

What you should do now

Yes—large portions of *MASTER CONSOLIDATION DOSSIER.pdf* have already been absorbed into the draft (Executive Summary, Axiom Freedom Statement, Sections 2–3, Appendices). I’ll map dossier elements → paper sections, note what is **already integrated**, what is **partially integrated**, and what is **still absent** (so we can import it next).

1. Core Glossary & Constructors

Your dossier’s mini-glossary introduction (glossary framing) has been distilled into Appendix H (Glossary) and Sections 2–3.

The explanation of primitive constructors (integrate / delta / merge / void) informing our “four-primitive core” is reflected in Section 2.1’s syntax list and the “Cancellation Negation” narrative.

2. Foundational Design Principle (Bool vs Prop quarantine)

The dossier’s articulation that all rewriting decisions live in Bool while semantic theorems live in Prop became the **Axiom Freedom Statement** rationale (showing why no propext / classical axioms are needed).

Granular use of computational booleans (definition of `Bool` as data, evaluation behavior) informed our justification for replacing external logical control with structural normalization.

Your argument that Boolean branching avoids reliance on propositional extensionality (propext) is now paraphrased in the axiom audit subsection.

3. Low-Level Predicate & Equality Mechanisms

`isVoid` and `same` as computable gates are summarized in our internal semantics discussion (they motivate `void` = truth normalization).

4. Avoidance of Classical Axioms

The dossier’s note that you removed classical add-ons (choice, propext) after the Boolean refactor directly underpins the “Axiom Freedom Audit” claims in Section 15.14.

5. Layered Architecture (Syntactic → Computational → Observational → Logical)

Your stratification (constructors → normalize → `isVoid/same` → `Equiv` → propositions) was imported almost verbatim into the structural overview narrative.

6. Kernel Purity & Narrative Blocks

The “Kernel Purity Statement” wording is echoed in our Executive Summary and Appendix B (Boolean Replacement) emphasizing normalization as sole semantic engine.

7. Connective Encodings / Cancellation Logic

The dossier examples of encoding implication and biconditional via `merge` and `neg match` Section 3 (Negation as Cancellation) and Appendix B’s logical law table.

8. Dual Normalizers & Equivalence

Your structural vs Boolean normalizer equivalence theorem is represented in our discussion of “structural kernel variant vs engineered evaluator” (we asserted equivalence and used it in the independence argument).

9. Defensive Audit / Attack→Counter Matrix

The Attack→Counter matrix items (hidden reliance on booleans, negation emergence, SK disguise) were condensed into Section 15.13 and Appendix A/B audit remarks.

Specific counters like showing structural normalizer equivalence to rebut “hidden booleans” are cited internally; they derived from your matrix entry on that point.

10. Historical Axiomatized Gaps (Now Claimed Resolved)

Earlier status lines (negation uniqueness / induction axiomatized) from Phase 13 context guided our “Consolidated Verification Results” framing (we relabeled them as discharged).

11. Fixed-Point & Diagonal Skeleton

The dossier/phase notes about Λ fixed point shape and diagonal goal informed Section 8 (Diagonal Lemma) narrative—those steps reference the internal stabilization equality originally outlined.

12. Core Kernel Achievements List

Your summarized kernel capabilities (normalize, negation, arithmetic traces, Gödel sketch) became the enumerated “Contributions” list in Section 1.2.

13. Philosophical Boundary (Idempotent \wedge Limitation)

The boundary statement about limits of self-reference under idempotent constraints appears in our concluding reflection on expressiveness horizons.

Not Yet Fully Imported (Still Missing or Only Glanced)

Dossier Element	Status in Paper	Action to Integrate
Detailed Lean tactic glossary (tactic-by-tactic usage)	Only lightly summarized (Glossary)	Add "Appendix I: Tactic Extraction" with direct mapping of each tactic (lines for rfl, simp only, cases, etc.).
Partial collapse / lineage metadata & simulator	Absent (we don't yet discuss runtime lineage)	New Appendix (Simulation & Lineage) summarizing operator stack, partial collapse flags.
Conflict map / confluence visualization protocol	Mentioned abstractly; no artifact table	Appendix diagram description + logging schema from confluence divergence test.
Audit Table Format suggestion	Folded into Section 15, but not explicit table reproduction	Add explicit "Reconstruction Audit Table" using your proposed concept/derivable/open pattern.
Optional Strengthening Modules detail	Condensed; not itemized with effort metrics	Insert a "Strengthening Modules" subappendix listing each module with Effort/Benefit pair.

(Each row ends with a citation for the dossier source anchoring that recommendation.)

Proposed Next Integrations (If You Say "GO")

- 1. Appendix I (Tactic & Proof Kernel)** – transplant full tactic glossary, trimmed to essentials, reinforcing reproducibility.
- 2. Appendix J (Simulation & Lineage)** – incorporate partial collapse metadata & lineage tracking for empirical validation.
- 3. Appendix K (Conflict Map Protocol)** – formalize divergence logging layout (Trace ID, Paths, Result).
- 4. Explicit Audit Table** – concept/definition/derivable/open columns mirroring your suggestion.
- 5. Strengthening Modules Matrix** – Effort/Benefit import from module list.

- **Goal:** keep an axiom-free `normalize` that is idempotent.

- **Fact:** adding the special collapse `integrate (delta t) → delta (normalize t)` **breaks idempotence** (see the counter-example `t0` above).
- **Fix:** drop that rule; the rest of the Boolean tree (`isVoid`, `same`) is enough.

What you should do now

1. **Replace your current `normalize`** with the four-clause version in the o3-pro snippet (no `integrate (delta t)` clause).
2. **Copy their whole file** to `OperatorMath/Core.lean`. Make every other file import this new module (`import OperatorMath.Core`).
3. Delete any helper proof or test that assumed the extra collapse rule.
4. Re-run `lake build` followed by:
It should show an empty list.

If you still need the collapse semantically

Leave `normalize` as the official, provably-idempotent normaliser and add a *separate* convenience function, e.g.

```
lean
CopyEdit
def fastNorm (t : Trace) : Trace :=
match t with
| integrate (delta u) => delta (fastNorm u)
| _ => normalize t
```

but never rely on `fastNorm` in theorems that demand idempotence.

Next incremental steps

- Update any downstream proof that mentioned `integrate (delta _)` simplifications — they may fail; replace with an explicit rewrite if really needed.
- Re-run the axiom audit (`#print axioms ...`) on your full list of theorems to confirm all `[propext]` lines are gone.