THE ZERO PARADOX

ZP-P: The Fixed-Point Fork

Initial: June 2026 | Current: 1.37, November 2026

Synthesis layer. The abstract fork schema is proved sorry-free and choice-free in Lean 4 (FixedPointFork.lean); the number-system instance via Ostrowski (Ostrowski.lean) and the categorical-parent instance via QPF.Fix / QPF.Cofix (Coalgebra.lean) are Lean-witnessed; the set-theory and computation instances are referenced to ZP-J and ZP-K. Generalizes the ZFC+Foundation / ZFC+AFA orthogonal-contact-point claim.

A monotone self-map of a complete lattice has a least fixed point and a greatest fixed point (Knaster–Tarski); monotonicity is required. Whether the framework's own self-referential operators instantiate that schema is not claimed here — see the fences in Section III. ZP-P records a single elementary fact and develops its consequences across the framework: these two fixed points coincide — the fork collapses to one point — exactly when the operator has a unique fixed point. Read intuitively, the least fixed point is the inductive (well-founded) closure and the greatest is the coinductive (non-well-founded) closure; the collapse point is where the two closures meet. In the Zero Paradox that point is the diagonal fixed point, filled by $\bot$ of the state semilattice in the Quine-atom role (CC-2), the role of a point that steps to itself and to nothing else, whose occupancy is settled chart by chart: under a self-application map that fixes $\bot$ and nothing else it is the only occupant (selfApp_quine_role_iff); in Mathlib's ZFSet no set occupies the role (zfset_quine_role_unoccupied).

This layer is a synthesis layer, not a foundational one: it adds no axiom or primitive. It names a pattern that several existing layers already realise, and it is organised in three tiers held to three different standards of proof. Tier 1 (Section I) is the abstract schema, proved in Lean over a complete lattice. Tier 2 (Section II) is the instance catalogue: each framework forks at its own contact point, and the instances are theorems in their home layers. Tier 3 (Section III) is the unification, stated as a role rather than an identity: one role, defined once, against which each chart either has an occupant or provably has none, and no equation across charts is claimed. The fences in Section III are load-bearing: keeping the tiers separate is what makes the formal content honest.

Section I: The Fork Schema (Tier 1)

Over a complete lattice, a monotone self-map f has a least fixed point (lfp f) and a greatest fixed point (gfp f), and lfp f $\leq$ gfp f always. The width between them is the fork. The schema theorem states that the fork collapses to a single point precisely when f has a unique fixed point — and in that case the common value is that fixed point. The proofs rest only on Mathlib's order-theoretic fixed-point API (OrderHom.lfp / OrderHom.gfp); they make no claim about the categorical inductive/coinductive reading, which is offered as analogy only.

Definition: the fork (FixedPointFork.lean)

For a complete lattice α and a monotone self-map $f : \alpha \to \alpha$:

Definition: the fork (FixedPointFork.lean)

lfp f = the least fixed point of f (Mathlib OrderHom.lfp)

gfp f = the greatest fixed point of f (Mathlib OrderHom.gfp)

The fork is the ordered pair (lfp f, gfp f), always satisfying lfp f ≤ gfp f.

Collapse = lfp f = gfp f: the two ends meet at a single contact point.

Proposition: fork_le (FixedPointFork.lean)

lfp f ≤ gfp f

The inductive closure never exceeds the coinductive closure: the fork has non-negative width. (Mathlib OrderHom.lfp_le_gfp.)

Lean purity: [propext, Quot.sound] — choice-free. ✓

Lemma: collapse_of_unique (FixedPointFork.lean)

If x is the unique fixed point of f, then lfp f = x and gfp f = x.

Both ends of the fork land on the sole fixed point.

Proof: lfp f and gfp f are fixed points (map_lfp, map_gfp); uniqueness sends each to x.

Lean purity: [propext, Quot.sound] — choice-free. ✓

Lemma: unique_of_collapse (FixedPointFork.lean)

If lfp f = gfp f, then every fixed point of f equals that common value.

Every fixed point lies between lfp f and gfp f, so collapse forces uniqueness.

Proof: lfp_le_fixed and le_gfp bracket any fixed point y; antisymmetry closes it.

Lean purity: [propext, Quot.sound] — choice-free. ✓

Theorem: fork_collapse_iff (FixedPointFork.lean) — the schema spine

lfp f = gfp f ↔ ∃! x, f x = x

The fork collapses to a single contact point if and only if f has a unique fixed point.

This is the abstract form of "the framework forks, and the diagonal fixed point is where the two closures meet."

Proof: collapse_of_unique and unique_of_collapse, both directions.

Lean purity: [propext, Quot.sound] — choice-free. ✓

The fork spine is choice-free. All four results depend only on [propext, Quot.sound] — propositional extensionality and quotient soundness, two of Lean's standard axioms, used throughout Mathlib. None depends on the Axiom of Choice. This is consistent with the choice-free core (see ZeroParadox/AxiomProfile.lean): the conceptual skeleton needs no choice; choice enters mostly in the analytic realisations (that file's § III) — not only there, since the framework also carries bare classicals of its own; ZeroParadox/Category/ChoiceCannotBe.lean § IV collects cases among them whose principles reduce to a taboo, with those reductions.

Section II: The Instance Catalogue (Tier 2)

The same fork appears across frameworks. ZP-P does not re-prove each instance — each is a theorem in its home layer — it records the catalogue and pins the number-system instance to a checkable Lean witness. The original ZFC+Foundation / ZFC+AFA orthogonal-contact-point claim is the set-theory entry: in the standard category-theoretic characterisation (Aczel; Lambek), Foundation is the initial algebra (μ) and AFA the final coalgebra (ν) of the bounded powerset functor, and the Quine atom (CC-2), a set equal to its own singleton, $x = \{x\}$, is exactly an element present in ν but not in μ. No equation identifies it with $\bot$ of a ZPSemilattice; the two are measured against the Quine-atom role of Section III.

Domain	The two closures fork into	Contact point / home layer
Set theory	Foundation (μ, well-founded) vs AFA (ν, non-well-founded)	Quine atom (CC-2), $x = \{x\}$; ZP-J / ZP-K
Number systems	Archimedean $\mathbb{R}$ vs non-Archimedean $\mathbb{Q}_p$	0: in $\mathbb{R}$ no first step above it (AX-B1's form); $\mathbb{Q}_2$ removes the topological obstruction to an irreversible step (c3_irreversible, a case of Mathlib's TotallyDisconnectedSpace.eq_of_continuous: in $\mathbb{Q}_2$ no continuous path joins any two distinct points, so the step out from 0 is blocked as well) and supplies no first step; ZP-B / ZP-F
Computation	total (well-founded) vs partial (non-terminating)	the Kleene quine / self-application; ZP-K
Category theory	initial algebra (W-types) vs final coalgebra (M-types)	QPF.Fix vs QPF.Cofix; Coalgebra.lean

The number-system instance is the one with a genuine classification theorem behind it. Ostrowski's theorem says the nontrivial absolute values on $\mathbb{Q}$ are exactly the real (Archimedean) one and the p-adic (non-Archimedean) ones — the complete, pairwise-inequivalent list. The contact point is 0: in the Archimedean completion $\mathbb{R}$, a densely ordered field, 0 has no first step above it, AX-B1's form (f_snap_impossible, which, despite its name, states that no linearly ordered field has a least positive element). In the 2-adic completion $\mathbb{Q}_2$, 0 is the unique fixed point of $x \mapsto 2x$ (q2_unique_fp) and the only point of infinite valuation ($v_2(0) = \infty$, padic_addVal_eq_top_iff); $\mathbb{Q}_2$ is totally disconnected (padic_snaps, an instance Mathlib already supplies), which removes the topological obstruction to an irreversible step (c3_irreversible: no continuous path in $\mathbb{Q}_2$ from a nonzero point to 0) and supplies no first step: under the norm preorder 0 is least and has no first step (ZeroParadox/BottomCannotBe.lean § ROLE), and on $\mathbb{Z}_2$, the 2-adic integers, 0 is least in the norm preorder and greatest in the valuation order (z2_two_views). T-SNAP's

shape needs no first step: t_snap_given holds at a step taken in a densely ordered join-semilattice ($\mathbb{Q}_{\geq 0}$ under max, whose $\bot$ is 0; ZeroParadox/Order/Snap.lean).

Theorem: completions_exhaustive (Ostrowski.lean)

Every nontrivial absolute value on $\mathbb{Q}$ is equivalent to the real absolute value, or to a p-adic absolute value for a unique prime p.

The two kinds of completion are the complete list. (Ostrowski's theorem, Mathlib Rat.AbsoluteValue.equiv_real_or_padic.)

Lean purity: [propext, Classical.choice, Quot.sound] — choice-carrying. ✓

Theorem: real_not_equiv_padic (Ostrowski.lean)

The real absolute value is inequivalent to every p-adic absolute value.

The two kinds of completion are genuinely distinct — never the same metric. This is the orthogonality leg of the number-system fork. (Mathlib Rat.AbsoluteValue.not_real_isEquiv_padic.)

Lean purity: [propext, Classical.choice, Quot.sound] — choice-carrying. ✓

Remark: choice-free spine, choice-carrying realisation

The fork spine (Section I) is choice-free [propext, Quot.sound]. The Ostrowski instance carries Classical.choice, inherited from Mathlib's classical analysis and number theory. This split is the framework's standing pattern: the conceptual core is choice-free, while the analytic realisations are choice-carrying.

The number-system fork is theorem-backed on its own terms — Ostrowski is a genuine classification theorem, stronger backing than the metatheoretic Foundation/AFA orthogonality. It is NOT claimed to be an instance of the μ/ν (least-vs-greatest fixed point) schema: Ostrowski concerns absolute values, not fixed points of a functor. The thread to the diagonal fixed point runs through the contact point 0 (the unique fixed point of $x \mapsto 2x$ in $\mathbb{Q}_2$), not through the schema theorem.

The categorical instance is the genuine μ/ν case — the parent of the set-theory entry. On a one-shape, one-child polynomial functor (no leaf), the initial algebra (W-type, μ) is empty while the final coalgebra (M-type, ν) is inhabited by the infinite self-referential element: the non-well-founded closure contains a point the well-founded closure lacks — the categorical analog of the Quine atom present in νF but not μF.

Theorem: categorical_fork_strict (Coalgebra.lean)

IsEmpty (Fix idPF_Coalgebra.Obj) ∧ Nonempty (Cofix idPF_Coalgebra.Obj)

The initial algebra (least fixed point, μ) is empty; the final coalgebra (greatest fixed point, ν) is inhabited. (Mathlib QPF.Fix / QPF.Cofix.)

Theorem: categorical_fork_strict (Coalgebra.lean)

Split footprint: fix_isEmpty (μ empty) is choice-free [propext, Quot.sound]; cofix_nonempty (ν inhabited) carries Classical.choice from the QPF corecursion machinery — a QUOTIENT-LAYER artifact: QPF.Cofix carries the choice in the TYPE. The M-type FORMER and constructors are axiom-free while the M-type DESTRUCTOR (M.children / M.dest) is not — that is where the axiom originates — and strict_cofix_nonempty proves the same inhabitation over the M-type with no axioms, by only building. For a polynomial functor the final coalgebra is constructible choice-free (Ahrens–Capriotti–Spadotti, TLCA 2015). ✓

Remark: where choice appears in the μ/ν fork

The Classical.choice in cofix_nonempty is a library dependency, not a fact about the mathematics, and it is worth locating exactly. Measured: the M-type former and its constructors are axiom-free (PFunctor.M, M.mk, M.corec), while its destructor carries the axiom (M.children, M.dest) — and QPF.Cofix inherits it in the type through the congruence it quotients by. That is why strict_cofix_nonempty is axiom-free: it only builds, and never destructs. For a polynomial functor the final coalgebra is constructible choice-free (Ahrens–Capriotti–Spadotti, TLCA 2015, who construct it as an ω-limit). Their ambient setting is intensional type theory with univalence, but the construction itself uses only function extensionality — univalence enters only for their uniqueness result.

The finite-powerset functor is a different case — it is not polynomial, and the argument above does not cover it. It is also not this document's subject. The per-presentation detail there, with citations, is recorded in the Lean source (Coalgebra.lean) and is deliberately not restated here.

The remaining two instances are referenced, not re-proved here. The set-theory fork (Foundation vs AFA) has its Lean witness in ZP-J: quine_atom_unique proves that any two Quine atoms are equal — a uniqueness statement about self-MEMBERSHIP in the sense $x = \{x\}$, a set that is its own sole member. Bare self-membership is not unique under AFA: $0^* = \{\varnothing, 0^*\}$ also contains itself (Aczel 1988, Example 1.5). That the bottom is one of them is a separate result (bot_is_quine_atom), and self-APPLICATION is a third (selfApp_pinnable, in the experimental Lawvere bridge rather than ZP-J proper). The computation fork (total vs partial) has its Lean witness in ZP-K: the Kleene quine. Each is the contact point of its own fork.

Section III: The Unification and Its Fences (Tier 3)

The tempting claim is that the contact points of all the instances — the Quine atom, the 2-adic 0, the Kleene quine, the self-referential element of the final coalgebra — are faces of one object, the diagonal fixed point. ZP-P does not state that identity. It states what the contact points are measured against as a role, defined once in the corpus's shared vocabulary of step relations (ZeroParadox/Die.lean, called the die; each carrier with its own step relation is a chart): the Quine-atom role, a point that steps to itself and to nothing else (Motion.SelfOnlyAt). Each chart either has an occupant or provably has none, and each is a theorem: the self-referential element of the final coalgebra occupies it (cofix_quine_role_occupied), no well-founded set does (zfset_quine_role_unoccupied), on the 2-adic integers 0 is the only occupant under doubling (z2_double_selfOnlyAt_iff), ⊥ is the only occupant under a self-application map on a semilattice, given as an AbstractSelfApp instance (selfApp_quine_role_iff), and on Kleene's codes, stepping from a code to the codes whose output is what a map assigns it, no single code occupies it while, for a computable map respecting equal output, some class of codes up to equal output does (codes_vs_classes). Under a function step, occupying the role is being a fixed point (Motion.selfOnlyAt_ofFun_iff), so the 2-adic and

self-application rows restate their fixed points; the self-application row holds by the AbstractSelfApp class fields, so the constant map to ⊥ satisfies it too. The final-coalgebra chart has one point, so its occupancy restates its fork and adds nothing further. The fences are not hedging; they mark a genuine boundary, and per-instance forks remain full theorems on either side of them.

Hard fence (permanent): cross-instance identity is not a theorem

The claim "the contact points are one object" is not a theorem and cannot become one. Identity requires a shared type: the 2-adic 0, the Quine atom (a set), the Kleene quine (a code), and an element of a final coalgebra (an object up to isomorphism) are terms of different types in different categories. The proposition "x = y" across them is not false — it is not well-formed. So this is a type boundary, not a missing proof, and no such identity is committed to either. What relates the instances is the one role each chart is measured against. What is claimed formally is that each framework forks at its own contact point, and which charts have an occupant of the role.

Soft fence (conjecture): not every fork is a μ/ν instance

That every domain fork is an instance of the least-vs-greatest fixed point schema is an open generalisation, not established here. The schema is a theorem in this layer and for canonical functors (W-types vs M-types). The number-system fork (Ostrowski) is the clear case that is theorem-backed yet NOT visibly a μ/ν instance — it is the live caution against overstating the generalisation.

Per-instance forks are theorems and are not hedged. The fences scope only the cross-instance identity (Tier 3) and the universal schema claim. Each individual fork — Foundation/AFA, $\mathbb{R}/\mathbb{Q}_2$, total/partial — stands on its own proof.

Theorem Summary

Result	File / Section	Axioms
fork_le	FixedPointFork.lean §I	choice-free
collapse_of_unique	FixedPointFork.lean §I	choice-free
unique_of_collapse	FixedPointFork.lean §I	choice-free
fork_collapse_iff	FixedPointFork.lean §I	choice-free
completions_exhaustive	Ostrowski.lean §II	Classical.choice
real_not_equiv_padic	Ostrowski.lean §II	Classical.choice
fix_isEmpty	Coalgebra.lean §II	choice-free
cofix_nonempty	Coalgebra.lean §II	Classical.choice

Axiom Purity
Fork spine (FixedPointFork.lean): [propext, Quot.sound] — choice-free. The schema needs no Axiom of Choice.
Number-system instance (Ostrowski.lean): [propext, Classical.choice, Quot.sound] — Classical.choice inherited from Mathlib's classical analysis / number theory (Ostrowski).
Categorical instance (Coalgebra.lean): split — fix_isEmpty (μ empty) is choice-free [propext, Quot.sound]; cofix_nonempty (ν inhabited) carries Classical.choice from the QPF corecursion machinery (the quotient layer).
Core choice-free, realisation choice-carrying — the framework's standing pattern.
The footprints listed here are copies; the live record is the #print axioms output of each named .lean file.

End of ZP-P | The Fixed-Point Fork | fork_collapse_iff (choice-free spine) | Ostrowski number-system instance | categorical μ/ν instance (Fix empty / Cofix inhabited) | hard fence: cross-instance identity is a type boundary; what relates the instances is one role, which a point of each chart occupies or provably none does | soft fence: not every fork is μ/ν | All FixedPointFork.lean / Ostrowski.lean / Coalgebra.lean theorems verified; the role occupancies of Section III are verified at the declarations Section III names.