



OPEN **Optimized quantum folding Barrett reduction for quantum modular multipliers**

Jian Zhang¹, Seong-Min Cho¹, Changyeol Lee¹ & Seung-Hyun Seo²✉

Due to the reversibility constraints of quantum circuits, the development of modular operations in quantum modular multiplication has been significantly limited. In this work, we propose an optimized modular operation scheme based on Barrett reduction and implement quantum circuits for three versions of Barrett reduction, including our optimized approach. The proposed quantum modular operation circuit can be integrated with quantum-quantum and quantum-classical multiplication circuits. Additionally, we analyze the quantum resource requirements of the proposed circuit and compare the results across the three versions. Our optimized folding Barrett reduction scheme demonstrates superior performance. Furthermore, general quantum modular reduction relies on recursive subtractors and controlled adders to avoid division operation, resulting in a T-depth of approximately $O(2^n)$, where n is the bit length of the modulus. In contrast, our approach achieves a T-depth of only $O(n^2)$ by utilizing the Barrett reduction.

Quantum computers can solve numerous problems faster, based on quantum properties such as superposition and entanglement, than classical computers. For example, Shor's algorithm¹, proposed by Peter Shor in 1994, solves the integer factorization problem in polynomial time. Grover's algorithm², proposed by Lov Grover in 1996, reduces the complexity of unstructured search problems from $O(n)$ in classical computing to $O(\sqrt{n})$ in quantum computing. The construction of quantum algorithms relies on many quantum arithmetic circuits, which are composed of quantum gates that execute operations on qubits. In particular, quantum algorithms, including Shor's algorithm, are required to solve mathematical problems, requiring quantum operations over modular arithmetic.

Currently, extensive studies are concentrated on designing quantum arithmetic circuits for addition and multiplication. However, studies on quantum circuit designs for modular addition and multiplication have been less advanced due to the difficulty of converting the reduction process to a value less than the modulus number into a reversible quantum circuit. The conventional approach for modular multiplication is to first compute the product of the input values and then derive the final result using division. However, due to the reversibility requirement of quantum circuits, classical modular multiplication is difficult to implement in quantum circuits. Reversibility mainly limits the conversion of division operations. In recent years, some studies have attempted to directly convert the classical division operation into quantum circuits. Yuan et al.³ proposed a quantum circuit based on the binary long division algorithm, which is composed of quantum comparators and quantum subtractors. Their quantum subtractor circuit reduces qubit usage by directly performing a "reset operation" to the auxiliary qubit to $|0\rangle$, but does not account for the quantum gate cost associated with qubit initialization. In practice, the two common modular multiplication methods are the Montgomery algorithm⁴ and Barrett reduction⁵. The Montgomery algorithm utilizes a specific representation known as the "Montgomery form". Initially, it converts the numbers into Montgomery form, subsequently executing modular operations. The conversion eliminates the necessity of division operations during the computation. The Barrett reduction, conversely, precomputes parameters to convert the division operation into multiplication and bit-shifting operations. In classical modular multiplication, Barrett reduction provides benefits over the Montgomery algorithm as it requires only a single precomputation of parameters for multiple reductions utilizing the same modulus.

In 1986, Barrett proposed a classical modular multiplication algorithm⁵ suitable for hardware implementation, which reduces the resource usage in the modular operation by avoiding division operations. To improve the efficiency of the Barrett reduction in practical applications, Kong⁶ optimized the parameters of the approximation process, reducing the number of subtractions required for result correction to a single instance. Hasenplaugh et al.⁷ proposed the folding Barrett reduction algorithm, a recursive method that enhances the efficiency by

¹Department of Electrical Engineering, Graduate School of Hanyang University, Seoul 04763, South Korea. ²Division of Electrical Engineering, Hanyang University ERICA, Ansan 15588, South Korea. ✉email: seosh77@hanyang.ac.kr

reducing the length of integers requiring modular reduction. Wu et al.⁸ further improved the folding Barrett reduction by combining Kong's method, optimizing the correction process to require only a single subtraction, and using the Karatsuba algorithm for integer multiplication to improve the computational efficiency.

Although any classical algorithm can be converted into a reversible circuit in principle⁹, careful consideration must be given to the use of quantum resources during the circuit conversion process. Generally, classical algorithms possess substantial size and circuit depth after conversion. Consequently, even though classical modular multiplication can be converted into quantum circuits, substantial efficiency constraints exist. In 2018, Rines et al.¹⁰ proposed an efficient theoretical model for a quantum modular multiplication circuit using Barrett reduction, but they did not present specific quantum circuits. Thus, there remains a lack of practical quantum circuit models for modular multiplication arithmetic.

In this paper, we propose an optimized variant of Barrett reduction and the corresponding quantum circuit derived from it. Moreover, we implemented three versions of the quantum Barrett reduction circuit based on identical concepts. In contrast to the Barrett reduction proposed by Wu et al.⁸, which is exclusively combinable with Karatsuba multiplication, our approach is compatible with the Karatsuba, Toom-Cook, and Schönhage-Strassen multiplication algorithms. Furthermore, the theoretical quantum circuit model, proposed by Rines et al.¹⁰, focuses solely on quantum-classical modular multiplication without providing detailed design or analysis for quantum-quantum modular multiplication. Our quantum circuit, however, applies to both quantum-classical and quantum-quantum modular multiplication. In the NISQ (Noisy Intermediate Scale Quantum) era, the use of quantum resources is still greatly limited. The actual consumption of quantum resources is an important indicator of the efficiency of a quantum algorithm. Unlike the estimate of Yuan et al.³, our analysis explicitly includes the gate overhead required to reset ancilla qubits. As an important component of Shor's algorithm, modular exponential operation can be decomposed into sequential modular multiplication operations with a fixed modulus N . Our scheme precomputes the parameters once and reuses them throughout the sequence, which gives it a decisive advantage over division-based and other simplified methods. Based on our quantum circuit designs, we estimated the quantum resources required for each quantum circuit. Our optimized quantum folding Barrett reduction circuit uses about half the T-depth of a general quantum Barrett reduction circuit. Additionally, compared to the quantum resources required for general quantum modular operations, our proposed approach shows superior performance in terms of T-depth.

Preliminaries

We introduce the general modular multiplication methods using Barrett reduction and elementary quantum gates of quantum circuits in this section.

Barrett reduction

General Barrett reduction

Barrett reduction⁵ is an efficient method for modular reduction, particularly suitable for computational scenarios where division operations are costly. This is particularly advantageous when multiple modular operations must be performed using the same modulus. The core idea of Barrett reduction is to replace division with multiplication and bit-shifting operations. Given a modulus N and an integer t that needs to be reduced, Barrett reduction first requires the precomputation of a parameter $\mu = \lfloor \frac{2^{2n}}{N} \rfloor$, where n is the bit length of the modulus. The Barrett reduction can be defined as $R = t - \hat{q}N$, where $\hat{q} = \lfloor \lfloor \frac{t}{2^{n-1}} \rfloor \frac{\mu}{2^{n+1}} t \rfloor$. By estimating the quotient \hat{q} , the value of R is constrained to lie within in range $0 \leq R < 3N - 1$. Typically, at most two additional subtractions of the modulus N are required to obtain the final result. The description of this procedure can be found in Algorithm 1.

Input: Modulus $2^{n-1} < N < 2^n$, Integer $0 \leq t < 2^{2n}$

Output: $r = t \pmod N$

- 1: Precompute $\mu = \lfloor \frac{2^{2n}}{N} \rfloor$
 - 2: $\hat{q}' = \lfloor \lfloor \frac{t}{2^{n-1}} \rfloor \frac{\mu}{2^{n+1}} t \rfloor$
 - 3: $r_0 = (t - \hat{q}'N) \pmod{2^{n+1}}$
 - 4: $r_1 = r_0 - N$
 - 5: **if** $r_1 \geq 0$ **then**
 - 6: $r = r_1$
 - 7: **else**
 - 8: $r = r_0$
 - 9: **end if**
 - 10: **return** r
-

Algorithm 1. General Barrett reduction.

Folding Barrett reduction

The folding Barrett reduction⁷ modifies the general Barrett reduction process to reduce the size of the integers involved in the modular operation. Unlike the general Barrett reduction, which requires only one precomputed

value, the folding Barrett reduction relies on two constants related to the modulus N . These constants can be precomputed since they only depend on the modulus N . Assuming the modulus length is n and $s = \frac{n}{2}$, the constants are $\mu = \lfloor \frac{2^{3s}}{N} \rfloor$ and $N' = 2^{3s} \bmod N$. Using the constant N' , an integer t of length $4s$ can be reduced to $3s + 1$ bits, represented as $t' = t \bmod 2^{3s} + \lfloor \frac{t}{2^{3s}} \rfloor N'$. It shows that $t' = t \bmod N$ and this gives $t' \bmod N$ as the result instead of $t \bmod N$, effectively reducing the computational effort required. The estimated quotient is represented as $\hat{q} = \lfloor \lfloor \frac{t'}{2^{2s}} \rfloor \frac{\mu}{2^s} \rfloor$, and the final result is $R = t' - \hat{q}N$. Since R lies between 0 and $4N - 1$, at most three additional subtractions are required to correct the result. The operation of the folding Barrett Reduction is provided in Algorithm 2.

Input: Modulus $2^{n-1} < N < 2^n$, Integer $0 \leq t < 2^{2n}$, $s = \frac{n}{2}$

Output: $r = t \bmod N$

- 1: Precompute $N' = 2^{3s} \bmod N$, $\mu = \lfloor \frac{2^{3s}}{N} \rfloor$
 - 2: $t' = t \bmod 2^{3s} + \lfloor \frac{t}{2^{3s}} \rfloor N'$
 - 3: $\hat{q}' = \lfloor \lfloor \frac{t'}{2^{2s}} \rfloor \frac{\mu}{2^s} \rfloor$
 - 4: $r_0 = (t' - \hat{q}'N) \bmod 2^{2s+1}$
 - 5: $r_1 = r_0 - N$
 - 6: **if** $r_1 \geq 0$ **then**
 - 7: $r = r_1$
 - 8: **else**
 - 9: $r = r_0$
 - 10: **end if**
 - 11: **return** r
-

Algorithm 2. Folding Barrett reduction.

Modular multiplier with folding Barrett reduction

To minimize the number of subtractions required for result correction in folding Barrett reduction, Wu et al.⁸ combined the folding Barrett reduction with a method proposed by Kong⁶ that reduces the number of correction operations. They proposed a modular multiplier based on folding Barrett reduction that requires only one additional subtraction. Additionally, they applied the Karatsuba algorithm for integer multiplication, significantly reducing the resources needed compared to classical schoolbook multiplication methods. As in the folding Barrett reduction, this approach assumes a modulus length of n and $s = \frac{n}{2}$. The precomputed constants are $\mu = \lfloor \frac{2^{3s+3}}{N} \rfloor$ and $N' = 2^{3s} \bmod N$. Using these constants, the length of the integer t needing modular reduction is reduced from $4s$ bits to $3s + 2$ bits. The estimated quotient is given by $\hat{q} = \lfloor \lfloor \frac{t'}{2^{2s-2}} \rfloor \frac{\mu}{2^{s+5}} \rfloor$. By adjusting the parameters during computation, the number of subtractions required for final result correction is reduced to one, thereby enhancing the overall computational efficiency of the algorithm.

Quantum arithmetic circuits

Quantum arithmetic circuits are fundamental building blocks in quantum computing. These operations include, but are not limited to, addition, subtraction, and multiplication. They are constructed from basic quantum gates and play a crucial role in quantum algorithms.

(1) Quantum addition/subtraction

The quantum adder is the most fundamental module in quantum arithmetic, serving as the cornerstone for building other quantum computing circuits. According to the way of handling the carry in bitwise addition, quantum adders are primarily classified into a quantum ripple-carry adder, quantum carry-save adder, quantum carry-lookahead adder, and quantum hybrid adder.

The ripple-carry structure is the simplest architecture among quantum adders. The first quantum adder utilizing this structure was proposed by Vedral et al. in 1996¹¹. In a ripple-carry adder, the carry from each bitwise addition is sequentially propagated to the next bit, resulting in a circuit depth that scales linearly with the number of input qubits. In 2018, Gidney proposed a new quantum ripple-carry adder¹² that reduces the number of T gates by using logical-AND gates instead of Toffoli gates. By computing and then uncomputing the logical-AND of two qubits, the circuit significantly halved the cost of the quantum ripple-carry adder. The primary feature of the quantum carry-save adder is that it defers the immediate processing of carries generated from lower bits during bitwise addition. Instead, these carries are temporarily stored for subsequent handling. This method reduces the delay associated with addition operations and enhances computational efficiency. The carry-lookahead structure significantly accelerates addition by handling carries in parallel. In 2008, Draper et al.¹³ proposed an efficient quantum carry-lookahead adder based on this architecture. The quantum hybrid adder combines the qubit-count and Toffoli-count advantages of the quantum ripple-carry adder with the Toffoli-

depth benefits of the quantum carry-lookahead adder. In 2008, Takahashi et al.¹⁴ proposed the first quantum hybrid adder by integrating the strengths of these two adder architectures.

The quantum subtractor has a similar structure to the quantum adder. By making slight modifications to the adder, a subtraction circuit can be constructed. In both one's-complement and two's-complement arithmetic, we have the identity $a - b = (a' + b)'$, where the symbol $'$ denotes bitwise complementation. Hence, by adding only two time slices, each composed exclusively of the NOT gates, a quantum circuit of the subtraction can be realized.

(2) Quantum multiplication

The quantum general schoolbook multiplier is a method that achieves multiplication by repeatedly utilizing quantum adders to compute the result. In general, multiplication is performed by bitwise multiplication of two integers, followed by sequential addition of partial products.

Due to the reversibility of the quantum circuits, quantum multiplication circuits are typically divided into three stages: partial product setting, addition, and partial product cleanup. The addition stage is implemented using quantum adder circuits. Based on the type of inputs, quantum multiplication circuits can be categorized into quantum-quantum multipliers and quantum-classical multipliers. In the partial product setting stage, these two types of multipliers use different quantum gates to implement the setting operations. In quantum-quantum multipliers, the Toffoli gates are used to set partial products between two qubits, whereas in quantum-classical multipliers, the NOT gates are employed to set partial products between qubits. Finally, in the partial product cleanup stage, the inverse circuit of the partial product setting can be used to eliminate the partial products. After this stage, the ancillary qubits in the quantum circuit are reset to their initial states, making them available for subsequent operations.

Quantum Barrett reduction

In this section, we propose an efficient folding Barrett reduction and its quantum circuit implementation. We introduce our optimized algorithm in "Optimized folding Barrett reduction" and compare the complexity of three versions of Barrett reduction in "Comparison of reduction algorithm". In "Quantum implementation of optimized folding Barrett reduction", we show the detailed design of a quantum circuit based on the optimized folding Barrett reduction.

Optimized folding Barrett reduction

Wu et al.⁸ proposed a modular multiplier based on improved folding Barrett reduction⁶, which reduces the number of final subtractions for correction and enhances the efficiency of the multiplier by integrating the Karatsuba multiplication algorithm. Currently, various multiplication algorithms exist for computing integer products. To improve the applicability of the reduction algorithm in modular multiplication, we propose an optimized folding Barrett reduction.

First, given an n -bit modulus N , we precompute two constant values based on the modulus N : $\mu = \lfloor \frac{2^{3s+3}}{N} \rfloor$ and $N' = 2^{3s} \bmod N$. Next, following the method proposed by Hasenplaugh et al.⁷, a $4s$ -bit integer t can be folded into a $3s + 1$ -bit integer $t' = t \bmod 2^{3s} + \lfloor \frac{t}{2^{3s}} \rfloor N'$, where both terms are $3s$ -bit long integers. This folding operation reduces the size of the multiplications involved in the Barrett reduction process. Finally, by utilizing the precomputed constant μ , the final result $r_0 = (t' - \lfloor \frac{t'}{2^{2s-2}} \rfloor \lfloor \frac{\mu}{2^{s+5}} \rfloor N) \bmod 2^{2s+1}$ can be computed by two multiplications and one subtraction, and only one additional final subtraction is required to correct the result. The series of operations in Fig. 1 illustrates the computing process of our algorithm.

Let $\mu = \lfloor \frac{2^{3s+3}}{N} \rfloor < 2^{3s+3-(2s-1)} = 2^{s+4}$, then the quotient may be represented as $q = \lfloor \frac{t'}{N} \rfloor = \lfloor \frac{1}{2^{s+5}} \times \frac{t'}{2^{2s-2}} \times \frac{2^{3s+3}}{N} \rfloor$ and the quotient estimate can be defined as $\hat{q} = \lfloor \frac{1}{2^{s+5}} \times \lfloor \frac{t'}{2^{2s-2}} \rfloor \times \lfloor \frac{2^{3s+3}}{N} \rfloor \rfloor$. Since t' is a $3s + 1$ -bit integer, $t' < 2^{3s+1}$. Thus, the relation between q and \hat{q} can be written as follows:

$$0 \leq \hat{q} \leq q = \left\lfloor \frac{1}{2^{s+5}} \times \frac{t'}{2^{2s-2}} \times \frac{2^{3s+3}}{N} \right\rfloor$$

Let α and β , where $0 \leq \alpha, \beta < 1$, be defined as $\alpha = \frac{t'}{2^{2s-2}} - \lfloor \frac{t'}{2^{2s-2}} \rfloor$ and $\beta = \frac{2^{3s+3}}{N} - \lfloor \frac{2^{3s+3}}{N} \rfloor$, the inequality can thus be expressed as follows.

$$\begin{aligned} 0 \leq \hat{q} \leq q &= \left\lfloor \frac{1}{2^{s+5}} \times \left(\left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor + \alpha \right) \times \left(\left\lfloor \frac{2^{3s+3}}{N} \right\rfloor + \beta \right) \right\rfloor \\ &= \left\lfloor \frac{1}{2^{s+5}} \times \left(\left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor \times \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor + \left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor \beta + \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor \alpha + \alpha \beta \right) \right\rfloor \\ &< \left\lfloor \frac{1}{2^{s+5}} \times \left(\left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor \times \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor + \left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor + \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor + 1 \right) \right\rfloor \\ &= \left\lfloor \hat{q} + \frac{1}{2^{s+5}} \times \left(\left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor + \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor + 1 \right) \right\rfloor \end{aligned}$$

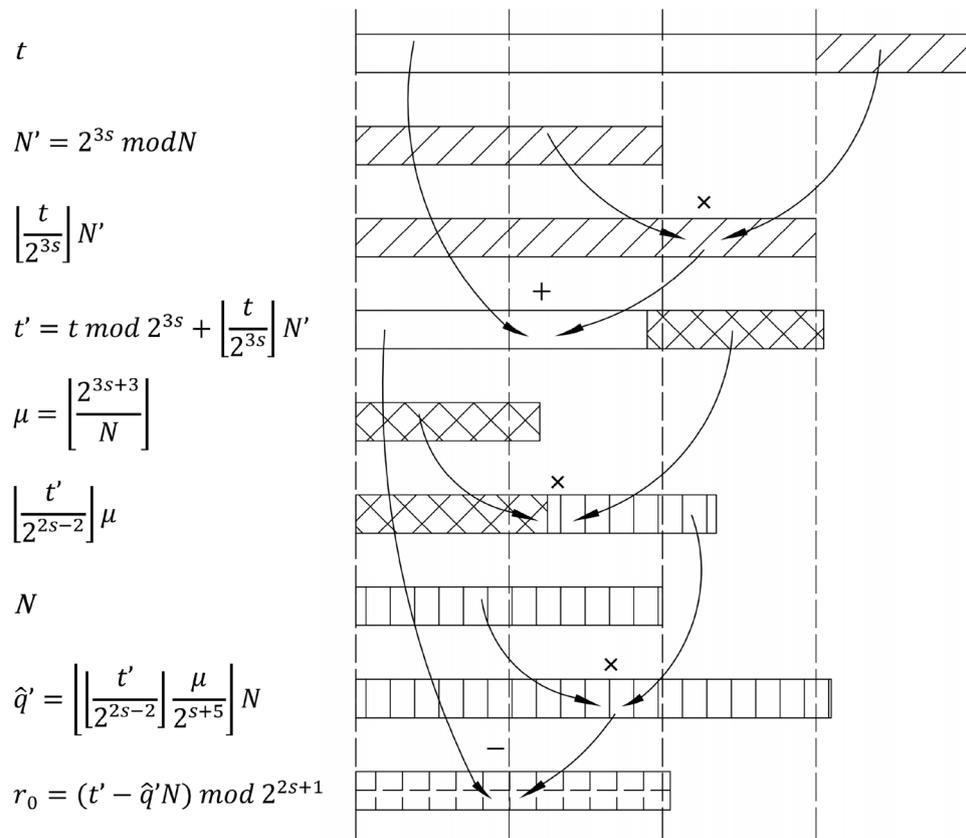


Fig. 1. Computing process of the optimized folding Barrett reduction.

Since t' is a $3s + 1$ -bit integer and $\mu = \lfloor \frac{2^{3s+3}}{N} \rfloor < 2^{s+4}$, the above inequality becomes:

$$0 \leq \hat{q} \leq q < \left\lfloor \hat{q} + \frac{2^{s+3} + 2^{s+4} + 1}{2^{s+5}} \right\rfloor$$

$$0 \leq \hat{q} \leq q < \hat{q} + 1$$

By the evaluation, the estimation of the quotient will be $q - 1 < \hat{q} \leq q$. Meanwhile, $R = t' - qN$ should be range in $[0, N)$, thus the remainder will adhere to $R = t - \hat{q}N \in [0, 2N)$. Consequently, at most one additional modular subtraction would need to be performed. The entire reduction procedure is described in Algorithm 3.

Input: Modulus $2^{n-1} < N < 2^n$, Integer $0 \leq t < 2^{2n}$, $s = \frac{n}{2}$

Output: $r = t \pmod N$

- 1: Precompute $N' = 2^{3s} \pmod N$, $\mu = \lfloor \frac{2^{3s+3}}{N} \rfloor$
- 2: $t' = t \pmod{2^{3s}} + \lfloor \frac{t}{2^{3s}} \rfloor N'$
- 3: $\hat{q}' = \lfloor \lfloor \frac{t'}{2^{2s-2}} \rfloor \frac{\mu}{2^{s+5}} \rfloor$
- 4: $r_0 = (t' - \hat{q}'N) \pmod{2^{2s+1}}$
- 5: $r_1 = r_0 - N$
- 6: **if** $r_1 \geq 0$ **then**
- 7: $r = r_1$
- 8: **else**
- 9: $r = r_0$
- 10: **end if**
- 11: **return** r

Algorithm 3. Optimized folding Barrett reduction.

Comparison of reduction algorithm

The main differences between the various Barrett reduction algorithms are the choice of parameters during the computation, the number of final subtractions required for result correction, and the size of the additions, subtractions, and multiplications involved.

In “Optimized folding Barrett reduction”, we have already analyzed the number of subtractions required for the final result correction in the optimized folding Barrett reduction. We applied the same analytical method to the other two versions of the Barrett reduction for comparison.

(1) General Barrett reduction

Following the representation of the optimized folding Barrett reduction as described above, the quotient and the estimated quotient can be defined as $q = \lfloor \frac{t}{N} \rfloor = \lfloor \frac{1}{2^{n+1}} \times \frac{t}{2^{n-1}} \times \frac{2^{2n}}{N} \rfloor$ and $\hat{q} = \lfloor \frac{1}{2^{n+1}} \times \lfloor \frac{t}{2^{n-1}} \rfloor \times \lfloor \frac{2^{2n}}{N} \rfloor \rfloor$, respectively. In this form, the precomputed constant is $\mu = \lfloor \frac{2^{2n}}{N} \rfloor$. By the definition of the floor function, the relationship between these two quotients can be derived as follows:

$$0 \leq \hat{q} \leq q = \left\lfloor \frac{1}{2^{n+1}} \times \frac{t}{2^{n-1}} \times \frac{2^{2n}}{N} \right\rfloor$$

We first define α and β , where $0 \leq \alpha, \beta < 1$ such that $\alpha = \frac{t}{2^{n-1}} - \lfloor \frac{t}{2^{n-1}} \rfloor$ and $\beta = \frac{2^{2n}}{N} - \lfloor \frac{2^{2n}}{N} \rfloor$. Then the relationship can be rewritten as below.

$$0 \leq \hat{q} \leq q = \left\lfloor \frac{1}{2^{n+1}} \times \left(\lfloor \frac{t}{2^{n-1}} \rfloor + \alpha \right) \times \left(\left\lfloor \frac{2^{2n}}{N} \right\rfloor + \beta \right) \right\rfloor$$

After the distribution and the bounds on the value α and β , q can be represented by the following inequality.

$$\begin{aligned} 0 \leq \hat{q} \leq q &< \left\lfloor \frac{1}{2^{n+1}} \times \left(\lfloor \frac{t}{2^{n-1}} \rfloor \times \left\lfloor \frac{2^{2n}}{N} \right\rfloor + \lfloor \frac{t}{2^{n-1}} \rfloor + \left\lfloor \frac{2^{2n}}{N} \right\rfloor + 1 \right) \right\rfloor \\ &= \left\lfloor \hat{q} + \frac{1}{2^{n+1}} \times \left(\lfloor \frac{t}{2^{n-1}} \rfloor + \left\lfloor \frac{2^{2n}}{N} \right\rfloor + 1 \right) \right\rfloor \end{aligned}$$

To find the upper bound of q , we evaluate this inequality at $t = 2^{2n}$ and $\mu = \lfloor \frac{2^{2n}}{N} \rfloor < 2^{n+1}$.

$$\begin{aligned} 0 \leq \hat{q} \leq q &< \left\lfloor \hat{q} + \frac{2^{n+1} + 2^{n+1} + 1}{2^{n+1}} \right\rfloor \\ 0 \leq \hat{q} \leq q &< \hat{q} + 2 \end{aligned}$$

Thus, the estimated quotient \hat{q} will be at most equal to q and at least equal to $q - 2$. Substituting this into the estimate for the result, we get that $R = t - \hat{q}N < t - (q - 2)N = t - qN + 2N \in [0, 3N)$. Clearly, at most two additional subtractions by the modulus need to be applied to arrive at the correct remainder.

(2) Folding Barrett reduction

Following the same derivation procedure as before, the corresponding quotient can be described as follows:

$$\begin{aligned} q &= \left\lfloor \frac{t'}{N} \right\rfloor = \left\lfloor \frac{1}{2^s} \times \frac{t'}{2^{2s}} \times \frac{2^{3s}}{N} \right\rfloor \\ \hat{q} &= \left\lfloor \frac{1}{2^s} \times \left\lfloor \frac{t'}{2^{2s}} \right\rfloor \times \left\lfloor \frac{2^{3s}}{N} \right\rfloor \right\rfloor \end{aligned}$$

And the relationship can be written as $0 \leq \hat{q} \leq q = \left\lfloor \frac{1}{2^s} \times \frac{t'}{2^{2s}} \times \frac{2^{3s}}{N} \right\rfloor$. Then, the value $\alpha = \frac{t'}{2^{2s}} - \lfloor \frac{t'}{2^{2s}} \rfloor$ and $\beta = \frac{2^{3s}}{N} - \lfloor \frac{2^{3s}}{N} \rfloor$, where $0 \leq \alpha, \beta < 1$, can be used to rewrite this inequality.

$$\begin{aligned} 0 \leq \hat{q} \leq q &= \left\lfloor \frac{1}{2^s} \times \left(\left\lfloor \frac{t'}{2^{2s}} \right\rfloor + \alpha \right) \times \left(\left\lfloor \frac{2^{3s}}{N} \right\rfloor + \beta \right) \right\rfloor \\ &= \left\lfloor \frac{1}{2^s} \times \left(\left\lfloor \frac{t'}{2^{2s}} \right\rfloor \times \left\lfloor \frac{2^{3s}}{N} \right\rfloor + \left\lfloor \frac{t'}{2^{2s}} \right\rfloor \beta + \left\lfloor \frac{2^{3s}}{N} \right\rfloor \alpha + \alpha \beta \right) \right\rfloor \\ &< \left\lfloor \frac{1}{2^s} \times \left(\left\lfloor \frac{t'}{2^{2s}} \right\rfloor \times \left\lfloor \frac{2^{3s}}{N} \right\rfloor + \left\lfloor \frac{t'}{2^{2s}} \right\rfloor + \left\lfloor \frac{2^{3s}}{N} \right\rfloor + 1 \right) \right\rfloor \end{aligned}$$

$$= \left\lfloor \hat{q} + \frac{1}{2^s} \times \left(\left\lfloor \frac{t'}{2^{2s}} \right\rfloor + \left\lfloor \frac{2^{3s}}{N} \right\rfloor + 1 \right) \right\rfloor$$

Similarly, t' is a $3s + 1$ -bit integer and $\mu = \left\lfloor \frac{2^{3s}}{N} \right\rfloor < 2^{s+1}$. We evaluate this inequality as

$$0 \leq \hat{q} \leq q < \left\lfloor \hat{q} + \frac{2^{s+1} + 2^{s+1} + 1}{2^s} \right\rfloor$$

$$0 \leq \hat{q} \leq q < \hat{q} + 4$$

$$0 \leq \hat{q} \leq q \leq \hat{q} + 3$$

Thus, the estimated quotient \hat{q} will be at most equal to q and at least equal to $q - 3$. And the estimate of the result will be $R = t - \hat{q}N \leq t - (q - 3)N = t - qN + 3N \in [0, 4N]$. Therefore, at most three additional subtractions by the modulus need to be performed to correct the final remainder.

As shown in Table 1, by adjusting the size of the constants during the precomputation process, the number of subtractions for result correction can be optimized.

To evaluate the size of the arithmetic operations in each algorithm, we analyzed the input size at each step of the calculation process for each algorithm. For example, in the case of our optimized folding Barrett reduction, we conducted a detailed analysis of the input and output sizes involved at each stage of the computation.

As shown in Algorithm 3, without considering the result correction step, after the precomputation of the two constants, the entire algorithm consists of three multiplications, one addition, and one subtraction operation.

In step 2 we calculate the intermediate value $t' = t \bmod 2^{3s} + \left\lfloor \frac{t}{2^{3s}} \right\rfloor N'$ by one multiplication and one addition. Since t is a $2n$ -bit integer, where $n = 2s$, the first term can be considered as the least significant $3s$ bits of t . And the second term is computed by multiplication of size $(s \times 2s)$ -bit, where $\left\lfloor \frac{t}{2^{3s}} \right\rfloor$ represents a right shift operation by $3s$ bits on t , and N' is a precomputed constant of $2s$ bits. Then, the result of the folding step is computed via one $(3s + 3s)$ -bit addition and the result of this addition is a $(3s + 1)$ -bit integer.

In step 3 we calculate the estimated quotient $\hat{q}' = \left\lfloor \left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor \frac{\mu}{2^{s+5}} \right\rfloor$ by performing one multiplication. First, a right shift of $2s - 2$ bits is applied to t' to obtain $\left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor$, then a $(s + 3) \times (s + 4)$ -bit size multiplication is performed to compute $\left\lfloor \frac{t'}{2^{2s-2}} \right\rfloor \mu$. Finally, the estimated quotient is obtained by applying a right shift of $s + 5$ bits to the above result and it will be a $s + 2$ -bit integer.

To calculate the result of the reduction operation, step 4 needs another multiplication and one subtraction. Since the modulus is a $2s$ -bit integer, the second term in this equation is computed by a multiplication of size $(s + 2) \times 2s$ -bit. And the reduction is completed by subtracting using the least significant $2s + 1$ bits of the two terms. Therefore, the result of the reduction operation is a $(2s + 1)$ -bit integer.

Additionally, Table 1 shows that our algorithm needs one additional subtraction to correct the final result. In this step, a subtraction by the modulus N is first performed, followed by a check to determine whether the result is positive or negative. If the result is positive, it is the final output. Otherwise, an additional addition with the modulus is required to ensure that the outcome of the modular reduction is greater than zero. To determine the sign of the result, we use the two's complement operation for the modulus in the subtraction. Since the maximum possible result of the reduction operation is $2s + 1$ bits, the two's complement representation of the modulus N requires $2s + 2$ bits. Additionally, the reduction result r_0 needs an extra most significant bit set to zero. These two terms in subtraction both use the most significant bit to determine the sign. Thus, in the result correction step, we need at most one $(2s + 2) - (2s + 2)$ -bit subtraction and one $(2s + 2) - (2s + 2)$ -bit addition.

We applied the same analytical method to evaluate the size of arithmetic operations in Algorithm 1 and Algorithm 2, and the results are summarized in Table 2. The general Barrett reduction uses the original $2n$ -bit integer for computation, resulting in the largest multiplication size. Meanwhile, the folding operation used in the last two reduction algorithms achieves a similar scale in terms of multiplication size. For the addition/subtraction, a single $3s$ -size addition is required due to the folding operation. The optimization of precomputed values leads to the best performance for the number of subtractions in the optimized folding Barrett reduction.

Algorithm	Precomputed parameters {size}	The times of final subtraction
General Barrett Reduction	$\mu = \left\lfloor \frac{2^{2n}}{N} \right\rfloor \{2s + 1\}$	2
Folding Barrett Reduction	$N' = 2^{3s} \bmod N \{2s\}$	3
	$\mu = \left\lfloor \frac{2^{3s}}{N} \right\rfloor \{s + 1\}$	
Optimized Folding Barrett Reduction	$N' = 2^{3s} \bmod N \{2s\}$	1
	$\mu = \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor \{s + 4\}$	

Table 1. Comparison of parameters and the times of final subtraction ($s = \frac{n}{2}$).

Therefore, considering the arithmetic operations, the folding Barrett reduction offers a slight advantage in terms of multiplication size, while the optimized folding Barrett reduction provides a significant advantage in subtraction.

Quantum implementation of optimized folding Barrett reduction

In Algorithm 3, we described the classical algorithm to perform the modular operation on a $2n$ -bit integer with a n -bit modulus. Our optimized quantum folding Barrett reduction primarily consists of sequential multiplication and addition/subtraction operations based on this classical algorithm.

Circuit overview

The overview of our quantum circuit is illustrated in Fig. 2. In the figure, the first three registers N , μ , and N' are classical. And the precomputation of these two constants $\mu = \left\lfloor \frac{2^{3s+3}}{N} \right\rfloor$ and $N' = 2^{3s} \bmod N$ can be implemented in the classical computer since the known modulus N . The register labeled $|g\rangle$ is used to store the results of each computation step, while the register labeled $|a\rangle$ contains the ancillary qubits used during the calculation, all initialized to the state $|0\rangle$. The whole circuit mainly consists of three stages as follows:

(1) Folding operation stage

At this stage, we need to fold the $2n$ -qubit integer $|t\rangle$ into a $(3s + 1)$ -qubit integer $|t'\rangle$, where $n = 2s$. The intermediate value is calculated via $|t'\rangle = |t\rangle \bmod 2^{3s} + \left\lfloor \frac{|t\rangle}{2^{3s}} \right\rfloor N'$. The first term can be seen as the least significant $3s$ qubits of $|t\rangle$. And the second term is computed by a quantum-classical multiplication of size $s(\text{qubit}) \times 2s(\text{bit})$, where $\left\lfloor \frac{|t\rangle}{2^{3s}} \right\rfloor$ represents a right shift operation by $3s$ qubits on quantum value $|t\rangle$, and N' is a precomputed classical constant of $2s$ bits. The result of this stage is computed via one $(3s + 3s)$ -qubit quantum addition, where both terms are all quantum values.

(2) Reduction stage

The folding operation efficiently reduces the size of multiplications in the Barrett reduction step. The remainder can be calculated by $|R\rangle = (|t'\rangle - \lfloor \frac{|t'\rangle}{2^{2s-2}} \rfloor \frac{\mu}{2^{s+5}} N) \bmod N$. First, a right shift of $2s - 2$ qubits is performed to $|t'\rangle$ to obtain $\lfloor \frac{|t'\rangle}{2^{2s-2}} \rfloor$, then a $(s + 3)(\text{qubit}) \times (s + 4)(\text{bit})$ size quantum-classical multiplication is performed to obtain $\lfloor \frac{|t'\rangle}{2^{2s-2}} \rfloor \mu$. This result is then right-shifted by $s + 5$ qubits and subsequently multiplied by the modulus N , yielding the second term using a $(s + 2)(\text{qubit}) \times 2s(\text{bit})$ size quantum-classical multiplication. Finally, the reduction is completed by subtracting using the least significant $2s + 1$ qubits of the two terms to obtain the result. In this stage, the division operation is implemented using two multiplications and one subtraction.

(3) Result correction stage

As described in ‘‘Optimized folding Barrett reduction’’, the final result requires at most one subtraction to correct the output of the reduction operation. In this stage, a subtraction by the modulus is first performed, followed by a check to determine the sign of the result. To determine the sign of the result, we use the two’s complement arithmetic on the modulus N . Since the possible result of the reduction is $2s + 1$ qubits, the two’s complement representation of the modulus N requires $2s + 2$ bits. The reduction result $|R\rangle$, however, needs an additional qubit set to $|0\rangle$ as the most significant qubit, which serves as its sign bit. As the modulus N is a classical value, the two’s complement operation can be performed on a classical computer. Then, a quantum-classical adder of size $(2s + 2)(\text{qubit}) + (2s + 2)(\text{bit})$ is used to add the reduction result to the two’s complement of the modulus, with the final result determined by measuring the most significant qubit to check whether the outcome is less than zero. If the most significant qubit of the result is $|1\rangle$, an inverse circuit of this quantum-classical adder of the same size is required to correct the final result.

Assuming the input of the circuit is an 8-qubit number $|t\rangle$, then $n = \frac{8}{2} = 4$ and $s = \frac{n}{2} = 2$. According to the stage (1), after a $2(\text{qubit}) \times 4(\text{bit})$ quantum-classical multiplication and a $(6 + 6)$ -qubit quantum addition, the result of the folding operation is a 7-qubit number $|t'\rangle$. Then, the reduction is performed through a

Algorithm	Multiplication {times[bits]}	Addition/Subtraction {times[bits]}
General Barrett Reduction	$1[(2s + 1) \times (2s + 1)]$	$1[(2s + 1) - (2s + 1)]$
	$1[(2s + 1) \times 2s]$	$1[(2s + 2) - (2s + 2)]$
		$1[(2s + 2) + (2s + 2)]$
Folding Barrett Reduction	$1[s \times 2s]$	$1[3s + 3s]$
	$1[(s + 1) \times (s + 1)]$	$1[(2s + 1) - (2s + 1)]$
	$1[(s + 2) \times 2s]$	$3[(2s + 2) - (2s + 2)]$
		$1[(2s + 2) + (2s + 2)]$
Optimized Folding Barrett Reduction	$1[s \times 2s]$	$1[3s + 3s]$
	$1[(s + 3) \times (s + 4)]$	$1[(2s + 1) - (2s + 1)]$
	$1[(s + 2) \times 2s]$	$3[(2s + 2) - (2s + 2)]$
		$1[(2s + 2) + (2s + 2)]$

Table 2. Comparison of computational complexity ($s = \frac{n}{2}$).

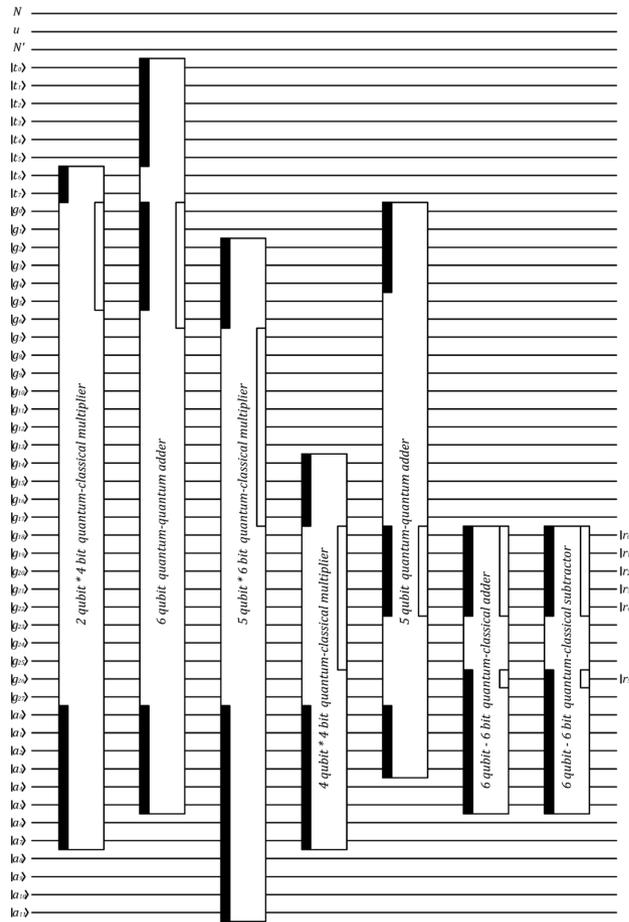


Fig. 2. Quantum optimized folding Barrett reduction ($n = 4$).

$5(\text{qubit}) \times 6(\text{bit})$ multiplication, a $4(\text{qubit}) \times 4(\text{bit})$ multiplication, and a $(5 + 5)$ -qubit addition, as detailed in stage (2). Following that, as described in stage (3), one more $6(\text{qubit}) + 6(\text{bit})$ quantum-classical addition and $6(\text{qubit}) - 6(\text{bit})$ subtraction are needed to complete the final result correction.

Quantum circuit module

(1) Quantum-quantum adder

For the adder that takes quantum values as input, we use an adder based on Gidney’s design, which effectively reduces quantum resources on T-count and T-depth by using the logical-AND gate¹². The construction of this gate is shown in Figure 3.

By applying pairs of computation and uncomputation logical-AND gates, the AND operation of two qubits can be performed and stored on an ancilla qubit using four T gates. Subsequently, the result on the ancilla qubit can be cleared using zero additional T gate. The paired use of the logical-AND gates reduces the T-depth to just one.

Gidney’s adder did not account for the carry on the most significant bit. We modified this circuit to handle the carry for the most significant bit. Figure 4 shows a 5 bits modified adder.

The first four pairs of logical-AND gates perform the carry generation and clearance during the addition of the first four bits. To optimize quantum resource, we utilize the quantum AND gate¹⁵ to compute the carry for the most significant bit instead of the Toffoli gate. In Fig. 5, the quantum AND gate has four T gates with a T-depth of one but requires one ancilla qubit, which can be used in future additions in the multiplier circuit.

(2) Quantum-quantum subtractor

It is convenient to directly use our modified adder to compute $a - b$. According to the definition of two’s complement arithmetic, the bitwise complement of a is equal to $-a - 1$. Thus, by performing a bitwise complement operation on both input a and the output of the adder, subtraction can be effectively implemented using this adder. The computation process is as follows:

$$(a' + b)' = (-a - 1 + b)' = a - b$$

In a quantum circuit, the two’s complement operation can be implemented by applying NOT gates to both the input and output. Since the complement operation can be performed independently on each bit, the NOT gates

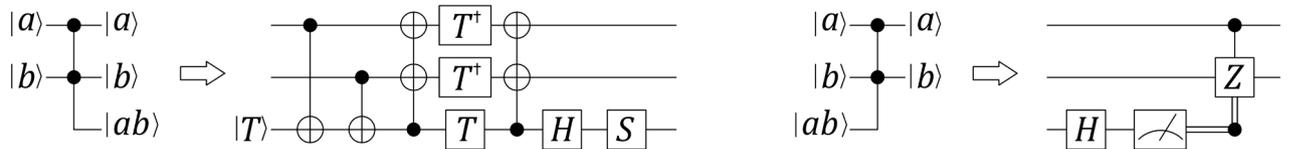


Fig. 3. Computation and uncomputation of the logical-AND gate.

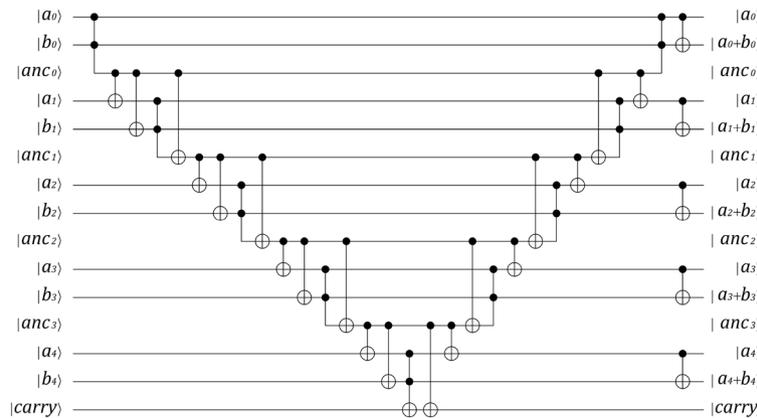


Fig. 4. In-place modified adder for 5 bits.

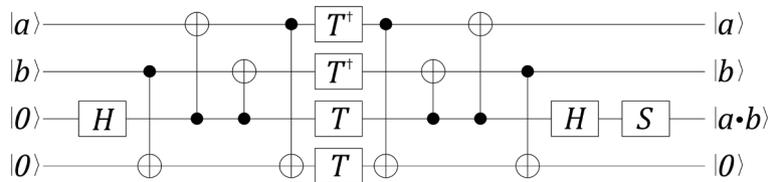


Fig. 5. Quantum AND gate.

can be executed in parallel, ensuring that the subtractor increases the circuit depth by two additional layers of NOT gates.

(3) Quantum-classical adder/subtractor

Currently, widely used quantum addition circuits, such as the quantum ripple-carry adder proposed by Cuccaro and the carry-lookahead adder proposed by Draper et al.¹³, are designed for quantum-quantum addition. When one of the inputs is classical value, appropriate modifications to these existing adder circuits are required. However, in these circuits, the carry can affect the values of the input bits during the computation, making it challenging to modify them when one input is classical.

Therefore, we chose the quantum ripple-carry adder proposed by Vedral et al.¹¹, as this adder does not allow the carry to affect the input bits during the computation. It also can be easily adapted for addition with a classical input by replacing certain Toffoli and CNOT gates with CNOT and NOT gates, as shown in Fig. 6.

In our modular reduction circuit, the quantum-classical adder/subtractor is used during the final result correction stage. By applying the two's complement operation to the classical value, subtraction can be directly implemented using the adder circuit. We employ a garbage qubit to measure the most significant bit of the result. If the measurement yields $|1\rangle$, the result is negative, indicating that the reduction result is less than the modulus. In this case, the inverse of the addition circuit is applied to recompute the correct result. If the measurement yields $|0\rangle$, the result is positive, and the output of the adder represents the final result of the reduction.

(4) Quantum-classical multiplier

The optimized quantum folding Barrett reduction requires a total of three multiplications of different sizes throughout the computation, each a quantum-classical multiplication. We use CNOT gates to set the partial products, with the number of CNOT gates determined by the number of ones in the binary expansion of the classical value.

For the partial product addition stage, we use the addition circuit described in Fig. 4. While reducing the number of T-gates used, we also minimize the number of quantum bits required by reusing the ancilla qubits within the adder circuit. As shown in Fig. 7, in addition to the partial product setting and addition stages, we

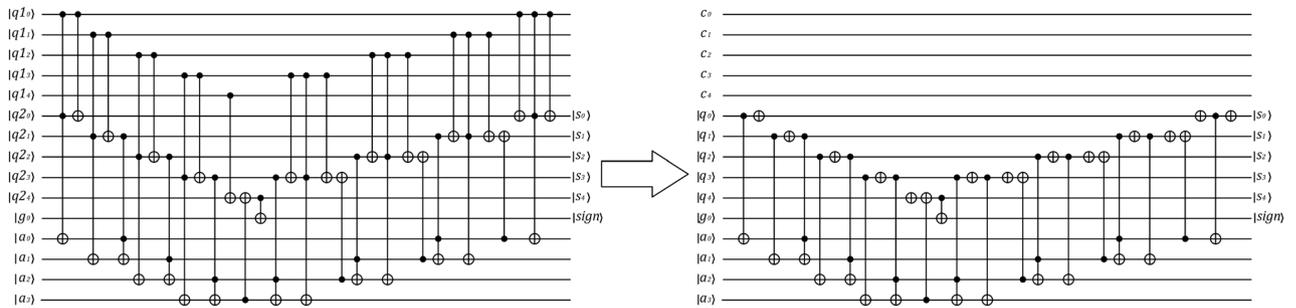


Fig. 6. Quantum-classical adder.

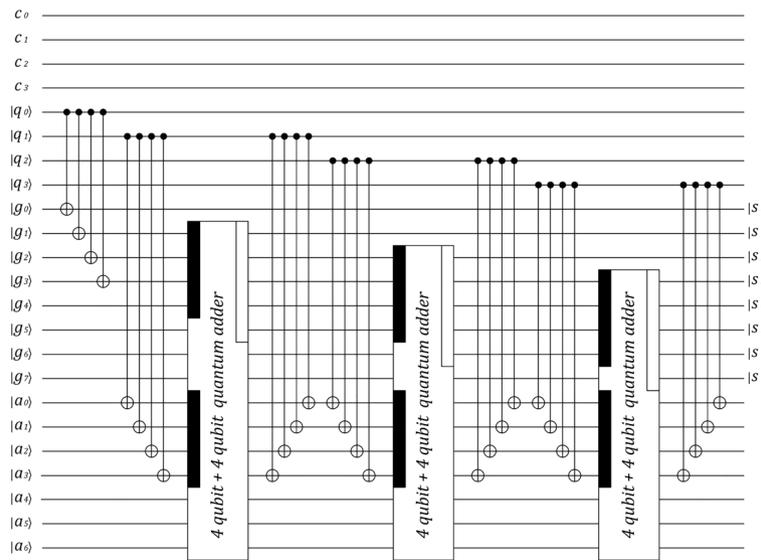


Fig. 7. Quantum-classical multiplier ($n = 4$).

also use the inverse partial product setting operation. Reinitializing the state of the ancilla qubits allows the continuation of the next partial product setting and addition operations.

Complexity analysis

In this section, we analyze the cost of our quantum reduction circuit. As described in “Quantum Barrett reduction”, our quantum circuit mainly consists of quantum–quantum adders, quantum-classical adders, and quantum-classical multipliers. Using the same circuit design principles, we have converted the General Barrett Reduction, the Folding Barrett Reduction, and our Optimized Folding Barrett Reduction into quantum circuits and estimated their quantum resource requirements. Since the implementation of the T gate is more costly than other Clifford gates, our analysis focuses primarily on the number of T gates and the T-depth.

We estimated quantum resources for different types of adders and subtractors, with the results summarized in Table 3.

The quantum resources required by the multiplier are determined by the size of the multiplication, specifically, the number of times the adder is used. A $a(\text{qubit}) \times b(\text{bit})$ quantum-classical multiplier, can be considered as requiring $a - 1$ adder operations to perform the multiplication. The size of an addition circuit is equal to the bit length of the classical value. Therefore, in the partial products addition stage, the quantum resources of the multiplier are determined by the product of the number of adders and the quantum resources of the adder. In the partial product setting stage, since one of the inputs is classical value, the setting operation is implemented utilizing CNOT gates, which do not affect the usage of T gates. Overall, the number of qubits required for the multiplier of the above size is $2a + 3b - 1$, the number of T gate is $(a - 1) \times 4b$ and the T-depth is $(a - 1) \times b$.

Based on the analysis above, we compared the quantum resources of the three Barrett reductions. The costs for each algorithm are listed in Table 4, and the comparison of T-count and T-depth is shown in Fig. 8. As shown below, our optimized folding Barrett reduction achieves the best performance in both T-count and T-depth.

To validate the performance of our algorithm in practical modular multiplication, we combined it with Karatsuba, Toom–Cook, and Schönhage–Strassen multiplication algorithms and compared it with the general quantum modular operation. In general quantum modular operation, to avoid the use of division operation, one

Arithmetic operation	#Qubit	T-count	T-depth
Quantum-quantum adder	$3n + 1$	$4n$	n
Quantum-quantum subtractor	$3n - 1$	$4n - 4$	$n - 1$
Quantum-classical adder/subtractor	$2n + 1$	$8n - 16$	$2n - 4$

Table 3. Quantum resources of arithmetic circuit.

Algorithm	#Qubit	#NOT	#CNOT	T-count	T-depth
General Barrett Reduction	$6n + 6$	$12n + 15 - 9y(N)$	$(2n + 1)w(\mu) + (2n + 1)w(N) + 12n^2 + 9n + 8 - 6y(N)$	$8n^2 + 32n$	$2n^2 + 8n$
Folding Barrett Reduction	$6n + 9$	$15n + 19 - 12y(N)$	$(n - 1)w(N') + (n + \frac{1}{2})w(\mu) + (n + 3)w(N) + \frac{15}{2}n^2 + 21n + 6 - 8y(N)$	$5n^2 + 44n$	$\frac{5}{4}n^2 + 11n$
Optimized Folding Barrett Reduction	$6n + 12$	$9n + 11 - 6y(N)$	$(n - 1)w(N') + (n + 5)w(\mu) + (n + 3)w(N) + \frac{15}{2}n^2 + 30n + 34 - 4y(N)$	$5n^2 + 38n + 32$	$\frac{5}{4}n^2 + \frac{19}{2}n + 8$

Table 4. Comparison of quantum resources. Let $w(n)$ denote the numbers of ones in the binary expansion of n and $y(n)$ denote the numbers of zeros in the binary expansion of n

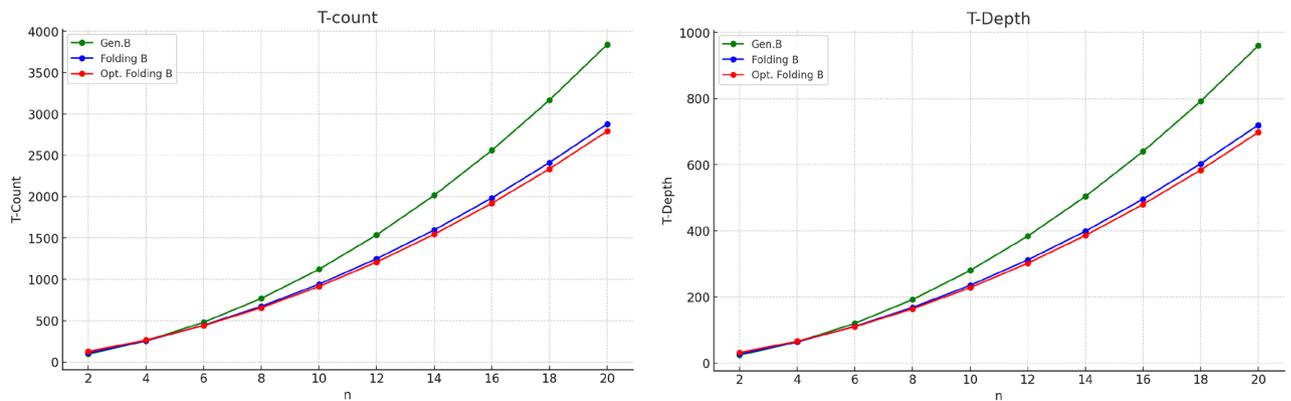


Fig. 8. Comparison of T-count and T-depth of quantum Barrett reduction.

can recursively use subtraction and controlled addition to perform modular reduction. However, this approach suffers from a significant disadvantage in terms of circuit depth, the complexity of T-depth is approximately $O(2^n)$. Figure 9 illustrates the comparisons of T-depth between our algorithm and the general modular operation. In Fig. 9, “SS_Mod”, “KS_Mod”, and “TC_Mod” denote modular multiplications that pair the general quantum modular operation with the Schönhage-Strassen, Karatsuba, and Toom–Cook multiplication algorithms, respectively. Conversely, “SS_Bar”, “KS_Bar”, and “TC_Bar” refer to modular multiplications that combine our optimized Barrett reduction scheme with the same three multiplication algorithms. The complexity of modular multiplication is primarily determined by the chosen modular reduction algorithm. Our algorithm shows a substantial advantage in T-depth over the general modular operation since the T-depth of our scheme is $O(n^2)$.

Conclusion

In this paper, we proposed an optimized folding Barrett reduction algorithm that ensures the feasibility of integration with existing multiplication algorithms while requiring only one additional subtraction for result correction. Furthermore, we converted three versions of Barrett reduction into quantum circuits and conducted a quantum resource analysis. By comparing the T-count and T-depth, our optimized folding Barrett reduction demonstrates the best performance. In addition, we compared our design with the general quantum modular reduction, and our approach shows a significant advantage in terms of T-depth. Tomčala¹⁶ demonstrated that modular exponentiation $a^x \bmod N$ can be calculated as a serial sequence of modular multiplications. When the circuit is constructed to the specific values of a and N rather than built as a universal design, the quantum resource cost of Shor’s algorithm drops significantly. Because the same modulus N is reused throughout the sequence, our Barrett reduction scheme offers a clear advantage for large integer factorization.

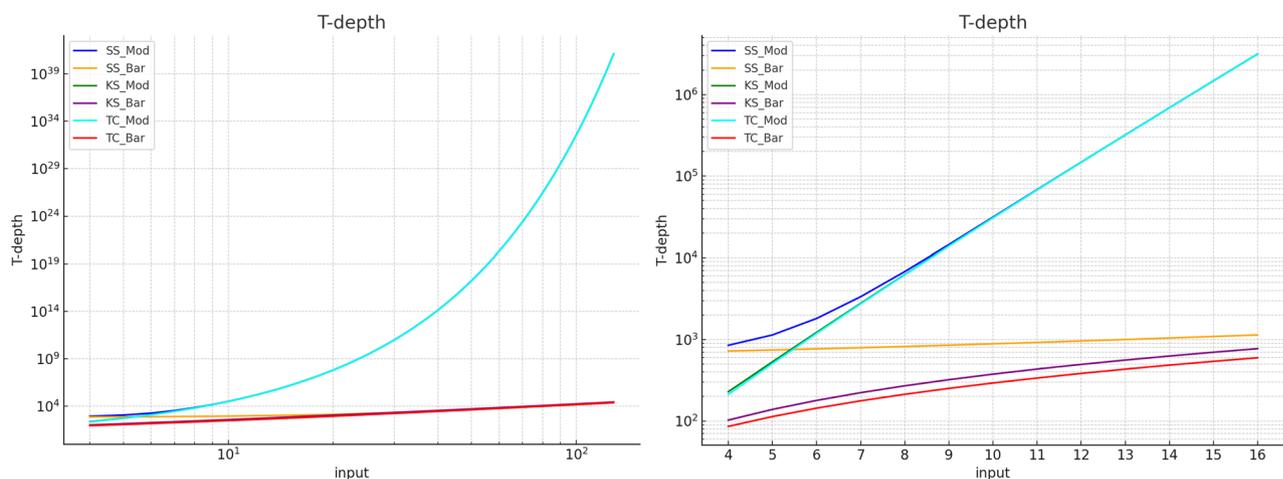


Fig. 9. Comparison of T-depth for various of modular multiplication.

Data availability

All data generated or analyzed during this study are included in this published article.

Received: 2 December 2024; Accepted: 30 May 2025

Published online: 02 July 2025

References

- Shor, P. W. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 124–134 (IEEE, 1994).
- Grover, L. K. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. 212–219 (1996).
- Yuan, S. et al. A novel fault-tolerant quantum divider and its simulation. *Quantum Inf. Process.* **21**, 182 (2022).
- Montgomery, P. L. Modular multiplication without trial division. *Math. Comput.* **44**, 519–521 (1985).
- Barrett, P. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In *Conference on the Theory and Application of Cryptographic Techniques*. 311–323 (Springer, 1986).
- Kong, Y. Optimizing the improved Barrett modular multipliers for public-key cryptography. In *2010 International Conference on Computational Intelligence and Software Engineering*. 1–4 (IEEE, 2010).
- Hasenplaugh, W., Gaubatz, G. & Gopal, V. Fast modular reduction. In *18th IEEE Symposium on Computer Arithmetic (ARITH'07)*. 225–229 (IEEE, 2007).
- Wu, T., Li, S.-G. & Liu, L.-T. Modular multiplier by folding Barrett modular reduction. In *2012 IEEE 11th International Conference on Solid-State and Integrated Circuit Technology*. 1–3 (IEEE, 2012).
- Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information* (Cambridge University Press, 2010).
- Rines, R. & Chuang, I. High performance quantum modular multipliers. arXiv preprint [arXiv:1801.01081](https://arxiv.org/abs/1801.01081) (2018).
- Vedral, V., Barenco, A. & Ekert, A. Quantum networks for elementary arithmetic operations. *Phys. Rev. A* **54**, 147 (1996).
- Gidney, C. Halving the cost of quantum addition. *Quantum* **2**, 74 (2018).
- Draper, T. G., Kutin, S. A., Rains, E. M. & Svore, K. M. A logarithmic-depth quantum carry-lookahead adder. arXiv preprint [quant-ph/0406142](https://arxiv.org/abs/quant-ph/0406142) (2004).
- Takahashi, Y. & Kunihiro, N. A fast quantum circuit for addition with few qubits. *Quantum Inf. Comput.* **8**, 636–649 (2008).
- Jaques, S., Naehrig, M., Roetteler, M. & Virdia, F. Implementing Grover oracles for quantum key search on AES and LOWMC. In *Advances in Cryptology—EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part II* **30**. 280–310 (Springer, 2020).
- Tomčala, J. On the various ways of quantum implementation of the modular exponentiation function for Shor's factorization. *Int. J. Theor. Phys.* **63**, 14 (2024).

Acknowledgements

This work was supported by Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (Ministry of Science and ICT(MSIT)) ((Q Crypton), No.2019-0-00033, Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity), by the Creation of the Quantum Information Science R&D Ecosystem (Grant No. 2022M3H3A106307411) through the National Research Foundation of Korea (NRF) funded by the Korean government (Ministry of Science and ICT), and by Quantum Computing based on Quantum Advantage challenge research through the National Research Foundation of Korea (NRF) funded by the Korean government (MSIT) (RS-2023-00256221).

Author contributions

J.Z. developed the optimized folding Barrett reduction and wrote the preliminary version of the manuscript, C.Y.L. provided advice on quantum circuit conversion, while S.M.C and S.H.S. edited it. All authors participated in analyzing the quantum resources of the circuits and reviewed the manuscript.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to S.-H.S.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025