Bulletin of Electrical Engineering and Informatics
Vol. 15, No. 2, April 2026, pp. 1061~1069
ISSN: 2302-9285, DOI: 10.11591/eei.v15i2.10477

A review of the hardware implementation of CRYSTALS-Kyber post-quantum cryptography algorithm

Ardhi Wijayanto[1,2], Nabihah Ahmad[1], Salman Ahmed[1,3]
[1]Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia, Parit Raja, Malaysia
[2]Department of Informatics, Faculty of Information Technology and Data Science, Universitas Sebelas Maret, Surakarta, Indonesia
[3]Department of Electrical Engineering, Sukkur IBA University, Sukkur, Pakistan

Article Info

Article history:

Received Apr 14, 2025
Revised Feb 4, 2026
Accepted Mar 10, 2026

Keywords:

CRYSTALS-Kyber
Hardware implementation
Post-quantum cryptography
Quantum computing
Systematic literature review

ABSTRACT

The development of quantum computing is escalating the vulnerability of conventional cryptography algorithms. To answer this challenge, researchers develop the post-quantum cryptography (PQC) algorithms. The PQC algorithms are immune from attacks deployed by quantum computers. CRYSTALS-Kyber (abbreviated as Kyber) is a PQC algorithm, originally constructed as public key encryption (PKE), then extended as the key encapsulation mechanism (KEM) algorithm to securely transfer a shared key to other parties over unsecured communication channels. The implementation of Kyber algorithm in hardware ensures a better security standard for securing systems that prioritize performance. This study provides a literature review of the Kyber hardware implementations. The review is delivered by a systematic literature review method to discuss resource and performance optimization, key design constraints, performance trade-offs, and future research directions in hardware implementation of Kyber based on existing studies. Area utilization and energy efficiency are achieved through the optimization of memory and architecture. The trade-off between performance, flexibility, and utilization remains relevant in the deployment context. Future work should accommodate holistic solutions, security, and performance enhancements as well as fabrication for real-world solutions.

Corresponding Author:

Nabihah Ahmad
Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia
86400 Parit Raja, Johor, Malaysia
Email: nabihah@uthm.edu.my

1. INTRODUCTION

Quantum computing is a multidisciplinary study that is currently being developed to escalate the capability of conventional computers using the foundation of quantum mechanics. Complicated mathematics problems can be solved by quantum computers more efficiently compared to conventional computers [1]. By utilizing Shor's algorithm [2], quantum computers can resolve the factorization of large prime integers employed in public-key cryptosystems. As the quantum computers mature, the risk of security breaches in widely used cryptosystems such as Rivest-Shamir-Adleman (RSA) protocol will increase. In response, researchers develop the post-quantum cryptography (PQC) algorithms which is resistant to attacks deployed by quantum computers. PQC algorithms are based on intricate mathematical problems and encoding techniques such as lattice theory, learning with error problems, hash functions, multivariate polynomials, and error correction codes [3].

Among these PQC algorithms, Kyber evolved as the standardized algorithm for encryption and key exchange, recognized by the National Institute of Standards and Technology (NIST). This reflects NIST's

confidence in Kyber's ability to secure crucial communications data against quantum threats. Kyber builds its security based on the module learning-with-errors (MLWE) problem on module lattices, originally constructed as public key encryption (PKE) and then extended as the key encapsulation mechanism (KEM) algorithm to securely transfer a shared key to other parties over unsecured communication channels [4].

Because of Kyber's strong security and standardized role, hardware implementations are essential for securing performance-critical IoT and embedded systems like unmanned vehicles and industrial control systems [5]. Kyber hardware implementations can exploit the inherent computational parallelism [6], resulting in better performance than its software counterparts. This study provides a systematic literature review of the Kyber hardware implementations, especially in application-specific integrated circuits (ASIC) and field-programmable gate arrays (FPGA). This paper discusses existing study in resource and performance optimization, key design constraints, performance trade-offs, and future research directions in hardware implementation of Kyber. The goal of this paper is to present a comprehensive understanding of the state-of-the-art and highlight research gaps that can be valuable for directing future hardware-oriented PQC developments.

2. METHOD

This study follows the systematic literature review (SLR) method [7], [8] to collect, identify, evaluate, and interpret findings from multiple references on the specific research questions. Figure 1 displays the SLR guide to deploy literature review development that consists of four phases: planning, selection, extraction, and execution.

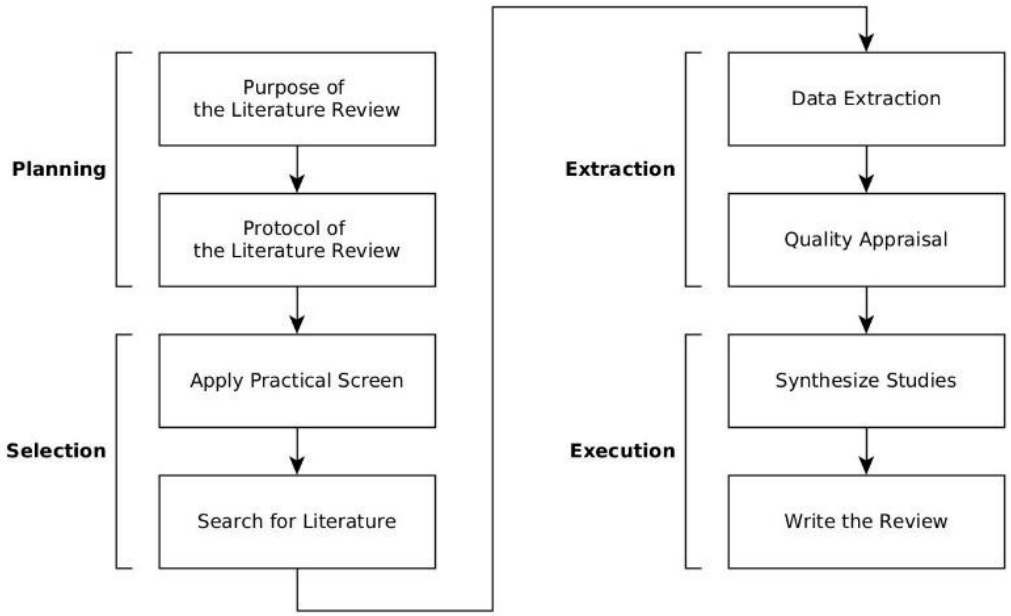

Figure 1. The phases to conduct systematic literature review [9]

2.1. Planning

The planning phase constructs a clear foundation for conducting the literature review. The first step is to formulate the literature review: to provide an overview of current advancements and identify the research gaps for directing future Kyber hardware research. The second step is to develop a protocol for the literature review.

In this study, three research questions are specified to guide the literature review process and ensure the consistency of the scope and focus. The research questions are defined as follows:

RQ1: What is the current state-of-the-art in resource and performance optimization in hardware implementations of Kyber?

RQ2: What are the key design constraints and trade-offs that influence optimization strategies in hardware implementations of Kyber?

RQ3: What are the future works highlighted in the literature on the hardware implementations of Kyber?

2.2. Selection

The selection phase ensures only relevant and high-quality references are included. Practical screening and exclusion criteria are used to determine whether a literature can be selected. Only English papers of at least three pages reporting FPGA or ASIC implementation of Kyber are considered. Next step is specifying the method for searching the literature. This study searches papers and conference proceedings from IEEEXplore, Scopus, and ScienceDirect that published from 2021 until 2025. The keywords used for searching the literature are "Kyber hardware" and "hardware implementation Kyber".

2.3. Extraction

In the extraction phase, the literature that met the inclusion criteria is analyzed. Relevant information that is directly related to Kyber hardware implementations is extracted. The key data items include the target hardware platform: FPGA and ASIC; performance metrics: area utilization, throughput, latency, energy, and power consumption; and implementation techniques: pipelining, parallelization, and memory optimization. The next step is a quality appraisal, which involves screening and evaluating the selected literature for quality and relevance. Exclusion criteria ensure only high-quality literature is selected.

2.4. Execution

The extracted data is then analyzed in the execution phase, involving study synthesis of studies to identify current state-of-the-arts, key-design constraints, and research gaps in Kyber hardware implementation. This synthesis identifies best practices and potential directions for future Kyber hardware research. To ensure academic rigor, the systematic process and results of the review are documented. This paper also details the methodology of the literature review, compares Kyber hardware implementation across selected studies, highlights key findings, and outlines future work in Kyber hardware studies.

3. RESULTS AND DISCUSSION

The extracted data from the extraction phase is analyzed in the execution phase. This involves synthesis of studies to identify the best practices and potential directions for future research in Kyber hardware design. The writing process includes the methodology for conducting the literature review process, a comparison of Kyber hardware implementation across selected literature, key findings that highlight performance and hardware-specific design considerations, and future work for advancing Kyber hardware implementation.

3.1. RQ1: What is the current state-of-the-art in resource and performance optimization in hardware implementations of Kyber?

The key optimization strategies to improve resource and performance are highlighted in Table 1, summarizing the current state-of-the-art in Kyber hardware implementation. Area reduction has been discussed in studies [10]–[16], significant improvement in area utilization is achieved by employing specialized number theoretic transform (NTT) and inverse NTT (INTT) architectures such as block RAM (BRAM)-free pipelined designs, lookup table (LUT)-based approaches, and unified cores. Other studies [17]–[23] adopt resource reuse techniques to realize lower hardware costs across NTT/INTT modules. This resource reuse techniques including register sharing, reconfigurable data paths, and first in first out (FIFO)-based pipelines. Efficient memory management also contributes to area savings. This strategy includes compact addressing schemes and single-bank memory for polynomial-wise multiplication [24]–[29]. Another approach targets the hash and arithmetic modules. Compact secure hash algorithm 3 (SHA-3) modules, folded Keccak implementations, and tightly integrated hashing with NTT or sampling reduce overheads, while arithmetic optimizations leverage efficient multiplication and modular reduction strategies [20], [25], [30]–[34].

Time optimization is predominantly achieved through pipelining and parallelization. Inter and intra-module pipelining, loop pipelining, and pipelined pointwise multiply-accumulate reduce latency within transform computations [12]–[16], [19], [20], [22]–[24]. Parallelization techniques, such as parallel polynomial sampling, merging preprocessing with NTT, and compact scheduling of operations, enable immediate execution of independent computations [26], [27], [29]–[33], [35]–[38]. Complementary scheduling and control approaches, such as inter-module cooperation and instruction-set-based synchronization, further enhance execution efficiency [14], [15], [19].

Other optimizations have been applied to improve the throughput by fully pipelining SHA-3, modified sampling techniques, and NTT/INTT computations, as well as parallelizing transform modules [22], [39]. In addition, several works emphasize polishing security-oriented designs. The goal is to provide side-channel resistance through masking-protected implementations, duplication with randomized clocks, and combined countermeasures [17], [40], [41]. Finally, energy optimization has attracted attention as an improvement

strategy. A study [42] has reduced energy consumption per bit by minimizing I/O data movement and pipelining entire operations of Kyber hardware.

Overall, the surveyed works demonstrate a consistent trend of balancing compactness, speed, throughput, security, and energy efficiency. The breadth of techniques reflects the multi-objective nature of Kyber hardware design, where improvements in one dimension often require careful trade-offs in others.

Table 1. Current state-of-the-art in resource and performance improvements of Kyber hardware

Optimization target	Techniques	Literature
Area	- BRAM-free, unified cores, LUT-based, reconfigurable architectures, compact/specialized NTT designs. - Register reuse, reusing computing cores, reconfigurable datapath, continuous pipelines with FIFO interleaving, NTT/INTT modules reuse. - Avoid BRAMs, compact memory addressing, efficient memory access, single-bank memory for PWM, FIFO-based designs. - Compact SHA-3 module, half-fold Keccak, tightly integrated Keccak with NTT or sampling, unified hashing and arithmetic units. - Optimized SPM, Barrett reduction, configurable PEs with Dadda tree modular reduction.	[10]-[34]
Time	- Inter and intra-module pipelining, loop pipelining, dataflow pipelining, pipelined pointwise multiplication, memory-based pipelined NTT/INTT. - Parallel polynomial sampling, customized memory, merge preprocessing and NTT, compact scheduling, immediate computation of parallel iterations. - Inter-module cooperation, instruction-set-based scheduling, synchronized timing diagrams.	[12]–[16], [19], [20], [23], [24], [26], [27], [29]–[32], [36]
Throughput	- Fully pipelined SHA-3, sampling, and NTT computations. - Parallelization of NTT/INTT modules.	[22], [39]
Security-oriented design	- Masking-protected designs, duplication with clock randomization, combined SCA countermeasures.	[17], [40], [41]
Energy	- Reduce I/O data movement, pipelining entire operations to minimize energy per bit.	[42]

3.2. RQ2: What are the key design constraints and trade-offs that influence optimization strategies in hardware implementations of Kyber?

Table 2 highlights the key design constraints and trade-offs that shape existing Kyber hardware implementations. One of the most prominent challenges is balancing time and area efficiency, as reducing execution time often requires additional hardware resources and vice versa. For example, replacing NTT method with schoolbook polynomial multiplication (SPM) in Kyber can reduce the area for polynomial multiplication but produce slower performance [10]. Whereas a pipelined decimation in frequency (DIF)-based NTT with pre-processing integration [19] and unified NTT/INTT cores helps save area, but it suffers from bigger latency and requires mitigation by implementing pipelining and modified reduction algorithms [12]. Similarly, increasing the number of butterfly units (BUs) or parallelizing polynomial cores can significantly reduce computation time but expose higher area utilization [13], [14], [20], [22]. To overcome this, several works [14], [30] employing pipelined design combined with architectural reuse, where inter and intra-module pipelining is applied to enhance speed while minimizing additional area overhead.

Other strategies to optimize performance and keep low area utilization involve Keccak cores design [15], where dual implementations are applied for either low-area or high-performance [14]. Several works [23], [24], [34] utilized lightweight Keccak cores designs. Improvement also applied to arithmetic operations such as adopting unified multipliers that combine NTT, auxiliary modules, and sampling into a compact Datapath [18]. Other strategies include employing approximate modular reduction [29], adopt polynomial arithmetic with synchronized scheduling [26], [27], [32], and implementing configurable processing elements (PE) with Dadda tree reduction [32]. Another method to overcome the area vs time trade-off is employing compact scheduling with predefined sequence tables combined with a configurable FIFO mechanism [31]. Additionally, joint pipelining with parallel hashing has also been considered [24], [32], [33].

A second trade-off involves area versus data capacity, where the capacity of data processed will determine the area utilization. This trade-off is considered for resource-constrained platforms. Signed number representations have been explored to reduce memory requirements, though with a limited data range [10]. Another strategy involves FPGA, replacing BRAM with FIFO-based memory reduces block memory consumption but introduces minor latency overheads [16]. Similarly, employing efficient memory access schemes and single-bank storage reduces area but sacrifices flexibility and throughput [21], [25]. Another study use compact memory addressing to reduce hardware footprint, but it also reduces the memory capacity [32].

The trade-off between throughput and area is another critical factor. Several studies [20], [22], [32], [33], [39] focus on improving the throughput by employing pipelined design on the SHA-3, sampling, and NTT architectures. Similarly, parallelization of NTT/INTT modules and polynomial multipliers, substantially increases throughput but also adds resource utilization significantly.

In addition to performance and resource considerations, designers must also account for security versus performance/area trade-offs. Side-channel countermeasures such as masking-protected decapsulation [40], duplication with clock randomization [41], and combined side-channel attacks (SCA) protections [17] are proven methods to enhance protection from SCA but inevitably lead to larger area usage and increased latency. Finally, energy vs. performance trade-offs is important in low-power environments. Reducing I/O by pipelining operations can minimize data movements to save energy, though it may slightly increase execution time [42].

Taken together, these trade-offs illustrate the multi-dimensional nature of hardware design for Kyber PQC. No single strategy can optimize all constraints simultaneously. The choice of techniques depends on the priority, whether it concerns on compactness, performance, or security robustness. A combined strategy may be considered to achieve the best results.

Table 2. Key design constraints and trade-off

Trade-offs	Design constraints	Strategy	Literature
Time vs area	Balanced compact area and fast execution time	- Use SPM instead of NTT (slower but smaller area).	[10]
		- Unified NTT/INTT core, pipelined reduction algorithms.	[12]
		- Increase number of BUs, faster but more area utilization.	[16], [20]
		- Parallel polynomial cores, faster but larger area.	[13], [22]
		- Reuse NTT/INTT architectures, inter/intra-module pipelining.	[14], [30]
		- Reuse Keccak cores; configurable BUs and ping-pong RAM.	[15]
		- High-performance vs. low-area SHA3 core.	[17]
		- Combine NTT, auxiliary modules, and rejection sampling into compact datapath.	[18]
		- Pipelined DIF NTT, pre-processing integration, and twiddle factor.	[19]
		- Compact SHA-3/Keccak cores.	[23], [24], [34]
		- Unified polynomial arithmetic and synchronized scheduling.	[26], [27], [32]
		- Barrett reduction with approximation, internal instruction sets.	[29]
		- Compact scheduling combined with configurable FIFOs.	[31]
		- Configurable PE and modular reduction via Dadda tree.	[32]
		- Joint pipelining and parallelism with hashing/sampling.	[24], [32], [33]
Area vs data capacity	Minimal area for resource-constrained devices	- Use signed number representation.	[10]
		- Replace BRAM with FIFO-based memory.	[16]
		- Efficient memory access and single-bank storage.	[21], [25]
		- Compact memory addressing for polynomial arithmetic.	[32]
Throughput vs area	Balancing area utilization and throughput	- Fully pipelined SHA-3, sampling, and NTT computations.	[22], [39]
		- Parallelization of NTT/INTT and polynomial multipliers.	[20], [32], [33]
Security vs performance/area	Keep low area and latency while provide side-channel resistance	- Use masking-protected decapsulation.	[40]
		- Duplication with clock randomization for SCA resistance.	[41]
		- Employ multiple SCA countermeasures.	[17]
Energy vs performance	Energy saving for low-power environments	- Reduce I/O by pipelining entire operation.	[42]

3.3. RQ3: What are the future works highlighted in the literature on the hardware implementations of Kyber?

Table 3 outlines future research directions for Kyber hardware implementations. A key discussion is to support multiple lattice-based PQC schemes through a unified architecture. Several works [18], [27], [29], [32], [36] started to design unified NTT/INTT cores to enable Kyber, Dilithium, and Saber to be implemented on a single chip. The next studies can explore the trade-off between performance and utilization of a unified design and its optimization.

Another interesting direction is the development of comprehensive protection against implementation attacks. Current methods such as masking, duplication, clock randomization, and secure memory management have proven effective but have the prohibited cost in area, latency, and power [14], [17], [21], [23]–[25], [27], [32], [36], [39]–[41]. Future works must highlight the strategies to achieve balance between security and efficiency, ensure the suitability for both constrained and high-performance scenarios.

The future research on Kyber implementation must address complete algorithm support and parameter scalability. To get more realistic evaluations and readiness of implementation, the hardware designs should deliver key generation, encapsulation, and decapsulation operations across all NIST security levels [19], [28],

[31]. Fabrication and real-world evaluation are the next topics for future research. Most implementations of Kyber have been demonstrated on FPGA platforms. ASIC-based prototypes can be investigated to get more accurate long-term performance evaluation [20], [39], [42].

Another future research direction lies in optimizations for constrained devices, where resources are limited. Technical approaches such as BRAM-free designs, compact addressing, and reduced I/O should be refined to retain acceptable performance in resource-constrained environments [10], [16], [21], [25], [41], [42]. Collaboration between software libraries and hardware accelerators can be considered to deliver flexible, high-performance, and portable Kyber deployment [38]. Additionally, energy-efficient design remains essential for portable and embedded devices [16], [21], [25], [42]. Several techniques such as optimized pipelining depth, reduced I/O activity, and energy-aware datapath design can be explored to improve the energy efficiency.

Performance enhancement is also mentioned in the literature. Future work on Kyber hardware implementation should consider cross-scheme techniques such as advanced modular reduction, specialized scheduling, or hashing optimizations to be adapted in Kyber [19], [30], [32], [33], [37]. Complementary, further exploration is needed to integrate Kyber into real-world applications such as communication systems, embedded devices, and cloud hardware accelerators. Designing Kyber hardware for practical systems should verify that the security standard and performance meet the reliability of existing infrastructure [24], [26], [32], [39]. Standardized benchmarking needs to be refined to verify the performance and provide a fair comparison across the implementation [20].

Overall, future research should shift from isolated optimizations toward holistic design frameworks that integrate scalability, security, adaptability, and energy efficiency, while ensuring comparability and readiness for real-world deployment.

Table 3. Future research directions in Kyber hardware implementation

Future research direction	Description	Literature
Adaptability to other lattice-based PQC schemes	Extend architectures to support multiple lattice-based cryptosystems.	[18], [27], [29], [32], [36]
Protection against implementation attacks	Develop lightweight and effective countermeasures for side-channel and fault attacks.	[14], [17], [21], [23]–[25], [27], [32], [36], [39]–[41]
Full algorithm support and parameter scalability	Expand implementations to cover all operations and all parameter sets in Kyber algorithm.	[19], [28], [31]
Fabrication and real-world evaluation	Encourage ASIC chip fabrication. Evaluate area, energy, and performance in practical deployment environments.	[20], [39], [42]
Optimizations for constrained devices	Tailor architectures for resource-constrained devices, while retaining acceptable performance.	[10], [16], [21], [25], [41], [42]
Integration into real-world applications	Explore how optimized Kyber cores can be integrated into real-world cases and meet the real applications standard.	[24], [26], [32], [39]
Performance enhancement through cross-scheme techniques	Borrow optimization methods from other PQC or cryptographic schemes to further accelerate Kyber.	[19], [30], [32], [33], [37]
Improving energy efficiency	Energy-optimized designs to enable low-power Kyber implementations.	[16], [21], [25], [42]
Hardware–software co-design approaches	Investigate co-optimization between software libraries and hardware accelerators to improve Kyber deployments.	[38]
Standardized benchmarking and fair comparison	Refining common evaluation metrics and benchmarking methods across platforms to enable fair performance comparison.	[20]

4. CONCLUSION

This review paper presents a detailed analysis of the existing scientific literature focused on the hardware implementation of the Kyber PQC algorithm, highlights the recent advancements and state-of-the-art techniques. The surveyed works demonstrate a consistent trend of balancing performance and resource utilization through optimization. However, the improvements come with trade-offs in different dimensions, illustrating the multi-dimensional nature of hardware design for Kyber. The choice of improvement techniques should be guided by design priorities. A combined strategy may be considered to achieve the best results. Future research should move toward holistic design frameworks that integrate performance, efficiency, and security while ensuring comparability and readiness for real-world applications. These findings provide valuable insights for the next iterations of studies in the field of Kyber PQC hardware design. Exploration and comparison with other PQC schemes will be the next step to obtain more comprehensive results.

ACKNOWLEDGMENTS

Communication of this research is made possible through monetary assistance by the UTHM Publisher's Office via Publication Fund E15216 and the Ministry of Higher Education (MOHE) through the Fundamental Research Grant Scheme under Grant FRGS/1/2022/TK07/UTHM/02/20.

FUNDING INFORMATION

This work was supported by the Ministry of Higher Education (MOHE) through the Fundamental Research Grant Scheme under Grant FRGS/1/2022/TK07/UTHM/02/20.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Ardhi Wijayanto	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	
Nabihah Ahmad	✓	✓				✓	✓	✓		✓	✓	✓	✓	✓
Salman Ahmed			✓	✓			✓	✓	✓	✓	✓			

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] L. Gyongyosi and S. Imre, "A survey on quantum computing technology," *Computer Science Review*, vol. 31, pp. 51–71, Feb. 2019, doi: 10.1016/j.cosrev.2018.11.002.
[2] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134, doi: 10.1109/SFCS.1994.365700.
[3] S. Li *et al.*, "Post-quantum security: Opportunities and challenges," *Sensors*, vol. 23, no. 21, Oct. 2023, doi: 10.3390/s23218744.
[4] R. Avanzi *et al.*, "CRYSTALS-Kyber algorithm specifications and supporting documentation," *NIST PQC Round*, vol. 2, no. 4, pp. 1–42, 2019.
[5] S. Farooq *et al.*, "Resilience optimization of post-quantum cryptography key encapsulation algorithms," *Sensors*, vol. 23, no. 12, Jun. 2023, doi: 10.3390/s23125379.
[6] W. Guo and S. Li, "Split-radix based compact hardware architecture for CRYSTALS-Kyber," *IEEE Transactions on Computers*, vol. 73, no. 1, pp. 97–108, Jan. 2024, doi: 10.1109/TC.2023.3320040.
[7] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," EBSE Technical Report EBSE-2007-01, 2007.
[8] D. Pati and L. N. Lorusso, "How to write a systematic review of the literature," *HERD: Health Environments Research & Design Journal*, vol. 11, no. 1, pp. 15–30, Jan. 2018, doi: 10.1177/1937586717747384.
[9] C. Okoli, "A guide to conducting a standalone systematic literature review," *Communications of the Association for Information Systems*, vol. 37, 2015, doi: 10.17705/1CAIS.03743.
[10] S. He, H. Li, F. Li, and R. Ma, "A lightweight hardware implementation of CRYSTALS-Kyber," *Journal of Information and Intelligence*, vol. 2, no. 2, pp. 167–176, Mar. 2024, doi: 10.1016/j.jiixd.2024.02.004.
[11] Z. Ni, A. Khalid, W. Liu, and M. O'Neill, "Towards a Lightweight CRYSTALS-Kyber in FPGAs: an Ultra-lightweight BRAM-free NTT Core," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2023, pp. 1–5, doi: 10.1109/ISCAS46773.2023.10181340.
[12] N. N. Dinh, T. T. Nguyen, T. H. Vu, V. T. D. Le, and H. L. Pham, "Hardware/software Co-design of a multi-mode CRYSTALS-Kyber accelerator for quantum-secure Internet-of-Things systems," in *2024 International Conference on Advanced Technologies for Communications (ATC)*, Oct. 2024, pp. 473–478, doi: 10.1109/ATC63255.2024.10908296.
[13] W. Guo, S. Li, and L. Kong, "An efficient implementation of Kyber," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1562–1566, Mar. 2022, doi: 10.1109/TCSII.2021.3103184.
[14] Z. Ni, A. Khalid, D.-S. Kundi, M. O'Neill, and W. Liu, "HPKA: A high-performance CRYSTALS-Kyber accelerator exploring

efficient pipelining," *IEEE Transactions on Computers*, vol. 72, no. 12, pp. 3340–3353, Dec. 2023, doi: 10.1109/TC.2023.3296899.
[15] Q. Zeng, Q. Li, B. Zhao, H. Jiao, and Y. Huang, "Hardware design and implementation of post-quantum cryptography Kyber," in *2022 IEEE High Performance Extreme Computing Conference (HPEC)*, Sep. 2022, pp. 1–6, doi: 10.1109/HPEC55821.2022.9926344.
[16] Z. Ni, A. Khalid, W. Liu, and M. O'Neill, "A highly hardware efficient ML-KEM accelerator with optimised architectural layers," *ACM Transactions on Embedded Computing Systems*, vol. 24, no. 2, pp. 1–24, Mar. 2025, doi: 10.1145/3708469.
[17] A. Jati, N. Gupta, A. Chattopadhyay, and S. K. Sanadhya, "A configurable CRYSTALS-Kyber hardware implementation with side-channel protection," *ACM Transactions on Embedded Computing Systems*, vol. 23, no. 2, pp. 1–25, Mar. 2024, doi: 10.1145/3587037.
[18] S. Chowdhury, N. Mishra, S. Bhattacharya, and D. Mukhopadhyay, "Unified FPGA design of Kyber and dilithium with provable fault tolerance," in *2025 IEEE 36th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, Jul. 2025, pp. 158–165, doi: 10.1109/ASAP65064.2025.00033.
[19] K. Yao, D.-E.-S. Kundi, C. Wang, M. O'Neill, and W. Liu, "Towards CRYSTALS-Kyber: A M-LWE cryptoprocessor with area-time trade-off," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2021, pp. 1–5, doi: 10.1109/ISCAS51556.2021.9401253.
[20] M. Bisheh-Niasar, R. Azarderakhsh, and M. Mozaffari-Kermani, "A monolithic hardware implementation of Kyber: comparing apples to apples in PQC candidates," in *Progress in Cryptology – LATINCRYPT 2021*, 2021, pp. 108–126, doi: 10.1007/978-3-030-88238-9_6.
[21] V. B. Dang, K. Mohajerani, and K. Gaj, "High-speed hardware architectures and FPGA benchmarking of CRYSTALS-Kyber, NTRU, and saber," *IEEE Transactions on Computers*, vol. 72, no. 2, pp. 306–320, Feb. 2023, doi: 10.1109/TC.2022.3222954.
[22] M. Bisheh-Niasar, R. Azarderakhsh, and M. Mozaffari-Kermani, "High-speed NTT-based polynomial multiplication accelerator for post-quantum cryptography," in *2021 IEEE 28th Symposium on Computer Arithmetic (ARITH)*, Jun. 2021, pp. 94–101, doi: 10.1109/ARITH51176.2021.00028.
[23] T.-H. Nguyen, D.-T. Dam, P.-P. Duong, B. Kieu-Do-Nguyen, C.-K. Pham, and T.-T. Hoang, "Efficient hardware implementation of the lightweight CRYSTALS-Kyber," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 72, no. 2, pp. 610–622, Feb. 2025, doi: 10.1109/TCSI.2024.3443238.
[24] T.-H. Nguyen *et al.*, "An area-time efficient hardware architecture for ML-KEM post-quantum cryptography standard," *IEEE Access*, vol. 13, pp. 103834–103847, 2025, doi: 10.1109/ACCESS.2025.3579379.
[25] W. Guo and S. Li, "Highly-efficient hardware architecture for CRYSTALS-Kyber with a novel conflict-free memory access pattern," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 11, pp. 4505–4515, Nov. 2023, doi: 10.1109/TCSI.2023.3306347.
[26] H. Jung, Q. D. Truong, and H. Lee, "Highly-efficient hardware architecture for ML-KEM PQC standard," *IEEE Open Journal of Circuits and Systems*, vol. 6, pp. 356–369, 2025, doi: 10.1109/OJCAS.2025.3591136.
[27] A. Aikata, A. C. Mert, M. Imran, S. Pagliarini, and S. S. Roy, "KaLi: A crystal for post-quantum security using kyber and dilithium," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 2, pp. 747–758, Feb. 2023, doi: 10.1109/TCSI.2022.3219555.
[28] P. Dobiáš and L. Malina, "Optimizing components for Dilithium and Kyber unified hardware implementation," in *Proceedings II of the 30st Conference STUDENT EEICT 2024: Selected papers*, 2024, pp. 171–175, doi: 10.13164/eeict.2024.171.
[29] Y. Cui, J. Chen, Z. Ni, Z. Zhang, C. Wang, and W. Liu, "Instruction-based high-performance hardware controller of CRYSTALS-Kyber with balanced resource utilization," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 72, no. 5, pp. 2394–2407, May 2025, doi: 10.1109/TCSI.2025.3547799.
[30] H. Kim, H. Jung, A. Satriawan, and H. Lee, "A configurable ML-KEM/Kyber key-encapsulation hardware accelerator architecture," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 11, pp. 4678–4682, Nov. 2024, doi: 10.1109/TCSII.2024.3442228.
[31] Y. Xing and S. Li, "A compact hardware implementation of CCA-secure key exchange mechanism CRYSTALS-Kyber on FPGA," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 328–356, Feb. 2021, doi: 10.46586/tches.v2021.i2.328-356.
[32] M. Bisheh-Niasar, R. Azarderakhsh, and M. Mozaffari-Kermani, "Instruction-set accelerated implementation of CRYSTALS-Kyber," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 11, pp. 4648–4659, Nov. 2021, doi: 10.1109/TCSI.2021.3106639.
[33] J. Zhang *et al.*, "Super-K: A superscalar CRYSTALS-Kyber processor based on efficient arithmetic array," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 9, pp. 4286–4290, Sep. 2024, doi: 10.1109/TCSII.2024.3382772.
[34] A. Li *et al.*, "A 273μW 0.34mm 2 efficient CRYSTALS-Kyber processor for PQC towards edge computing," in *2024 IEEE European Solid-State Electronics Research Conference (ESSERC)*, Sep. 2024, pp. 472–475, doi: 10.1109/ESSERC62670.2024.10719541.
[35] X. Carril *et al.*, "Hardware acceleration for high-volume operations of CRYSTALS-Kyber and CRYSTALS-Dilithium," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 17, no. 3, pp. 1–26, Sep. 2024, doi: 10.1145/3675172.
[36] P. Dobias, L. Malina, and J. Hajny, "Efficient unified architecture for post-quantum cryptography: Combining Dilithium and Kyber," *PeerJ Computer Science*, vol. 11, Mar. 2025, doi: 10.7717/peerj-cs.2746.
[37] T. T. Nguyen, S. Kim, Y. Eom, and H. Lee, "Area-time efficient hardware architecture for CRYSTALS-Kyber," *Applied Sciences*, vol. 12, no. 11, May 2022, doi: 10.3390/app12115305.
[38] P. Jedlicka and J. Hajny, "VHDL-based implementation of CRYSTALS-Kyber components on FPGA," in *Proceedings II of the 28st Conference STUDENT EEICT 2022: Selected papers.*, 2022, pp. 297–301, doi: 10.13164/eeict.2022.297.
[39] S.-H. Chou, Y.-H. Yang, W.-L. Chin, C. Chen, C.-Y. Tsao, and P.-L. Tung, "High-throughput post-quantum cryptographic system: CRYSTALS-Kyber with computational scheduling and architecture optimization," *Electronics*, vol. 14, no. 15, Jul. 2025, doi: 10.3390/electronics14152969.
[40] Y. Zhao *et al.*, "Side channel security oriented evaluation and protection on hardware implementations of Kyber," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 12, pp. 5025–5035, Dec. 2023, doi: 10.1109/TCSI.2023.3288600.
[41] M. Moraitis, Y. Ji, M. Brisfors, E. Dubrova, N. Lindskog, and H. Englund, "Securing CRYSTALS-Kyber in FPGA using duplication and clock randomization," *IEEE Design & Test*, vol. 41, no. 5, pp. 7–16, Oct. 2024, doi: 10.1109/MDAT.2023.3298805.
[42] T. Shimada and M. Ikeda, "High-speed and energy-efficient crypto-processor for post-quantum cryptography CRYSTALS-Kyber," in *2022 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, Nov. 2022, pp. 12–14, doi: 10.1109/A-SSCC56115.2022.9980610.

BIOGRAPHIES OF AUTHORS

Ardhi Wijayanto is a Ph.D. student in Electrical Engineering at Universiti Tun Hussein Onn Malaysia. He is working on IC design in post-quantum cryptography research. He is also a lecturer within the Faculty of Information Technology and Data Science at Universitas Sebelas Maret, Indonesia. His interests encompass IoT, open-source software, and data mining. He has worked with students on IoT device implementations specifically for agricultural purposes. He can be contacted at email: he220015@student.uthm.edu.my.

Nabihah binti Ahmad is a senior lecturer and the Head of the Department of Graduate Studies at the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia (UTHM). She obtained her Bachelor's degree with Honors in Electrical, Electronic, and Systems Engineering from Universiti Kebangsaan Malaysia in 2002. In 2006, she earned a Master's degree in Electronic Engineering (Microelectronics) from Kolej Universiti Tun Hussein Onn Malaysia. She earned her Ph.D. in Electronic Engineering at Massey University, New Zealand, in 2014. Her research focuses on digital and analog IC design, low-power VLSI circuit design, cryptographic co-processors, and FPGA technology. She teaches courses such as sensors and embedded systems, digital electronics, VLSI systems, digital design, research methodology, and engineering management. Additionally, she is a member of the Board of Engineers Malaysia (BEM). She can be contacted at email: nabihah@uthm.edu.my.

Salman Ahmed is currently a Ph.D. candidate and a graduate research assistant at the Faculty of Electrical and Electronics Engineering, Universiti Tun Hussein Onn Malaysia (UTHM). He obtained his Bachelor's degree in Electronic Engineering from Mehran University of Engineering and Technology, Jamshoro, Pakistan, in 2015. In 2021, he completed his Master of Engineering in Industrial Automation and Control at Quaid-e-Awam University of Science and Technology, Nawabshah, Pakistan. Between 2017 and 2022, he served as a lecturer in Electronics Engineering at Sukkur IBA University, Pakistan. His research focuses on low-power ASIC hardware design for cryptographic algorithms and resource-efficient IoT devices. He can be contacted at email: he220025@student.uthm.edu.my.