An SRAM Compute-in-Memory based NTT Accelerator for CRYSTALS-KYBER

Jinyang Hu[1], Xinyuan Pang[1], Dong Jiang[1], Gaopeng Fan[1], Enyi Yao[1]
[1]School of Microelectronics, South China University of Technology, Guangzhou 511442, China
Corresponding author: Enyi Yao (Email: yaoenyi@scut.edu.cn)

2025 IEEE International Symposium on Circuits and Systems (ISCAS) | 979-8-3503-5683-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/ISCAS56072.2025.11043803

***Abstract*—Among various post-quantum cryptography (PQC) proposed by researchers, lattice-based cryptography is considered to be one of the most promising post-quantum public key cryptography systems. It has significant advantages over other PQC schemes in terms of security, computational efficiency, and versatility in designing public key encryption, digital signatures, and cryptographic negotiation protocols. Efficient implementations of the number theoretic transform (NTT) operations are crucial for many lattice-based encryption algorithms. This article presents an NTT hardware acceleration structure based on SRAM in-memory computing technology, which optimizes the key butterfly structure in the NTT algorithm by improving the 6T-SRAM array structure and incorporating near-memory computing structures. Compared with existing technologies, this accelerator can significantly reduce area and power consumption.**

***Index Terms*—Number Theoretic Transform, Compute-In-Memory, Post-Quantum Cryptography, Lattice-Based Cryptography, CRYSTALS-KYBER**

I. Introduction

The security of current public key cryptosystems are threatened as the computing power of adversaries increases. The emergence of Shor's algorithm in 1994 indicates that adversaries can potentially break public key cryptographic schemes based on difficult problems such as large composite integer factorization and discrete logarithm in polynomial time on quantum computers [1]. To mitigate the threat posed by quantum computers to existing cryptographic algorithms and ensure the security of information, different post-quantum cryptographic algorithms are developed during the past decade.

Among the recognized post-quantum cryptographic schemes, lattice-based cryptography exhibits significant advantages in terms of security and key length. In the fourth round of the NIST Post-Quantum Cryptography Standardization process announced in 2023, the CRYSTALS-KYBER algorithm was successfully selected as a lattice-based key encapsulation mechanism. Most of the polynomial multiplication computations in lattice-based cryptography are performed over polynomial rings, and the number theoretic transform operations are usually used to improve the computational speed of polynomial multiplication ,reducing the computational complexity from $O(n^2)$ to $O(nlogn)$. Consequently, NTT is widely adopted in many lattice-based cryptographic systems. Previous works have been conducted to accelerate polynomial multiplication using NTT operations in post-quantum cryptographic systems, including implementations on FPGA [2] [3] [4] [5], ASIC [6] [7], and software platforms [8] [9] [10]. However, none of the above designs can properly address the performance bottlenecks that may be encountered in application scenarios that deal with large-scale data or high-precision computation. In large-scale NTT computing scenarios that are control-flow driven with frequent reads and writes to memory, memory walls and power walls are also encountered as the number of NTT points increases. By introducing the compute-in-memory (CIM) technique into NTT, it is possible to effectively reduce data transfer time and overhead, thereby improving computational performance and efficiency. In [11], an NTT accelerator implemented using 6T-SRAM was proposed. Each column in the array is capable of performing butterfly operations. However, since the structure only supports bit-serial operations, this would result in a significant delay. The work mentioned in [12] implemented GS butterfly structures and reduction algorithms using RRAM array. The frequent data movement of intermediate results within the array may potentially lead to durability issues in RRAM devices.

In this paper, we propose an NTT hardware accelerator in combination with SRAM-based CIM techniques for the acceleration of the NTT of the CRYSTALS-KYBER algorithm. Our design leverages an SRAM array and a set of near-memory compute circuits to perform the butterfly structures in NTT. Additionally, a modular reduction method is also implemented with the CIM [13], enabling the preloading of the weights that require modulo operations. Compared to the previous CIM based NTT design, our proposed array layout achieves higher parallelism, with reduced energy consumption.

II. Preliminaries

NTT plays a crucial role in encryption algorithms, particularly in those based on the Learning With Errors (LWE) problem. It is commonly used to accelerate polynomial operations. Similar to the FFT in the complex number field, NTT is a discrete Fourier transform over the ring of integers modulo q. An n-point NTT operation is mathematically expressed as (1):

$$\bar{a}_i = \sum_{j=0}^{n-1} a_j \omega^{i \times j} \ (mod\ q)\ ,\ i = 0, 1, \cdots, n-1 \qquad (1)$$

This work is supported by the GJYC program of Guangzhou (Grant No. 2024D03J0006).

Algorithm 1 Radix-2 NTT operation with CT structure

Require: $a(x) \in \mathbb{Z}_q[x]/(x^n+1)$, $\omega_n \in \mathbb{Z}_q$, $n = 2^l$, q
Ensure: $\bar{a}(x) \in \mathbb{Z}_q[x]/(x^n+1)$

$p \leftarrow 1$
for $i = 1$ to $l+1$ **do**
  $m \leftarrow 2^{l-i}$
  for $j = 0$ to 2^{i-1} **do**
    $base \leftarrow j * 2^{l-i+1}$
    for $k = 0$ to m **do**
      $A, B \leftarrow a[k+base], a[k+base+m]$
      $W \leftarrow \omega^{br_{l-1}(p)} mod\, q$
      $c \leftarrow B * W$
      $S \leftarrow c\ mod\ q$
      $E, O \leftarrow (A+S) mod\, q, (A-S) mod\, q$
      $a[k+base], a[k+base+m] \leftarrow E, O$
    end for
    $p \leftarrow p+1$
  end for
end for
$\bar{a}(x) \leftarrow BitReverse(a(x))$
return $\bar{a}(x)$

Here, q is a proper modulus and satisfies the form $q = c * N + 1$ ($N = 2^l$), g is the corresponding primitive root of q, a_j represents the coefficient of an n-degree polynomial $a(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}$ in the ring of integers modulo q. The term $\omega = g^{\frac{q-1}{n}}$, as a rotation factor in the NTT, refers to the primitive n-th root of unity modulo q.

An n-point Inverse Number Theoretic Transform (INTT) is mathematically expressed as (2):

$$a_j = \frac{1}{n}\sum_{i=0}^{n-1} \bar{a}_i \omega^{-i\times j}\ (mod\ q) \quad , i = 0, 1, \cdots, n-1 \qquad (2)$$

In hardware implementations, the Cooley-Tukey (CT) and Gentleman-Sande (GS) structures are commonly used to perform NTT and INTT operations. The CT structure corresponds to time-domain extraction, while the GS structure is frequency-domain extraction. Operations for NTT using the CT structure are shown in Algorithm 1.

CRYSTALS-KYBER is a PKE/KEM scheme based on module learning with errors (MLWE). It consists of three operations: key generation, encryption, and decryption, each of which involves polynomial multiplication [10]. The polynomial multiplication in CRYSTALS-KYBER is achieved through NTT, with a modulus q chosen as 3329 and a polynomial dimension of 256. The public key matrix $\mathbf{A}$ used in KYBER is a square matrix, and the security strength can be configured by choosing the number of rows (or columns) k (k=2,3,4) of the public key matrix.

III. The Proposed Design

A. Overall Structure

The overall architecture, as shown in Fig. 1, consists of a controller, a CIM butterfly unit (CBU), a mod3329 reduction unit, and an interunit router which can be utilized for performing NTT and INTT operations in CRYSTALS-KYBER PQC scheme. The CBU serves a dual purpose in the system, functioning both as a storage for the data involved in each butterfly operation and the processing unit for that as well. To adapt to different application scenarios, three NTT architectures, high-performance (HP), balanced (BL) and lightweight (LW) are proposed, corresponding to 128, 16 and 1 CBUs in the architecture, respectively. The decoder and controller are used to control the read and write-back of data in the CBU during butterfly operation. The Mod 3329 Reduction Unit and CBU collaborate to perform 12 bit modular operation.

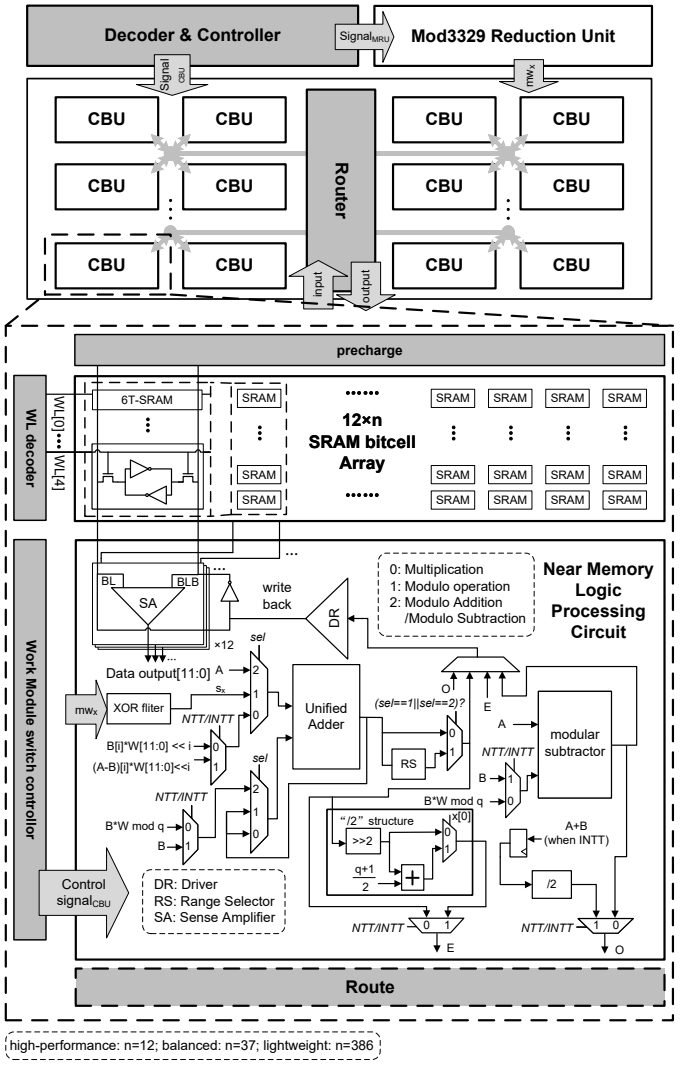

Fig. 1. Overall structure and CIM Butterfly Unit (CBU) detailed structure

B. CIM Butterfly Unit

The proposed ClM butterfly unit is shown in Fig. 1, providing the configuration of SRAM array and critical functional components. The size of SRAM array is configured according to the type of NTT architecture. The sense amplifier (SA) is utilized for data readout in the SRAM array, enabling simultaneous parallel retrieval of 12 bit operands for compu-

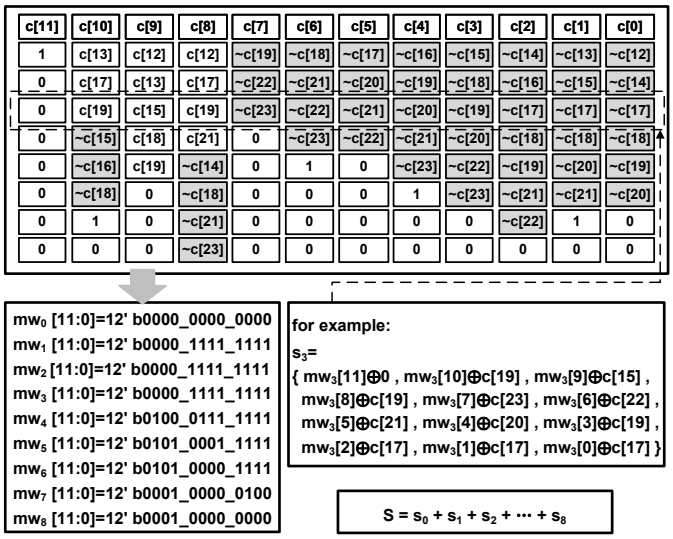

Fig. 2. Recursive result and weight illustration

tation. The NTT operation requires the use of the CT butterfly structure to perform $A + B\omega \ (mod\ q)$ and $A - B\omega \ (mod\ q)$, while the INTT operation needs the GS butterfly structure to perform $(A + B) \ (mod\ q)$ and $(A - B)\omega \ (mod\ q)$. To simplify hardware design, share hardware modules and logic circuits and improve hardware resource utilization, the proposed design integrates CT and GS butterfly structures within the CBU, with the controller to determine the operating mode of the CBU. When the selection signal "NTT/INTT" is set to 0, the CBU operates in CT butterfly mode for NTT and vice versa in GS mode for INTT.

Due to the multiple steps involved in butterfly operation with adders, the hardware functionality for addition is proposed to be unified into a unified adder (UA), aiming to enhance hardware resource utilization efficiently. The inputs and outputs of the unified adder are 24 bit data. Due to the constraints on the input data for the NTT/INTT operation in the CRYSTALS-KYBER scheme, the output width will not exceed 24 bits. When sel = 0, the unified adder functions as an accumulator for performing 12 bit by 12 bit operations, accumulating the shifted 1 bit by 12 bit data in each cycle. When sel = 1, the unified adder is utilized as an accumulator for performing modulo 3329 operations. It accumulates the intermediate results of the Mod 3329 reduction unit in each cycle, and then, through a range selector (RS), derives the modulo-reduced result (this step will be detailed in section C). When sel = 2, the unified adder works as a modular adder, performing modular addition in conjunction with RS.

In conventional INTT operations, the final step involves multiplying the computed result by 1/256. However, performing this multiplication directly in hardware can be highly resource-intensive. The introduction of the "/2" structure in the GS structure will optimize this situation [2]. Computing modulo division by 2 on the result at the end of each stage of the butterfly operation, with a total of 8 stages, will ultimately be equivalent to the last step of the 1/256 (mod q) multiplication operation. This can be seen as a decomposition of this complex operation. For a given odd prime number q, when x is an even number, $x/2$ mod q is equivalent to right-shifting 1 bit. When x is an odd number, $x/2$ can be expressed as follows:

$$\frac{x}{2}(mod\ q) = (x >> 1) + (\frac{q+1}{2})$$

This function can be implemented in hardware using shifters, adders, and multiplexers.

C. Mod3329 reduction unit

In the CRYSTALS-KYBER, the modulus q for the NTT operation is 3329. Therefore, a dedicated high-width modular reduction unit (MRU) is proposed around the modulo 3329 operation [14]. The MRU is specifically designed to handle the 24 bit output data from the multiplication of 12 bit by 12 bit operands by the multiplier, recursively employing the property expressed as $2^{12} \equiv 2^9 + 2^8 - 1 \pmod{3329}$, as demonstrated below:

$$\begin{aligned}
c[23:0] &= 2^{12}c[23:12] + c[11:0] \\
&= 2^9c[23:12] + 2^8c[23:12] - c[23:12] + c[11:0] \\
&= 2^{12}c[23:15] + 2^9c[14:12] + 2^{12}c[23:16] \\
&\quad + 2^8c[15:12] - c[23:12] + c[11:0] \\
&= (2^9 + 2^8 - 1)(c[23:15] + c[23:16]) \\
&\quad + 2^9c[14:12] + 2^8c[15:12] - c[23:12] \\
&\quad + c[11:0] \\
&= 2^9c[23:15] + 2^8c[23:15] - c[23:15] \\
&\quad + 2^9c[23:16] + 2^8c[23:16] - c[23:16] \\
&\quad + 2^9c[14:12] + 2^8c[15:12] - c[23:12] \\
&\quad + c[11:0]
\end{aligned}$$

This approach allows for the decomposition of 24-bit operands into multiple numbers with a bit width of 12 bits or less. Some steps in the process utilize the equation: $2^x c[y] + 2^x c[y] = 2^{x+1}c[y]$ to eliminate redundant operations. The final recursive result of the 24-bit operand c[23:0] is shown in Figure 3. The Mod 3329 reduction unit reads the weight mw_x row by row and performs an XOR operation with the corresponding bits of the operand in each cycle. The resulting operation results s_x from each cycle are accumulated to obtain S. Since the weight array is set with nine rows, our modular reduction operation requires nine cycles to complete. The aforementioned accumulated result falls within the range of 9271 to -3264. To adjust the result to the target range of [0, 3329), an additional range selector (RA) was added after the Unified Adder in the CBU. Within the RA, $(S+q), (S), (S-q)$ and $(S-2q)$ are calculated respectively, and the result within the range [0, q) is selected for output.

D. Workflow

In the HP-CBU, the first and second rows of the SRAM array are preloaded with A and B (step 7 of Algorithm 1). The

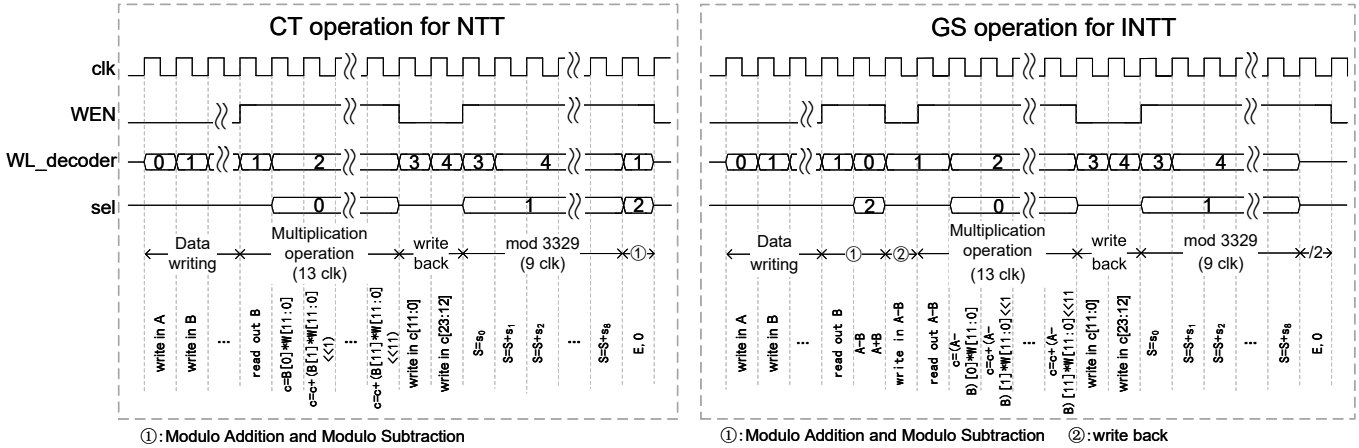

Fig. 3. Workflow of NTT/INTT operation

third through tenth rows are used to store $W_0 \sim W_7$ required for stages 1∼8 for butterfly operations (step 8 of Algorithm 1). Rows 10 and 11 are used to store the intermediate results generated during the computation. In BL-CBU rows 1 to 8 and 9 to 16 are preloaded with the 8 sets of operands A and B needed for the current stage, respectively. The remaining addresses of the array are used to store the weights W and intermediate results.

From Fig. 2, it can be observed that during the mod 3329 operation, only the first cycle requires the use of c[11:0], while cycles two to nine utilize c[23:12]. This means that the data from the lower 12 bits and the higher 12 bits of the operand c are not read out simultaneously. Therefore, we write c[11:0] and c[23:12] to the two intermediate result addresses respectively.

The 12 bit by 12 bit multiplication operation (step 9 in Algorithm 1) is completed in 12 cycles, as each cycle handles a 1 bit by 12 bit multiplication operation followed by accumulation. The modulo operation for 24-bit operands requires nine cycles (step 10 in Algorithm 1). The complete flow of CT and GS operation is illustrated in Fig. 3. Due to the additional "/2" step in GS operation and the requirement to compute and write back one of the operands for the multiplication operation using CBU, the overall completion of GS operation consumes three more cycles compared to CT operation.

IV. Results

The proposed hardware architecture is implemented using a standard 40 nm CMOS technology. The digital part of the design is implemented using Verilog and synthesized using Synopsys Design Compiler to evaluate the overall performance. According to the synthesis results, approximately 73.64% of the energy is consumed by the near-memory circuitry, while the energy consumption of SRAM accounts for 26.36%. The evaluation results of the proposed design compared to existing NTT accelerators are presented in Table I. Our design exhibits notable advantages compared to the works shown in Table I. It achieves a lower power consumption and faster processing speed compared to previous works. Our design demonstrates a satisfactory throughput and operating frequency. Additionally, the adoption of bit-parallel design significantly enhances throughput and computational speed, resulting in a 94.696% reduction in computation time for a single NTT operation in HP-NTT.

TABLE I
Comparison Table of This Work and Previous Designs

	HP/BL/LW	[11]	[14]	[15]	[12]	[16]	[13]
Method	SRAM	SRAM	serial butterfly	parallel butterfly	RRAM	RRAM	FPGA (Artix-7)
Tech.	40nm	65nm	40nm	40nm	45nm	28nm	28nm
Freq.[a]	200	151	72	300	909	400	172
Bitwidth	12	14	13	13	16	14	12
Order	256	256	256	256	256	256	256
Latency[b]	1.22/5.28/ 138.24	23	17.9	0.533	68.7	0.28	0.4002
Tp.[c]	820k/189k/ 7.23k	42k	56k	1.8M	553k	3.57M	2.5M
Energy[d]	32.3904/ 31.4981/-	144	166	31	2580	145	-

[a](MHz); [b](us); [c]Throughput ;[d](nJ/NTT) .

V. Conclusion

A hardware accelerator based on SRAM CIM technique is proposed in this paper to accelerate the NTT and INTT operations in the CRYSTALS-KYBER PQC scheme. Compared to previous NTT hardware accelerators, our design incorporates the storage and computation units together using the CIM technique, thereby mitigating the issue of frequent data transfers between the computation and storage units and effectively reducing energy consumption. In comparison to the latest SRAM CIM works, our design achieves higher computational efficiency due to the adoption of a parallel computing structure, enabling a single CBU to process 12 bit data in a single cycle.

REFERENCES

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, pp. 303–332, 1999.

[2] N. Zhang, B. Yang, C. Chen, S. Yin, S. Wei, and L. Liu, "Highly efficient architecture of NewHope-NIST on FPGA using low-complexity NTT/INTT," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 49–72, 2020.

[3] Y. Doröz, E. Öztürk, E. Savaş, and B. Sunar, "Accelerating LTV based homomorphic encryption in reconfigurable hardware," in *International Workshop on Cryptographic Hardware and Embedded Systems*, 2015, pp. 185–204.

[4] S. Khan, A. Khalid, C. Rafferty, Y. A. Shah, M. O'Neill, W.-K. Lee, and S. O. Hwang, "Efficient, error-resistant NTT architectures for CRYSTALS-kyber FPGA accelerators," in *IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, 2023, pp. 1–6.

[5] 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*, vol. 2021, no. 2, pp. 328–356, Feb. 2021.

[6] N. Zhang, Q. Qin, H. Yuan, C. Zhou, S. Yin, S. Wei, and L. Liu, "NTTU: An area-efficient low-power NTT-uncoupled architecture for NTT-based multiplication," *IEEE Transactions on Computers*, vol. 69, pp. 520–533, Apr. 2020.

[7] T. Fritzmann and M. J. Sepúlveda, "Efficient and flexible low-power NTT for lattice-based cryptography," *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 141–150, 2019.

[8] G. Seiler, "Faster AVX2 optimized ntt multiplication for Ring-LWE lattice cryptography," *IACR Cryptology ePrint Archive*, p. 39, 2018.

[9] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Dilithium: A lattice-based digital signature scheme," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 1, pp. 238–268, Feb. 2018.

[10] J. W. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, and D. Stehlé, "CRYSTALS-Kyber: A CCA-Secure module-lattice-based KEM," *IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 353–367, 2017.

[11] D. Li, A. Pakala, and K. Yang, "MeNTT: A compact and efficient processing-in-memory number theoretic transform (NTT) accelerator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 5, pp. 579–588, 2022.

[12] H. Nejatollahi, S. Gupta, M. Imani, T. S. Rosing, R. Cammarota, and N. Dutt, "CryptoPIM: In-memory acceleration for lattice-based cryptographic hardware," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[13] F. Yaman, A. C. Mert, E. Öztürk, and E. Savaş, "A hardware accelerator for polynomial multiplication operation of CRYSTALS-KYBER PQC scheme," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1020–1025, 2021.

[14] U. Banerjee, A. Pathak, and A. P. Chandrakasan, "2.3 an energy-efficient configurable lattice cryptography processor for the quantum-secure Internet of things," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2019, pp. 46–48.

[15] S. Song, W. Tang, T. Chen, and Z. Zhang, "LEIA: A 2.05mm^2 140mW lattice encryption instruction accelerator in 40nm CMOS," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2018, pp. 1–4.

[16] Y. Park, Z. Wang, S. Yoo, and W. D. Lu, "RM-NTT: An RRAM-based compute-in-memory number theoretic transform accelerator," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 8, no. 2, pp. 93–101, 2022.