A Lightweight and Efficient Encryption/Decryption Coprocessor for RLWE-based Cryptography

Yushu Yang, Zihang Wang, Jianfei Wang, Jia Hou, Yang Su, Chen Yang

***Abstract*—Lattice-based cryptography has experienced significant advancements in recent years due to its versatility and simplicity. The ring learning with errors (RLWE) problem is widely adopted in lattice-based cryptography. However, the polynomial multiplication is the performance bottleneck of RLWE-based cryptography, which requires further examination. In this brief, a lightweight and efficient encryption/decryption coprocessor for RLWE-based cryptography is proposed. The time complexity of the Schoolbook polynomial multiplication (SPM) is reduced from n^2 to $n^2/8$ by enhancing multiplication parallelism. Moreover, an optimized structure for the Compressed cumulative distribution table (CDT) Gaussian sampler is proposed, resulting in 22.2% reduction in storage resource. The proposed SPM structure demonstrates a 2.3× performance speedup and 2.7× hardware efficiency for the encryption core, compared with state-of-the-art SPM accelerators. Additionally, it achieves a 2.4× performance speedup and 3.2× improvements on hardware efficiency for the decryption core.**

***Index Terms*—Schoolbook polynomial multiplication (SPM), ring learning with errors (RLWE), Saber, FPGA.**

I. Introduction

With the continuous advancement of quantum computing, efficient quantum algorithms have emerged that can overcome the mathematical challenges posed by mainstream RSA, ECC, and other public key cryptosystems, rendering them no longer secure. Consequently, in the post-quantum era, there is an urgent need to develop more robust cryptosystems. Lattice-based cryptographic schemes are gaining prominence as one of the leading alternatives. The expansion of security parameters in the learning with errors (LWE) public-key cryptographic scheme leads to a rapid increase in the dimensions of both public and private keys. Consequently, there is a growing emphasis on investigating a more resource-efficient cryptographic scheme. Currently, the ring binary learning with error (Ring-Bin LWE) [1], [2] cryptographic scheme and the ring learning with errors (RLWE) cryptographic scheme are deemed appropriate for lightweight devices. There are many studies focusing on cryptographic schemes with RLWE assumption [3], [4]. These schemes hold promising application potentials in IoT devices. Furthermore, the RLWE assumption can be extended to the module learning with errors (MLWE) assumption.

Polynomial multiplication directly impacting the overall computational efficiency of the cryptosystem. Typically, polynomial multiplication is implemented through either Schoolbook or number theoretic transform (NTT) algorithms. The NTT algorithm exhibits low complexity and can significantly enhance the speed of polynomial multiplication; however, it is constrained by parameter selection and its intricate nature renders it unsuitable for lightweight devices. On the other hand, the Schoolbook algorithm boasts simplicity in implementation and minimal resource consumption, making it well-suited for resource-constrained devices [5], [6], [7].

The Schoolbook polynomial multiplication (SPM) has been extensively studied in literatures [6], [8]. However, previous research has identified certain limitations that need to be further addressed. Specifically, the work [6] achieved a lightweight structure of the Schoolbook algorithm, but it suffered from slow calculation speed. Conversely, the work [8] utilized 256 multiply accumulate (MAC) units to achieve high parallelism for SPM, but this approach consumed substantial circuit resources. Therefore, while SPM is widely employed for its versatility, existing research primarily focuses on either high performance or lightweight structure, without effectively addressing the trade-off between performance and resource.

This brief focuses on Schoolbook algorithm and cumulative distribution table (CDT) Gaussian sampler. A lightweight and efficient RLWE cipher processor based on this algorithm has been developed. The time complexity of the SPM is reduced from n^2 to $n^2/8$ by enhancing multiplication parallelism. The proposed structure utilizes Look-Up Table (LUT) and Flip-Flop (FF) resources instead of Digital Signal Processing (DSP) to implement a multiplication unit, effectively conserving computational resources. Furthermore, CDT Gaussian sampler is employed for key generation, and redundant data within the classical circuit structure is further compressed, resulting in significant savings in storage resources [9]. Consequently, this approach achieves a well-balanced trade-off between performance and resource utilization.

II. Algorithm Analysis and Improvements

A. The Encryption and Decryption of RLWE

The process of key generation, encryption and decryption is shown in Algorithm 1. In Algorithm 1, the inputs consist of a public key, a secret key that has undergone Gaussian sampling by CDT, and a plaintext vector. Upon completion of the

This work was supported in part by the National Natural Science Found ation of China under Grant 62176206, and in part by the National Key R& D Program of China under Grant 2023YFB4403500.(Corresponding author : Chen Yang.)

Yushu Yang, Zihang Wang, Jianfei Wang, Jia Hou and Chen Yang, are with the School of Microelectronics, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China (email: {437139, zihangw, jfwang, hjwst0314 }@stu.xjtu.edu.cn; chyang00@xjtu.edu.cn).

Yang Su is with the School of Cryptography Engineering, Engineering University of People's Armed Police, Xi'an 710086, China (e-mail: wj_suyang@126.com).

Algorithm 1: The encryption and decryption of RLWE

Input: Public key: $\boldsymbol{p} \leftarrow R$ and $\boldsymbol{a} \leftarrow R$;
Secret key: $\boldsymbol{r2} \leftarrow D\sigma$; Plaintext: $\boldsymbol{m} \leftarrow \{0,1\}^n$.
Output: Decrypted plaintext: $\boldsymbol{m}'$.
1: Choose the noise terms $\boldsymbol{e1}, \boldsymbol{e2}, \boldsymbol{e3} \leftarrow \boldsymbol{D\sigma}$;
2: Encode plaintext $\bar{\boldsymbol{m}} = \boldsymbol{ENCODE}(\boldsymbol{m})$;
3: Encryption $\boldsymbol{c1} = \boldsymbol{ae1} + \boldsymbol{e2}, \boldsymbol{c2} = \boldsymbol{pe1} + \boldsymbol{e3} + \bar{\boldsymbol{m}}$;
4: Decrypting $\boldsymbol{c} = \boldsymbol{c1} \times \boldsymbol{r2} + \boldsymbol{c2}$;
5: Decode plaintext $\boldsymbol{m}' = \boldsymbol{DECODE}(\boldsymbol{c})$;
6: return $\boldsymbol{m}'$.

Algorithm 2: The proposed CDT sampling algorithm

Input: Random numbers: $\boldsymbol{r}$, table of functions: $\boldsymbol{Freq}(\boldsymbol{x})$.
Output: Gauss value: $\boldsymbol{value}$.
1: $min = 0, cur = 16, \max = 24$.
2: **while** $(min + 1 != max)$ **do**
3: **if** $r > Freq(cur)$ **then**
4: $min = cur$;
5: $cur = (max - cur)/2 + cur$;
6: **else then**
7: $max = cur$;
8: $cur = (max - min)/2 + min$;
9: **end if**
10: **end while**
11: $\boldsymbol{value} = \{r[0], min + 1\}$;
12: **return** $\boldsymbol{value}$.

TABLE I
RESOURCE IMPROVEMENTS OF CDT SAMPLER

set (n, q, s, λ) [a]	Original FF	Optimized FF	Reduced Ratio
(256,4096,8.35,36)	864	672	**22.2%**
(320,4093,8.0,64)	1248	912	**26.9%**
(256,7681,11.31,40)	1280	984	**23.1%**
(192,4093,8.87,40)	1080	784	**27.4%**

[a] The parameter $n, q, s = \sqrt{2\pi}\delta, \lambda$ represents the polynomial dimension, modulus of the polynomial, δ is the standard deviation, bit width of the Gaussian sampled values.

aforementioned calculation process, the decrypted plaintext vector m' is produced as an output.

In Algorithm 1, the multibit plaintext m is transformed into a polynomial $\bar{m}$ by the encoding module, as follows (1).

$$\bar{m} = ENCODE(m) = \frac{(q-1)}{2}m \tag{1}$$

The plaintext c after decryption is transformed into bit plaintext m' by the decoding unit, as shown in Equation (2) below.

$$m' = DECODE(c) = \left(\frac{q-1}{4}\right) \le \bar{m} \le \frac{3(q-1)}{4} \ ? \ 1 : 0 \tag{2}$$

B. Improved Gaussian Discrete Sampling Algorithm

Current discrete Gaussian sampling algorithms mainly include the rejection sampling method [10], Bernoulli sampling method, Knuth-Yao sampling method, and CDT Gaussian sampling method. The CDT Gaussian sampling method is widely preferred for its straightforward hardware implementation. The algorithm flow is illustrated in Algorithm 2. In the Algorithm 2, a 36-bit random number r is generated using a random number generator. If r satisfies expression (3), then the sampled value $x = k$.

$$Freq(k-1) \le r \le Freq(k) \tag{3}$$

III. THE PROPOSED ENCRYPTION/DECRYPTION COPROCESSOR FOR RLWE-BASED CRYPTOGRAPHY

A. RLWE-based Encryption and Decryption Structure

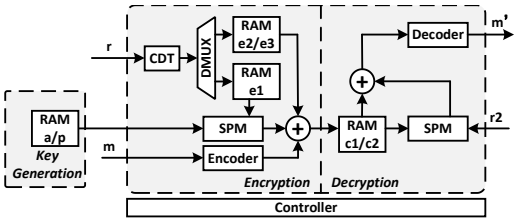

Fig. 1. The proposed Encryption/decryption core structure.

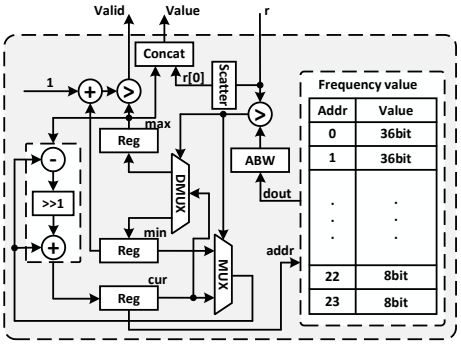

Fig. 2. The proposed CDT Gaussian sampler structure.

The proposed structure of the encryption/decryption core is illustrated in Fig. 1. The key generation module is used to store the public key a/p generated offline. The proposed CDT Gaussian sampler is utilized for the generation of the secret key, with the sampled error term being stored across two BRAM blocks, each possessing a capacity of 8KB. Furthermore, a control unit is employed to precisely manage the read and write operations of the BRAM.

The parameter selection proposed by work [11] introduced an enhanced parameter set $(256,4096,8.35)$ to achieving a medium level of security. The selected parameter set aims to ensure a medium level of security strength while minimizing the utilization of hardware resources. To achieve a lightweight structure, a hardware-friendly parameter set $(256,4096,8.35)$ modulo 4096 is chosen. The proposed hardware architecture is easily extensible and can be adapted to other parameter sets.

B. Lightweight CDT Sampler Structure

In previous research, the optimization of the CDT Gaussian sampler typically centered around reducing the bit width required for storing the secret key. The previous research [11] shows that the impact of reducing the bit width of the sampling value on the security level is within an acceptable range. This approach building on prior research utilizes the symmetry of the Gaussian distribution to limit the secret key within a range of $[0,23]$, thereby reducing its bit width from 12 to 5 bits, with an additional bit indicating its sign. Although this methodology results in a reduction in storage resources, the overall impact of this reduction is limited.

The CDT Gaussian sampler necessitates the storage of 24 cumulative distribution probability values, each comprising 36 bits. This requirement can significantly consume storage resources on lightweight devices. To address this, compression techniques are employed to eliminate redundant information within the probability values, enhancing the efficiency of storage resource utilization. The structure of the proposed

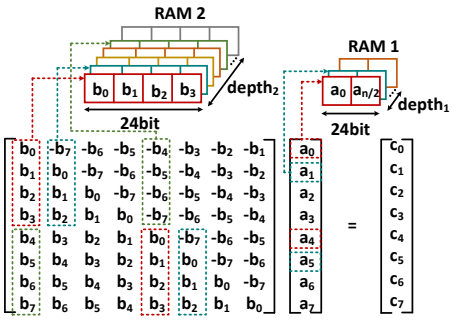

Fig. 3. The proposed high-parallel data read and store scheme.

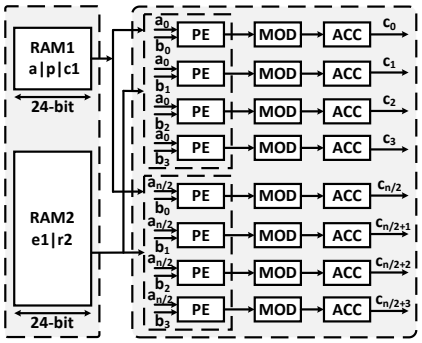

Fig. 4. The proposed efficient SPM structure.

Compressed CDT Gaussian sampler is depicted in Fig 2.

Based on the proposed data compression method, the CDT sampler scheme is proposed in Fig 2, which mainly consists of the following four steps, and the specific process is shown in Algorithm 2: (1) The input random number r is contrasted with the data read out from the probabilistic pre-storage table (corresponding Algorithm 2 line 3). (2) Different comparison results enable DMUX and MUX selectors, changing the values of parameters max, min, cur (corresponding Algorithm 2 line 4-8). (3) The data valid signal is pulled up only when $min + 1 == max$ (corresponding Algorithm 2 line 2). (4) Output the correct value $value$ (corresponding Algorithm 2 line 11).

The selected parameter for the Gaussian sampler determines that as the probability approaches 1, the function $Freq(x)$ exhibits an increasing number of ones in the high bits. Specifically, the focus is on compressing consecutive sequences of four ones ($(F)_H = (1111)_B$). The optimization method proposed in this brief has been applied to various parameter sets, and the resulting optimization results are presented in Table I. These results indicate that the approach can achieve a maximum saving of 27.4% in FF resources. The data recovery module (ABW) is capable of restoring the compressed probability values back to their original standard bit width. The proposed CDT Gaussian sampler consumes 262 LUT resources, 672 FF resources, and 135 mW of power. Compared to the original Gaussian sampler, the proposed CDT Gaussian sampler reduces hardware resource usage by 26.6% and power consumption by 6.25%. The proposed optimization method reduces the hardware resource consumption of the CDT Gaussian sampler without affecting the computation speed of the encryption and decryption core, thereby improving the overall hardware efficiency. The CDT Gaussian sampler exhibits excellent scalability, as it can be adapted to various parameter sets by adjusting the size of the store table.

Algorithm 3: The proposed parallel and efficient SPM algorithm

Input: $A(x) = \sum_{i=0}^{n-1} a_i x^i \in Z_q^n[x], B(x) = \sum_{i=0}^{n-1} s_i x^i \in Z_q^n[x]$
q prime modular, n dimension;
Output: $C(x) \in Z_q^n[x]$.
for $(i = 0; i < n/2 - 4; i = i + 4)$ **do** // loop1
  for $(j = 0; j < n; j = j + 1)$ **do** // loop2
    if $j < n/2$ **then** // the transformed matrix A
      $A0 = a[j]$;
      $A1 = a[(j + n/2) mod\ n]$;
    else then
      $A0 = -a[j]$;
      $A1 = a[(j + n/2) mod\ n]$;
    end if
    $B0 = b[(i - j)\ mod\ n]$; // the transformed matrix B
    $B1 = b[(i - j + 1)\ mod\ n]$;
    $B2 = b[(i - j + 2)\ mod\ n]$;
    $B3 = b[(i - j + 3)\ mod\ n]$;
    $Temp\_c1 = (A0 * B0)\ mod\ q$;
    $Temp\_c2 = (A0 * B1)\ mod\ q$;
    $Temp\_c3 = (A0 * B2)\ mod\ q$;
    $Temp\_c4 = (A0 * B3)\ mod\ q$;
    $Temp\_c5 = (A1 * B0)\ mod\ q$;
    $Temp\_c6 = (A1 * B1)\ mod\ q$;
    $Temp\_c7 = (A1 * B2)\ mod\ q$;
    $Temp\_c8 = (A1 * B3)\ mod\ q$;
    $C[i] = Temp\_c1 + C[i]$; // the results of the accumulation
    $C[i + 1] = Temp\_c2 + C[i + 1]$;
    $C[i + 2] = Temp\_c3 + C[i + 2]$;
    $C[i + 3] = Temp\_c4 + C[i + 3]$;
    $C[i + n/2] = Temp\_c5 + C[i + n/2]$;
    $C[i + n/2 + 1] = Temp\_c6 + C[i + n/2 + 1]$;
    $C[i + n/2 + 2] = Temp\_c7 + C[i + n/2 + 2]$;
    $C[i + n/2 + 3] = Temp\_c8 + C[i + n/2 + 3]$;
  end for
end for
return $C(x)$

C. Parallel Data Reading Scheme

In the implemented structure, eight parallel multiplication units are employed to concurrently execute eight 12-bit×5-bit multiplications. Based on the existing structure, a parallel data reading scheme is proposed.

The matrix of $n = 8$ is utilized as an illustrative example for enhanced comprehension. The data reading scheme for polynomial multiplication in an eight-dimensional matrix is depicted in Fig. 3, where $depth_1$ and $depth_2$ signify the RAM depth. In this specific example, $depth_1$ is set to 4 and $depth_2$ to 8. The polynomial computation implemented conforms to the formulation provided in expression (4).

$$\boldsymbol{c} = \boldsymbol{ab} = \left[\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_i\, b_j x^{i+j}\right] mod\ (x^n + 1) \tag{4}$$

In the data reading scheme illustrated in Fig. 3, an inverse relationship is observed between the $b(x)$ coefficient obtained from rows 0-3 and 4-7 during the fifth cycle (corresponding to the green box). If we were to adopt this aforementioned data reading scheme, the complete storage of the matrix B would be necessary, resulting in increased consumption of storage resources. Therefore, the previous storage scheme is optimized by extracting the negative sign located in the upper right corner of the matrix B. This method ensures symmetry between the upper and lower portions of matrix B. The sign bits are

TABLE II
COMPARISON OF IMPLEMENTATION RESULTS BASED ON FPGA

Design	Platform	Type	Freq. (MHz)	Cycle	Latency (μs)	LUT / FF / Slice [e]	ENS [d]	DSP [a] / BRAM [b]	Thr. [c] (Kbps)	TPS (Kbps/Slice)
2018 APCCAS SPM [13]	Kintex-7	Enc	288	131604	457	1098 / 407 / 337	439.4	1 / 0	560.23	1.27
		Dec	288	65802	228	609 / 318 / 182	284.4	1 / 0	1120.45	3.94
2019 TVLSI SPM [6]	Kintex-7	Enc	304.69	69654	229	898 / 815 / 303	573.4	1 / 3	1119.83	1.95
		Dec	303.40	34436	114	635 / 190 / 194	352.4	1 / 1	2255.50	6.40
2020 TCAS-II SPM [5]	Kintex-7	Enc	280	35478	127	1254 / 1046 / 402	718.8	2 / 2	2020.40	2.81
		Dec	292	17732	61	722 / 558 / 249	453.8	2 / 0	4215.65	9.29
2021 DAC SPM [14]	Artix-7	Dec	100	19471	195	541 / 301/ 173*	173	0 / 0	1314.78	7.60
2022 ISCAS SPM [7]	Artix-7	Dec	130	16384	126	561 / 302/ 178*	382.8	2 / 0	2031.25	5.31
2022 TCAS-II SPM [15]	Kintex-7	Dec	328.9	8204	25	1301 / 758 / 461	870.6	4 / 0	10263.1	11.79
Ours (SPM4)	**Kintex-7**	**Enc**	**328**	**34761**	**106**	**779 / 1298 / 357**	**469**	**0 / 2**	**2415.09**	**5.15**
		Dec	**332**	**16480**	**50**	**507 / 554 / 197**	**253**	**0 / 1**	**5120**	**20.24**
Ours (SPM8)	**Kintex-7**	**Enc**	**327**	**18240**	**56**	**1142 / 1642 / 487**	**599**	**0 / 2**	**4589.47**	**7.66**
		Dec	**329**	**8224**	**25**	**879 / 902 / 289**	**345**	**0 / 1**	**10241.2**	**29.68**
Ours (SPM16)	**Kintex-7**	**Enc**	**296**	**9772**	**33**	**1974 / 2698 / 831**	**943**	**0 / 2**	**7757.58**	**8.22**
		Dec	**305**	**4186**	**14**	**1698 / 1958 / 669**	**725**	**0 / 1**	**18285.7**	**25.22**

[a] One DSP is equivalent to 102.4 Slices according to work [5]. [b] One 8K BRAM is equivalent to 56 Slices according to work [5].
[c] Throughput = Degree/Latency according to work [5]. [d] ENS = Slices + BRAMs × 56 + DSPs × 102.4 according to work [5].
[e] In instances where the Slice count is not provided (marked by *), Slice is approximated by LUT × 0.25 + FF × 0.125 according to work [16].

generated by the control unit as input to the MOD unit. According to Algorithm 3, when the index j of loop 2 satisfies $j \geq n/2$, the control unit generates the sign bit for parameter a (corresponding Algorithm 3 line 3-8).

In the context of matrix A, it is imperative to execute a corresponding transformation on matrix A. The post-transformed matrix A can be expressed as follows:

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & -a_4 & -a_5 & -a_6 & -a_7 \\ a_4 & a_5 & a_6 & a_7 & a_0 & a_1 & a_2 & a_3 \end{bmatrix}^T \quad (5)$$

Upon analysis of the matrix characteristics, it is determined that only the leftmost four columns of parameters need to be stored. Following the above data reading scheme, for 256-dimensional polynomial multiplication, only 128 parameter pairs $(\text{e.g.}, a_0a_{128}, \dots, a_{127}a_{255})$ need to be stored for the A matrix, corresponding to a memory $depth_1 = 128$. For the original B matrix, 508 parameter sets need to be stored. $(\text{e.g.}, -b_1-b_2-b_3-b_4, -b_2-b_3-b_4-b_5, \dots, b_{252}b_{253}b_{254}b_{255})$ After removing sign bits in the top-right quarter matrix, $(-b_1-b_2-b_3-b_4, \dots, -b_{128}-b_{129}-b_{130}-b_{131})$ consecutive 128 parameter groups do not need to be stored, the transformed B matrix only needs to store 380 parameter groups, the corresponding $depth_2 = 380$. Extending the conclusion to n-dimensional polynomial multiplication, it follows that $depth_1 = \frac{n}{2}$ and $depth_2 = \frac{3n}{2} - 4$. The proposed matrix transformation method can reduce the storage resource consumption of RAM2 by 25%.

D. Efficient Polynomial Multiplier Structure

The proposed algorithm scheme with a parallel degree of 8 is derived from the aforementioned data reading scheme. The steps are summarized as follows.

1) During a single clock cycle, data is read from BRAM, including the coefficients of $a[j]$, $a[(j + n/2)\ mod\ n]$, $b[(i-j) mod\ n]$, $b[(i-j+1)\ mod\ n]$, $b[(i-j+2)\ mod\ n]$, $b[(i-j+3)\ mod\ n]$.
2) Perform 12-bit×5-bit multiplication, calculate 8 partial products, and modulo the partial products.
3) Accumulate the partial products modulo.
4) When the loop is finished, output all the results.

Based on Algorithm 3, an efficient and lightweight Schoolbook polynomial multiplication structure is proposed, as illustrated in Fig. 4. The proposed SPM structure consists of eight PE units, organized into two groups of four PE units each. Each PE unit inputs one coefficient a and one coefficient b. According to the data reading scheme illustrated in Fig. 3, each cycle involves reading a set of B matrix data (e.g., $b_0b_1b_2b_3$) and a set of A matrix data (e.g., $a_0a_4, a_1a_5, \dots$). One group of PE units receives the four b coefficients, while the two a coefficients are distributed across the two groups of PE units.

The proposed SPM structure is designed to support a variety of commonly used parameter sets. When changing the parameter set, the hardware architecture of the SPM only requires modification of the MOD unit structure to accommodate the different number of modules. This adjustment has minimal impact on the overall resource consumption. Consequently, the SPM structure is capable of supporting a wide range of different parameters.

IV. IMPLEMENTATION RESULTS AND COMPARISON

The proposed structure was synthesized and implemented on the Xilinx Kintex-7 FPGA platform. Table II provides detailed information on the hardware implementation results, including resource consumption, performance metrics (frequency, latency, throughput), and the throughput per slice (TPS) as a representation of hardware efficiency. We show three different parallelism structures in Table II, where the SPM with eight parallelism demonstrates better hardware efficiency. Therefore, the eight-parallelism structure is used for comparison with other structure. For fair comparison, FPGA BRAMs and DSPs are

converted to an equivalent number of slices (ENS), thus higher ENS value represents a larger area occupied by the structure. TPS is an efficiency indicator, and higher TPS value represents a structure with better resource utilization.

The structure proposed in work [13] utilizes a parameter set with medium security strength (256,7681,11.31). The strategy outlined in work [5] maximizes the utilization of DSP resources by employing two DSPs to perform four multiplications simultaneously, thereby enhancing the calculation speed. On the one hand, the proposed structure computes eight multiplications in one cycle, which speeds up the computation. On the other hand, the CDT sampler is used to compress the redundant data when sampling data, and the multiplier unit composed of LUT and FF is used instead of DSP, which greatly reduces the equivalent slice of the structure and achieves better hardware efficiency. In this brief, the proposed SPM structure achieving a 2.3× performance speedup and 2.7× TPS improvements for the encryption core, compared with state-of-the-art solution described in work [5]. Additionally, it achieves a 2.4× performance speedup and 3.2× TPS improvements for the decryption core.

The structure presented in work [14] introduces a lightweight multiplier and incorporates a series of optimization strategies. This approach consumes 541 LUTs and 301 FFs on a small Artix-7 FPGA. In contrast, the proposed structure has higher parallelism, reduces the number of clock cycles by half, our proposed structure achieves a 7.8× performance speedup and 3.9× TPS improvements compared to work [14].

The proposed structure by work [7] introduces a lightweight polynomial multiplier for Schoolbook. The structure incorporates an effective multiplication strategy that computes four coefficients per cycle, along with a specially designed technology for loading multiplication operands in the compact multiplier. The proposed structure has a shorter critical path due to the influence of the selected parameters and has a faster clock frequency. The proposed structure achieves a 5.0× performance speedup and 5.6× TPS improvements compared to work [7].

The latest research [15] introduces a time-sharing approach that employs varying degrees of parallelism for polynomial multiplication. To ensure fair comparison, SPM8 was selected due to its parallelism level, which aligns with that of the proposed structure. The proposed structure uses multiplication units consisting of LUT and FF resources instead of DSPs, thus significantly reducing the ENS. The structure proposed reduces the Slice resources by 60.3% maintaining a comparable computation speed, resulting in 2.5× TPS improvements.

V. Conclusion

This brief presents a structure for a lightweight and efficient encryption/decryption coprocessor for RLWE-based cryptography. It utilizes the computational characteristics of polynomial multiplication to realize parallel design, reduces the iteration cycle to 1/8, and compresses the memory resources of CDT sampler, which effectively balances performance and resource utilization. The proposed structure outperforms other structures in terms of both performance and efficiency metrics.

VI. References

[1] Karim Shahbazi and Seok-Bum Ko, "Area and power efficient post-quantum cryptosystem for IoT resource-constrained devices," Microprocessors and Microsystems, vol. 84, 2021.

[2] K. Shahbazi and S. -B. Ko, "An Optimized Hardware Implementation of Modular Multiplication of Binary Ring LWE," *IEEE Transactions on Emerging Topics in Computing*, vol. 11, no. 3, pp. 817-821, 1 July-Sept. 2023.

[3] J. Wang, C. Yang, F. Zhang, Y. Meng, S. Xiang and Y. Su, "A High-Throughput Toom-Cook-4 Polynomial Multiplier for Lattice-Based Cryptography Using a Novel Winograd-Schoolbook Algorithm," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 1, pp. 359-372, Jan. 2024.

[4] H. Lee, H. Kwon and Y. Lee, "16.1 A 2.7-to-13.3μJ/boot/slot Flexible RNS-CKKS Processor in 28nm CMOS Technology for FHE-Based Privacy-Preserving Computing," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, San Francisco, CA, USA, 2024, pp. 296-298.

[5] Y. Zhang, C. Wang, D. E. S. Kundi, A. Khalid, M. O'Neill and W. Liu, "An Efficient and Parallel R-LWE Cryptoprocessor," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 5, pp. 886-890, May. 2020.

[6] W. Liu, S. Fan, A. Khalid, C. Rafferty and M. O'Neill, "Optimized Schoolbook Polynomial Multiplication for Compact Lattice-Based Cryptography on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 10, pp. 2459-2463, Oct. 2019.

[7] Y. Zhang, Y. Cui, Z. Ni, D. -E. -S. Kundi, D. Liu and W. Liu, "A Lightweight and Efficient Schoolbook Polynomial Multiplier for Saber," in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, Austin, TX, USA, 2022, pp. 2251-2255.

[8] S. S. 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, 2020.

[9] C. P. Rentería-Mejía and J. Velasco-Medina, "High-Throughput Ring-LWE Cryptoprocessors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 8, pp. 2332-2345, Aug. 2017.

[10] J. Howe, A. Khalid, C. Rafferty, F. Regazzoni and M. O'Neill, "On Practical Discrete Gaussian Samplers for Lattice-Based Cryptography," *IEEE Transactions on Computers*, vol. 67, no. 3, pp. 322-334, March. 2018.

[11] T. Pöppelmann and T. Güneysu, "Area optimization of lightweight lattice-based encryption on reconfigurable hardware," in *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, Melbourne, VIC, Australia, 2014, pp. 2796-2799.

[12] T. Pöppelmann and T. Güneysu, "Towards practical lattice-based public key encryption on reconfigurable hardware," in *Selected Areas in Cryptography-SAC*, pp. 68-85, Springer Berlin Heidelberg, 2014.

[13] S. Fan, W. Liu, J. Howe, A. Khalid and M. O'Neill, "Lightweight Hardware Implementation of R-LWE Lattice-Based Cryptography," in *IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, Chengdu, China, 2018, pp. 403-406.

[14] A. Basso and S. S. Roy, "Optimized Polynomial Multiplier Architectures for Post-Quantum KEM Saber," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA, 2021, pp. 1285-1290.

[15] Y. A. Birgani, S. Timarchi and A. Khalid, "Area-Time-Efficient Scalable Schoolbook Polynomial Multiplier for Lattice-Based Cryptography," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 12, pp. 5079-5083, Dec. 2022.

[16] M. Li, J. Tian, X. Hu and Z. Wang, "Reconfigurable and High-Efficiency Polynomial Multiplication Accelerator for CRYSTALS-Kyber," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 8, pp. 2540-2551, Aug. 2023.

[17] J. Wang, C. Yang, F. Zhang, J. Hou, Y. Meng, S. Xiang and Y. Su, "A High-Throughput and Scalable Schoolbook Polynomial Multiplier for Accelerating Saber on FPGA Using a Novel Winograd-Based Architecture," *IEEE Transactions on Circuits and Systems II: Express Briefs (Early Access)*, Dec. 2023.

[18] Y. Su, B. -L. Yang, C. Yang, Z. -P. Yang and Y. -W. Liu, "A Highly Unified Reconfigurable Multicore Architecture to Speed Up NTT/INTT for Homomorphic Polynomial Multiplication," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 8, pp. 993-1006, Aug. 2022.