Universal Binary Principle (UBP) Framework v3.2+

Euan Craig, New Zealand

September 3, 2025

Abstract

The Universal Binary Principle (UBP) Framework provides a deterministic, information-centric model that seeks to computationally simulate and analyze fundamental aspects of reality through the manipulation of binary state toggles within high-dimensional spaces. Built upon axiomatic principles, modular software architecture, and rigorous coherence metrics, the UBP offers a foundation for scientifically exploring both physical and non-classical domains with precision and extensibility [1].

1 Introduction

The Universal Binary Principle (UBP) postulates that all observable phenomena emerge from discrete binary state changes, termed *toggles*, operating within multidimensional manifolds. The goal of the framework is to realize a fully rigorous, extensible system enabling new forms of computational experimentation—calculating, discovering, and validating realities not attainable with conventional methods. By encoding information in nuanced 24-bit OffBits and integrating persistent, content-addressable storage with advanced error correction, UBP bridges data science, quantum modeling, and fundamental physics under a unified formalism [2].

2 Core Principles of the UBP Framework

1. **OffBit**: The atomic binary unit. Each OffBit contains 24 bits partitioned into *identity*, *dynamic state*, and *relational context*. Unlike conventional bits, OffBits capture potentiality and layered properties.

2. **6D Bitfield Spatial Mapping**: All OffBits reside on a dynamic 6D spatial manifold, supporting the representation and simulation of complex relationships beyond classical 3D mapping. Mapping parameters adapt to hardware profiles and experiments.

3. **HexDictionary Universal Storage**: Persistent, content-addressable repository indexed by SHA256; supports immutability and reproducibility of all computational states and experiment outputs. Data is compressed (`gzip`), with standardized metadata for rich querying.

4. **BitTab Encoding**: Specialized 24-bit encoding translates physical or informational properties (such as atomic number, valence, block) into binary strings for experiment and simulation.

5. **Multi-Realm Physics Integration**: UBP supports quantum, electromagnetic, gravitational, biological, cosmological, nuclear, and plasma realms. Each realm receives unique resonance parameters, error correction, and toggle behaviors.

3 Framework Modules Overview

The UBP is modular, with each submodule building toward full functionality:

- `ubp_config.py`, `system_constants.py`: Central nervous system housing all UBP, mathematical, and physical constants; supports dynamic configuration across hardware.

- `state.py`: Implements OffBit and MutableBitfield (6D arrays for binary states).

- `toggle_ops.py`: Toggle algebra (AND, XOR, resonance, entanglement, superposition, spin transition).

- `kernels.py`: Provides resonance kernel, coherence calculations, global coherence invariants.
- `energy.py`: The UBP energy equation:

$$E = M \times C \times (R \times S_{\text{opt}}) \times P_{\text{GCI}} \times O_{\text{observer}} \times c_{\infty} \times I_{\text{spin}} \times \sum(w_{ij} M_{ij})$$

- `metrics.py`: NRCI (Non-Random Coherence Index), Coherence Pressure, Fractal Dimension, Spatial Resonance Index.
- `global_coherence.py`: Computes global phase-locking using weighted frequency averages.
- `enhanced_nrci.py`: Advanced NRCI, Golay-Leech integration, temporal weighting.
- `observer_scaling.py`: Models observer intent and purpose tensor interactions.
- `carfe.py`: Implements Cycloid Adelic Recursive Expansive Field Equation (CARFE) for nonlinear dynamic system evolution.
- `dot_theory.py`: Encodes purpose tensor mathematics and intentionality.
- `spin_transition.py`: Quantum spin dynamics, Zitterbewegung modeling, quantum information quantification.
- `p_adic_correction.py`, `glr_base.py`, `level_7_global_golay.py`: Multi-realm error correction, BCH, Hamming, Golay codes, p-adic lifting, adelic corrections.
- `prime_resonance.py`: Prime-based coordinate systems tuned via Riemann zeta zeros.
- `tgic.py`: Triad graph constraints, Leech lattice, dodecahedral projections, enforcing geometric coherence (3/6/9 rules).
- `hardware_emulation.py`, `hardware_profiles.py`: Simulate different hardware profiles and architectures.
- `ubp_lisp.py`: S-expression based ontology, executing UBP primitives via a native language.
- `crv_database.py`, `enhanced_crv_selector.py`: Dynamic resonance value management, CRV optimization.
- `htr_engine.py`: Harmonic Toggle Resonance engine for physical and abstract resonance behaviors.
- `ubp_pattern_analysis.py`, `ubp_256_study_evolution.py`, `visualize_crv_patterns.py`: Pattern generation/analysis, storing and visualizing cymatic-like coherence states.
- `materials_research.py`: Predictive modeling of materials (e.g. tensile strength in alloys) based on resonance and coherence.

- `rgdl.py`: Resonance Geometry Definition Language for dynamic geometry generation and emergent 3D field export.
- `optimize_route.py`: TSP solver leveraging resonance and NRCI optimization.
- `detect_anomaly.py`: NRCI-based anomaly detection in real time signals.
- `runtime.py`: Virtual Machine managing high-level state, semantic execution, simulation orchestration.
- Utility modules (`cli.py`, `dsl.py`, etc.): automation, command-line, and persistent state management.

4 The UBP Self-Contained Formula

The central computational pipeline of UBP is:

$$U(x) = H^{-1}\left(R\left[C\left(\Phi_t\left(E_R(T_1(x))\right)\right)\right]\right)$$

where:

- $T_1(x)$: BitTab 24-bit encoding of input x; b_{1-8} identity, b_{9-16} dynamic state, b_{17-24} relational context.
- $E_R(x)$: Realm-specific error correction; e.g., BCH, Hamming, Golay, p-adic, and Fibonacci strategies selected per R.
- Φ_t: Evolution operator, $\Phi_t(b) = \exp(tL_{\text{CARFE}}) * b$, $L_{\text{CARFE}} = \lambda C + \mu A + \nu R$.
- C: Coherence maximization (NRCI), parameter tuning for λ, μ, ν to maximize $NRCI(\Phi, T)$.
- $R[f]$: Rune protocol, fixed point closure via self-evaluating UBP-Lisp expressions.
- H^{-1}: HexDictionary retrieval of outputs and augmentation with all NRCI-coherent historical states.

5 Design Philosophy

UBP emphasizes:

- **Scientific rigor**: All computations are based on mathematically exact models rather than approximations.
- **Completeness**: Each module is fully functional; no placeholder or mock algorithms.
- **Persistence**: Robust SHA256-indexed content-addressable storage enables transparency and reproducibility.

- **Modularity**: Logical separation of concerns among modules for independent development and validation.
- **Adaptability**: Dynamic optimization for hardware, experiment types, and realm switching.
- **Discovery**: Uncover novel relationships and structures via binary, resonant, and coherence-driven analysis.

6 Example: Materials Modeling with UBP

The UBP framework has been demonstrated on atomic-scale modeling of resonant steel. Using a BCC lattice simulation, the Harmonic Resonance Transfer engine calculated NRCI of 0.9219, and the Resonant Geometry Definition Language engine produced a unique material "fingerprint." The experiment highlighted the framework's ability to link elemental properties, atomic structure, classical mechanics, and resonance analytics within one workflow [2].

7 Conclusion

UBP v3.2+ represents a leap toward unified computation grounded in fundamental binary information, modular architecture, error correction, and resonance-driven modeling. Its fully implemented modules and scientifically rigorous design provide new tools for physical modeling, discovery, and experimental science.

References

[1] Universal Binary Principle: A Meta-Temporal Framework for a Computational Reality. Technical Whitepaper, Euan R A Craig, 2025.

[2] A Computational Framework for Atomic-Scale Material Modeling: A Case Study on Resonant Steel using UBP, Euan R A Craig, 2025.