ISSN: 2182-2069 / E-ISSN: 2182-2077

High-Performance ALTEQ Hardware Accelerator

Yong Ryeol Choi[1], Young Beom Kim[2], Willy Susilo[3], Okyeon Yi[4],
Ilsun You[5], and Seog Chung Seo[6*]

[1]Department of Cyber Security, Kookmin University, South Korea. mike0726@kookmin.ac.krs,
https://orcid.org/0009-0005-5117-4523

[2]Department of Financial Information Security, Kookmin University, South Korea.
darania@kookmin.ac.kr, https://orcid.org/0000-0003-4715-8393

[3]Institute of Cybersecurity and Cryptology (iC2), University of Wollongong, Australia.
wsusilo@uow.edu.au, https://orcid.org/0000-0002-1562-5105

[4]Department of Cyber Security, Kookmin University, South Korea. oyyi@kookmin.ac.kr,
https://orcid.org/0000-0001-8156-2360

[5]Department of Cyber Security, Kookmin University, South Korea. isyou@kookmin.ac.kr,
https://orcid.org/0000-0002-0604-3445

[6*]Department of Cyber Security, Kookmin University, South Korea. scseo@kookmin.ac.kr,
https://orcid.org/0000-0001-8016-2808

Received: August 11, 2025; Revised: September 19, 2025; Accepted: October 29, 2025; Published: November 28, 2025

Abstract

With the advent of quantum computing threatening the security of modern public-key cryptography, the standardization of Post-Quantum Cryptography (PQC) is actively underway worldwide. However, optimization research has predominantly focused on software implementations and lattice-based algorithms, leaving a significant gap in hardware acceleration for non-lattice-based schemes, particularly on platforms like FPGAs. This paper addresses this gap by proposing the first hardware/software (HW/SW) co-design to accelerate ALTEQ, a non-lattice-based digital signature algorithm submitted to the NIST PQC Additional Signatures competition. Our approach strategically offloads ALTEQ's most computationally intensive operations, namely hash functions and modular arithmetic, to a dedicated hardware accelerator, thereby it could maximize performance. To the best of our knowledge, this is the first study to investigate hardware acceleration architecture for ALTEQ or similar schemes. The proposed design demonstrates that even algorithms not selected for standardization can achieve significant performance improvements through hardware optimization, suggesting the potential for re-evaluating such schemes for specific application environments and encouraging further research into hardware implementations for a broader range of PQC algorithms.

Keywords: Post-Quantum Cryptography (PQC), ALTEQ Digital Signature, Hardware Acceleration, HW/SW co-design, FPGA Implementation, Modular Arithmetic Optimization.

Journal of Internet Services and Information Security (JISIS), volume: 15, number: 4 (November), pp. 473-483.
DOI: *10.58346/JISIS.2025.I4.034*

*Corresponding author: Department of Cyber Security, Kookmin University, South Korea.

1 Introduction

The proposal of Shor's algorithm in 1994 (Shor, 1994) revealed that modern public-key cryptosystems based on the discrete logarithm and integer factorization problems could be solved in polynomial time in a quantum computing environment. With recent advancements in quantum computing technology, the quantum computer is no longer a theoretical concept but is poised to become a reality soon. As the commercialization of this technology would render current cryptosystems insecure, extensive research has been conducted on Post-Quantum Cryptography (PQC), which aims to provide security even in the quantum era.

Since 2017, the National Institute of Standards and Technology (NIST) has been holding a PQC competition to select standardization algorithms, culminating in the selection of two Key Encapsulation Mechanisms (KEMs) and two Digital Signature Algorithms (DSAs) in 2024. To select additional schemes, NIST has proceeded with a 4th round for KEMs and an "Additional Signatures" track, with the final KEM algorithm having been recently chosen. Similarly, South Korea initiated its own KpqC competition in 2021 to select national standard PQC algorithms, and in 2025, two KEMs and two DSAs were selected, akin to the NIST process (Cheon & Johansson, 2017).

The algorithms submitted to these competitions have been the subject of numerous optimization studies to achieve faster performance while ensuring quantum security across all devices (Braun-Dubler et al., 2020). Indeed, significant research has been conducted on algorithms not selected for final standardization, particularly in embedded environments like Cortex-M4 (Abdulrahman, 2022; Greconici, 2021) and RISC-V (Li, 2024; Greconici, 2020), or SIMD environments such as AVX2 (Yu, 2024) (Seiler). However, optimization on hardware devices like FPGAs has received comparatively little attention. Furthermore, non-lattice-based algorithms have been explored far less than their lattice-based counterparts.

To address this research gap and explore the acceleration potential of non-lattice-based algorithms, we investigated schemes from the ongoing NIST Additional Signatures track. Among them, we identified ALTEQ (Bläser, 2023) as a promising candidate, as its computational load is dominated by hash functions and modular arithmetic, suggesting a high potential for performance improvement through a HW/SW co-design approach. Consequently, we propose the first HW/SW co-design for performance acceleration targeting ALTEQ, a Round 1 submission. To the best of our knowledge, no prior research has focused on accelerating ALTEQ or similar schemes. This work demonstrates that even algorithms that did not advance to later rounds of the competition possess significant potential for acceleration (Singhal et al., 2019). This could, in turn, provide an opportunity to re-evaluate cryptographic schemes that, while not selected for standardization, may hold potential for specific application environments (Alagic et al., 2024).

2 Preliminary

2.1 Notations

In this paper, A^T denotes a transpose of matrix A. For finite field $\mathbb{F}_q$ and trilinear form $\phi: \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$, linear map ϕ is called Alternating Trilinear Form (ATF) if and only if $\phi(u,u,w) = \phi(u,v,v) = \phi(w,v,w) = 0, \forall u,v,w \in \mathbb{F}_q^n$ holds. For general linear group $GL(n, \mathbb{F}_q)$ of degree n over $\mathbb{F}_q$, $\circ$ denotes a group action that satisfies $(A,\phi) \mapsto \phi \circ A: (u,v,w) \mapsto \phi(A^T u, A^T v, A^T w)$. A matrix $C \in F^{n\times n}$ is a column matrix if and only if C is the following form:

$$C = \begin{pmatrix} 1 & 0 & \cdots & c_1 & \cdots & 0 & 0 \\ 0 & 1 & & c_2 & & 0 & 0 \\ & & & \vdots & & & \\ \vdots & & \ddots & c_i & \ddots & & \vdots \\ & & & \vdots & & & \\ 0 & 0 & \cdots & c_{n-1} & \cdots & 1 & 0 \\ 0 & 0 & & c_n & & 0 & 1 \end{pmatrix}$$

2.2 Alternating Trilinear Form Equivalence (ATFE)

The Alternating Trilinear Form Equivalence (ATFE) problem is defined as the challenge of finding the existence of $A \in GL(n, \mathbb{F}_q)$ that satisfies the relation $\phi(Au, Av, Aw) = \psi(u, v, w), \forall u, v, w \in \mathbb{F}_q$ for given ATFs, ϕ and ψ. Also, the promised search version of the ATFE is defined as follows: for given ATFs, ϕ and ψ such that $\phi, \psi: \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ with $\phi \sim \psi$ is guaranteed, find $A \in GL(n, \mathbb{F}_q)$ such that $\phi = \psi \circ A$. This version is called psATFE shortly.

This concept is extended to the C-psATFE problem. Given C ATFs, $\phi_1, \cdots, \phi_C$, for which the equivalence $\phi_i \sim \phi_j$ is guaranteed for all $i, j \in [1, C]$, the problem is to find a matrix $A \in GL(n, \mathbb{F}_q^n)$ and distinct indices i, jsuch that $\phi_i = \phi_j \circ A, i \neq j$. It has been proven that the best-known attacks against psATFE have a quantum attack complexity of $O(q^{n/3})$ using Tani's algorithm (Tani, 2007).

2.3 ALTEQ

Table 1: Keys and signature sizes for balanced/shortsig-ALTEQ

Version	Parameter Set	Parameters (n, q, r, K, C)	Private Key (bytes)	Public Key (bytes)	Signature (bytes)
Balanced	I	$(13,\ 2^{32}-5, 84, 22, 7)$	32	8040	15928
	III	$(20,\ 2^{32}-5, 201, 28, 7)$	48	31968	49048
ShortSig	I	$(13,\ 2^{32}-5, 16, 14, 458)$	32	523984	9560
	III	$(20,\ 2^{32}-5, 39, 20, 229)$	48	1044288	32552

ALTEQ is a Digital Signature Algorithm that was submitted to Round 1 (NIST, 2022) of the NIST Additional Signatures competition. Unlike many preceding algorithms based on lattices, codes, or multivariate polynomials, the security of ALTEQ is founded on a novel hard problem known as Alternating Trilinear Form Equivalence (ATFE). The signature scheme is designed by combining the Goldreich-Micali-Wigderson (GWM) (Goldreich, 1991) protocol with the Fiat-Shamir transformation (Fiat, 1986). To achieve Existential Unforgeability under Chosen Message Attack (EUF-CMA) security, ALTEQ specifically utilizes the C-psATFE variant of the ATFE problem. ALTEQ offers two parameter sets, I and III, which correspond to NIST security categories I/II and III/IV, respectively. Furthermore, for each parameter set, two versions are provided: "Balanced", which features compact signature and public key sizes, and "ShortSig," which is optimized for a shorter signature size. The public key, private key, and signature sizes for each security strength are detailed in [Table 1]. In its parameters, n, q, r, K, and C denote the vector space dimension, the finite field order, the round number, the number of ATFs, and the weight of the challenge, respectively.

In ALTEQ, there are group action functions that have noticeable performance bottlenecks. It is described at Section 3, so in this section, we describe some of them for understanding ALTEQ. There are two functions that compute the group action on ATFs: actingOnATFS and actingOnATFwColoumn_vec32. actingOnATFS $(\phi_{atf}, \{A^{col}\}, c)$ outputs c ATFs by each column matrix in $\{A^{col}\}$ independent acting on ϕ_{atf}. actingOnATFwColoumn_vec32 is called inside of actingOnATFS repeatedly, and it computes group action for each column matrix. Since group action is computed over $\mathbb{F}_q$, there are significant number of modular multiplications and reductions, which we will discuss in the next section. Since the group action on ATFs are denoted as $\circ$, one can easily find the action in the algorithm in the overall ALTEQ scheme. We describe the key generation algorithm in [Algorithm 1], there are large amount of group action in line 3 repeatedly, which causes bottleneck. Also, group action is computed in the sign generation and verification function. In the 'Commitment' phase, which in the sign generation function, a total of the number of rounds (r) group actions are performed to create the commitment ATFs ($\psi_i \leftarrow \phi_C \circ B_i$). This constitutes the main computational workload for this operation in the signing algorithm. In the sign verification, another group actions ($\psi_i' \leftarrow \phi_{\{c_i\}} \circ D_i'$) are performed to recalculate the commitments. Algorithm overviews of key generation, sign generation and verification function are descripted in [Algorithm 1-3].

ALTEQ also takes SHA-3 family as its hash function. Especially, ALTEQ uses SHA-256 (384, 512 resp.) for instance I (III, IV resp.). It is denoted as H in the algorithm description. During the entire ALTEQ process, H is called with various input length: 32 to 130K bytes for instance I. Due to the large amount of input data, the Keccak f-1600 permutation, which is called in absorb stage of SHA-3 family, is executed numerous times. We recall it at Section 3 and 4 to notice its computation bottleneck.

Input: The variable number $n \in \mathbb{N}$, a prime power q, the alternating trilinear form number $C+1$.
Output: Public key: $C+1$ alternating trilinear forms $\phi_i \in \mathrm{ATF}(n,q)$ such that $\phi_i \cong \phi_j$ for any $i, j \in \{0, \dots, C\}$.
Private key: C matrices $A_0, \dots, A_{C-1}$, such that $\phi_i \circ A_i = \phi_C$.
1 Randomly sample an alternating trilinear form $\phi_C : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$.
2 Randomly sample C invertible matrices, $A_0, \dots, A_{C-1} \in \mathrm{GL}(n,q)$.
3 For every $i \in \{0, \dots, C-1\}$, $\phi_i \leftarrow \phi_C \circ A_i$.
4 For every $i \in \{0, \dots, C-1\}$, $A_i \leftarrow A_i^{-1}$.
5 **return** *Public key:* $\phi_0, \phi_1, \phi_2, \dots, \phi_C$. *Private Key:* $A_0, \dots, A_{C-1}$.

Algorithm 1: Pseudo-code of ALTEQ key generation function (Bläser, 2023)

Input: The public key $\phi_0, \dots, \phi_C \in \mathrm{ATF}(n,q)$. The private key $A_0, \dots, A_{C-1} \in \mathrm{GL}(n,q)$. r, C, $\lambda \in \mathbb{N}$. Let $A_C = I$, the identity matrix. The message M. A hash function $H : \{0,1\}^* \to \{0,1\}^{2\lambda}$. An expander $\mathsf{Expand} : \{0,1\}^{2\lambda} \to \{a_i\}_{i \in \{0,\dots,r-1\}}$, where $a_i \in \{0, \dots, C\}$.
Output: The signature S on M.
for $i \in \{0, \dots, r-1\}$ **do**
  Randomly sample $B_i \in \mathrm{GL}(n,q)$.
  $\psi_i \leftarrow \phi_C \circ B_i$.
end
Compute $\mathsf{cha} = H(M|\psi_0|\dots|\psi_{r-1}) \in \{0,1\}^{2\lambda}$.
$(b_0, \dots, b_{r-1}) \leftarrow \mathsf{Expand}(\mathsf{cha})$
for $i \in \{0, \dots, r-1\}$ **do**
  $D_i \leftarrow A_{b_i} B_i$; // Note that $\phi_{b_i} \circ D_i = \psi_i$.
end
return $S = (\mathsf{cha}, D_0, \dots, D_{r-1})$.

Algorithm 2: Pseudo-code of ALTEQ sign generation function (Bläser, 2023)

Input: The public key $\phi_0, \dots, \phi_C \in \mathrm{ATF}(n,q)$. The signature $S = (\mathsf{cha}, D_0, \dots, D_{r-1})$, $b_i \in \{0, \dots, C\}$, $D_i \in \mathrm{GL}(n,q)$. The message M. A hash function $H : \{0,1\}^* \to \{0,1\}^{2\lambda}$. An expander $\mathsf{Expand} : \{0,1\}^{2\lambda} \to \{a_i\}_{i \in \{0,\dots,r-1\}}$, where $a_i \in \{0, \dots, C\}$.
Output: "Yes" if S is a valid signature for M. "No" otherwise.
for $i \in \{0, \dots, r-1\}$ **do**
  Compute $\psi_i' = \phi_{b_i} \circ D_i$.
end
Compute $\mathsf{cha}' = H(M|\psi_0'|\dots|\psi_{r-1}') \in \{0,1\}^{2\lambda}$.
$(b_0', \dots, b_{r-1}') \leftarrow \mathsf{Expand}(\mathsf{cha}')$
if *for every* $i \in \{0, \dots, r-1\}$, $b_i = b_i'$ **then**
  return *Yes*
else
  return *No*

Algorithm 3: Pseudo-code of ALTEQ sign verification function (Bläser, 2023)

3 Profiling Result

To accelerate the performance of an algorithm, it is a common approach to optimize its most computationally intensive operations, that is, the bottleneck. Therefore, to accelerate ALTEQ, we first identified its bottlenecks through profiling. The profiling was conducted on an Intel i7-13700K CPU, where the key generation, signing, and verification functions were each executed 1,000 times using gprof (Fenlason, 1998). The results of the profiling are presented in [Figure 1].

As shown in [Figure 1], the most significant performance load in ALTEQ stems from Keccak-related functions, including KeccakF1600_Permutation. Excluding these, the next highest load is the actingOnATFwColoumn_vec32 function, which internally calls multiplicationModuloP and reductionModuloP repeatedly. This function is structured with nested triple or quadruple for loops, each handling a portion of the ATF calculations. Consequently, its computational load is primarily caused by modular multiplication, modular reduction, and the overhead from these extensive loops. Other functions contribute less than 2% of the total load, indicating no other significant bottlenecks. Therefore, we targeted the most demanding operations, Keccak and modular multiplication/reduction, for hardware module design. The hardware design, detailed in the following section, could dramatically improve the performance of ALTEQ.

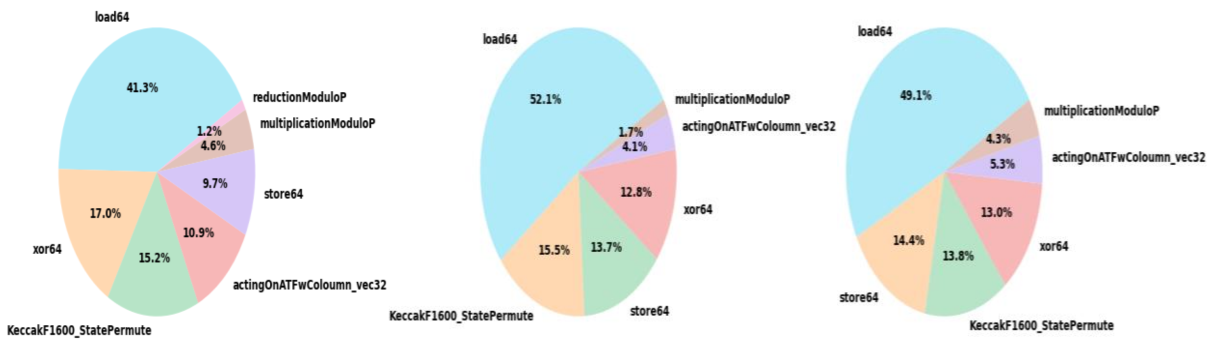

Figure 1: Profiling Result of ALTEQ: each circle graphs denote the result of KeyGen, Sign, and Verify function, left to right.

4 Hardware Design

In this chapter, we propose a hardware design to mitigate the primary bottlenecks of ALTEQ by implementing its most computationally intensive operations Keccak and modular multiplication/reduction in hardware. The overall system assumes a platform such as the Xilinx Zynq SoC, partitioned into a Processing System (PS) based on an ARM processor and a Programmable Logic (PL) area. The system architecture is configured with reference to the design by Mao et al. (Mao, 2023) and utilizes the Advanced eXtensible Interface (AXI) protocol for efficient data transfer between the PS and the PL. The data flow is as follows:

PS to PL Data Transfer: The ARM processor in the PS initializes a Direct Memory Access (DMA) controller to transfer the source data, stored in DDR memory, to the hardware accelerator in the PL. The DMA facilitates high-speed transfer of large data volumes to the PL area without processor intervention.

Data Buffering: Data transferred via DMA is sequentially stored in an AXI-Stream Data FIFO within the PL. This FIFO serves as a buffer to reconcile the speed mismatch between the high-speed data transmission from the PS and the processing speed of the internal modules in the PL.

Accelerator Module Operation: Data from the AXI-Stream Data FIFO is forwarded, in required chunks (e.g., 64-bit), to a dedicated Read FIFO for each accelerator (either the Keccak or the modular arithmetic module). Each module then reads the data from its Read FIFO to perform high-speed parallel computations.

PL to PS Result Return: Upon completion of the computation, the results are passed through each module's Write FIFO, back to the AXI-Stream Data FIFO, and are finally written to the DDR memory in the PS area via DMA. This allows the processor to proceed with subsequent operations.

This dual-FIFO structure was adopted to enhance modularity and reusability. By decoupling the AXI FIFO, which is directly connected to the AXI interface, from the local FIFOs dedicated to each processing module, the impact on the overall system is minimized when adding new types of accelerators or modifying existing ones in the future. The overall system is illustrated at [Figure 2]. The red lines denotes the data, and the blue lines denote the control signals.

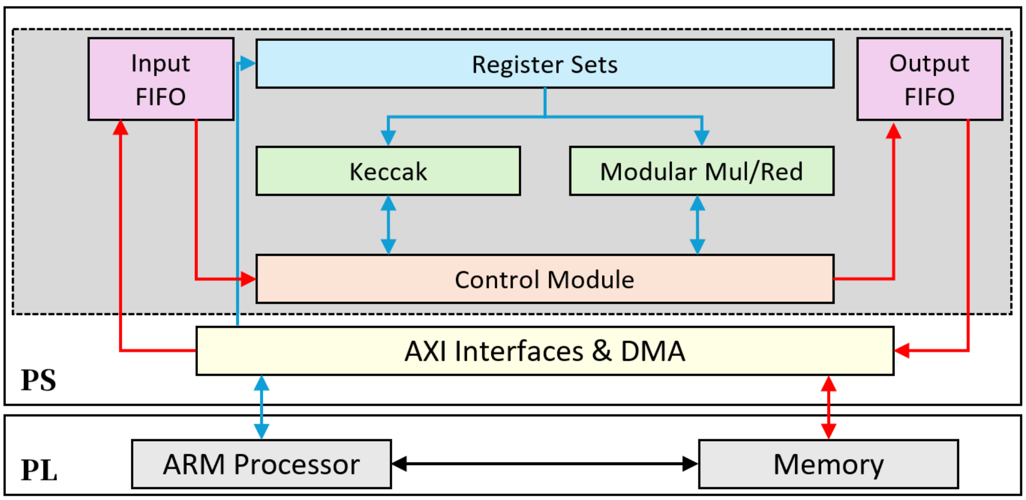

Figure 2: Overview of the entire system of HW/SW co-design architecture

4.1 Keccak Accelerator

The Keccak function performs crucial roles in the ALTEQ algorithm, such as hashing and random number generation, and profiling results have shown it to be the most computationally intensive component. Therefore, designing a high-performance Keccak accelerator is essential for improving overall system performance. The internal architecture of this accelerator is implemented based on the SHAKE module design by Mao et al. The Keccak accelerator is composed of three main parts: a Read Control Unit, a Keccak Core Unit, and a Post-processing Unit. The entire process is designed with a pipelined architecture to maximize data throughput.

Read Control Unit: This unit manages data input and operates via a Finite State Machine (FSM). It reads 64-bit data from an external FIFO. If the Keccak Core is busy and cannot accept new data, the FSM transitions to a hold state to temporarily pause the data flow.

Keccak Core Unit: This is the core of the Keccak computation, based on the Keccak-f1600 Permutation implementation from OpenCores, with an added interface for system integration.

Input Padder: It pads the 64-bit input data to match the block size of the absorbing phase (1088 or 1344 bits).

Permutation: This block consists of the combinatorial logic and registers that perform the Keccak-f1600 permutation. A single permutation is pipelined to complete in 48 clock cycles. Notably, to account for cases where the Post-processing Unit might operate at a slower speed, it includes a feature to pause the permutation state via a hold control signal, enabling flexible synchronization of the entire pipeline.

Post-processing Unit: This unit is responsible for the final processing of the output data from the Keccak Core. Depending on the requirements of the ALTEQ, it either outputs the hash result directly or generates random numbers according to a specific format. This process is also configured in a multi-stage pipeline to keep pace with the Keccak Core's operational speed.

By tightly coupling the Keccak Core and the post-processing stage into a single module, the significant data transfer overhead that would otherwise arise from sending intermediate results to and from the PS is fundamentally eliminated. This is a key design strategy for resolving communication bottleneck, which is a primary factor of performance degradation in a SW/HW co-design.

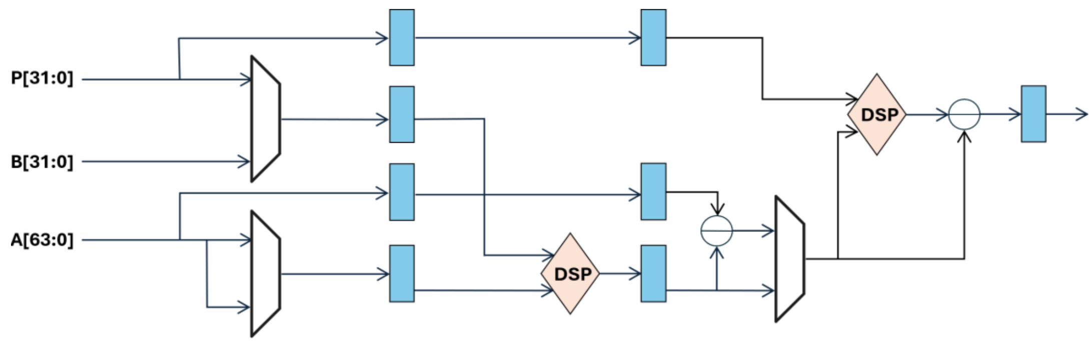

Figure 3: Overview of unified modular multiplication and reduction module. The blue box denotes registers, and the line in the circle denotes the subtraction of two inputs. Input $\boldsymbol{A}$ should be separated into two parts before the first MUX: low 32-bits and high 32-bits according to the control signal.

4.2 Modular Multiplication and Reduction Accelerator

ALTEQ, a non-lattice-based cryptographic algorithm, relies on polynomial arithmetic, which demands numerous modular multiplication and addition/subtraction operations. In this paper, we designed a dedicated hardware module to accelerate these computations. ALTEQ performs its modular arithmetic using Barrett multiplication and reduction. While a straightforward approach would be to design separate modules for these two operations, such a method has the disadvantage of increased area and resource consumption. To mitigate this drawback, we propose the first design that integrates modular multiplication and reduction into a single, unified module. This module preserves the original data flows for both operations while minimizing resource usage, requiring only two DSPs compared to the four that would be needed for separate modules.

A schematic of the modular arithmetic unit is depicted in [Figure 2], illustrating the distinct data flows for multiplication and reduction. Since modular multiplication requires two inputs (A and B) while modular reduction needs only one (A), a control signal configures the data path to bypass input B entirely during reduction operations. Furthermore, the inputs have different bit widths: the multiplication takes 32-bit inputs, whereas the reduction takes 64-bit input. Consequently, the unit's input A is designed to be 64 bits wide to accommodate both operations, while input B, used exclusively for multiplication, is 32 bits wide. The proposed module computes modular multiplication or reduction with 4 clock cycles latency, and it outputs the result for every clock cycle after then while the module is working.

5 Discussions

In this section, we discuss aspects of the proposed HW/SW co-design that warrant further consideration. This discussion is intended to outline potential avenues for advancing the design proposed in this paper.

Necessity of Actual Implementation: This paper has proposed a design for an ALTEQ HW/SW co-design; however, it has not yet been physically implemented. An actual implementation and synthesis process is necessary to accurately measure precise clock cycles and resource utilization. Therefore, additional implementation work is required to verify the performance metrics not presented in this paper.

Optimization of Loop Calls: In ALTEQ, the modular multiplication/reduction loop processes a large data volume of approximately 130,000 bytes in a single iteration. Transmitting such a large amount of data via the AXI protocol can create significant communication overhead and may exceed the internal storage capacity of the hardware module. Consequently, ALTEQ's performance could be further enhanced by segmenting the loop or by exploring other optimization strategies for handling large data transfers more effectively.

6 Conclusion

This paper proposes the first HW/SW co-design architecture for ALTEQ, an algorithm submitted to the NIST Additional Signatures competition. Profiling of the ALTEQ algorithm revealed that much of the execution time is consumed by the Keccak hash function and modular multiplication and reduction operations. Accordingly, our design offloads these two core computational bottlenecks to a dedicated FPGA-based hardware accelerator, while the ARM processor manages the overall algorithm flow and data handling in an efficient, partitioned structure.

To ensure a robust hardware design, the overall system architecture was configured with reference to the verified prior research of Mao et al. To minimize the communication bottleneck between the PS and the PL, we established a high-speed data transfer path using the AXI protocol and DMA. Furthermore, a dual-FIFO structure was adopted to reliably control the data flow and secure system modularity and reusability. The Keccak accelerator maximizes its pipeline structure and tightly couples the data I/O, permutation, and post-processing stages into a single module, thereby fundamentally eliminating communication overhead by removing unnecessary intermediate data exchanges with the PS. Additionally, the modular arithmetic accelerator integrates the multiplication and reduction units, which were previously implemented as separate modules, into a single unified module. This novel structure minimizes hardware resource consumption, such as DSP blocks, by half while maintaining an efficient data flow.

This research holds significant meaning in that it demonstrates that even cryptographic algorithms not selected in the final stages of standardization, such as ALTEQ, can achieve substantial performance improvements through hardware optimization. This suggests that non-standardized algorithms could be viable alternatives depending on specific application environments or system requirements, providing an opportunity for their re-evaluation. Furthermore, the design methodology proposed in this paper can serve as a foundation for promoting future hardware implementation research across a broader range of PQC algorithms.

Certainly, this paper has a limitation in that performance measurements through actual physical implementation and synthesis have not been conducted. Future work will need to involve implementing the proposed architecture on an actual FPGA to measure and analyze concrete performance metrics such as precise clock cycles, resource utilization, and power consumption. Moreover, if communication optimization strategies for more efficiently segmenting or processing the large data loops present in ALTEQ's operational flow are further explored, it is expected that the algorithm's performance can be elevated to the next level.

7 Acknowledgement

This work was supported by the Institute of Information and communications Technology Planning and Evaluation (IITP) Grant by the Korean Government through Ministry of Science and ICT (MSIT) (A study on PQC optimization and security protocol migration to neutralize advanced quantum attacks in Beyond 5G-based next-generation IoT computing environments, 100%) under Grant 2022-00207416.

References

[1] Abdulrahman, A., Hwang, V., Kannwischer, M. J., & Sprenkels, A. (2022). Faster kyber and dilithium on the cortex-m4. *In International Conference on Applied Cryptography and Network Security* (853-871). Cham: Springer International Publishing.

[2] Alagic, G., Bros, M., Ciadoux, P., Cooper, D., Dang, Q., Dang, T., ... & Waller, N. (2024). Status report on the first round of the additional digital signature schemes for the nist post-quantum cryptography standardization process. https://doi.org/10.6028/NIST.IR.8528

[3] Bläser, M., Duong, D. H., Narayanan, A. K., Plantard, T., Qiao, Y., Sipasseuth, A., & Tang, G. (2023). The alteq signature scheme: Algorithm specifications and supporting documentation. *NIST PQC Submission*.

[4] Braun-Dubler, N., Gier, H. P., Bulatnikova, T., Langhart, M., Merki, M., Roth, F., ... & Perdrisat, S. (2020). *Blockchain: Capabilities, economic viability, and the socio-technical environment* (Vol. 73). vdf Hochschulverlag AG.

[5] Cheon, J. H., & Johansson, T. (2017). Post-Quantum Cryptography. *National Institute of Standards and Technology, Tech. rep*.

[6] Fiat, A., & Shamir, A. (1986). How to prove yourself: Practical solutions to identification and signature problems. *In Conference on the theory and application of cryptographic techniques* (186-194). Berlin, Heidelberg: Springer Berlin Heidelberg.

[7] Goldreich, O., Micali, S., & Wigderson, A. (1991). Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM (JACM), 38*(3), 690-728. https://doi.org/10.1145/116825.116852

[8] Greconici, D. (2020). Kyber on risc-v. Master's *Thesis*.

[9] Greconici, D. O., Kannwischer, M. J., & Sprenkels, A. (2021). Compact dilithium implementations on Cortex-M3 and Cortex-M4. *IACR Transactions on Cryptographic Hardware and Embedded Systems,* 1-24. https://doi.org/10.46586/tches.v2021.i1.1-24

[10] Li, L., Tian, Q., Qin, G., Chen, S., & Wang, W. (2024). Compact instruction set extensions for Dilithium. *ACM Transactions on Embedded Computing Systems, 23*(2), 1-21. https://doi.org/10.1145/3643826

[11] Mao, G., Chen, D., Li, G., Dai, W., Sanka, A. I., Koç, Ç. K., & Cheung, R. C. (2023). High-performance and configurable SW/HW co-design of post-quantum signature CRYSTALS-Dilithium. *ACM Transactions on Reconfigurable Technology and Systems, 16*(3), 1-28.

[12] Seiler, G. (2018). Faster AVX2 optimized NTT multiplication for Ring-LWE lattice cryptography. *Cryptology ePrint Archive*.

[13] Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. *In Proceedings 35th annual symposium on foundations of computer science* (124-134). Ieee. https://doi.org/10.1109/SFCS.1994.365700

[14] Singhal, S. P., Gupta, S., & Nuzzo, P. (2019). Profiling minisat based on user defined execution time--GPROF. https://doi.org/10.48550/arXiv.1909.13058

[15] Tani, S. (2007). An improved claw finding algorithm using quantum walk. In *International Symposium on Mathematical Foundations of Computer Science* (536-547). Berlin, Heidelberg: Springer Berlin Heidelberg.

[16] Yu, X., Zhang, J., Huang, J., Chen, D., & Zhou, L. (2024). Multi-way High-Throughput Implementation of Kyber. *In International Conference on Information Security* (41-60). Cham: Springer Nature Switzerland.

Authors Biography

Yongryeol Choi received B.S in Dept. Information Security, Cryptology, and Mathematics of Kookmin University, Seoul, Korea in 2024. Currently, he is pursuing the M.S degree in Cyber Security. His research interests are optimization of cryptographic algorithms, Post-Quantum Cryptography implementation, module validation program, and SW/HW co-design for IoT devices.

Youngbeom Kim received B.S in Dept. Information Security, Cryptology, and Mathematics of Kookmin University, Seoul, Korea in 2021 and M.S degree in Financial Information Security, Kookmin University in 2023. Currently, he is pursuing the Ph.D. degree at Kookmin University. His research interests are optimization of cryptographic algorithms, Post-Quantum Cryptography implementation, module validation program, and SW/HW co-design for IoT devices.

Willy Susilo received the B.S. degree in Computer Science from Universitas Surabaya, Indonesia, in 1993, the M.S. degree in Computer Science from the University of New Brunswick, Canada, in 1996, and the Ph.D. degree in Computer Science from the University of Wollongong, Australia, in 2001. He has been with the School of Computer Science and Software Engineering at the University of Wollongong since 2000, where he is currently a Distinguished Professor. He serves as the Director of the Institute of Cybersecurity and Cryptology (iC2) and was previously the Head of the School of Computing and Information Technology from 2015 to 2020. He is a Fellow of the IEEE, the IET, and the Australian Computer Society (ACS). His research interests include cryptography, cybersecurity, network security, and computer security.

Okyeon Yi received the Ph.D. degree in mathematics from the University of Kentucky, in 1996. From July 1999 to August 2001, he was with the Electronics and Telecommunications Research Institute, Daejeon, South Korea, as a Team Leader of mobile information security. He is currently a Professor with Kookmin University, Seoul, South Korea. His research interests include the 5G, 5G+, 6G mobile telecommunication security, drone security, quantum cryptographic module security, quantum key distribution security, and post quantum cryptography security.

Ilsun You (SM'13) received the M.S. and Ph.D. degrees in computer science from Dankook University, Seoul, South Korea, in 1997 and 2002, respectively, and the Ph.D. degree from Kyushu University, Japan, in 2012. From 1997 to 2004, he was with THIN multimedia, Inc., Internet Security Co., Ltd., and Hanjo Engineering Co., Ltd., as a Research Engineer. He is currently an Associate Professor with the Department of Information Security Engineering, Soonchunhyang University. His main research interests include internet security, authentication, access control, and formal security analysis. He is a fellow of the IET. He has served or is currently serving as a main organizer of international conferences and workshops, such as MobiWorld, MIST, SeCIHD, AsiaARES, IMIS, and so forth. He is the Editor-in-Chief of the Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications. He is on the Editorial Board for Intelligent Automation \& Soft Computing, the Journal of Network and Computer Applications, the

International Journal of Ad Hoc and Ubiquitous Computing, Computing and Informatics, the Journal of High Speed Networks, and Security and Communication Networks.

Seog Chung Seo received the B.S. in Information and Computer Engineering from Ajou University, Suwon, Korea, and the M.S. in Information and Communications from Gwangju Institute of Science and Technology (GIST), Gwangju, Korea, in 2005 and 2007, respectively. He received the Ph.D. degree at Korea University, Seoul, Korea in 2011. He worked as a research staff member at SAIT (Samsung Advanced Institute of Technology) and Samsung DMC Research and Development Center, from Sep. 2011 to April 2014. He was a senior research member of the Affiliated Institute of ETRI in South Korea from 2014 to 2018. Currently, he works as an associate professor at Kookmin University in South Korea. His research interests include public-key cryptography, its efficient implementations on various IT devices, cryptographic module validation program, network security, and data authentication algorithms.