THE ZERO PARADOX

ZP-R Addendum: The Diagonal Family

Initial: July 2026 | Current: 1.25, November 2026

Addendum to ZP-R. ZP-R treats the framework's self-application fixed point as a role filled per face: in the fork / AFA face by ⊥ of the ZPSemilattice (the order-bottom), in the computability face by a Kleene code, a term of another type; it locates the face where the role is filled by a Lawvere fixed point. In this addendum, "at ⊥" and "tied to ⊥" refer to that role, not to one object. The addendum maps the faces formalized here, organized by the μ/ν fork — the wall faces where self-reference cannot close (no fixed point) and the floor faces where it does (a fixed point exists; for Löb / Gödel 2nd, from a hypothesized Löb diagonal). Every entry except the Gödel-first row carries a machine-checked Lean 4 witness. The diagonal-family unification is Lawvere (1969) / Yanofsky (2003), cited; the contribution is the formalization and the tie to the self-application role (the engine, the wall faces, and the Quine-atom and Löb / Gödel 2nd floor faces choice-free; the computability floor faces choice-carrying); abstract Löb and Gödel 2 are also formalized by Popescu & Traytel (Isabelle, 2019) and Saitou & Noguchi (Lean 4, arXiv:2609.13780). It supersedes the earlier private "Zero as a Wall" working draft.

The diagonal arguments of Cantor, Russell, Gödel and Tarski are special cases of one theorem, Lawvere's fixed-point theorem (Lawvere 1969, whose introduction names exactly those four). Yanofsky (2003) carries the same scheme to the halting problem and to Kleene's recursion theorem, with Rice's theorem as an application. ZP-R asks where the framework's self-application role is filled by an instance of that fixed point, and finds one face: the computability face, where the occupant is a Kleene code, a term of another type than ⊥ of the ZPSemilattice. This addendum lays out the faces formalized here in one place, each with its Lean witness where one exists.

The organizing principle is the μ/ν fork (ZP-R §I; ZP-P). The engine is negation, which has no fixed point, and Lawvere's theorem, with Yanofsky's extensions, turns that one fact into the wall faces. One direction is proved about the fork's two regimes: a self-loop rules out well-foundedness (mu_nu_branch_exclusion). The wall (μ) faces are Cantor, Russell, Turing, Tarski and Curry; the floor (ν) faces are the Quine atom, the Kleene quine, the Löb sentence (from a hypothesized Löb diagonal), and (with a twist) Rice. Most of the engine and the classical faces are already formalized in the framework's Wall module; this addendum adds the four faces formalized in the diagonal-family build (Tarski, Curry, Rice, Löb).

Section I: The Engine

The root of the wall faces is one fact: negation has no fixed point. No proposition p satisfies p ↔ ¬p. Lawvere's theorem is the amplifier: a point-surjection e : A → (A → B) forces every f : B → B to have a fixed point, so a fixed-point-free f (negation) refutes the surjection. Every wall face below is this contrapositive.

The engine (Wall.lean)
negation_no_fixedpoint: ¬ (p ↔ ¬ p) — negation is fixed-point-free (the root).

The engine (Wall.lean)
lawvere_fixedpoint: a point-surjection A → (A → B) forces every f : B → B to have a fixed point (Lawvere 1969, the general engine).
Lean purity: choice-free. ✓

Section II: The Wall Faces (μ) — self-reference cannot close

On a well-founded relation there is no self-loop: no x with x pointing at itself (wf_no_selfloop). Under Foundation this is "no set is self-membered."

The wall, general (Wall.lean)
wf_no_selfloop: a well-founded relation admits no self-loop — the object-level core the metatheoretic boundary reduces to. no_quine_atom: under Foundation, no set is self-membered.
Lean purity: wf_no_selfloop choice-free; no_quine_atom [propext, Quot.sound]. ✓

Cantor / Russell / Turing (Wall.lean) — the classical wall faces
Cantor (cantor_via_engine): no g : A → (A → Prop) is surjective — no self-surjection onto predicates.
Russell (russell_via_engine): no membership relation realizes every predicate (naive comprehension is impossible).
Turing (no_self_decider): no g : A → (A → Bool) is surjective — the halting diagonal flips the decider.
Placement is by witness, not by phenomenon: no_self_decider is the Cantor-over-Bool skeleton of the halting argument and sits on the wall. The halting problem as an undecidability fact (halting_undecidable, Rice.lean) is an instance of Rice's theorem and sits with the Rice face in Section III.
Lean purity: choice-free (all three, off the engine). ✓

Tarski (Tarski.lean) — the truth face
tarski_no_internal_truth: no internal universal truth-naming exists — truth is not definable inside a system that can name all its own predicates (Tarski, conventionally dated 1936). The liar sentence (tarski_liar_from_naming) is the explicit diagonal; the T-schema at the liar is absurd (tarski_Tschema_liar_absurd).
The bottom relationship: no floor — tarski_no_truth_bottom (¬ HasLawvereWitness Prop). Truth is the wall dual of Gödel's provability.
Lean purity: choice-free. ✓

Curry (Curry.lean) — the explosion face
curry_paradox: a self-referential p ↔ (p → C) proves C, for ANY C — a strict generalization of the engine (the liar is the C = False case, liar_is_curry_at_false). curry_from_naming: a surjective internal comprehension proves everything.

Curry (Curry.lean) — the explosion face

The bottom relationship: no floor, and pretending otherwise explodes — curry_no_bottom (¬ Surjective naming, via explosion at C = False).

Lean purity: choice-free. ✓

Section III: The Floor Faces (ν) — self-reference closes

At each floor face a fixed point is supplied, and the supplier differs by face. In the fork / AFA face it is ⊥ of the ZPSemilattice, by the class fields (a commitment, not engine output). In the computability face it is a Kleene code (Rogers' fixed point). In the Löb face it is given by a hypothesized Löb diagonal: the Lean takes it as the hypotheses hfix1, hfix2, and in arithmetic the diagonal lemma supplies it (ZeroParadox/Settheory/Loeb.md).

The Quine atom (SetTheoryAFA.lean) — the set-theoretic floor

The Quine atom (CC-2) is a role: a point that steps to itself and to nothing else (Motion.SelfOnlyAt, ZeroParadox/Die.lean), whose occupancy is settled chart by chart, with no equation across charts. In the lattice chart its occupant is ⊥ of the ZPSemilattice, under a self-application map supplied by AbstractSelfApp (selfApp_quine_role_iff) and in the QuineAtomHost an AFAStructure on L builds (quineAtomHost_quine_role_iff with afaStructure_isQuineAtomHost); t_exec / T-EXEC (t_exec_iff): every Quine atom equals ⊥ of the ZPSemilattice (IsQuineAtom q ↔ q = ⊥), derived from the ZPSemilattice / AFAStructure class fields — location and uniqueness of the floor in this face. In Mathlib's ZFSet, where the role reads x = {x} (zfset_selfOnlyAt_mem_iff), it has no occupant (zfset_quine_role_unoccupied; walled off under Foundation, Section II). In the final coalgebra of the one-child polynomial functor it is occupied (cofix_quine_role_occupied), but that chart has one point (cofix_subsingleton), so the occupancy restates the existence fork and adds nothing further.

Lean purity: t_exec choice-free. ✓

The Kleene quine (Lawvere.lean) — the computability floor

computability_face_fixedPoint: every computable self-map on codes has a fixed point up to eval — two codes computing the same function, not a literal one — Rogers' form of the recursion theorem (Mathlib fixed_point), on the reflexive object of the computable category (the crossing realized in ZP-R). A program that prints its own code takes one further s-m-n step, not in this declaration. The fixed point is machine-checked here; that it is an instance of Lawvere's theorem is cited: Yanofsky (2003), Theorem 5, and Bauer (2017), Theorem 5.2 and Corollary 5.3, a version of Lawvere's theorem for multi-valued maps, in synthetic computability.

Lean purity: [propext, Classical.choice, Quot.sound] — Mathlib recursion theory. ✓

Löb (Loeb.lean) — the provability floor, given a Löb diagonal; and Gödel's second incompleteness
loeb: given a Löb diagonal ($\psi \leftrightarrow (\Box\psi \to A)$), $\vdash \Box A \to A$ yields $\vdash A$. Given such a diagonal, ψ is a fixed point up to provability (loeb_sentence_is_fixedpoint).
godel_two: given a Löb diagonal for falsum, a consistent system does not prove $\Box\text{falsum} \to \text{falsum}$ — its own consistency (Gödel's second incompleteness, the A = falsum corollary). The consistency reading is Gödel's second theorem when the class is instantiated at a theory of arithmetic and falsum is its falsum; in the class, godel_two binds falsum as an arbitrary sentence (ProvabilityLogic has no falsum constant and no ex-falso), so it covers any unprovable sentence with a Löb diagonal.
Two readings of the face: it is filed with the floors because the diagonal closes; the unprovability of consistency is a limitation, which ZeroParadox/Settheory/Loeb.lean calls the wall on the provability axis, in another sense than the μ wall faces of Section II.
Built from scratch (no modal or provability logic located in the pinned Mathlib, searched 2026-08-30 on three axes; ZeroParadox/Settheory/Loeb.lean; prior art for abstract provability: Popescu & Traytel, CADE-27 (2019), and their Archive of Formal Proofs entry "An Abstract Formalization of Gödel's Incompleteness Theorems", in Isabelle; Saitou & Noguchi, "Mechanizing Gödel's Incompleteness Theorems and Provability Logic", arXiv:2609.13780, in Lean 4): an abstract ProvabilityLogic typeclass with the Hilbert–Bernays–Löb derivability conditions.
Lean purity: choice-free. ✓

Rice (Rice.lean) — a fixed point beside an undecidable property
quine_exists_yet_rice is a conjunction: for computable f, a fixed point of f up to eval EXISTS (recursion theorem), AND membership in any non-trivial extensional set C of codes is UNDECIDABLE (rice_face, which cites Mathlib's ComputablePred.rice$_2$, Rice's theorem in its form over program codes; Rice 1953 states it for classes of recursively enumerable sets). The undecidability is of C over all codes; the second conjunct does not mention f, so nothing is stated about membership at f's fixed point. The standard recursion-theorem proof of Rice's theorem (Yanofsky 2003, p. 19; Mathlib's ComputablePred.rice, via fixed_point$_2$) turns on the fixed point of a different map, built from the assumed decider for C. Rice's 1953 proof takes another route: his undecidability result, the Corollary B that follows his Theorem 6, rests on his Theorems 4 and 6, and Theorem 6 reduces from Theorem 5, that the unit class of the empty set is not completely recursively enumerable. rice_face_has_bottom states the first conjunct alone; reading that fixed point as the face's floor is the family's criterion, not the theorem.
halting_undecidable: whether a program halts on input n is not decidable, the concrete Rice instance (Mathlib's halting_problem, proved by rice).
Reading: the pivot face. The fixed point is not refuted (as on the wall) but present (ν), and undecidability sits beside it; no theorem here makes one depend on the other.
Lean purity: [propext, Classical.choice, Quot.sound] — Mathlib recursion theory. ✓

Section IV: The Formalized Faces

The faces formalized here, in one table. Gödel's first incompleteness ($G \leftrightarrow \neg\text{Prov}(G)$) sits between the columns and is not formalized here: a first-incompleteness declaration was not located as of 2026-10-06

(searched ZeroParadox/**/*.lean for first-incompleteness and Gödel-sentence declarations, by name and by statement shape), and lawvere_fixedpoint says nothing about provability. Every row except the Gödel-first row is a machine-checked Lean witness.

Variant	Face	What self-reference does	Lean witness
the engine	—	negation has no fixed point (the root)	negation_no_fixedpoint, lawvere_fixedpoint (Wall.lean)
Cantor	μ	no self-surjection onto predicates	cantor_via_engine (Wall.lean)
Russell	μ	no membership realizes every predicate	russell_via_engine (Wall.lean)
Turing	μ	the halting decider is flipped	no_self_decider (Wall.lean)
Tarski	μ	truth is not internally definable	tarski_no_internal_truth (Tarski.lean)
Curry	μ	self-reference proves everything (explosion)	curry_paradox, curry_no_bottom (Curry.lean)
Gödel 1st	between	self-reference is true-but-unprovable	not formalized here
Quine atom	ν	the Quine-atom role (a point that steps to itself and to nothing else), occupied by ⊥ of a ZPSemilattice	t_exec / t_exec_iff (SetTheoryAFA.lean); selfApp_quine_role_iff (QuineDichotomy.lean)
Kleene quine	ν	every computable self-map on codes has a fixed point up to eval	computability_face_fixedPoint (Lawvere.lean)
Löb / Gödel 2nd	ν	the provability diagonal closes given a Löb diagonal ψ (hypotheses hfix1, hfix2); consistency unprovable, for a consistent system given a Löb diagonal for falsum (Gödel's second theorem when the class is instantiated at a theory of arithmetic and falsum is its falsum; in the class, falsum may be any sentence)	loeb, loeb_sentence_is_fixedpoint, godel_two (ZeroParadox/Settheory/Loeb.lean)
Rice	ν	a fixed point exists; every non-trivial semantic property is undecidable over all codes	rice_face_has_bottom, quine_exists_yet_rice, halting_undecidable (Rice.lean)

The map is a placement, not a new theorem. The unification of the diagonal family is Lawvere (1969) and Yanofsky (2003); each face is a classical result (each wall face a direct instance of the engine), formalized here choice-free except the computability floor faces (Kleene, Rice), which carry Classical.choice inherited from Mathlib's recursion theory. What the framework adds is the tie to the self-application role, for the faces formalized here.

Fence: the cross-face identity is a type boundary

The faces are the same relationship, not the same object. The Quine atom (a set), the Kleene quine (a code), the Löb sentence — these are terms of different types in different categories; "x = y" across them is not false, it is not well-formed (ZP-P hard fence; ZP-R F2). What is claimed is that each face formalized here forks at its own self-referential contact point, and that the self-application fixed point is a role filled per face by that face's own occupant: in the fork / AFA face, ⊥ of the ZPSemilattice, by the AbstractSelfApp class fields fixed_bot and unique_fp (a commitment; Lawvere's premise is false in any ZPSemilattice with an element other than ⊥, nontrivial_lattice_no_witness, which unlike t_exec carries Classical.choice); in the computability face, a Kleene code, a fixed point up to eval (that it is a Lawvere instance is cited, Rogers' form); in the Löb face, the sentence a hypothesized Löb diagonal supplies. On the wall faces the role has no occupant. No occupant is identified with another. How the wall, fork / AFA and computability faces relate to Lawvere's theorem is collected in ZP-R, Section III, in the box "Lawvere's theorem, face by face"; ZP-R conjectures nothing global.

Relation to Prior Work

The diagonal-family unification via one fixed-point theorem is Lawvere, "Diagonal Arguments and Cartesian Closed Categories," LNM 92 (1969) 134–145, read in its reprint, Reprints in Theory and Applications of Categories 15 (2006); and Yanofsky, "A Universal Approach to Self-Referential Paradoxes, Incompleteness and Fixed Points," Bull. Symbolic Logic 9(3) (2003), read as arXiv:math/0305282v1. Lawvere's paper treats Cantor, Russell, Tarski's undefinability of truth and an incompleteness theorem; Yanofsky adds, among others, Turing's halting problem, the recursion theorem (his Theorem 5) and Rice's theorem. Bauer, "On fixed-point theorems in synthetic computability," Tbilisi Math. J. 10(3) (2017), proves a version of Lawvere's theorem for multi-valued maps, in synthetic computability, and derives the Kleene–Rogers recursion theorem from it (Theorem 5.2, Corollary 5.3). The individual faces are classical: Cantor; Russell (1901); Gödel (1931); Turing's halting problem; Tarski (1936); Curry (1942); Löb (1955); Kleene's recursion theorem; and Rice, "Classes of recursively enumerable sets and their decision problems," Trans. Amer. Math. Soc. 74(2) (1953) 358–366. The years given for Russell and Tarski are their conventional datings; those primary publications were not read for this addendum. The point-surjective theorem is in Mathlib (Function.exists_fixed_point_of_surjective); the framework's engine re-derives it axiom-free for a self-contained family. The contribution of this addendum is the formalization of the faces formalized here (the engine, the wall faces, and the Quine-atom and Löb / Gödel 2nd floor faces choice-free; the computability floor faces choice-carrying) and its tie to the self-application role — a placement, not an extension.

Axiom Purity

Axiom Purity
The engine and the wall faces (negation_no_fixedpoint, lawvere_fixedpoint, cantor_via_engine, russell_via_engine, no_self_decider, wf_no_selfloop, Tarski, Curry) and the Quine-atom and Löb / Gödel 2nd floor faces (t_exec; Löb, godel_two) are choice-free — the faces listed here need no Axiom of Choice.
The computability floor faces (Kleene: computability_face_fixedPoint; Rice: rice_face_has_bottom, quine_exists_yet_rice, halting_undecidable) carry [propext, Classical.choice, Quot.sound], inherited from Mathlib's classical recursion theory — disclosed, not claimed otherwise.
Core choice-free, computability realization choice-carrying — the framework's standing pattern. Each declaration's footprint is the #print axioms block in its file's PurityCheck section (ZeroParadox/Settheory/Loeb.lean, for example).

End of ZP-R Addendum | The Diagonal Family | wall faces μ (Cantor, Russell, Turing, Tarski, Curry), each the contrapositive of one engine (negation has no fixed point) | floor faces ν (Quine atom, Kleene, Löb / Gödel 2nd from a hypothesized Löb diagonal, Rice), each fixed point supplied per face | the faces formalized here, Lean-witnessed except Gödel 1st | cross-face identity a type boundary | the map is a placement, not a new theorem.