required to use it formally.
Complete state	The full state before visible projection, including information absent from the local visible image.
Directed view	One of four directional information states associated with each glyph.
Escrow	The retained complement of visible projection needed to reconstruct the complete state.
Fano recovery	Reconstruction of Fano incidence from routed algebraic data, separated from route residue not determined by recovered incidence.
Glyph	One of the 42 canonical spectral-operation states.
Kernel-checked proof	A theorem whose proof term is verified directly by Lean's trusted kernel.
Locality	The visible quotient generated by projection. In KTLean, locality is not identified with complete information loss because visible state plus escrow is reconstructive.
Monad	A glyph paired with one of four directed views. The formal cardinality is $42 \times 4 = 168$.
Native decision certificate	A finite computation discharged by `native_decide`. It is visible in an axiom audit and is not `sorry`.
Normal form	A unique canonical representative for a lawful framed operation.
OMBT	The projection and routing layer connecting complete pre-geometric states, visible local states, escrow, directed events, and monads.
Planck token system	The dimensional construction generated from positive action, causal, and gravity normalizations, yielding Planck mass, length, and time.
Projection depth	A positive dimensionless ratio comparing a macroscopic normalization with a Planck-scale token.
Provenance	A theorem-level record showing that a constructed physical quantity descends from the relevant monadic or token normalization rather than being inserted as an unexplained numeral.
Quasi-stochastic	Visible behavior that is not closed or injective even though the complete dynamics are deterministic.
Route residue	Information about the path or routing history not determined by the visible or recovered exterior structure alone.
Spinorial boundary	A two-sheet recurrence structure in which one step reaches the deck transformation without returning, while two steps return.

Term	Meaning in KTLean
Structural alpha invariant	The exact combinatorial inverse-alpha value 137 established before the 4D projection correction.
Tokenization	A reversible encoding of mathematical state into a dimensional physical-token representation.
Visible projection	The generally non-injective map from complete state to locally observable state.

4 Key Theorem Tour

4.1 Triadic closure

```
completion_satisfies_axiom6
```

This theorem verifies that canonical triadic completion satisfies the sixth Kosmoplex axiom. The axiom audit reports no additional dependencies.

4.2 Canonical 42-glyph emergence

```
GlyphSpectralCompleteEmergence
  .complete_canonical_glyph_spectrum_emerges
```

This capstone assembles seven six-element spectral families, proves their canonical registration, proves total length 42, identifies the assembled list with `GlyphSpectrum.values`, and carries the spinorial boundary properties.

4.3 Spinorial double cover

```
GlyphSpectralCompleteEmergence
  .completed_spectrum_retains_spinorial_double_cover
```

This theorem establishes

$$\text{ReachesDeckAt}(1), \qquad \neg\,\text{ReturnsAt}(1), \qquad \text{ReturnsAt}(2).$$

The structure is therefore spinorial before ordinary locality is imposed.

4.4 Locality with recoverable escrow

```
OMBTLocalityGeneration
  .locality_is_projection_not_information_loss
```

This theorem packages four facts: the visible projection has residue; the visible observation map is non-injective; decomposition into visible state and escrow is injective; and reconstruction returns the complete source state.

4.5 Exactly 168 monads

```
OMBTMonadEmergence.ombt_monads_emerge
```

This theorem connects the canonical glyph spectrum to the directed-event layer:

$$42 \text{ glyphs} \times 4 \text{ directed views} = 168 \text{ monads}.$$

4.6 Exact structural inverse alpha

```
PhysicalAlphaInvariant
  .alphaInverseMagnitude_eq_oneThirtySeven
```

The structural count is

$$\binom{8}{4} = 70, \qquad 2\binom{8}{4} = 140, \qquad 140 - 3 = 137.$$

This is the exact 8D structural invariant, not yet the full projected measured value.

4.7 Planck-scale emergence

```
PhysicalPlanckScale.physical_planck_scale_emerges
```

Given explicit positive normalizations for action, causal propagation, and gravity, this theorem constructs unique positive Planck mass, length, and time and proves their exact square identities.

4.8 Projection-state grammar

```
PhysicalProjectionState
  .physical_projection_state_grammar_emerges
```

This theorem proves that four candidate dimensionless state counts are well formed and positive:

$$\frac{L}{\ell_P}, \qquad \frac{A}{\ell_P^2}, \qquad \frac{V}{\ell_P^3}, \qquad \frac{S}{\hbar}.$$

It deliberately does not declare one candidate canonical.

4.9 Canonical-state frontier

```
PhysicalCanonicalStateCount
  .canonical_state_count_follows_from_certification
```

Once a criterion and uniqueness certificate are supplied, the canonical state count and alpha-compatible projection state follow. The unresolved task remains visible as `CanonicalStateCountObligation`.

4.10 Certified projected alpha

```
PhysicalAlphaCertifiedValue.certified_projected_alpha
```

This theorem closes the architecture of exact symbolic evaluation and interval certification. It does not assert the decimal value 137.035999143 without a supplied certificate.

4.11 Omnibus structural theorem

```
MainTheoremChain.ktlean_structural_spine
```

This is the principal navigational capstone. It collects triadic closure, the exact 42-glyph spectrum, the spinorial two-sheet boundary, locality as visible quotient with escrow, four directed views, 168 monads, and the exact structural inverse-alpha value 137.

5 Open Problems

OP1. Canonical projection-state selection. Construct a physically and mathematically motivated criterion and prove that exactly one of the four candidate projection grammars satisfies it with state count greater than one.

OP2. Exact interval certification of projected alpha. Provide rational lower and upper bounds for the exact symbolic expression tightly enough to certify the claimed decimal expansion.

OP3. Axiom minimization and evaluator hardening. Determine which remaining `native_decide` certificates can be replaced economically by kernel-reducible proofs or explicit finite lemmas.

OP4. Axiom independence. Construct alternate models satisfying selected subsets of the Kosmoplex axioms to determine whether each axiom is independent.

OP5. Uniqueness of the 42-glyph realization. Characterize the canonical spectrum uniquely among structures satisfying an abstract collection of closure, spectral, and routing properties.

OP6. Abstract characterization of the 168 monads. Prove a uniqueness or universal property for the monad object beyond its present product construction.

OP7. Full Hurwitz boundary theorem. Extend the present dimensional and quaternion-doubling results into the relevant formal classification, including octonionic nonassociativity.

OP8. Octonions as braided quaternions. Complete the formal equivalence between braided quaternion construction, routed Cayley–Dickson multiplication, and octonionic multiplication.

OP9. Bell and Tsirelson reconstruction. Formalize the distinction between Bell-local hidden variables and hidden pre-geometric generative structure, and derive the relevant correlation boundary.

OP10. Projection-induced quasi-stochasticity. Generalize the existing examples into an abstract theorem about deterministic reversible complete dynamics under non-injective observation.

OP11. Spin-statistics bridge. Connect the pre-geometric spinorial double cover to a formal spacetime spin-statistics statement.

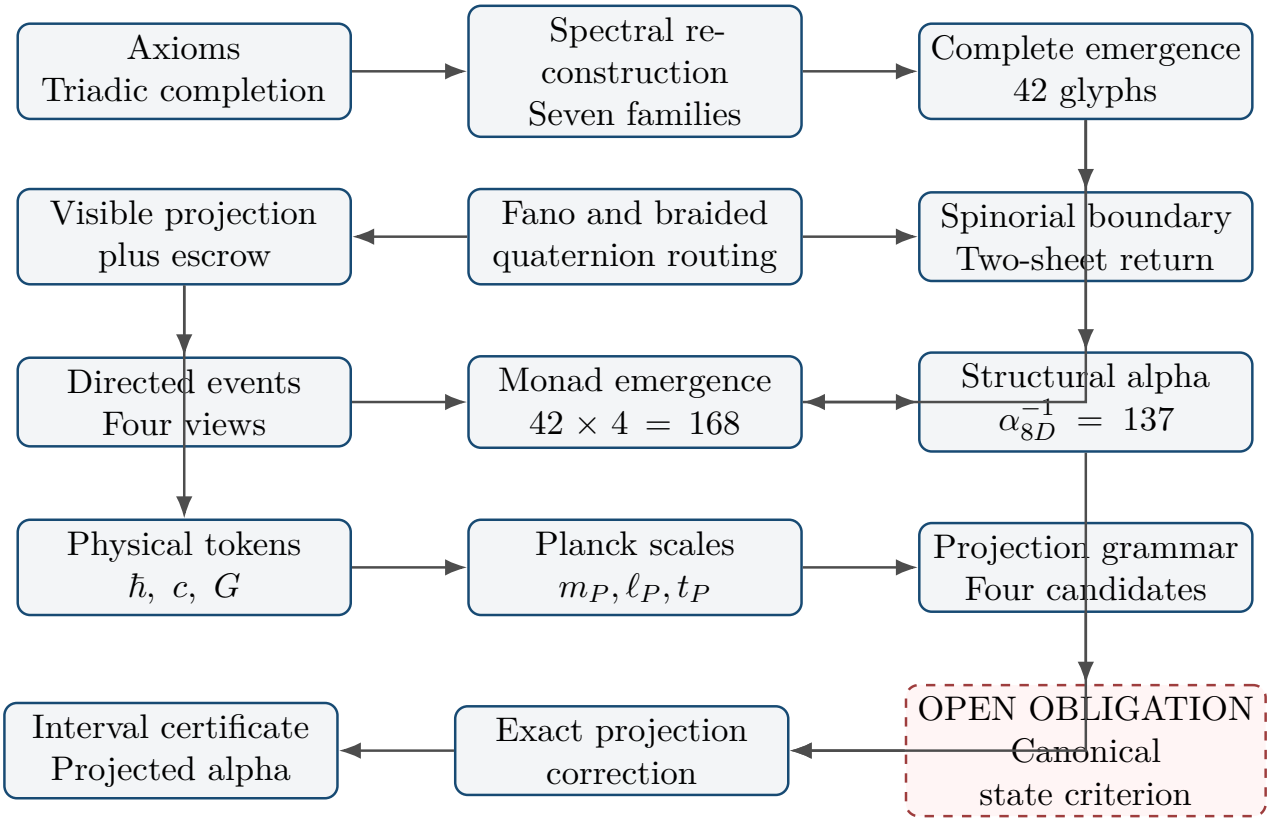
OP12. Dimensional-token universality. Characterize when a reversible mathematical encoding admits a unique dimensional-token interpretation.

OP13. Additional dimensionless constants. Extend the structural invariant, projection correction, state-selection, and interval-certificate method beyond alpha.

OP14. Standard Model parameter program. Build a machine-auditable dependency graph classifying parameters as derived, conditional, fitted, or still external.

OP15. Independent reimplementation. Encourage an independent team to reconstruct the main theorem chain from the published definitions without copying proof scripts.

6 Dependency Diagram



Solid boxes denote formalized modules or theorem families. The dashed red box is intentionally open. The diagram therefore shows both what has been proved and where the current derivational frontier lies.

The main structural theorem terminates at the exact structural invariant 137. The projected-alpha path continues only after a canonical-state certificate is provided. This is the central claim-status distinction in the present repository.

7 How to Participate

The repository is intended to become a research object rather than a closed personal archive. Productive contributions include:

- independent audits of definitions and theorem statements;
- replacement of finite native certificates by clearer kernel proofs where worthwhile;
- construction of alternate models for axiom-independence work;

- formalization of the open canonical-state criterion;
- exact interval certification of projected physical constants;
- extension of the routed algebra, octonionic, Bell/Tsirelson, and spinorial programs;
- improvements to documentation, dependency visualization, and reproducibility testing.

Researchers should begin with `MainTheoremChain.lean`, inspect `AxiomAudit.lean`, then follow the module roadmap in this guide. Questions and proposed contributions should refer to exact theorem or module names whenever possible.

Public repository

https://github.com/KosmoNexus/KTLean

Project website

https://theKTProject.org