Quantum Information Processing (2025) 24:35
https://doi.org/10.1007/s11128-025-04647-3

Quantum circuit designs of efficient squaring

Seong-Min Cho[1] · Changyeol Lee[1] · Seung-Hyun Seo[2]

Received: 8 April 2024 / Accepted: 4 January 2025

Abstract
Quantum squaring circuits have been used as helpful arithmetic modules in various quantum algorithms for calculating series expansions or distances of vectors, etc. Quantum multipliers can replace quantum squaring circuits, but squaring with quantum multipliers is inefficient because it involves using quantum gates for unnecessary bitwise multiplication. In this paper, we propose a depth-optimized quantum circuit dedicated to squaring by eliminating these unnecessary quantum gates and implementing quantum gates in parallel. We also discuss the optimal distribution of the partial products to reduce further the gate cost of the quantum adder used for the sum of the partial products. The proposed partial product distribution method lowers the quantum adder's number of gates and depth by half. Our quantum squaring circuit is the most efficient, with an average improvement of 68% and 79.7% in T-count and T-depth, respectively, compared to existing quantum squaring circuits. Despite the increased qubit counts caused by the depth optimization, we demonstrate that the proposed circuit has the smallest KQ_T.

Keywords Quantum circuits · Squaring · T-depth optimization

1 Introduction

Quantum squaring circuits function as a fundamental arithmetic circuit, as demonstrated by their use in computing Euclidean distance on quantum circuits. A quantum circuit for calculating integer Euclidean distance is a critical component in the devel-

✉ Seung-Hyun Seo
seosh77@hanyang.ac.kr

Seong-Min Cho
smcho3315@hanyang.ac.kr

Changyeol Lee
sshdjjim@hanyang.ac.kr

1 Department of Electrical and Electronic Engineering, Graduate School of Hanyang University, Seongdong-gu, Seoul 04763, South Korea

2 School of Electrical Engineering, Hanyang University ERICA, Ansan 15588, Gyeonggi-do, South Korea

Published online: 24 January 2025

opment of quantum circuits and quantum cryptanalysis, particularly for cryptography operating on finite fields. Quantum squaring circuits also facilitate the acceleration of quantum exponentiation operations.

Generally, in quantum circuits, we can implement the square operation simply by putting the same number twice into the quantum multiplier's input. So then, why do we need a dedicated quantum squaring circuit rather than a simple quantum multiplier?

The quantum multiplier computes n^2 partial products by performing bitwise multiplications for two inputs of n qubits. However, these partial products include unnecessary and redundant things in a quantum squaring circuit, so squaring with a quantum multiplier costs additional quantum gates. To be more specific, for two inputs x_1x_0 and y_1y_0 represented in binary, the quantum multiplier must perform four bitwise multiplications to compute the partial products x_0y_0, x_0y_1, x_1y_0, and x_1y_1. In squaring, the two inputs are equal, so the partial products that we need to find for the input x_1x_0 are x_0x_0, x_0x_1, x_1x_0, and x_1x_1. Based on the bitwise multiplication properties, we know that $x_ix_i = x_i$ and $x_ix_j = x_jx_i$. Therefore, x_0x_0 and x_1x_1 are unnecessary partial products in a squaring process because they can be replaced by x_0 and x_1, respectively. In addition, x_0x_1 and x_1x_0 are redundant partial products because they are equal, so a single bitwise multiplication can compute both partial products. As a result, squaring requires only one bitwise multiplication to get the partial product x_0x_1. Utilizing these properties of squaring results in only $n(n-1)/2$ bitwise multiplications, while using the same process as a quantum multiplier for a squaring would result in n^2 bitwise multiplications. Notably, bitwise multiplication is typically implemented using Toffoli gates, which have a much larger overhead compared to other gates, in quantum circuits. So, the use of quantum multipliers for squaring leads to inefficiencies due to the increased quantum resources of quantum algorithms that require quantum squaring circuits. Therefore, it is essential to optimize the quantum resources by designing the quantum circuit to consider the properties of squaring.

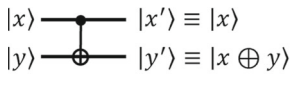
Since H. V. Jayashree et al. firstly proposed a dedicated quantum squaring circuit in 2014, several researchers have conducted research on the optimized design of quantum squaring circuits. H. V. Jayashree et al. implemented a quantum squaring circuit using fewer quantum resources than a quantum multiplier by eliminating quantum gates to compute redundant and unnecessary partial products [1]. Their quantum squaring circuit uses 63–85% fewer qubits and gates than existing quantum multipliers. Until now, optimization for quantum squaring circuits has been studied in the direction of reducing the number of ancilla or garbage qubits. In [2] and [3], A. Banerjee et al. proposed quantum squaring circuits by dividing the operands into bits and recursively squaring them. Their recursive method effectively reduced the number of qubits, but at the cost of a significant increase in depth. They also designed a quantum adder that requires only one ancilla qubit and no garbage qubits and used it to sum bitwise multiplications, reducing the number of ancilla and garbage qubits by 10–20 compared to [1]. Seeing this drop in qubits, Jayashree et al. designed a quantum squaring circuit that uses at least fifty percent fewer qubits than previous works when the input size exceeds eight [4]. They utilized Karatsuba's recursive method, which computes multiplication by dividing the input in half, and designed an inverse computation unit to implement a quantum squaring circuit with zero garbage qubits. While previous proposals for quantum squaring circuits have focused on reducing the number of qubits in this way,

A. N. Nagamani et al. proposed a combined approach that reduces the number of gates and qubits in a quantum squaring circuit. They designed a quantum squaring circuit to compute squares based on long multiplication and the Baugh-Wooley algorithm [5] and eliminated garbage qubits by using an inverse circuit to regenerate the inputs at the output. They also proposed a garbage-cost optimized version, designed to be reusable by initializing only ancilla qubits instead of inputs. Their optimized version uses some additional garbage qubits compared to the garbage-free version, but uses 27.3% fewer gates as a trade-off.

However, with the increase in available qubits due to the recent developments in large-scale quantum computers, the optimization of quantum circuits is shifting toward reducing depth, which is closely related to the execution time of quantum circuits, instead of qubits. In December 2023, IBM unveiled Condor, a quantum chip with 1121 qubits [6]. The increase in available qubits is loosening the constraints on the number of qubits in quantum circuit design. So, cutting down on the number of gates and depth is important for reducing the execution time of quantum circuits. Recent studies proposing quantum arithmetic circuits, such as quantum adders and multipliers, also aim to reduce the depth of the quantum circuits [7, 8]. In particular, T-gates have a higher implementation overhead than other quantum gates due to additional physical resources and complex control sequences when implemented on quantum computers [9]. Therefore, designing a quantum circuit with low T-gate costs (T-count and T-depth) further reduces the execution time of quantum circuits. Moreover, we can implement fault-tolerant quantum circuits by increasing the tolerance to noise errors. However, these fault-tolerant circuits further increases the cost overhead of implementing T-gates. Thus, designing quantum squaring circuits with low T-gate costs is essential for increasing both the quantum circuit's efficiency and fault tolerance. Most recently, in 2024, A. Sultana and E. Muñoz-Coreas proposed a gate resource-efficient quantum squaring circuit [10]. They optimized the number and depth of quantum gates by halving the number of quantum adders used. They reduced the T-count by 66–77% and the T-depth by 50–68% compared to the designs in [1] and [11]. From the perspective of implementation, we should more efficiently design quantum circuits by optimizing T-count and T-depth while minimizing the increase in qubit count. The trade-off between qubit count and gate cost can be evaluated by a metric called KQ_T.

In this paper, we present efficient quantum squaring circuit designs in terms of T-count and T-depth by utilizing the properties of the bitwise squaring operation. Leveraging the properties of bitwise squaring allows carry to be precomputed and reduces the number of partial products. For additional quantum resource reduction, we also propose a parallel implementation of the partial product setup and a distribution method to minimize the size of quantum adders used. Our circuit uses additional ancilla qubits to copy the state of the operand qubits and then computes all partial products with a depth of 1. Our distribution method also enables us to compute the sum of the partial products using quantum adders with half the input size by properly rearranging the partial products going into the input of the quantum adders. This allows us to use a few more qubits, but with a reduced number and depth of T-gates. Our design achieves a T-count and T-depth reduction of up to 20% and 60%, respectively, compared to most efficient existing quantum squaring circuits, while also consuming the least KQ_T.

Fig. 1 The Feynman gate



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

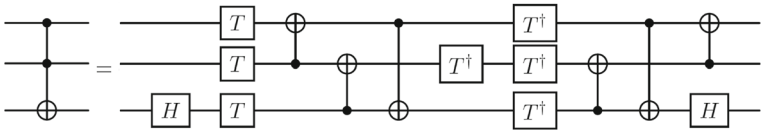


Fig. 2 The Toffoli gate with T-count 7 and T-depth 3

The rest of this paper is organized as follows. We will introduce the preliminaries in Sect. 2 and the existing works proposing the quantum squaring circuit designs in Sect. 3. In Sect. 4, we describe the proposed quantum squaring circuit design and discuss the quantum resources analysis. Finally, we include a short conclusion in Sect. 5.

2 Preliminaries

In this section, we introduce the elementary quantum gates used in our quantum squaring circuit and the performance evaluation metrics for quantum circuits.

2.1 Elementary quantum gates

Feynman gate
The Feynman gate, also called the CNOT (Controlled-NOT), is a 2×2 reversible gate with two input qubits and two output qubits. When two qubit states $|x\rangle$ and $|y\rangle$ are inputs to the Feynman gate, the states of output qubits are $|x\rangle$ for the input $|x\rangle$, and $|y \oplus x\rangle$ for the input $|y\rangle$. Feynman gates are often used to copy the qubit state in quantum circuit design. Taking the Feynman gate for $|y\rangle$ initialized to $|0\rangle$, the state of $|x\rangle$ is copied to $|y\rangle$. We utilize Feynman gates to copy qubits storing the state of an operand (Fig. 1).

Toffoli gate
The Toffoli gate, also called the CCNOT (controlled-controlled-NOT) gate, is a 3×3 reversible gate with three input qubits and three output qubits. When three qubit states $|x\rangle$, $|y\rangle$, and $|z\rangle$ are inputs to the Toffoli gate, the states of output qubits are $|x\rangle$ and $|y\rangle$ for the input $|x\rangle$ and $|y\rangle$, and $|z \oplus xy\rangle$ for the input $|z\rangle$. Quantum arithmetic circuits commonly utilize Toffoli gates to implement bitwise multiplications. Toffoli gates are implemented in quantum computing as a composition of three T gates, four T-dagger gates, two Hadamard gates, and six Feynman gates (Fig. 2).

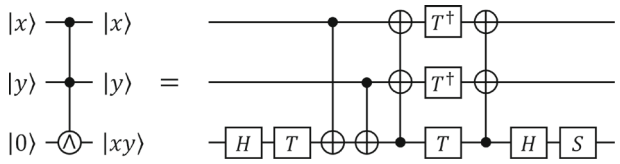


Fig. 3 The quantum logical-AND gate with T-count 4 and T-depth 2

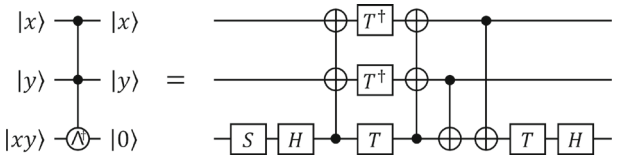
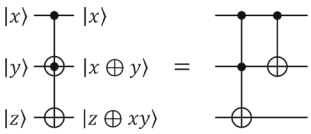


Fig. 4 The dagger of quantum logical-AND gate with T-count 4 and T-depth 2

Fig. 5 The quantum Peres gate



Quantum logical-AND gate

The quantum logical-AND (Q-AND) gate presented in [12] computes bitwise multiplication in quantum circuits, similar to the Toffoli gate. The Q-AND gate is more resource efficient than the Toffoli gate, as it employs ancillary qubits initialized to the $|T\rangle$ state, resulting in a T-count of 3 (one T-gate and two T-dagger gates) and a T-depth of 1. The dagger operation of the Q-AND gate in [12] is executed by measuring the target qubit, so it does not consume T-count and T-depth. However, it has the drawback that the target qubit cannot be subsequently reused. The non-reusability of qubits necessitates the allocation of new ancillary qubits with each utilization of the Q-AND gate. This leads to excessive qubit consumption in quantum arithmetic circuits utilizing Q-AND gates as components. In our circuit, we implement a Q-AND gate by utilizing a Hadamard gate followed by a T-gate applied to the target qubit instead of using $|T\rangle$ state, and we construct a dagger of the Q-AND gate by reversing the sequence of the gates used in the Q-AND configuration. Our Q-AND gate and its dagger have a T-count of 4 and a T-depth of 2 as shown in Figs. 3 and 4.

Quantum Peres gate and double Peres gate

The quantum Peres gate shown in Fig. 5 is a 3×3 reversible gate with three input qubits and three output qubits. When three qubit states $|x\rangle$, $|y\rangle$, and $|z\rangle$ are inputs to the quantum Peres gate, the states of output qubits are $|x\rangle$ and $|x \oplus y\rangle$ for the input $|x\rangle$ and $|y\rangle$, and $|z \oplus xy\rangle$ for the input $|z\rangle$.

The quantum double Peres gate shown in Fig. 6 is a 4×4 reversible gate with four input qubits and four output qubits. When four qubit states $|w\rangle$, $|x\rangle$, $|y\rangle$, and $|z\rangle$ are inputs to the quantum double Peres gate, the states of output qubits are $|w\rangle$, $|w \oplus x\rangle$, $|w \oplus x \oplus z\rangle$, and $|(w \oplus x)z \oplus wx \oplus y\rangle$ for the input $|w\rangle$, $|x\rangle$, $|y\rangle$, and $|z\rangle$.

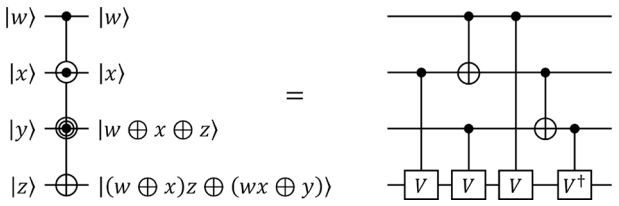


Fig. 6 The quantum double Peres gate

2.2 Performance evaluation metrics for quantum circuits

T-count and T-depth

Quantum circuits are prone to noise errors. Error resistance in quantum circuits can be achieved through the use of fault-tolerant gates and error-correcting codes. Clifford+T-gates are fault tolerant with error-correcting codes. In quantum error correction schemes, T-gates, which are non-Clifford gates, require many operators and qubit resources to be fault tolerant, while Clifford gates require few resources to be fault tolerant. As a result, T-count and T-depth are considered important metrics for measuring the efficiency of quantum circuits, and reducing T-count and T-depth is essential for efficient quantum circuit design. T-count refers to the total number of T-gates used in a quantum circuit, while T-depth refers to the number of T-gate layers.

KQ_T

The concept of quantum volume, first introduced by IBM [13], is a comprehensive metric for evaluating the performance of quantum computers. The KQ_T, a concept similar to quantum volume, is measured by multiplying the number of qubits by the depth of the T-gate used in a quantum circuit. This metric is useful for evaluating the connectivity of qubits as the gate depth increases and is often used to evaluate the trade-off of increasing the number of qubits as the depth decreases. To design efficient quantum circuits, it is important to reduce the quantum circuit's gate depth, but it is also necessary to keep the number of qubits with decreasing depth low, thus lowering the KQ_T.

3 Related works

There have been several studies on designing quantum circuits for square operations [1–4, 10, 11]. They aim to utilize the properties of square operations to design efficient quantum circuits. In 2014, H. V. Jayashree et al. were the first to propose a quantum circuit for efficiently computing squares in [1]. They eliminated the quantum gates that compute the redundant and unnecessary partial products in the squaring operation. This method leads to the use of fewer quantum resources in their quantum circuits compared to multiplication circuits. The result of squaring is obtained by computing the sum of the partial products using Peres gates and double Peres gates in their implementation. Their quantum squaring circuit demonstrated a 63% to 85% improvement in terms of the number of qubits and gates compared to existing quantum multiplication circuits.

Following the proposal of the quantum squaring circuit by Jayashree et al., subsequent studies have focused on reducing the number of garbage qubits. In 2014, Arindam Banerjee and Debesh Kumar Das [2] proposed designs leveraging recursive structure to reduce the usage of garbage and ancilla qubits in quantum squaring circuits relative to the circuit proposed in [1]. They further optimized the qubit usage by substituting the double Peres gates with the quantum full adder proposed in [14]. In 2016, they also proposed a quantum adder requiring only a single ancillary input qubit without any other garbage qubits in [3]. They utilized this adder along with an $(n-1)$-bit quantum squarer to recursively design an n-bit quantum squarer. Along with this approach, they further proposed a quantum square circuit with zero garbage and fewer ancilla qubits by designing the input to be reusable at the output. They reduced the number of qubits used in quantum squaring circuits by employing a recursive structure in [2] and [3], but their circuits have an increased number of gates compared to [1]. In 2017, Jayashree et al. proposed a quantum circuit that applies Karatsuba's recursive method [15] to recursively compute the square by dividing the input in half [4]. They also improved qubit usage by designing inverse computation units to retrieve inputs and achieve a quantum circuit without garbage. Their quantum squaring circuit uses fewer qubits than previous studies [1–3] when the input size exceeds eight, while using more quantum gates.

In 2018, A. N. Nagamani et al. [11] proposed garbage-free and garbage-cost optimized quantum squaring circuits using long multiplication and the Baugh-Wooley multiplication algorithm [5]. They explored an optimal trade-off point between garbage qubits and quantum cost to design a garbage-cost optimized quantum circuit, achieving a 27.3% improvement in gate count using slightly additional garbage qubits compared to their garbage-free circuit.

Most recently, in 2024, A. Sultana and E. Muñoz-Coreas introduced a quantum squaring circuit that used fewer quantum resources than [1] and [11]. By adding the rearranging of partial products, they reduced the number of quantum adders used for quantum squaring circuits by half compared to [1] and [11]. Their quantum squaring circuit shows a 66.67% reduction in T-count and a 50% reduction in T-depth than the design of [1], and a 77.27% reduction in T-count and a 68.75% reduction in T-depth than the design of [11].

Studies on quantum squaring circuits have mostly been aimed at minimizing the qubit count since Jayarashee et al. introduced the concept. They have proposed effective ways to reduce the number of qubits in quantum squaring circuits, but as a trade-off, the gate count and the depth of the circuits have significantly increased. Recent studies on quantum circuit design have focused on reducing the number of gates and circuit depth, taking advantage of the growing availability of qubits made possible by advancements in quantum computing. Therefore, it is necessary to design depth-efficient quantum squaring circuits.

				x_3	x_2	x_1	x_0
				x_3	x_2	x_1	x_0
				x_0x_3	x_0x_2	x_0x_1	x_0x_0
			x_1x_3	x_1x_2	x_1x_1	x_1x_0	
		x_2x_3	x_2x_2	x_2x_1	x_2x_0		
	x_3x_3	x_3x_2	x_3x_1	x_3x_0			
s_7	s_6	s_5	s_4	s_3	s_2	s_1	s_0

Fig. 7 The bitwise squaring operation on classical computer ($n = 4$)

4 Design of proposed quantum squaring circuit

In this section, we propose an efficient quantum circuit design for squaring with a lower T-count and T-depth compared to the previous studies [1–4, 11] by employing parallelization and reducing the input sizes of quantum adders used.

4.1 Optimized quantum squaring circuit design

We first capitalize bitwise multiplication and carry propagation properties in square operations to implement the square operation efficiently in a quantum circuit. Figure 7 shows the process of bitwise squaring for some size $n = 4$.

The square operation possesses distinct properties compared to general multiplication because it involves multiplying two identical numbers as follows:

- **Multiplications between identical bits** As shown in Fig. 7, n^2 partial products x_ix_i $(0 \leq i < n)$ always occur in the squaring process. The squaring operation can omit n bitwise multiplications compared to general multiplication due to the property of bitwise multiplication, where $x_ix_i = x_i$.
- **The repetition of the same partial products** Since $x_ix_j = x_jx_i$, the same partial products are repeated twice within the squaring operation, as $x_ix_j + x_ix_j = 2x_ix_j$. This implies that the partial product itself determines the occurrence of a carry. Specifically, if $x_ix_j = 0$, then $2x_ix_j = 00_2$, leading to no carry, whereas if $x_ix_j = 1$, then $2x_ix_j = 10_2$, resulting in a carry.

$$x_ix_j + x_jx_i = (x_ix_j)0_2 \tag{1}$$

The above properties of squaring lead to the transformation of the squaring result, as depicted in Fig. 8. This methodology reduces the number of bitwise multiplications from n^2 to $n(n-1)/2$. We propose an appropriate distribution technique for partial products based on these fundamental properties, achieving lower T-count and T-depth. Our quantum squaring circuit consists of a partial product setup unit, a partial product addition unit, and an initialization unit.

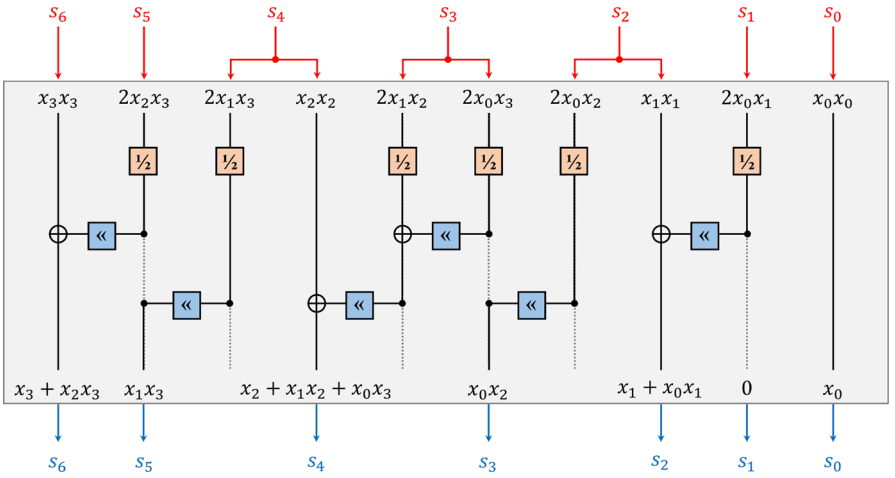


Fig. 8 The transformation of squaring result ($n = 4$)

A. Partial product setup unit

The partial product setup unit (PPSu) computes the reduced partial products as shown at the bottom of Fig. 8. Toffoli gates are commonly employed in the implementation of bitwise multiplication for computing partial products within quantum circuits. However, the Toffoli gate, with a T-count of 7 and a T-depth of 3, is a resource-intensive gate for implementation. We utilize a Q-AND gate for bitwise multiplication to reduce our circuit's overall T-count and T-depth. The Q-AND gate possesses a T-count of 4 and a T-depth of 2.

First, as shown in Fig. 8, the $2i$-th bit s_{2i} of the squaring includes the input bit x_i. CNOT gates directly add qubits $|x_i\rangle$ to $|s_{2i}\rangle$ individually. Then, PPSu copies the input qubits $|x_i\rangle$ to ancilla qubits $|anc_i\rangle$ to minimize the depth of the quantum circuit required to compute the remaining $n(n-1)/2$ partial products. The CNOT gates used for qubit copying do not contribute to the T-depth, allowing us to design a reduced T-depth instead of adding more CNOT gates. Subsequently, PPSu uses Q-AND gates with ancilla qubits $|anc_i\rangle$ and input qubits $|x_i\rangle$ to compute the remaining partial products with a T-depth of two. Figure 9 shows the PPSu of our quantum squaring circuit.

B. Partial product addition unit

We need to add the partial products to calculate the squared result s. The partial product addition unit (PPAu) performs additions on the partial products set by the PPSu using quantum full adders. A general quantum squaring circuit with an input size of n utilizes $\lfloor n/2 \rfloor$ quantum full adders, each having $2n-1$ input qubits, as shown in the left of Fig. 10. The quantum full adder used in existing quantum squaring circuits is an in-place adder that adds two quantum registers of size $2n-1$, depicted by the black and gray rectangles in Fig. 10, and stores the result of the addition in one of the input registers.

Quantum full adders consume Toffoli gates proportional to the input size, resulting in significant T-count and T-depth when the input size is large. A quantum full adder with an input size of n has a T-count of $4n-4$ and a T-depth of $2n-2$. We design the

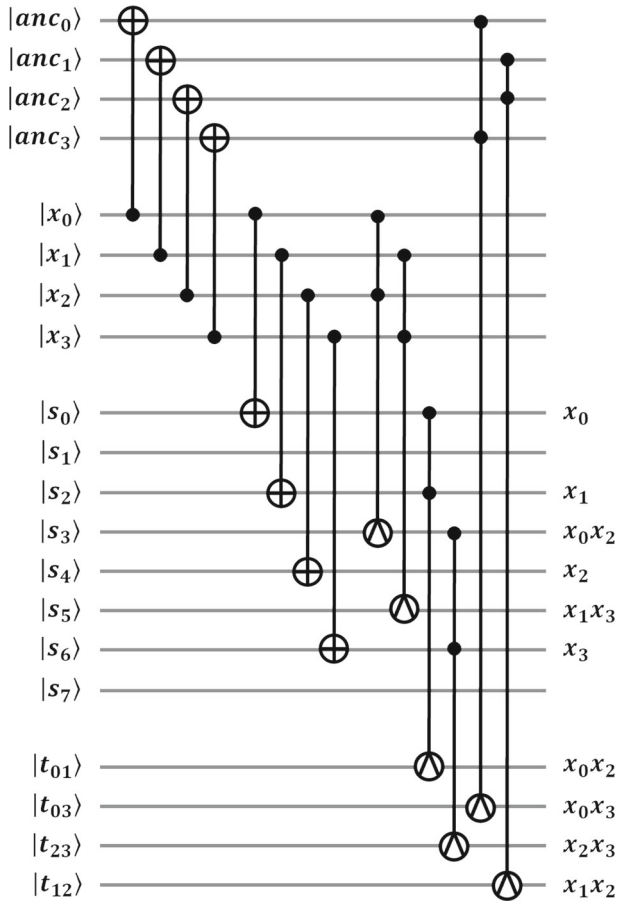


Fig. 9 The partial product setup unit ($n = 4$)

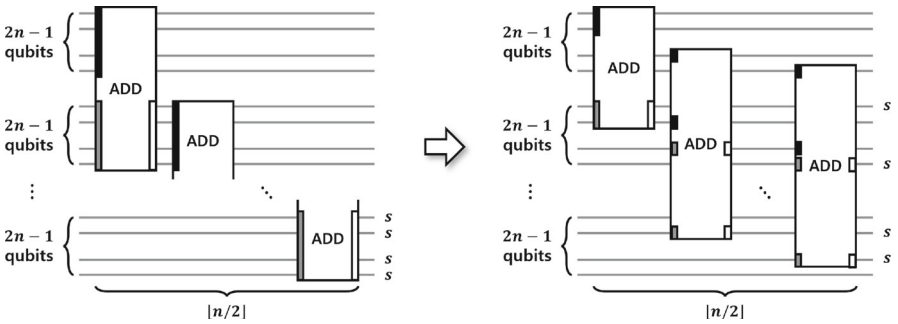


Fig. 10 Design of the PPAu using quantum full adders with half the input size

PPAu to reduce the input size of quantum full adders ($2n-1 \rightarrow n$) while maintaining the number of full adders by appropriately distributing the partial products to the input registers of full adders as shown in the right of Fig. 10.

First, the partial products represented as monomials like s_5 in Fig. 8 and the initial terms of the partial products represented as polynomials such as s_2, s_4, and s_6 are placed into the first quantum register. Note that s_0 and s_1 are not employed in PPAu, as they are immediately represented as the outcome of squaring. The i-th terms of the partial products, represented as polynomials, are placed into the i-th quantum register. The initial quantum full adder thereafter performs the addition of the upper n qubits from the first register (black rectangle of the first adder in the right of Fig. 10) and the upper n qubits from the second register (gray rectangle of the first adder). The second quantum full adder performs the addition of the n qubits depicted by the black rectangle (the upper $n/2$ qubits from the lower n qubits of the first register (the first black rectangle of the second adder) and the lower $n/2$ qubits from the upper n qubits of the second register (the second black rectangle of the second adder)) with the n qubits depicted by the gray rectangle (the upper $n/2$ qubits from the lower n qubits of the second register (the first gray rectangle of the second adder) and the lower $n/2$ qubits from the upper n qubits of the third register (the second gray rectangle of the second adder)). This procedure is repeated until the lower $n/2$ qubits of the lower n qubits in the first register are accumulated in the result register. Our quantum squaring circuit employs Gidney's quantum full adder [12], characterized by its minimal T-count and T-depth.

C. Initialization unit

The initialization unit (Iu) restores the ancilla qubits $|anc\rangle$ and temporary qubits $|t\rangle$ set by the PPSu to their original states for reuse. The Iu is composed of dagger gates of Q-AND and CNOT gates, with gate placement in reverse order of PPSu, excluding the CNOT gates that set $|x_i\rangle$ to $|s_{2i}\rangle$. Figure 11 shows the entire quantum squaring circuit when the input size n is 4.

4.2 Quantum resources for quantum squaring circuits

We estimate the quantum resource requirements (qubit count, gate cost, and depth) of proposed quantum squaring circuit generalized for an input size n, and compare the quantum resources with those proposed in existing studies [1–4, 11].

4.2.1 Qubit count

The proposed quantum squaring circuit computes the square of an n-qubit register $|a\rangle$ and outputs it to a $2n$-qubit register $|s\rangle$. The $n-3$ n-qubit registers $|anc_i\rangle$s ($0 \leq i < n-3$) are added to duplicate the input register $|a\rangle$ for the setup of the partial product with just 1 depth. The $\frac{n^2-3n+4}{2}$-qubit register $|t\rangle$ temporarily stores the remaining partial products, excluding those utilized in the result. The intermediate quantum adders in our circuit require additional ancillary qubits, specifically a $(n-1)$-qubit register $|AncAdd\rangle$ for storing intermediate carries and a carry qubit $|c\rangle$ that retains the final carry (the most significant bit) to be forwarded to the next quantum adder.

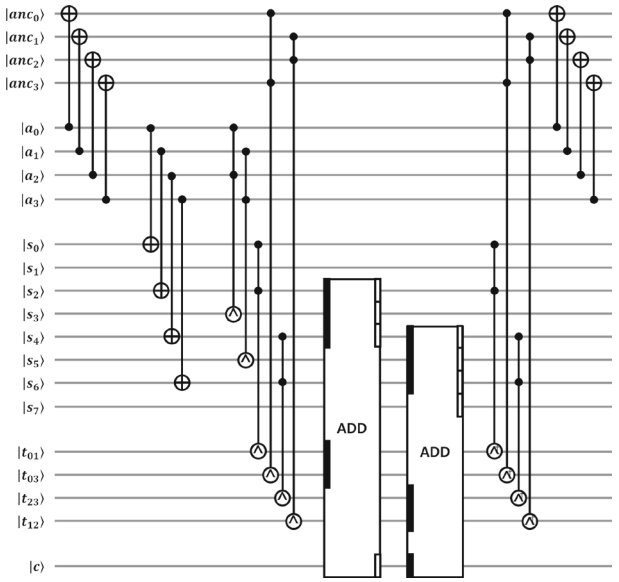


Fig. 11 The quantum squaring circuit ($n = 4$)

Table 1 The number of qubits for quantum squaring circuit

Quantum registers	Qubit counts
Input	n
Output	$2n$
Ancilla	$n(n-3)$
Temp	$(n^2 - 3n + 4)/2$
Carry	$\lfloor n/2 \rfloor - 1$
AncAdd	$n - 1$
Total	$\approx 3n^2/2$

Table 1 shows the quantum registers used in our quantum squaring circuit and their respective qubit counts. Our circuit requires $\frac{3n^2-n}{2} + \lfloor \frac{n}{2} \rfloor \approx \frac{3}{2}n^2$ logical qubits to square the number of n qubits.

4.2.2 Gate cost

Partial product setup unit

In the partial product setup unit, a set consisting of n CNOT gates duplicates the input register $|a\rangle$ to the ancillary register $|anc\rangle$ in a single depth. Furthermore, the input values are also used as factors in the cumulative sum of the partial products to obtain the square. The set of CNOT gates is also employed to duplicate the input to the register $|s\rangle$.

Table 2 The counts and depths of quantum gates and addition units utilized in the quantum squaring circuit

Gates		Counts	Depth
PPSu	CNOT	$n(n-2)$	$n-2$
	Q-AND	$\frac{n(n-2)}{2}$	1
n-qubit quantum adder		$\lfloor \frac{n}{2} \rfloor$	$\lfloor \frac{n}{2} \rfloor$
PPSu†	Q-AND†	$\frac{n-2}{2}(n-1)+1$	1
	CNOT	$n(n-2)$	$n-2$

Partial product addition unit

After the duplication of the input, Q-AND gates compute $n(n-1)/2$ partial products in a single depth. These partial products enter as inputs to an n-qubit quantum full adder, with $\lfloor n/2 \rfloor$ quantum full adders proceeding sequentially and accumulating in $\lfloor n/2 \rfloor$ depths to compute the square. The n-qubit quantum adder used in this paper, developed by Gidney, requires a T-count of $4n$ and a T-depth of n. Table 2 presents the count and depth of quantum gates and addition units utilized in the quantum squaring circuit.

4.2.3 KQ_T

As mentioned in Sect. 4.2.3, KQ_T is measured by multiplying the number of qubits by the depth of the gate used in a quantum circuit. In this paper, we specifically consider the depth of T-gates, which are expensive to implement. We estimate the KQ_T of our quantum squaring circuit based on the Qubit count (Q-count) and T-gate cost analysis in Sects. 4.2.1 and 4.2.2. The KQ_Ts of our circuit by input size are given in Sect. 4.2.4, along with a comparison with existing works.

$$KQ_T = \text{total Q-count} \times \text{T-depth} \tag{2}$$

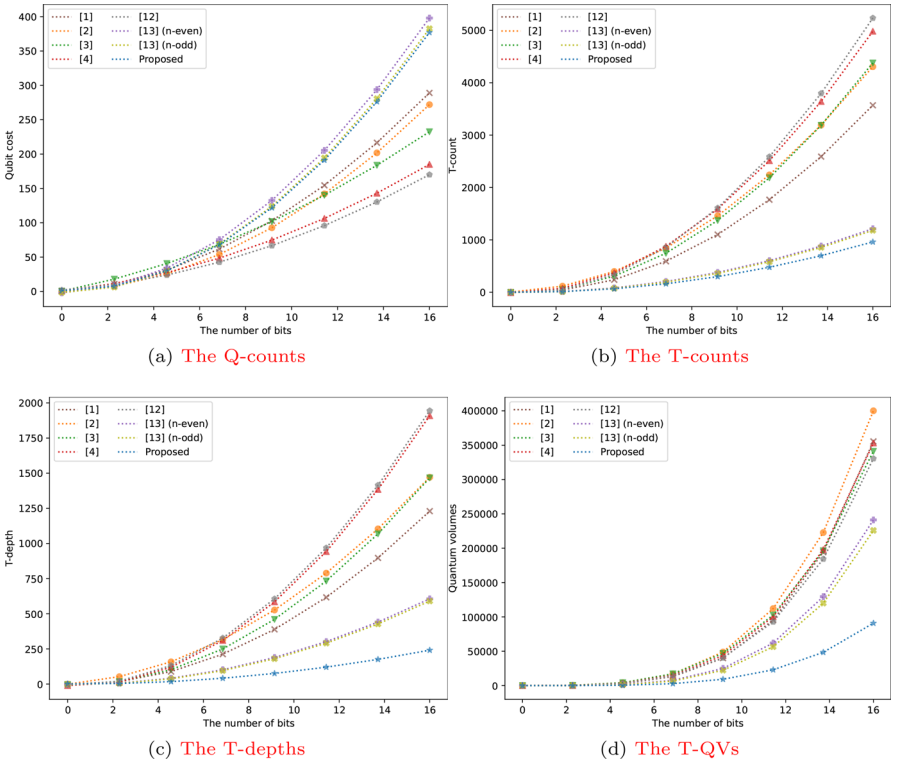
4.2.4 Cost comparison

Existing quantum squaring circuit designs [1–4, 11] have progressed toward reducing the number of qubits due to the limitations on the available number of qubits in quantum computers. The initial quantum squaring circuit [1] utilized the highest number of qubits but was the most efficient in terms of gate count and depth. Subsequent designs decreased the number of qubits but at the cost of increased gate count and depth. Unlike previous works, recent study [10] has focused on reducing the gate cost of quantum squaring circuits. Table 3 shows the Q-counts, T-counts, and T-depths of our circuit and the existing quantum squaring circuits.

Additionally, Fig. 12 illustrates the progression of Q-counts, T-counts, T-depths, and KQ_Ts as the input bit size increases for each quantum squaring circuit. Our quantum squaring circuit uses more qubits than the least qubit-costly circuit in [11], yet achieves

Table 3 The quantum resources comparison of the proposed quantum squaring circuit with the existing squaring circuits

Cost measures	Q-counts	T-counts	T-depths
[1]	$n^2 + 2n + 1$	$15n^2 - 17n + 2$	$5n^2 - 3n - 2$
[2]	$n^2 + n$	$16n^2 + 13n + 3$	$5n^2 + 12n - 1$
[3]	$0.5n^2 + 6.5n + 0.5$	$18n^2 - 14n - 6$	$6n^2 - 4n - 5$
[4]	$0.5n^2 + 3.5n + 1$	$20n^2 - 8n - 9$	$8n^2 - 8n - 11$
[11]	$0.5n^2 + 2.5n + 2$	$22n^2 - 24n - 12$	$8n^2 - 6n - 8$
[10] (n-even)	$1.5n^2 + n - 2$	$5n^2 - 4n - 4$	$2.5n^2 - 2n - 2$
[10] (n-odd)	$1.5n^2 - 1.5$	$5n^2 - 6n - 3$	$2.5n^2 - 3n - 1.5$
Proposed	$1.5n^2 - 0.5n + 1$	$4n^2 - 4n$	$n^2 - n + 1$



(a) The Q-counts (b) The T-counts

(c) The T-depths (d) The T-QVs

Fig. 12 Change in quantum resources in quantum squaring circuits as input bits increase

a reduction of 81.8% in T-count and 87.5% in T-depth. Our design utilizes 73.3% fewer T-counts and 80% less T-depth than the circuit in [1], and even reduces the T-count by 20% and the T-depth by 60% from the design in [10], which requires the lowest T-count and T-depth to date. Our circuit also shows the best results in terms of the KQ_T. The T-depth of our quantum squaring circuit is 1/2.5 to 1/8 of that of existing circuits, but the qubit count is only increased by a factor of 1/2 to 3. The trade-off between the width and depth of the quantum circuit is ideal for our quantum squaring circuit due to the slight increase in qubit count but the significant decrease in T-depth, resulting in a 50–70% reduction in KQ_T.

5 Conclusion

In this paper, we have proposed the design of a depth-optimized quantum squaring circuit. The proposed quantum squaring circuit consists of three units: PPSu, PPAu, and Iu. We first eliminated unnecessary Q-AND gates for bitwise multiplications and then implemented the PPSu in parallel with additional ancilla qubits. The implemented PPSu has a reduced Q-AND gate count from n^2 to $n(n-1)/2$ and a T-depth of two. We have also made efforts to optimize the quantum resources of the PPAu, which performs cumulative sums of partial products. We have presented a partial product distribution method to reduce the input size of the quantum adder from $2n-1$ to n, which results in a 50% reduction in both the gate count and the depth of the quantum adder used. The Iu initializes the partial products set by the PPSu and operates with a T-depth of two. The proposed quantum squaring circuit has the smallest T-count and T-depth compared to previous works, showing an average improvement of 68% in T-count and 79.7% in T-depth. Although the trade-off for optimizing the T-gate cost is a 71.4% increase in Q-count on average, the absolute increase is negligible compared to the T-gate cost. Consequently, our quantum squaring circuit has the ideal width-depth trade-off, resulting in the least consumption of KQ_T.

Appendix A The quantum squaring circuit for $n = 4$ drawn using Qiskit

See Fig. 13.

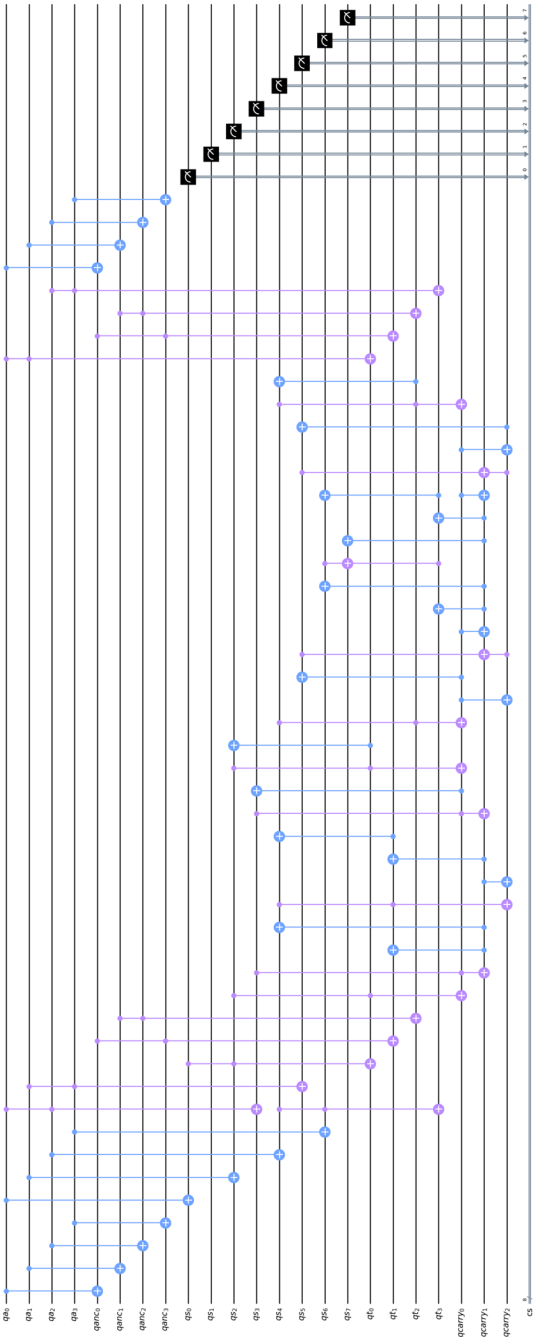


Fig. 13 The quantum squaring circuit for $n = 4$ drawn using Qiskit

Author Contributions S.C. and C.L. made substantial contributions to the design of the work and analysis and drafted the work and S.S. revised it critically for important intellectual content and approved the version to be published. All authors reviewed the manuscript.

Funding This work was supported by Quantum Computing based on Quantum Advantage challenge research through the National Research Foundation of Korea (NRF) funded by the Korean government (Ministry of Science and ICT (MSIT)) (RS-2023-00256221) and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2021R1A2C1095591).

Availability of data and materials All data generated or analyzed during this study are included in this published article.

References

1. Jayashree, H., Thapliyal, H., Agrawal, V.K.: Design of dedicated reversible quantum circuitry for square computation. In: 2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems, pp. 551–556. IEEE (2014)
2. Banerjee, A., Das, D.K.: Squaring in reversible logic using iterative structure. In: Proceedings of IEEE East-West Design & Test Symposium (EWDTS 2014), pp. 1–4. IEEE (2014)
3. Banerjee, A., Das, D.K.: Squaring in reversible logic using zero garbage and reduced ancillary inputs. In: 2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID), pp. 385–390. IEEE (2016)
4. Jayashree, H., Thapliyal, H., Agrawal, V.K.: Efficient circuit design of reversible square. Trans. Comput. Sci. **29**, 33–46 (2017)
5. Baugh, C.R., Wooley, B.A.: A two's complement parallel array multiplication algorithm. IEEE Trans. Comput. **100**(12), 1045–1047 (1973)
6. Castelvecchi, D.: Ibm releases first-ever 1,000-qubit quantum chip. Nature **624**(7991), 238–238 (2023)
7. Bhat, H.A., Khanday, F.A., Kaushik, B.K.: Optimized quantum implementation of novel controlled adders/subtractors. Quantum Inf. Process. **22**(4), 174 (2023)
8. Luo, Q.-B., Li, X.-Y., Yang, G.-W., Li, Q.: Quantum reversible circuits for gf (2 8) multiplication based on composite field arithmetic operations. Quantum Inf. Process. **22**(1), 58 (2023)
9. Amy, M., Maslov, D., Mosca, M.: Polynomial-time t-depth optimization of clifford+ t circuits via matroid partitioning. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **33**(10), 1476–1489 (2014)
10. Sultana, A., Muñoz-Coreas, E.: Resource optimized quantum squaring circuit. arXiv preprint arXiv:2406.01875 (2024)
11. Nagamani, A., Ramesh, C., Agrawal, V.K.: Design of optimized reversible squaring and sum-of-squares units. Circuits Syst. Signal Process. **37**, 1753–1776 (2018)
12. Gidney, C.: Halving the cost of quantum addition. Quantum **2**, 74 (2018)
13. Cross, A.W., Bishop, L.S., Sheldon, S., Nation, P.D., Gambetta, J.M.: Validating quantum computers using randomized model circuits. Phys. Rev. A **100**(3), 032328 (2019)
14. Cuccaro, S.A., Draper, T.G., Kutin, S.A., Moulton, D.P.: A new quantum ripple-carry addition circuit. arXiv preprint arXiv:quant-ph/0410184 (2004)

15. Portugal, R., Figueiredo, C., et al.: Reversible karatsubas algorithm. J. Univ. Comput. Sci. **12**(5), 499–511 (2006)