Optimized Implementation of Quantum Binary Field Multiplication with Toffoli Depth One*

Kyungbae Jang, Wonwoong Kim, Sejin Lim,
Yeajun Kang, Yujin Yang, and Hwajeong Seo**

IT Department, Hansung University, Seoul, South Korea,
{starj1023, dnjsdndeee, dlatpwls834,
etus1211, yujin.yang34, hwajeong84}@gmail.com

Abstract. Shor's algorithm models discrete logarithms on binary elliptic curves and provides polynomial-time solutions. One of major overheads in applying Shor's algorithm is implementing binary elliptic curve arithmetic in quantum circuits. Among operations of elliptic curves over binary fields, the multiplication is essential and cost-critical even in the quantum field.
In this paper, we aim to optimize quantum binary field multiplication. Previous works on quantum multiplication focused on minimizing the number of Toffoli gates or qubits. In contrast, our work presents strategies for optimizing Toffoli depth and full depth, which are key factors in the Noisy Intermediate-Scale Quantum (NISQ) era. To achieve our goal, Karatsuba multiplication using divide-and-conquer approach is adopted. In a nutshell, we present an optimized quantum multiplication with Toffoli depth one. Furthermore, under the influence of the optimized Toffoli depth, the full depth is naturally reduced.
In order to show the effectiveness of proposed method, the performance is evaluated by various metrics, such as, qubits, quantum gates, depth, and qubits-depth product. To the best of our knowledge, this is the first study on quantum multiplication that optimizes Toffoli depth and full depth.

Keywords: Shor's Algorithm · Binary Elliptic Curves · Quantum Multiplication · Toffoli depth.

1 Introduction

The large-scale quantum computers that will emerge in the near future are considered to be a major threat to the cryptography community. Quantum com-

* This work was partly supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00264, Research on Blockchain Security Technology for IoT Services, 50%) and this work was partly 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, 50%)
** Corresponding Author

puters with quantum algorithms can model and solve problems for which cryptographic algorithms are based on their security. Quantum computers capable of running the Shor algorithm [1] can solve factorization and discrete logarithm problems, which are the challenges of Rivest-Shamir-Adleman (RSA) and Elliptic Curve Cryptography (ECC). As such, it is well known that the Shor's algorithm violates the security of public key cryptography. In this current situation, the analysis of potential quantum computer attacks on public key cryptography should be considered for a robust security system.

In [2], Häner et al. presented that $2n + 2$ qubits are required to apply the Shor algorithm to RSA with an n-bit key. As another quantum analysis of RSA, Gideny presented that RSA with an n-bit key requries $2n + 1$ qubits when applying the Shor algorithm [3].

In Asiacrypt'17 [4], Roetteler et al. estimated the quantum resources required to solve the elliptic curve discrete logarithms, and showed that ECC is more vulnerable to quantum computers than RSA. In PQCrypto'20 [5], Häner et al. reduced the number of qubits and circuit depth by improving the results of [4]. In [4,5], prime curves are targeted, and when solving discrete logarithm problems, optimizing scalar multiplication on elliptic curves, which is the most expensive, is key in determining the cost.

In CHES'20, Banegas et al. estimated the cost of solving discrete logarithm problems on binary curves and showed that it is possible to attack with fewer qubits and depth than prime curves. In the work of Banegas et al., Van Hoof's work [6] was used to implement quantum multiplication of binary fields. Van Hoof's quantum multiplication is based on the Karatsuba algorithm and features a space-efficient implementation that reduces the number of qubits, while reducing the number of Toffoli gates.

As it can be seen from previous works, optimizing binary field multiplication on a quantum computer is an essential building block for high-performance quantum cryptanalysis. Most recent works about quantum multiplication focus on reducing the number of qubits or Toffoli gates, but do not give much consideration to the depth of the circuit [6,7,8,9,10]. A few years ago, the number of available qubits in a quantum computer was small. But today's quantum computers are no longer considered small. Also, the upcoming quantum computers are obviously not small, as seen in IBM's quantum computer development roadmap[1]. Toffoli depth is an important metric for quantum computing where errors should be controlled [11], and full depth determines the circuit's operating time [12]. Regarding the importance of full depth, in the quantum security requirements[2] of the National Institute of Standards and Technology (NIST), the complexity of quantum attacks is calculated as the product of the number of gates and the full depth (the number of qubits is not included) [13].

The main target of this paper is to optimize binary field multiplication on a quantum computer. We focus on Toffoli depth and full depth, and propose a

[1] https://research.ibm.com/blog/ibm-quantum-roadmap

[2] https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf

quantum binary filed multiplication that is optimized with Toffoli depth one and has the lowest full depth. The proposed multiplication requires additional qubits, and the number of qubits is still an important factor in quantum computers. In this trade-off, the proposed work is evaluated by various metrics, including the number of qubits, number of quantum gates, depth, product of Toffoli depth, and number of qubits.

In most quantum-related studies, quantum circuits are simulated on a classical computer using quantum programming tools, such as ProjectQ [14], Qiskit [15], and Q# [16] (i.e. logical level with no errors). This is because it is difficult to access real quantum computers and errors that occur must be considered (i.e. physical level with errors). In this work, the quantum programming tool ProjectQ is used to implement quantum circuits and estimate detailed quantum resources.

1.1 Our Contribution

The contribution of this paper is summarized as follows:

1. **Optimized quantum binary field multiplication using the Karatsuba algorithm.** (Sections 3.1 and 3.2). We present the optimized quantum binary field multiplication with Toffoli depth one. Since all Toffoli gates operate in parallel, all products are generated, simultaneously. The full depth of quantum multiplication is mostly derived from Toffoli gates. The full depth of proposed quantum multiplication is naturally reduced.
2. **Efficient quantum circuit implementation techniques**. Additionally, efficient implementation techniques are presented together. How to offset the overhead of increasing qubits in our quantum multiplication (Section 3.3), quantum multiplication of T-depth one (Section 3.4), and optimal modular reduction (Section 3.5) are discussed.
3. **Optimized primitives for quantum cryptanalysis of ECC.** Our work can be used to optimize quantum cryptanalysis of elliptic curves over binary fields. Quantum binary field multiplication is essential for the Shor algorithm to solve discrete logarithm problems on binary elliptic curves. Our work shows the best trade-off between Toffoli depth and number of qubits.

2 Background

2.1 Binary Field Multiplication

Multiplication of $\mathbb{F}_{2^n}$ performs n-bit polynomial multiplication and modular reduction by irreducible polynomial N. The multiplication of f and g of $\mathbb{F}_{2^n}$ is as follows.

$$h = f \cdot g \bmod N \tag{1}$$

For the generated product of f and g, the modular reduction is performed over n bits in length. As a result, h, the product of f and g, becomes an element of $\mathbb{F}_{2^n}$.

2.2 Karatsuba Multiplication

It is well known that the Karatsuba algorithm [17] can reduce the complexity of multiplication by using addition operations. For the multiplication of polynomials f and g of size n (i.e. $h = f \cdot g$), the Karatsuba algorithm divides the two input polynomials f and g into the size of $s = n/2$ as follows:

$$\begin{aligned} f &= f_1 x^s + f_0 \\ g &= g_1 x^s + g_0 \end{aligned} \tag{2}$$

After two input polynomials (f and g) are divided as above, the Karatsuba multiplication is done as:

$$f_0 \cdot g_0 + \{(f_0 + f_1) \cdot (g_0 + g_1) + f_0 \cdot g_0 + f_1 \cdot g_1\} x^s + f_1 \cdot g_1 x^{2s} \tag{3}$$

Using the Karatsuba algorithm, the multiplication complexity of $O(n^2)$ is reduced to $O(n^{\log_2 3})$ by performing addition operations.

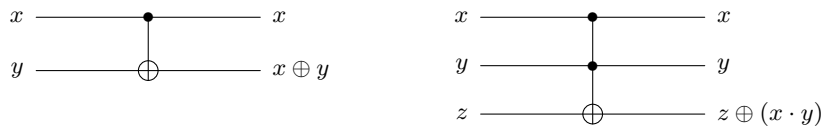
2.3 Quantum Gates

This Section briefly describes the CNOT and Toffoli gates for implementing binary field multiplication consisting of XOR and AND operations.

Figure 1(a) is a quantum CNOT gate that can replace the classical XOR operation. In the CNOT gate, one control qubit is used to determine the value of the target qubit ($\text{CNOT}(x, y) = (x, x \oplus y)$).

Figure 1(b) is a quantum Toffoli gate that can replace the classical AND operation. In the Toffoli gate, two control qubits are used to determine the value of the target qubit ($\text{Toffoli}(x, y, z) = (x, y, z \oplus (x \cdot y))$). In practice, the Toffoli gate in Figure 1(b) has a high cost as it is implemented as a combination of quantum gates [18].

Quantum gates in reversible quantum computing can compute a unique input from a given output. In quantum circuits, the reverse operation means returning from the output to the input by reversing the previous quantum gates. When the reverse operation is performed on the output of the quantum gates in Figure 1, it returns to the input.



(a) CNOT gate (b) Toffoli gate

Fig. 1: Reversible quantum gates.

3 Optimized Implementation of Quantum Binary field Multiplication

In quantum multiplication, the majority of the cost is spent on Toffoli gates to compute the products (i.e. performing AND operations). Karatsuba algorithm, which reduces the number of multiplication operations, is obviously an efficient technique in quantum computers.

We propose a special quantum Karatsuba multiplication that reduces the number of Toffoli gates, Toffoli depth, and circuit depth. With this technique, we present a quantum binary field multiplication of Toffoli depth one that performs all multiplication operations, simultaneously. The proposed quantum multiplication reduces the full depth, since the Toffoli depth in multiplication has an impact on counting the full depth.

3.1 Parallel Quantum Multiplication with the Karatsuba Algorithm

Let h be the product of two polynomials f and g of size n (i.e. $h = f \cdot g$). As illustrated in Figure 2, $f \cdot g$ with Schoolbook multiplication (which is general) requires n^2 Toffoli gates.

In the proposed method, we apply the Karatsuba algorithm once to reduce the size of the multiplication, which is called Level-1. In Level-1, it is divided into three multiplications $f_0 \cdot g_0$, $f_1 \cdot g_1$, and $(f_0 + f_1) \cdot (g_0 + g_1)$. The size of each multiplication is reduced to $(n/2)$ and $3 \cdot (n/2)^2$ Toffoli gates are required. This result reduces the use of Toffoli gates from n^2 to $3 \cdot (n/2)^2$, but may not be completely parallel, which means the three multiplications are not performed, simultaneously. In the Level-1 layer of Figure 2, multiplications $f_0 \cdot g_0$ (low part) and $f_1 \cdot g_1$ (high part) are performed, simultaneously, but $(f_0 + f_1) \cdot (g_0 + g_1)$ in the rectangle (middle part) is performed sequentially after the previous multiplications are finished. Operands of the multiplication of the middle part (f_0+f_1, g_0+g_1) are overwritten in f_0, g_0 or f_1, g_1 (i.e. $f_0 = f_0+f_1, g_0 = g_0+g_1$), and it is possible only after the multiplication of the high part and the low part is finished, as in [6,8,9].

Before multiplications, we allocate clean qubits (rectangle of Level-1 layer in Figure 2) and independently prepare the middle part operands $(f_0 + f_1)$ and $(g_0 + g_1)$ using CNOT gates. Preparing this middle part requires n clean qubits and $2n$ CNOT gates. (i.e. $2/n$ clean qubits $= f_0 + f_1$ and $2/n$ clean qubits $= g_0 + g_1$) Three multiplications of low, middle, and high become independent of each other and can be performed, simultaneously. As a result, in Level-1, quantum multiplication of reduced Toffoli depth is presented. If the Karatsuba multiplication is terminated, additions between the generated products $f_0 \cdot g_0$, $f_1 \cdot g_1$, and $(f_0 + f_1) \cdot (g_0 + g_1)$ are performed using CNOT gates to complete the multiplication. Furthermore, we apply the Karatsuba algorithm recursively to optimize the Toffoli depth and full depth.

The quantum resources of multiplication according to the Karatsuba Level is shown in Table 1. In practice, the Toffoli gate is implemented as a combination

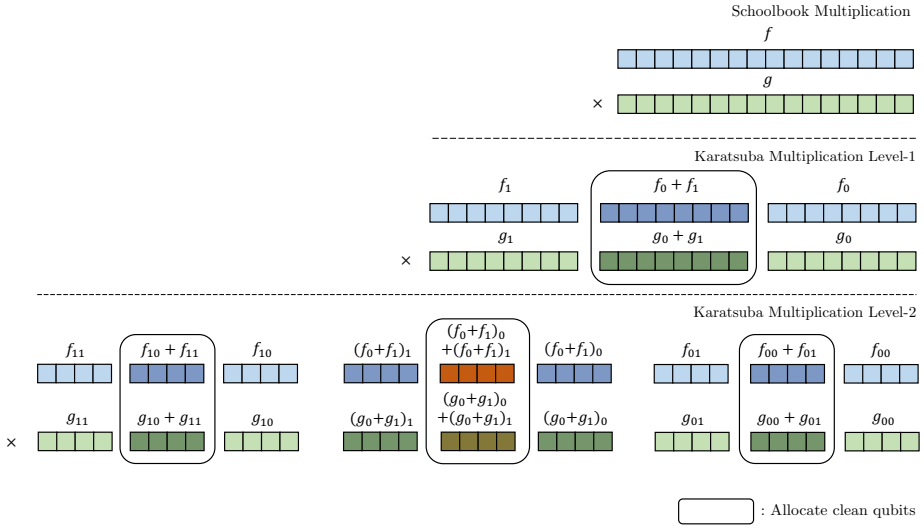


Fig. 2: Overview of the proposed method.

of several quantum gates. In this work, we decompose the Toffoli gate into 7 T gates + 8 Clifford gates, T-depth 4, and full depth 8 following the frequently adopted method [18]. However, for a bird's eye view of comparison, in Table 1, without decomposing the Toffoli gate, only the full depth is assumed to be 8. In subsequent resource estimates, we completely decompose the Toffoli gates.

Modular reduction is not included in Table 1 and will be described in Section 3.5, because the complexity slightly different depending on the irreducible polynomial of the field and does not significantly affect the overall cost.

Table 1: Quantum resources required for each Karatsuba level of multiplication.

Field size 2^n	#CNOT	#Toffoli	Toffoli depth	#Qubits	Full depth
Schoolbook	·	n^2	$3n-2$	$4n-1$	$8\cdot(3n-2)$
Karatsuba Level-1	$5n-4$	$3\cdot(n/2)^2$	$3n/2-2$	$3\cdot(2n-1)$	$8\cdot(3n/2-2)+5$
Karatsuba Level-2	$(5n-4)+$ $3\cdot(5n/2-4)$	$3^2\cdot(n/2^2)^2$	$3n/2^2-2$	$3^2\cdot(n-1)$	$8\cdot(3n/2^2-2)+10$
Karatsuba Level-3	$(5n-4)+3\cdot(5n/2-4)$ $+9\cdot(5n/4-4)$	$3^3\cdot(n/2^3)^2$	$3n/2^3.-2$	$3^3\cdot(n/2-1)$	$8\cdot(3n/2^3-2)+15$

3.2 Toffoli Depth Optimization with Recursive Karatsuba Algorithm

In Level-2, the Karatsuba algorithm is applied to each of the three multiplications divided by Level-1. Similar to Level-1, in Level-2 there is a dependency on multiplications for middle parts. Thus, we allocate $3\times$ clean qubits and prepare the $3\times$ middle parts using CNOT gates again (rectangles of Level-2 layer in Figure 2). Preparing these middle parts requires $3 \cdot (n/2)$ clean qubits and $3n$ CNOT gates. As a result, nine multiplications become completely independent of each other and are performed, simultaneously. In Level-2, Toffoli gates and Toffoli depth that were reduced in Level-1 are reduced once more to $3^2 \cdot (n/2^2)^2$ for Toffoli gates and $(3n/2^2 - 2)$ for Toffoli depth.

In this way, the Karatsuba algorithm is applied recursively until it is divided into multiplications of size one, and the dependencies of all multiplications are released. Then, we can perform quantum multiplication of Toffoli depth one by generating all products in parallel. Finally, the proposed quantum multiplication circuit achieves the best performance of Toffoli depth one, providing high performance even for full depth.

According to field size 2^n, the required Karatsuba Level for multiplication of Toffoli depth one is different. This is calculated as Level-$\log_2 n$ according to field size 2^n. For example, Level-2 for field size $n = 4$ and Level-3 for field size $n = 8$. In Table 2, we compare quantum resources required for multiplication of Toffoli depth one by field size. As the Karatsuba level increases, there is a trade-off between Toffoli gates, depth, and number of qubits (See Table 1). Proposed quantum multiplication provides the best performance at the highest Karatsuba Level, but implementation designer can flexibly set the Karatsuba Level by considering this trade-off.

Table 2: Quantum resources required for multiplication of Toffoli depth one.

Field size 2^n	Karatsuba Level	#CNOT	#1qCliff	#T	T-depth※	#Qubits	Full depth
$n = 4$	2	88	18	63	4	27	17
$n = 8$	3	300	54	189	4	81	23
$n = 16$	4	976	162	567	4	243	28

※: Toffoli depth one has a T-depth of four.

3.3 Recycling Qubits with Reverse Operation

In our quantum multiplication, new qubits for the middle parts are allocated each time the Karatsuba algorithm is applied, which is obviously an overhead. However, these qubits can be initialized by the reverse operation of the CNOT gates used previously in the middle parts.

The initialization of the qubits is performed after the operation of the Toffoli gates that generate the products. If all products are generated at once in the lowest layer, we initialize (cleaning) the qubits from the lower layer to the upper layer by performing the reverse of the operations that prepared the middle parts. As a result, the qubits allocated for the middle parts are initialized to zero (i.e. clean state).

This initialization, which cleans the qubits, can be used if the multiplication is merged with other operations (i.e. not stand-alone multiplication). This means that in subsequent multiplications, the previously initialized qubits can be reused without allocating new qubits for middle parts. Besides multiplication, these initialized qubits can be reused for other operations that require new clean qubits. Since the multiplication is primitive in cryptography, the qubit initialization is effective. Also, we found that this reverse operation for initialization does not increase the circuit depth, as it can be done while the multiplication is being performed (when combining products or modular reduction). With this technique, we can effectively offset the overhead of qubits in our quantum multiplication. When qubits are reused in subsequent multiplications, $17, 43$, and 113 qubits are required for $n = 4, 8,$ and 16, respectively (reduced from $27, 81$, and 243 qubits).

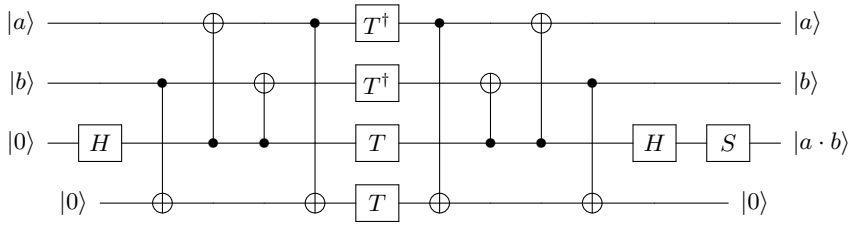
3.4 Optimized Implementation of Quantum Multiplication of T-Depth One

For T-depth optimization, we apply the quantum AND gate of T-depth one introduced in [19] instead of the Toffoli gate. The AND gate illustrated in Figure 3(a) uses one ancilla qubit and has a T-depth of one. In the AND gate, an ancilla qubit is initialized to zero. It can be reused in the next AND gate. However, without reuse, we allocate a new ancilla qubit for each AND gate to operate all AND gates, simultaneously.

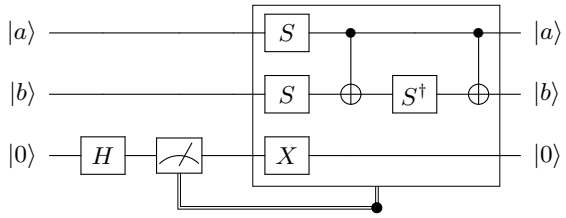
Since all Toffoli gates operate in parallel (i.e. Toffoli depth is one) in the proposed quantum multiplication, we allocate the same number of ancilla qubits as the number of Toffoli gates required. As a result, all AND gates are successfully operated in parallel, and a quantum multiplication of T-depth one is implemented. AND gates require additional qubits, but these qubits are initialized at the end. They can be reused in subsequent multiplications or other operations that require clean qubits (Similar to Section 3.3). Table 3 shows the quantum resources required for quantum multiplication of T-depth one when the AND gate is adopted.

3.5 Quantum Modular Reduction

In quantum multiplication, the modular reduction can be customized according to the irreducible polynomial of the field. In this Section, we customize modular reduction for $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$ (used in AES) and analyze the required quantum resources.



(a) Quantum AND gate.



(b) Quantum AND† gate.

Fig. 3: Quantum AND gate of T-depth one.

Table 3: Quantum resources required for multiplication of T-depth one using AND gate.

Field size 2^n	Karatsuba Level	#CNOT	#1qCliff	#T	T-depth	#Qubits	Full depth
$n = 4$	2	106	27	36	1	36	16
$n = 8$	3	354	81	108	1	108	22
$n = 16$	4	1138	243	324	1	324	27

In [8], after generating products, the authors omit the combining step and compute according to linear combinations of products after modular reduction. That is, combining and modular reduction are considered as one step without separate, and the coefficients are computed. For $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$ with this approach, if we consider combining and modular reduction as one step and compute linear combinations of products, then 70 CNOT gates are used and the full depth is 27 for this. However, unlike in [8], if the coefficients are computed by separating the combining step and modular reduction, 85 CNOT gates are used and the full depth is 17.

In our implementation, we separate the combining step and modular reduction (which is kind of general) because it is more valuable to reduce the depth instead of using more CNOT gates. Firstly, we perform optimal CNOT operations on linear combinations of combining (62 CNOT gates) and then complete quantum multiplication by performing optimal CNOT operations on linear combinations of modular reduction (23 CNOT gates).

When we complete combining our quantum Karatsuba multiplication, $2n-1$ products of $c_0, c_1, ..., c_{2n-2}$ are generated. Then, we customize the quantum implementation of modular reduction with the irreducible polynomial. Table 4 shows the coefficients after performing modular reduction of $\mathbb{F}_{2^8}/(x^8+x^4+x^3+x+1)$. It is efficient to prepare duplicated elements among the coefficients in Table 4 (same colors) and paste them into x^n. For example, perform CNOT(c_9, c_8), CNOT(c_{14}, c_8) to prepare $c_8 = c_8+c_9+c_{14}$ (red color) in qubit c_8, and paste qubit c_8 into $c_1(x^1)$ and $c_4(x^4)$ by performing CNOT(c_8, c_1), CNOT(c_8, c_4). The naïve implementation requires 30 CNOT gates, but the customized implementation requires 23 CNOT gates and the depth is also reduced.

Table 4: Coefficients after performing modular reduction of $\mathbb{F}_{2^8}/(x^8+x^4+x^3+x+1)$.

x^n	Coefficient
$n=0$	$c_0 + c_8 + c_{12} + c_{13}$
$n=1$	$c_1 + c_8 + c_9 + c_{12} + c_{14}$
$n=2$	$c_2 + c_9 + c_{10} + c_{13}$
$n=3$	$c_3 + c_8 + c_{10} + c_{11} + c_{12} + c_{12} + c_{13} + c_{14}$
$n=4$	$c_4 + c_8 + c_9 + c_{11} + c_{14}$
$n=5$	$c_5 + c_9 + c_{10} + c_{12}$
$n=6$	$c_6 + c_{10} + c_{11} + c_{13}$
$n=7$	$c_7 + c_{11} + c_{12} + c_{14}$

Quantum resources required for multiplication of $\mathbb{F}_{2^8}/(x^8+x^4+x^3+x+1)$ including modular reduction are shown in Table 5 (Tables 1, 2, and 3 only include up-to the combining step).

Table 5: Quantum resources required for multiplication of $\mathbb{F}_{2^8}/(x^8+x^4+x^3+x+1)$.

Field size 2^n	Karatsuba Level	#CNOT	#1qCliff	#T	T-depth	#Qubits	Full depth
$n=8^\star$	3	323	54	189	4	81	32
$n=8^\diamondsuit$	3	377	81	108	1	108	31

$\star$: Using the Toffoli gate decomposition in [18].
$\diamondsuit$: Using AND gate.

4 Performance

In this Section, we review previous works on quantum multiplication and discuss the effectiveness of proposed work.

In [7], Banegas et al. introduced quantum binary field multiplication to solve discrete logarithm problems for binary elliptic curves. Authors implemented schoolbook multiplication. n^2 Toffoli gates are used for the multiplication of $\mathbb{F}_{2^n}$. They prioritize the upper products $c^n, c^{n+1}, \ldots, c^{2n-2}$ (where modular reduction occurs) for $h = f \cdot g$ to optimize the number of qubits. Then, the result of modular reduction of the upper products is stored in n-qubit h. Through this, quantum multiplication is implemented using $3n$ qubits for f, g, and h. Their works use a small number of qubits. Since it is based on a general schoolbook multiplication, many Toffoli gates are used and the Toffoli depth is also high. The quantum resources required for the multiplication of $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$ using their schoolbook multiplication are shown in Table 6 (Source [7]).

In [8], Kepley et al. presented quantum multiplication using the Karatsuba algorithm, which classically reduces the complexity of multiplication. Since the Karatsuba algorithm using the divide-and-conquer approach reduces the number of multiplication operations, quantum multiplication with a reduced number of Toffoli gates is presented in [8]. The quantum resources required for the multiplication of $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$ using their Karatsuba multiplication are shown in Table 6 (Source [8]).

In [6], Van Hoof presented another quantum multiplication using the Karatsuba algorithm. The work in [6] reduced the number of qubits used in [8]. In [8], additional qubits are used to store the Karatsuba products, but the author of [6] uses the LUP decomposition. $3n$ qubits equal to schoolbook multiplication are used. In [6], the number of gates and qubits used in their implementation is reported, but not the full depth. The full depth estimated by decomposing Toffoli gates is not reported. Only the depth estimated at the NCT (NOT, CNOT, and Toffoli) level is reported. Multiplication in [6] uses fewer qubits than that of [8], but we assume the full depth is higher in [6]. This is because the operation of gates in a reduced space (reduced number of qubits) becomes a bottleneck for parallelism and increases the depth. The depth estimated in [6] for $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$ is 139 (NCT level), and it increases when the full depth is estimated.

Table 6 compares the quantum resources required for the multiplication of $\mathbb{F}_{2^8}/(x^8 + x^4 + x^3 + x + 1)$. Although no specific full depth is reported in [6], it is confirmed that the NCT depth of [6] is higher than the full depth of [8].

Our quantum multiplication is optimized with Toffoli depth one for any field size. In quantum multiplication, Toffoli depth has a huge impact on counting full depth. Thus, under the influence of the optimized Toffoli depth, our work achieves the lowest full depth.

Quantum computers in the upcoming NISQ era are no longer small, but they still need quantum error correction. In error correction, the metric of Toffoli depth is probably the most important. However, since the number of qubits is still important, it is important to consider the trade-off between qubits and

Table 6: Comparison of quantum resources required for multiplication of $\mathbb{F}_{2^8}/(x^8+x^4+x^3+x+1)$.

Field size 2^n	Source	#CNOT	#1qCliff	$\#T$	Toffoli depth	#Qubits	Full depth	$TD \cdot M$
$n=8$	This work (Sec. 3.5)	323	54	189	1	81	32	81
	[7]	405	30	448	28	24	216	672
	[8]	270	54	189	8	43	88	344
	[6]	382	54	189	N/A	24	N/A	N/A

TD = Toffoli depth, M = number of qubits.

depth. Our quantum multiplication requires relatively more qubits, but achieves the best trade-off of $TD \cdot M$ (TD is Toffoli depth, M is the number of qubits). This metric ($TD \cdot M$) represents the trade-off for quantum circuits and is adopted in [11].

As mentioned in Section 3.3, if it is not stand-alone multiplication, the overhead of using more qubits can be offset. One representative example is an inversion based on the Itoh-Tsujii algorithm [20] that requires multiple multiplications (squaring operations are also required, but they are implemented simply with fewer quantum resources compared to multiplication). In this case, multiplications after the first multiplication are implemented more efficiently because previously allocated qubits can be reused.

5 Conclusion

In response to the upcoming post-quantum era, optimizing quantum cryptanalysis is of great interest to the cryptographic community.

In this paper, we present an optimized quantum binary field multiplication, which is essential for quantum cryptanalysis of ECC. Our main contribution is quantum multiplication that is optimized with Toffoli depth one for any field size. Further, we describe the reverse operation to offset the overhead of qubits, optimization with T-depth one, and an efficient implementation of modular reduction. This is the first implementation of quantum multiplication to optimize Toffoli depth and full depth. We show the impact of our work by comparing it to other implementations in various metrics.

Future work is to find another optimization for quantum cryptanalysis building blocks of ECC. We note the direction of optimization that should be pursued in quantum implementations. We will also explore an optimized quantum cryptanalysis of ECC consisting of efficient quantum arithmetic operations.

References

1. P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999. 2

2. T. Häner, M. Roetteler, and K. M. Svore, "Factoring using $2n + 2$ qubits with Toffoli based modular multiplication," *arXiv preprint arXiv:1611.07995*, 2016. 2
3. C. Gidney, "Factoring with $n + 2$ clean qubits and $n - 1$ dirty qubits," *arXiv preprint arXiv:1706.07884*, 2017. 2
4. M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, "Quantum resource estimates for computing elliptic curve discrete logarithms," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 241–270, Springer, 2017. 2
5. T. Häner, S. Jaques, M. Naehrig, M. Roetteler, and M. Soeken, "Improved quantum circuits for elliptic curve discrete logarithms," in *International Conference on Post-Quantum Cryptography*, pp. 425–444, Springer, 2020. 2
6. I. Van Hoof, "Space-efficient quantum multiplication of polynomials for binary finite fields with sub-quadratic Toffoli gate count," *arXiv preprint arXiv:1910.02849*, 2019. 2, 5, 11, 12
7. D. Cheung, D. Maslov, J. Mathew, and D. K. Pradhan, "On the design and optimization of a quantum polynomial-time attack on elliptic curve cryptography," in *Workshop on Quantum Computation, Communication, and Cryptography*, pp. 96–104, Springer, 2008. 2, 11, 12
8. S. Kepley and R. Steinwandt, "Quantum circuits for $\mathbb{F}_{2^n}$-multiplication with sub-quadratic gate count," *Quantum Information Processing*, vol. 14, no. 7, pp. 2373–2386, 2015. 2, 5, 9, 11, 12
9. K. Jang, S. J. Choi, H. Kwon, Z. Hu, and H. Seo, "Impact of optimized operations $A \cdot B$, $A \cdot C$ for binary field inversion on quantum computers," in *International Conference on Information Security Applications*, pp. 154–166, Springer, 2020. 2, 5
10. K. Jang, G. J. Song, H. Kim, H. Kwon, W.-K. Lee, Z. Hu, and H. Seo, "Binary field montgomery multiplication on quantum computers," *Cryptology ePrint Archive*, 2021. 2
11. J. Zou, Z. Wei, S. Sun, X. Liu, and W. Wu, "Quantum circuit implementations of AES with fewer qubits," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 697–726, Springer, 2020. 2, 12
12. D. Bhattacharjee and A. Chattopadhyay, "Depth-optimal quantum circuit placement for arbitrary topologies," *arXiv preprint arXiv:1703.08540*, 2017. 2
13. NIST., "Submission requirements and evaluation criteria for the post-quantum cryptography standardization process," 2016. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. 2
14. D. S. Steiger, T. Häner, and M. Troyer, "ProjectQ: an open source software framework for quantum computing," *Quantum*, vol. 2, p. 49, 2018. 3
15. A. Cross, "The IBM Q experience and Qiskit open-source quantum computing software," in *APS March meeting abstracts*, vol. 2018, pp. L58–003, 2018. 3
16. K. Svore, A. Geller, M. Troyer, J. Azariah, C. Granade, B. Heim, V. Kliuchnikov, M. Mykhailova, A. Paz, and M. Roetteler, "Q# enabling scalable quantum computing and development with a high-level dsl," in *Proceedings of the real world domain specific languages workshop 2018*, pp. 1–10, 2018. 3
17. A. Karatsuba, "Multiplication of multidigit numbers on automata," in *Soviet physics doklady*, vol. 7, pp. 595–596, 1963. 4
18. M. Amy, D. Maslov, M. Mosca, M. Roetteler, and M. Roetteler, "A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, p. 818–830, Jun 2013. 4, 6, 10

19. S. Jaques, M. Naehrig, M. Roetteler, and F. Virdia, "Implementing grover oracles for quantum key search on AES and LowMC," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 280–310, Springer, 2020. 8
20. T. Itoh and S. Tsujii, "A fast algorithm for computing multiplicative inverses in GF(2^m) using normal basis," *Information and Computing*, vol. 78, pp. 171–177, 1988. 12