Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Integer Based Fully Homomorphic DSP Accelerator using Weighted-Number Theoretic Transform

Shakirah Hashim[1,*], Mohammed Benaissa[2]

[1] School of Computing Sciences, College of Computing, Informatics and Media, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia
[2] Department of Electronic and Electrical, University of Sheffield, Sheffield, United Kingdom

ARTICLE INFO

Article history:
Received 7 December 2022
Received in revised form 28 April 2023
Accepted 5 May 2023
Available online 22 May 2023

Keywords:
Fully Homomorphic Encryption;
Number-Theoretic Transform;
Montgomery Multiplication

ABSTRACT

Fully Homomorphic Encryption (FHE) has gained wide attention in cloud security as it allows computation on encrypted data. However, it requires a huge key size, resulting in impractical execution time. In this paper, we proposed an FHE hardware accelerator employing Weighted-Number Theoretic Transform (NTT) multiplier. NTT parameters are selected, in a way that the proposed design is executable on Digital Signal Processing (DSP) multiplier, to exploit its high clock rate. As the NTT kernel, is in general form, it can be pre-computed and stored in Look-up Tables (LUTs). The same LUTs are also usable for weight-factor as they both have symmetric periodicity properties. This optimization has saved 70% of LUTs utilization. Next optimization is proposed on reduction within NTT. The special prime moduli are exploited to accomplish a simple operation, where inverse Montgomery multiplication is replaced with shift and subtraction. The proposed optimizations are implemented for FHE encryption and realized on Kintex 7 platform. A magnitude of 93.2% speedup improvement is achieved for Toy, compared to benchmark software implementation. As the proposed design is targeted for full DSP implementation, it achieved a higher clock frequency (249.19 MHz), while consuming lower hardware resources.

1. Introduction

Cloud computing has been the most evolved technology in recent years and its usage has become wider in organizations and personal use. The advancement in cloud security has allowed computation on the encrypted data, however, requires them in a dynamic state. This is different from traditional encryption which has a static ciphertext such as RSA [1] and LUC-type cryptosystem [2]. Fortunately, a breakthrough Fully Homomorphic Encryption (FHE) by Gentry & Halevi [3] offers flexibility to allow unlimited arbitrary computation on the ciphertext without the need for decryption. Research on the theoretical development of FHE has derived 5 variants: Lattice-Based by Poppelmann [4], Learning with Error (LWE) by Nguyen [5], Integer-based by Coron *et al*., [6], NTRU-based by Dai *et al.,* [7] and Identity-based by Chillotti *et al.,* [8].

* *Corresponding author.*
E-mail address: shakirahhashim@uitm.edu.my

https://doi.org/10.37934/araset.30.3.362371

However, noises are simply accumulated during the encryption, hence could possibly fail the decryption process. To overcome this issue, key switching by Brakerski *et al.,* [9] and re-linearization by Brakerski and Vaikuntanathan [9] are proposed. An implementation of FHE on hardware [10–14] has shown significant improvement towards practical deployment compared to software [15-16] and software-hardware platform [7]. For instance, hardware implementation of Integer-based FHE proposed in [14] was able to achieve 45% shorter encryption time than software implementation in [17]. Proposing the pre-computation of Fast Fourier Transform (FFT) parameters and spectral technique, the work in [18] has significantly speed-up the encryption process. Meanwhile, Yang & Yang in [19] proposed four levels of pipeline with smaller modulus has achieved faster encryption by 1.62 speedup factor than work in [20]; both used FFT multiplier, but with different parameters. Noticed that all works discussed previously were targeted on a modern hardware platform like 7-series Field Programmable Gate Array (FPGA). It offers high-throughput Digital Signal Processing (DSP) block to execute DSP-related operations efficiently while allow parallelization. In fact, these features can be exploited to achieve excellent time performance while minimizing the usage of fabricated resources.

1.1 Motivation and the Novelty of the Proposed Work

The previous works of FHE Integer-Based on hardware has shown a practical solution towards real life deployment. Motivated by this success, this work is proposed to accelerate the encryption on FHE Integer-Based by focusing on the multiplication operation. Compared with other scheme, the parameters of Integer-Based scheme are clearly defined and easier to implement as the operations are performed over the integers. Meanwhile, Number Theoretic Transform (NTT) multiplier is chosen because it has been widely used in large multiplication such as in [21–23]. However, half of the NTT length, *N* are required for zero-padding. This condition has limited the multiplier size by *N/2*, due to cyclic convolution properties [24].

In this work, we adopt an optimized NTT, known as Weighted-NTT to eliminates the requirement of zero-padding to allow entire *N* filled with operands. Our work is different from others as we dictate our Weighted-NTT multiplier on DSP blocks. Therefore, parameters are bound to DSP operand size. The details contributions of this work are summarized as follows

i. A Weighted-NTT multiplier is proposed to multiply large numbers during FHE encryption
ii. Optimization in LUTs usage where precomputed NTT kernel is also usable for Weight-factor
iii. Optimization in NTT reduction by replacing inverse Montgomery multiplication with bitwise operation and subtraction

The rest of the paper is organized as follows. Section 2 explains on Integer Based FHE and brief explanation on Weighted-NTT. Section 3 describes the proposed methodology whereas in Section 4, the implementation of FHE is presented, and followed by Results and Discussion in Section 5. The paper is concluded in Section 6.

2. Integer-Based FHE

Still using Gentry's blueprint, FHE over the integer has been proposed for a simpler concept as the addition and multiplication are done over the standard integer and are not limited to an ideal lattice. Zhang and Li [12] proposed LWE scheme over the integers to adopt the concept of simplicity

of this scheme. Earlier, Integer-based FHE suffered from huge key size until Coron *et al.,* [6] proposed to encrypt the ciphertext together with a public key element quadratically instead of linearly. Later, Coron *et al.,* proposed a compression technique to generate public key on the fly thus making it most practical key size in literature [25].

The encryption process of a message m where $m \in \{0,1\}$ is $x_i = q_i \cdot p + r_i$, p is a secret key, q_i is a large random integer, and r_i is a small random integer. The encryption of m to a ciphertext c is shown in Eq. (1); b_i is a random integer vector, τ is a number of x_i in a public key while x_0 is a noise-free variant.

$$c \leftarrow m + 2r + 2\sum_{i=1}^{\tau} x_i \cdot b_i \ mod \ x_0 \tag{1}$$

Based on Eq. (1), two central operations are: Multiplication and modular reduction where both have quadratic complexity. Meanwhile, operands and in Table 1 are huge. These are the main reasons for slowness during encryption. Thus, we proposed a multiplier that can accommodate large operand on a single unit of multiplier so that iterative multiplication can be avoided.

Table 1
Test instance for the encryption process

Test Instance	Bit-length of x_i	Bit-length of b_i	T
Toy	150K	936	158
Small	830K	1476	572
Medium	4.2M	2016	2110
Large	19.0M	2556	7659

2.1 Weighted-Number Theoretic Transform

Weighted-NTT is an alternative for a large operand as it offers 50% larger multiplier size than standard NTT. As shown in [26] it has successfully accelerate large multiplication during FHE encryption and key exchange protocol [27], where 1024 NTT lengths are fully utilized to encode the message chunk into multiple points. Meanwhile, a work in [4] proposed a scalable design with a 14.10% speed-up improvement than software implementation of Lattice-based in [28]. A compact Weighted-NTT is proposed for Ring-LWE encryption block in [29] where parameters are computed on the fly, to utilize small resource and faster encryption than [4] by 12.60 speed-up factors. A reconfigurable design of NTT multiplier is targeted to have high throughput performance for Lattice-based encryption. The NTT point is scalable depending on operand size [22].

NTT parameters of the previous study are summarized in Table 2. Noted that a small moduli in [31] can produce a large multiplier size due to the employment of Weighted-NTT. Comparing this with [30], where Standard NTT is used, it needs 129 bits of moduli to produce the same multiplier size. This indicates that Weighted-NTT is exploitable to provide large multiplier although with a small modulus.

Table 2
NTT parameters of the previous studies

Work	NTT Parameter		Multiplier size (bit)	NTT Type
	NTT Length *N*	Moduli *P*		
[17]	64	$2^{64} - 2^{32} + 1$	896	NTT
[30]	512	$2^{128} + 1$	4096	NTT
[31]	1024	12289	4096	Weighted-NTT
[4]	1024	1061093377	9063	Weighted-NTT
[32]	32768	General form of moduli	-	NTT
[33]	4096	$2^{124} - 2^{64} + 1$	1×10^6	Weighted-NTT

3. Proposed Methodology

Three levels of optimization are proposed in this work: NTT parameter, memory, and Montgomery reduction.

3.1 NTT Parameter Selection

In this work, moduli P is prioritized over the other parameters in a sense that all computations in NTT are performed over moduli. The selected moduli satisfy the following criteria

i. There exist Nth roots of moduli which enables nega-cyclic convolution for Weighted-NTT
ii. Moduli *p* size must fit for DSP embedded multiplier as computation is targeted on the DSP core

For that, *p* is selected as 12289, following the parameter set of NewHope key exchange as proposed by Alkim *et al.,* in [34]. Details of the parameters are presented in Table 3. As noticed, the size of *p* is perfectly fit on DSP embedded multiplier, thus allow entire NTT computation to be executed within DSP embedded multiplier.

Table 3
Parameter selection for the proposed multiplier

Parameter	Moduli p	NTT length N	Dynamic range, b	NTT Kernel, α Weight Factor, ϕ	NTT multiplier size, N_c
Size	12289	1024 point	4 bits	α=49 ϕ=7	4096-bits

This work employs relatively small moduli which results in a small dynamic range and NTT coefficient. Fortunately, Weighted-NTT is adopted in this work to enable all NTT lengths to be employed with 4-bit operand without zero pads. So, 4-bit coefficient still can produce 4096 bits of multiplier size. This benefitted FHE encryption as the proposed multiplier can locate operand b_i in a single block. At least, iterative multiplication is only needed for operand x_i. This could reduce multiplication count and carry accumulation chain, hence speeding up the multiplication process.

3.2 Memory Optimization

We proposed precomputation of α and ϕ be stored in the LUTs. During the pre-computation, both are reduced to *P*. This way, several magnitudes of speed are achieved as iterative reduction could be avoided. At the same time, both have consistent bit size which is also efficient for memory

management. The nature of Weighted-NTT holds the relationship between two parameters α and ϕ where $\alpha = \sqrt{\phi}$. So, the same LUTs can be referred to both α and ϕ. In fact, due to symmetric periodicity properties in the ring of roots of unity, their inverse also can be obtained from the same parameters as stated by original work [35]. LUTs usage of Weighted-NTT and this work are presented in Table 4. The proposed optimization reduces almost 66.67% of LUTs usage than Weighted-NTT.

Table 4
LUTs usage of pre-computed Weighted-NTT and this work

	Stored Value in LUTs (Weighted-NTT)	Stored Value in LUTs (This work)
Pre-computed ϕ and ϕ^{-1}	2(N)	N
Pre-computed α and α^{-1}	2(N/2)	0
Total	3N	N

3.3 Montgomery Reduction Optimization

Our selected *p* is a special prime, known as Proth number. It is generalized as $q = k \cdot 2^m + 1$, k is a Low Hamming Weight (LHW) integer. Exploiting this prime, we replace the inverse Montgomery multiplication with shift and subtraction as shown in Algorithm 1. Line 2 reduces Q to R to avoid Q grows more than DSP multiplier size, thus enough for execution within a DSP for 5 clock cycles.

Algorithm 1: Optimized Montgomery	
1: $T = A \times B$	
2: $Q = (T \gg (n) + T)$ mod R	(n=12287, R=2^{14})
3: $Z = {(T - Q.P)}/{R}$	(P=12289)
4: $If\ (Z < 0)$	
5: $\{Z = Z + P\}$	
6: $Else$	
7: $Return\ Z$	

Since modular multiplication is processed in the Montgomery domain, the final answer is in the form of $ABr^{-1} mod\ M$ where $r^{-1} = 9216$. So, conversion back to the standard domain is required. Final NTT coefficient requires de-factoring of weight which is then followed by multiplication with Montgomery factor to convert them into a standard domain as $z_i = \hat{z}_i \times {\phi^{-1}}_i \times R$, where ϕ is retrieved from $LUTs_\phi$, ${\phi^{-1}}_i = LUTs_\phi[\phi]_{n-i}$.

Assume reading α_i from LUTs needs 1 clock cycle while butterfly computation on each stage requires 5 clock cycle for modular multiplication using the Modified Montgomery. This makes the overall time cost of forward/reverse transform is $6 \times log_2 N$. Meanwhile, weight-factor involves multiplication between NTT coefficients a_i, b_i and ϕ_i, produces $\widehat{a_i}$ and $\widehat{b_i}$. ϕ_i are retrieved from LUT thus, need 1 clock cycle for reading, while additional 5 clock cycles are for Montgomery multipliers that are parallelized to *N*. Pointwise multiplication of A_i and B_i is performed over moduli p. Thus, Modified Montgomery unit is employed and parallelized to N units, within 5 clock cycle. Next, decomposition is performed on $\widehat{C_i}$ requires a multiplication with ${\phi^{-1}}_i$ which retrieved from LUTs and de-factoring the Montgomery to convert the final answer into a standard representation. Both need 6 clock cycle. The summary of timing costs for the proposed Weighted-NTT is presented in Table 5.

Table 5
Timing costs for the proposed Weighted-NTT

Weighted-NTT Step	Estimated timing cost (clock cycle)
Forward and Inverse Transform	$2(6 \times log_2 N)$
Weight Factor Multiplication	6
Pointwise Multiplication Unit	5
De-factor and Montgomery Conversion	6×2
Overall Costs	$\Delta_{NTT} = 12\ log_2 N + 23$

4. FHE Implementation

The proposed NTT is then implemented for Integer-based FHE encryption. The design must be able to accommodate FHE parameters of million bits size. An assumption is made that hardware resources are enough to store all parameters. Meanwhile, multiplication of FHE operand requires multiple iterations that split into two processes

i. Inner multiplication where block of the proposed NTT multiplier multiplies x_i and b_i iteratively and generates partial products; and
ii. Outer accumulation where a shifter and an adder are needed to accumulate the partial products.

Total timing cost for inner multiplication Δ_{IM} and outer iteration Δ_{OA} are expressed in Eq. (2) and Eq. (3) respectively. Noted that $Z = \left\lceil \frac{X_i}{N_c} \right\rceil$; the iteration count of NTT and N_c is 4096-bit. Details of operand for multiplication and reduction blocks are shown in Table 6. Overall costs of FHE encryption for multiplication, $\Delta_{FHE\_encrypt\_mult}$ is expressed in Eq. (4).

$$\Delta_{IM} = Z(\Delta_{NTT}) \tag{2}$$

$$\Delta_{OA} = Z + 1 \tag{3}$$

$$\Delta_{FHE\_encrypt\_mult} = \Delta_{IM} + \Delta_{OA} \tag{4}$$

Table 6
Operand size of multiplication and reduction building blocks

Group	Multiplication block		Reduction block	
	Operand 1 (x_i)	Operand 2 (b_i)	Operand 1 ($b_i + \tau$)	Operand 2 ($x_i + b_i + \tau$)
Toy	150k	936	1094	150K
Small	830k	1476	2048	830K
Medium	4.2m	2016	4126	4.2M
Large	19.0m	2556	10251	19.35M

Barrett method [36] requires two large multiplications; thus, the same NTT multiplier can be re-used during reduction. This approach is also used by [37] to minimize hardware usage. Somehow, Barrett needs a larger operand size. Thus, iterative multiplication is needed with an iteration count for reduction building block is $Z_{reduc} = \left\lceil \frac{Y_1}{N_c} \right\rceil \cdot \left\lceil \frac{Y_2}{N_c} \right\rceil$ and this is also the timing cost for inner multiplication, $\Delta_{IM_{reduc}}$.

Meanwhile, timing cost for outer accumulation $\Delta_{OA_{reduc}}$ needs extra 1 clock cycle. Noted Y_1 and Y_2 is operand1 and operand2 of reduction building block respectively. So, total reduction time

$\Delta_{FHE\_encrypt\_reduc}$ and total encryption time $\Delta_{FHE\_encrypt}$ as expressed in Eq. (5) and Eq. (6) respectively. Noted that multiplication by 2 is incurred to the timing cost because Barrett requires two large number multiplications.

$$\Delta_{FHE\_encrypt\_reduc} = \Delta_{IM_{reduc}} + \Delta_{OA_{reduc}} \tag{5}$$

$$\Delta_{FHE\_encrypt} = \left(\Delta_{FHE\_encrypt\_mult} \times \tau\right) + \left(\Delta_{FHE\_encrypt\_reduc} \times 2\right) \tag{6}$$

5. Results and Discussions

The proposed design is targeted on Kintex 7 (7K480T) and reaches a maximum frequency of 249.19 MHz, and hardware usage is presented in Table 7. As the frequency is known, an encryption time can be obtained by multiplying the generated frequency with the cycle count of multiplication and reduction of each group presented in Table 8.

Table 7
Hardware results of the optimized Weighted-NTT

Resource	Register	BRAMs	LUTs	DSPs
Estimated Device Utilization	14989	32	13348	1536

Table 8
Cycle count and encryption time of each FHE group

Group	$\Delta_{\text{FHE\_encrypt\_mult}}$ (cycles)	$\Delta_{FHE\_encrypt\_reduc}$ (cycles)	$\Delta_{FHE\_encrypt}$ (cycles)	Encryption Time (s)
Toy	5329	5329	852640	0.0034
Small	29233	29233	16779742	0.067
Medium	147745	295489	312332928	1.25
Large	668017	2027377	5120396957	20.53

Encryption time is excellent for Toy and Small groups. However, as the operand grows, the encryption time becomes slower, especially for Medium and Large. The main reason is due to higher partial product iteration, mainly in Medium and Large reduction blocks. Both iterate for 2 and 3 times respectively, while others need for 1 iteration. The significant contribution is noted from its low-area utilization. An optimized Montgomery allows the NTT multiplier to be implemented on the DSP core, thus minimizing LUTs usages to store the pre-computed parameters.

A performance comparison of the proposed design is presented in Table 9. It is compared to the benchmark software [6] and high-speed FHE [38]. Significant improvement is noticed as the proposed design achieves 93.2% higher speed (Toy) than software implementation. Upon comparing with high-speed design, we are slower. However, as the proposed design is targeted for full DSP implementation, it achieved 33.33% higher frequency than high-speed design [38]. Moreover, the proposed design utilized 91.70 % lesser DSP than high-speed design.

Table 9
Performance and DSP usage comparison of the proposed work and previous works

Works	DSP Usage	Freq (MHz)	Toy	Small	Medium	Large
The proposed Design	1536	249.19	0.0034s	0.067s	1.25s	20.53s
Software implementation [6]	n/a	n/a	0.05s	0.79s	10s	2min 57s
High-speed design [38]	18496	166.45	0.00082s	0.013s	0.22s	3.96

6. Conclusions

In this work, we selected Weighted-NTT parameter in a way that multiplier is executable within DSP core to exploit its high throughput multiplier. The optimization also proposed on memory to store both NTT kernel and weight factor in the same LUT, which significantly reduces LUT usage by 66.67%. During the reduction within Weighted-NTT, we eliminated multiplication of inverse Montgomery and replaced it with simpler addition and bitwise operation. The proposed methodology has significant results on overall hardware performance. It encrypted 93.2% faster than software implementation and 33.33% higher frequency than higher speed design [38]. We believe the encryption time of the proposed design can be improved if it is implemented on FPGA with higher DSP such as Virtex 7, as more multiplier unit can be parallelized.

Acknowledgement

The registration fees are funded by Pembiayaan Yuran Prosiding Berindeks (PYPB), Tabung Dana Kecemerlangan Pendidikan (DKP), Universiti Teknologi MARA (UiTM), Malaysia.

References

[1] Rivest, Ronald L., Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems." *Communications of the ACM* 21, no. 2 (1978): 120-126. https://doi.org/10.1145/359340.359342

[2] Sarbini, Izzatul Nabila, Tze Jin Wong, Lee Feng Koo, Ahmad Fadly Nurullah Rasedee, Fatin Hana Naning, and Mohammad Hasan Abdul Sathar. "Security Analysis on LUC-type Cryptosystems Using Common Modulus Attack." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 29, no. 3 (2023): 206-213. https://doi.org/10.37934/araset.29.3.206213

[3] Gentry, Craig, and Shai Halevi. "Implementing gentry's fully-homomorphic encryption scheme." In *Advances in Cryptology–EUROCRYPT 2011: 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings 30*, pp. 129-148. Springer Berlin Heidelberg, 2011. https://doi.org/10.1007/978-3-642-20465-4_9

[4] Pöppelmann, Thomas, and Tim Güneysu. "Towards efficient arithmetic for lattice-based cryptography on reconfigurable hardware." In *Progress in Cryptology–LATINCRYPT 2012: 2nd International Conference on Cryptology and Information Security in Latin America, Santiago, Chile, October 7-10, 2012. Proceedings 2*, pp. 139-158. Springer Berlin Heidelberg, 2012. https://doi.org/10.1007/978-3-642-33481-8_8

[5] Nguyen, Phong Q., and Oded Regev. "Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures." In *Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28-June 1, 2006. Proceedings 25*, pp. 271-288. Springer Berlin Heidelberg, 2006. https://doi.org/10.1007/11761679_17

[6] Coron, Jean-Sébastien, Avradip Mandal, David Naccache, and Mehdi Tibouchi. "Fully homomorphic encryption over the integers with shorter public keys." In *Advances in Cryptology–CRYPTO 2011: 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings 31*, pp. 487-504. Springer Berlin Heidelberg, 2011. https://doi.org/10.1007/978-3-642-22792-9_28

[7] Dai, Wei, and Berk Sunar. "cuHE: A homomorphic encryption accelerator library." In *Cryptography and Information Security in the Balkans: Second International Conference, BalkanCryptSec 2015, Koper, Slovenia, September 3-4, 2015, Revised Selected Papers 2*, pp. 169-186. Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-29172-7_11

[8] Chillotti, Ilaria, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. "Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds." In *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22*, pp. 3-33. Springer Berlin Heidelberg, 2016. https://doi.org/10.1007/978-3-662-53887-6_1

[9] Brakerski, Zvika. "Fully homomorphic encryption without modulus switching from classical GapSVP." In *Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pp. 868-886. Springer Berlin Heidelberg, 2012. https://doi.org/10.1007/978-3-642-32009-5_50

[10] Hashim, Shakirah, and Mohammed Benaissa. "Accelerating integer based fully homomorphic encryption using frequency domain multiplication." In *Information and Communications Security: 20th International Conference,*

ICICS 2018, Lille, France, October 29-31, 2018, Proceedings, pp. 161-176. Springer International Publishing, 2018. https://doi.org/10.1007/978-3-030-01950-1_10

[11] Paul, Bikram, Tarun Kumar Yadav, Balbir Singh, Srinivasan Krishnaswamy, and Gaurav Trivedi. "A resource efficient software-hardware co-design of lattice-based homomorphic encryption scheme on the fpga." *IEEE Transactions on Computers* (2022). https://doi.org/10.1109/TC.2022.3198628

[12] Zhang, Ailuan, and Ziehen Li. "A New LWE-based Homomorphic Encryption Algorithm over Integer." In *2021 International Conference on Computer Information Science and Artificial Intelligence (CISAI)*, pp. 521-525. IEEE, 2021. https://doi.org/10.1109/CISAI54367.2021.00106

[13] Hu, Xiao, Minghao Li, Jing Tian, and Zhongfeng Wang. "Efficient Homomorphic Convolution Designs on FPGA for Secure Inference." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 30, no. 11 (2022): 1691-1704. https://doi.org/10.1109/TVLSI.2022.3197895

[14] Cao, Xiaolin, Ciara Moore, Máire O'Neill, Elizabeth O'Sullivan, and Neil Hanley. "Optimised multiplication architectures for accelerating fully homomorphic encryption." *IEEE Transactions on Computers* 65, no. 9 (2015): 2794-2806. https://doi.org/10.1109/TC.2015.2498606

[15] Halevi, Shai, and Victor Shoup. "Faster homomorphic linear transformations in HElib." In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I 38*, pp. 93-120. Springer International Publishing, 2018. https://doi.org/10.1007/978-3-319-96884-1_4

[16] Kim, Jeongsu, and Aaram Yun. "Secure fully homomorphic authenticated encryption." *IEEE Access* 9 (2021): 107279-107297. https://doi.org/10.1109/ACCESS.2021.3100852

[17] Cao, Xiaolin, and Ciara Moore. "New integer-FFT multiplication architectures and implementations for accelerating fully homomorphic encryption." *Cryptology ePrint Archive* (2013).

[18] Doröz, Yarkın, Erdinç Öztürk, and Berk Sunar. "Accelerating fully homomorphic encryption in hardware." *IEEE Transactions on Computers* 64, no. 6 (2014): 1509-1521.

[19] Su, Yang, Bailong Yang, Chen Yang, and Luogeng Tian. "Fpga-based hardware accelerator for leveled ring-lwe fully homomorphic encryption." *IEEE Access* 8 (2020): 168008-168025. https://doi.org/10.1109/ACCESS.2020.3023255

[20] Pedrosa, Alejandro Ranchal. "Implementing fully homomorphic encryption schemes in FPGA-based systems." *ETS de Ingenieros Informáticos (UPM), Madrid, Spain, Tech. Rep* (2016).

[21] Gueron, Shay, and Fabian Schlieker. "Speeding up R-LWE post-quantum key exchange." In *Secure IT Systems: 21st Nordic Conference, NordSec 2016, Oulu, Finland, November 2-4, 2016. Proceedings*, pp. 187-198. Cham: Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-47560-8_12

[22] Derya, Kemal, Ahmet Can Mert, Erdinç Öztürk, and Erkay Savaş. "CoHA-NTT: A configurable hardware accelerator for NTT-based polynomial multiplication." *Microprocessors and Microsystems* 89 (2022): 104451. https://doi.org/10.1016/j.micpro.2022.104451

[23] Liu, Zhe, Hwajeong Seo, Sujoy Sinha Roy, Johann Großschädl, Howon Kim, and Ingrid Verbauwhede. "Efficient Ring-LWE encryption on 8-bit AVR processors." In *Cryptographic Hardware and Embedded Systems--CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings 17*, pp. 663-682. Springer Berlin Heidelberg, 2015. https://doi.org/10.1007/978-3-662-48324-4_33

[24] Agarwal, Ramesh C., and C. Sidney Burrus. "Number theoretic transforms to implement fast digital convolution." *Proceedings of the IEEE* 63, no. 4 (1975): 550-560. https://doi.org/10.1109/PROC.1975.9791

[25] Coron, Jean-Sébastien, Tancrede Lepoint, and Mehdi Tibouchi. "Scale-invariant fully homomorphic encryption over the integers." In *Public-Key Cryptography–PKC 2014: 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings 17*, pp. 311-328. Springer Berlin Heidelberg, 2014. https://doi.org/10.1007/978-3-642-54631-0_18

[26] Feng, Xiang, and Shuguo Li. "Accelerating an FHE integer multiplier using negative wrapped convolution and ping-pong FFT." *IEEE Transactions on Circuits and Systems II: Express Briefs* 66, no. 1 (2018): 121-125. https://doi.org/10.1109/TCSII.2018.2840108

[27] Kuo, Po-Chun, Wen-Ding Li, Yu-Wei Chen, Yuan-Che Hsu, Bo-Yuan Peng, Chen-Mou Cheng, and Bo-Yin Yang. "High performance post-quantum key exchange on FPGAs." *Cryptology ePrint Archive* (2017).

[28] Rückert, Markus, and Michael Schneider. "Estimating the security of lattice-based cryptosystems." *Cryptology ePrint Archive* (2010).

[29] Roy, Sujoy Sinha, Frederik Vercauteren, Nele Mentens, Donald Donglong Chen, and Ingrid Verbauwhede. "Compact ring-LWE cryptoprocessor." In *Cryptographic Hardware and Embedded Systems–CHES 2014: 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings 16*, pp. 371-391. Springer Berlin Heidelberg, 2014.

[30] Roy, Sujoy Sinha, Frederik Vercauteren, Nele Mentens, Donald Donglong Chen, and Ingrid Verbauwhede. "Compact ring-LWE cryptoprocessor." In *Cryptographic Hardware and Embedded Systems–CHES 2014: 16th International*

Workshop, Busan, South Korea, September 23-26, 2014. Proceedings 16, pp. 371-391. Springer Berlin Heidelberg, 2014.

[31] Longa, Patrick, and Michael Naehrig. "Speeding up the number theoretic transform for faster ideal lattice-based cryptography." In *Cryptology and Network Security: 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings 15*, pp. 124-139. Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-48965-0_8

[32] Öztürk, Erdinç, Yarkın Doröz, Berk Sunar, and Erkay Savaş. "Accelerating somewhat homomorphic evaluation using FPGAs." *Cryptology ePrint Archive* (2015).

[33] Pöppelmann, Thomas, Michael Naehrig, Andrew Putnam, and Adrian Macias. "Accelerating homomorphic evaluation on reconfigurable hardware." In *Cryptographic Hardware and Embedded Systems--CHES 2015: 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings 17*, pp. 143-163. Springer Berlin Heidelberg, 2015. https://doi.org/10.1007/978-3-662-48324-4_8

[34] Alkim, Erdem, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. "Post-quantum key exchange-A New Hope." In *USENIX security symposium*, vol. 2016. 2016.

[35] Cooley, James W., and John W. Tukey. "An algorithm for the machine calculation of complex Fourier series." *Mathematics of computation* 19, no. 90 (1965): 297-301. https://doi.org/10.1090/S0025-5718-1965-0178586-1

[36] Barrett, Paul. "Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor." In *Advances in Cryptology—CRYPTO'86: Proceedings*, pp. 311-323. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000. https://doi.org/10.1007/3-540-47721-7_24

[37] Moore, Ciara, Máire O'Neill, Neil Hanley, and Elizabeth O'Sullivan. "Accelerating integer-based fully homomorphic encryption using Comba multiplication." In *2014 IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 1-6. IEEE, 2014. https://doi.org/10.1109/SiPS.2014.6986063

[38] Cao, Xiaolin, Ciara Moore, Máire O'Neill, Neil Hanley, and Elizabeth O'Sullivan. "High-speed fully homomorphic encryption over the integers." In *Financial Cryptography and Data Security: FC 2014 Workshops, BITCOIN and WAHC 2014, Christ Church, Barbados, March 7, 2014, Revised Selected Papers 18*, pp. 169-180. Springer Berlin Heidelberg, 2014. https://doi.org/10.1007/978-3-662-44774-1_14