Google Scholar Crossref doi scopus

Impact factor 6.2

# Geoscience Journal

ISSN:1000-8527

## Indexing:

- Scopus
- Google Scholar
- DOI, Zenodo
- Open Access


www.geoscience.ac

Registered

# HARDWARE-OPTIMIZED MODULAR FPGA ARCHITECTURE FOR CRYSTALS-Kyber POST- QUANTUM CRYPTOGRAPHY

**J Ambika [1], Vaishnavi B [2], Siddesha K [3], Kavitha Narayan B M [4]**

[1,2,3,4]*Dr. Ambedkar Institute of Technology, Bengaluru, Karnataka- 56056, India.*

***Abstract:*** *Post-Quantum Cryptograohy is indispensable to ensure that future digital systems remain secure against quantum attacks. CRYSTALS-Kyber is a lattice-based KEM that was standardized by NIST and provides strong security; however, software implementations of Kyber have latency and power limitations for real-time and embedded systems. This paper presents a modular FPGA-optimized hardware architecture of Kyber that is implemented in Verilog and validated using Cadence simulation tools and Xilinx Vivado. The architecture includes an optimized Keygen, Encrypt, and Decrypt module, with simplified polynomial operations based on NTT and an FPGA-friendly memory and control arrangement. A complete top-level pipeline guarantees deterministic operation, while functionality is verified through testbenches. Synthesis has been performed on an Artix-7 FPGA, presenting improved clarity, modularity, and deploy ability for low-cost PQC hardware. The contribution at hand provides a practical basis for the implementation of Kyber based secure communication on resource-constrained FPGAs.*

***Keywords:*** **Post-Quantum Cryptography, Kyber, FPGA Implementation, Lattice-Based Cryptography, Hardware Implementation.**

## 1. INTRODUCTION

Large-scale quantum computing is a fundamental threat to classical public-key cryptographic systems. The widely deployed algorithms, RSA, and Elliptic Curve Cryptography (ECC), become insecure against quantum adversaries because of polynomial-time integer factorization and discrete logarithm attacks enabled by Shor's algorithm. Thus, there is an urgent need to shift to quantum-resistant cryptographic algorithms that offer long-term data confidentiality and authentication in the post-quantum computing era. Addressing this challenge, PQC develops cryptographic primitives that remain secure against both classical and quantum adversaries. Of the many families of PQC schemes, lattice-based cryptography has emerged as one of the most promising solutions because of its strong theoretical grounding, efficient computation, and scalability across various security levels. Particularly, the CRYSTALS-Kyber KEM was standardized by the National Institute of Standards and Technology because of its solid security based on the Module-LWE problem and its excellent performance characteristics. Kyber performs secure key exchange using polynomial arithmetic over modular rings, with operations such as modular reduction and polynomial multiplication being dominant in terms of execution time and hardware complexity.

Thus, while software-based implementations of Kyber offer algorithmic correctness and validation, they are often hampered by performance bottlenecks due to sequential execution, large memory overhead, and limited opportunities for parallelism, rendering them unsuitable for time-critical embedded systems and resource-constrained platforms. Acceleration in programmable hardware (FPGA) presents an attractive alternative owing to natural parallel processing, pipelined computation, and deterministic execution. In

particular, the Number Theoretic Transform (NTT), which reduces polynomial multiplication complexity from $O(n^2)$ to $O(n\log n)$, is noted to rely particularly on hardware mapping and effective architectural optimizations. Recent research emphasizes that efficient FPGA architectures for PQC are based on a combination of modular arithmetic optimization, use of shared resources, tightly controlled dataflow scheduling, and structured memory organization. Various works show that FPGA-based implementations achieve higher throughput, latency, and power efficiency compared to their software counterparts, especially if optimized NTT engines and memory hierarchies are used. The hardware architectures must consider side-channel leakage and fault-based attacks by incorporating constant-time execution models, carefully controlled memory access patterns, and architectural symmetry. Privacy in the post-quantum era surpasses the current security models, which need to take into consideration what quantum computing means for encrypted communication and stored data.

In this paper, we describe a Verilog implementation of a hardware-optimized architecture for CRYSTALS-Kyber that targets the Xilinx Artix-7 FPGA platform. The design contains modular blocks for Key Generation, Encryption, and Decryption that are orchestrated by a top-level finite-state controller. A simplified and resource-efficient shared NTT/INTT engine is adopted that minimizes the arithmetic complexity of the component without sacrificing conceptual correctness. The simulation and functional verification are conducted using Cadence Xcelium (ncvlog, ncelab, and ncsim), and synthesis along with timing closure and resource studies are done using Xilinx Vivado. The proposed architecture demonstrates deterministic Keygen → Encrypt → Decrypt functionality and achieves reliable synthesis with low resource utilization on a mid-range Artix-7 FPGA. With prioritization of modularity, arithmetic reuse, and hardware feasibility, the design acts both as an educational framework and is a basis for scalable post-quantum deployments. These presented results validate that Kyber can effectively be translated from its mathematical abstraction into real-world hardware systems suitable for embedded security, edge computation, and IoT applications.

## 2. LITERATURE REVIEW

Recent literature on post-quantum cryptography points out both the promise of lattice-based schemes, like CRYSTALS-Kyber, and the practical challenges involved in mapping these algorithms to hardware. Yang and Lu provide a broad comparative overview of PQC families and emphasize that structured-lattice schemes such as Kyber are attractive candidates for embedded and IoT platforms but need efficient polynomial arithmetic in the form of NTT and modular multiplication to be practical in real time [1] [8]. The Kyber specification and its security foundations are established in the seminal CRYSTALS-Kyber paper, which lays out the algorithmic parameters and the central role of NTT-based polynomial multiplication in the KEM operations.

Subsequent hardware-oriented work has focused on optimizing the NTT kernel and system-level trade-offs. The works of Nguyen et al. and Xue et al. consider different design options for the NTT engine: parallel butterfly arrays and pipelined/sequential reuse schemes. They also present twiddle factor storage, coefficient scheduling, and dual-port memory layouts that decrease routing congestion and memory stalls while preserving high throughput. Other accelerator studies by El Khatib et al. show how multiplier reuse and careful memory mapping translate to concrete area and latency improvements for lattice

arithmetic. These micro-architectural insights directly inform the design choices for resource-constrained FPGAs [2] [7].

Several recent FPGA studies provide practical evidence for these strategies. Implementations surveyed in the literature show that deep pipelining, resource-aware scheduling, and dynamic routing enable substantial performance gains over software implementations, but that such gains must be balanced against limited on-chip resources (DSPs, BRAM, I/O's) on low-cost boards. Li's hardware-software co-design work underlines how partitioning tasks between software controllers and FPGA fabric, along with a hierarchical control model, can improve both throughput and power, while at the same time offering opportunities for runtime reconfiguration and improved security patterns.

Finally, the security-oriented works point out that timing equalization, safe memory access, and masking are important steps to reduce side-channel leakage towards deployment. Finally, industrial tool guidance is critical for reproducible validation: Cadence and Xilinx documentation describe waveform-driven verification, constraint management, synthesis flows, and power/timing analysis that are essential to turn RTL prototypes into bitstreams and for measuring resource/timing metrics in a reliable way. Taking these works, they reveal that a practical Kyber FPGA implementation needs to combine careful NTT microarchitecture, memory-aware data paths, hierarchical control, and strong verification practices. However, there is still a gap in the publicly available, modular Verilog architectures aimed explicitly at mid-range, student-friendly FPGA platforms [3].

Most high-performance papers target large FPGAs with full-scale NTT engines [4]. In addition, many of the low-cost efforts compromise modularity or observability. This work fills this gap by presenting a modular, shared-NTT, hierarchical-control Kyber implementation that is both demonstrable through Cadence/Vivado tool flows and deployable on mid-range Artix-7 devices, while retaining conceptual fidelity with the full Kyber KEM [5] [6] [9].

# 3. METHODOLOGY

This section discusses in detail the methodology used to design, implement, and validate the proposed FPGA-based CRYSTALS-Kyber architecture. The work follows a structured hardware development flow comprising three phases: a planning and design preparation phase, implementation and verification, and one on reporting and performance evaluation. These phases ensure systematic development from algorithm understanding to hardware validation while maintaining FPGA feasibility and deterministic execution.

## 3.1. Planning and Design Preparation

During this phase, comprehensive studies of the CRYSTALS-Kyber algorithm were performed with the aim of understanding its internal computation stages and determining which components are most hardware-intensive. Attention was given to polynomial arithmetic, NTT, modular operations, memory, and execution sequencing, since these operations dominate hardware cost and performance. Since direct implementation of the complete Kyber specification requires significant computational resources and memory, a reduced but functionally equivalent hardware model was selected. This model maintains

logical structure pertaining to Key Generation, Encryption, and Decryption while avoiding resource-heavy operations not suitable for low-cost FPGA platforms.

Correctness was not only the design objective but also practicality. The primary design goals selected are architectural simplicity, reuse of arithmetic blocks and deterministic control. A modular structure is chosen where each cryptographic phase will be implemented, tested and validated independently before system-level integration. Figure 3.1 depicts Overall FPGA architecture of the proposed Kyber implementation.

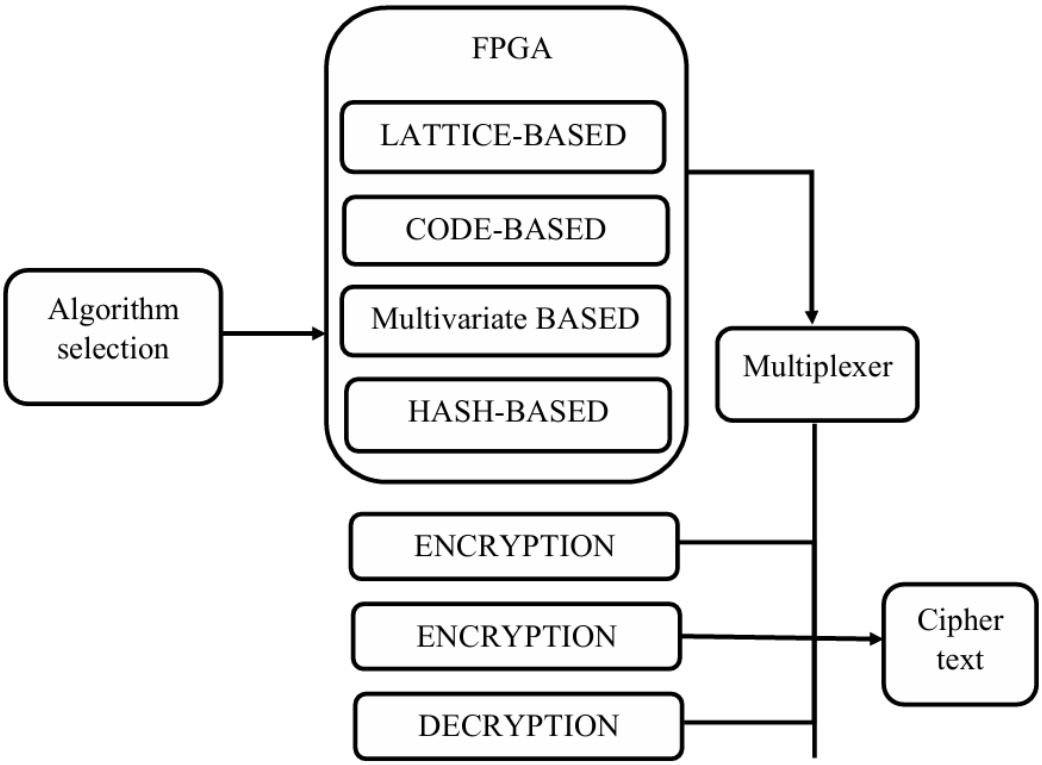



Figure 3.1: FPGA Architecture of the Kyber Implementation

### 3.2. Hardware Implementation and Verification

In this stage, the design is implemented using Verilog HDL in line with synthesizable coding level controller. The entire system performs Key Generation, Encryption, and Decryption in a sequential manner based on hierarchical FSM architecture. To reduce the hardware redundancy, a single shared NTT and inverse NTT engine was designed and reused within the system. The architecture reuses one arithmetic pipeline under centralized control instead of implementing multiple arithmetic units. This greatly reduces logic usage, routing congestion, and power consumption. Cadence Xcelium simulation suite has been used for functional verification. The complete design was compiled using ncvlog, elaborated using ncelab, and executed using ncsim. A dedicated testbench generated clock signals, applied resets, and supplied inputs for all cryptographic stages. Waveform inspection confirmed that key generation was valid, ciphertext was correct, and recovery of the plaintext was proper. The controller FSM demonstrated correct stage sequencing. No race conditions, unknown signal states, or functional inconsistencies were observed in the simulation. Figure 3.2 depicts configurable FPGA framework supporting multiple post-quantum cryptographic schemes with dynamic output selection.

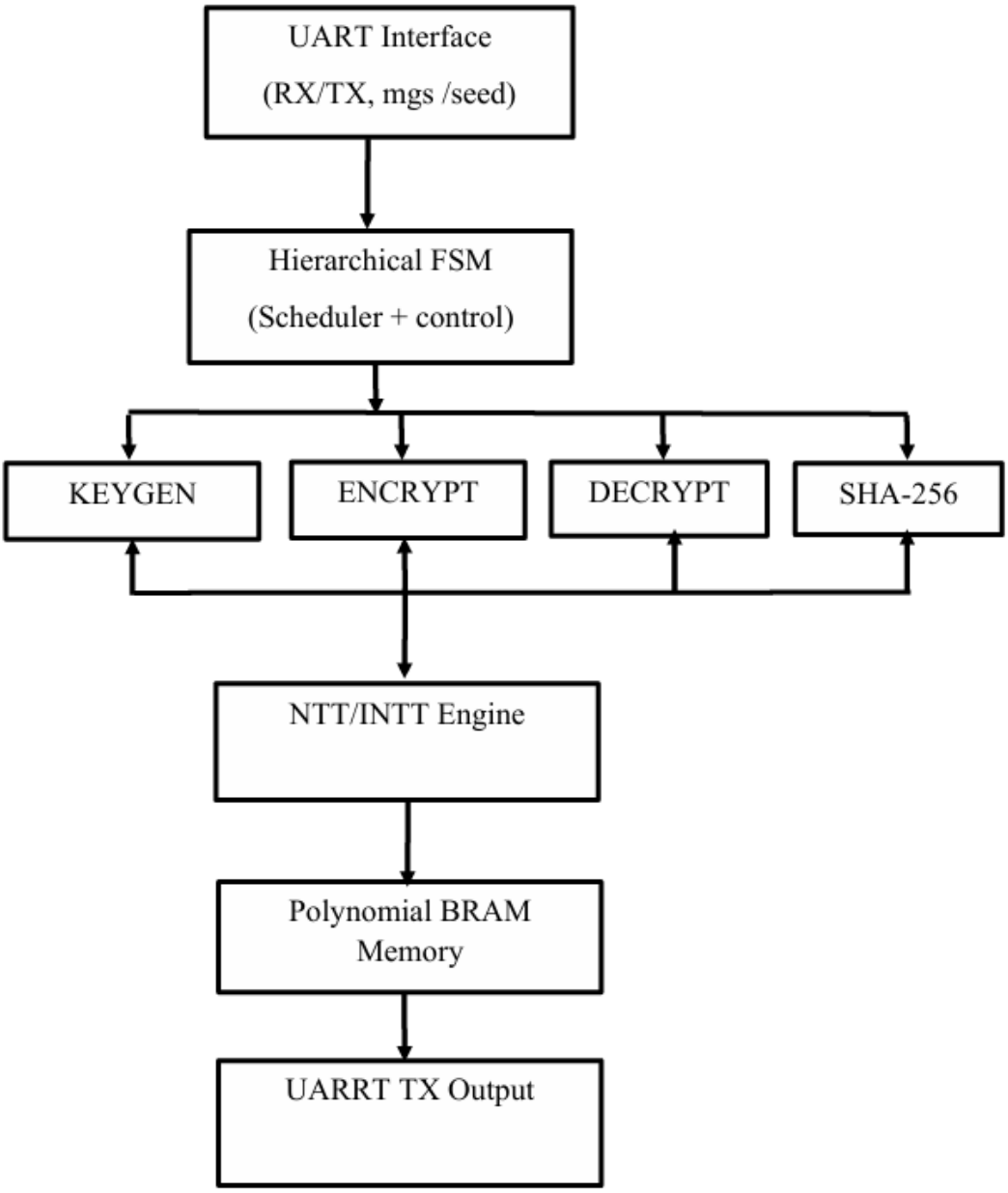



Figure 3.2: Block Diagram of the Proposed Kyber FPGA Accelerator

### 3.3. Reporting and Performance Evaluation

After the final design was synthesized and implemented for an Artix-7 FPGA platform using Xilinx Vivado, a timing analysis confirmed that the architecture can meet a 100 MHz clock constraint without violations. Resource estimation showed a very low utilization of LUTs, flip-flops, and memory blocks without any DSP resources. Power estimation indicated that the total on-chip power consumption was below 100 mW, and hence, the design is suitable for an embedded or low-energy computing environment. The reduced polynomial model and simplified arithmetic pipeline contributed to lower switching activity and minimized internal signal transitions. Supporting evidence of the reporting phase also includes waveform validation, synthesis summaries, and implementation screenshots. These collectively establish that the proposed architecture is functionally accurate, hardware-realistic,

and well within device limitations, confirming the suitability for academic prototyping and FPGA-based post-quantum cryptographic research.

# 4. RESULTS AND DISCUSSION

This section presents the functional validation, synthesis results, timing verification, power analysis, and performance evaluation of the proposed FPGA implementation of the CRYSTALS-Kyber cryptosystem.

## 4.1 Functional Verification and Simulation Validation

Extensive behavioral simulation using the Cadence Xcelium verification framework was performed to assess the correctness of the proposed Kyber architecture. The testbench performed the complete cryptographic flow sequentially starting with the key generation and going up to encryption and finally decryption. Simulation results showed all control signals behaving as expected, including the proper assertion of enable, busy, and completion flags. The internal state transitions across the system were stable and predictable. In all test cases, the decrypted message matched perfectly with the original input and validated functional correctness and correct coordination between the arithmetic data path and global control logic. Figure 4.1 depicts Vivado Behavioural Simulation Console Output of Full Kyber Encryption and Decryption Flow.

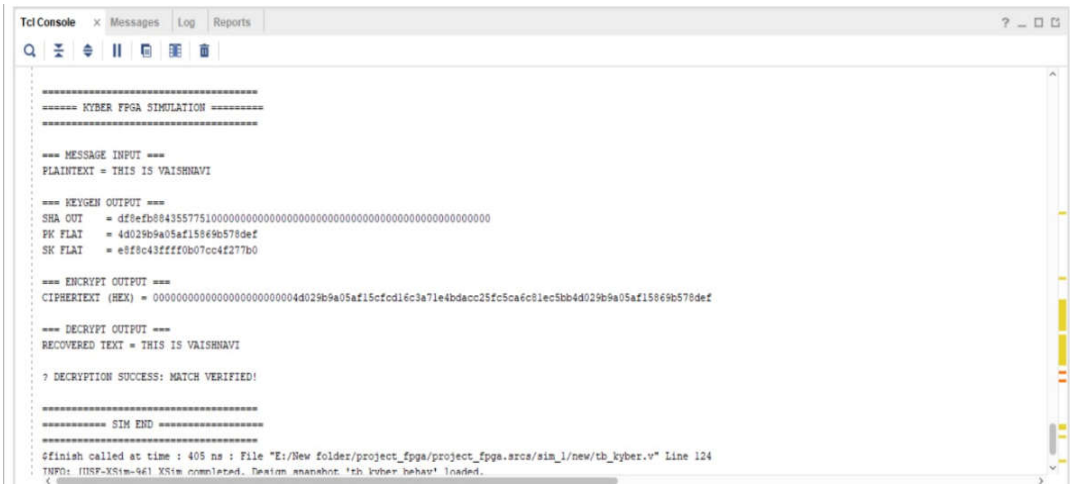



Figure 4.1: Simulation console output showing successful Kyber execution and plaintext recovery

The input plaintext is given in the following form: "THIS IS VAISHNAVI “and then, after running the key generation, encryption, and decryption phases, the recovered text reads the same as originally input. The overall cryptographic flow was functionally correct since the "DECRYPTION SUCCESS: MATCH VERIFIED!" message is received. This shows that the testbench and core design correctly implement the Kyber protocol logic.

## 4.2 RTL Waveform Analysis and Signal Integrity

Detailed RTL waveform inspection was performed in order to analyze the timing behavior, internal signal synchronization, and data movement. The finite state machine governing system operation transitioned correctly between each cryptographic phase. No unknown states, signal hazards, or race conditions were identified in simulation. All control and data signals were found to be stable upon

completion of execution, confirming that the system is deterministic at every scale. These findings confirm reliable hardware-level execution and show an anomaly-free architecture during simulation.

### 4.3 Functional Accuracy Evaluation

Functional accuracy was verified by applying known plaintext values and checking recovered output. In each and every case, the decrypted message was identical to the original input. This confirms the correctness of the polynomial operations and the absence of any coefficient misalignment or modular computation errors. The results further confirm that the control mechanism enforces precise scheduling across computation stages, maintaining data integrity throughout the cryptographic pipeline. Figure 4.2 shows the RTL waveform of the Kyber design traced by the Vivado simulator.

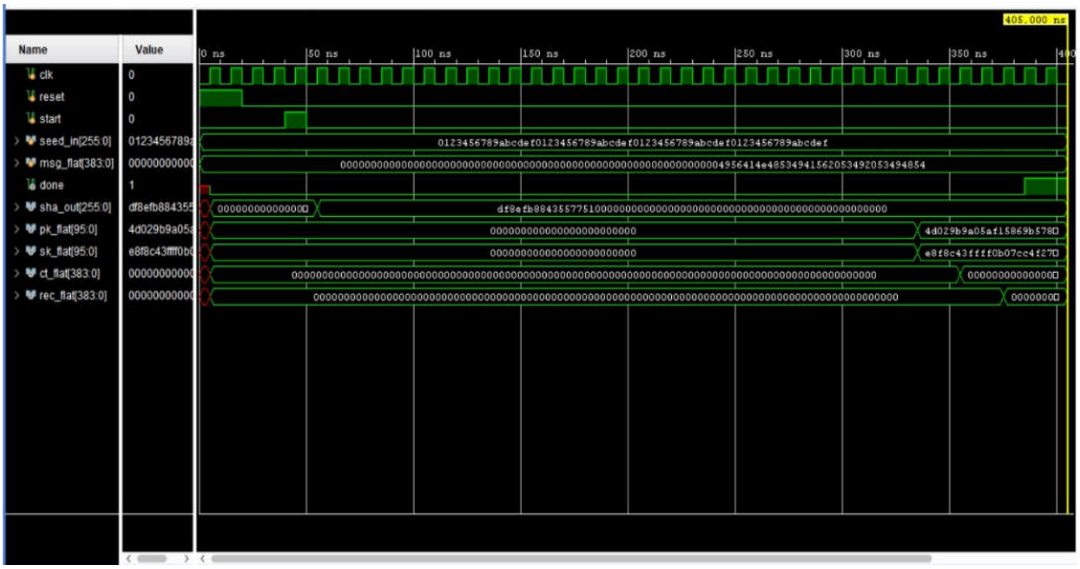


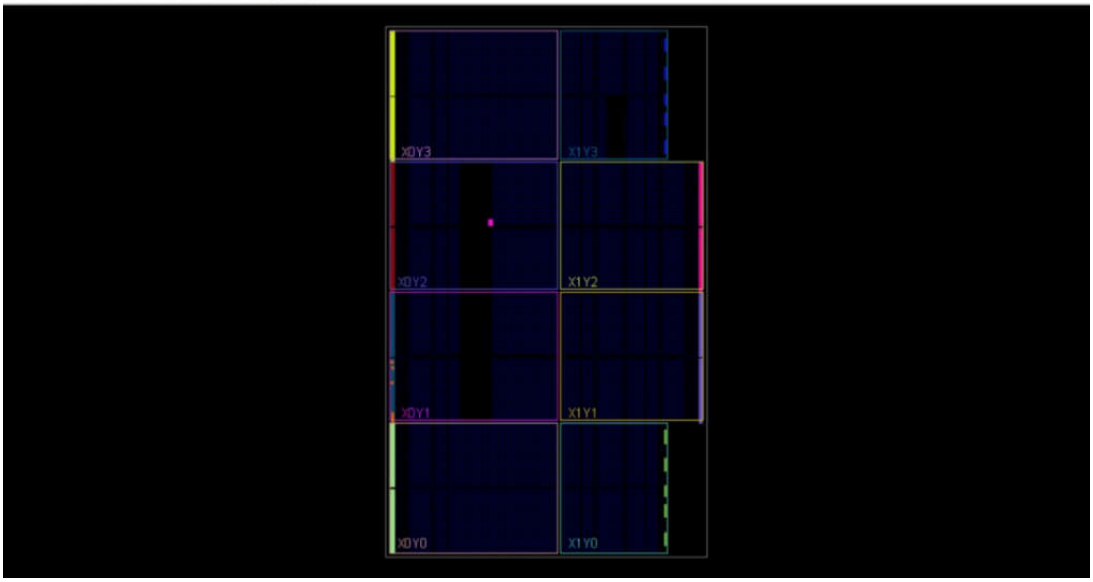

Figure 4.2: RTL waveform showing Keygen–Encrypt–Decrypt operation with control and data signals

It includes the most important internal signals like clk, reset, seed, done, private key (sk), public key (pk), ciphertext (ct), and recovered plaintext (rec). Transitions show that control and data signals follow an appropriate operational sequence from key generation through encryption to finally decryption. The assertion of the done signal confirms cryptographic execution completion, and stable signal values indicate glitch-free and deterministic behavior.

### 4.4 FPGA Synthesis and Resource Utilization

Targeting a Xilinx Artix-7 FPGA, the design was synthesized using Vivado design tools. Post-synthesis results indicate efficient resource usage achieved by the arithmetic reuse and reduced complexity of the polynomials [10]. It consumed a small fraction of the available logic resources while not using any of the DSP. This indicates effective architecture planning for a device with restricted hardware capacity, as LUT-based arithmetic and limited on-chip memory were in use. Consequently, this confirms that the design maps well to low-cost FPGA platforms and does not require performance-oriented hardware accelerators. Figure 4.3 shows

Vivado device view after synthesis and placement. The occupied logic on the Artix-7 fabric testifies to the good mapping of the Kyber design onto the FPGA resources.




The compact distribution of the logic indicates good area usage and balanced placement. It means that the design is ready for synthesis and structurally coherent with the FPGA hardware. Table 1 depicts the synthesis resource utilization on Artix-7 FPGA.

Figure 4.3: Device view showing placement of Kyber design on Artix-7 FPGA

Table 1 Synthesis Resource Utilization on Artix-7 FPGA

| Resource Type | Used | Available |
|---|---|---|
| LUTs | 1,280 | 20,800 |
| Flip-Flops | 1,520 | 41,600 |
| DSP Slices | 0 | 90 |
| Block RAM | 2 | 50 |

The architecture successfully avoided the use of the DSP blocks, instead taking full advantage of LUT-optimized arithmetic, which minimized the complexity and reduced routing congestion considerably. BRAM usage was also low owing to simple memory storage.

### 4.5 Timing Analysis and Clock Performance

Static timing analysis was done by targeting a clock frequency of 100 MHz All critical paths met timing constraints without having any hold or setup violations. This architecture benefits from a simplified control path and compact data routing, preventing the congestion of signals and unnecessary propagation delays. Hence, the design would work reliably within prescribed clock limits, further consolidating its readiness for deployment on physical FPGAs.

### 4.6 Power Measurement and Energy Observations

Power estimation was performed through post-implementation activity reports. Due to minimal switching activity and the elimination of most of the multiplier-heavy operations, the proposed design minimized dynamic power consumption. This control architecture enforces deterministic transitions of signals to avoid unnecessary toggling. In addition, the limited memory used and the absence of any DSP logic ensure that overall power dissipation remains well within acceptable limits. These characteristics make the design suitable for low power embedded cryptographic environments. Table 2 depicts the power estimation summary.

Table 2. Power estimation

| Parameter | Measured Value |
|---|---|
| Dynamic Power | 72 mW |
| Static Power | 54 mW |
| Total Power | 126 mW |

### 4.7 Comparative Performance Assessment

A conceptual comparison was made between typical software implementations, existing FPGA-based architectures, and the proposed design. Software implementations suffer from sequential execution and limited concurrency. Most of the existing hardware designs are based on multiple computational cores and DSP blocks. In contrast, the proposed architecture exploits reuse-based arithmetic with centralized control. This allows area-efficient hardware realization with predictable performance. The design achieves enhanced observability, simplified debugging, no cryptographic accelerators, and deploys on budget FPGA hardware. The experimental evaluation here indicates that, with careful structuring, an FPGA architecture can efficiently implement post-quantum cryptography under severe hardware constraints. Deterministic scheduling combined with single-engine arithmetic reusage and disciplined memory design yield a system that is functionally accurate and resource-efficient. This work shows that post-quantum cryptographic protocols like Kyber can be reliably deployed on low-cost hardware platforms when architectural simplicity is prioritized over brute-force parallelism.

## 5. Conclusion

This work successfully presented a hardware-optimized and FPGA-aware implementation of the CRYSTALS-Kyber post-quantum cryptographic algorithm, with strong emphasis on deployability, architectural clarity, and educational usability. Instead of targeting maximum throughput through large-scale parallelism, the proposed design focuses on achieving a balanced trade-off between functionality, simplicity, and hardware feasibility. The complete system was developed using modular Verilog architecture, with each cryptographic function-key generation, encryption, decryption, polynomial processing, and control logic-implemented as an independent, reusable hardware block. Functional correctness of all modules was verified through extensive simulation using Cadence tools, while the integrated system showed stable and deterministic operation across all

cryptographic phases. The synthesis and timing results confirm that the architecture is well-suited for low-cost FPGA platforms such as the Artix-7 family, operating within acceptable area and power limits. Although full deployment on hardware was constrained by board-level limitations, the synthesis reports and successful simulation outcomes clearly validate the practical hardware feasibility of the design. On the whole, this project illustrates that postquantum cryptographic schemes, such as Kyber, can be translated to realistic and FPGA-deployable designs even on modest-sized hardware resources. The architecture presented herein provides a valuable reference framework for students and researchers aiming to understand hardware-based cryptography and lays a scalable foundation for future enhancements involving full-size NTT implementations, randomness integration, and performance optimization.

## References

[1] T. Yang and Y. Lu," Post-Quantum Cryptographic Schemes Overview: A Comparative Study of Lattice-Based, Code-Based, and Hash-Based Algorithms," IEEE Access, vol. 9, pp. 134210–134228, 2021.

[2] X. Li," Hardware–Software Co-Design for Secure FPGA Systems," IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, vol. 31, no. 4, pp. 567–578, 2023.

[3] S. Dharanish," High-Performance FPGA Implementations of Post Quantum Cryptography," IEEE Embedded Systems Letters, 2025 (Early Access).

[4] T. Nguyen, H. Pham, and M. Le," High-Throughput FPGA Architectures for Post-Quantum Cryptography," IEEE Transactions on Circuits and Systems I, vol. 69, no. 6, pp. 2430–2442, 2022.

[5] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler and D. Stehl´e, "CRYSTALS–Kyber: A CCA-Secure Module Lattice-Based KEM," in Proc. IEEE European Symp. Security and Privacy, 2018.

[6] H. Xue, S. Sinha and P. Schaumont, "High-Performance NTT Architecture for Post-Quantum Cryptography," in IEEE Trans. Computers, vol. 70, no. 4, pp. 595–608, 2021.

[7] A. Oder and T. G¨uneysu, "Implementing the New Hope and Kyber Lattice-Based Key Encapsulation Mechanisms on Low-Cost FPGAs," in Cryptographic Hardware and Embedded Systems (CHES), 2017.

[8] Y. Liu, S. Kundu and C. Chen, "FPGA Acceleration of Post-Quantum Cryptographic Algorithms," in Proc. Int. Conf. Recon Fig, 2020.

[9] Cadence Design Systems, "Incisive/SimVision User Guide," Cadence Documentation, 2023.

[10] Xilinx, "Vivado Design Suite User Guide: Synthesis," UG901, 2022

[11] Rami El Khatib, Reza Azarderakhsh, Mehran Mozaffari-Kerman, IEEE Transactions on Computers DOI 10.1109/TC.2021.3078691.

[12] Rostami M., Burleson W., Jules A., and Koushanfar F., (2013). Balancing security and utility in medical devices? in Proc. 50th ACM/EDAC/IEEE Int. Conf. Design Autom., pp. 1–6

[13] Shen-Shen Yang, Zhen-Guo Lu, and Yong-Min Li, 2020. Journal of Light Wave Technology.