Lightweight Hardware Implementation of Binary Ring-LWE PQC Accelerator

Benjamin J. Lucas, Ali Alwan, Marion Murzello, Yazheng Tu, Pengzhou He, Andrew J. Schwartz, David Guevara, Ujjwal Guin, *Member, IEEE*, Kyle Juretus, *Member, IEEE*, and Jiafeng Xie, *Senior Member, IEEE*

Abstract—Significant innovation has been made in the development of public-key cryptography that is able to withstand quantum attacks, known as post-quantum cryptography (PQC). This paper focuses on the development of an efficient PQC hardware implementation. Specifically, an implementation of the binary Ring-learning-with-errors (BRLWE)-based encryption scheme, a promising lightweight PQC suitable for resource-constrained applications, is proposed. The paper first develops the mathematical formulation to present the proposed algorithmic process. The corresponding hardware accelerators are then described in detail. Finally, comparisons with previous implementations are provided to demonstrate the superior performance of the proposed design. For instance, the proposed low-complexity accelerator has 34.7% less area-delay product (ADP) than the state-of-the-art design for $n = 256$ in the field-programmable gate array (FPGA) platform. Apart from the efficiency of the hardware architectures, the proposed design also has a complete input/output processing setup, and thus is feasible for emerging lightweight applications.

Index Terms—Binary Ring-LWE, complete processing setup, hardware design, lightweight post-quantum cryptography

1 Introduction

A majority of the current public-key cryptosystems, such as Rivest Shamir Adleman (RSA) and Elliptic Curve Cryptography (ECC), have been proven to be insecure against quantum attacks [1], [2]. With the view that large-scale quantum computers will be available in the next 12-15 years, the National Institute of Science and Technology (NIST) has started the standardization process of post-quantum cryptography (PQC) to neutralize quantum attacks [3]. Among all the proposed schemes, lattice-based PQC is one of the most promising categories due to its small implementation complexity and strong security proof [3], [4], [5].

Many lattice-based schemes are based on the learning-with-errors (LWE) problem (standard LWE) or its variants such as Ring-LWE (ideal LWE) [4]. The Ring-LWE based scheme uses the arithmetic operation over ring $\mathbb{Z}_q/(x^n+1)$ (smaller computational complexity than the standard LWE) and hence is widely studied in [5], [6], [7], [8], [9], [10]. A lightweight variant of Ring-LWE, known as binary Ring-LWE (BRLWE), was introduced in [11] to target resource-constrained applications. The BRLWE-based scheme deploys binary errors to achieve smaller complexity than the standard Ring-LWE based scheme, but is able to retain sufficient security for lightweight applications.

Prior Work. The ability of BRLWE-based scheme to achieve a lightweight implementation has led to a variety of previous implementations. The first software implementation of the BRLWE-based PQC was carried out in [11]. An efficient hardware structure for the BRLWE-based scheme decryption phase was proposed in [12]. A pair of high-speed and low-speed structures were then developed in [13]. A new high-speed BRLWE-based hardware structure was presented in [6]. Recently, a new compact hardware structure was proposed in [14]. Additionally, one high-speed BRLWE-based PQC hardware structure was presented in [15]. Another BRLWE-based PQC arithmetic circuit was proposed in [16]. A low-speed architecture was reported in [17]. A fault-resistant software implementation [18] and the design to detect faults [19] (based on [13]) have also been proposed. However, the specific focus of these implementations prevents them from being included in the comparison with the standard implementations.

While significant effort has been placed into BRLWE-based PQC implementations, the existing designs still require significant improvements to enhance usability. The existing structures, such as [13], [14], [15], (i) the proposed structure still involves a complicated hardware setup, or (ii) do not include the necessary input processing component for practical processing. An instance of issue (i) is observed in [13] when two sign inversion cells are used to execute sign related operations and an example of issue (ii) includes [14] assuming one input (2,048 bits when $n = 256$) is directly fed to an 8-bit n-to-1 MUX.

Contributions. This paper proposes a new lightweight BRLWE-based PQC hardware implementation with (i) an efficient hardware structure, and (ii) a complete processing setup, to address the limitations of prior work. We have made three stages of coherent interdependent efforts (main contributions):

- We have rigorously formulated the major arithmetic operation of the BRLWE-based scheme into the desired form to derive the proposed algorithmic operation.
- We have obtained the novel hardware accelerators from the proposed algorithmic operation based on efficient algorithm-architecture co-optimization techniques.
- We have implemented the design and compared the results with the competing solutions to demonstrate the superior performance of the proposed structures.

The rest of the paper is organized as follows. Section 2 provides the preliminary information. Formulation of the proposed algorithmic strategy is presented in Section 3. Proposed hardware accelerators/structures are described in Section 4. Complexity and comparison are presented in Section 5. Conclusions are provided in Section 6.

2 Preliminaries

The BRLWE-based PQC has three main phases [11], see Fig. 1.

- *Key generation*. Alice calculates $p = r_1 - a \cdot r_2$ and Bob gets p ($n\log_2 q$) as the public key, where a is a global parameter shared by Alice and Bob and r_1 and r_2 (the secret key) are randomly selected binary polynomials.
- *Encryption*. Bob uses errors (binary polynomials) e_1, e_2, and e_3 to produce ciphertext c_1 and c_2. $\tilde{m}$ is obtained by multiplying each coefficient of the input m with $q/2$. c_1 and c_2, both $n\log_2 q$-bit, are then sent to Alice.
- *Decryption*. Alice uses r_2 to recover the original message m through the decoding process $(c_1 r_2 + c_2)$, where a threshold decoder returns '1' if the coefficient is in the range of $(q/4, 3q/4)$, or '0'otherwise.

- *Benjamin J. Lucas, Ali Alwan, Marion Murzello, Yazheng Tu, Pengzhou He, Kyle Juretus, and Jiafeng Xie are with the Department of Electrical and Computer Engineering, Villanova University, Villanova, PA 19085 USA. E-mail: {blucas6, aalwan, mmurzell, ytu1, phe, kyle.juretus, jiafeng.xie}@villanova.edu.*
- *Andrew J. Schwartz and David Guevara are with the Department of Computer Science, Villanova University, Villanova, PA 19085 USA. E-mail: {aschwar7, dguevara}@villanova.edu.*
- *Ujjwal Guin is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849 USA. E-mail: ujjwal.guin@auburn.edu.*

Manuscript received 3 Feb. 2022; accepted 16 Feb. 2022. Date of publication 18 Mar. 2022; date of current version 31 Mar. 2022.
The work of Jiafeng Xie was supported by NSF under Grants SaTC-2020625 and in part by NIST-60NANB20D203. This work received funding from Villanova University's Falvey Memorial Library Scholarship Open Access Reserve (SOAR) Fund.
(Corresponding author: Jiafeng Xie.)
Digital Object Identifier no. 10.1109/LCA.2022.3160394

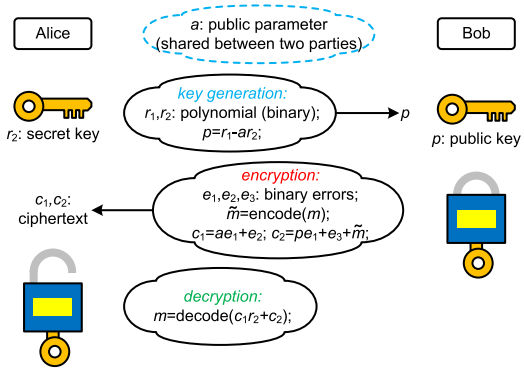

Fig. 1. The BRLWE-based encryption scheme, based on [11].

Remark. The authors in [13] proposed to use an inverted BRLWE-based scheme, where the integer coefficients are represented in the inverted range $(-\lfloor \frac{q}{2} \rfloor, \lfloor \frac{q}{2} \rfloor - 1)$, for the ease of using 2's complement. This paper also uses the same strategy.

Security of the BRLWE-Based Scheme. The BRLWE-based scheme is based on the average-case hardness of the BRLWE problem [11]. It is also shown that the BRLWE-based PQC achieves 73-bits and 140-bits quantum security for the parameters of $n = 256, q = 256$ and $n = 512, q = 256$, respectively [20], which appropriately fits lightweight applications [11], [12].

3 Proposed Algorithmic Operation

One can conclude that a typical arithmetic operation involved within the BRLWE-based PQC (Fig. 1) is a polynomial multiplication followed by two polynomial additions, represented as

$$W = GB + D + T \bmod f(x), \tag{1}$$

where $f(x) = x^n + 1$, $W = \sum_{i=0}^{n-1} w_i x^i$, $G = \sum_{i=0}^{n-1} g_i x^i$, $D = \sum_{i=0}^{n-1} d_i x^i$, $T = \sum_{i=0}^{n-1} t_i x^i$, $B = \sum_{i=0}^{n-1} b_i x^i$ ($t_i, b_i \in \{0, 1\}$), and w_i, g_i, and d_i are $\log_2 q$-bit integers over ring $\mathbb{Z}_q/(x^n + 1)$). While polynomial multiplication can be

$$GB \bmod f(x) = (Gb_0 + \cdots + Gb_{n-1}x^{n-1}) \bmod f(x), \tag{2}$$

which can be further derived as (substituting $x^n \equiv -1$)

$$\begin{aligned} GB \bmod f(x) &= Gb_0 \\ &+ (-g_{n-1} + g_0 x + \cdots + g_{n-2}x^{n-1})b_1 + \cdots \\ &+ (-g_1 - g_2 x - \cdots + g_0 x^{n-1})b_{n-1}, \end{aligned} \tag{3}$$

which is then transferred into (define $U = GB \bmod f(x)$)

$$\begin{aligned} U = \sum_{i=0}^{n-1} u_i x^i &= (g_0 b_0 - g_{n-1}b_1 - \cdots - g_1 b_{n-1}) \\ &+ (g_1 b_0 + g_0 b_1 - \cdots - g_2 b_{n-1})x + \cdots \\ &+ (g_{n-1}b_0 + g_{n-2}b_1 + \cdots + g_0 b_{n-1})x^{n-1}, \end{aligned} \tag{4}$$

where u_i is a $\log_2 q$-bit integer over ring. Then, we have

$$\begin{aligned} B^{(')} &= \{b_0, b_1, \ldots, b_{n-1}\}, \\ G^{(0)} &= \{g_0, -g_{n-1}, \ldots, -g_1\}, \\ G^{(1)} &= \{g_1, g_0, \ldots, -g_2\}, \quad \ldots \ \ldots \ \ldots \\ G^{(n-1)} &= \{g_{n-1}, g_{n-2}, \ldots, g_0\}, \end{aligned} \tag{5}$$

where $B_0^{(')} = b_0$, $B_1^{(')} = b_1$, $\ldots$, $B_{n-1}^{(')} = b_{n-1}$. Similarly, we have $G_0^{(0)} = g_0, \ldots, G_{n-1}^{(0)} = -g_1, \ldots, G_{n-1}^{(n-1)} = g_0$. Meanwhile, we can note that one can circularly shift the coefficients in $G^{(j)}$ (with the shifted far-right coefficient's sign inverted) to obtain $G^{(j+1)}$ $(0 \le j \le n-1)$.

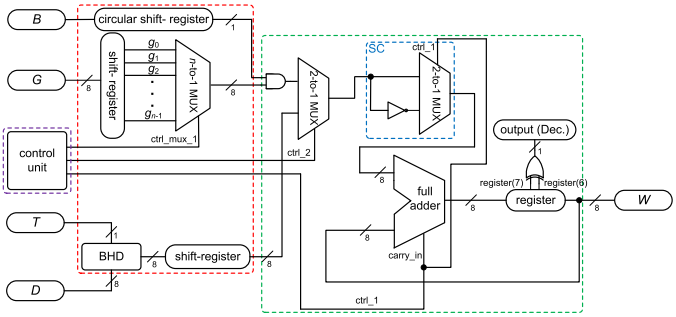

Fig. 2. Proposed hardware accelerator for BRLWE-based PQC. BHD: bit-level half-adder. Dec.: decryption. SC: sign control.

Then, we have

$$u_j = \sum_{i=0}^{n-1} G_i^{(j)} B_i^{(')}. \tag{6}$$

Then, we can have the proposed algorithmic operation as

Algorithm 1. Proposed Algorithmic Operation for BRLWE-Based Encryption Scheme

Input: G, B, T, and D (G and D are $\log_2 q$-bit integer polynomials; B and T are binary polynomials;
Output: $W = GB + D + T \bmod f(x)$ ($f(x) = x^n + 1$);
Initialization step
1 Obtain $v_k = d_k + t_k$ $(0 \le k \le n-1)$; // $V = \sum_{k=0}^{n-1} v_k x^k$ (v_k is $\log_2 q$-bit integer over ring)
2 Load (serially) G, B, and V into shift-registers;
3 $\overline{Z} = 0$;
Main step
4 **for** $j = n-1$ to 0 **do**
5 **for** $i = 0$ to $n-1$ **do**
6 $\overline{Z} = \overline{Z} + G_i^{(j)} B_i^{(')}$.
7 **end**
8 $w_j = \overline{Z} + v_j$;
9 Get $G^{(j-1)}$ from $G^{(j)}$; // until $G^{(0)}$ is obtained
10 **end**
Final step
11 Deliver the output w_j serially; // serially delivered W

Note that the decryption phase needs a decoder function.

Primary Novelty of Algorithm 1. The primary novelty of the proposed algorithmic operation, when compared with existing work [13], [14], [15], lies in two aspects: (i) the algorithm contains all the necessary operations for a complete BRLWE-based PQC accelerator design (while existing work assumes certain/partial operations, e.g., Line 2 and Line 9, are provided by outward resources); (ii) the algorithmic sequence has not been proposed before. Besides, this algorithm can be mapped with low-speed/high-speed structures (structural flexibility).

4 Proposed Hardware Accelerators

Following Algorithm 1, we can have the proposed accelerator for the BRLWE-based PQC as shown in Fig. 2. The major components of the proposed accelerator are: the input loading SRs (red dotted box), the major computation unit (green dotted box), and the control unit (purple dotted box). These components are described as below, where the bit-width of the coefficient (integer polynomial) is $\log_2 q = \log_2 256 = 8$.

Input loading SRs. Unlike prior designs (e.g., [14]) that assume certain input operands are directly fed to the structure, the proposed accelerator uses three SRs and one n-to-1 MUX to execute complete input processing related operations. Based on Lines 1-2 of

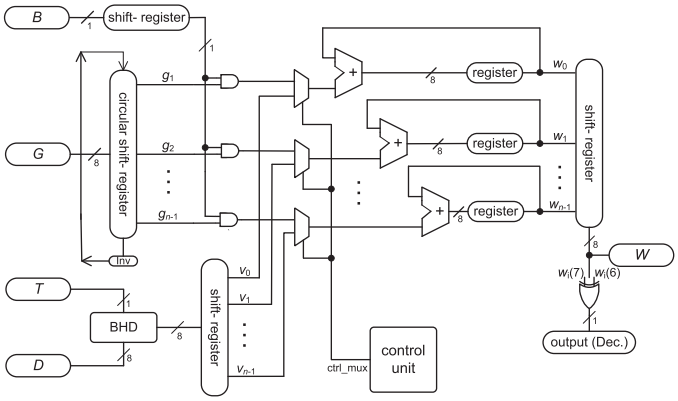

Fig. 3. The extended high-speed architecture version.

Algorithm 1, the coefficients of D and T are added together (through a bit-level half-adder (BHD)) to form a new polynomial to be serially loaded into the SR. While the coefficients of B are serially loaded into a circular shift-register (CSR) since the coefficients of B need to be delivered to the major computation unit one by one (repeats n times, Lines 4-6 of Algorithm 1). Lastly, a serial-in parallel-out SR is used to feed all the coefficients of G to an n-to-1 MUX to produce the correct output to the following computation unit (Line 6), as determined by the control signal "ctrl_mux_1". Note that the loadings of the SRs/CSR are all controlled by the control unit.

Major Computation Unit. The entry of the major computation unit is one AND cell (multiplication operation) followed by a 2-to-1 MUX (the control signal "ctrl_2" determines if the output of the AND cell or the coefficient of V will be delivered to the following sign control (SC) cell). The control signal to the MUX in the SC cell ("ctrl_1") is also connected to the carry_in of the adder in the accumulation cell (AC) according to the 2's complement number requirement when a negative sign is involved (see (5)). While AC executes the accumulation operation of Line 6 of Algorithm 1 in every n cycle and then adds one value from the SR (for V, Line 8) to produce one output value w_j. Note that a decoder (an XOR gate) connecting with the two most significant bits of the register is used to produce the output of the decryption phase.

Control Unit. A finite state machine (FSM) is used to constitute the control unit. In total five states are used in the FSM, including "reset", "load", "multiplication", "addition", and "done". In total $n(n+2)$ cycles are needed for the overall computation (the extra one cycle is for the output delivery).

Extended High-Speed Architecture. The hardware accelerator of Fig. 2 can be extended to obtain a high-speed version, as shown in Fig. 3. This proposed high-speed architecture is the parallel processing version of the accelerator shown in Fig. 2, i.e., Lines 5-7 of Algorithm 1 are executed in parallel. In this case, a CSR, with the help of an inverter (Inv cell in Fig. 3, according to the 2's complement requirement), is enough to obtain $G^{(j-1)}$ from $G^{(j)}$ based on Line 9 of Algorithm 1 (the n-to-1 MUX is no longer needed). After n cycles of accumulation, the coefficients of V are added with the accumulated results, respectively, to produce the desired output W. An output SR is then used to transfer W into a serial format. The control unit has a similar setup as that in Fig. 2 yet with updated cycles (the computation time is now $(n+1)$ cycles).

Novelty of the Proposed Accelerators. When comparing with existing designs of similar throughput, the proposed design has the following advantages. (i) Efficient structural layout, e.g., T and D are processed by one SR, and thus the proposed structure involves one less SR than [14]; additionally, the proposed accelerator has one less SC than [13]. (ii) Full control coverage, i.e., the control unit generates all necessary signals for operation of the accelerator while existing designs [13], [14], [15] do not provide related control signals for certain input operands (such as G or D). (iii) Complete input/output processing setup, i.e., the proposed accelerators operate appropriately once all the inputs have been loaded into respective SRs, while [13], [14], [15] assume certain operands (like G) are fed to the structure directly without considering the related SR usage (8-bit, size n).

TABLE 1
Comparison of Area-Time Complexities for the Proposed and Existing BRLWE-Based PQC Architectures

design	SR[1]	MUX[2]	SC	AC	latency[3]	CIOPS[4]?
[13][5]	1*	2	2	1	n^2+n	N
[14][5]	3*	1	1	1	n^2	N
Fig. 2[5]	3	1	1	1	n^2+2n	Y
[13][6]	1*	0	1	n	$n+1$	N
Fig. 3[6]	4	0	0	n	$n+1$	Y

[1] *Including different-size SRs and CSRs.*
[2] $\log_2 q$*-bit* n*-to-1 MUX.*
[3] *Not including input/output processing.*
[4] *Complete input/output processing setup.*
[5] *Low-complexity design.*
[6] *High-speed design.*
* *[13] only provides the decryption structure, where the low-speed one assumes two polynomials* ($2n \times \log_2 q$ *bits) are directly attached to two* n*-to-1 MUXes (two SRs are missing) and the high-speed one has three SRs missing. While [14] assumes one polynomial* ($n \times \log_2 q$ *bits) is directly attached to the* n*-to-1 MUX: one SR is missing.*

5 Complexity Analysis and Comparison

Complexity Analysis. The parameter setting of the BRLWE-based scheme is specified here: (i) n is the security level of the PQC scheme; (ii) the integer and binary polynomials have n number of $\log_2 q$-bit/1-bit coefficients, respectively.

The proposed BRLWE-based PQC accelerator (Fig. 2) has three SRs (one 1-bit CSR and two $\log_2 q$-bit SRs), one $\log_2 q$-bit n-to-1 MUX, one SC, and one AC. The latency time (not including the input/output processing time) is (n^2+2n) cycles. The proposed high-speed one (Fig. 3) has four SRs (one extra for output delivery) and n ACs (latency is now $(n+1)$ cycles).

We have also listed the area-time complexities of the proposed and those recently released designs (with similar throughput) in Table 1. It is shown that the proposed structure (Fig. 2) involves less complexities than the existing ones. The low-speed design of [13] involves two missing SRs. The design of [14] has one missing SR, and thus, it actually has one more SR than the proposed Fig. 2. While the existing high-speed architecture of [13] has three SR missing: two for input and one for output. Considering that the design of [13] only provides decryption structures, the proposed high-speed one is more efficient than [13] (both encryption and decryption operations are involved). Note that [15] has used a different structure, i.e., a lookup table (LUT)-like design, we thus do not include it here (will compare it based on the FPGA implementation results).

FPGA-Based Implementation & Comparison. The experimental setup for implementation & comparison is described below.

(i) We have coded the proposed structures (Figs. 2 and 3) in VHDL and used ModelSim to verify functionality (source code is available at[1]). We have also obtained implementation results on an Intel Stratix-V 5SGXMABN1F45C2 device (following [14], [15]) with Intel Quartus Prime 17.0.
(ii) The parameter settings of $n=256$ and $n=512$ with $q=256$ were used for the proposed structures (follow [11], [12], [13], [14], [15], [16], [17]).
(iii) The obtained implementation results, namely the number of adaptive logic modules (ALMs), maximum frequency in MHz (Fmax), latency cycles (not including loading/

1. https://www.ece.villanova.edu/ jxie02/lab/

TABLE 2
Comparison of FPGA Implementation Results for Various BRLWE-Based PQC Structures

design	#ALMs	Fmax	latency[1]	delay	ADP
low-complexity BRLWE-based PQC structures ($n = 256$)					
[13][2]	3,472	201.25	65,792	327	1,135,344
[14][4]	1,864	316.96	65,536	207	385,848
Fig. 2	846 (488*)	221.29	66,048	298	252,108
low-complexity BRLWE-based PQC structures ($n = 512$)					
[13][2]	6,901	171.32	262,656	1,533	10,579,233
[14][4]	3,551	296.65	262,144	884	3,139,084
Fig. 2	1,596 (876*)	203.87	263,168	1,291	2,060,436
high-speed BRLWE-based PQC structures ($n = 256$)					
[13][3]	5,734	369.14	257	696	3,990,834
[15]	4,495	321.03	258	0.804	3,613.980
Fig. 3	4,446 (3,999*)	379.22	257	0.678	3,014.388
high-speed BRLWE-based PQC structures ($n = 512$)					
[13][3]	11,470	336.36	513	1.525	17,491.750
[15]	9,038	317.06	514	1.621	14,650.598
Fig. 3	8,864 (8,002*)	327.98	513	1.564	13,863.296

[1] *Latency cycle is based on the decryption phase of the PQC scheme.*
[2] *Re-implemented results from [14].*
[3] *Results from [15].*
[4] *One polynomial ($n \times \log_2 q$ bits) is directly attached to a n-to-1 MUX (virtual pin based implementation).*
* *Area usage for the proposed structures excluding the input SR for G/D (follow [13], [14] that this part of resource is not included).*

delivery time), delay (μs, delay=critical-path×latency, where critical-path=1/Fmax), and area-delay product (ADP=#ALM×delay), are listed in Table 2. Note power consumption is not reported as a large portion of the FPGA power is static power.

(iv) Note that the design of [16] focuses more on the arithmetic circuit (the input processing components are not that much included). Meanwhile, the structure presented in [17] requires more actual SRs than [14] for complete input processing. We thus do not include these two designs for comparison.

(v) For a better understanding of the efficiency of the proposed design, we have listed the area usage for the proposed structures excluding the input SR for G/D (follow [13], [14]).

As shown in Table 2, the proposed structures involve significantly less area-time complexities than the existing ones. For instance, the proposed architectures (Figs. 2 and 3) have 34.7% and 16.6% less ADP than the recent designs of [14] and [15], respectively, for $n = 256$ (similar situation for $n = 512$).

Discussion. The existing designs, somehow, do enjoy the benefits of feeding/producing input/output directly to/from the structure, such as less resource usage (SRs) and simpler control signals (loading/delivery control is not required). However, this kind of design setup cannot be directly implemented on an FPGA device (large I/O), and the alternative virtual pin-based implementation cannot achieve optimized place & route with desired mapping performance. This issue, however, does not happen to the proposed architectures due to the complete input/output processing setup. Additionally, the proposed structures have better complexities than the competing designs, as analyzed in Table 1. These two factors contribute to the superior performance of the proposed accelerators.

Since this paper focuses on BRLWE-based PQC structures, we do not compare them with the existing regular Ring-LWE based designs (different errors/schemes/structures). Nevertheless, Ring-LWE based work, such as [10] and [21], represent important PQC implementations in the field. The Ring-LWE of [10] deploys DSPs and BRAMs for parallel-processing, and the AxRing-LWE based design in [21] uses an approximate method for resource-limited applications. Future work directions include side-channel attacks [22] and algorithm innovations.

6 Conclusion

This paper aims to deliver a lightweight hardware accelerator implementation for BRLWE-based PQC scheme. We have proposed two efficient hardware structures with complete input/output processing setup through three interdependent efforts. Overall, the proposed implementation demonstrates significantly better area-time complexities over existing ones. The lower complexity makes the proposed architecture well suited for deployment in emerging lightweight applications.

Acknowledgments

Fig. 2 is a senior design capstone project. Benjamin J. Lucas, Ali Alwan, Marion Murzello, and Yazheng Tu are contributed equally.

References

[1] D. Bernstein, "Introduction to post-quantum cryptography," in *Post-Quantum Cryptography*. Berlin, Germany: Springer, 2009.
[2] W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. Annu. Symp. Found. Comput. Sci.*, 1994, pp. 124–134.
[3] PQC round 3 submissions, 2020. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions
[4] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, 2009, Art. no. 34.
[5] V. Lyubashevsky *et al.*, "On ideal lattices and learning with errors over rings," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2010, pp. 1–23.
[6] J. Xie, K. Basu, K. Gaj, and U. Guin, "Special session: The recent advance of hardware implementation of post-quantum cryptography," in *Proc. IEEE 38th VLSI Test Symp.*, 2020, pp. 1–10.
[7] D. Liu, C. Zhang, H. Lin, Y. Chen, and M. Zhang, "A resource-efficient and side-channel secure hardware implementation of ring-LWE cryptographic processor," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 4, pp. 1474–1483, Apr. 2019.
[8] S. Roy *et al.*, "Compact Ring-LWE cryptoprocessor," in *Proc. Int. Workshop Cryptogr. Hardware Embedded Syst.*, 2014, pp. 371–391.
[9] S. Bian, M. Hiromoto, and T. Sato, "Filianore: Better multiplier architectures for LWE-based post-quantum key exchange," in *Proc. 56th ACM/IEEE Des. Automat. Conf.*, 2019, pp. 1–6.
[10] Y. Zhang, C. Wang, D. E. S. Kundi, A. Khalid, M. O'Neill, and W. Liu, "An efficient and parallel R-LWE cryptoprocessor," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 67, no. 5, pp. 886–890, May 2020.
[11] J. Buchmann *et al.*, "High-performance and lightweight lattice-based public-key encryption," in *Proc. 2nd ACM Int. Workshop IoT Privacy Trust Secur.*, 2016, pp. 1–8.
[12] A. Aysu, M. Orshansky, and M. Tiwari, "Binary Ring-LWE hardware with power side-channel countermeasures," in *Proc. Des. Automat. Test Eur. Conf. Exhib.*, 2018, pp. 1253–1258.
[13] S. Ebrahimi, S. Bayat-Sarmadi, and H. Mosanaei-Boorani, "Post-quantum cryptoprocessors optimized for edge and resource-constrained devices in IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5500–5507, Jun. 2019.
[14] P. He, U. Guin, and J. Xie, "Novel low-complexity polynomial multiplication over hybrid fields for efficient implementation of binary Ring-LWE post-quantum cryptography," *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 11, no. 2, pp. 383–394, Jun. 2021.
[15] J. Xie, P. He, and W. Wen, "Efficient implementation of finite field arithmetic for binary Ring-LWE post-quantum cryptography through a novel lookup-table-like method," in *Proc. 58th ACM/IEEE Des. Automat. Conf.*, 2021, pp. 1279–1284.
[16] J. Xie, P. He, X. M. Wang, and J. L. Imana, "Efficient hardware implementation of finite field arithmetic $AB + C$ for binary Ring-LWE based post-quantum cryptography," *IEEE Trans. Emerg. Top. Comput.*, to be published, doi: 10.1109/TETC.2021.3091982.
[17] K. Shahbazi and S.-B. Ko, "Area and power efficient post-quantum cryptosystem for IoT resource-constrained devices," *Microprocessors Microsyst.*, vol. 84, 2021, Art. no. 104280.
[18] S. Ebrahimi and S. Bayat-Sarmadi, "Lightweight and fault-resilient implementations of binary Ring-LWE for IoT devices," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6970–6978, Aug. 2020.
[19] A. Sarker, M. M. Kermani, and R. Azarderakhsh, "Fault detection architectures for inverted binary Ring-LWE construction benchmarked on FPGA," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 68, no. 4, pp. 1403–1407, Apr. 2021.
[20] F. Gopfert *et al.*, "A hybrid lattice basis reduction and quantum search attack on LWE," in *Proc. Int. Workshop Post-Quantum Cryptogr.*, 2017, pp. 184–202.
[21] D. Kundi, A. Khalid, S. Bian, C. Wang, M. O'Neill, and W. Liu, "AxRLWE: A multi-level approximate Ring-LWE co-processor for lightweight IoT applications," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2021.3122276.
[22] T. Schneider *et al.*, "ParTI – Towards combined hardware countermeasures against side-channel and fault-injection attacks," in *Proc. Annu. Int. Cryptol. Conf.*, 2016, pp. 302–332.