MT-TMVP: Modular Tiled TMVP-based Polynomial Multiplication for Post-Quantum Cryptography on FPGAs
(Expanded Version)

Shekoufeh Neisarian[1] and Elif Bilge Kavun[1,2]

[1]*Barkhausen Institut, Dresden, Germany*
[2]*TU Dresden, Germany*
shekoufeh.neisarian@barkhauseninstitut.org,
elif.kavun@barkhauseninstitut.org,elif_bilge.kavun@tu-dresden.de

Abstract. As quantum technology advances, developing cryptographic solutions resistant to quantum attacks is crucial. Post-Quantum Cryptography (PQC) provides a practical approach by running on classical computers. They rely on hard mathematical problems, with lattice-based being one of the National Institute of Standards and Technology (NIST)-recognized schemes known for its small key sizes. Hardware implementation of these schemes faces challenges due to the computational intensity of operations like polynomial multiplication, especially for resource-constrained devices. This paper proposes a novel Modular Tiled Toeplitz Matrix-Vector Polynomial Multiplication (MT-TMVP) for lattice-based PQC algorithms and presents a resource-optimized Field Programmable Gate Array (FPGA) architecture. The proposed implementation significantly reduces resource utilization and Area-Delay Product (ADP) compared to state-of-the-art polynomial multipliers. It utilizes 99.68% and 84.22% fewer Look-Up Tables (LUTs) on Artix-7 and Zynq Ultrascale+ FPGAs, respectively, and achieves 99.94% and 80.02% ADP improvements on these FPGAs compared to the best results in the literature. By leveraging Block RAM (BRAM), the proposed architecture offers robustness against timing-based Side-Channel Attacks (SCAs), and the design is modular and scalable to any polynomial degree.

Keywords: Polynomial multiplication · TMVP · Tile · KEM · Lattice-based Post-Quantum Cryptography (PQC) · Resource-constrained devices · FPGA · Timing Side-Channel Attack (SCA).

1 Introduction

Cryptography is essential to ensure the confidentiality and integrity of information and guarantee the security of sensitive data during communication [27]. Quantum computing is a threat to the security of classical public-key cryptography, such as Rivest–Shamir–Adleman (RSA) and Elliptic Curve Cryptography (ECC). Shor's algorithm [33] can potentially break these encryption methods,

which makes it crucial to develop quantum-resistant algorithms. To mitigate quantum computing threats, two strategies have emerged. The first leverages quantum mechanics to create theoretically unbreakable encryption [10], but it requires costly new infrastructure. The second, Post-Quantum Cryptography (PQC), develops cryptographic algorithms that run on classical computers, without requiring major infrastructure changes [23]. Several PQC methods exist based on the hardness of mathematical problems, including lattices [20], coding theory [4], multivariate quadratic equations [35], hash functions [8], and isogenies [21]. The National Institute of Standards and Technology (NIST) has an important role in standardizing PQC public-key encryption algorithms [26].

Among these approaches, lattice-based stands out for its smaller key sizes, which makes it suitable for practical implementation, specifically for deployment on resource-constrained devices such as Internet of Things (IoT) and embedded systems. NTRU [13] is a candidate in the NIST PQC competition [6], upon which several research efforts have been built. In the NTRUPrime project, modifications have been recommended to enhance NTRU's security [30]. FALCON, a NIST-standardized digital signature, is based on the hardness assumption of NTRU [5]. Additionally, a public-key KEM, NTRU+PKE, was introduced by Kim *et al.* [16], leveraging the NTRU structure. Furthermore, NTRU+ is a KEM that is standardized in the Korean PQC (KPQC) standardization process. In addition, Saber is a lattice-based candidate based on the hardness of Module Learning-With-Error (MLWR) and is IND-CCA secure [7]. These two schemes are the focus of this paper due to their polynomial ring structure.

Lattice-based PQC primitives depend on polynomial multiplication, a computationally intensive operation, especially on resource-constrained devices. Optimizing this operation is crucial for efficient implementations [37]. Among various multiplication methods like Schoolbook [31], Karatsuba [1], Toom-Cook [14], Number Theoretic Transform (NTT) [38], and Toeplitz Matrix-Vector Product (TMVP) [9], we focus on TMVP-based approach as it allows polynomial reduction within the multiplication step [28], unlike Toom-Cook and Karatsuba, which require an additional reduction phase [36]. NTT is limited in direct applicability for NTRU and Saber, which use non-NTT-friendly polynomial rings and specific modulus q requirements [28]. Recent studies have implemented TMVP for PQC on FPGAs [12,34] and explored its use in software [29,28]. However, to the best of our knowledge, no resource-optimized FPGA implementation of TMVP-based multiplier targeting resource-constrained devices exists. To address this gap, this paper proposes and implements a novel Modular Tiled TMVP-based (MT-TMVP) polynomial multiplier focusing on resource utilization and Area-Delay Product (ADP) efficiency over state-of-the-art FPGA implementations.

Contributions. The key contributions of this paper are outlined as follows:

- A novel MT-TMVP polynomial multiplication algorithm is proposed, which uses a TMVP-based approach with a tiling technique optimized for lattice-based PQC schemes, which reduces the computational complexity of this fundamental operation in cryptographic protocols.

- The MT-TMVP design is implemented in a modular and parametric architecture, such that adjustments to matrix dimensions and tile sizes in the top-level module are systematically reflected across all modules. Therefore, this design allows flexible dimension scaling.
- An architecture for the proposed MT-TMVP algorithm is designed and implemented on FPGA platforms recommended by NIST in its call-for-proposals. Comprehensive evaluations confirm its practicality, for real-world PQC applications, particularly on resource-constrained devices such as IoT systems and embedded platforms. The design also uses Block RAMs (BRAMs) to ensure security against timing Side-Channel Attacks (SCAs).
- Optimization techniques used to achieve minimal resource utilization and the best ADP compared to state-of-the-art hardware implementations for polynomial multiplication.
- The implementation of the proposed MT-TMVP algorithm is available upon request via a GitHub repository link.

Paper Structure. Section 2 reviews multiplier implementations in PQC. Section 3 covers TMVP and lattice-based PQC. Section 4 details the proposed MT-TMVP algorithm and FPGA implementation. Results and comparison are presented in Section 5, followed by conclusion and future work in Section 6.

2 Related Work

This section reviews existing studies on polynomial multiplication implementations in lattice-based PQC. Given the limitations of other multiplier approaches (Section 1), the focus is on TMVP-based designs, along with a brief review of other multiplier techniques. Software-based TMVP implementations have been studied to enhance performance on ARM Cortex-M4 [29,28] and GPUs [11]. Some studies have explored hardware/software co-design to accelerate TMVP-based polynomial multiplication [22]. This work focuses on hardware implementation targeting resource-constrained devices. Tu *et al.* [34] implemented a Lookup Table (LUT)-based pointwise TMVP-based multiplication accelerator for the NTRU-based FALCON PQC scheme on FPGAs, aiming to enhance multiplication speed; however, it consumes significant resources. Cui *et al.* [7] implemented a TMVP-based multiplier for Saber on FPGA, focusing on throughput optimization. They redesigned the accumulation phase of Schoolbook by leveraging the properties of the Toeplitz matrix. He *et al.* [12] implemented TMVP for Saber to achieve high performance. Allam *et al.* [1] implemented NTT for NTRU on FPGA, using a newly defined NTT-compatible prime $Q > n \cdot q^2$, where n is the polynomial degree and q the modulus. This approach is limited by specific requirements for q. Qin *et al.* [31] implemented Schoolbook in hardware for NTRU, Dang *et al.* [9] explored a Toom-Cook multiplier for NTRU, and Tu *et al.* [34] used Schoolbook for FALCON. Additionally, Li *et al.* [19] implemented Schoolbook for Saber in hardware. However, these implementations do not prioritize resource optimization for hardware and are not well-suited

for resource-constrained devices. While previous works offer valuable insights into software and hardware multiplier implementations, they still consume significant area. Building on these studies, this paper proposes a new MT-TMVP polynomial multiplication algorithm that reduces computational complexity and is suitable for resource-constrained devices. By leveraging the tile property with TMVP-2 [12,7,34], this approach aims to achieve minimum resource utilization and the best ADP compared to state-of-the-art multiplier implementations. To the best of the authors' knowledge, this is the first resource-optimized TMVP-based multiplier for lattice-based PQC that applies tiling.

3 Preliminaries

This section provides the background on TMVP multiplication and lattice-based PQC schemes needed to understand the proposed algorithm.

3.1 Toeplitz Matrix-Vector Product (TMVP)

The Toeplitz matrix-vector product, abbreviated as TMVP, is used in cryptography as a polynomial multiplier technique. The multiplication is conducted in the ring $R = \mathbb{Z}[x]/\langle x^n + 1\rangle$. When the polynomial coefficients are in the range $[0, q)$, the ring is represented as $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$, where n represents the polynomial degree and q is the modulus. A Toeplitz matrix, denoted as A, is structured as an $m \times n$ matrix where the element located at the junction of the i-th row and j-th column is described by the equation $A_{i,j} = A_{i-1,j-1}$, for $i = 2, \ldots, m$ and $j = 2, \ldots, n$. Let A be a square ($m = n$) Toeplitz matrix and B be a vector of dimension $n \times 1$ represented in Eq. (1). The multiplication of A and B results in a vector W of dimension $n \times 1$, given by Eq. (2).

$$A = \begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_n & a_0 & \cdots & a_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{2n-2} & a_{2n-3} & \cdots & a_0 \end{bmatrix}, B = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{bmatrix} \tag{1}$$

$$W = [w_0, w_1, \ldots, w_{n-1}]^\top = A \times B \tag{2}$$

In TMVP-2, the matrix A is segmented into four submatrices, each of size $\frac{n}{2} \times \frac{n}{2}$. The MT-TMVP uses TMVP-2 in combination with tiling to reduce computational complexity and optimize resource utilization. The TMVP-2 formulas are presented in Eq. (3)-Eq. (4). As can be seen, TMVP-2 requires three multiplications. If the dimension of matrix A is large, traditional Schoolbook multiplication becomes inefficient. In such cases, it is more practical to use TMVP [28].

$$\begin{bmatrix} W_0 \\ W_1 \end{bmatrix} = \begin{bmatrix} A_0 & A_2 \\ A_1 & A_0 \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \end{bmatrix} = \begin{bmatrix} s_1 + s_2 \\ s_1 + s_3 \end{bmatrix} \tag{3}$$

$$s_1 = A_0(B_0 + B_1), \quad s_2 = (A_2 - A_0)B_1, \quad s_3 = (A_1 - A_0)B_0 \tag{4}$$

3.2 Lattice-based Post-Quantum Cryptography (PQC)

Traditional public-key encryption, such as RSA [32] and ECC [17], which depend on integer factorization and discrete logarithms, are vulnerable to quantum attacks [33]. To address this challenge, PQC is being developed, which uses mathematically hard problems to secure data against quantum threats. Lattice-based PQC is well-known for its small key sizes. They rely on solving hard mathematical problems in grids of points. At the core of lattice-based PQC is the Shortest Vector Problem (SVP), which involves finding the shortest vector in a lattice that is nearest to the origin yet distinct from it [20]. Several lattice-based schemes are presented in the following, each of which relies on polynomial multiplication.

NTRU, developed by Hoffstein *et al.* [13], integrates concepts from NTRU-Encrypt and NTRU-HRSS-KEM. Its parameters n, p, and q dictate polynomial operations, where n sets the polynomial length, and p and q are used for modulo calculations in encryption. Optimal parameter choices enhance both efficiency and security [29]. Another NIST finalist, FALCON, offers a compact and efficient lattice-based signature scheme based on the Short Integer Solution (SIS) problem in NTRU lattices, achieving a strong balance of security and performance [25]. Polynomial multiplication is critical in FALCON, performed over the ring $\mathbb{Z}_q/(x^n+1)$, which distinguishes it from NTRU's use of $\mathbb{Z}_q/(x^n-1)$. Saber, also a finalist, is based on the Module Learning With Rounding (MLWR) problem and includes three variants corresponding to NIST security levels [24]. Polynomial multiplication is vital in Saber, yet the scheme avoids NTT due to its reliance on power-of-2 moduli, which sets a restriction on its modulus. Saber employs alternative techniques as well such as Karatsuba, Toom-Cook, and TMVP [12].

4 Implementation

This section presents the proposed MT-TMVP polynomial multiplication algorithm and provides details of the resource-optimized hardware architecture, which targets lattice-based PQC on FPGAs and is suitable for resource-constrained devices. The algorithm calculates the product of a Toeplitz matrix and a vector across three dimensions, supporting both encryption and decryption phases in PQC. The design is modular and flexible, with parameter adjustments made through the top module. Section 5 evaluates the effectiveness of MT-TMVP for PQC, including benchmarks and comparisons that demonstrate its suitability for embedded applications. The core design concept is to utilize tiled matrix multiplication. By leveraging the symmetry of the Toeplitz matrix, the product is computed with fewer multiplication operations than standard methods. The tile sizes are flexible, with a typical approach dividing a Toeplitz matrix into four submatrices, each with dimensions half the size of the original matrix. With this tile size, calculations are performed more efficiently using the proposed method. As shown in Eq. (3), the calculation requires 3 multiplications instead of 4, resulting in a 25% reduction in the number of operations. While this approach decreases the number of multiplications, it uses BRAMs to store the results of each multiplication. This setup enables multiplication through a

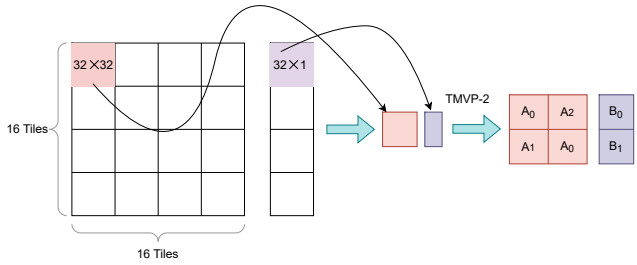

Fig. 1: MT-TMVP for N=512

straightforward method that requires no precalculation, though it results in a larger total number of multiplications.

4.1 Algorithmic Steps for Hardware Implementation

The proposed MT-TMVP is represented in Algorithm 1. The goal is to implement the multiplication of four square matrices and vectors with dimensions of 509, 677, 701, and 821 (using NTRU parameter sets as a case study). The modulus q is set to 256, which defines the size of the matrix elements as 8-bit integer values. The multiplication implementation does not require explicit modulo reduction since the power-of-two structure can be exploited using bitwise operations. The aforementioned matrices will be zero-padded to new dimensions of 512, 720, 720, and 864, making them divisible by 16. The Toeplitz matrix and vector are generated from the first row and first column of the matrix in MATLAB. To perform the multiplication, the tiling and TMVP are combined to take advantage of both approaches. The final multiplication step follows Eq. (3). To prepare for this step, each matrix is divided into submatrices of the size required for the final calculation. For instance, when $N = 512$, the dimension for the last step is 32, which is double the tile size of 16 used in the classic matrix-vector product method. This results in 16×16 submatrices, where Eq. (5) is first applied, followed by Eq. (6) for the full block-matrix calculation, where, the first row of the Toeplitz matrix A is $[A_{15}, A_{14}, \ldots, A_0]$, and the last row is $[A_{30}, A_{29}, \ldots, A_{16}]$. The vector B is given by $B = [B_0, B_1, \ldots, B_{15}]^\top$. Fig. 1 visualizes the general idea of the algorithm. This process is the same for other values of N. For $N = 720$, the final dimension is 20, and for $N = 864$, the dimension is 24, both resulting in 36×36 submatrices.

$$\begin{bmatrix} W_0 \\ W_1 \end{bmatrix} = \begin{bmatrix} A_0 & A_2 \\ A_1 & A_0 \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \end{bmatrix} = \begin{bmatrix} A_0B_0 + A_2B_1 \\ A_1B_0 + A_0B_1 \end{bmatrix} \tag{5}$$

$$\left[W_0, \ldots, W_{15}\right]^\top = \left[\textstyle\sum_{i=0}^{15} A_{15-i}B_i, \sum_{i=0}^{15} A_{16-i}B_i, \ldots, \sum_{i=0}^{15} A_{30-i}B_i\right]^\top \tag{6}$$

A top module manages the padding algorithm and discards extra elements from the result based on matrix dimensions. It is important to highlight that

Algorithm 1 MT-TMVP Polynomial Multiplier

Input: Matrix $T \in \mathbb{R}^{N \times N}$, Vector $v \in \mathbb{R}^{N \times 1}$
Step 1: Zero-pad matrix T and vector v to dimensions divisible by 16:
    Set padded dimension m based on N:
        If $N = 509$, pad to $m = 512$
        If $N = 701$ or $N = 677$, pad to $m = 720$
        If $N = 821$, pad to $m = 864$
    Pad T to form $A^{m \times m}$ and v to form $B^{m \times 1}$ with zeros
Step 2: Divide T into tiles for TMVP-based multiplication:
    Set tile size t based on N:
        $t = 16$ for $N = 512$, $t = 10$ for $N = 720$, $t = 12$ for $N = 864$
    Define TMVP tile size as $2 \times t$
Step 3: TMVP reduction on each $2 \times t$ tile til final reduced dimensions are reached
Step 4: Schoolbook multiplication for reduced matrix of size $t \times t$ with vector B
Step 5: Block Matrix Addition using:
    Eq. (5) for 2×2 structure and Eq. (6) for full block-matrix combination
Output: Result of $T \times v \mod (x^n - 1)$ with padding removed

only the first row and first column of the Toeplitz matrix need to be retained to access all its elements. This property reduces the amount of data that must be stored while still enabling efficient reconstruction of the entire matrix. The design is explained from the top layer of the hierarchy, systematically moving down through each underlying module. This approach provides a detailed view of the interconnections and functions of each module, providing a thorough understanding of the entire design architecture. The design follows a hierarchical structure where Top_TMVP serves as the top-level wrapper module. It instantiates MainMultiplier, which contains TMVP-2, and the innermost module is the MatrixVectorMultiplier. According to N, which defines the matrix dimension, certain aspects of each module's design change. Each section will address these changes in detail. The MainMultiplier module implements the straightforward multiplication algorithm described in Eq. (6). Meanwhile, the TMVP-2 module computes the matrix-vector product using Eq. (3), while the MatrixVectorMultiplier is designed as a simple module for matrix-vector multiplication. This layered approach reflects the modular organization of the hardware implementation. The explanation starts with the most fundamental module and describes how it is used in the higher-level modules. This process continues until it reaches the Top_TMVP module. The match between the MATLAB results and hardware output confirms the correctness of the MT-TMVP design.

MatrixVectorMultiplier. This module is the fundamental component of the design, tasked with computing the product of a matrix and a vector. Given that the matrix is Toeplitz, loading the elements becomes more straightforward. After loading the first row and the vector, the next row can be prepared by shifting the current row elements to the right and reading the element from the first column. By employing this method, there will be no pause or delay in calculating the

result vector. The tree structure used for calculating the summation of the product results varies with different values of N. However, what remains consistent is that there will be no more than three operands for the summation at each step. Fig. 2 illustrates how this module is implemented. It should be noted that a pipelining technique is employed in the addition of the products of element multiplications, that enabled the use of higher frequencies in the design.

TMVP-2. This module implements the algorithm outlined in Eq. (3). In this algorithm, three multiplications are required, as detailed in Eq. (4). Since $s1$ is a common component in both parts of the result, it is calculated first, and the result is stored in `RESULT_RAM`. After calculating $s1$, the process proceeds with the calculation of $s2$, which, along with the stored value, produces the first part of the result. The same process applies to $s3$ to complete the final part of the result. Fig. 3 shows the implementation of this module. The Finite State Machine (FSM) within the control unit manages the flow of the module. The address generator generates the addresses needed to access the required data. Referring to Eq. 3, one can identify the order in which the matrix elements should be read. The same applies to the addresses for accessing vector data. Once the data is read, it undergoes some precalculations before being ready to enter the MatrixVectorMultiplier module. As shown in Fig. 3, there are two adders and two multiplexers involved in this process. $s2$ and $s3$ are quite similar, so their preprocessing is the same, though they use different input data. It is important to note that the only effect of parameter N is on the counters and addresses; it does not introduce any significant structural differences.

MainMultiplier. This module is designed to implement the tiled matrix-vector multiplication algorithm. To accumulate intermediate results from individual matrix-vector multiplications, a BRAM, denoted as $[RESULT\_RAM]$, is used for data storage. Once all calculations are completed and stored in $[RESULT\_RAM]$, the results can be accessed from the RAM and made available at the module's output. An FSM controls the process within this module. A key component of the control unit is the address modifier. Due to the symmetric properties of the Toeplitz matrix, there are only $2n - 1$ distinct submatrices, where n represents the quotient of the matrix dimension divided by the submatrix dimension. Submatrices with the same difference between row and column indices are identical. Therefore, to determine which data to access, the focus is on the difference between the row and column indices. Based on this difference, the addresses generated by TMVP-2 are modified and sent to the top module (Top_TMVP). The architecture of this module is illustrated in Fig. 4. There are only three possible multiplicands corresponding to the dimensions of the submatrix. Therefore, we implement the multiplication in the address modifier using shifts and additions. This approach eliminates the need for Digital Signal Processing (DSP) resources, resulting in lower resource consumption, a shorter critical path, and higher operating frequencies.

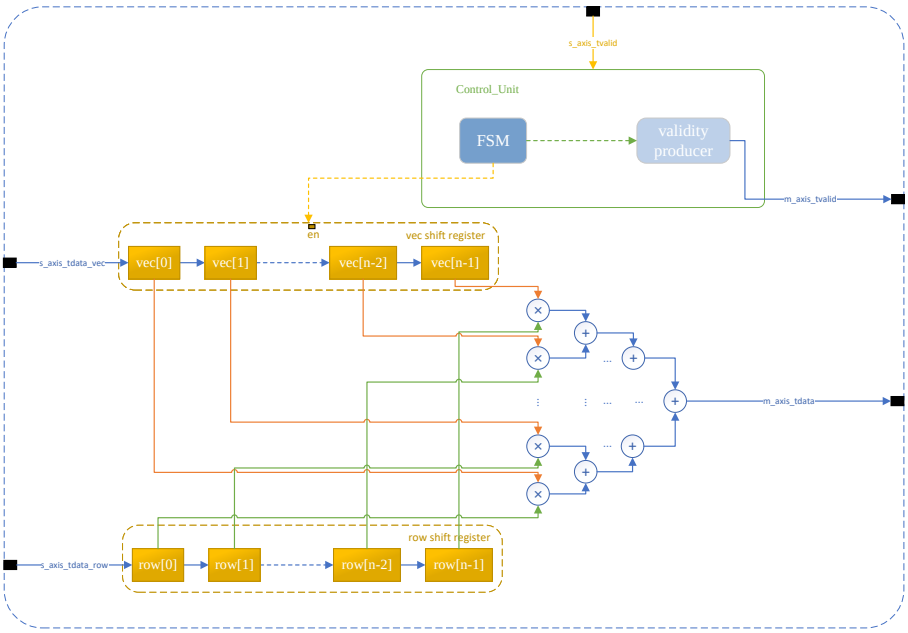

Fig. 2: MatrixVectorMultiplier's Architecture

Top_TMVP. The Top_TMVP module is designed to manage the flow of operations involved in reading coefficients from external sources, using one of the vectors to construct the matrix. The matrix is formed by storing elements from the row and column vectors in a specific order. Additionally, this module is responsible for filling the vector RAM with data from the second set of coefficients, which is read after the first vector is processed. Since the RAM is initialized with zero values, it is not required to pad zeros after writing data to the RAMs. The module then carries out the required multiplications and produces the final result. Fig. 5 illustrates the functionality of the module. The module operates through an FSM with three distinct states. In the `IDLE` state, the module is prepared to initiate operations. Upon receiving the `start` signal, it transitions to the `LOADING` state, during which it reads the two external vectors. Subsequently, the module advances to the `BUSY` state, where it performs the multiplication and outputs the result through its output ports. Once the multiplication is complete, a `done` signal is asserted for one clock cycle to indicate task completion, after which the module returns to the `IDLE` state. It should be noted that in all modules, except for the MatrixVectorMultiplier, the `start` signal indicates the beginning of the process.

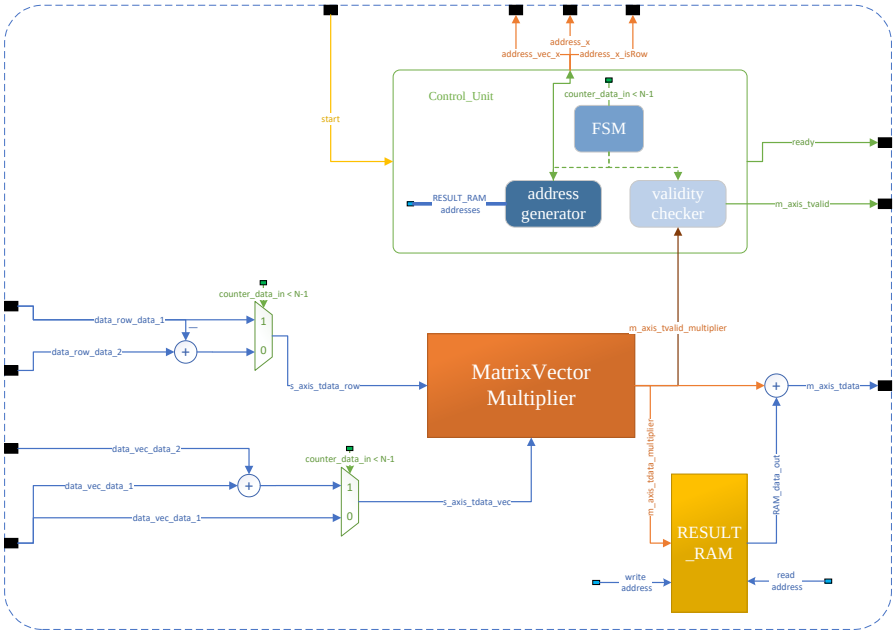

Fig. 3: TMVP-2's Architecture

5 Results and Discussion

This section presents the results of the novel MT-TMVP multiplier implementation on FPGAs, with a focus on resource utilization, which is crucial for practical implementation on resource-constrained devices. NIST recommends the Artix-7 [2] and Zynq Ultrascale+ [3] FPGAs for PQC development and the results are compared with existing multiplier implementations on the same FPGAs for fair comparison. The detailed results of the proposed implementation and previous works in the literature are shown in Tables 1-3.

5.1 Resource Utilization & Performance

The resource utilization of the MT-TMVP's design is shown in Table 1, with the data width set to 8. Synthesis and implementation are conducted using Vivado 2023.2 with default settings. The results are obtained on the Artix-7 and Zynq Ultrascale+ FPGAs with the part numbers xc7a200tffv1156-3 and xczu7ev-ffvf1517-3-e, respectively. The resources utilized for the entire design (including padding and the controller in the top module) are reported separately from those used solely for the multiplier. In the proposed design, the clock frequency is set to 270 MHz. The time required for each multiplication is determined by counting the clock cycles between the assertion of the `start` and `done` signals. Table 2 presents the results for various values of N. The total processing time, including

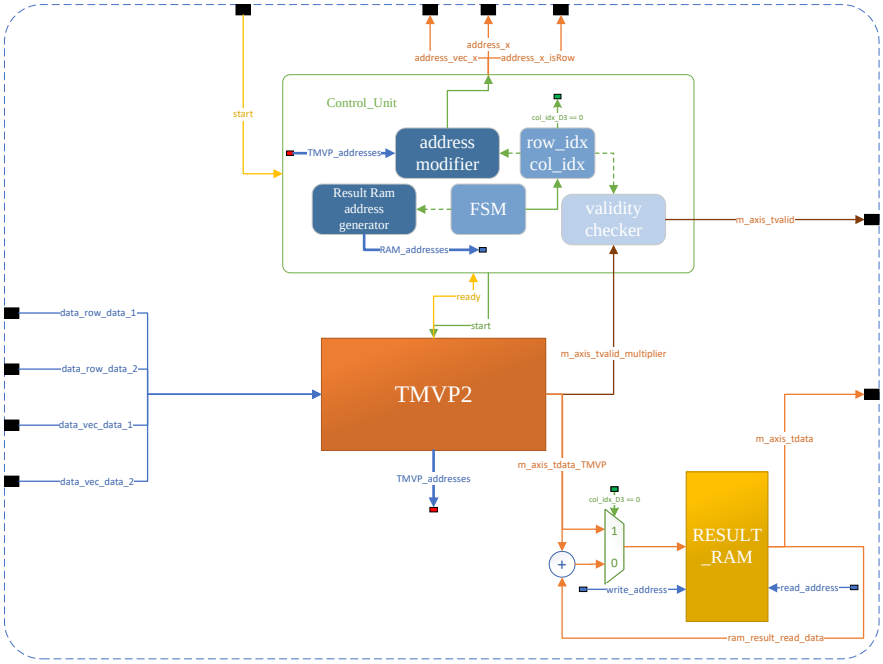

Fig. 4: MainMultiplier's Architecture

Table 1: Resource Utilization for Full and Multiplier-Only Implementations

N	FPGA	LUT		FF		BRAM		DSP	
		Mult. Only	Total	Mult. Only	Total	Mult. Only	Total	Mult. Only	Total
512	Artix-7	308	361	398	528	1	2.5	16	16
	Zynq Ultrascale+	280	329	397	529	1	2.5	16	16
720	Artix-7	350	418	371	515	1	2.5	10	10
	Zynq Ultrascale+	346	394	370	516	1	2.5	10	10
864	Artix-7	356	421	385	529	1	2.5	12	12
	Zynq Ultrascale+	345	393	384	530	1	2.5	12	12

Table 2: Simulation Time Taken for Various Values of N

N	# of Clock Cycles (Mult. Only)	# of Clock Cycles (Total)	Time Elapsed (μs) (Mult. Only)	Time Elapsed (μs) (Total)
512	28,421	28,934	105.262	107.163
720	94,037	94,741	348.285	350.893
864	109,733	110,557	406.419	409.470

data writing to RAMs and the multiplication operation, is reported separately from the time spent solely on multiplication.

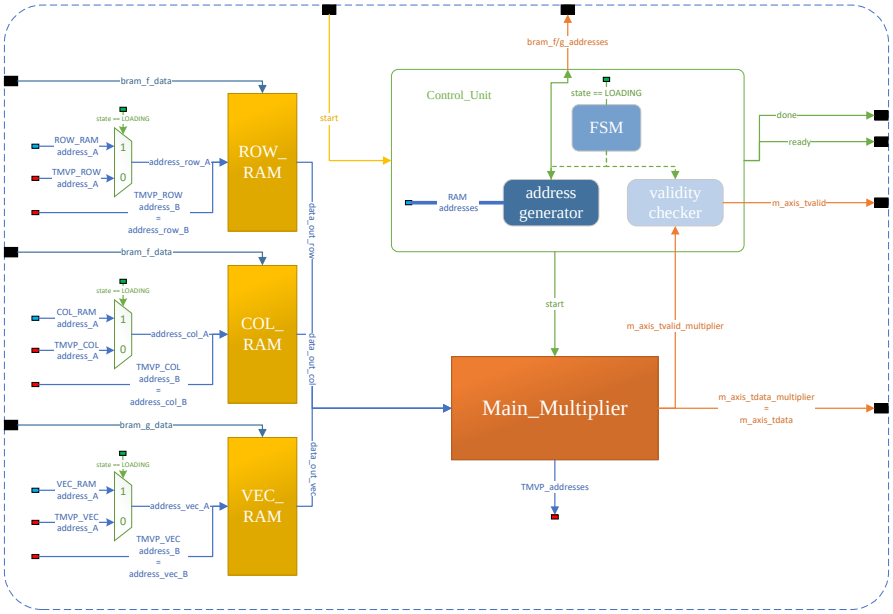

Fig. 5: Top_TMVP's Architecture

5.2 Comparison

The comparison of the proposed MT-TMVP multiplier with other state-of-the-art multiplier implementations for N=512 is presented in Table 3. MT-TMVP is implemented with a focus on optimizing resource utilization, unlike previous works, which prioritized maximizing performance and, as a result, led to higher resource consumption. Tu *et al.* [34] measure ADP as a metric for comparison to achieve a balanced area-time consideration, evaluating both area and time for efficiency assessment. Our proposed scheme shows the best ADP (i.e., LUT count times delay, where delay is defined as latency over maximum frequency) with **99.94%** and **80.02%** improvement and **99.68%** and **84.22%** fewer LUTs on Artix-7 and Zynq Ultrascale+ FPGAs compared to the best existing results.

In comparison with Peng *et al.* [30]'s Schoolbook design on the Zynq Ultrascale+, a direct comparison is not fair due to the difference in modulus q. Their design does not employ BRAM, which makes it vulnerable to timing-based SCAs, whereas the MT-TMVP leverages BRAM to enhance SCA resistance. Li *et al.* [19] design Schoolbook with 7,554 LUTs, compared to which the MT-TMVP achieves a **95.9%** reduction in LUT usage on Artix-7 and a **96.3%** reduction on Zynq Ultrascale+. Note that they did not specify the FPGA type used in their implementation. Both designs employ BRAM, enhancing SCA resistance. The MT-TMVP achieves a **99.92%** improvement in ADP-LUT compared to Li *et al.* (assuming that the FPGA is Zynq Ultrascale+), which indicates its

Table 3: Comparison of MT-TMVP with Multiplier Implementations for N=512

Reference	Multiplier	FPGA	LUT	FF	BRAM	DSP	Freq. (MHz)	Latency (μs)	# of Clock Cycles	ADP-LUT
Peng *et al.* [30]	Schoolbook	Zynq Ultrascale+	1,775	818	0	0	290	89.25	25,881	**546.27**
Li *et al.* [19]	Schoolbook	-	7,554	5,776	3	0	130	2,313	301,000	**134,403.09**
Tu *et al.* [34]	Schoolbook	Zynq Ultrascale+	88,267	35,159	0	0	525	515	270,900	**86,585.72**
Tu *et al.* [34]	Schoolbook	Artix-7	97,322	35,159	0	0	254	516	131,064	**197,709.26**
Allam *et al.* [1]	NTT	Zynq Ultrascale+	2,831	1,394	10.5	12	263	2,830	1,287	**30,462.85**
Tu *et al.* [34]	TMVP	Zynq Ultrascale+	157,686	84,226	0	0	529	260	137,540	**77,541.75**
This Work	MT-TMVP	Artix-7	**308**	**398**	2.5	16	270	105.26	28,421	**120.17**
This Work	MT-TMVP	Zynq Ultrascale+	**280**	**397**	2.5	16	270	105.26	28,421	**109.13**

superior area-time efficiency. Tu *et al.* [34] implemented a Schoolbook-based multiplier on both the Zynq Ultrascale+ and Artix-7 platforms. Their results are obtained with the modulus q set to 2^{13}, making it unfair to directly compare their ADP with our results, which use $q = 256$. However, unlike Tu *et al.*'s design, which lacks BRAM and may be vulnerable to timing-based SCAs, the MT-TMVP incorporates BRAM to enhance SCA resistance. Additionally, they implemented TMVP for the FALCON PQC scheme on the Zynq Ultrascale+. Remarkably, they **did not implement their design on the Artix-7 due to excessive resource utilization** observed on the Zynq Ultrascale+, indicating the impracticality of this approach for resource-constrained devices. Allam *et al.* [1] implemented an NTT multiplier on FPGA that relies on a specific NTT-compatible prime. Unlike NTT, MT-TMVP does not require any modification in polynomial parameters. The MT-TMVP design improves LUT usage by **90.1%**, reducing from their 2831 LUTs to 280 on Zynq Ultrascale+. Both designs use BRAM to resist timing-based SCAs; however, the proposed MT-TMVP achieves this with only 2.5 BRAMs, compared to the 10.5 BRAMs used in their design. Kundi *et al.* [18] implemented NTT and Schoolbook multiplications on FPGAs, prioritizing speed. However, since their results are presented for $N = 256$, a direct comparison with our work is not fair.

In summary, the proposed MT-TMVP shows LUT usage improvements ranging from **84.22%** to **99.68%** compared to existing designs. The MT-TMVP also leverages BRAM in both Artix-7 and Zynq Ultrascale+ implementations, providing security against timing-based SCAs. The MT-TMVP achieves the best ADP-LUT results, with improvements ranging from **80.02%** to **99.94%** over the closest state-of-the-art designs. Additionally, the modular property of the MT-TMVP allows it to be scaled to support various matrix sizes and polynomial degrees by adjusting the tile size in the top module which provides flexibility.

6 Conclusion and Future Directions

In this paper, a novel polynomial multiplication algorithm (MT-TMVP) is proposed for lattice-based PQC schemes targeting resource-constrained devices. A

modular architecture is implemented using BRAMs on Zynq Ultrascale+ and Artix-7 FPGAs, recommended by NIST. Due to the modular property of the design, it provides flexibility and can be scaled up to any other dimensions. The results demonstrate a significant reduction in LUT utilization and achieve the best ADP compared to the state-of-the-art multiplier implementations. In future, we plan to extend the approach to other algorithms for broader benchmarking. Since the current design is secure against timing SCAs due to BRAM usage [30,15], further security analysis is planned to assess its resistance to physical attacks like power analysis. Potential countermeasures, such as masking techniques, will be investigated, though they introduce additional resource costs.

7 Source Code

The source code is available on GitHub in the following link: `https://github.com/sec-int/mt-tmvp`.

8 Acknowledgement

This work is partially supported by the DFG Project Number 543352068 "NSF-DFG: SaTC: Core: Small: A Unified Hardware Design for the USA and German Post-Quantum Standards".

References

1. Allam, H., Mandal, S., Roy, D.: A comparative analysis between Karatsuba, Toom-Cook and NTT multiplier for polynomial multiplication in NTRU on FPGA. In: 2023 Asian Hardware Oriented Security and Trust Symposium (AsianHOST). pp. 1–6. IEEE (2023)
2. AMD: Artix-7 data sheet. `https://docs.amd.com/v/u/en-US/ds181_Artix_7_Data_Sheet` (2024), last accessed 2024/09/29
3. AMD: Zynq Ultrascale+ overview. `https://docs.amd.com/v/u/en-US/ds891-zynq-ultrascale-plus-overview` (2024), last accessed 2024/09/29
4. Balamurugan, C., Singh, K., Ganesan, G., Rajarajan, M.: Post-quantum and code-based cryptography—some prospective research directions. Cryptography **5**(4), 38 (2021)
5. de Boer, K., van Woerden, W.: Lattice-based cryptography: A survey on the security of the lattice-based NIST finalists. `https://eprint.iacr.org/2025/304.pdf` (2025)
6. Braun, K., Fritzmann, T., Maringer, G., Schamberger, T., Sepúlveda, J.: Secure and compact full NTRU hardware implementation. In: 2018 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC). pp. 89–94. IEEE (2018)
7. Cui, Y., Zhang, Y., Ni, Z., Yu, S., Wang, C., Liu, W.: High-throughput polynomial multiplier for accelerating Saber on FPGA. IEEE Transactions on Circuits and Systems II: Express Briefs **70**(9), 3584–3588 (2023)

8. Dam, D., Tran, T., Hoang, V., Pham, C., Hoang, T.: A survey of post-quantum cryptography: start of a new race. Cryptography **7**(3), 40 (2023)
9. Dang, V., Mohajerani, K., Gaj, K.: High-speed hardware architectures and FPGA benchmarking of CRYSTALS-Kyber, NTRU, and Saber. IEEE Transactions on Computers **72**(2), 306–320 (2022)
10. Easttom, C.: Quantum computing and cryptography. Springer International Publishing, Cham (2022)
11. Hafeez, M., Lee, W., Karmakar, A., Hwang, S.: Efficient TMVP-based polynomial convolution on GPU for post-quantum cryptography targeting IoT applications. IEEE Internet of Things Journal (2024)
12. He, P., Xie, J.: Novel implementation of high-performance polynomial multiplication for unified KEM Saber based on TMVP design strategy. In: 2023 24th International Symposium on Quality Electronic Design (ISQED). pp. 1–8. IEEE (2023)
13. Hoffstein, J., Pipher, J., Silverman, J.: Ntru: A ring-based public key cryptosystem. In: International Algorithmic Number Theory Symposium. pp. 267–288 (1998)
14. Karmakar, A., Mera, J., Roy, S., Verbauwhede, I.: Saber on ARM: CCA-secure module lattice-based key encapsulation on ARM. IACR Transactions on Cryptographic Hardware and Embedded Systems **2018**(3), 243–266 (2018)
15. Khan, S., Lee, W.K., Khalid, A., Majeed, A., Hwang, S.: Area-optimized constant-time hardware implementation for polynomial multiplication. IEEE Embedded Systems Letters (2022)
16. Kim, J., Park, J.: NTRU+PKE: Efficient public-key encryption schemes from the NTRU problem. `https://eprint.iacr.org/2024/1282.pdf` (2024)
17. Koblitz, N.: Elliptic curve cryptosystems. Mathematics of Computation **48**(177), 203–209 (1987)
18. Kundi, D., Zhang, Y., Wang, C., Khalid, A., O'Neill, M., Liu, W.: Ultra high-speed polynomial multiplications for lattice-based cryptography on FPGAs. IEEE Transactions on Emerging Topics in Computing **10**(4), 1993–2005 (2022)
19. Li, D., Zhong, J., Cheng, S., Zhang, Y., Gao, S., Cui, Y.: High-performance hardware implementation of the Saber key encapsulation protocol. Electronics **13**(4), 675 (2024)
20. Malygina, E., Kutsenko, A., Novoselov, S., Kolesnikov, N., Bakharev, A., Khilchuk, I., Shaporenko, A., Tokareva, N.: Post-quantum cryptosystems: open problems and solutions. lattice-based cryptosystems. Journal of Applied and Industrial Mathematics **17**(4), 767–790 (2023)
21. Malygina, E., Kutsenko, A., Novoselov, S., Kolesnikov, N., Bakharev, A., Khilchuk, I., Shaporenko, A., Tokareva, N.: Post-quantum cryptosystems: open problems and current solutions. isogeny-based and code-based cryptosystems. Journal of Applied and Industrial Mathematics **18**(1), 103–121 (2024)
22. Meszlényi, L., Kavun, E., Keskinkurt-Paksoy, , Khalid, A., Yalçın, T.: A scalable hardware/software co-design approach for efficient polynomial multiplication. In: 2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD). pp. 1–5. IEEE (2023)
23. del Moral, J., iOlius, A., Vidal, G., Crespo, P., Martinez, J.: Cybersecurity in critical infrastructures: a post-quantum cryptography perspective. arXiv preprint arXiv:2401.03780 (2024)
24. Nejatollahi, H., Dutt, N., Ray, S., Regazzoni, F., Banerjee, I., Cammarota, R.: Post-quantum lattice-based cryptography implementations: A survey (2019)
25. NIST: Falcon NIST finalist. `https://falcon-sign.info/` (2024), last accessed 2024/09/07

26. NIST: Post-quantum cryptography. `https://csrc.nist.gov/projects/post-quantum-cryptography` (2024), last accessed 2024/10/10
27. Paar, C., Pelzl, J.: Understanding cryptography: a textbook for students and practitioners. Springer Science & Business Media, New York (2009)
28. Paksoy, , Cenk, M.: TMVP-based multiplication for polynomial quotient rings and application to Saber on ARM Cortex-M4. Cryptology ePrint Archive (2020)
29. Paksoy, , Cenk, M.: Faster NTRU on ARM Cortex-M4 with TMVP-based multiplication (2022)
30. Peng, B.Y., Marotzke, A., Tsai, M.H., Yang, B.Y., Chen, H.L.: Streamlined NTRU Prime on FPGA. Journal of Cryptographic Engineering **13**(2), 167–186 (2023)
31. Qin, Z., Tong, R., Wu, X., Bai, G., Wu, L., Su, L.: A compact full hardware implementation of PQC algorithm NTRU. In: 2021 International Conference on Communications, Information System and Computer Engineering (CISCE), IEEE. pp. 792–797 (2021)
32. Rivest, R., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM **21**(2), 120–126 (1978)
33. Shor, P.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Review **41**(2), 303–332 (1999)
34. Tu, Y., Bai, S., Xiong, J., Xie, J.: Novel schoolbook-originated polynomial multiplication accelerators for NTRU-based PQC. In: 5th NIST PQC Standardization Conference. pp. 1–13 (2024)
35. Ustimenko, V.: On historical multivariate cryptosystems and their restorations as instruments of post-quantum cryptography. Cryptology ePrint Archive (2024)
36. Wang, J., Yang, C., Zhang, F., Meng, Y., Su, Y.: Tcpm: A reconfigurable and efficient Toom-Cook-based polynomial multiplier over rings using a novel compressed postprocessing algorithm. IEEE Transactions on Very Large Scale Integration (VLSI) Systems (2023)
37. Yaman, F., Mert, A., Öztürk, E., Savaş, E.: A hardware accelerator for polynomial multiplication operation of CRYSTALS-KYBER PQC scheme. In: 2021 Design, Automation Test in Europe Conference Exhibition (DATE), IEEE. pp. 1020–1025 (2021)
38. Zheng, J., Zhang, H., Tian, L., Zhang, Z., Wei, H., Chu, Z., Yang, Y., Zhao, Y.: ESPM-D: efficient sparse polynomial multiplication for Dilithium on ARM Cortex-M4 and Apple M2. arXiv preprint arXiv:2404.12675 (2024)