Quantum implementation of Encoding algorithm for HQC

Yujin Oh, Sejin Lim, Kyungbae Jang, and Hwajeong Seo *
Hansung University
{oyj0922, dlatpwls834, starj1023, hwajeong84}@gmail.com

Abstract

As quantum computers continue to advance rapidly, the security of traditional public-key encryption systems, which rely on challenges like factorization and discrete logarithms, is being weakened and potentially compromised by the Shor algorithm's ability to solve these problems in polynomial time. In anticipation of the quantum computing era, NIST has organized a competition to promote post-quantum cryptography, focusing on quantum-resistant algorithms to ensure secure encryption even in the presence of powerful quantum computers. In the quantum computing environment, analyzing encryption through quantum circuits is crucial for evaluating the security strength of cryptographic systems. This paper proposes a partial implementation of the encoding process for the PKE (Public Key Encryption) version of HQC (Hamming Quasi-Cyclic), a fourth-round candidate algorithm in NIST's Post-Quantum Cryptography competition. Specifically, we focus on the implementation of quantum circuits for key operations involving binary field arithmetic and encoding operations using shortened Reed-Solomon codes. Additionally, we estimate the necessary quantum resources for this implementation.

Keywords: Quantum Circuit, HQC, code-based algorithm, Reed-Solomon algorithm.

1 Introduction

Quantum computers, leveraging the properties of superposition and entanglement, are expected to bring significant advantages in fields such as machine learning, finance, and optimization due to their exceptionally fast computational speeds. As the development of quantum computers intensifies, active research is also underway in the field of quantum cryptanalysis. Quantum algorithms for quantum cryptanalysis include the Grover algorithm[1] and the Shor algorithm[2]. The Grover algorithm reduces the complexity of symmetric key search to the square root. Moreover, from the perspective of current public-key cryptography (e.g., RSA, ECC), Shor's algorithm, which operates on quantum computers, can solve mathematical problems such as factorization and discrete logarithms—fundamental to the security of public-key cryptography—in polynomial time. Therefore, it is anticipated that the security of existing cryptographic systems is approaching a critical point with the advent of quantum computers. In preparation for the quantum computing era, NIST is organizing a Post-Quantum Cryptography Standardization competition to develop new public-key cryptography standards that can maintain security even in the presence of quantum computers. In the quantum computing environment, it is crucial to implement encryption as quantum circuits to assess the cryptographic strength[3, 4]. This paper proposes a partial implementation of the encoding process for the PKE (Public Key Encryption) version of HQC (Hamming Quasi-Cyclic)[5], a fourth-round candidate algorithm in NIST Post-Quantum Cryptography competition. We suggest the quantum circuit implementation for the necessary binary arithmetic and shortened Reed-Solomon[6] code operations during the encoding process, along with estimating the required resources for this implementation.

The 7th International Conference on Mobile Internet Security (MobiSec'23), Dec. 19-21, 2023, Okinawa, Japan, Article No.W-1

*Corresponding author: CryptoCraft Lab, Hansung University, 116, Samseongyo-ro 16-gil, Seongbuk-gu, Seoul, Republic of Korea

1.1 Our Contribution

Contributions of this paper are:

1. **Quantum implementation of Binary Field Operations.** We employ the method presented in [7] to realize an optimized binary field multiplication, which is used in encoding of HQC.

2. **Quantum implementation of the shortened Reed-Solomon and encoding algorithm.** We implement the encoding process of HQC. Additionally, we present for the first time a quantum circuit of the Reed-Solomon algorithm used in encoding of HQC.

3. **Quantum circuit cost estimation.** We estimate the quantum resources required to implement the circuit.

2 Background

2.1 HQC

HQC (Hamming Quasi-Cyclic) is a code-based cryptography that combines Hamming codes with randomly generated Quasi-Cyclic codes. Code-based cryptography is based on the principle of intentionally injecting errors into messages and allowing only users who are aware of the errors to restore the message using error-correcting codes. This is based on the challenging problem of syndrome decoding, which is NP-complete. Additionally, Quasi-Cyclic ensures that certain relationships cycle within the matrix, optimizing operations. By utilizing this, it is possible to know the entire matrix by only storing the first row, efficiently reducing the key size. Among the code-based encryption candidates in the fourth round, such as Classic McEliece, BIKE, and HQC. HQC has larger parameters than BIKE but smaller parameters than Classic McEliece. While many code-based cryptography have vulnerabilities due to the process of modifying the randomness of the Generator matrix, HQC uses the Generator matrix code as it is publicly available, indicating that there are no issues with the code-based structure. This is a distinctive and significant advantage of HQC.

The PKE version of HQC consists of three main processes: key generation ($sk = (x, y)$), encryption (encoding) ($ct = (u, v)$), and decryption (decoding). In the key generation step, a primitive polynomial in the binary field is $(X^n - 1)/(X - 1)$, where q must be a prime number. For instance, in the case of hqc-128, n is 17669, and when this is simplified, it becomes $(x^{n-1} + x^{n-2} + ... + x + 1)$, ultimately resulting in $\mathbb{F}_{2^{17668}}/(x^{17668} + x^{17667} + ... + x + 1)$ being used.

Furthermore, in the encoding operation, which produces the ciphertext by multiplying the message with the Generator matrix, a shortened Reed-Solomon code is utilized in the binary field multiplication process. Here, the binary field size is 8, and the primitive polynomial of $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x^2 + 1)$ is employed.

Due to the significant size of the error term introduced during encoding in HQC, decoding on its own is not feasible. Only users possessing the private key can effectively perform decoding by reducing the error term. Decoding may still fail depending on probability, but the authors of the HQC paper have demonstrated through detailed and precise mathematical analysis that the probability of failure is negligible. Therefore, HQC can be considered to provide a high level of security.

3 Proposed Method

Encoding is an operation that involves calculating $u \leftarrow r_1 + h \cdot r_2$ and $v \leftarrow m \cdot G + s \cdot r_2 + e$ for r_1 and r_2 existing in the same binary field. In this multiplication operation of $m \cdot G$, unlike other operations, a shortened Reed-Solomon code is used.

In this paper, due to the imperfect implementation of u and v used as ciphertext, they are separately implemented and the separated resources are estimated.

3.1 Implementation of Binary Field Operations

In encoding, addition and multiplication operations are performed in the binary field. For HQC-128, operations are carried out in the binary field of $\mathbb{F}_{2^{17668}}$. However, due to the limitations of quantum simulation, we implement the quantum circuit by reducing the binary field to 12. The primitive polynomial used here is for $\mathbb{F}_{2^{12}}$. The XOR operation used in addition is implemented in the quantum circuit with CNOT gates, allowing it to have a depth of 1 by using CNOT gates equivalent to the field size. However, multiplication operations performed through AND and XOR operations in binary field arithmetic have high computational complexity. In a binary field of size, Schoolbook multiplication requires AND operations. In quantum circuits, AND operations are implemented using Toffoli gates, which come with high implementation costs. Therefore, several studies have been conducted to optimize multiplication operations in binary field arithmetic [8, 9, 10, 7]. In this paper, we applied the most recent quantum multiplication technique [7] that optimizes Toffoli-depth to 1, implementing binary field multiplication. This multiplication technique recursively applies the Karatsuba algorithm [11], which reduces the complexity of multiplication by performing additional additions, and optimizes Toffoli-depth to 1 regardless of the field size by removing dependencies between multiplication factors through additional qubit allocation. This allows us to perform multiplication with a very small overall depth. Compared to previous studies [8, 9, 10], it requires the most qubits, but it has the best performance in terms of Toffoli gate count, Toffoli-depth, and Full-depth. Although there is a trade-off relationship between the number of qubits and depth, for HQC, since operations are performed in a very large binary field, the multiplication technique [7] with Toffoli-depth always equal to 1 is the most suitable for this implementation, regardless of the field size. Furthermore, the modular operation within the multiplication varies depending on the primitive polynomial of the field (even for the same field, the primitive polynomial used may differ depending on the encryption scheme). The binary arithmetic of HQC employs a primitive polynomial $(x^{11} + x^{10} + ... + x + 1)$(Derived from $(X^n - 1)/(X - 1)$), making the modular operation using it straightforward. The modular operation of multiplication in $\mathbb{F}_{2^{12}}$ is shown in Algorithm 1.

Algorithm 1 Quantum circuit implementation of modular operation in multiplication.

```
Input: $result[0:23]$
Output: $result[0:11]$
for $i = 0$ to 12 do
    $result[i] \leftarrow \mathrm{CNOT}(result[12], result[i])$
end for
for $i = 0$ to 10 do
    $result[i] \leftarrow \mathrm{CNOT}(result[13+i], result[i])$
end for
return $result[0:11]$
```

Using these binary field arithmetic, we can calculate $u \leftarrow r_1 + h \cdot r_2$ and $s \cdot r_2 + e$ of v. The operation $m \cdot G$ involves more than simple multiplication, and one of the method used for this operation is Shortened Reed-Solomon. Thus, we utilize arithmetic in the binary field of partially $\mathbb{F}_{2^{12}}$ to implement the operation $u \leftarrow r_1 + h \cdot r_2$ and estimate the resources(Reed-Solomon involves operations in the different field, which will be explained in Section 3.2).

3.2 Implementation of shortened Reed-Solomon

The encoding quantum circuit for the shortened Reed-Solomon code is identical to Algorithm 2. In this algorithm, the crucial operation involves binary field operations that multiply the coefficient matrix of the publicly available RS-S1 polynomial by the message vector. The constant values used in the algorithm are denoted as $K = 16$, $G = 31$, and $N1 = 46$ respectively. The coefficients of the polynomial matrix are represented as qubit arrays (Line 1 and 2 in Algorithm 2). Since only 30 out of 31 coefficients are utilized in the encoding operation, calculations were performed up to $G - 1$.

The function Copy_gate_value(Line 7-9 in Algorithm 2) involves copying the gate_value to handle 30 subsequent binary field multiplications in parallel in the next step. The process of copying values was divided into parallel segments, resulting in 29 parallelized value copy operations. Therefore, we can optimize the depth in copy operations.

As a result, the binary field multiplication carried out is optimized with a Toffoli-depth of 1, utilizing the multiplication [7] (implementing Karatsuba algorithm recursively). Since this operation is repeated a total of 16 times, the Toffoli-depth is 16. The variable *tmp* serves the purpose of holding the result qubit, as the multiplication operation is implemented in-place. Furthermore, to reuse the qubit array (*copy*$[j]$), we utilize the reverse operation (Line 19-21 in Algorithm 2). The reverse operation employs only CNOT gates with minimal impact on full depth, and even these are further parallelized, resulting in an even more negligible influence on the full depth.

4 Performance

In this section, we will evaluate the quantum resources needed for our proposed quantum circuit. We employed the quantum programming tool ProjectQ [9] for this purpose. Utilizing ProjectQ enables us to verify the outcomes of our implemented quantum circuit and provides us with an estimate of the necessary quantum resources.

Table 1 illustrates the quantum resource costs for the essential operations of addition and multiplication, implemented in the binary field $\mathbb{F}_{2^8}$ of the Shortened Reed-Solomon algorithm and $\mathbb{F}_{2^{12}}$, which is the reduced binary field, respectively. In Table 1, By utilizing the multiplier from [7], it can be observed that the Toffoli-depth remains 1 regardless of the binary field size. Table 2 shows the quantum resources for implementing $u \leftarrow r_1 + h \cdot r_2$, which is the one of being the ciphertext. In Table 2, both multiplication and addition are used only once. Since we employ the approach mentioned in Section 3.1, the Toffoli depth is 1 during the multiplication operation. Furthermore, the full depth is estimated to be the sum of the full depth for both addition and multiplication as calculated in Table 1 (the same applies to the number of CNOT gates). Table 3 presents the quantum resource costs of implementing the shortened Reed-Solomon code using a quantum circuit. As explained earlier, it can be observed that the implementation has optimized Toffoli-depth and depth, resulting in a lower depth compared to the number of gates used.

Algorithm 2 Quantum circuit implementation of shortened Reed-Solomon.

```
Input: 8-qubit array $msg[K]$ , $RS\_POLY[G-1]$, $cdw[N1]$, $gate\_value$, $copy[G-2]$, $ancilla$
    qubits array $ac[30]$
Output: $cdw$
for $i = 0$ to $G-1$ do
    $RS\_POLY[i] \leftarrow$ CNOT8($RS\_COEFS, RS\_POLY[i]$)
end for
for $i = 0$ to $K$ do
    $gate\_value[i] \leftarrow$ CNOT8($cdw[N1-K-1], gate\_value[i]$)
    $gate\_value[i] \leftarrow$ CNOT8($msg[K-1-i], gate\_value[i]$)
    for $j = 0$ to $G-2$ do
        $copy[j] \leftarrow$ Copy_gate_value ($gate\_value[i], copy[j]$)
    end for
    $tmp[0] \leftarrow$ Multiplication($gate\_value[i], RS\_POLY[0], ac[0]$)
    for $j = 1$ to $G-1$ do
        $tmp[j] \leftarrow$ Multiplication($copy[i], RS\_POLY[j], ac[j]$)
    end for
    for $j = N1-K-1$ to $0$ do
        $cdw[j] \leftarrow$ CNOT8($cdw[j-1], cdw[j]$)
        $cdw[j] \leftarrow$ CNOT8($tmp[j], cdw[j]$)
    end for
    $cdw[0] \leftarrow$ CNOT8($tmp[0], cdw[0]$)
    for $j = 0$ to $G-2$ do
        $copy[j] \leftarrow$ Copy_gate_value ($gate\_value[i], copy[j]$)
    end for
end for
for $i = 0$ to $K$ do
    $cdw[i] \leftarrow$ CNOT8($msg[i], cdw[i+30]$)
end for
return $cdw$
```

Table 1: Required quantum resources for Binary Field Operations

Field	Arithmetic	Qubits	#CNOT	#Toffoli	Toffoli depth	Full depth
$\mathbb{F}_{2^8}$	Multiplication	81	164	27	1	26
$\mathbb{F}_{2^{12}}$	Addition	24	12	-	-	1
	Multiplication	162	495	54	1	32

5 Conclusion

In this paper, we propose a quantum circuit implementation for the core operations involving binary field arithmetic and shortened Reed-Solomon code in the encoding process of the HQC PKE version, a fourth-round candidate algorithm in the NIST competition. We also perform resource estimation, with a focus on optimizing the multiplication operation in the binary field to reduce the cost of quantum resources. Additionally, implementing the shortened Reed-Solomon code in the quantum circuit using the parameters used in HQC and measuring the resources

Table 2: Required quantum resources for implementing $u \leftarrow r_1 + h \cdot r_2$

Field	Operation	Qubits	#CNOT	#Toffoli	Toffoli depth	Full depth
$\mathbb{F}_{2^{12}}$	$u(r_1 + h \cdot r_2)$	174	507	54	1	33

Table 3: Required quantum resources for Shortened Reed-Solomon quantum circuit implementation

shortened Reed-Solomon	Qubits	#CNOT	#Toffoli	Toffoli depth	Full depth
HQC-128	28,696	94,320	12,960	16	545

is significant. It is expected that the presented quantum circuit will contribute to the security analysis of HQC. In the future, we plan to complete the encoding operations by implementing not only Reed-Solomon codes but also other operations, and we will also implement the key generation and decoding to complete the entire quantum circuit of HQC. Furthermore, we plan to adjust the range for feasible simulations and expand the binary field to its maximum extent.

6 Acknowledgment

This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (<Q|Crypton>, No.2019-0-00033, Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity, 75%) and this work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-00627, Development of Lightweight BIoT technology for Highly Constrained Devices, 25%).

References

[1] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[2] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[3] Thomas Häner, Samuel Jaques, Michael Naehrig, Martin Roetteler, and Mathias Soeken. Improved quantum circuits for elliptic curve discrete logarithms. In *International Conference on Post-Quantum Cryptography*, pages 425–444. Springer, 2020.

[4] Martin Roetteler, Michael Naehrig, Krysta M Svore, and Kristin Lauter. Quantum resource estimates for computing elliptic curve discrete logarithms. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 241–270. Springer, 2017.

[5] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loıc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, and IC Bourges. Hamming quasi-cyclic (hqc). *NIST PQC Round 4*, 2(4):13, 2018.

[6] Nicolas Aragon, Philippe Gaborit, and Gilles Zémor. Hqc-rmrs, an instantiation of the hqc encryption framework with a more efficient auxiliary error-correcting code. *arXiv preprint arXiv:2005.10741*, 2020.

[7] Kyungbae Jang, Wonwoong Kim, Sejin Lim, Yeajun Kang, Yujin Yang, and Hwajeong Seo. Optimized implementation of quantum binary field multiplication with toffoli depth one. In *International Conference on Information Security Applications*, pages 251–264. Springer, 2022.

[8] Donny Cheung, Dmitri Maslov, Jimson Mathew, and Dhiraj K Pradhan. On the design and optimization of a quantum polynomial-time attack on elliptic curve cryptography. In *Theory of Quantum Computation, Communication, and Cryptography: Third Workshop, TQC 2008 Tokyo, Japan, January 30-February 1, 2008. Revised Selected Papers 3*, pages 96–104. Springer, 2008.

[9] Shane Kepley and Rainer Steinwandt. Quantum circuits for f ˍ 2ˆ n f 2 n-multiplication with subquadratic gate count. *Quantum Information Processing*, 14:2373–2386, 2015.

[10] Iggy Van Hoof. Space-efficient quantum multiplication of polynomials for binary finite fields with sub-quadratic toffoli gate count. *arXiv preprint arXiv:1910.02849*, 2019.

[11] Anatolii Karatsuba. Multiplication of multidigit numbers on automata. In *Soviet physics doklady*, volume 7, pages 595–596, 1963.