IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 21, 2020, accepted November 9, 2020, date of publication November 18, 2020, date of current version December 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3039167

Quantum Modular Multiplication

SEONG-MIN CHO[1], (Student Member, IEEE), AEYOUNG KIM[2], (Member, IEEE), DOOHO CHOI[3], (Member, IEEE), BYUNG-SOO CHOI[3], (Member, IEEE), AND SEUNG-HYUN SEO[1,4], (Member, IEEE)

[1]Department of Electrical Engineering, Graduate School, Hanyang University, Ansan 15588, South Korea
[2]Research Institute of Engineering & Technology, Hanyang University ERICA, Ansan 15588, South Korea
[3]Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea
[4]Division of Electrical Engineering, Hanyang University ERICA, Ansan 15588, South Korea

Corresponding author: Seung-Hyun Seo (seosh77@hanyang.ac.kr)

This work was supported in part by the Institute for Information & Communications Technology Planning & Evaluation (IITP) Grant funded by the Korea Government (MSIT) (<Q|Crypton> Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity) under Grant 2019-0-00033, in part by the Ministry of Science and ICT (MSIT), South Korea, through the Information Technology Research Center (ITRC) Support Program, supervised by the Institute for Information & Communications Technology Planning & Evaluation (IITP), under Grant IITP-2020-2018-0-01417, and in part by the research fund of Hanyang University under Grant HY-2018-N.

ABSTRACT Quantum modular multiplication circuit is one of the basic quantum computation circuits which are basic functions in quantum algorithms. However, since quantum-quantum modular multipliers require a high cost reversible modular inversion routine for modular multiplication, researchers have been unable to propose a feasible quantum-quantum modular multiplier. In this paper, we proposed efficient quantum-classical modular multipliers and the first quantum-quantum modular multipliers that do not require a reduction stage by transforming the partial product used in multiplication utilizing bit-shift operation. Then, we calculated quantum resource complexity and analyzed it compared to other quantum modular multipliers and utilized ETRI (Electronics and Telecommunications Research Institute) Qcrypton to analyze quantum resource complexity in the practical quantum computing situation. The proposed quantum modular multipliers show an improvement of 50% in terms of gates and circuit depth compared to the most recently proposed high-performance quantum modular multipliers.

INDEX TERMS Quantum computing, quantum algorithm, modular multiplication.

I. INTRODUCTION

In 1981, Richard Feynman proposed a quantum computer utilizing quantum superposition. While the classical computers use the bit with a value of 0 or 1 as the elementary unit of information, the quantum computer uses the qubit in which the state of 0 and 1 exists simultaneously as a probabilistic superposition state. Due to this superposition state, quantum computers can express the data in a high dimensional form, and even a small number of qubits can simultaneously represent a large number of cases. Thus, quantum computers can efficiently solve problems that are difficult to deal with in classical computers. The robust computing speed of quantum computers is expected to contribute to improving human life quality by solving difficulties in various areas such as IT, chemical, medical, and pharmaceutical. Currently, the development of quantum computers is being led by several IT companies. D-WAVE systems have developed a 128 qubit quantum computer using quantum annealing technique, and Google developed a quantum processor 'Sycamore' that solved a problem that would take 10,000 years in 200 seconds as the best performing supercomputer in 2019 [1]. A variety of quantum algorithms were proposed to solve difficult problems taking advantage of quantum computers. In 1985, David Deutsch proposed the Deutsch algorithm which produced the first exponential performance improvement [2]. And in 1994, Peter Shor proposed the Shor algorithm to solve the problem of factoring and discrete logarithm within a polynomial time [3], [4].

These quantum algorithms use a large number of quantum computation circuits. Therefore, it is important to design efficient quantum computation circuits. In particular, addition and multiplication circuits are the most basic quantum computation circuits, and if the complexity of these circuits can be reduced, the complexity of the entire algorithm can be greatly reduced. Quantum computation circuits are classified into the 'quantum-classical' circuits and 'quantum-quantum' circuits according to the type of input. The quantum-classical

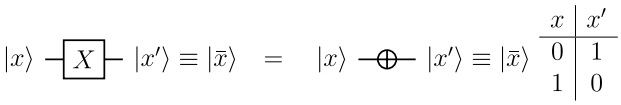
The associate editor coordinating the review of this manuscript and approving it for publication was Siddhartha Bhattacharyya.

circuits fix one input value as a classical parameter, and the quantum-quantum circuits use two quantum registers as inputs. We need both kinds of quantum computation circuits to implement quantum algorithms. Basically, implementing a quantum-quantum circuit requires more quantum resources such as qubits, gates, and circuit depth. In 1998, Phil Gossett proposed a quantum addition circuit [5], which borrowed the classical carry-save method [6] and reduced the depth of the circuit from $O(n)$ to $O(\log n)$. In 2008, Draper *et al.* proposed QCLA (Quantum Carry-Lookahead) adder [7], which borrowed the classical carry-lookahead method [8]–[10] and the depth of the circuit is O(logn). Both of these quantum adders are capable of quantum-quantum computation. In 2018, R. Rines *et al.* proposed quantum-classical modular multipliers using classical reduction techniques such as Montgomery residue arithmetic [11] and Barrett reduction [12]. However, they did not provide specific circuits of quantum-quantum modular multipliers in [13]. In fact, it is difficult to design a quantum-quantum modular multiplier circuit that efficiently computes the multiplication because the reversible modular inversion routine that inverts the reduction stage used for modular multiplication has a too high cost.

The complexity of quantum algorithms is evaluated by the number of qubits, quantum gates, and circuit depth. The T gate has a higher implementation cost than the other gates, and the quantum resource complexity can be evaluated by the Toffoli gate where the T gate is the most used. However, the number of qubits, gates, and circuit depth simply required in the circuit may vary due to a number of environmental factors when operating in the practical quantum computing situation. Therefore, there is a need for a quantum resource complexity analysis that takes into account the practical situation. The ETRI (Electronics and Telecommunications Research Institute) proposed Qcrypton, a quantum computing software platform that can accurately analyze the performance of quantum computing resources considering the practical quantum computing situation [14].

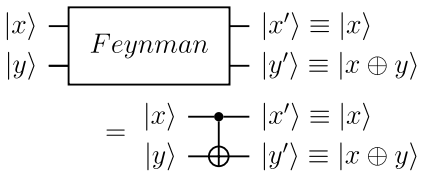
In this paper, we propose an efficient modular multiplication method in the quantum circuit. We use a bit shift and bit circulation method that transforms the form of partial product in the multiplication process for efficient quantum modular multiplication instead of the reduction stage. Using these methods, we propose quantum-classical and quantum-quantum modular multipliers, reducing the multiplier complexity. The contributions of this paper are as follows:

- To eliminate the reduction stage, we proposed the quantum modular multiplication circuits which proceed multiplication using the method of transforming the partial product using bit shift operation. The quantum-classical circuits compute the bit shift operation that transforms the partial product in the classical computer, and the quantum-quantum circuits compute the bit shift operation on the quantum circuit using the Toffoli gate.
- We calculated the quantum resource complexity of our quantum modular multipliers and confirmed that it is efficient compared to other quantum modular multipliers. We also utilized the ETRI Qcrypton to analyze the complexity of quantum resources in the practical quantum computing situation.



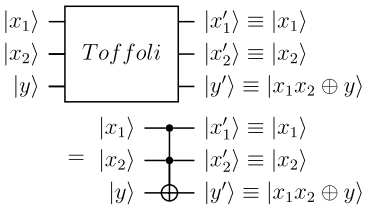
x	x'
0	1
1	0

FIGURE 1. NOT Gate.



x	y	x'	y'
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

FIGURE 2. CNOT Gate.



x_1	x_2	y	x_1'	x_2'	y'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

FIGURE 3. CCNOT Gate.

II. PRELIMINARIES

We introduce elementary quantum gates of quantum circuits, quantum modular adders, and quantum modular multipliers in this section.

A. ELEMENTARY QUANTUM GATES

1) NOT GATE

The operation of X Gate in quantum computing, as shown in Fig.1, is the same as the operation of NOT Gate in Classical computing.

2) CNOT GATE

The operation of the CNOT(Controlled NOT) Gate in quantum computing, as shown in Fig.2, is the same as an XOR operation in Classical computing. When two qubits x and y are input to this CNOT Gate, the CNOT Gate results are x for the input x and $x \oplus y$ for the input y.

3) CCNOT GATE

CCNOT (Controlled-Controlled-NOT) gate in Figure 3 is also called the Toffoli gate. The Toffoli gate has three inputs, and the output is almost the same as the input, except that the third qubit is flipped only if the first and second qubits are all 1. That is, when three qubits x, y, and z are input to this Toffoli Gate, the result of this Toffoli Gate is x and y for the x and y, and $z \oplus xy$ for the input z.

B. QUANTUM MODULAR ADDER

Quantum modular adders are divided into Quantum Ripple-Carry Adder (QRCA), Quantum Carry-Save Adder (QCSA), and Quantum Carry-Lookahead Adder (QCLA) depending on the method of handling carry occurring in bitwise additions. The Ripple Carry Adder (RCA) is the simplest circuit among the adders by merely handling the carry, which is calculated in each bit addition and inputs to the next bit addition called Ripple. The QRCA is a quantum addition circuit based on the RCA, and a representative QRCA is a quantum adder proposed by Vedral *et al.* in 1996 [15]. Vedral *et al.*'s QRCA has a depth O(n) instead of constructing the simplest circuit.

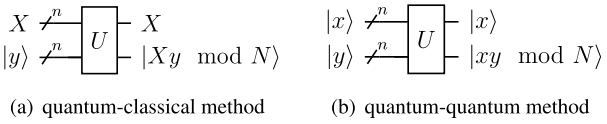
The Carry Save Adder (CSA) calculates bit-by-bit subtotals and carry for each sub-total for three or more inputs, and it calculates the final sum by using these sub-part results. QCSA is a quantum circuit based on the CSA, and Phil Gossett proposed a QCSA with O(logn) depth in 1998 by reducing the depth than the previous QCSA [5]. The Carry Lookahead Adder (CLA) is the fastest addition circuit because it solves RCA's carry propagation problem by introducing a method of calculating carry by every bitwise addition in parallel. QCLA is a quantum circuit based on this CLA, and Draper *et al.* proposed a QCLA in 2008 with O(logn) depth that is a reduced depth than the previous QCLA [7]. Quantum modular adders are applied to construct quantum modular multipliers. In this paper, we utilize QCLA to design quantum modular multipliers to provide efficient performance.

C. QUANTUM MODULAR MUTIPLIER

Quantum modular multipliers are implemented by using quantum modular adders repeatedly. Depending on the type of input value, quantum modular multipliers are classified into the ''quantum-classical'' modular multipliers and the ''quantum-quantum'' modular multipliers. Classically, for the number of n-bits x and y, the modular multiplication $x \cdot y \mod N(= 2^n)$ proceeds as follows: First, we compute the result of multiplication $x \cdot y$ by repeatedly using additions. Second, we divide the $x \cdot y$ by N. Then, we get the quotient q and remainder r, and $x \cdot y - qN$ will be the result of modular multiplication. This process is called reduction. Most quantum modular multipliers rely on the reduction to compute modular multiplication. Recently, Rines and Chuang [13] proposed quantum modular multipliers using classical reduction techniques such as Montgomery residue arithmetic [11] and Barrett reduction [12]. They [13] did not provide the practical quantum-quantum modular multiplier, because it requires a reversible modular inversion routine [16] with a heavy overhead in the reduction stage.

1) QUANTUM-CLASSICAL MODULAR MULTIPLIER

The quantum-classical modular multiplier fixes one input value as a classical parameter, as shown in Fig. 4(a). This quantum-classical modular multiplier realizes $|y\rangle_n \rightarrow |Xy - qN\rangle_n = |Xy \mod N\rangle_n$ where n-bit classical multiplier X and n-bit quantum-classical modular multiplier.



(a) quantum-classical method (b) quantum-quantum method

FIGURE 4. The quantum modular multiplication.

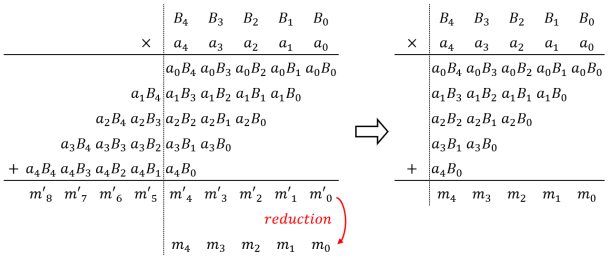


FIGURE 5. The modular multiplication method over GF(2^n) where $n = 5$.

2) QUANTUM-QUANTUM MODULAR MULTIPLIER

The quantum-quantum modular multiplier uses two quantum registers as inputs. This quantum-quantum modular multiplier, as shown in Fig. 4(b), realizes $|x\rangle_n |y\rangle_n \rightarrow |x\rangle_n |xy - qN\rangle_n = |x\rangle_n |xy \mod N\rangle_n$ for two quantum registers x and y.

III. QUANTUM MODULAR MULTIPLIER OVER GF(2^n)

In this section, we propose quantum-classical and quantum-quantum modular multiplier over GF(2^n).

A. QUANTUM-CLASSICAL MODULAR MULTIPLIER OVER GF(2^n)

Given n-bit quantum value a and constant value B, our quantum-classical modular multiplier over GF(2^n) computes $(a \times B) \mod 2^n$, the modular multiplication of a and B. Our quantum-classical modular multiplier over GF(2^n) consists of two stages as shown in Fig.6: partial product setting stage and modular addition stage. By performing the bit-shift operation on the classical value B to compute reduced partial product in the partial product setting stage, our quantum-classical modular multiplier over GF(2^n) does not require a reduction stage on the quantum circuit.

1) PARTIAL PRODUCT SETTING STAGE

When a number m exceeds 2^n, the partial product setting stage computes the reduction of m to a number on GF(2^n) by just discarding bits above n-th position. Fig.5 shows how the partial product is transformed for the multiplication over GF (2^n) when $n = 5$.

As shown in Fig.5, the i-th partial product is transformed by multiplying a_i, the i-th bit of a, with each bit of B. And then the partial product is shifted as much as i to the left and the bits that go over a bit position n are discarded. The modular multiplication is computed by adding transformed partial products over GF (2^n). As shown in Fig.10, our quantum-classical modular multiplier consists of quantum-classical modular adders that use one qubit of quantum register a as

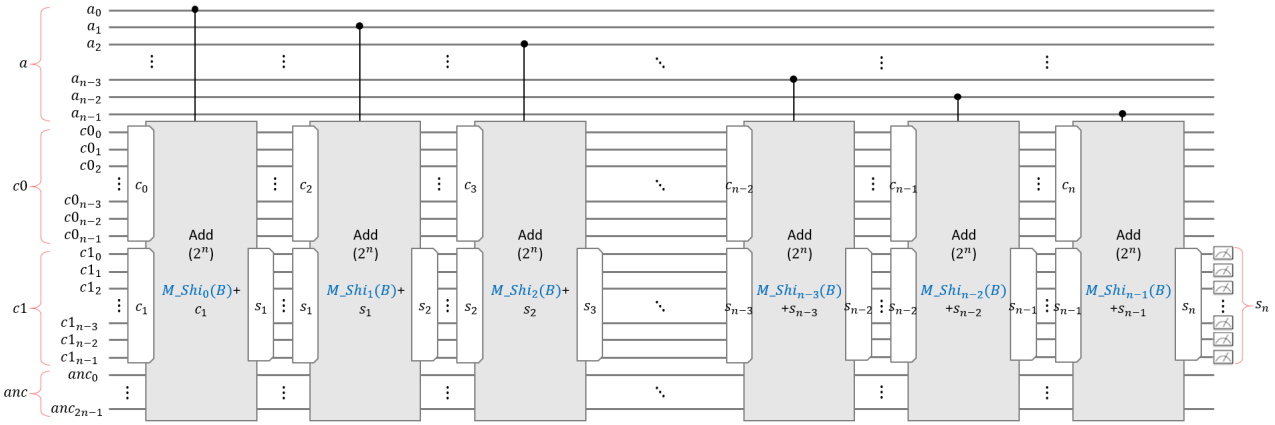


FIGURE 6. The quantum circuit for the quantum-classical modular multiplication over GF(2^n) where $s_n = (a \times B) \mod 2^n$. The part $M\_Shi_i(B)$ with blue color is the value after the i-th partial product setting stage.

a control qubit. Each quantum modular adder has a control qubit a_i (i : 0 *to* $n-1$). So, if the a_i is 0, the quantum-classical modular adder does not work, and the partial product becomes zero. If the a_i is 1, the quantum-classical modular adder works, and the partial product becomes the result of the circular-shift operation for B. Thus, this partial product setting stage shifts the classical input B as much as i (i : 0 *to* $n-1$) to the left and discards the bits that go over a bit position n in the classical computing environment. Each of these values is the classical input of the i-th quantum-classical modular adder.

2) MODULAR ADDITION STAGE

The modular addition stage gathers the i-th partial products computed in the partial product setting stage using modular adder when the i-th quantum input a_i is 1. For that, we use Draper's efficient Quantum Carry-Lookahead Adder (QCLA) as the quantum-classical modular adder. In this stage, a total of n quantum-classical modular adders are used.

B. QUANTUM-QUANTUM MODULAR MULTIPLIER OVER GF(2^n)

Our quantum-quantum modular multiplier over GF (2^n) consists of three stages; the qubit setting stage(shown in blue in Fig.10), the modular addition stage(in red), and the inverse setting stage (in yellow) which work together and are repeated.

1) QUBIT SETTING STAGE

Our quantum-quantum modular multiplier sets the 0-th partial product to register $c1$, sets remaining partial products to register $c0$, and gathers partial products in register $c1$ using quantum-quantum modular adders to compute modular multiplication of two quantum values. To this end, this stage computes the left-shift operation of the partial product on the quantum circuit. The operation of the qubit setting stage is in Algorithm 1, which computes the left-shifted partial products using only Toffoli gates.

Algorithm 1 Qubit Setting

input : quantum registers a, b, $c0$, and $c1$
output: quantum registers $c0$ and $c1$

1 **for** $i = 0$ **to** $n-1$ **do**
2 Toffoli($a_0, b_i, c1_i$);
3 **for** $i = 0$ **to** $n-1$ **do**
4 **for** $j = 0$ **to** $n-1-i$ **do**
5 Toffoli($a_i, b_j, c0_{i+j}$);
6 Return $c0$, $c1$

This stage sets the left-shifted partial products to $c0$ ($c0_{n-1}c0_{n-2}\ldots c0_1c0_0$) and $c1$ ($c1_{n-1}c1_{n-2}\ldots c1_1c1_0$) each using two quantum input registers a ($a_{n-1}a_{n-2}\ldots a_1a_0$) and b ($b_{n-1}b_{n-2}\ldots b_1b_0$). In Algorithm 1, setting the 0-th partial product to $c1$ in lines 1 and 2 is to repeat the Toffoli operations from (b_0, $c1_0$) to (b_{n-1}, $c1_{n-1}$) with a_0. So, Toffoli($a_0, b_i, c1_i$) is able to store $c1_i + a_0b_i$ in qubit $c1_i$ where $c1_i$ becomes a_0b_i if $c1_i$ is zero. If we repeat this process from 0 to $n-1$ for i, the 0-th partial product is set to the register $c1$. And then, setting the remaining i-th partial products to register $c0$ is to perform the Toffoli operation for (b_j, $c0_{i+j}$) with a_i where $i = 0$ *to* $n-1$ and $j = 0$ *to* $n-1-i$, repeatedly. In this process, the Toffoli operation stores the j-th bit of i-th partial product in qubit $c0_{i+j}$, and it performs the i-th left-shift operation for the i-th partial product.

2) MODULAR ADDTION STAGE

The modular addition stage adds two quantum register $c0$ and $c1$, received from the qubit setting stage, on GF(2^n). QCLA is applied for performing quantum-quantum modular adder in this stage. The result of this stage is $c0+c1 \mod 2^n$ in the register $c1$.

3) INVERSE SETTING STAGE

Finally, the inverse setting stage initializes the register $c0$ to zero to return the register to its state before the qubit setting

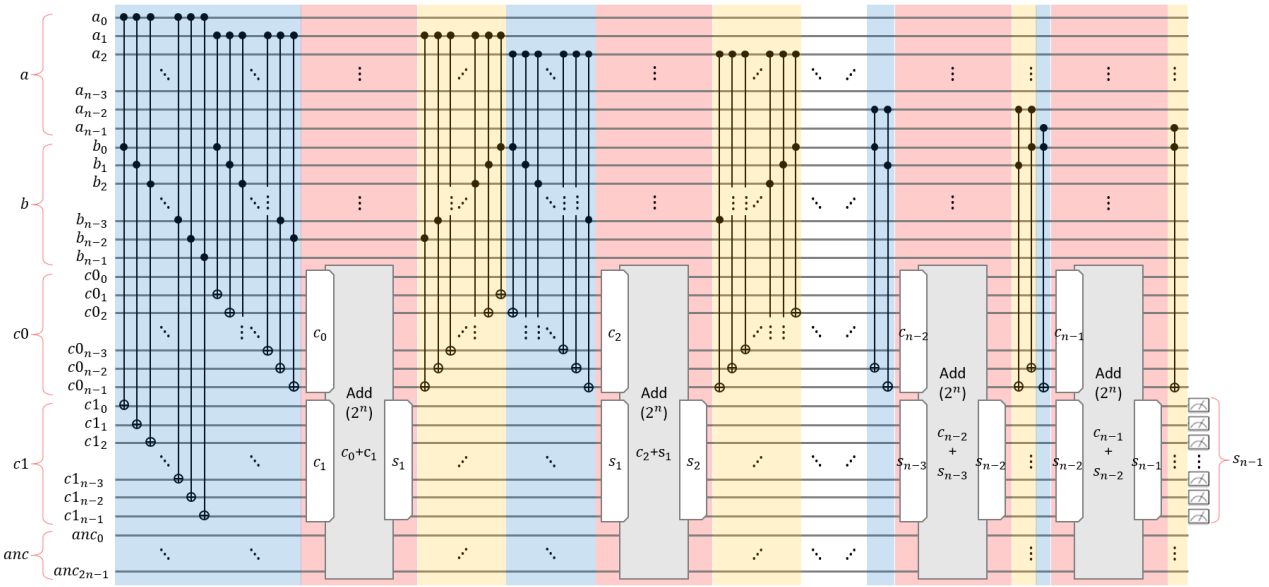


FIGURE 7. The quantum circuit for the quantum-classical modular multiplication over GF(2^n) where $s_{n-1} = (a \times b) \mod 2^n$. The blue parts are qubit setting stages and the red parts are modular addition stages. The yellow parts are inverse setting stages.

stage. This process allows the quantum-quantum modular multiplier to proceed with the qubit setting stage again. After the Inverse setting stage, the qubit setting stage is run again to proceed modular addition. Our quantum-quantum modular multiplier performs these three stages n times to compute the modular multiplication.

IV. QUANTUM MODULAR MULTIPLIER OVER GF($2^n - 1$)

In this section, we propose quantum-classical and quantum-quantum modular multiplier over GF($2^n - 1$).

A. QUANTUM-CLASSICAL MODULAR MULTIPLIER OVER GF($2^n - 1$)

Our quantum-classical modular multiplier over GF($2^n - 1$) inputs n-bit quantum value a and constant value B. It computes $(a \times B) \mod 2^n - 1$, by using the modular multiplication of a and B through the partial product setting stage and the modular addition stage as shown in Fig.6. By utilizing the property of Mersenne number to transform the partial product, our quantum-classical modular multiplier over GF($2^n - 1$) does not require a reduction stage on the quantum circuit.

1) PARTIAL PRODUCT SETTING STAGE

The Mersenne number refers to a number of the form $M_n = 2^n - 1$. If a Mersenne number m exceeds $2^n - 1$, it can be reduced to a number on GF($2^n - 1$) as follows: First, the m is split into two n-bit numbers. And then, the two split numbers are added. If the sum of the two numbers exceeds $2^n - 1$, the two processes above are repeated. The partial product is transformed by utilizing this method of Mersenne number reduction in this stage. Fig.8 shows an example of a transformation of the partial product for modular multiplication over GF ($2^n - 1$) when $n = 5$.

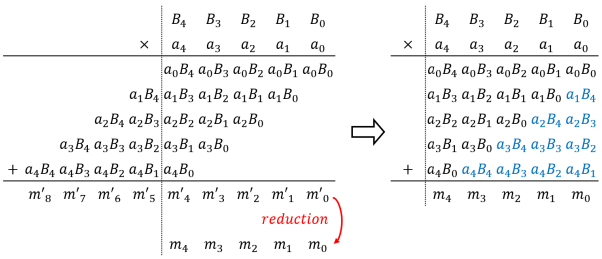


FIGURE 8. The modular multiplication method over GF($2^n - 1$) using property of the Mersenne number where $n = 5$.

The i-th partial product is the value shifted as much as i to the left after multiplying a_i, the i-th bit of a, with each bit of B. And then the bits of the partial product that go over a bit position n are added to the least significant position as shown in the right side of Fig.8. This process is the same as performing circular-shift on the original partial product. As shown in Fig.10, our quantum-classical modular multiplier consists of quantum-classical modular adders that use one qubit of quantum register a as a control qubit. Each quantum modular adder has a control qubit a_i ($i : 0$ *to* $n - 1$). So, if the a_i is 0, the quantum-classical modular adder does not work, and the partial product becomes zero. If the a_i is 1, the quantum-classical modular adder works, and the partial product becomes the result of the circular-shift operation for B. This partial product setting stage performs circular shift operations in the classical computing environment to move the quantum-classical modular adder input B as much as i ($i : 0$ *to* $n - 1$).

2) MODULAR ADDITION STAGE

In this stage, the i-th quantum-classical modular adder can only operate when the control qubit a_i is 1. QCLA is used as

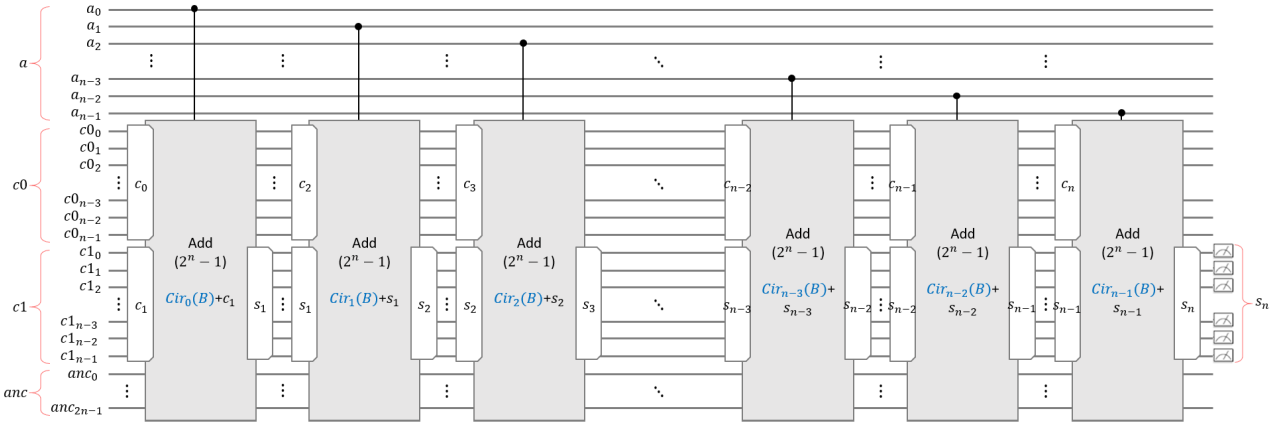


FIGURE 9. The quantum circuit for the quantum-classical modular multiplication over GF(2^n-1) where $s_n = (a \times B) \mod 2^n - 1$. The part $Cir_i(B)$ with blue color is the value after the i-th partial product setting stage.

the quantum-classical modular adder in this stage. The i-th partial product transformed in the previous stage is an input for QCLA. QCLA is performed a total of n times in this part of the process.

B. QUANTUM-QUANTUM MODULAR MULTIPLIER OVER GF(2^n-1)

Our quantum-quantum modular multiplier over GF(2^n-1) consists of three stages: the qubit setting stage (shown in blue in Fig.10), the modular addition stage (in red), and the inverse setting stage (in yellow) which work together and are repeated.

1) QUBIT SETTING STAGE

Our quantum-quantum modular multiplier sets the 0-th partial product to register $c1$, sets remaining partial products to register $c0$, and gathers partial products in register $c1$ using quantum-quantum modular adders to compute modular multiplication of two quantum values. To this end, this stage computes the left-circular-shift operation of the partial product on the quantum circuit. Algorithm 2 shows the qubit setting operation which computes the left-circular-shifted partial products using only Toffoli gates.

Algorithm 2 Qubit Setting

input : quantum registers a, b, $c0$, and $c1$
output: quantum registers $c0$ and $c1$

1 **for** $i = 0$ **to** $n-1$ **do**
2 Toffoli($a_0, b_i, c1_i$);
3 **for** $i = 1$ **to** $n-1$ **do**
4 **for** $j = 0$ **to** $n-1$ **do**
5 Toffoli($a_i, b_j, c0_{(i+j) \mod n}$);
6 Return $c0, c1$

This stage sets the left-circular-shifted partial products to $c0$ ($c0_{n-1}c0_{n-2}\ldots c0_1c0_0$) and $c1$ ($c1_{n-1}c1_{n-2}\ldots c1_1c1_0$) each respectively using two quantum input registers a ($a_{n-1}a_{n-2}\ldots a_1a_0$) and b ($b_{n-1}b_{n-2}\ldots b_1b_0$). In order to set the 0-th partial product to $c1$, the Toffoli operations are repeatedly performed from ($b_0, c1_0$) to ($b_{n-1}, c1_{n-1}$) with a_0 in lines 1 and 2 of Algorithm 2. So, Toffoli($a_0, b_i, c1_i$) is able to store $c1_i + a_0b_i$ in qubit $c1_i$ where $c1_i$ becomes a_0b_i if $c1_i$ is zero. If we repeat this process from 0 to $n-1$ for i, the 0-th partial product is set to the register $c1$. And then, the remaining partial products are set to register $c0$ in lines 3 to 5 of Algorithm 2. In order to set the i-th left-circular-shifted partial product to register $c0$, the Toffoli operations for ($b_j, c0_{(i+j) \mod n}$) with a_i where $i = 1$ *to* $n-1$ and $j = 0$ *to* $n-1$, are repeatedly performed. In this process, Toffoli operation stores the j-th bit of i-th partial product in qubit $c0_{(i+j) \mod n}$, and it performs the i-th left-circular-shift operation for the i-th partial product.

2) MODULAR ADDTION STAGE

The modular addition stage adds the partial products into two quantum registers $c0$ and $c1$ on GF(2^n-1) using a quantum-quantum modular adder over GF(2^n-1). QCLA is used for the quantum-quantum modular adder.

3) INVERSE SETTING STAGE

This stage initializes the register $c0$ to zero to return the register to its original state for the next qubit setting stage. The i-th inverse setting stage is a reversed operation of the i-th qubit setting stage to initialize the register $c0$ to zero. After the inverse setting stage, the qubit setting stage and the modular addition stage are run again. Our quantum-quantum modular multiplier computes modular multiplication by performing these three stages $n-1$ times.

V. COMPLEXITY ANALYSIS

In this section, we evaluate the quantum resource complexity of our quantum modular multipliers. Our quantum modular multipliers use the X gate, CNOT gate, and Toffoli gate. The Toffoli gate consists of the T gate, CNOT gate, and

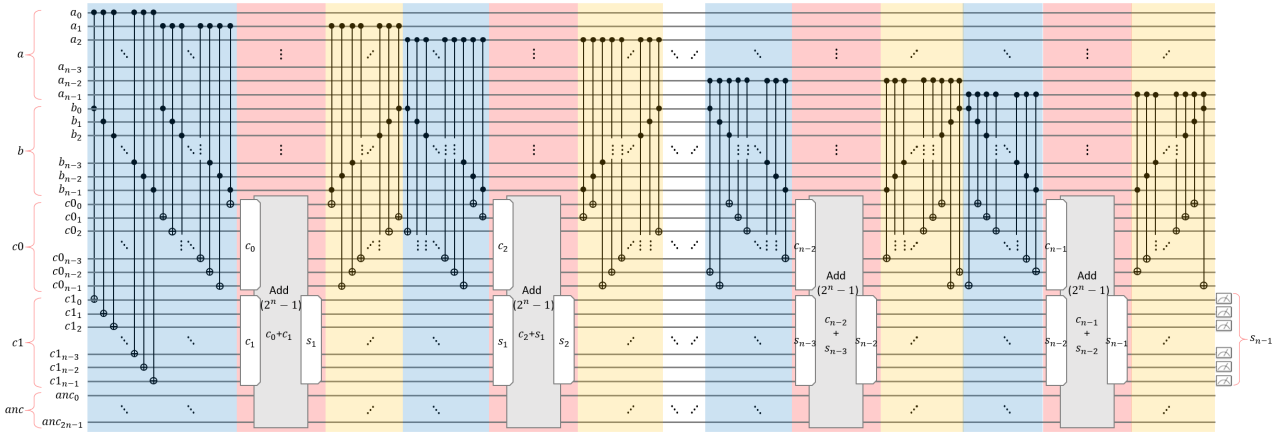


FIGURE 10. The quantum circuit for the quantum-quantum modular multiplication over GF($2^n - 1$) where $s_{n-1} = (a \times b) \mod 2^n - 1$. The blue parts are qubit setting stages and the red parts are modular addition stages. The yellow parts are inverse setting stages.

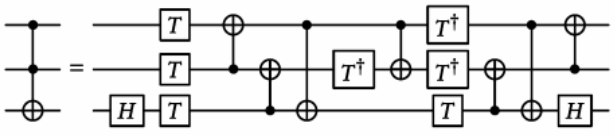


FIGURE 11. The Toffoli gate which consists of CNOT gate, T gate, and Hadamard gate.

Hadamard gate in practical quantum computing implementation, as shown in Fig.11. The T gate is more costly to implement than other gates. Thus, we analyze quantum resource complexity based on the most expensive Toffoli gate.

We also utilize Qcrypton, developed by ETRI, for a more practically accurate analysis of quantum resource complexity in practical quantum computing situations. The Qcrypton is a quantum computing software platform that models a large-scale quantum computing system. Utilizing Qcrypton, we are able to accurately analyze the performance of quantum computing resources taking into account practical situations.

A. QUANTITATIVE ANALYSIS

The proposed quantum modular multipliers have so far computed modular multiplication using the following methods: First, accumulate the partial products multiplied by one bit of input a and another input b. Then, the cumulative sum is reduced by the modular operation. This stage requires a complicated reduction method. Unlike the existing methods, our quantum modular multipliers eliminated the complex reduction stage by transforming the partial products using bit shift operation and then computing their cumulative sum using quantum modular adders.

Our quantum-classical modular multipliers compute bit-shifted partial products in a classical computer rather than on the quantum circuit, and then they become the classical inputs for the quantum-classical modular adders. Thus, our quantum-classical modular multipliers only need to perform a total of n modular additions on a quantum circuit. We use QCLA as an efficient quantum-classical modular adder that can compute modular addition over GF($2^n - 1$). The implementation of QCLA requires $4n$ qubits and $10n$ gates with $4 \log_2 n$ depth. So, our quantum-classical modular multipliers require $5n$ qubits (n qubits for n-bit quantum input and $4n$ qubits for the quantum modular adder [7]) and $10n^2$ gates with $4n \log_2 n$ quantum circuit depth. Compared to the depth of the quantum modular multiplication circuit using Cuccaro's adder, which is $12n^2$, ours at $10n^2$ is more efficient. In addition, our quantum-classical modular multipliers use only one-sixth of the number of gates and circuit depth required for the multiplication method using Draper's adder, and only half of the number of gates and circuit depth required for the modular multiplier in [13].

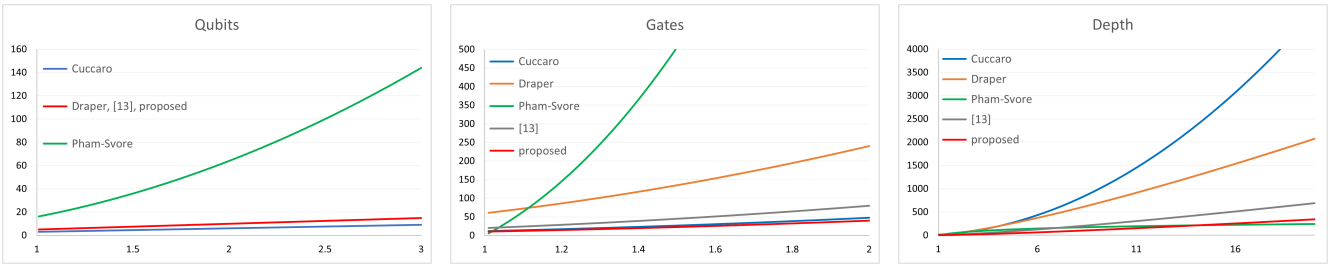
A quantum-quantum modular multiplier has never been proposed for a specific circuit due to the reason that modular inversion at the reduction stage is very costly. In order to eliminate this cost, we applied a bit shift method to our proposed design that transforms partial products without the reduction stage, creating a feasible quantum-quantum modular multiplier. We used the Toffoli gates to compute the reduced partial products on a quantum circuit. Our quantum-quantum modular multiplier over GF(2^n) requires $\frac{n(n+1)}{2}$ Toffoli gates and uses $6n$ qubits and $11n^2$ gates with n^2 depth. Quantum-quantum modular multiplier over GF($2^n - 1$) requires n^2 Toffoli gates and uses $6n$ qubits and $11n^2$ gates with $\frac{1}{2}n^2$ depth.

Tab.1 and Fig.12 show the quantum resource complexity comparison of our quantum-classical and quantum-quantum modular multiplier and other quantum modular multipliers.

As shown in Fig.12, our multiplier has the lowest gate complexity. Although our multiplier does not seem to be much different from the multiplication circuit using Cuccaro's adder in terms of gate complexity, the multiplication circuit using Cuccaro's adder has a too high depth complexity. Also, our multiplier has the second-lowest depth complexity. The multiplication circuit using Pham-Svore's adder has the lowest depth complexity but too high gate complexity.

TABLE 1. Comparison of quantum modular multipliers.

Method	Resources	Cuccaro (mod 2^n)	Draper (mod 2^n)	Pham-Svore (mod 2^n)	[13] (mod 2^n)	proposed (mod 2^n)	proposed (mod $2^n - 1$)
quantum	Qubits	$3n$	$5n$	$16n^2$	$5n$	$5n$	$5n$
-	Gates	$12n^2$	$60n^2$	$384n^2 \log_2 n$	$20n^2$	$10n^2$	$10n^2$
classical	Depth	$12n^2$	$24n \log_2 n$	$56 \log_2 n$	$8n \log_2 n$	$4n \log_2 n$	$4n \log_2 n$
quantum	Qubits	X	X	X	X	$6n$	$6n$
-	gates	X	X	X	X	$11n^2$	$11n^2$
quantum	Depth	X	X	X	X	$\frac{1}{2}n^2$	n^2



(a) the number of qubits (y axis) for n qubits (x axis) (b) the number of gates (y axis) for n qubits (x axis) (c) the number of depth (y axis) for n qubits (x axis)

FIGURE 12. The quantum resource complexity comparison of quantum-classical modular multipliers (mod 2^n).

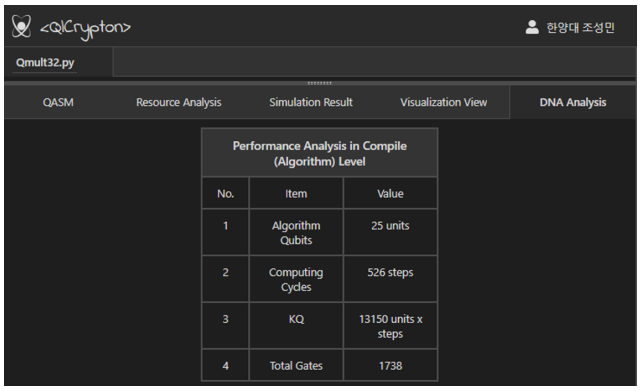


FIGURE 13. Quantum resource analyzation of quantum-quantum modular multiplier over GF($32 = 2^n (n = 5)$) by ETRI Qcrypton.

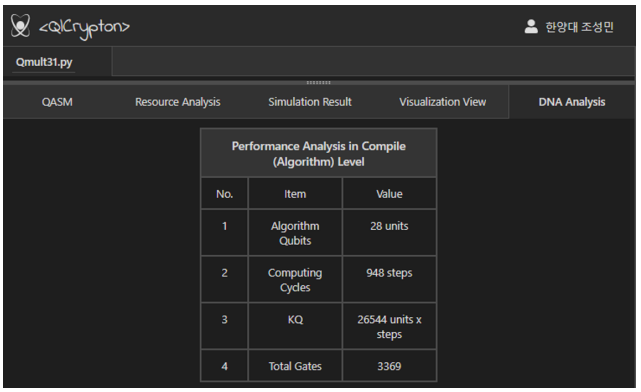


FIGURE 14. Quantum resource analyzation of quantum-quantum modular multiplier over GF($31 = 2^n - 1 (n = 5)$) by ETRI Qcrypton.

B. ANALYSIS USING QCRYPTON

To evaluate the feasibility of our quantum algorithms in the practical quantum computing situation, it is necessary to analyze the amount of quantum resources required. But existing quantum simulators focused on statistical examination that simply calculates the performance and resources based on individual quantum computing components. ETRI proposed Qcrypton, which provides an analysis of quantum resources and performance based on the practical quantum computing situation. We used ETRI Qcrypton to analyze the quantum resources and performance of our quantum modular multipliers. We implemented quantum modular multipliers over the GF($2^5 - 1$) and GF(2^5), when $n = 5$, for quantum resource analysis through Qcrypton. Fig.13 and Fig.14 show the results of the analysis of our multipliers over the GF($2^5 - 1$) and GF(2^5) using ETRI Qcrypton.

The KQ (KQ = # of Algorithm Qubits $\times$ Computing Cycles) in Fig.13 and Fig.14 represent the circuit cost of the two quantum modular multipliers in the practical situation. The quantum modular multiplier over GF(2^5) uses 25 qubits and 1738 quantum gates with 526 quantum circuit depth. And the quantum modular multiplier over GF($2^5 - 1$) uses 28 qubits and 3369 gates with 948 circuit depth. Because the performance of quantum algorithms can vary under the influence of various quantum computing components such as distance between qubits, we can more accurately analyze our quantum modular multipliers by implementing quantum circuits through Qcrypton and simulating them in the practical quantum computing situation.

VI. CONCLUSION

In this paper, we designed the quantum-classical and quantum-quantum modular multipliers by applying a bit shift method that transforms the partial product to reduce the complexity of the quantum circuit by eliminating the reduction stage. Our quantum-classical modular multipliers

pre-calculate the transformed partial product using the classical value, thus there is no need for a reduction stage on a quantum circuit. In addition, our quantum-quantum modular multipliers also do not require a reduction stage by computing transformed partial products using only Toffoli gates on a quantum circuit. To analyze the quantum resource required for our proposed quantum modular multipliers, we calculated computational resources through statistical examination. Then we implemented our quantum modular multipliers using Qcrypton and analyzed the complexity of quantum resources and performance while considering the practical quantum computing situation.

REFERENCES

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, and B. Burkett, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 779, pp. 505–510, 2019.
[2] D. Deutsch, "Quantum theory, the Church–Turing principle and the universal quantum computer," *Proc. Roy. Soc. London A, Math. Phys. Sci.*, vol. 400, no. 1818, pp. 97–117, 1985.
[3] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th Annu. Symp. Found. Comput. Sci.*, Nov. 1994, pp. 124–134.
[4] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997.
[5] P. Gossett, "Quantum carry-save arithmetic," 1998, *arXiv:quant-ph/9808061*. [Online]. Available: https://arxiv.org/abs/quant-ph/9808061
[6] K. Hwang, *Computer Arithmetic: Principles, Architecture, and Design*. Hoboken, NJ, USA: Wiley, 1979.
[7] T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore, "A logarithmic-depth quantum carry-lookahead adder," *Quantum Inf. Comput.*, vol. 6, nos. 4–5, pp. 351–369, 2006.
[8] "Description of a relay calculator," Ann. Comput. Lab., Harvard Univ., Cambridge, MA, USA, Tech. Rep., vol. 24, 1949.
[9] A. Weinberger and J. L. Smith, "A one-microsecond adder using one-megacycle circuitry," *IEEE Trans. Electron. Comput.*, vol. EC-5, no. 2, pp. 65–73, Jun. 1956.
[10] Y. Ofman, "On the algorithmic complexity of discrete functions," *Sov. Phys. Doklady* , vol. 7, no. 7, pp. 589–591, 1963.
[11] P. L. Montgomery, "Modular multiplication without trial division," *Math. Comput.*, vol. 44, no. 170, pp. 519–521, Apr. 1985.
[12] P. Barrett, "Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor," in *Advances in Cryptology*, vol. 263. 1987, pp. 311–323.
[13] R. Rines and I. Chuang, "High performance quantum modular multipliers," 2018, *arXiv:1801.01081*. [Online]. Available: http://arxiv.org/abs/1801.01081
[14] Y. Hwang, T. Kim, C. Baek, and B.-S. Choi, "Integrated analysis of performance and resources in large-scale quantum computing," *Phys. Rev. A, Gen. Phys. Appl.*, vol. 13, no. 5, May 2020, Art. no. 054033.
[15] V. Vedral, A. Barenco, and A. Ekert, "Quantum networks for elementary arithmetic operations," *Phys. Rev. A, Gen. Phys.*, vol. 54, no. 1, pp. 147–153, Jul. 1996. [Online]. Available: http://xxx.lanl.gov/abs/quant-ph/9511018
[16] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," *Quantum Inf. Comput.*, vol. 3, no. 4, pp. 317–344, 2003.

SEONG-MIN CHO (Student Member, IEEE) received the B.S. degree from the Division of Electrical Engineering, Hanyang University ERICA, South Korea, in 2019. He is currently pursuing the master's degree with the Department of Electrical Engineering, Hanyang University.

His research interests include the IoT security, security of embedded systems, and post-quantum cryptography.

AEYOUNG KIM (Member, IEEE) received the B.S. degree in computer science and statistics from Hanshin University, South Korea, in 2000, and the M.S. and Ph.D. degrees in computer science and engineering from Ewha Womans University, South Korea, in 2003 and 2012, respectively.

She was a Postdoctoral Researcher and a Research Professor of computer science and engineering with Ewha Womans University for a period of four years and a Researcher with the Cryptographic Technology Team, National Institute for Mathematical Sciences (NIMS), South Korea, for a period of two years. Since 2018, she has been a Research Professor with Hanyang University. She is currently a Research Professor with Hanyang University. Her research interests include post-quantum cryptography (optimization and quantum analysis), lightweight cryptography (the IoT and blockchain security), and bio-cryptography (privacy and key share with biometrics).

DOOHO CHOI (Member, IEEE) received the B.S. degree in mathematics from Sungkyunkwan University, South Korea, in 1994, and the M.S. and Ph.D. degrees in mathematics from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 1996 and 2002, respectively.

Since 2002, he has been a Principal Researcher with the Electronics and Telecommunications Research Institute (ETRI). Since 2015, he has been a Professor with the University of Science and Technology (UST). From 2016 to 2017, he was a Visiting Research Fellow with Queens University Belfast, U.K. His main research interests include side channel analysis and its countermeasure design, quantum crypto analysis, and security technologies of IoT.

BYUNG-SOO CHOI (Member, IEEE) received the B.S. degree in computer engineering from Chungnam National University, South Korea, in 1996, and the M.S. and Ph.D. degrees in information and communication from the Gwangju Institute of Science and Technology (GIST), South Korea, in 1998 and 2004, respectively.

From 2004 to 2006, he was a Postdoctoral Researcher working on quantum algorithms with The University of York, U.K. From 2011 to 2013, he was a Research Scientist working on fault-tolerant quantum computation with Duke University, USA. From 2013 to 2015, he was a Researcher working on Quantum-Dot qubit device with The University of Tokyo, Japan. Since 2015, he has been a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI) working on quantum computing research and development. His main research interests include all areas of quantum computing from device implementation to quantum algorithms.

SEUNG-HYUN SEO (Member, IEEE) received the B.S. degree from the Department of Mathematics, Ewha Womans University, Seoul, South Korea, in 2000, and the M.S. and Ph.D. degrees in computer science from Ewha Womans University, in 2002 and 2006, respectively.

She was an Assistant Professor with Korea University, Sejong, for a period of two years. She was a Postdoctoral Researcher of computer science with Purdue University for a period of two and half years, a Senior Researcher with the Korea Internet and Security Agency (KISA) for a period of two years, and a Researcher for a period of three years with the Financial Security Agency (FSA), South Korea. She is currently a Professor with Hanyang University. Her main research interests include cryptography, the IoT security, mobile security, secure cloud computing, and malicious code analysis.