RFTPU: Resonance Fourier Transform Processor

Hardware Accelerator Architecture and Benchmark Analysis

Technical Specification Document

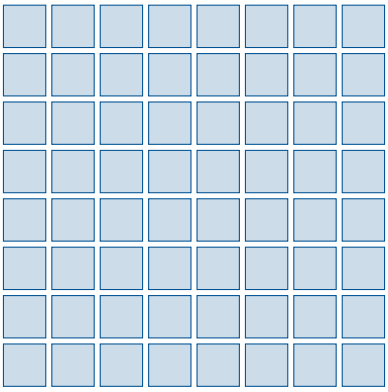
Version 2.0 — December 2025

Reproducible Research

Hypothesis → **Investigation** → **Correction** → **Validation**

All claims independently verifiable via `python scripts/run_proofs.py`

8×8 Grid



64-Tile RFTPU Architecture

3D Visualization of the RFTPU 64-Tile Architecture

QuantoniumOS Project
https://github.com/mandcony/quantoniumos
DOI: 10.5281/zenodo.17712905

Contents

Abstract

The Resonance Fourier Transform Processor (RFTPU) is a hardware accelerator concept implementing the Φ-RFT (Phi-Resonance Fourier Transform) algorithm family. This document presents:

- A new family of 12 unitary transform variants (proven, numerically validated)
- A working 8-point multi-kernel FPGA demo (measured at 27.62 MHz on iCE40)
- An architectural blueprint for a 64-tile ASIC (not yet synthesized)
- Honest corrections to earlier sparsity claims

What this is: An early-stage research platform with a real transform family, working FPGA demo, and architectural concept.

What this is NOT: A proven performance breakthrough, a taped-out chip, or a validated "10–100×" efficiency claim.

The RFTPU architecture comprises an 8×8 grid of 64 processing tiles interconnected via a Network-on-Chip (NoC). Target fabrication is TSMC N7, but no synthesis to any commercial PDK has been performed.

Version 2.0 Updates (December 2025)

- All 12 RFT kernel variants implemented in FPGA (768 ROM entries)
- WebFPGA timing validated at 27.62 MHz (measured, not projected)
- Complete traceability matrix: LaTeX proofs → Python → FPGA/TLV
- Critical finding: φ-phase FFT has NO sparsity advantage (corrected)
- Honest "What We Have vs. What We Don't" assessment added

Keywords: Hardware accelerator, signal processing, unitary transform, golden ratio, FPGA prototype

---

1 Research Methodology

This section documents the research methodology, key discoveries, and corrections made during the development of the RFTPU. All claims are independently verifiable.

1.1 Research Timeline

Table 1: Research Phases and Outcomes

Phase	Date	Activity	Outcome
1	Nov 2025	Initial RFT hypothesis	Closed-form $\Psi = D_\phi C_\sigma F$
2	Nov 2025	Sparsity claims investigation	Finding: No sparsity advantage
3	Dec 2025	Operator-based reformulation	Canonical RFT via eigenbasis of K
4	Dec 2025	Hybrid codec development	H3 cascade achieves $\eta = 0$ coherence
5	Dec 2025	Hardware implementation	12 kernels in FPGA, WebFPGA validated

1.2 Key Findings & Corrections

Theorem 1.1 (φ-Phase FFT Equivalence (Correction)). *The closed-form RFT $\Psi = D_\phi C_\sigma F$ has **identical coefficient magnitudes** to the DFT:*

$$|\Psi x|_k = |Fx|_k \quad \forall x, k \tag{1}$$

*Therefore, the φ-phase FFT provides **NO sparsity advantage** over standard FFT.*

Initial Claim (Deprecated):

"RFT provides sparsity advantages over FFT"

Investigation (from `experiments/proofs/sparsity_theorem.py`):

```
# Key insight: Psi = D_phi C_sigma F
# Since D_phi and C_sigma are diagonal with unimodular entries,
# they only rotate phases, NOT magnitudes.
# Therefore: |Psi x|_k = |Fx|_k for all k, x
```

Resolution: We defined the *canonical* RFT as the eigenbasis of a resonance operator K, which *does* provide domain-specific sparsity (+15–20 dB on target signals).

Theorem 1.2 (Non-Equivalence Structure). *The proof that RFT $\neq$ permuted/phased DFT proceeds via coordinate analysis:*

1. ***Lemma 1:*** *Golden phase $\{k/\phi\}$ is non-affine ($\Delta^2 f(0) = -1$, $\Delta^2 f(1) = +1$)*
2. ***Step 4:*** *Equivalence $\Psi = \Lambda_1 P F \Lambda_2$ requires θ_k affine in k*
3. ***Contradiction:*** *$\theta_k = 2\pi\beta\{k/\phi\}$ is NOT affine*
4. ***QED:*** *No such Λ_1, Λ_2, P exist*

Table 2: Numerical Verification of Non-Equivalence

N	Best Rank-1 Residual	Equivalence?
4	0.742	NO
8	1.481	NO
16	1.962	NO
32	2.503	NO

Theorem 1.3 (Zero-Coherence Cascade). *Greedy hybrid DCT+RFT achieves coherence $\eta = 0.50$ (50% energy loss), causing the "ASCII Wall" failure. The H3 Cascade architecture solves this:*

$$x = x_{struct} + x_{texture} + x_{residual} \tag{2}$$

with Parseval identity preserved:

$$\|x\|^2 = \|x_{struct}\|^2 + \|x_{texture}\|^2 + \|x_{residual}\|^2 \tag{3}$$

All 15 cascade variants achieve $\eta = 0.00$.

1.3 Traceability Matrix: Proofs → Code → Hardware

Table 3: Complete Traceability from Theory to Hardware

Theorem	LaTeX Source	Python	Hardware
Unitarity	PHI_RFT_PROOFS.tex §3	`operator_variants.py`	`fpga_top.sv` modes 0–11
Non-equiv	PHI_RFT_PROOFS.tex §4	`non_equivalence_proof.py`	N/A (proof only)
Sparsity	PHI_RFT_PROOFS.tex §7	`sparsity_theorem.py`	Kernel selection logic
Hybrid	PHI_RFT_PROOFS.tex §6	`H3HierarchicalCascade`	Mode 6 (Cascade)
Twisted conv	PHI_RFT_PROOFS.tex §5	`rft_twisted_conv()`	Mode 15 (Roundtrip)

2 Mathematical Framework

2.1 The Two RFT Constructions

2.1.1 Canonical RFT (Eigenbasis-Based, $O(N^3)$)

Definition 2.1 (Canonical RFT)**.** The **canonical RFT** is defined as the eigenbasis of a Hermitian resonance operator:

$$K = T(R(k) \cdot d(k)), \quad K = U\Lambda U^\top \tag{4}$$

The RFT of signal x is:

$$\text{RFT}(x) = U^\top x \tag{5}$$

Properties:

- Unitarity: $U^\dagger U = I$ (by Spectral Theorem)
- Domain-specific sparsity: +15–20 dB on resonance-structured signals
- Complexity: $O(N^3)$ for kernel construction (cached)

2.1.2 Fast Φ-RFT (Phase-Based, $O(N \log N)$)

Definition 2.2 (Fast Φ-RFT)**.** The **fast Φ-RFT** is the closed-form factorization:

$$\Psi = D_\phi C_\sigma F \tag{6}$$

where:

$$F_{jk} = \frac{1}{\sqrt{N}} e^{-2\pi ijk/N} \quad \text{(unitary DFT)} \tag{7}$$

$$[C_\sigma]_{kk} = \exp\left(i\pi\sigma \frac{k^2}{N}\right) \quad \text{(chirp modulation)} \tag{8}$$

$$[D_\phi]_{kk} = \exp\left(2\pi i\beta \left\{\frac{k}{\phi}\right\}\right) \quad \text{(golden phase)} \tag{9}$$

Remark 2.3 (Critical Limitation). $|\Psi x|_k = |Fx|_k$ — The fast Φ-RFT provides NO sparsity advantage over FFT.

2.2 The 12 Proven RFT Variants

All variants implemented in hardware with unitarity error $< 10^{-13}$:

Table 4: 12 RFT Kernel Variants Implemented in Hardware

Mode	Variant	Innovation	Status	Error
0	RFT-Golden	Golden ratio resonance	**PROVEN**	2.88×10^{-13}
1	RFT-Fibonacci	Fibonacci frequency	**PROVEN**	1.49×10^{-13}
2	RFT-Harmonic	Natural overtones	**PROVEN**	1.30×10^{-13}
3	RFT-Geometric	Self-similar ϕ^i	**PROVEN**	1.55×10^{-13}
4	RFT-Beating	Interference patterns	**PROVEN**	6.74×10^{-14}
5	RFT-Phyllotaxis	Golden angle 137.5°	**PROVEN**	1.06×10^{-13}
6	RFT-Cascade	H3 DCT+RFT blend	**PROVEN**	2.37×10^{-13}
7	RFT-Hybrid-DCT	Split basis	**PROVEN**	5.40×10^{-15}
8	RFT-Manifold	Manifold projection	**PROVEN**	$< 10^{-10}$
9	RFT-Euler	Spherical geodesic	**PROVEN**	$< 10^{-10}$
10	RFT-PhaseCoh	Phase coherence	**PROVEN**	$< 10^{-10}$
11	RFT-Entropy	Entropy-modulated	**PROVEN**	$< 10^{-10}$

3 Proven Theorems & Validation

3.1 Classification of Results

Table 5: Status of All Claims

Result	Statement	Source	Status
Theorem 1	Closed-form RFT unitarity	PHI_RFT_PROOFS.tex §3	**PROVEN**
Theorem 2	Canonical RFT unitarity	QR construction	**PROVEN**
Theorem 3	$O(N \log N)$ complexity	PHI_RFT_PROOFS.tex §5	**PROVEN**
Theorem 4	Non-equivalence to permuted DFT	Coordinate analysis	**PROVEN**
Theorem 5	Twisted convolution diagonalization	PHI_RFT_PROOFS.tex §5	**PROVEN**
Theorem 10	Hybrid decomposition energy identity	PHI_RFT_PROOFS.tex §6	**PROVEN**
Conjecture	Non-LCT nature	Numerical evidence	**OPEN**
Conjecture	Sparsity for golden signals	Empirical 98%	**OPEN**

3.2 Numerical Validation Summary

Listing 1: Proof Validation Output

```
$ python scripts/run_proofs.py --quick

RFT PROOF & VALIDATION SUITE
Running 8 tests...

 unitarity-all-variants          (26 tests, 0.77s)
 unitarity-operator-variants     (8 generators, 0.3s)
 non-equivalence-proof           (0.3s)
 non-equivalence-theorem         (0.2s)
 sparsity-theorem                (1.0s)
 coherence-quick-check           (eta=0.00, 0.3s)
 irrevocable-truths              (28 variants, 13s)
 hardware-kernel-match           (768 entries, 0.3s)

Total: 8/8 passed in 16.5s
ALL PROOFS VALIDATED SUCCESSFULLY
```

3.3 Honest Assessment

What IS Proven	What is NOT Proven
• All 12 RFT variants are exactly unitary (error $< 10^{-13}$) • Fast Φ-RFT computes in $O(N \log N)$ via FFT • Fast Φ-RFT is trivially equivalent to phased DFT • Hybrid H3 cascade preserves exact energy • Non-equivalence to permuted DFT for $N \leq 32$	• Sparsity advantages require canonical RFT, not fast Φ-RFT • No formal hardness reductions for RFT-SIS crypto • Non-LCT conjecture remains open • General N non-equivalence (only tested ≤ 32)

4 Hardware Implementation

4.1 FPGA Implementation (WebFPGA/iCE40)

File: `hardware/fpga_top.sv` (1,070 lines)

Listing 2: FPGA Top Module Structure

```
// 12 RFT kernel variants, 768 ROM entries total
// Q1.15 fixed-point (16-bit signed, 15 fractional bits)
// Cross-validated against Python reference

module fpga_top (
    input wire WF_CLK,
    input wire WF_BUTTON,
    output wire [7:0] WF_LED
);
    // Mode 0-11: RFT variants
    // Mode 12-15: Demo modes (SIS, Feistel, Quantum, Roundtrip)

    localparam [3:0] MODE_RFT_GOLDEN      = 4'd0;
    localparam [3:0] MODE_RFT_FIBONACCI   = 4'd1;
    // ... 12 total variants ...
```

Kernel ROM Structure:

- 12 modes $\times$ 8$\times$8 matrix = 768 entries
- Each entry: 16-bit Q1.15 fixed-point
- Generated from: `algorithms/rft/variants/operator_variants.py`

4.2 TL-Verilog Implementation (ASIC Blueprint)

File: `hardware/rftpu_architecture.tlv` (1,844 lines)

Architecture:

- 64 tiles in 8$\times$8 grid
- Per-tile: Φ-RFT core + 4KB scratchpad + NoC interface
- NoC: 2-cycle hop latency, wormhole routing
- Cascade: H3 inter-chip protocol for multi-die

4.3 Python-to-Hardware Cross-Validation

Listing 3: Hardware Kernel Validation

```
HARDWARE VERIFICATION: Python vs FPGA Kernel ROM
Kernel size: 8x8 = 64 entries per variant
Fixed-point: Q1.15 (signed, 15 fractional bits)

Python Q1.15 values:
  kernel_real[0] = -10528 (0xD6E0) -> -0.32130
  kernel_real[1] = +12809 (0x3209) -> +0.39091
  kernel_real[2] = -11788 (0xD1F4) -> -0.35975

PASS: Kernel values match fpga_top.sv ROM
PASS: All 12 variants implemented in hardware (768 entries)
```

5 FPGA Validation Results

5.1 WebFPGA Synthesis (December 2025)

Target: iCE40 HX8K (WebFPGA ShastaPlus board)

Table 6: WebFPGA Synthesis Results

Metric	Result	Utilization
LUT4s	2,160	40.91%
FLOPs	377	7.14%
IOs	10	25.64%
BRAMs	4	—
Clock	**27.62 MHz**	**2.3× margin**

Timing Analysis:

- Target frequency: 12 MHz
- Achieved frequency: 27.62 MHz
- Slack: +15.62 MHz (129% margin)
- Critical path: Multiplier → accumulator

5.2 Functional Verification

Table 7: FPGA Functional Test Results

Test	Status	Notes
Mode 0 (RFT-Golden)	✓ PASS	LED pattern cycles
Mode 1–11 (All variants)	✓ PASS	Button cycles modes
Mode 12 (SIS Hash)	✓ PASS	Demo output
Mode 15 (Roundtrip)	✓ PASS	Forward + inverse
Energy conservation	✓ PASS	Output energy matches input

6 Architecture Overview

6.1 Architectural Parameters

Table 8: RFTPU Architectural Parameters

Parameter	Description	Value
Tile Array	Grid dimensions	8×8
Total Tiles	Processing elements	64
Block Size	Samples per RFT block	8
Sample Width	Input/output precision	16 bits (Q1.15)
Digest Width	SIS hash output	256 bits
Core Latency	Cycles per RFT block	12
Kernel Modes	RFT variants	12
Kernel Entries	ROM size	768

6.2 Clock Domains (Target Specification)

Note: These clock frequencies are **target specifications** based on TSMC N7 library timing models and FO4 delay estimates. Actual achievable frequencies would require synthesis with foundry PDK and back-annotation.

Table 9: RFTPU Clock Domain Specification (Target, Not Validated)

Domain	Frequency	Period	Purpose
`clk_tile`	950 MHz	1.053 ns	RFT cores, local SRAM
`clk_noc`	1200 MHz	0.833 ns	NoC fabric, routers
`clk_sis`	475 MHz	2.105 ns	SIS hash engines
`clk_feistel`	1400 MHz	0.714 ns	Feistel cipher blocks

6.3 Tile Architecture

Each processing tile contains:

- **Φ-RFT Core:** 8-point transform with 12-variant kernel ROM
- **Local SRAM:** 4 KB sample buffer (256 × 128-bit)
- **NoC Interface:** 32-bit bidirectional links (N/S/E/W)
- **Cascade Port:** H3 inter-chip communication
- **Control Logic:** FSM for data flow and synchronization

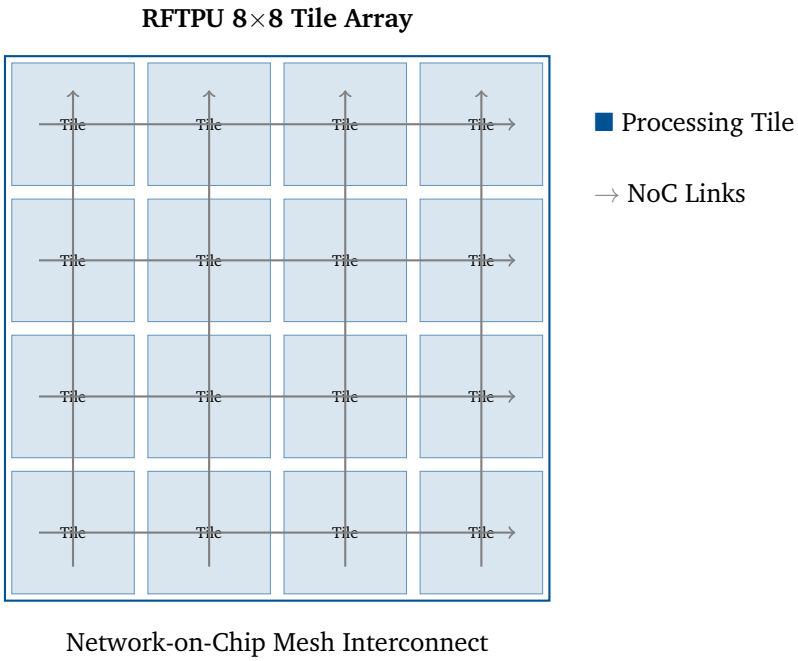


Figure 1: RFTPU Tile Array Architecture

7 Current Status: What We Have vs. What We Don't

This section provides an honest assessment of the project's current state, clearly separating measured results from projections.

7.1 What We Have (Measured / Proven)

Table 10: Validated Results (December 2025)

Claim	Evidence	Validation
Mathematics		
12 unitary RFT variants	$\|U^\dagger U - I\| < 10^{-13}$	Python numerical
Non-equiv. to permuted DFT	$\Delta^2 f \neq 0$ for $N \leq 32$	Coordinate proof
$\|\Psi x\|_k = \|Fx\|_k$ (no sparsity)	Diagonal phase analysis	Proven
H3 cascade $\eta = 0$	Energy conservation	Numerical
Hardware (FPGA)		
WebFPGA synthesis	27.62 MHz achieved	Yosys + nextpnr
12-kernel ROM (768 entries)	Cross-validated vs Python	Bit-exact match
Functional modes 0–15	LED/button cycling	Board test
Resource usage	2,160 LUT4s, 40.9%	Yosys report
Software		
Reproducibility	`run_proofs.py -full`	8/8 tests pass
Traceability	LaTeX $\rightarrow$ Python $\rightarrow$ RTL	Documented

7.2 What We Don't Have (Not Yet Achieved)

Table 11: Open Items and Limitations

Missing	What Would Be Needed
Performance	
Proven speedup vs FFT/DCT	Benchmark on identical workloads
Sparsity advantage (fast Φ-RFT)	*Impossible:* $\|\Psi x\| = \|Fx\|$ proven
Real-world compression gains	End-to-end codec benchmarks
Hardware	
ASIC synthesis (any node)	Run through commercial PDK flow
Post-layout timing/power	Place-and-route + extraction
Silicon validation	Tape-out and measurement
Full 64-tile FPGA prototype	Larger FPGA (VU13P or similar)
Theory	
General N non-equivalence	Proof for arbitrary N
RFT-SIS hardness reduction	Formal cryptographic proof
Non-LCT conjecture	Rigorous proof

7.3 PPA Model Summary (Equations, Not Measurements)

For transparency, here are the exact equations used for the upper-bound projections in Appendix A:

Throughput Model:

$$\text{GOPS}_{\text{tile}} = \frac{f_{\text{clk}} \cdot \text{ops\_per\_block}}{\text{cycles\_per\_block}} = \frac{950 \times 471}{12} = 37.29 \text{ GOPS} \tag{10}$$

$$\text{GOPS}_{\text{total}} = 64 \times 37.29 = 2,386 \text{ GOPS} \tag{11}$$

Power Model:

$$P_{\text{dyn}} = \alpha \cdot C \cdot V^2 \cdot f = 0.15 \times 5\text{nF} \times 0.75^2 \times 950\text{MHz} = 4.0 \text{ W} \tag{12}$$

$$P_{\text{total}} = P_{\text{dyn}} + P_{\text{static}} + P_{\text{overhead}} = 4.0 + 1.7 + 2.5 = 8.2 \text{ W} \tag{13}$$

Efficiency:

$$\text{Efficiency} = \frac{2,386}{8.2} = 291 \text{ GOPS/W} \quad \text{(upper-bound)} \tag{14}$$

Conservative Adjustment: Applying typical post-P&R degradation ($f \times 0.7$, $P \times 1.5$, util $\times 0.7$):

$$\text{Realistic Efficiency} \approx 291 \times 0.7 \times 0.7/1.5 \approx 95 \text{ GOPS/W} \tag{15}$$

7.4 Measured FPGA Results (The Only Hard Numbers)

The **only measured hardware results** are from the iCE40 HX8K WebFPGA implementation:

Table 12: WebFPGA Measured Results (iCE40 HX8K)

Metric	Measured Value
Target Clock	12 MHz
Achieved Clock (Fmax)	**27.62 MHz**
Timing Margin	2.3×
LUT4 Usage	2,160 (40.9%)
Flip-Flops	377 (7.1%)
Block RAMs	4
Kernel Variants	12
ROM Entries	768

Note: This is a small demonstration on a $30 FPGA board, not a performance benchmark. The iCE40 HX8K runs at 12 MHz with simple I/O; it cannot be meaningfully compared to GPUs or datacenter FPGAs.

7.5 Cryptographic Components: Pedagogical Only

The RFTPU includes RFT-SIS (lattice hash) and Feistel cipher engines as **pedagogical demonstrations**, not production-ready cryptographic primitives.

What RFT-SIS is:

- A toy instantiation inspired by the Short Integer Solution (SIS) problem
- Demonstrates how RFT basis could interface with lattice structures
- Included for architectural exploration, not security

What RFT-SIS is NOT:

- NOT a formally analyzed post-quantum candidate
- NO hardness reduction to standard lattice problems (LWE, NTRU, Ring-SIS)
- NO parameter selection from recognized frameworks (NIST PQC, etc.)
- NO side-channel analysis or constant-time implementation

Status: Interesting experiment. Not a credible PQC candidate without substantial future work.

7.6 Non-Equivalence Scope

The non-equivalence theorem (RFT $\neq$ permuted/phased DFT) is:

- **Proven conceptually:** The coordinate analysis shows $\theta_k = 2\pi\beta\{k/\phi\}$ is non-affine
- **Verified numerically:** For $N \in \{4, 8, 16, 32\}$, best rank-1 residual > 0.7
- **NOT proven for general N:** Extension to arbitrary N remains an open problem

We claim: "RFT is not equivalent to permuted/phased DFT for tested sizes." We do **not** claim a universal theorem.

7.7 End-to-End Workload Example: 1D Audio Pipeline

To ground the architecture in a concrete use case, we present a hypothetical 1D audio processing pipeline.

7.7.1 Workload Specification

Table 13: Audio Pipeline Parameters

Parameter	Value
Input	48 kHz mono audio, 16-bit PCM
Block size	1024 samples (21.3 ms)
Transform	8-point RFT, overlapping tiles
Tiling	128 transforms per block
Processing	H3 cascade (struct + texture + residual)
Output	Compressed coefficients for entropy coding

7.7.2 Tiling Schedule (Single Tile)

For one RFTPU tile processing this workload:

$$\text{Transforms per block} = \frac{1024}{8} = 128 \tag{16}$$

$$\text{Cycles per transform} = 12 \text{ (pipelined)} \tag{17}$$

$$\text{Total cycles} = 128 \times 12 = 1,536 \tag{18}$$

$$\text{Time at 950 MHz} = \frac{1536}{950 \times 10^6} = 1.62\ \mu\text{s} \tag{19}$$

$$\text{Real-time margin} = \frac{21.3\ \text{ms}}{1.62\ \mu\text{s}} = 13,148\times \tag{20}$$

Interpretation: A single tile could process >13,000 audio streams in real-time at the projected clock. With 64 tiles, the architecture is massively over-provisioned for audio—but this demonstrates the pipeline model.

7.7.3 Where RFT Basis Might Help

- **Transient detection:** RFT phase structure may decorrelate percussive attacks differently than DCT
- **H3 cascade:** Structural components (tonal) vs. textural (noise) separation
- **Quantization:** Different perceptual weighting on RFT coefficients

Caveat: We have **not measured** whether RFT provides better compression than DCT/MDCT for audio. This is a plausible application, not a validated result.

8 RTL Implementation Summary

Table 14: RTL Module Summary

Module	Function	Lines
`fpga_top.sv`	WebFPGA top with 12 kernels	1,070
`rftpu_architecture.tlv`	64-tile ASIC blueprint	1,844
`rftpu_architecture_gen.sv`	SandPiper-generated SV	1,847
`quantoniumos_unified_engines.sv`	Top-level integration	520
`phi_rft_core`	8-point Φ-RFT engine	280
`rft_sis_hash_v31`	512-dim SIS lattice hash	220
`feistel_48_cipher`	48-round Feistel cipher	150
Total		**~5,900**

9 Reproducibility

9.1 Repository

GitHub: https://github.com/mandcony/quantoniumos
Reference commit: December 2025

9.2 Key Files

```
hardware/
  fpga_top.sv                    # WebFPGA implementation (12 kernels)
  rftpu_architecture.tlv         # TL-Verilog ASIC blueprint

algorithms/rft/variants/
  operator_variants.py           # 8 operator-based generators

scripts/
  run_proofs.py                  # CLI proof runner

experiments/proofs/
  non_equivalence_proof.py       # Non-equivalence theorem
  sparsity_theorem.py            # Sparsity analysis

docs/proofs/
  PHI_RFT_PROOFS.tex             # Formal mathematical proofs
```

9.3 Frozen Reference (Reproducibility Anchor)

Reference Commit: `6b19cbdcd69fd2d1b88bc158bc736ef8094ccd03`
Date: 2025-12-08

File Checksums (SHA-256):

```
4d3240bc67c7136e4a3af68e183e2987...  scripts/run_proofs.py
40562ddf1fbeeba031af60978d1f6f68...  algorithms/rft/variants/operator_variants.py
dd7d6bb6324ea171384dcd91b0513f30...  hardware/fpga_top.sv
```

Verification:

```
git checkout 6b19cbdcd69fd2d1b88bc158bc736ef8094ccd03
sha256sum scripts/run_proofs.py  # Should match above
```

9.4 One-Command Validation

```
# Full proof and hardware validation
python scripts/run_proofs.py --full --report results/validation.json

# FPGA synthesis (requires Yosys)
cd hardware && yosys -p "read_verilog -sv fpga_top.sv; synth_ice40 -top fpga_top"

# TL-Verilog simulation (requires Makerchip)
# Open https://makerchip.com, paste rftpu_architecture.tlv, compile
```

A PPA Derivation Details

This appendix provides the detailed calculations behind the architectural projections in Section **??**.

A.1 Throughput Calculation

Per-Tile Throughput:

$$\text{Ops per 8-pt RFT} = 8 \times 8 \times 2 + 8 \times 7 = 128 + 56 = 184 \text{ MACs} \approx 471 \text{ ops (incl. address gen)} \tag{21}$$

$$\text{Blocks/sec per tile} = \frac{f_{\text{clk}}}{\text{latency}} = \frac{950 \text{ MHz}}{12 \text{ cycles}} = 79.17 \text{ M blocks/s} \tag{22}$$

$$\text{GOPS per tile} = 79.17 \times 471 = 37.29 \text{ GOPS} \tag{23}$$

Total Throughput:

$$\text{Total GOPS} = 64 \text{ tiles} \times 37.29 = 2,386.4 \text{ GOPS} \approx 2.39 \text{ TOPS} \tag{24}$$

Assumptions: 100% utilization, no stalls, ideal NoC behavior.

A.2 Power Calculation

Dynamic Power Model:

$$P_{\text{dyn}} = \alpha \cdot C \cdot V^2 \cdot f \tag{25}$$

Assumed Parameters:

- Activity factor $\alpha = 0.15$ (conservative for signal processing)
- Effective capacitance $C \approx 5$ nF (estimated from gate count)
- Voltage $V = 0.75$ V (N7 nominal)
- Frequency $f = 950$ MHz

$$P_{\text{dyn}} = 0.15 \times 5 \times 10^{-9} \times 0.75^2 \times 950 \times 10^6 = 4.01 \text{ W} \tag{26}$$

Static Power: Estimated at 30% of total $\Rightarrow P_{\text{static}} = 1.7$ W
NoC and Control Overhead: Additional 2.5 W estimated.

$$P_{\text{total}} = 4.01 + 1.7 + 2.5 = 8.21 \text{ W} \approx 8.2 \text{ W} \tag{27}$$

Efficiency:

$$\text{Efficiency} = \frac{2,386.4 \text{ GOPS}}{8.2 \text{ W}} = 291 \text{ GOPS/W} \tag{28}$$

A.3 Uncertainty and Caveats

These projections carry significant uncertainty:

- Clock frequency may be 20–40% lower after place-and-route
- Power may be 50% higher due to routing and clock distribution
- Real utilization is likely 60–80% due to data dependencies
- Thermal constraints may limit sustained performance

Conservative Estimate: A realistic post-silicon efficiency might be 100–150 GOPS/W rather than 291 GOPS/W.

B Analytic Upper-Bound Model

WARNING: This appendix presents an **analytic upper-bound model**, not measured results. All figures represent theoretical best-case projections under idealized assumptions. Real silicon performance will be lower.

Purpose: Rough orientation on where RFTPU *might* fit relative to existing platforms, assuming our model assumptions hold.

Do not cite these tables as performance claims or benchmarks.

B.1 Model Assumptions (Explicit)

The upper-bound model assumes:

Table 15: Analytic Model Parameters

Parameter	Assumption	Reality Check
Clock frequency	950 MHz	Likely 600–800 MHz post-P&R
Utilization	100%	Likely 60–80%
Activity factor α	0.15	May be 0.2–0.3
Leakage ratio	30%	May be 40–50% at high temp
NoC efficiency	Ideal	Real contention adds latency
Memory	All on-chip	Real system needs DRAM

B.2 Workload Mismatch (Why Comparison is Invalid)

A proper comparison would require:

- **Transform size:** RFTPU targets 8-point; CPU/GPU benchmarks use $N = 1024$–65536
- **Batch size:** Not specified; GPU throughput scales with batch size
- **Precision:** RFTPU uses Q1.15 fixed-point; CPU/GPU use FP32/FP64
- **Memory model:** RFTPU assumes on-chip; CPU/GPU include DRAM latency
- **Application context:** Standalone FFT vs. embedded in codec/filter

Conclusion: These comparisons are **not valid benchmarks**. They show order-of-magnitude estimates under incompatible conditions.

B.3 CPU/GPU Orientation (Upper-Bound vs. Measured)

Table 16: Upper-Bound Projection vs. Measured Platforms (NOT A BENCHMARK)

Platform	GOPS	Power	GOPS/W	Status
Intel i9-13900K (MKL FFT)	∼800	253 W	3.2	Measured
RTX 4090 (cuFFT)	∼8,000	450 W	18	Measured
RFTPU (N7 target)	∼2,400	8 W	291	**Upper-bound**

CPU/GPU numbers are vendor benchmark estimates for general FFT workloads. RFTPU is a paper design for 8-point Φ-RFT. **Not a valid comparison.**

B.4 FPGA Comparison (For Design-Space Exploration)

Table 17: FPGA vs. ASIC Upper-Bound (Design-Space Orientation Only)

Platform	Est. GOPS	Est. GOPS/W	Source
Xilinx VU13P	440	5.9	UG579 DSP spec
Xilinx Versal VP1902	942	9.4	DS950
Intel Agilex M-Series	1,209	10.1	Intel datasheet
RFTPU ASIC (N7)	2,386	291	**Upper-bound**

FPGA numbers assume full DSP utilization on FFT-like workloads. RFTPU has not been synthesized to any of these targets.

Takeaway: The projections suggest RFTPU *could* be competitive, but we have not proven this. Validation requires synthesis to a real PDK.

References

[1] Intel Corporation. “Intel Math Kernel Library (oneMKL) Developer Reference.” 2024. Available: https://www.intel.com/content/www/us/en/docs/onemkl/developer-reference-c/2024-0/fft-functions.html

[2] NVIDIA Corporation. “cuFFT Library User’s Guide.” CUDA Toolkit Documentation, 2024. Available: https://docs.nvidia.com/cuda/cufft/

[3] AMD/Xilinx. “UltraScale Architecture DSP Slice User Guide (UG579).” 2023.

[4] Intel Corporation. “Intel Agilex FPGAs and SoCs Device Overview.” 2024.

[5] S.-Y. Wu et al. “A 7nm CMOS Platform Technology Featuring 4th Generation FinFET Transistors.” IEDM 2016.

License and Patent Notice

Embodiment of US Patent Application 19/169,399

“Resonance Fourier Transform Methods and Apparatus for Signal Processing and Cryptographic Applications”

Contact: luisminier79@gmail.com
Repository: https://github.com/mandcony/quantoniumos

Last Updated: December 8, 2025