High-Performance Polynomial Multiplication Hardware Accelerators for KEM Saber and NTRU

Elizabeth Carter, Pengzhou He, and Jiafeng Xie,

***Abstract*—Along the rapid development in building large-scale quantum computers, post-quantum cryptography (PQC) has drawn significant attention from research community recently as it is proven that the existing public-key cryptosystems are vulnerable to the quantum attacks. Following this direction, this paper presents a novel implementation of high-performance polynomial multiplication hardware accelerators for key encapsulation mechanism (KEM) Saber and NTRU, two PQC algorithms that are currently under the consideration by the National Institute of Standards and Technology (NIST) PQC standardization process. In total, we have carried out three layers of efforts to obtain the proposed work. First of all, we have proposed a new Dual Cyclic-Row Oriented Processing (Dual-CROP) technique to build a high-performance polynomial multiplication hardware accelerator for KEM Saber. Then, we have extended this hardware accelerator to NTRU with proper innovation and adjustment. Finally, through a series of complexity analysis and implementation based comparison, we have shown that the proposed hardware accelerators obtain better area-time complexities than known existing ones. It is expected that the outcome of this work can impact the ongoing NIST PQC standardization process and can be deployed further to construct efficient cryptoprocessors.**

***Index Terms*—Dual cyclic-row oriented processing (Dual-CROP), high-performance, key encapsulation mechanism (KEM) Saber, NTRU, polynomial multiplication hardware accelerator, post-quantum cryptography (PQC)**

I. Introduction

With the rapid progression in quantum computing, it is proven that most of the existing public key cryptographic algorithms will no longer be secure as they can be solved by large-scale quantum computers executing Shor's algorithm [1], [2], [3], [4]. Consequently, it is predicted that the widely used Rivest Shamir Adleman (RSA) algorithm and Elliptic-Curve Cryptography (ECC), will no longer be secure or viable options within 10-15 years[1], [2], [3], [4].

In response, cryptosystems resistant to quantum, attacks known as post quantum cryptography (PQC), have been proposed as alternatives. In particular, the National Institute of Standards and Technology (NIST) has started the PQC standardization process with the goal to standardize algorithms for public-key encryption, key-establishment and digital authentication which will remain secure even with use of quantum computers. In total, NIST has received 69 submissions for potential standards and NIST has moved forward to the 3rd round of the standardization process with four finalists for public-key encryption and key-establishment schemes as of July of 2020 [5].

As indicated by the 3rd round PQC standardization process [5], there are lattice-based cryptography(LBC), code-based cryptography, isogeny-based cryptography (etc.) currently under consideration for PQC candidates. Of these PQC classifications, lattice-based cryptography is recognized as one of the most promising [5], [6], [7]. Furthermore, among the announced four public-key and key-establishment finalists, three out of the four are LBC schemes; these being Crystals-Kyber, Saber, and N-th degree truncated polynomial ring (NTRU)[8], [9], [10]. The announced finalists echoed the on-going research trend that LBC is widely recognized as one of the most prominent PQC classes due to its ease of implementation and strong security proof, which is based on worst-case hardness [6], [7], [11], [12].

In general, the LBC algorithms are built on two types of lattice problems, namely the NTRU problem and the learning-with-errors (LWE) problem. There also exist several variants of these problems, such as the Ring-LWE problem and the learning-with-round (LWR) problem as well as their module variants. Quite a good number of works have been released on the LWR problem [**?**], including the key encapsulation mechanism (KEM) Saber [5], which is one of the NIST 3rd round PQC finalists. Furthermore, among the NIST announced third round public-key finalists, NTRU is based on the NTRU problem, KYBER is based on the module Ring-LWE (MLWE) problem, and Saber is based on the module LWR (MLWR) problem [5].

One of the "drawbacks" for LBC, however, lies in the fact that LBC algorithms typically require relatively large key sizes [6], [7], [11], [12]. Following the NIST PQC standardization process, more attention has been gradually focused on the efficient hardware implementation of the two PQC schemes Saber and NTRU. As these algorithms move forward in the standardization process or for other purposes, research has been carried out on various implementations for optimizations to handle the size of data used and to additionally speed up the computation process. Within hardware platforms like field-programmable gate arrays (FPGAs) especially, there is potential for optimizations of LBC in aspects of resource consumption and time complexity.

Based on this consideration, this work proposes novel design methods for targeted LBC algorithms, in compact and time efficient hardware accelerators on the FPGA platform. In general, LBC algorithms require polynomial multiplication (PM) at many points of key generation, encryption, and

Manuscript received XXX, 2022. (corresponding author: Jiafeng Xie). The first two authors contribute equally.

P. He, E. Carter, and J. Xie are with the Department of Electrical and Computer Engineering, Villanova University, Villanova, PA, 19085 USA (e-mail: {phe, ecarter9, jiafeng.xie}@villanova.edu).

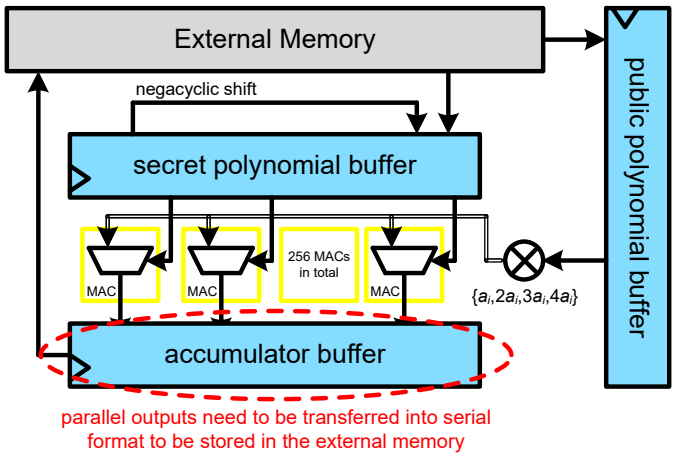

Fig. 1: The existing polynomial multiplication structure [20], where the produced outputs (in parallel) need to be transferred into serial format to be stored in the external memory.

decryption, and these PMs are extremely time and resource intensive processes. For instance, within Saber, PM takes up to 56% of overall computation time [13]. Reducing the clock cycles of PM can in turn reduce the overall implementation time complexity of the LBC schemes. To obtain the targeted performance metrics, this work specifically focuses on novel implementation methods of PM and related point-wise multiplication (PWM). Overall, this work focuses on optimizing the PM hardware accelerators for Saber and NTRU PQC schemes.

Existing Works. KEM Saber is built on the Module-LWR (MLWR) problem, which is a module variant of LWR. Upon its original introduction in [9], [**?**], many works have been released on this interesting PQC scheme, ranging from security level, implementation, and attack analysis [**?**], [14], [15]. Especially for the hardware implementations, including both the system-level and component-level designs (such as polynomial multiplication), we can categorize them into two types: (i) hardware-software co-design; and (ii) full hardware design. The first type includes the recent one of [16], where the Toom-Cook method is used to implement the polynomial multiplication for KEM Saber. Another recent design of [17] also uses the Toom-Cook approach to achieve high-performance implementation. A very recent report has proposed to use the number theoretic transform (NTT) [18] for the implementation of the polynomial multiplication of KEM Saber on the RISC-V accelerator. For the second type, a new coprocessor for KEM Saber is introduced in [13], where the polynomial multiplication is based on a schoolbook based method. A Karatsuba algorithm based KEM Saber is reported in [19] for high-performance operation. Optimized polynomial multiplication structures for the polynomial multiplication in KEM Saber is recently presented in [20], which has better area-time complexities than the previous designs of [16], [21]. Overall, these two types of designs are the major hardware implementation works for KEM Saber.

The polynomial multiplication over ring $\mathbb{Z}_l/(x^n+1)$ (l is either q or p [9], [13]) is the critical arithmetic operation of KEM Saber. But the existing works have not well covered its efficient implementation: (i) the existing high-performance works, such as the structure of Fig. 1 in [20] (see Fig. 1 here) produces the multiplication outputs in a parallel format, which actually requires extra resources such as multiplexers (MUXes) to transfer the parallel outputs into serial style to be stored in the external memory for further usage; (ii) not many efficient hardware structures for the polynomial multiplication have been proposed for KEM Saber. Noticing that polynomial multiplications over other fields such as binary field have been investigated widely in the literature [22], [23], [24], [25], we just follow this trend to propose an efficient implementation of polynomial multiplication for KEM Saber on the field-programmable gate array (FPGA) platform for high-performance applications. Specifically, we have followed the design style presented in [26] and have proposed a novel cyclic-row oriented processing (CROP) strategy that all the outputs are circularly accumulated and can be very easily transferred to the external memory in a serial format with little extra resource usage (simple operation).

Major Contributions. We have proposed a novel algorithmic derivation, architectural design, and implementation techniques to achieve efficient implementation of polynomial multiplication within KEM Saber and NTRU. In total, we have carried out three layers of innovative works, as:

- A new Dual Cyclic-Row Oriented Processing (Dual-CROP) technique to build a high-performance polynomial multiplication hardware accelerator for KEM Saber.
- A modified Dual CROP, in order to extended this hardware accelerator to NTRU with proper innovation and adjustment.
- A thorough complexity analysis and comparison (including both the theoretical analysis and FPGA based implementation performance) to show that the proposed polynomial multiplications have better area-time complexities than the state-of-the-art solutions.

Overall, the proposed polynomial multiplication possesses three main unique features: (i) simple and easy operation on the output result delivering; (ii) flexible offering of processing throughput; and (iii) low-complexity. Discussions about the further extension and application of the proposed polynomial multiplications have also been provided.

The rest of this paper is organized as follows. The preliminary knowledge is introduced in Section II. The formulation of the proposed Dual CROP strategy is detailed presented in Section III along with proposed algorithms. The proposed hardware polynomial multiplication structures, for both Saber and NTRU, are provided in Section IV. Complexity analysis and comparison are presented in Section V. Finally, conclusions are given in Section VI.

II. Preliminaries

In this section, we briefly give the introduction of the KEM Saber, NTRU, and the involved polynomial multiplication. Interested readers can refer to the original papers of [9], [**?**], [10] for details.

A. The MLWR Scheme (KEM Saber)

The LWR is a variant of LWE [27], which uses the rounding operation to replace the previous Gaussian distributed errors to obtain the hardness of the lattice problem. The LWR problem is based on the equation of $(a, b = \lfloor \frac{p}{q}\langle a, s\rangle \rceil_p) \in \mathbb{Z}_q^n \times \mathbb{Z}_p$ (where both p and q are power-of-two moduli), and the MLWR scheme is the module version of the LWR. It operates on the ring $R_q = Z_q[X]/(x^n + 1)$ with the modulus of $(x^n + 1)$ [9], [**?**].

Saber is an MLWR scheme based PQC, which achieves both classical and quantum security [**?**]. Saber is first constructed as a Chosen Plaintext Attack (CPA) secure public-key encryption scheme and then developed into KEM Saber through the Fujisaki-Okamoto transformation [28].

Similar to other PQC schemes, the Saber public-key encryption scheme consists of three operational phases, i.e., the key generation, the encryption, and the decryption phases [9], [**?**]. In the key generation phase, the public matrix of polynomials A and a secret vector of polynomials s are used to produce the scaling and rounding output of As (also the vector b), where the public key is composed of A and b and the secret key is the vector s. In the encryption phase, the original message is encrypted through $v' = s'b$ (generated new secret s') and the final produced ciphertext involves the vector b' (from rounding As'). The decryption phase uses the secret key to obtain v, which is approximately the same as v' in the encryption phase (allows the recovering of the original message from the ciphertext). KEM Saber uses the Fujisaki-Okamoto transformation [28] to further ensure its CCA-secure.

Parameter Setting [9]. The polynomial multiplication involved within KEM Saber is set as degree of $n = 256$ and the two moduli are $q = 2^{13}$ and $p = 2^{10}$, respectively. The related secrets are sampled from the binomial distribution. Additionally, KEM Saber uses a modulo ring of $(x^n + 1)$.

Polynomial Multiplication for Saber The polynomial multiplication (degree of 256) is the key arithmetic operation in the above mentioned phases. One polynomial involves coefficients generated from the binomial sampler, and these coefficients lie in the value range of -4 to $+4$ [20], while another polynomial operand consists of coefficients of either 10-bit or 13-bit (the 13-bit based design can also be used for the 10-bit based computation). The design of [20] has used the schoolbook algorithm to derive the desired high-performance structures. But the proposed polynomial multiplication structures have not fully optimized the output delivery, and hence further efforts are needed in this area.

B. The NTRU Scheme

The NTRU PQC scheme is built on the Nth degree TRUncated polynomial ring (NTRU) problem that is quantum secure and was originally defined in 1998 [**?**], [**?**]. NTRU is know to be a secure option for PQC standardization.

Parameter Setting The polynomial multiplication involved with NTRU is a set as a degree of $n = 821$ and $q = 2^{12}$. Variations in the parameters for different levels of security can be further explored in [10], however n is always a prime value and q is a power of two.

Polynomial Multiplication for NTRU The polynomial multiplication (degree of 821) is the key arithmetic operation in the above mentioned phases. Two polynomials of both consists of coefficients of 12-bit in signed magnitude form. Unlike Saber which uses a modulo ring of $(x^n + 1)$, NTRU uses a modulo of $(x^n - 1)$ [10], [11]. This reduces the overall complexity of computations since a negation is not required during the cyclic shifting process.

III. Dual CROP: Mathematical Formulation

This section presents the mathematics behind a novel hardware implementation technique to design PMs within Saber and NTRU. PM of polynomials $B = \sum_{i=0}^{n-1} b_i x^i$ and $A = \sum_{i=0}^{n-1} a_i x^i$ over a modulus ring can be represented by multiplication of a $n \times n$ matrix with a $n \times 1$ vector, shown in Figure 2. Here the multiplication is done with the modulus $(x^n + 1)$ and a degree of $n = 4$.

The Dual Cyclic Row Oriented Processes (Dual CROP) design offers a time-efficient yet lightweight format to calculate PMs within the targeted PQC schemes of Saber and NTRU. The proposed design is based on the CROP technique (originally proposed in [**?**]). This work proposes a new CROP-based method that offers hardware acceleration by reducing the over all number of cycles for computing the polynomial product. To do this, the operands A and B are divided into two groups. The Figure 2 also depicts how this deviation is made within the case study example matrix. Each divided group is fed into a respective PM unit and computed concurrently to produce the intermediate products, shown in Figure 3. For a degree of $n = 4$, the values $a_0, a_1, ..., a_{n-1}$ are multiplied with b_1 and b_0, while the modulus values $-a_{n-2}, -a_{n-1}, ..., a_1$ are multiplied with b_2 and b_3. This process is stated in Definition 4.1 for Saber. For NTRU, Definition 4.1 can be used by replacing the modulus (x^n+1), by simply replacing all negations generated by the ring with positives equivalents and inserting a zero value to account for the odd degree of n .

Definition 4.1 (Mathematical basics for Dual CROP based PM of Saber)
Define polynomials $A = \sum_{i=0}^{n-1} a_i x^i$, $B = \sum_{i=0}^{n-1} b_i x^i$ and $C = \sum_{i=0}^{n-1} c_i x^i$ where

$$C = BA \bmod (x^n + 1). \tag{1}$$

This can be rewritten as

$$C = b_0(Ax^i \bmod (x^n+1)) + ... + b_{n-1}(Ax^i \bmod (x^n+1)). \tag{2}$$

Also, since $x^n \equiv -1$, we can then substitute it in (4.2) to get

$$\begin{aligned} C =& b_0(a_0 + a_1x + ... + a_{n-1}x^{n-1}) \\ &+b_1(-a_{n-1} + a_0x + ... + a_{n-2}x^{n-1}) \\ &+.... \\ &+b_{n-1}(-a_1 - a_2x - ... + a_0x^{n-1}). \end{aligned} \tag{3}$$

As a further extension, C can be split into C_0 and C_1 where $C = C_0 + C_1$, for

$$\begin{aligned} C_0 =& b_0(a_0 + a_1x + ... + a_{n-1}x^{n-1}) \\ &+b_1(-a_{n-1} + a_0x + ... + a_{n-2}x^{n-1}) \\ &+.... \\ &+b_{n/2-1}(-a_{1+n/2} - a_{2+n/2}x - ... + a_2x^{n-1}), \end{aligned} \tag{4}$$

and

$$\begin{aligned} C_1 =& b_{n/2}(-a_{n/2} - a_{1+n/2}x - ... + a_{n/2-1}x^{n-1}) \\ &+b_{n/2+1}(-a_{n/2-1} - a_{n/2}x - ... + a_{n/2-2}x^{n-1}) \\ &+.... \\ &+b_{n-1}(-a_1 - a_2x - ... + a_0x^{n-1}). \end{aligned} \tag{5}$$

$$\begin{bmatrix} a_0 & -a_3 \\ a_1 & a_0 \\ a_2 & a_1 \\ a_3 & a_2 \end{bmatrix} \begin{bmatrix} -a_2 & -a_1 \\ -a_3 & -a_2 \\ a_0 & -a_3 \\ a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_0b_0 - a_3b_1 \\ a_1b_0 + a_0b_1 \\ a_2b_0 + a_1b_1 \\ a_3b_0 + a_2b_1 \end{bmatrix} \begin{matrix} -a_2b_2 - a_1b_3 \\ -a_3b_2 - a_2b_3 \\ +a_0b_2 - a_3b_3 \\ +a_1b_2 + a_0b_3 \end{matrix}$$

Fig. 2: Matrix representation of the proposed Dual CROP based technique for Saber, which can be extended to NTRU without negations.

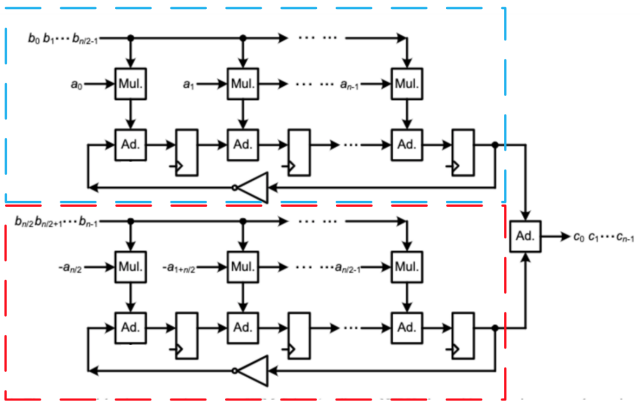

Fig. 3: Concept of a Dual CROP based PM (for Saber, which can apply to NTRU without the sign control).

By simultaneously computing the intermediate products, the final polynomial product is computed within half of the clock cycles as the original design. Similar to the original design, the final output is delivered through a serial processing format and therefore does not require the additional processing and multiplexers that are required in [20] (the hardware cost is thus reduced).

Overall, one cycle is required to reset the registers, n cycles for loading in the first set coefficients, $n/2$ cycles are needed for computing the products and finally n cycles for serially outputting the results. One advantage of this structure is that it can be utilized to produce the product of multiple polynomials sequentially without resetting the registers before each computation. This allows for the multitude of multiplications required within PQC schemes to occur back to back without the cost of an additional clock cycle with each operation.

In the following sections, we will introduce two main variants of the Dual CROP based structure, one for Saber and one for NTRU. The Saber-based design is first discussed in detail as a case study example, and then modifications are made to obtain the proposed structure for NTRU. The two proposed designs are largely the same except on the implementation methods for PWMs, related data bit sizes, and deployed polynomial sizes.

IV. Dual CROP based Hardware Structures (KEM Saber)

KEM Saber requires the PWM of coefficients with 13-bits or 10-bits with coefficients with 4-bits [20]. The 13-bit/10-bit values are in the form of two's complement while the 4-bit values are usually represented in signed magnitude form. This structure is designed for Saber which uses a secret (s) in the range of [-4, 4] instead of LightSaber or FireSaber which has a range of [-5, 5] or [-3, 3] respectively. Nevertheless, the implemented hardware structure could be expanded to LightSaber and FireSaber by altering the overall value ranges and related bit width. Saber utilizes modulus $(x^n + 1)$ and n values of power of two, and the proposed hardware structure requires a sign control unit for the modular related operation.

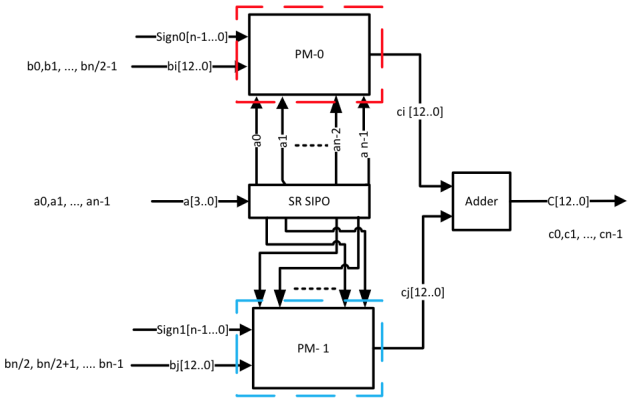

Fig. 4: Top-level discription of the Dual CROP based PM for Saber. (NTRU does not need the sign control).

The **top level entity** in Figure 4 depicts the overall hardware accelerator structure for PM within Saber. The original computation is split into two major PM components and it is also shown that how various types of components are shared between the two structures. The serial-in parallel-out shift register (SR-SIPO) is first loaded with a values (4-bits in Saber, which could be 12-bits for NTRU). Once all values are stored in the shift register, computations begin in the PM unit as two b values are delivered-in simultaneously. For the first PM, a values are fed into the accelerator in parallel and in order. For the second PM, the a values are selected in a manner that the inputs are shifted by $n/2$ rotations from their initial state (i.e. $a_{n/2}, a_{n/2+1}, ...a_{n/2-1}$). Each of the two PM components generate the intermediate products in series (i.e., C_0 and C_1), which are summed to return the final product, i.e., $C = c_0, c_1, ...c_{n-1}$.

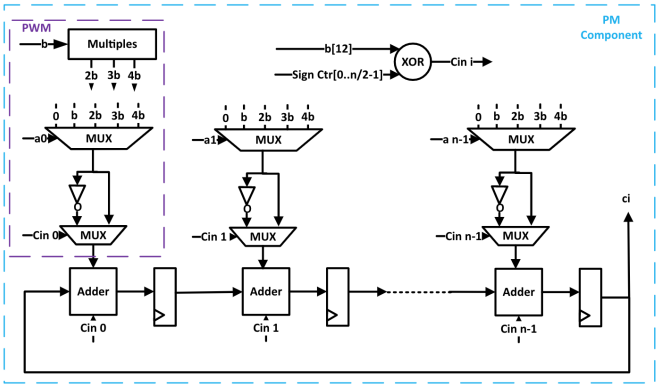

Fig. 5: PM unit for Dual CROP based hardware accelerator (Saber).

A single **PM component** for Saber is shown in Figure 5. This hardware unit calculates the point-wise products and accumulates those values in a circular fashion. No inputs are required to be shifted according to this setup, i.e., all the shifting is done by the circulate unit involved with the registers, and meanwhile a new b coefficient is fed to the structure on each iteration. After n cycles, the first intermediate product is produced at the first register. As the values are output serially, the results will be shifted through the cyclic structure. To avoid unnecessary accumulations during the output delivery time, the input b need to be set to zero. On the other hand, another set of operands can be input into the PM to compute back to back polynomial products.

Notice that the inverter is removed from the original PM design (Figure 3) and is replaced with sign controls driving the carry-ins. This sign control, creates the modulus $(x^n + 1)$ related operation that Saber uses. During the computational stage, a **sign control component** generates the negations required by Saber's modulus. Figure 6 depicts the sign control component.

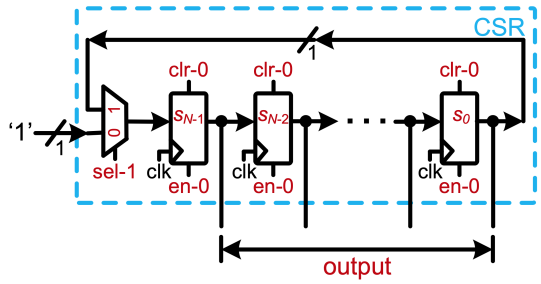

Fig. 6: Sign control component for Dual CROP based PM accelerator (for Saber, but not for NTRU).

The sign control generates half of the sign controls required (i.e., $sc_0, sc_1, ...sc_{n/2-1}$), and then they are fed into both PM units. The sign controlled values $(sc_{n/2}, sc_{n/2+1}, ..., sc_{n-1})$ are constants and hence there is no need for them to be generated via the sign control unit. This allows for a smaller sign control unit as only half the values need to be generated during each iteration. For PM0, the values $(sc_{n/2}, sc_{n/2+1}, ..., sc_{n-1})$ are always zero, while for PM1 they are always one. This is due to the nature of the matrix A in Figure 2 that the upper values (e.g., a_3 and a_2) in blue are never negated and they are directly sent into PM0, while the upper values in red are always negated and then sent into PM1. Within the PM component, the sign control and the highest order bit of b are used to decide whether a negation occurs. If a negation is going to occur then the carry-ins of the respective adder is set high to compute according to the two's complement negation by adding an additional one to the sum. Additionally, if the negation is going to occur, there is also an inversion needed for a two's complement negation inside the related PWM.

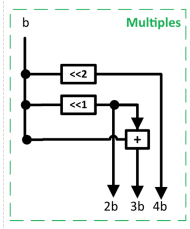

Fig. 7: Multiples component for PWM used in Dual CROP based hardware accelerator of Saber.

For Saber, the PM units use the lookup table based approach for the involved PWMs. [**?**] and [13] introduced a PWM scheme that one can select the product based on one of the operands. Since the range of all possible values is relatively small, the number of all possible products is also small. By computing all possible products a multiplexer can be used to select the appropriate precomputed product. Additionally, as the 4-bit value is of signed magnitude form, only 4 values absolute values must be precomputed $(0, d_i, 2d_i, 3d_i, 4d_i)$ and if the negative values $(-d_i, -2d_i, -3d_i, -4d_i)$ are required they are selected with the sign bit in a subsequent step. The sign bit controls a second multiplexer which either selects the inverse or the existing value.

The **multiples component** computes all absolute value combinations and then multiplexers are used to select the appropriate product, as shown in Figure 7. This unit can be shared among all multiplexer units, so only one multiples component is need for each of the two PM units. This greatly reduces the area consumption of all the PWMs. If a negative value is needed, the sign control selects the product's complement and a value of one is added via the carry-in within the PM unit.

V. Dual CROP based Hardware Structures (NTRU)

The design previously discussed for Saber in the previous section can be applied to NTRU as well. With modification to the modulus, the bit width, and the related PWMs. NTRU uses mod$(x^n - 1)$ with a degree $n = 821$ and coefficients of bit size of 12 ($q = 2^{12}$). As NTRU applies the modulus $(x^n - 1)$ rather than the modulus $(x^n + 1)$ that Saber applies, we can simplify the hardware Dual CROP based PM accelerator. The sign control unit can be removed as there is no need for negations during the accumulations. Additionally, the carry-in of each adder in the PM unit can be set to 0. Finally, operand

and product bit widths are adjusted to 12-bits according to the parameter setting of NTRU.

The method used in the Saber design (i.e. the lightweight look up table method) is not ideal for NTRU as the absolute values represented with 12-bits are too large for the original look up table method, when considering area-efficiency. The look up table design can be applied to other operand bit lengths, however, as the number of total possible products increases so does the area consumption. As a result reduced logic add and shift method is used with the NTRU design.

VI. Complexity Analysis and Comparison

Complexity Analysis: Table I summarizes the area-time complexities of the design proposed along with other competing designs in the literature. The theoretical area-complexity of the proposed Dual CROP based PM accelerator for Saber is listed as follows: each PM unit of the proposed accelerator has N 13-bit registers, N adders, and N multipliers. The Dual CROP based PM for KEM Saber also involves one final adder, a control unit, and a sign control unit.

For NTRU, the theoretical area-complexity of the proposed accelerator is listed as follows. Each PM units has N 12-bit registers, N adders, and N multipliers. Additional resources are allocated to the control unit and one final adder. Within NTRU, the number of multipliers and resources for PWMs depends on the multiplication method selected. For simplicity of discussion, we just use the proposed logic-reduced method discussed.

TABLE I: Area-Time Complexities for PM Accelerators

Complexity Comparison for Different PM Accelerators					
Scheme	#Mul.	#Add.	#Registers	Latency	Output Style
CROP[?]	N	N	N	2N-1	serial
HS CROP[?]	2N	2N	N	3N/2-1	serial
HS-I[20]	N	N	N	2N	parallel$*^1$
Dual CROP	2N	2N+1	2N	3N/2	serial
LFSR[?]*	N	N	N	2N	serial
Dual CROP*	2N	2N+1	2N	3N/2	serial

*: NTRU implementations.
$*^1$: Extra resources to transfer output into a serial form, mostly refer to MUXes and registers.
Latency refers to the loading and computational cycles

The latency defined includes processing time and loading time excluding the off loading time or the time to deliver all the outputs. The latency of the proposed accelerator is $3N/2$ cycles, i.e., N cycles to load A 's coefficients and $N/2$ cycles for computation. As seen in Table I, the Dual Crop based PM accelerator for Saber offers significant lower time-complexity when compared to both the original CROP design and the first high speed design (HS-I) proposed in [20]. It is also noted that when comparing out proposed design to the original high speed CROP design, the area-complexity of the proposed accelerator is reduced, however, there are more registers involved with the PM which offers further opportunity for area reduction. For NTRU, while the area complexity has been increased in comparison to [**?**], the time complexity is greatly reduced. Additionally, the area complexity of the NTRU design is comparable to the high speed designs for Saber.

Implementation Results: The proposed designs are coded in VHDL and their functions have been verified through software simulation. The implementation results are also obtained through Xilinx Vivado 2020.2 on the targeted device of Xilinx FPGA UltraScale+ XCZU9EG-FFVB1156-2. Table II reports the metrics of each implementation results, including the number of cycles, maximum clock frequency (MHz), dynamic power consumption (Watts), the number of LUTs, the number of FFs, the number of DSPs, and the number of slices/CLBs. The implementation results for the proposed PM accelerators of Saber and NTRU are listed in the same table along with a few existing designs. Note that the existing designs do not report the power consumption and the number of slices and thus we do not include them here.

TABLE II: Comparison of Different PM Hardware Accelerators for Saber and NTRU

Implementations of PM							
Scheme	FPGA	Cycles	Clk Freq.	LUT	FF	DSP	CLB
CROP[?]	CV	511	136.63	–	–	0	6,921
HS-CROP[?]	CV	383	103.62	–	–	0	14,579
HS-I [20]	U+	512	250	10,844	5,150	0	–
HS-II [20]	U+	387	250	15,625	14,136	128	–
Standard[13]	U+	512	250	13,869	5,150	0	–
Dual CROP	U+	384	310	22,127	7,841	0	3,427
Dual CROP$*^1$	U+	384	187	54,478	9,227	0	8,728
Dual CROP$*^2$	U+	384	171.9	66,138	9,240	0	11,166

Ultrascale+ (U+). Intel Cyclone V 5CSXFC6D6F31I7ES devices (CV). Clk Frequency measured in MHz, Power measured in W. Cycles of loading and multiplication(excludes outputting products)
$*^1$: NTRU design with reduced logic PWM with $n = 256$.
$*^2$: NTRU design with look up table based PWM with $n = 256$.

From Table II, one can see that the proposed Dual CROP based PM accelerator involves the highest operational frequency. Though the area usage of the proposed design, mainly the numbers of LUTs and FFs, is larger than the one in [20] (HS-I), the proposed accelerator has better time-complexity as well as the overall area-time efficiency. In comparison to (HS-II) [20], the area is reduced by removing DSP blocks, and implementing less FFs. Meanwhile, the implementation results also confirm the efficiency of the proposed PM accelerator for NTRU. Table II also depicts the comparison of two PWM methods utilized within the Dual CROP PM for NTRU. It is shown that the reduced logic method of PWM offers a smaller area complexity and higher frequency.

VII. Conclusion

This work introduced a novel Dual CROP based method to implement the hardware accelerators in Saber and NTRU. The proposed method and architectural details are detailed provided. Implementation results are also given to confirm the efficiency of the proposed work.

References

[1] V. Bhatia and K. R. Ramkumar, "An Efficient Quantum Computing technique for cracking RSA using Shor's Algorithm," *2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA)*, pp. 89–94, 2020, (*doi: 10.1109/ICCCA49541.2020.9250806*).

[2] W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Symp. Founda. of Computer Science*, 1994, pp. 124–134.

[3] J. X. et al., "Special session: The recent advance of hardware implementation of post-quantum cryptography," in *IEEE VTS*, 2020, pp. 1–10.

[4] D. Micciancio, "Lattice-based cryptography," in *Encyclopedia of Cryptography Security*, 2011.

[5] "Post-quantum cryptography round 3 submissions," April 2021. [Online]. Available: https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submission

[6] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," *J. ACM*, vol. 60, no. 6, nov 2013. [Online]. Available: https://doi.org/10.1145/2535925

[7] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, sep 2009. [Online]. Available: https://doi.org/10.1145/1568318.1568324

[8] L. D. E. K. T. L. V. L. J. M. S. P. S. G. S. D. S. Roberto Avanzi, Joppe Bos, "Crystals-kyber algorithm specifications and supporting documentation (version 3.0)," 2020.

[9] I. T. L. Computer Security Division, "Round 3 submissions - post-quantum cryptography: Csrc," https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions.

[10] J. H. A. H. J. R. J. M. S. T. S. P. S. W. W. K. X. T. Y. Z. Z. Cong Chen, Oussama Danba, "Ntru algorithm specifications and supporting documentation," September 2020.

[11] C. Peikert, "A decade of lattice cryptography," *Found. Trends Theor. Comput. Sci.*, vol. 10, no. 4, pp. 283–424, mar 2016. [Online]. Available: https://doi.org/10.1561/0400000074

[12] A. Langlois and D. Stehle, "Worst-case to average-case reductions for module lattices," Cryptology ePrint Archive, Report 2012/090, 2012, https://ia.cr/2012/090.

[13] S. Sinha Roy and A. Basso, "High-speed instruction-set coprocessor for lattice-based key encapsulation mechanism: Saber in hardware," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 4, pp. 443–466, Aug. 2020. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/8690

[14] A. K. et al., "Saber on arm cca-secure module lattice-based key encapsulation on arm," in *IACR Cryptology ePrint*, 2018.

[15] J. M. et al., "Time-memory trade-off in toom-cook multiplication: an application to module-lattice based cryptography," in *IACR TCHES*, vol. 2020, no. 2, 2020, pp. 222–244.

[16] J. Maria Bermudo Mera, F. Turan, A. Karmakar, S. Sinha Roy, and I. Verbauwhede, "Compact domain-specific co-processor for accelerating module lattice-based kem," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[17] V. D. et al., "Implementing and benchmarking three lattice-based post-quantum cryptography algorithms using software/hardware codesign," in *FPT 2019*, 2019, pp. 206–214.

[18] J. Pollard, "The fast fourier transform in a finite field," in *Mathematics of computation*, vol. 25, no. 114, 1971, pp. 365–374.

[19] Y. Zhu, M. Zhu, B. Yang, W. Zhu, C. Deng, C. Chen, S. Wei, and L. Liu, "A high-performance hardware implementation of saber based on karatsuba algorithm," Cryptology ePrint Archive, Report 2020/1037, 2020, https://ia.cr/2020/1037.

[20] A. Basso and S. S. Roy, "Optimized polynomial multiplier architectures for post-quantum kem saber," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 1285–1290.

[21] T. F. et al., "Risq-v: Tightly coupled risc-v accelerators for post-quantum cryptography," in *Cryptology ePrint*, 2020.

[22] P. Meher and X. Lou, "Low-latency, low-area, and scalable systolic-like modular multipliers for 2m based on irreducible all-one polynomials," in *IEEE TCAS-I*, vol. 64, 2017, pp. 399–408.

[23] J. L. Ima, "Lfsr-based bit-serial 2m multipliers using irreducible trinomials," in *IEEE Trans. Computers*, 2020(early access).

[24] P. Meher, "Systolic and super-systolic multipliers for finite field 2m based on irreducible trinomials," in *IEEE TVLSI*, vol. 55, no. 2, 2006, pp. 441–449.

[25] S. N. et al., "Low-power design for a digit-serial polynomial basis finite field multiplier using factoring technique," in *IEEE TVLS-I*, vol. 25, no. 2, 2017, pp. 441–449.

[26] J. X. et al., "Efficient hardware implementation of finite field arithmetic ab + c over hybrid fields for post-quantum cryptography," in *IEEE Trans. Emerging Topics In Computing*, 2021(accepted), pp. 1–6.

[27] C. P. A. Banerjee and A. Rosen, "Pseudorandom functions and lattices," in *In EUROCRYPT 2012*, 2012, pp. 719–737.

[28] D. H. et al., "A modular analysis of the fujisaki-okamoto transformation," in *15th International Conference Theory of Cryptography Proceedings Part I*, vol. 10677, 2017, pp. 341–371.