International Journal of Security and Its Applications
Vol. 10, No. 9 (2016), pp.9-18
http://dx.doi.org/10.14257/ijsia.2016.10.9.02

Memory Saving Architecture of Number Theoretic Transform for Lattice Cryptography

Sangook Moon[1]

[1]Mokwon University, 88 Doan North Road,
302-729 Daejeon, Korea
smoon@mokwon.ac.kr

Abstract

In realizing a homomorphic encryption system, the operations of 1encrypt, decrypt, and recrypt constitute major portions. The most important common operation for each back-bone operations include a polynomial modulo multiplication for over million-bit integers, which can be obtained by performing integer Fourier transform, also known as number theoretic transform. In this paper, we adopt and modify an algorithm for calculating big integer multiplications introduced by Schonhage-Strassen to propose an efficient Ring-LWE processor architecture which can save memory. The proposed architecture of Ring-LWE encryption processor has been implemented on an FPGA and evaluated.

***Keywords**: homomorphic encryption, cryptography, Ring-LWE*

1. Introduction

MIT selects 10 state-of-the-art technologies that are expected to have large effects later every year and announces the technologies on the website named Technology Review and the 10 technologies selected in 2011 are; cancer genomics, cloud streaming, crash-proof code, gestural interfaces, homomorphic encryption, social indexing, smart transformers, solid-state batteries, separating chromosomes, and synthetic cells [1]. Among them, the technology to be studied in this project is related to homomorphic encryption (HE) and this technology can process operations of encrypted data at high speed without decrypting the data.

HE is an encryption technique that encrypts texts to basically have random number characteristics so that when a certain operation is applied to two texts encrypted by this technique and the results are decrypted later, the results of decryption should be the same as the results of application of the same operation to the relevant plain texts. When the encryption technique shows this characteristic for one operation, it is called homomorphic and when this is simultaneously the case with both addition and multiplication and the encryption technique satisfies the ring characteristic mentioned in the number theory, it is called fully homomorphic. The characteristics of fully homomorphic encryption (FHE) can be expressed as a numerical formula as shown by equation 1 (E: encryption, m: message, k: key). As an example of HE for addition, it can be seen that, when the numbers 1 and 2 have been entered into a cloud system and encrypted into the numbers 33 and 54 respectively, if the encrypted numbers are added up and the user downloads the result 87 from the cloud system and decrypts the number, the result will be 3, which is the same as the value of addition of the relevant plain texts [2-5].

FHE is suitable for information protection in future society as it is applicable to sensitive data in cloud systems, hospital diagnosis records, National Health Insurance

1

Service or tax affairs information, and bank data and is internally based on lattice cryptography.

$$E_k(m_1) + E_k(m_2) = E_k(m_1 + m_2), E_k(m_1) * E_k(m_2) = E_k(m_1 * m_2) \tag{1}$$

2. Lattice Cryptography

Due to the advent of quantum computing, opinions that traditional encryption (RSA, ECC) based on difficulties in prime factorization or discrete logarithm solutions should not be safe any further are dominant. Encryption techniques in the contemporary times have been gradually reoriented largely from traditional methods to new methods based on vector lattice solutions. Lattices are defined as sets of points in n-dimensional spaces and have repetitive structures. One of important characteristics of lattices that enables applying lattices to encryption is that all elements of the lattices can be expressed as a linear summation of two vectors (because two-dimensional) or a combination of vectors. Those vectors are called basis vectors and lattice encryption is based on difficulties in finding the shortest vectors that can express such lattices (SVP; shortest vector problem). This cryptologic characteristic is known to be safe against attacks by quantum computing because it is based on difficulties in geometric solutions, which are fundamentally different from difficulties in arithmetic solutions of RSA or ECC [6-8]. The reason we chose the lattice cryptography against RSA-like algorithms is shown in Figure 1.

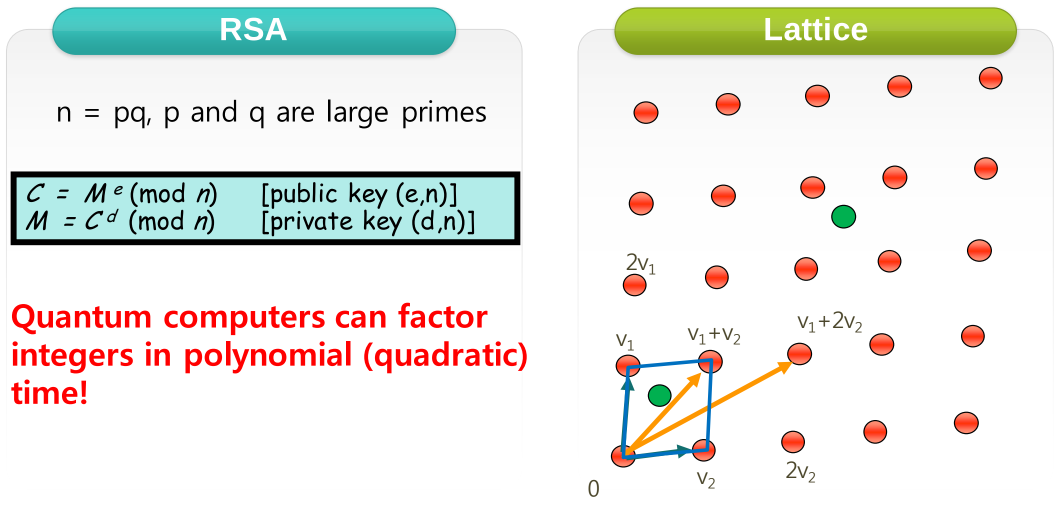
RSA
n = pq, p and q are large primes
C = M^e (mod n) [public key (e,n)]
M = C^d (mod n) [private key (d,n)]
Quantum computers can factor integers in polynomial (quadratic) time!
Lattice
2v1
v1
v1+v2
v1+2v2
0
v2
2v2

Figure 1. Comparison of RSA and Lattice Cryptography

When seen from the viewpoint of cryptology, the lattice cryptogram theory and FHE systems that use the lattice cryptogram theory are very important in the field of mathematics. Studies on lattice cryptograms can be retrieved from several papers although the number of such papers is not large. In Figure 2, we show one derivative of lattice cryptography known as Ring-LWE (Learning With Errors). In Ring-LWE, we use two public keys and trivial error vectors. By manipulating a few expressions, The message can be encrypted and decrypted.

$$R_q = Z_q[x] \,/ < f > \quad (f(x) = x^n + 1, \quad q \equiv 1 \; mod \; 2n)$$
$$R_2 = Z_2[x] \,/ < f > \quad (f(x) = x^n + 1)$$

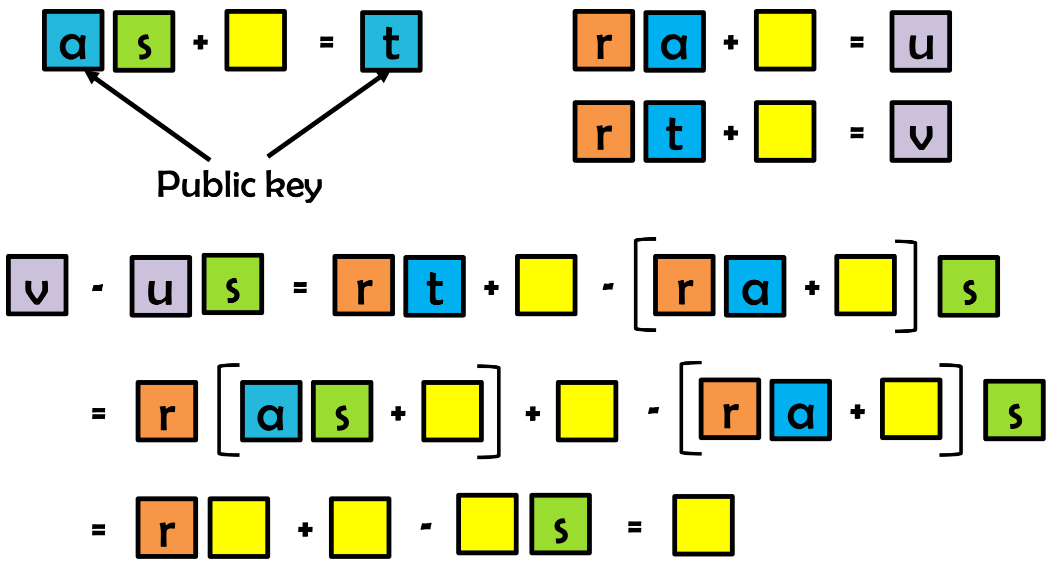

Figure 2. Ring-LWE Cryptosystem

3. Fully Homomorphic Encryption

FHE had been hardly studied until it was proved to be valid by Gentry in the USA in 2009. Recently, a paper of a university in South Korea that proposed a fully homomorphic cryptogram system from a mathematical viewpoint using the Chinese remainder theorem and the lattice theory was published in the conference EuroCrypt 2013, which is authoritative in the field of cryptology [12].

Although problems of HE were derived immediately after Professor Rivest of MIT presented the concepts of RSA and public key cryptogram algorithms in 1978, these problems remained as unsolved problems in cryptology for 30 years thereafter until they began to receive attention in the academia after the validity of HE was proved in September 2009 by Craig Gentry, a graduate student of Stanford University. As of July 2016 when less than seven years had passed since then, the number of times of citation of Gentry's doctoral dissertation reached 2924 (basis: Google scholar) to bring about explosive demand for research in the fields of mathematics, computer science, and electricity and electronics [13]. The technique and principle of somewhat homomorphic encryption (SWHE) are as follows. In Figure 3, we show how computing in encrypted data works in FHE.

3.1. Key Setting

- The following variables are determined from the given safety parameter.$\boldsymbol{\lambda}$

$\boldsymbol{\eta}$:bit length of the secret key, $\boldsymbol{\rho}$: bit length of the error, $\boldsymbol{\gamma}$: bit lengths of public keys $\mathbf{x_i}$ that include errors, $\boldsymbol{\tau}$: number of public keys $\mathbf{x_i}$, $\boldsymbol{\rho'}$: bit length of the secondary error used in encryption

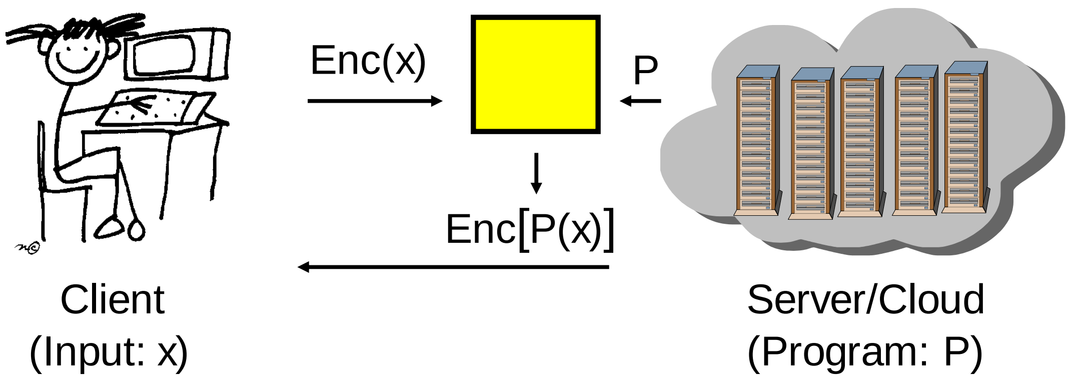

Figure 3. How Computing on Encrypted Data Works in FHE

- Select an odd number $\boldsymbol{\rho}$ of an arbitrary bit length $\boldsymbol{\eta}$.

- Determine integers $\mathbf{x_i}$ that will be used as public keys for $\mathbf{0 \leqq i \leqq \tau}$ as shown by equation 2 below. where, $\mathbf{q_i}$ is arbitrary integers smaller than $\mathbf{s^{\gamma-\eta}}$, $\mathbf{r_i}$ is error values that satisfy $\mathbf{s^{-\rho} \leqq i \leqq 2^{\rho}}$.

$$\mathbf{x_i = q_i * p + r_i} \tag{2}$$

- Rearrange the selected $\mathbf{x_i}$ in order of size so that $\mathbf{x_0}$ becomes the largest value. where, $\mathbf{x_0}$ is assumed to be an odd number that satisfies ‘ $\mathbf{x_0 (mod\ p) = even}$’.

- The final public keys are $(\mathbf{x_0, x_1, \cdots, x_\tau})$ and the secret key is p.

3.2. Encryption Stage

- Plain text m is assumed to be a value of 0 or 1. This is, m ∈ {0,1}.

-Arbitrarily select j that satisfies $1 \leqq j \leqq \tau$ and arbitrarily select j pieces of elements from among public keys$(x_1, \cdots, x_\tau)$. Assume the selected elements to be$(x_1', \cdots, x_j')$.

- In addition, select an arbitrary random number r with bit lengthρ'.

- Finally calculate the cryptogram for m as shown by equation 3 below.

$$c = \left(m + 2r + 2 \sum_{0 \leqq i \leqq j} x_i' \right) \mathrm{mod} x_0 \tag{3}$$

3.3. Decryption Stage

- Decrypt the given cryptogram c as shown by equation 4 below using secret key p.

$$m = (c \bmod p) \bmod 2 \tag{4}$$

In the encryption stage, plain text m is hidden by the sum of random number r and arbitrarily selected public keys. Since public keys x_i have been selected as $x_i = q_i * p + r_i$ respectively, the $q_i * p$ parts becomes extinct by the calculation of (c mod p), which is the first stage of the process of decryption and only the errors parts r_i remains. On review the encryption stage, it can be seen that all these error parts have been multiplied by constant 2. Therefore all of them are eliminated by the mod 2 operation, which is the second stage of decryption.

In addition, operations of cryptograms $c_1 = m_1 + 2r_1 + 2\sum x_i (\mathrm{mod}\ x_0)$and

$c_2 = m_2 + 2r_2 + 2\sum x_i (\mathrm{mod}\ x_0)$ for plain text m_1, m_2 can now be performed as shown by equations 5 and 6 below.

$$c_1 + c_2 = m_1 + 2r_1 + 2\sum x_i (\mathrm{mod}\ x_0) + m_2 + 2r_2 + 2\sum x_i (\mathrm{mod}\ x_0) \quad (5)$$

$$= (m_1 + m_2) + 2(r_1 + r_2) + 2\sum x_i (\mathrm{mod}\ x_0)$$

$$c_1 * c_2 = (m_1 + 2r_1 + 2\sum x_i (\mathrm{mod}\ x_0)) * (m_2 + 2r_2 + 2\sum x_i (\mathrm{mod}\ x_0)) \quad (6)$$

$$= (m_1 * m_2) + 2(m_2 r_1 + m_1 r_1 + m_1 r_2 + \sum a_i x_i)(\mathrm{mod}\ x_0)$$

As can be seen from the above equations, although the above operations are homomorphic, through the process of repeatedly performing these operations, the sizes of the errors included in the cryptograms gradually increase. In particular, in the case of multiplication operations, the sizes of these errors increase quite rapidly and when the sizes of these errors have exceeded the size of secret key p, accurate decryption becomes impossible so that only a limited number of times of operations become possible. Gentry's core idea is repeating the second stage encryption (recrypt) before the size of the errors exceed the tolerable size to refresh the error to enable perfect decryption regardless of the number of times of operation, which is called bootstrappable. Due to these repetitive multi-stage encryption operations, the FHE processing is currently very low to the extent that FHE cannot be applied to applications such as cloud computing yet and in the present project, yearly studies necessary to enable the application will be conducted.

4. Number Theoretic Transform

Encrypt, decrypt, recrypt operations form the large frame of FHE implementation. The commonly most important operation for the individual operations is modular multiplication of large integers exceeding a million bits and the modular multiplication is divided into two stages; multiplication operation (multiplication) for very large integers that can be obtained by repeatedly performing FFT and modular reduction of the results of the multiplication. In the first year, small area integer FFT structures suitable for vector processor cores will be proposed in order to implement algorithms necessary to efficiently perform the modular multiplication of large integers proposed by Schonhage-Strassen [8].

The integer FFT is essentially defined in finite fields and does not require complicated arithmetic operations such as floating points. When primitive n square root w has been given in finite field Z_p $(w^n = 1 \mathrm{mod} p)$, integer FFT for vector $\{a_0, \cdots, a_{n-1}\}$ into vector $\{A_0, \cdots, A_{n-1}\}$and inverse transformation, that is, $NTT_w(a)$ and $NTT_w{}^{-1}(A)$ are defined as follows.

$NTT_w(a)$:

$A_i = \sum_{j=0}^{n-1} a_j w^{ij} \mathrm{mod} p, i = 0,1,\cdots n-1$

$\mathrm{NTT_w}^{-1}(\mathrm{A})$:

$$a_i = n^{-1}\sum_{j=0}^{n-1} A_j w^{-ij} \mathrm{mod} p, i = 0,1,\cdots n-1$$

The necessary and sufficient condition for NTT to exist is that n can be divided by $q-1$ for all prime factors q of p and in this case, for arbitrary prime factor q of n, $w^{n/q} - 1 \neq 0 \mathrm{mod} p$ is valid. Figure 4 show the original algorithm. In this case, if n is assumed to be a square number of 2, when $n \mathrm{mod}(p-1) = 1$, NTT can be efficiently calculated within time O(nlogn).

```
Input: Polynomial $a(x) \in Z_q[x]$ of degree $n-1$ and $n-th$ primitive root $w_n \in Z_q$ of unity
Output: Polynomial $A(x) \in Z_q[x]$ = NTT($a$)
A <- bit_Inverse(a(x))
m <- 2
while m≤N {
          s <- 0
          while s < N {
                    for i to m/2-1 {
                              N <- i.n/m
                              a <- s + i
                              b <- s + i + m/2
                              c <- A[a]
                              d <- A[b]
                              A[a] <- c + $\omega^{N\mathrm{mod}\ n}$d mod q
                              A[b] <- c - $\omega^{N\mathrm{mod}\ n}$d mod q
                              }
                    s <- s + m
                    }
          m <- m*2
          }
return A/
```

Figure 4. Shonhage-Strassen Number Theoretic Algorithm

5. Gaussian Sampler

Gaussian sampler was adopted from Knuth-Yao's work. We applied the inverse sampling algorithm to achieve the hardware Gaussian sampler shown in Figure 5.

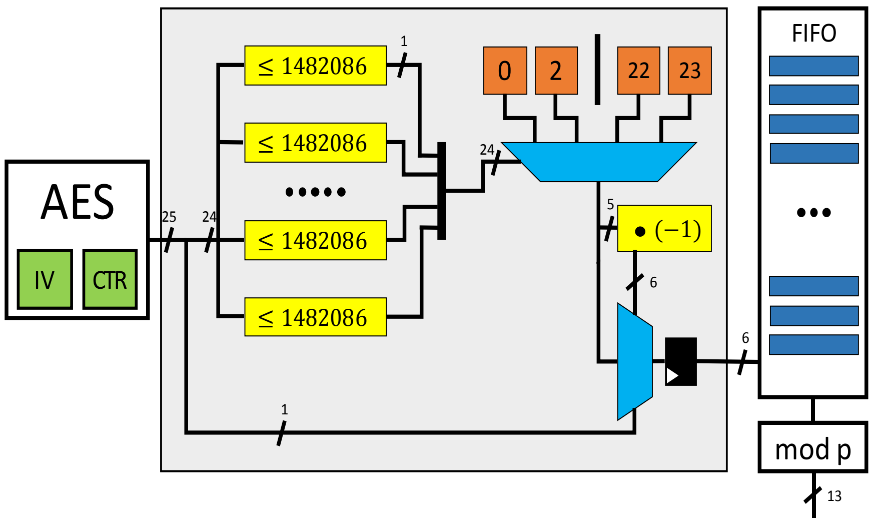

Figure 5. Gaussian Sampler

6. Ring-LWE Encryption and Decryption

In target LWE processor, we need to be able to encrypt the computed data. For simplicity, we consider two operations, addition and multiplication. In Figure 6 and Figure 7, we prove how encryption and decryption on computed data work respectively.

Message : m (polynomial) $\in R_2 = Z_2[x] \,/\, < x^n + 1 >$
Ciphertext : c (pair of polyomials) $\in R_q = Z_q[x] \,/\, < x^n + 1 >$

Encrypt

$$c = (c_0, c_1) \qquad c_0 = a * s + 2 * e + m, \quad c_1 = -a$$

$$\begin{aligned} c_{add} &= c + c' = (c_0 + c_0', \quad c_1 + c_1') \\ &= (a * s + 2 * e + m + a' * s\, 2 * e' + m', -(a + a')) \\ &= ((a + a') * s + 2 * (e + e') + \; m + m', -(a + a')) \end{aligned}$$

Decrypt (+)

$$c_{add} = (c_{add_0}, c_{add_1}) \qquad find \;\; c_{add_0} + c_{add1} * s$$

$$\begin{aligned} &(a + a') * s + 2 * (e + e') + \; m + m' \; - (a + a') * s \\ &= \; 2 * (e + e') + \; m + m' \cong m + m' \; (mod\; q) \end{aligned}$$

Figure 6. Encryption and Decryption of Plus Operation

Message : m (polynomial) $\in R_2 = Z_2[x] \,/\, < x^n + 1 >$
Ciphertext : c (pair of polyomials) $\in R_q = Z_q[x] \,/\, < x^n + 1 >$

Encrypt

$$c = (c_0, c_1) \qquad c_0 = a * s + 2 * e + m, \quad c_1 = -a$$

$$c_{mul} = c * c' = (c_0 * c_0', \qquad c_0 * c_1' + c_0' * c_1 \,, \quad c_1 * c_1')$$

$$\begin{aligned} c_0 * c_0' = (&-a * a' * s^2 + (c_0 * a' + c_0' * a) * s + \\ &2 * (2 * e * e' + e * m' + e' * m) + m * m' \end{aligned}$$

$$c_1 * c_1' = a * a \qquad\qquad c_0 * c_1' + \; c_0' * c_1 \; = \; \ldots$$

Decrypt (*)

$$c_{mul} = (c_{mul_0}, c_{add_1}, c_{mul_2}) \qquad find \;\; c_{mul_0} + c_{mul1} * s \; + c_{mul_2} * s^2$$

$$\begin{aligned} c_{mul_0} + c_{mul1} * s \; + c_{mul_2} * s^2 &= (c_0 + c_1 * s) * (c_0' + c_1' * s) \\ &\cong m * m' \quad (mod\; q) \end{aligned}$$

Figure 7. Encryption and Decryption of Multiplication Operation

7. Memory-Saving Number Theoretic Transform

In [14], the authors tried to save the memory space by dividing n-iterations into two dimensions to implement an efficient architecture. We further apply this concept to divide n-iterations into four as shown in Figure 8. With the trade of time and area, we achieve the low memory space at the cost of increased time, which is tolerable for functioning Ring-LWE encryption system (Figure 9).

```
Input: Polynomial $a(x) \in Z_q[x]$ of degree $n-1$ and $n-th$ primitive root $w_n \in Z_q$ of unity
Output: Polynomial $A(x) \in Z_q[x] = NTT(a)$
  begin
    $A \leftarrow BitInverse(a)$;
    for $(m = 2 ; n/4 ; m = 4m)$ {
        $\omega_m \leftarrow m-th$ root of unity;
        $\omega \leftarrow \sqrt{\omega_m}$ or 1
        for $(j = 0 ; m/4 - 1)$ {
            for $(k = 0 ; n/4 - 1 ; m)$ {
              $(t_1, u_1) \leftarrow (A[k+j+m/4], A[k+j])$
              $(t_2, u_2) \leftarrow (A[k+m+j+m/4], A[k+m+j])$
              $t_1 \leftarrow \omega \bullet t_1$;
              $t_2 \leftarrow \omega \bullet t_2$;
$(A[k+j+m/4], A[k+j]) \leftarrow (u_1 - t_1, u_1 + t_1)$;
$(A[k+m+j+m/4], A[k+m+j]) \leftarrow (u_2 - t_2, u_2 + t_2)$;
          $mem[k+j] \leftarrow (A[k+j+m], A[k+j])$;
          $mem[k+j+m/4] \leftarrow (A[k+j+m/4], A[k+j+m/4])$;
          $mem[k+j+2m/4] \leftarrow (A[k+j+2m/4], A[k+j+m/4])$;
          $mem[k+j+3m/4] \leftarrow (A[k+j+3m/4], A[k+j+m/4])$;
              }
              $\omega \leftarrow \omega \bullet \omega_n$;
          }
    }
    $m \leftarrow n$;
    $k \leftarrow 0$;
    $\omega \leftarrow \sqrt{\omega_m}$ or 1
    for $(j = 0 ; m/2 - 1)$ {
        $(t_1, u_1) \leftarrow (A[j+m/2], A[j])$
        $t_1 \leftarrow \omega \bullet t_1$;
        $(A[j+m/2], A[j]) \leftarrow (u_1 - t_1, u_1 + t_1)$;
        $mem[j] \leftarrow (A[j+m/2], A[j])$;
        $\omega \leftarrow \omega \bullet \omega_m$;
    }
  }
```

Figure 8. Modified Memory-Saving Algorithm

Table 1. Implementation Results

Implementation	Parameters	Device	LUT/ BRAM18	Freq(Mhz)	Cycles/Time (us)	
					Enc	Dec
ECC	Binary-233	V5LX85T	18k/0	156	1.9k/12.3	1.9k/12.3
NTRU	NTRU-251	XCV1600E	27k/0	62.3	-/1.54	-/1.4
[14]	(256,7681,11,32)	V6LX75T	13k/2	313	6.6k/20.1	2.8k/9.1
Our work	(256,7681,11,32)	V6LX75T	14k/1	306	13k/42	5k/18

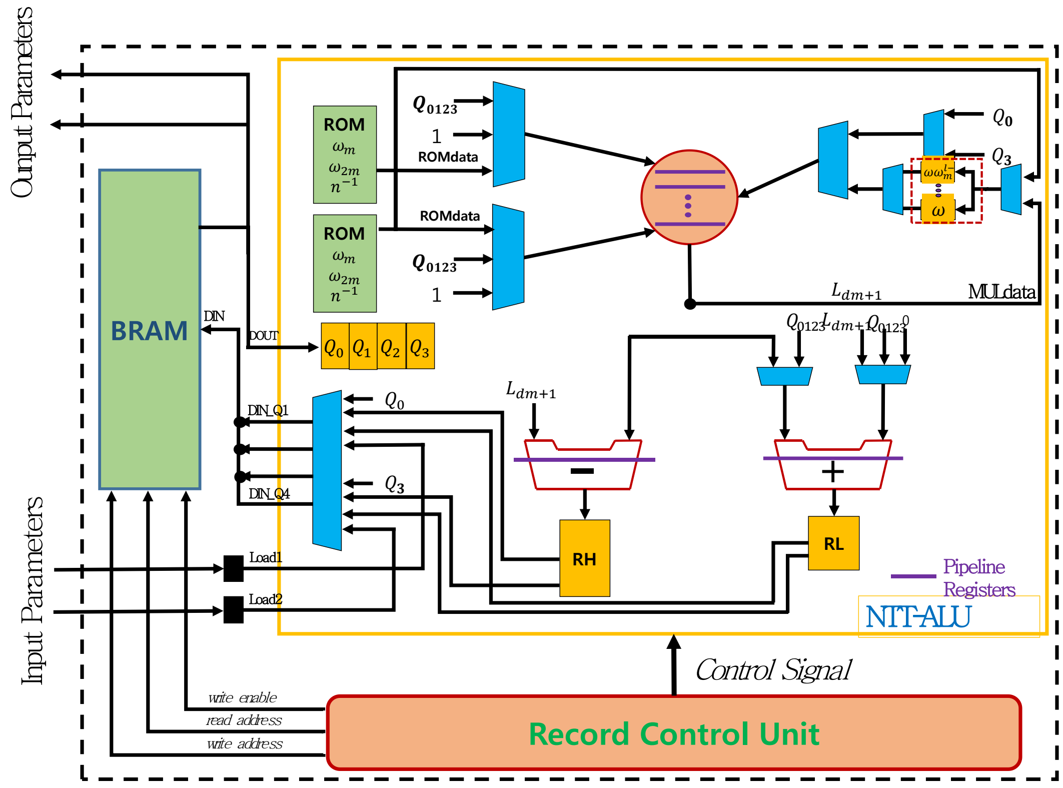

Figure 9. Ring-LWE Processor

8. Conclusion

The most important operation in implementing encryption systems is modular multiplication of large integers exceeding one million bits, which is multiplication operations for very large integers that can be obtained by repeatedly performing Fourier transform and integer Fourier transform, which is modular reduction, of the result of the multiplication operation. In the present study, the integer Fourier transform in Strassen method was improved to use the minimum capacity of memory so that it can be applied to the implementation of Ring-LWE on FPGA. In this work, we adopted an idea that can save the usage of memory so that we implemented a trade-off version of a Ring-LWE encryption processor.

Acknowledgments

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2014R1A1A2A16053925).

References

[1] E. Naone, "Homomorphic Encryption; Making cloud computing more secure", Magazine TR10, May **(2011)**: http://www.technologyreview.com/computing/37197/.
[2] C. Gentry, "A fully homomorphic encryption scheme", Ph.D. thesis, Stanford University, **(2009)**, http://crypto.stanford.edu/craig.
[3] R. A. Perlner and D. A. Cooper, "Quantum Resistant Public Key Cryptography: A Survey", 8th Symposium on Identity and Trust on the Internet (IDtrust 2009), Gaithersburg, MD, **(2009)** April, pp. 85-93.
[4] N. P. Smart and F. Vercauteren, "Fully homomorphic SIMD operations", IACR Cryptology ePrint Archive **(2011)**.

[5] C. Gentry and S. Halevi, "Implementing Gentry's Fully-Homomorphic Encryption Scheme", Advances in Ctyptology-EUROCRYPT 2011, **(2011)**, pp. 129-148.
[6] X. Chen, "The data protection of mapreduce using homomorphic encryption", 4th IEEE International Conference on Software Engineering and Service Science, **(2013)** May, pp. 419-421.
[7] D. Sowmya, "Cryptographic Cloud Storage with Hadoop Implementation," IOSR Journal of Computer Engineering, vol. 13, Issue 5, **(2013)** August, pp. 14-20.
[8] T.-W. Sze, "Schonhage-Strassen Algorithm with MapReduce for Multiplying Terabit Integers", Proceedings of the 2011 International Workshop on Symbolic-Numeric Computation, San Jose, California, **(2011)** June 7-9.
[9] M. Ayinala, M. Brown, K.K. Parhi, "Pipelined parallel FFT architectures via folding transformation", IEEE Transactions on VLSI Systems, vol. 20, no. 6, **(2012)** June, pp. 1068-1081.
[10] J. H. Cheon, "Batch Fully Homomorphic Encryption over the Integers", Advances in Cryptology – EUROCRYPT 2013, Lecture Notes in Computer Science, vol. 7881, (**2013)**, pp. 315-335
[11] W. Wang, "Exploring the Feasibility of Fully Homomorphic Encryption", IEEE Transactions on Computers, vol. PP, Issue 99, **(2013)**.
[12] 1363 working group of the C/MSC committee, IEEE P1363.1 D12 Draft Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices, **(2009)** March.
[13] Arvind, R. S. Nikhil, J. S. Emer and M. Vijayaraghavan, "Computer Architecture: A Constructive Approach, draft version", **(2012)**.
[14] S. S. Roy, "Compact Ring-LWE Cryptoprocessor", Lecture Notes in Computer Science, vol. 8731, **(2014)**, pp. 371-391.

Authors

Sangook Moon, He received his Bachelor's degree, the Master's degree, and the Ph.D. degree in the Department of Electrical and Electronic Engineering from Yonsei University, Seoul, Korea in 1995, 1997, and 2002, respectively. From 2002 to 2004, he was with the Hynix Semiconductor, Seoul, Korea, where he developed Bluetooth baseband SoCs. Since 2004 he has been with the Department of Electronic Engineering at Mokwon University, Daejeon, Korea, where he is currently serves as an Associate Professor. His research interests are in computer architecture, embedded systems, SoCs, data encryption, and computer arithmetic.