SCIENCE CHINA
Physics, Mechanics & Astronomy

• Article •

June 2022 Vol. 65 No. 6: 260311
https://doi.org/10.1007/s11433-021-1874-2

The circuit design and optimization of quantum multiplier and divider

Hai-Sheng Li[1*], Ping Fan[2], Haiying Xia[1], and Gui-Lu Long[3,4,5,6*]

[1] *College of Electronic Engineering, Guangxi Normal University, Guilin 541004, China;*
[2] *College of Information Engineering, East China JiaoTong University, Nanchang 330013, China;*
[3] *Department of Physics, Tsinghua University, Beijing 100084, China;*
[4] *State Key Laboratory of Low-Dimensional Quantum Physics, Tsinghua University, Beijing 100084, China;*
[5] *Beijing National Research Center for Information Science and Technology, Beijing 100084, China;*
[6] *Beijing Academy of Quantum Information Sciences, Beijing 100193, China*

Received December 29, 2021; accepted February 10, 2022; published online April 8, 2022

A fault-tolerant circuit is required for robust quantum computing in the presence of noise. Clifford + T circuits are widely used in fault-tolerant implementations. As a result, reducing T-depth, T-count, and circuit width has emerged as important optimization goals. A measure-and-fixup approach yields the best T-count for arithmetic operations, but it requires quantum measurements. This paper proposes approximate Toffoli, TR, Peres, and Fredkin gates with optimized T-depth and T-count. Following that, we implement basic arithmetic operations such as quantum modular adder and subtractor using approximate gates that do not require quantum measurements. Then, taking into account the circuit width, T-depth, and T-count, we design and optimize the circuits of two multipliers and a divider. According to the comparative analysis, the proposed multiplier and divider circuits have lower circuit width, T-depth, and T-count than the current works that do not use the measure-and-fixup approach. Significantly, the proposed second multiplier produces approximately 77% T-depth, 60% T-count, and 25% width reductions when compared to the existing multipliers without quantum measurements.

quantum multiplier, quantum divider, quantum fault-tolerant circuit, quantum computing

PACS number(s): 03.67.Lx, 42.30.Va, 03.67.Pp

Citation: H.-S. Li, P. Fan, H. Xia, and G.-L. Long, The circuit design and optimization of quantum multiplier and divider, Sci. China-Phys. Mech. Astron. **65**, 260311 (2022), https://doi.org/10.1007/s11433-021-1874-2

1 Introduction

Quantum principles enable fast quantum algorithms for factorization [1], database search [2], and second-order optimization [3]. Furthermore, quantum principles facilitate quantum communication ensuring the communication's unconditional security. Quantum communication encompasses topics like quantum key distribution [4], quantum teleportation [5], and quantum secure direct communication (QSDC) [6-8]. QSDC is one of the most important quantum communication modes and attracts much attention because it can transmit secret messages directly without sharing a key [9].

Rapid progress has been made in quantum computing hardware and quantum circuit simulation [10]. It is promising that quantum computing has the potential to be used to solve problems with practical significance. The realization of quantum computers is one of the most significant challenges in quantum computing [11]. The quantum circuit is

*Corresponding authors (Hai-Sheng Li, email: lhscqdx@163.com; Gui-Lu Long, email: gllong@tsinghua.edu.cn)

a more practical and realistic model of quantum computers than the Turing machine [12]. Toffoli, Fredkin, Peres, and TR gates are common logical gates for quantum circuits [13-15]. Furthermore, clifford + T circuits are widely used for fault-tolerant implementation [16, 17]. The Hadamard gate **H**, the phase gate **S**, and the non-Clifford gate **T** are defined by

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \mathbf{S} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{i} \end{bmatrix}, \mathbf{T} = \begin{bmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\pi/4} \end{bmatrix}.$$

Thus, **H**, **S**, **T**, the controlled-NOT gate (CNOT), and Pauli matrices

$$\mathbf{X} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix}, \mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

are elements in the Clifford + T set.

An instruction set $\{\mathbf{H}, \mathbf{S}, \mathbf{S}^\dagger, \mathrm{CNOT}, \mathbf{T}, \mathbf{T}^\dagger\}$ is universal for quantum computation [18]. Quantum circuits with T-depth 3 and T-count 7 for Toffoli, Fredkin, Peres, and TR gates have been proposed [18-21]. The T-depth one representation of the Toffoli gate was presented with four ancillae [19, 22]. Furthermore, Jones [23] used quantum measurement to reduce ancillae. In other words, the T-depth Toffoli gate performs an ancilla and a quantum measurement. There are two approximate Toffoli gates that are similar to the Toffoli gate except for relative phases and may be easier to implement [12, 24]. As a result, we design Clifford + T circuits for approximate Toffoli, TR, Peres, and Fredkin gates using the instruction set $\{\mathbf{H}, \mathbf{S}, \mathbf{S}^\dagger, \mathrm{CNOT}, \mathbf{T}, \mathbf{T}^\dagger\}$ in this study.

Quantum logic gates can realize arithmetic circuits for a quantum computer's reversible arithmetic logic unit [25], which is essential for quantum algorithms such as Shor's prime factorization algorithm and quantum exponential congruence [1,26]. The design of quantum circuits for arithmetic operations has been actively studied over the last decade [27-37]. Quantum adders for n-bit addition were proposed with $O(\log n)$ T-depth by exact Toffoli gates, requiring a large number of ancillae and at least $70n$ T-count [28-30]. As one of the best previous works for arithmetic circuits, Gidney presented a logical-AND gate and an uncomputation gate [32]. By using a measure-and-fixup approach, the uncomputation circuit has a T-count of zero and a measurement-depth of 1 [23]. Using logical-AND gates, Thapliyal et al. [33] proposed a carry-lookahead adder with $O(\log n)$ T-depth and $28n$ T-count. Furthermore, Gidney [32] demonstrated an adder for logical-AND and uncomputation gates with a T-count of $4n - 4$. But Gidney's adder used $2n - 2$ measurements and $n - 1$ ancillae. With one ancilla at most, quantum circuits for adders [21, 31, 34], modular adders and subtractors [21, 25, 31, 35], controlled modular adders and subtractors [21,25,36], and comparators [21,31] were designed. Despite the fact that parts of the above circuits have been optimized, they still require the T-depth $6n - 6$ and the T-count $14n - 7$ at least. Nam et al. [37] proposed automated methods for optimizing quantum circuits to obtain an adder with the T-count $8n - 8$ without taking the T-depth into account.

Quantum multipliers were realized with at most one ancilla and at least $4n$ qubits [21, 38, 39]. In other words, their circuit widths are no less than $4n$. In addition, the quantum Fourier transform [12, 40, 41] was used to realize a quantum divider with $4n$ qubits [42]. Furthermore, two division circuits with fewer qubits (only $3n + 3$ and $3n + 2$ qubits) were proposed [43], where $n + 1$ denotes the size of the input, representing an n-bit positive integer complement.

We designed Clifford + T circuits for basic arithmetic operations using approximate Toffoli, TR, Peres, and Fredkin gates, taking into account circuit width, T-depth, and T-count. The adder, modular adder, modular subtractor, and comparator are examples of arithmetic circuits. Then, using basic arithmetic operations, we proposed two multipliers and a divider that each required only $3n + 1$ and $3n$ qubits. There are outstanding works such as quantum multipliers with the best T-count [44, 45]. Our proposed multipliers and dividers are created by taking into account the circuit width, T-depth, and T-count. As a result, when compared with previous excellent works, they have the best circuit width.

The remainder of this paper is structured as follows. Clifford + T circuits for approximate gates are introduced in sect. 2. In sect. 3, we use approximate gates to design basic arithmetic operators. Sects. 4 and 5 each propose two quantum multipliers and a quantum divider. Sect. 6 describes the designs of vector and integer arithmetic operations. Sect. 7 provides a comparative analysis, while sect. 8 provides conclusions and future work.

2 Clifford + T circuits for approximate Toffoli, Peres, TR, and Fredkin gates

The implementations for the Toffoli, Fredkin, Peres, and TR gates are illustrated in Figure 1 [18, 21].

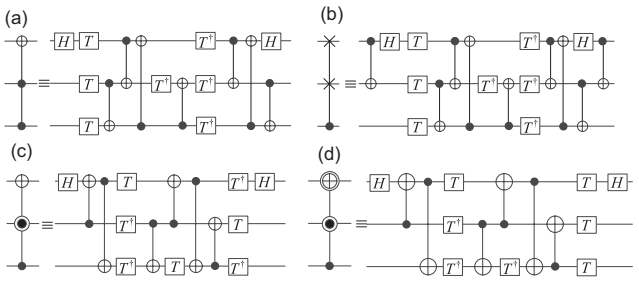

Figure 1 Implementation circuits for (a) the Toffoli gate, (b) the Fredkin gate, (c) the Peres gate, and (d) an inverse-Peres gate named TR in ref. [15].

T-depth 3 and T-count 7 are assigned to each of the four gates. TR gates are used to implement

$$TR : |C\rangle |B\rangle |A\rangle \rightarrow \left|A.\overline{B} \oplus C\right\rangle |A \oplus B\rangle |A\rangle , \quad (1)$$

where the symbols "." and "⊕" represents multiplication and exclusive-or operators, respectively. $(1 - B)$ is the value of $\overline{B}$.

We use symbols in Figure 2 to gain clarity, also to represent the controlled $S^{\otimes n}$ and $X^{\otimes n}$ gates.

The first approximate Toffoli gate is distinguished by the fact that it maps $|101\rangle$ to $-|101\rangle$ [11]. We present the approximate Toffoli gate with the first control bit 0 and its variant using the first approximate Toffoli gate. If the matrix $\mathbf{W}$ and its inverse $\mathbf{W}^{\dagger}$ are expressed as:

$$\mathbf{W} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \mathbf{W}^{\dagger} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \quad (2)$$

The controlled $\mathbf{W}$ gate can then be considered the second approximate Toffoli gate because it is identical to the Toffoli gate except for mapping $|111\rangle$ to $-|101\rangle$ [24]. Since $\mathbf{W}$ and $\mathbf{W}^{\dagger}$ can be written as $\mathbf{W} = \mathbf{S}^{\dagger}(-\mathrm{i}X)S$ and $\mathbf{W}^{\dagger} = S(-\mathrm{i}X)\mathbf{S}^{\dagger}$, we can obtain the fault-tolerant implementations for the controlled $\mathbf{W}$ gate and its inverse using the Clifford + T circuit for the controlled $-\mathrm{i}X$ gate proposed in ref. [22].

We design the fault-tolerant circuit for the first approximate Toffoli gate in Figure 3(a) by modifying the gates in Figure 1. The circuit in the top-left corner of Figure 3(a) illustrates that the function of the first approximate Toffoli gate is equal to the combination of a Toffoli gate and a controlled-Z gate. The first approximate Toffoli gate's variants are shown in Figure 3(b)-(d). The controlled $-\mathrm{i}X$ gate [22] is realized by the dashed box in Figure 3(e). We obtain the fault-tolerant implementations for the controlled $\mathbf{W}$ gate and its inverse using the controlled $-\mathrm{i}X$ gate, as shown in Figure 3(e) and (f).

The first approximate Toffoli gate is compared with the two Toffoli gates presented in refs. [32, 46]. The results are shown in Table 1 and show that complex matrices are used to express operators for the circuits in refs. [32, 46].

Using the first approximate Toffoli gate, we design the approximate Peres, TR, and Fredkin gates in Figure 4. Each of these gates has only T-depth 2 and T-count 4. Because the

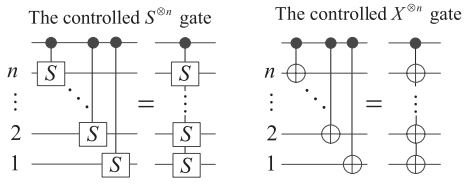

Figure 2 Symbols for the controlled $S^{\otimes n}$ and $X^{\otimes n}$ gates.

Table 1 Comparisons of approximate Toffoli gates

Approximate Toffoli gates	T-depth	T-count	Width	Matrix form
Proposed in Figure 3(a)	2	4	3	real number
[32]	2	4	3	complex number
[46]	4	4	3	complex number

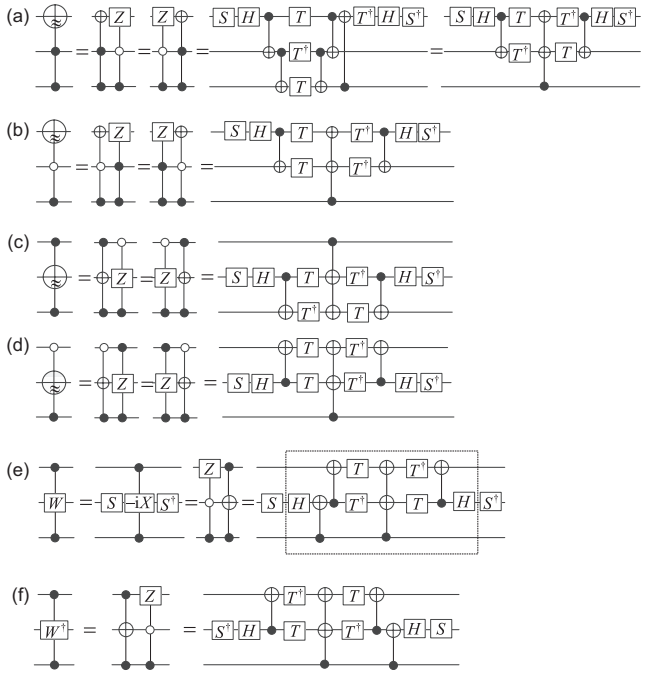

Figure 3 Clifford + T circuits for (a) the first approximate Toffoli gate, (b), (c) the first approximate Toffoli gate's variants, (e) the second approximate Toffoli gate (i.e., a controlled W gate), and (f) the inverse of the controlled $\mathbf{W}$ gate, and (d)-(f) the first approximate Toffoli gate's variants. The quantum symbols of these gates are shown on the lefts (a)-(f). Their Clifford + T circuits are depicted on the right.

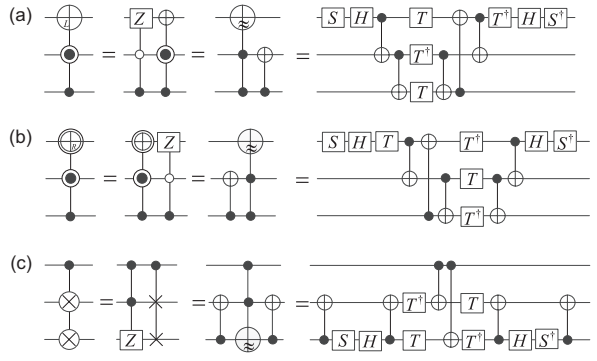

Figure 4 Clifford + T circuits for the approximate Peres (a), TR (b), and Fredkin (c) gates.

T-depths and T-counts of the Clifford + T circuits without ancillae for Toffoli, Peres, and TR gates are 3 and 7 [18-21], the proposed circuits have lower T-depths and T-counts.

3 Basic arithmetic operators for approximate Toffoli, Peres, and TR gates

We describe the designs of basic arithmetic operators and controlled arithmetic operators in this section. The former includes a quantum adder, a quantum comparator, a quantum modular adder, a quantum modular subtractor, and a quantum adder-subtractor. The latter consists of a controlled modular adder and a controlled modular subtractor. To be more specific, the "modular" arithmetic operators described above should be described as "2's complement" arithmetic operators (with/without carry output). However, to facilitate comparison with existing works, we continue to refer to these operators requiring the modulus to be a power of 2 as "modular" arithmetic operators.

3.1 Designs of basic arithmetic operators

For arithmetic operators, Toffoli, Peres, and TR gates are typically paired, as shown by the paired Toffoli and Peres gates in Figure 5(a). We present the following lemma for implementing paired Toffoli and Peres gates by approximate gates.

Lemma 1 Paired Toffoli and Peres gates in Figure 5(a) can be realized by corresponding approximate gates shown in Figure 5(d).

Proof For any single-qubit state $|x\rangle$, we have $Z|x\rangle = (-1)^x|x\rangle$. As a result, the circuit in Figure 5(b) on the left can be transformed into the one in the right, where U is a unitary gate. We eliminate the paired controlled-Z gates in Figure 5(c) because $Z^2 = I$. As a result, the approximate gates depicted in Figure 5(d) can realize paired Toffoli and Peres gates.

Combining the circuits in Figures 3-6, we obtain optimization rules for paired gates shown in Figure 7.

When the operation U in Figure 7 is one of the Peres, Toffoli, and NOT gates, we further optimize T-depth and total depth are further optimized to obtain Clifford + T circuits in Figure 8.

Using the above lemma, we give Corollary 1 as follows.

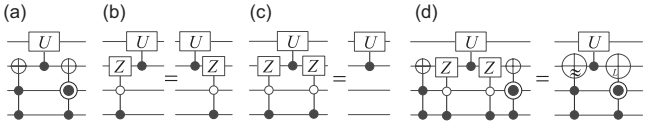

Figure 5 Implementation of (a) paired Toffoli and Peres gates, (b), (c) removing paired controlled-Z gates, and (d) paired approximate Toffoli and Peres gates.

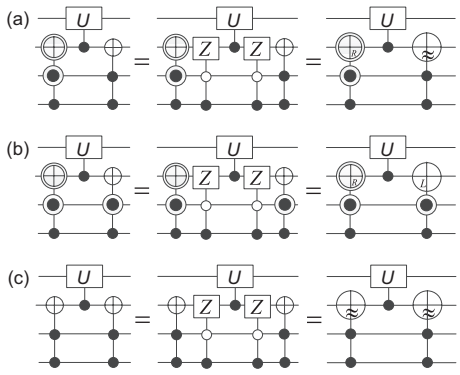

Figure 6 Implementations of (a) paired TR and Toffoli gates, (b) paired TR and Peres gates, and (c) paired Toffoli and Toffoli gates.

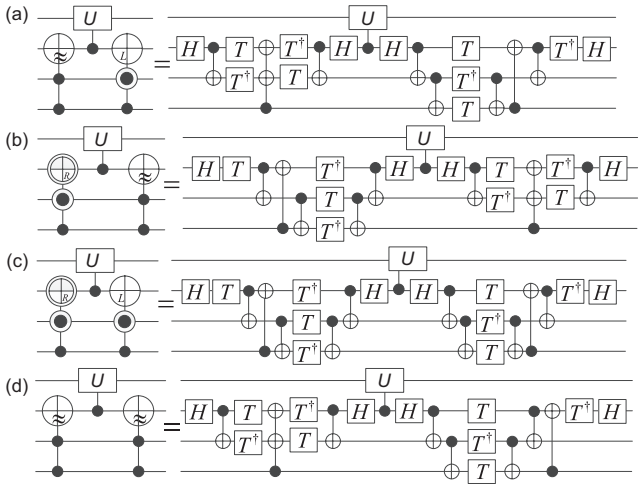

Figure 7 Optimizing rules for paired gates showing rules 1-4 in (a)-(d).

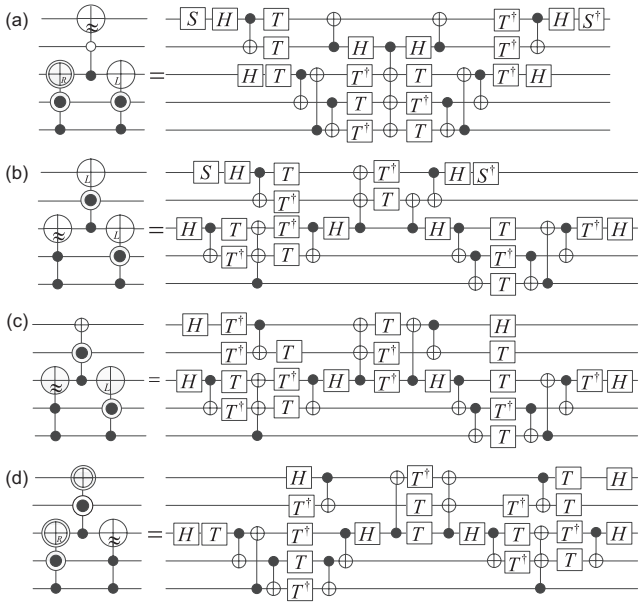

Figure 8 Further optimizing rules for paired gates showing rules 5-8 in (a)-(d).

Corollary 1 Arithmetic operators can be realized with a maximum of T-depth $2n+1$, T-count $8n-4$, and circuit width $2n+1$. Quantum adder, comparator, modular adder, modular subtractor, and adder-subtractor are among the arithmetic operators presented in ref. [21].

Proof Suppose that $|b\rangle = |b_{n-1}b_{n-2}\dots b_0\rangle$ and $|a\rangle = |a_{n-1}a_{n-2}\dots a_0\rangle$ are inputs of these arithmetic operations. $|S_A\rangle$, $|S_{MA}\rangle$, $|D_{MS}\rangle$, and $|C_{MAS}\rangle$ are outputs, i.e.,

$$\begin{cases} S_A = a + b, \\ S_{MA} = (a+b) \bmod 2^n, \\ D_{MS} = (b-a) \bmod 2^n, \\ C_{MAS} = (b + (-1)^e a) \bmod 2^n, \end{cases} \tag{3}$$

with $|S_A\rangle = |s_n s_{n-1}\dots s_0\rangle$, $|S_{MA}\rangle = |s_{n-1}\dots s_0\rangle$, $|D_{MS}\rangle = |d_{n-1}\dots d_0\rangle$, $|C_{MAS}\rangle = |h_{n-1}\dots h_0\rangle$, $s_k, d_k, h_k, d \in \{0,1\}$, and $k \in \{0,1,\dots,n\}$. By Lemma 1 and Corollary 1, using corresponding approximate gates, we implement paired gates in basic arithmetic operators and design the circuits for these basic arithmetic operations in Figure 9. Using the comparator presented in Figure 9(b), we obtain $|c\rangle = |0\rangle$ for $b \geq a$ and $|c\rangle = |1\rangle$ for $b < a$. The circuit in Figure 9(e) implements the modular addition for $|e\rangle = |0\rangle$ and the modular subtraction for $|e\rangle = |1\rangle$.

We obtain fault-tolerant implementations of these arithmetic operators with T-depth $2n+1$, T-count $8n-4$, circuit width $2n+1$ by combining rules in Figures 7 and 8, The detailed circuits of 3-bit arithmetic operators are shown as examples in Figure 10.

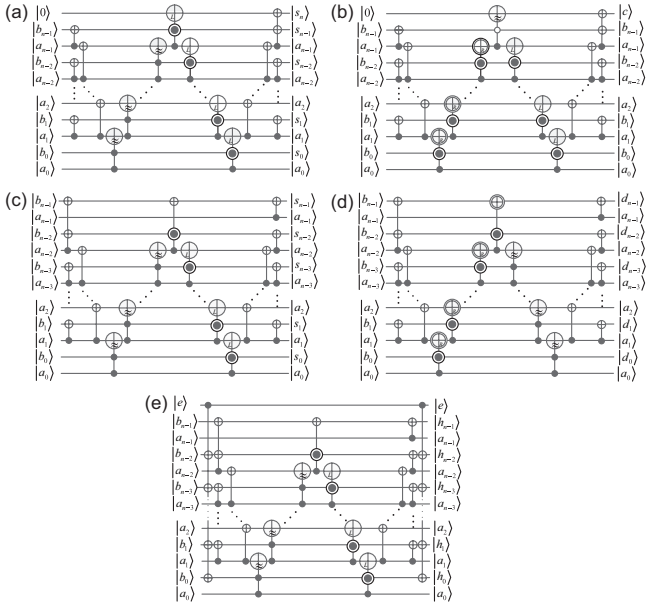

Figure 9 Basic arithmetic operator designs for approximate gates, including (a) a quantum adder, (b) a quantum comparator denoted as COM, (c) a quantum modular adder denoted as MA, (d) a quantum modular subtractor denoted as MS, and (e) a quantum adder-subtractor denoted as MAS.

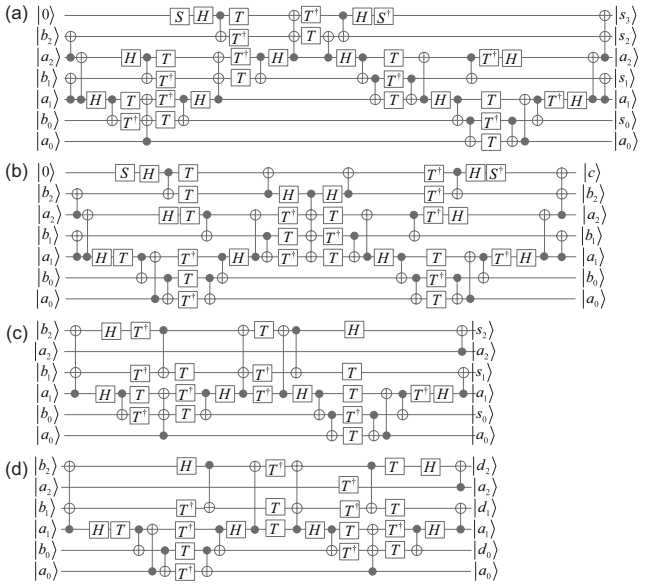

Figure 10 Clifford + T circuits for basic arithmetic operators illustrating (a) the quantum adder, (b) the quantum comparator, (c) the quantum modular adder, and (d) the quantum modular subtractor.

Due to $\mathbf{S}^\dagger(S|x\rangle) = \mathbf{S}^\dagger(\mathrm{i}^x|x\rangle) = |x\rangle$, we eliminate the paired S and $\mathbf{S}^\dagger$ gates in Clifford + T circuits and give Clifford + T circuits for the quantum adder, modular adder, modular subtractor, and comparator in Figure 10. We can deduce from Figure 9(e) infer that the Clifford + T circuit of the quantum adder-subtractor is made up of the circuit in Figure 10(c) and $2n-2$ CNOT gates.

3.2 Designs of controlled arithmetic operators

Lemma 2 The circuit consisting of paired gates (see Figures 5 and 6) and Toffoli gates can be realized using corresponding approximate gates, controlled $-\mathrm{i}X$ gates, and a controlled $S^{\otimes n}$ gate.

Proof A Toffoli gate can be decomposed into a controlled $-\mathrm{i}X$ gate and a controlled S gate [22]. A circuit made up of paired gates and Toffoli gates is shown on the left in Figure 11. Because $S|x\rangle = \mathrm{i}^x|x\rangle$, we obtain a similar result with the circuit in Figure 5(b). In other words, the middle circuit in Figure 11 is provided. Controlled-Z gates and controlled-S gates are commutable as demonstrated by the equation $SZ = ZS$. As a result, the optimized circuit is shown on the right in Figure 11.

We get similar results when using other paired gates and Toffoli gates in our circuits. As a result, the lemma holds.

Due to $T^2 = S$, $T^3 = ST$, $T^4 = Z$, $T^5 = ZT$, $T^6 = ZS$, $T^7 = T^\dagger$, and $T^8 = I$, we obtain $T^n \in \{T, S, ST, Z, ZT, SZ, T^\dagger, I\}$ for any integer n. Therefore, the Clifford + T circuit in Figure 12(a) reveals that the controlled

$S^{\otimes n}$ gate can be implemented with T-depth 2 and T-count $2n+1$, at most.

Corollary 2 T-depth $3n+1$, T-count $14n-7$, and width $2n+1$ can be used to implement a controlled modular adder and subtractor.

Proof Let $|C_{MA}\rangle$ and $|C_{MS}\rangle$ be the outputs of these controlled arithmetic operations, i.e.,

$$\begin{cases} C_{MA} = (b+da) \bmod 2^n, \\ C_{MS} = (b-da) \bmod 2^n, \end{cases} \tag{4}$$

with $|b\rangle = |b_{n-1}b_{n-2}\ldots b_0\rangle$, $|a\rangle = |a_{n-1}a_{n-2}\ldots a_0\rangle$, $|C_{MA}\rangle = |t_{n-1}\ldots t_0\rangle$, $|C_{MS}\rangle = |l_{n-1}\ldots l_0\rangle$, $t_k, l_k, d \in \{0,1\}$, and $k \in \{0,1,\ldots,n\}$. By Lemma 2, we design circuits of controlled arithmetic operations in Figure 13.

We use rules in Figure 12 to create fault-tolerant CMA and CMS implementations. Clifford + T circuits of 3-bit controlled arithmetic operators, for example, are shown in Figure 14. These fault-tolerant implementations demonstrate that the corollary is correct.

4 Designs for quantum multipliers

We propose two multipliers for the following multiplication in this section:

$$\begin{cases} |1\rangle|b\rangle|0\rangle^{\otimes n}|a\rangle \to |0\rangle|ba\rangle|a\rangle, \text{ for } a \neq 0, \\ |1\rangle|b\rangle|0\rangle^{\otimes n}|a\rangle \to |1\rangle|0\rangle^{\otimes 2n}|b\rangle, \text{ for } a = 0, \end{cases} \tag{5}$$

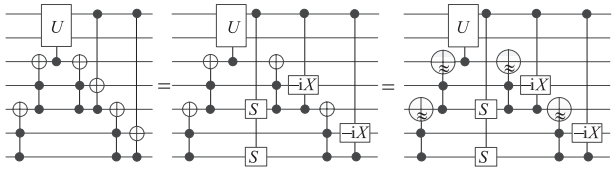

Figure 11 Implementation of paired gates and Toffoli gates.

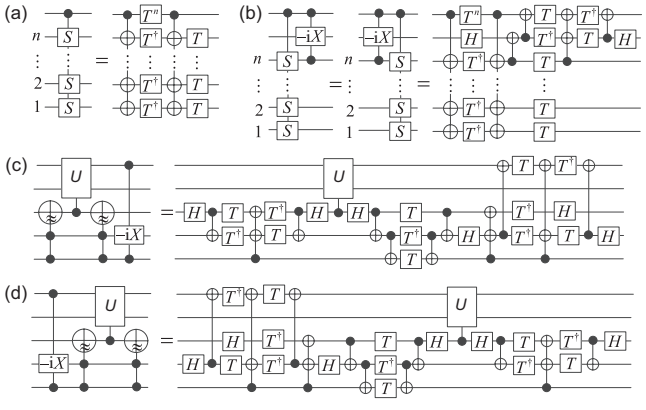

Figure 12 Optimization rules for controlled arithmetic operators shown rules 9-12 in (a)-(d).

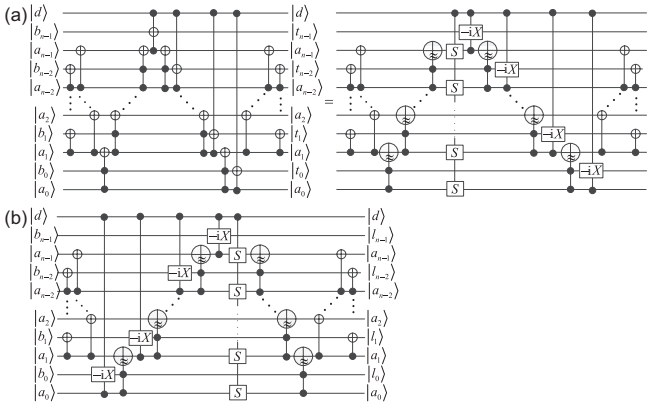

Figure 13 Controlled arithmetic operator designs for approximate gates showing (a) a controlled modular adder denoted as CMA and (b) a controlled modular subtractor denoted as CMS.

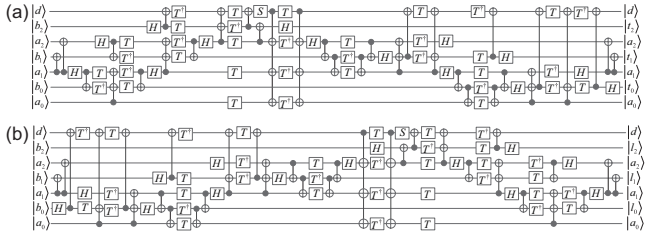

Figure 14 Clifford + T circuits for 3-bit controlled arithmetic operators, illustrating (a) the controlled modular adder and (b) the controlled modular subtractor.

where a and b are both n-bit integers.

4.1 First special multiplier for ab with $a \neq 0$

We define the following equations for the multiplication of ba with two n-bit integers $a \neq 0$ and b,

$$\begin{aligned} &s^0 = 0, \\ &s^1 = s^0 + b_0 \times a, \\ &\vdots \\ &s^n = s^{n-1} + b_{n-1} \times a \times 2^{n-1}. \end{aligned} \tag{6}$$

Due to $s = \sum_{k=0}^{n-1} 2^k b_k a$, we obtain $s^n = s$. The iteration circuit presented in Figure 15(a) realizes

$$\begin{aligned} &\left|s^k\right\rangle = \left|s^k_{n+k-1}\ldots s^k_{k+1}s^k_k\right\rangle\left|s^k_{k-1}\ldots s^k_1 s^k_0\right\rangle \\ &\to \left|s^{k+1}_{n+k}\ldots s^{k+1}_{k+1}s^{k+1}_k\right\rangle\left|s^k_{k-1}\ldots s^k_1 s^k_0\right\rangle = \left|s^{k+1}\right\rangle, \end{aligned} \tag{7}$$

with $k = 0, 1, \ldots, n-1$. As a result, we can perform n iterations to obtain the multiplication ba with $a \neq 0$, and design its circuit in Figure 15(d). We obtain by applying rules 13 and 14, the Clifford + T circuit of the iteration IM with T-depth $3n+2$ and T-count $14n-5$. The Clifford + T circuits of the 3-bit first special multiplier for example, Figure 16 to realize: $|b\rangle|0\rangle^{\otimes 3}|a\rangle \to |ba\rangle|a\rangle$.

4.2 First multiplier

Considering two n-bit integers a and b with $ab = 0$, and using the approximate Fredkin gate shown in Figure 4(c), we design the operator A_{swap} in Figure 17(a) to implement

$$|d\rangle|a\rangle|b\rangle \to |d\rangle|a'\rangle|b'\rangle, \tag{8}$$

with $a' = \bar{d}a + db$, $b' = \bar{d}b + da$, $\bar{d} = 1 - d$, $\bar{b} = 2^n - 1 - b$, and $d \in \{0, 1\}$.

For $a = 0$, the iteration IM only transforms $|b_k\rangle$ into $|\overline{b_k}\rangle$. The executed result is $|s^n\rangle = |\bar{b}\rangle|0\rangle^{\otimes n}$ by the circuit in Figure 17(a). We use the comparator in Figure 9(b) to give: $|c\rangle = |0\rangle$ for $a \neq 0$ and $|c\rangle = |1\rangle$ for $a = 0$.

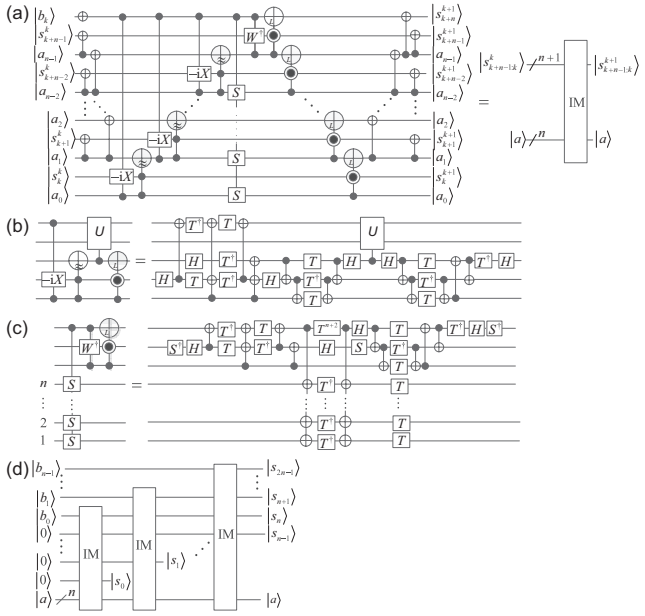

Figure 15 Design of the first special multiplier for ba with $a \neq 0$ showing (a) an iteration of the first special multiplier, denoted as IM, (b) rule 13, (c) rule 14, and (d) the circuit of the first special multiplier.

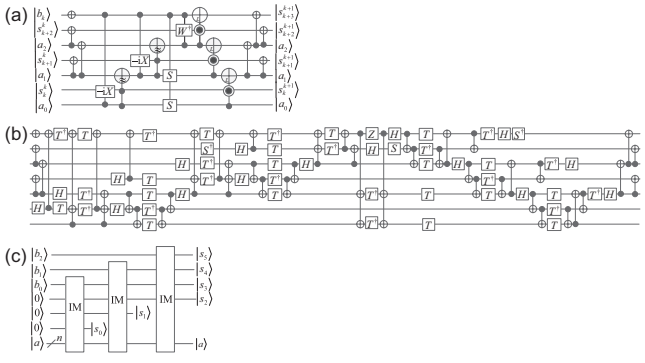

Figure 16 The 3-bit first special multiplier showing (a) the 3-bit IM, (b) the Clifford + T circuit for the 3-bit IM, and (c) the circuit of the 3-bit first special multiplier.

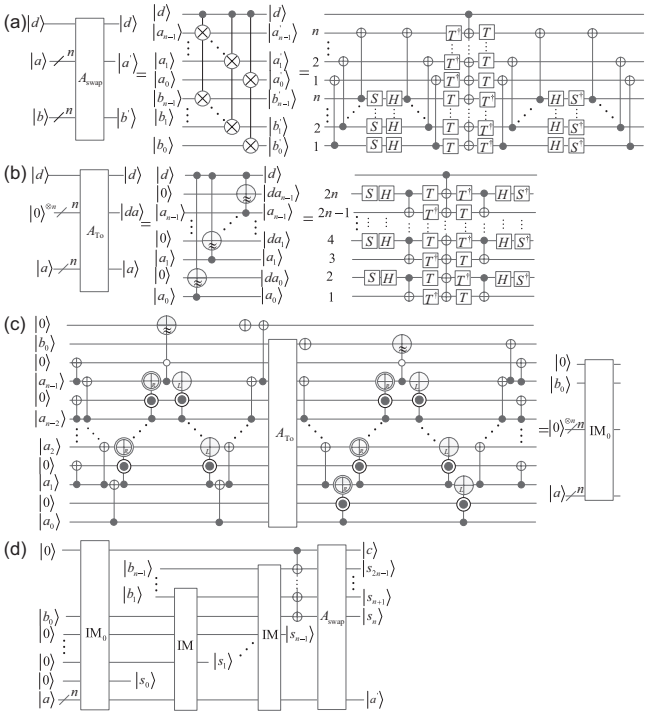

Figure 17 Implementation of the first multiplier showing (a) the operator A_{swap}, (b) the operator A_{TO}, (c) the first iteration IM_0, and (d) the complete circuit for the first multiplier.

Analyzing the input $|0\rangle|b_0\rangle|0\rangle|a_{n-1}\rangle\ldots|0\rangle|a_1\rangle|0\rangle|a_0\rangle$, we merge the comparator and the first iteration to give the first iteration, denoted as IM_0 in Figure 17(c). Thus, the first iteration IM_0 realize

$$\begin{cases} |0\rangle|b_0\rangle|0\rangle^{\otimes n}|a\rangle \to |0\rangle|b_0 a\rangle|a\rangle, & \text{for } a \neq 0, \\ |0\rangle|b_0\rangle|0\rangle^{\otimes n}|a\rangle \to |1\rangle|b_0 a\rangle|a\rangle, & \text{for } a = 0. \end{cases} \tag{9}$$

Using operators IM_0, IM, and A_{swap}, we present the first multiplier in Figure 17(d) to implement the multiplication in eq. (5).

We obtain the fault-tolerant implementation of the first iteration IM_0 with T-depth $4n$ and T-count $20n - 16$, by combining the operator A_{TO} in Figure 17(b) and rules in Figures 7 and 8, Furthermore, we give the Clifford + T circuit of A_{swap} in Figure 17(a) with T-depth 2 and T-count $4n$. As a result, the T-depth and T-count for the first multiplier are given by $4n + (3n + 2)(n - 1) + 2 = 3n^2 + 3n$ and $20n - 6 + (14n - 5)(n - 1) + 4n = 14n^2 + 5n - 11$, respectively, circuits for the 3-bit first multiplier, for example, are shown in Figure 18.

4.3 Second multiplier

The multiplication ba can be expressed as:

$$ba = [-\overline{b_0} + (-1)^{\overline{b_1}}]a + 2^{n-1}a + \sum_{k=2}^{n-1}(-1)^{\overline{b_k}}2^{k-1}a, \tag{10}$$

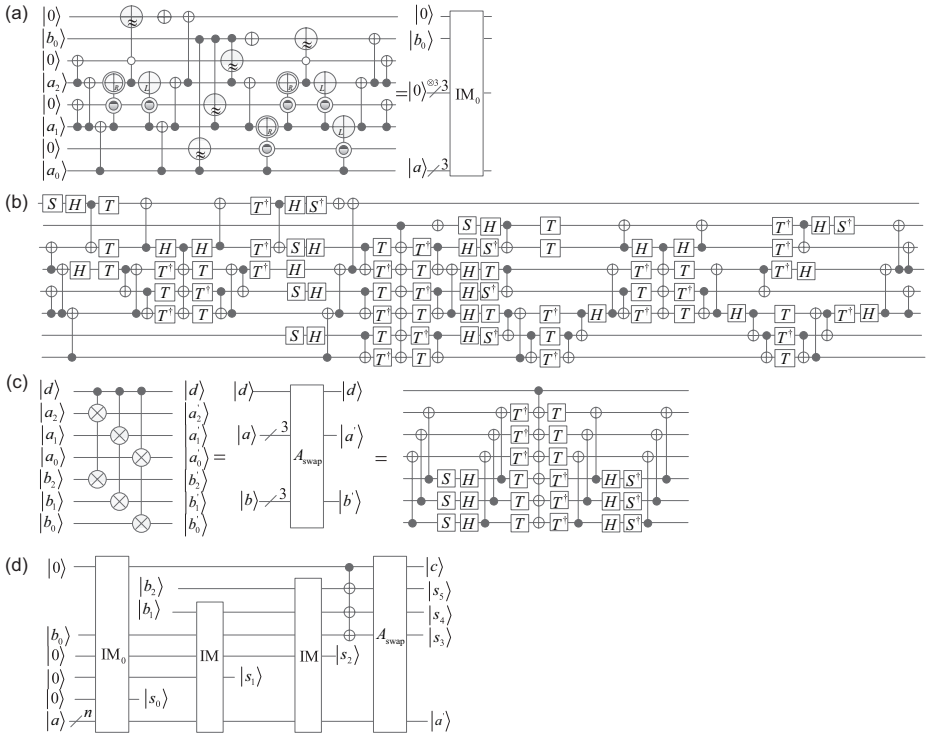

Figure 18 Circuits of the 3-bit first multiplier showing (a) the iteration of the first multiplier, denoted as IM_0, (b) the Clifford + T circuit for IM_0, (c) the Clifford + T circuit of A_{swap}, and (d) the circuit of the first multiplier.

with $\overline{b_k} = 1 - b_k$. We create an operator denoted as SM in Figure 19(a) to implement $[-\overline{b_0} + (-1)^{\overline{b_1}}]a$. As a result, multiplication is broken down into an operation SM, an addition, and $(n-2)$ addition-subtractions. We create the second multiplier by combining the operators SM, MA, A_{swap}, and MAS. Their circuits are shown in Figure 19. We obtain the Clifford + T circuit of the operator SM with T-depth $5n-1$ and T-count $20n-9$ by using rules 16 and 17 in Figure 19. As a result, the second multiplier is made up of an operator SM, a modular adder MA, an operator A_{swap}, and $(n-2)$ modular adder-subtractors. The following equations are used to calculate the T-depth and T-count for the second multiplier by $5n-1+2n-1+2+(2n-1)(n-2) = 2n^2+2n+2$ whereas, $20n-9+8n-9+4n+(8n-9)(n-2) = 8n^2+7n$. The circuits for the 3-bit second multiplier, for example can be seen in Figure 20.

5 Design for a quantum divider

We design the circuit ID in Figure 21(a) to implement:

$$\begin{aligned} &|s\rangle\,|b_{n-k}\rangle = |s_{n-1}\ldots s_1 s_0\rangle\,|b_{n-k}\rangle \\ &\rightarrow |q_{k-1}\rangle\left|s'_{n-1}\ldots s'_1 s'_0\right\rangle = |q_{k-1}\rangle\,|s'\rangle\,, \end{aligned} \tag{11}$$

where $|s'\rangle = |s-x\rangle$, $x = a$ for $s_{n-1}\ldots s_1 s_0 b_k \geq a$, and $x = 0$ for $s_{n-1}\ldots s_1 s_0 b_k < a$. Using rules 18-20 in Figure 21, using T-depth $3n+2$ and T-count $14n-5$, we obtain the Clifford + T circuit for the iteration ID. Analyzing the input of the first iteration $|0\rangle\,|0\rangle\,|a_{n-1}\rangle\ldots|0\rangle\,|a_1\rangle\,|b_{n-1}\rangle\,|a_0\rangle$, we simplify the iteration ID into ID_0 in Figure 21(b) . Similarly, we can give the Clifford + T circuit of ID_0 with T-depth $3n+1$ and T-count $12n-4$. Then, using the iterations ID_0 and ID, we provide the circuit of a quantum divider in Figure 21(f) to implement the division b/a. It has a quotient and remainder of $q = q_2q_1q_0$ and $r = r_2r_1r_0$, respectively. The circuits for the 3-bit divider, for example, are shown in Figure 22.

6 Implementation of vector-integer operations

A $2^m \times 1$ vector V_{2^m} can be stored in the following state [47-49]:

$$|\psi_m\rangle = \frac{1}{\sqrt{2^m}}\sum_{j=0}^{2^m-1}|V_{2^m}(j)\rangle\,|j\rangle\,, \tag{12}$$

where $|j\rangle = |j_{m-1}\ldots j_1 j_0\rangle$ and $V_{2^m}(j)$ denote the jth location of a vector and the corresponding element value, respectively.

For instance, the state

$$|\psi_2\rangle = (|000\rangle\,|00\rangle + |100\rangle\,|01\rangle + |101\rangle\,|10\rangle + |111\rangle\,|11\rangle)/2$$

encodes the column vector $V_4 = [0,4,5,7]^\dagger$. The vector-integer multiplication and division can then be implemented using the proposed multipliers and dividers.

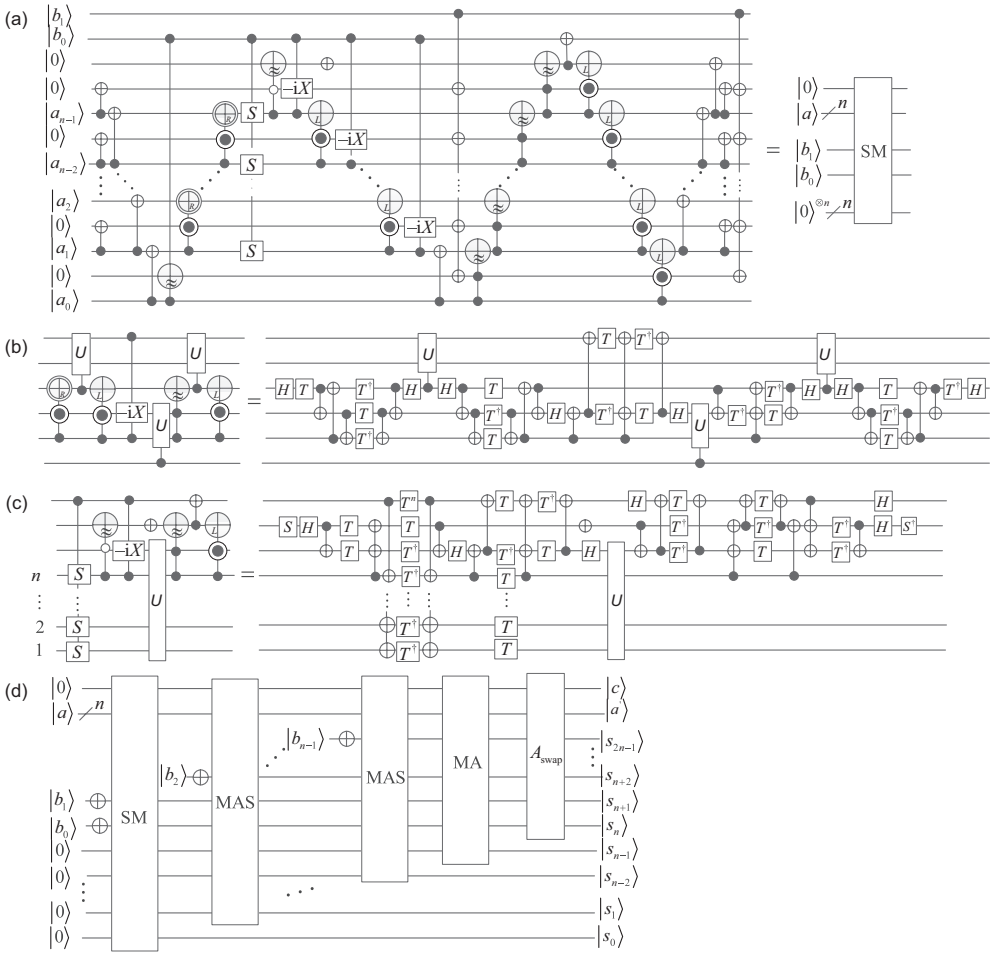

Figure 19 Design of the second multiplier including (a) the iteration SM, (b) rule 16, (c) rule 17, and (d) the circuit for the second multiplier.

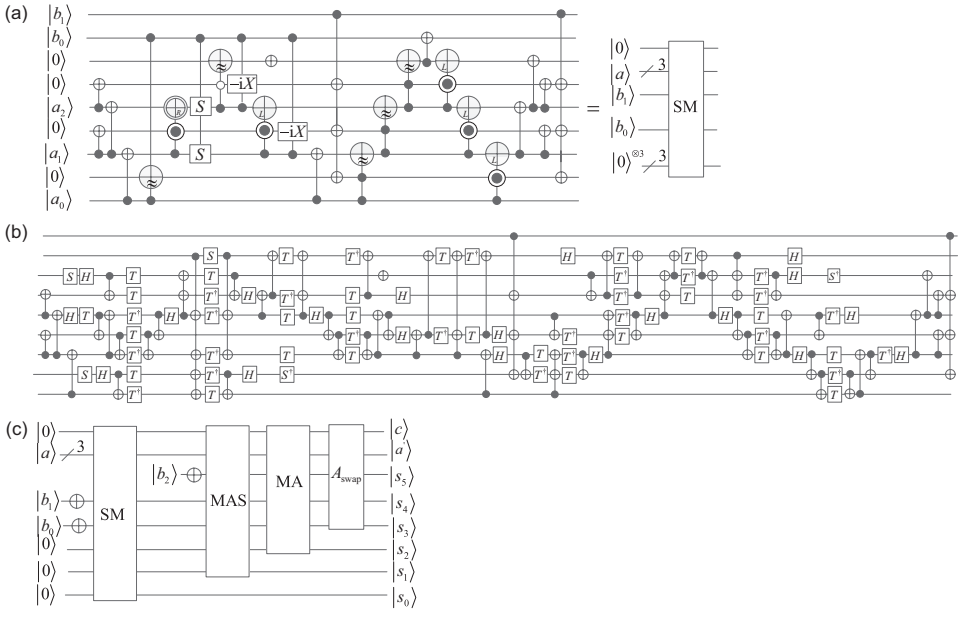

Figure 20 Circuits of the 3-bit second multiplier showing (a) the iteration of the first multiplier, denoted as SM, (b) the Clifford + T circuit for SM, and (c) the circuit of the second multiplier.

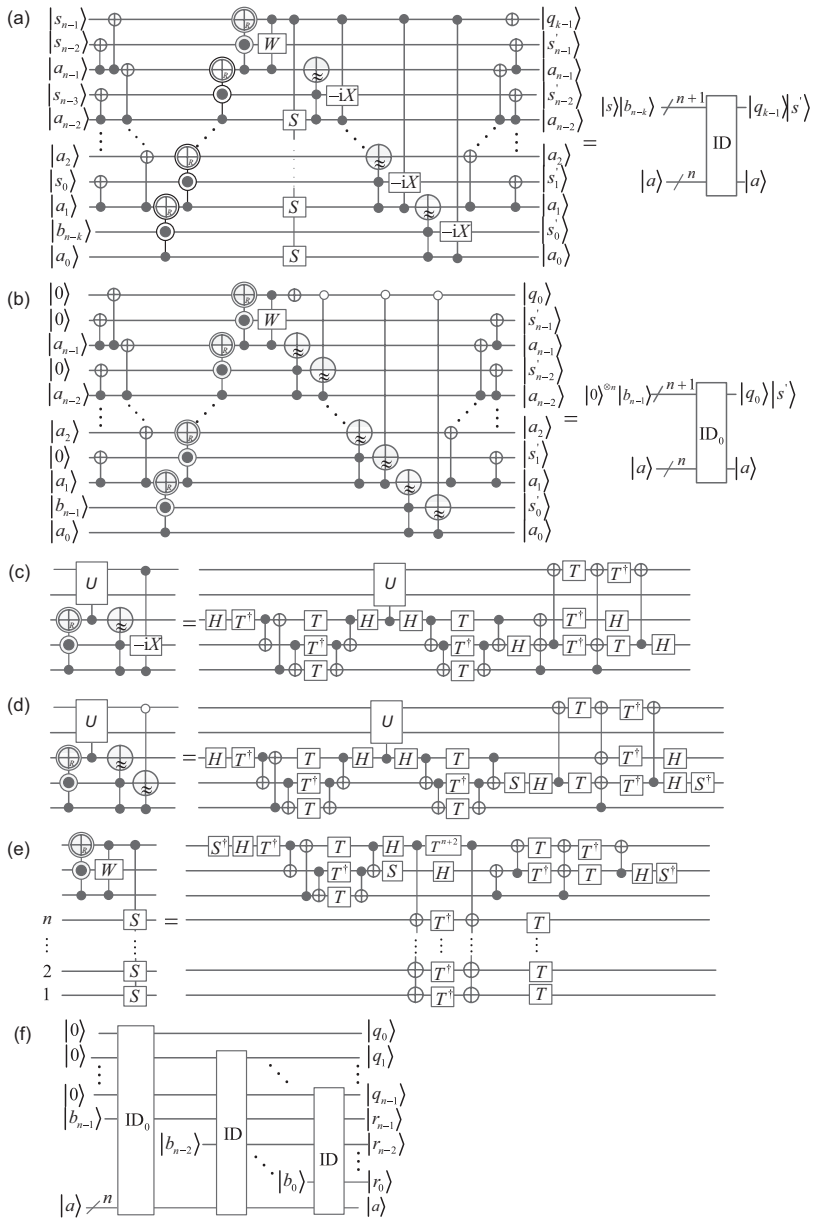

Figure 21 Implementation of a quantum divider showing (a) an iteration of the divider ID, (b) the first iteration ID_0, (c)-(e) rules 18-20, and (f) the circuit for the divider.

Set $\mathbf{V}_8 = [0,1,2,3,4,5,6,7]^\dagger$ and $a = a_2a_1a_0 = 101 = 5$. We use the following examples to describe vector-integer multiplication and division implementations:

$$\begin{cases} \mathbf{V}_8 \times a = [0,5,10,15,20,25,30,35]^\dagger, \\ (Q,R) = V_8/a, \end{cases} \tag{13}$$

with the quotient $\mathbf{Q} = [0,0,0,0,0,1,1,1]^\dagger$ and the remainder $\mathbf{R} = [0,1,2,3,4,0,1,2]^\dagger$. We can deduce from eq. (12) that the following state exists.

$$|\psi_3\rangle = \frac{1}{\sqrt{2^3}} \sum_{k=0}^{7} |V_8(k)\rangle\,|k\rangle$$

$$= \frac{1}{\sqrt{2^3}}(|000\rangle\,|000\rangle + |001\rangle\,|001\rangle + |010\rangle\,|010\rangle + |011\rangle\,|011\rangle + |100\rangle\,|100\rangle + |101\rangle\,|101\rangle + |110\rangle\,|110\rangle + |111\rangle\,|111\rangle)$$

encodes the column vector V_8.

We use three perfect shuffle permutations for clarity; $P_{2^{n-1},2}$, and $P_{2,2^{n-1}}$, Γ_{2^n}, and four quantum black box circuits I_f, I_f^{-1}, I_g, and I_g^{-1} are used to shift quantum lines [50], where

$$\begin{cases} P_{2^{n-1},2}\,|j_{n-1}\ldots j_2 j_1 j_0\rangle = |j_0 j_{n-1}\ldots j_2 j_1\rangle, \\ P_{2,2^{n-1}}\,|j_{n-1}\ldots j_1 j_0\rangle = |j_{n-2}\ldots j_1 j_0 j_{n-1}\rangle, \\ \Gamma_{2^n}\,|j_{n-1}\ldots j_1 j_0\rangle = |j_0 j_1\ldots j_{n-1}\rangle, \end{cases} \tag{14}$$

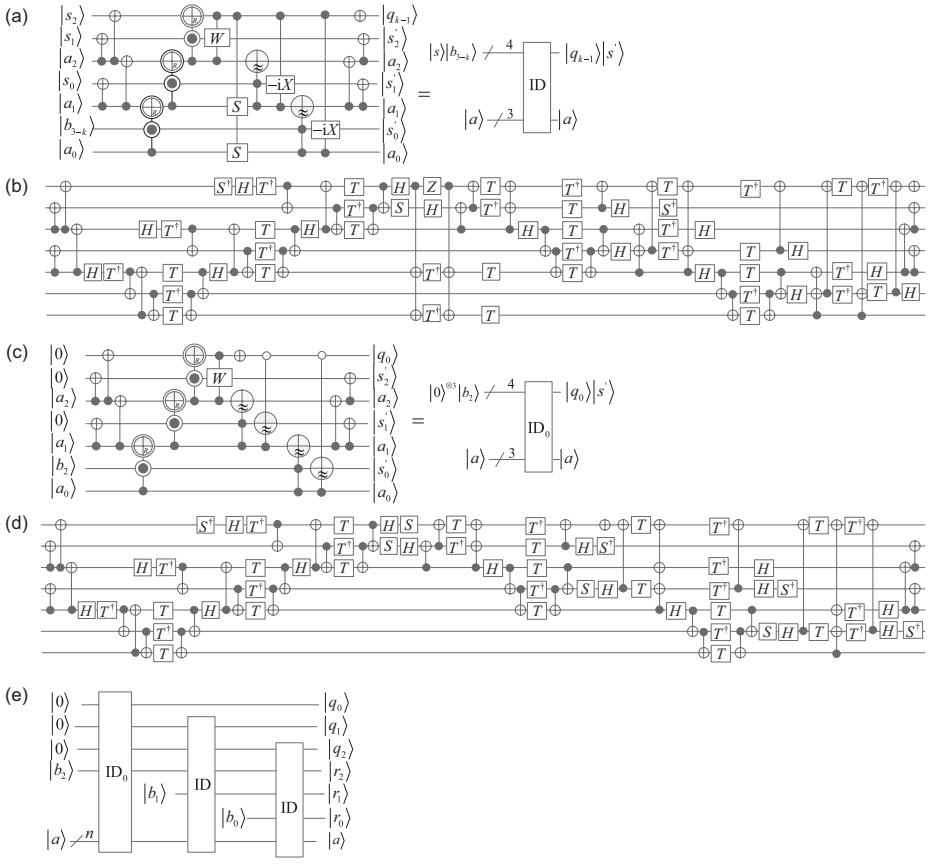

Figure 22 Circuits for the 3-bit divider showing (a) the 3-bit iteration ID, (b) the Clifford + T circuit of ID, (c) the 3-bit iteration ID_0, (d) the Clifford + T circuit of ID_0, and (e) the circuit of the 3-bit divider.

and

$$\begin{cases} I_f\,|a_{n-1}\ldots a_1a_0\rangle\,|b_nb_{n-1}\ldots b_1b_0\rangle \\ =|b_nb_{n-1}a_{n-1}\ldots b_1a_1b_0a_0\rangle\,, \\ I_f^{-1}\,|b_nb_{n-1}a_{n-1}\ldots b_1a_1b_0a_0\rangle \\ =|a_{n-1}\ldots a_1a_0\rangle\,|b_nb_{n-1}\ldots b_1b_0\rangle\,, \\ I_g\,|a_{n-1}\ldots a_1a_0\rangle\,|b_{n-1}\ldots b_1b_0\rangle \\ =|b_{n-1}a_{n-1}\ldots b_1a_1b_0a_0\rangle\,, \\ I_g^{-1}\,|b_{n-1}a_{n-1}\ldots b_1a_1b_0a_0\rangle \\ =|a_{n-1}\ldots a_1a_0\rangle\,|b_nb_{n-1}\ldots b_1b_0\rangle\,, \end{cases} \tag{15}$$

can be implemented by a maximum of n swap gates. We design the circuit in Figure 23 to realize the vector-integer division in eq. (13). using the iterations ID_0 and ID.

The input in Figure 23 is

$$|a\rangle\,|0\rangle^{\otimes 3}\,|\psi_3\rangle=\frac{1}{\sqrt{8}}\sum_{k=0}^{7}|101\rangle\,|000\rangle\,|V_8(k)\rangle\,|k\rangle. \tag{16}$$

The quantum circuit performs the following operations in parallel: $|101\rangle\,|000\rangle\,|V_8(k)\rangle\,|k\rangle$, $k=0,\ldots,6,7$. Consider $k=6$, i.e., $|101\rangle\,|000\rangle\,|V_8(6)\rangle\,|6\rangle=|101\rangle\,|000\rangle\,|110\rangle\,|110\rangle$; we illustrate the executing process of the vector-integer division as follows.

Due to $0001<101$, we from eq. (11) know that the iteration ID_0 keeps the input unchanged, i.e.,

$$\begin{aligned} &|101\rangle\,|000\rangle\,|110\rangle\,|110\rangle=|101000110\rangle\,|110\rangle \\ &\overset{1}{\Rightarrow}|001001110\rangle\,|110\rangle\overset{2}{\Rightarrow}|001001110\rangle\,|110\rangle \\ &\overset{3}{\Rightarrow}|101000110\rangle\,|110\rangle\,. \end{aligned} \tag{17}$$

Performing $P_{2,2^5}$ gives

$$|101000110\rangle\,|110\rangle\overset{4}{\Rightarrow}|101001100\rangle\,|110\rangle\,. \tag{18}$$

Similarly, due to $0011<101$, tells us that the iteration ID cannot change the input, i.e.,

$$\begin{aligned} &|101001100\rangle\,|110\rangle\overset{5}{\Rightarrow}|001101100\rangle\,|110\rangle \\ &\overset{6}{\Rightarrow}|001101100\rangle\,|110\rangle\overset{7}{\Rightarrow}|101001100\rangle\,|110\rangle\,. \end{aligned} \tag{19}$$

After step 8, we have

$$|101001100\rangle\,|110\rangle\overset{8}{\Rightarrow}|101011000\rangle\,|110\rangle\,. \tag{20}$$

For 0110 > 101, the iteration ID gives

$$\begin{aligned}&|101011000\rangle|110\rangle \overset{9}{\Rightarrow} |011100100\rangle|110\rangle\\ &\overset{10}{\Rightarrow} |101001100\rangle|110\rangle \overset{11}{\Rightarrow} |101100100\rangle|110\rangle .\end{aligned} \tag{21}$$

Performing steps 12 and 13, we get the following result:

$$\begin{aligned}&|101100100\rangle|110\rangle \overset{12}{\Rightarrow} |101001100\rangle|110\rangle\\ &\overset{13}{\Rightarrow} |101001001\rangle|110\rangle ,\end{aligned} \tag{22}$$

i.e., the remainder $|R(6)\rangle$ and the quotient $|Q(6)\rangle$ are both $|001\rangle$.

Following the same procedures for $k = 0, 1, 2, 3, 4, 5, 7$, we obtain the vector-integer division output as follows:

$$|\psi_D\rangle = \frac{1}{\sqrt{2^3}}|a\rangle \sum_{k=0}^{7} |R(k)\rangle|Q(k)\rangle|k\rangle, \tag{23}$$

where $R(k)$ and $Q(k)$ are elements of vectors $\mathbf{R}$ and $\mathbf{Q}$ in eq. (13), respectively.

We design the circuits in Figure 24 using the proposed multipliers to realize the vector-integer multiplication in eq. (13).

The outputs in Figure 24(a) and (b) are both

$$|\psi_M\rangle = \frac{1}{\sqrt{2^3}} \sum_{k=0}^{7} |c\rangle |a'\rangle |aV_8(k)\rangle |k\rangle , \tag{24}$$

with $|a'\rangle = |a\rangle$ for $a \neq 0$ and $|a'\rangle = |V_8(k)\rangle$ for $a = 0$, where $V_8(k)$ $(k = 0, 1, \ldots, 7)$ are elements of the vector $\mathbf{V}_8$ in eq. (13).

Finally, we only use a multiplier or a divider to perform the multiplication $V_{2^m} \times a$ or the division V_{2^m}/a, which necessitates 2^m multiplications or divisions by traditional computers. It demonstrates the efficiency of the proposed arithmetic operators.

7 Comparative analysis

7.1 Comparisons of basic arithmetic operators

The performance indices of the proposed basic arithmetic operators are shown in Table 2. Because the modular adder and modular subtractor are conjugate transposes of each other and no measurements are used in the implementation, the costs are, of course, identical. As a result, we leave out the modular subtractor and the controlled modular subtractor in Table 2.

Because modular adders can perform integer complement addition, they are also referred to as adders in refs. [32, 37]. The best T-count for the measure-and-fixup approach is $4n-4$ for the adder. Because the adder in ref. [32] employs measurement, it cannot be directly compared to the T-count.

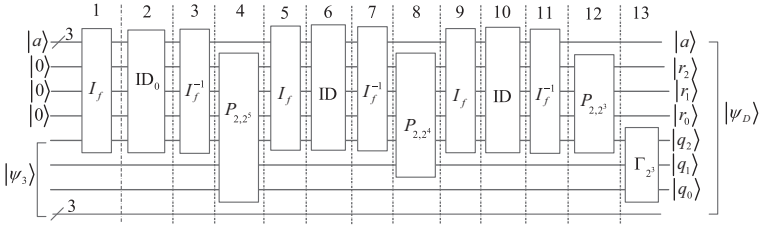

Figure 23 Implementation of the vector-integer division.

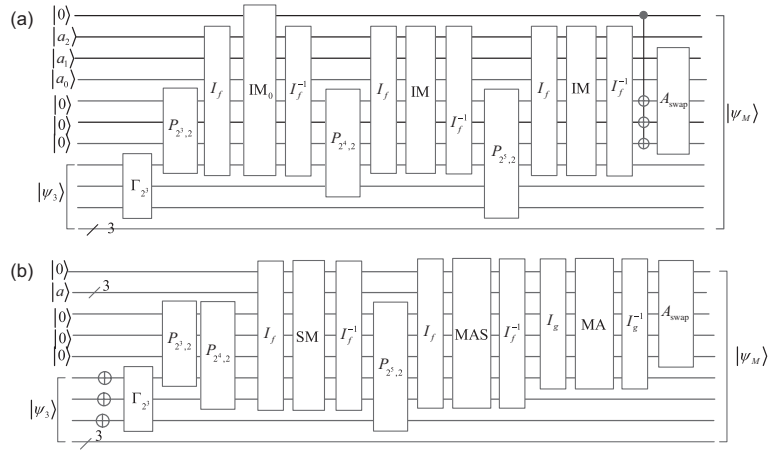

Figure 24 Implementations of the vector-integer multiplication by (a) the first multiplier and (b) the second multiplier.

Table 2 Performance indexes of basic arithmetic operators

Basic arithmetic operators	T-depth	Total depth	T-count	Total count	Width
Adder	$2n+1$	$12n-4$	$8n-4$	$25n+26$	$2n+1$
Modular adder	$2n-1$	$12n-15$	$8n-9$	$25n-36$	$2n$
Comparator	$2n$	$14n-13$	$8n-4$	$26n-12$	$2n+1$
Modular adder-subtractor	$2n-1$	$12n-13$	$8n-9$	$27n-38$	$2n+1$
Controlled modular adder	$3n+1$	$17n-7$	$14n-7$	$40n-27$	$2n+1$

Since the optimizing circuit [37] for the modular adder does not involve the T-depth optimization, we only compare the T-count, total count, and width of the modular adders. According to the results in Table 3, the T-count of the modular adder in ref. [37] is $8n-8$. Take note of the fact that the proposed modular adder has a lower total count than the modular adder in ref. [37].

We compare the proposed works with the rest of the basic arithmetic operators in refs. [21, 34, 35, 43] taking into account the circuit width, T-depth, and T-count. The results are shown in Table 4. From Table 4, we observe that the proposed basic arithmetic operators have lower T-depths and T-counts than the corresponding works in refs. [21, 34, 35, 43].

7.2 Comparisons of multipliers and dividers

The measure-and-fixup method was used to implement previous multipliers with the highest T-count. Gidney, for example, used adders to realize a multiplier with $6n^2 + O(n)$ T gates using the measure-and-fixup method [44]. Similarly, using the logical-AND and uncomputation gates [32], Munoz-Coreas and Thapliyal proposed a multiplier with T-count $8n^2 - 4n$ [45]. However, the two multipliers also necessitate $O(n^2)$ quantum measurements.

We compare the proposed arithmetic operations to best-known works without quantum measurements [21, 39, 43] because we design multipliers and dividers not using quantum measurements. Table 5 displays performance indices and comparison results, comparisons of quantum multiplication and division circuits are given in Tables 6 and 7 by increasing n from 8 to 2048 for clarity. The proposed divider and multipliers outperform the previous works in refs. [21, 39, 43] for the three performance indicators as shown in Tables 5-7. The second multiplier, in particular, reduces T-depth by approximately 77%, T-count by 60%, and width by 25% when compared with the existing works [21, 39].

Note: To implement the positive 2's complement division, two dividers (restoring and non-restoring) were proposed [43]. The complement of a n-bit integer requires $m = n + 1$ bits. As a result, their T-depths and T-counts are $15m^2 - 12m = 15n^2 + 18n + 3$, $35m^2 - 28m = 35n^2 + 42n + 7$, $6m^2 + 3m - 6 = 6n^2 + 9n - 3$, and $14m^2 + 7m - 14 = 14n^2 + 21n - 7$, respectively. In addition, their widths are $3m = 3n + 3$ and $3m - 1 = 3n + 2$.

Table 3 Comparison of modular adders

n	Heavy optimization in ref. [37]			Proposed modular adder		
	T-count	Total count	Width	T-count	Total count	Width
8	56	190	16	55	164	16
16	120	414	32	119	364	32
32	248	862	64	247	764	64
64	504	1758	128	503	1564	128
128	1016	3550	256	1015	3164	256
256	2040	7134	512	2039	6364	512
512	4088	14302	1024	4087	12764	1024
1024	8184	28638	2048	8183	25564	2048
2048	16376	57310	4096	16375	51164	4096

Table 4 Comparisons of basic arithmetic operators

Quantum arithmetic operators		T-depth	T-count	Width
Adder	proposed	$2n+1$	$8n-4$	$2n+1$
	[21]	$6n-3$	$14n-7$	$2n+1$
	[34]	$6n-3$	$14n-7$	$2n+1$
Modular adder	proposed	$2n-1$	$8n-9$	$2n$
	[21]	$6n-9$	$14n-21$	$2n$
	[35]	$6n-6$	$14n-14$	$2n$
Controlled modular adder	proposed	$3n+1$	$14n-7$	$2n+1$
	[21]	$9n-6$	$21n-14$	$2n+1$
	[43]	$9n-6$	$21n-14$	$2n+1$
Comparator	proposed	$2n$	$8n-4$	$2n+1$
	[21]	$6n-3$	$14n-7$	$2n+1$
Modular adder-subtractor	proposed	$2n-1$	$8n-9$	$2n+1$
	[35]	$6n-6$	$14n-14$	$2n+1$

Table 5 Comparisons of multipliers and dividers

Quantum arithmetic operators		T-depth	T-count	Width
Multiplier	proposed 1st	$3n^2+3n$	$14n^2+5n-11$	$3n+1$
	proposed 2nd	$2n^2+2n+2$	$8n^2+7n$	$3n+1$
	[21]	$9n^2-2n-3$	$21n^2-9n-5$	$4n$
	[39]	$9n^2-6$	$21n^2-14$	$4n+1$
Divider	proposed	$3n^2+2n-1$	$14n^2-7n+1$	$3n$
	restoring[43]	$15n^2+18n+3$	$35n^2+42n+7$	$3n+3$
	non-restoring [43]	$6n^2+9n-3$	$14n^2+21n-7$	$3n+2$

Table 6 Comparison of multipliers

n	Multiplier in ref. [39]			Proposed 2nd multiplier			Improvement (%)		
	T-depth	T-count	Width	T-depth	T-count	Width	T-depth	T-count	Width
8	570	1330	33	146	568	25	74.39	57.29	24.24
16	2298	5362	65	546	2160	49	76.24	59.72	24.62
32	9210	21490	129	2114	8416	97	77.05	60.84	24.81
64	36858	86002	257	8322	33216	193	77.42	61.38	24.90
128	147450	344050	513	33026	131968	385	77.60	61.64	24.95
256	589818	1376242	1025	131586	526080	769	77.69	61.77	24.98
512	2359290	5505010	2049	525314	2100736	1537	77.73	61.84	24.99
1024	9437178	22020082	4097	2099202	8395776	3073	77.76	61.87	24.99
2048	37748730	88080370	8193	8392706	33568768	6145	77.77	61.89	25.00
						Average	77.07	60.92	24.83

Table 7 Comparison of dividers

n	Divider in ref. [43]			Proposed divider			Improvement (%)		
	T-depth	T-count	Width	T-depth	T-count	Width	T-depth	T-count	Width
8	453	1057	26	207	841	24	54.30	20.44	7.69
16	1677	3913	50	799	3473	48	52.36	11.24	4.00
32	6429	15001	98	3135	14113	96	51.24	5.92	2.04
64	25149	58681	194	12415	56897	192	50.63	3.04	1.03
128	99453	232057	386	49407	228481	384	50.32	1.54	0.52
256	395517	922873	770	197119	915713	768	50.16	0.78	0.26
512	1577469	3680761	1538	787455	3666433	1536	50.08	0.39	0.13
1024	6300669	14701561	3074	3147775	14672897	3072	50.04	0.20	0.07
2048	25184253	58763257	6146	12587007	58705921	6144	50.02	0.10	0.03
						Average	51.02	4.85	1.75

8 Conclusions and future work

This paper proposed Clifford + T circuits for approximate Toffoli, Peres, TR, and Fredkin gates. Then, using these approximate gates, we designed circuits of basic arithmetic operators with lower T-depths and T-counts. We presented two multipliers and a divider while taking into account the circuit width, T-depth, and T-count. According to the comparative analysis, the proposed multipliers and dividers have the greatest width compared with the existing works. Specifically, the proposed multipliers and dividers require only $3n + 1$ and $3n$ qubits to perform n-bit integer multiplications and divisions. Furthermore, the proposed multipliers and divider have lower T-depth and T-count than previous works that do not use quantum measurements. Furthermore, the proposed multiplier and divider can be used to perform vector-integer operations. Quantum algorithms, such as quantum chemistry, are likely to be implemented in these noisy intermediate-scale quantum (NISQ) devices [51]. The results will be extended and applied in practical information processing in NISQ devices in future works. Analyzing a system's overall performance (e.g., scalability and fault tolerance percentages) in noisy environments will also be an interesting topic.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61762012, 61763014, and 62062035), and the Science and Technology Project of Guangxi (Grant No. 2020GXNSFDA238023).

1 P. W. Shor, SIAM J. Comput. **26**, 1484 (1997).
2 L. K. Grover, Phys. Rev. Lett. **79**, 325 (1997), arXiv: quant-ph/9706033.
3 P. Gao, K. Li, S. Wei, and G. L. Long, Sci. China-Phys. Mech. Astron. **64**, 100311 (2021).
4 C. H. Bennett, and G. Brassard, in *Quantum cryptography: Public key distribution and coin tossing: Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing* (IEEE, New York, 1984), pp. 175-179.
5 C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters, Phys. Rev. Lett. **70**, 1895 (1993).
6 G. L. Long, and X. S. Liu, Phys. Rev. A **65**, 032302 (2002), arXiv: quant-ph/0012056.
7 F. G. Deng, G. L. Long, and X. S. Liu, Phys. Rev. A **68**, 042317 (2003), arXiv: quant-ph/0308173.
8 L. Yang, J. W. Wu, Z. S. Lin, L. G. Yin, and G. L. Long, Sci. China-Phys. Mech. Astron. **63**, 110311 (2020).
9 Y. B. Sheng, L. Zhou, and G. L. Long, Sci. Bull. **67**, 367 (2021).
10 N. Grzesiak, R. Blümel, K. Wright, K. M. Beck, N. C. Pisenti, M. Li, V. Chaplin, J. M. Amini, S. Debnath, J. S. Chen, and Y. Nam, Nat.

Commun. **11**, 2963 (2020), arXiv: 1905.09294.
11 T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, Nature **464**, 45 (2010), arXiv: 1009.2267.
12 M. A. Nielsen, and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).
13 J. A. Smolin, and D. P. DiVincenzo, Phys. Rev. A **53**, 2855 (1996).
14 A. Peres, Phys. Rev. A **32**, 3266 (1985).
15 H. Thapliyal, and N. Ranganathan, *IEEE Computer Society Annual Symposium on VLSI* (IEEE, Tampa, 2009), pp. 229-234.
16 B. Giles, and P. Selinger, Phys. Rev. A **87**, 032332 (2013), arXiv: 1212.0506.
17 V. Kliuchnikov, D. Maslov, and M. Mosca, Phys. Rev. Lett. **110**, 190502 (2013), arXiv: 1212.0822.
18 M. Amy, D. Maslov, M. Mosca, and M. Roetteler, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **32**, 818 (2013).
19 M. Amy, D. Maslov, and M. Mosca, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **33**, 1476 (2014).
20 D. Gosset, V. Kliuchnikov, M. Mosca, and V. Russo, arXiv: 1308.4134.
21 H.-S. Li, P. Fan, H. Xia, H. Peng, and G.-L. Long, Sci. China-Phys. Mech. Astron. **63**, 280311 (2020).
22 P. Selinger, Phys. Rev. A **87**, 042302 (2013), arXiv: 1210.0974.
23 C. Jones, arXiv: 1709.06648v3.
24 A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Phys. Rev. A **52**, 3457 (1995), arXiv: quant-ph/9503016.
25 M. K. Thomsen, R. Glück, and H. B. Axelsen, J. Phys. A-Math. Theor. **43**, 382002 (2013).
26 W. Van Dam, and I. E. Shparlinski, in *Workshop on Quantum Computation, Communication, and Cryptography* (Springer, Berlin, Heidelberg, 2008), pp. 1-10.
27 V. Vedral, A. Barenco, and A. Ekert, Phys. Rev. A **54**, 147 (1996), arXiv: quant-ph/9511018.
28 T. G. Draper, S. A. Kutin, E. M.Rains, and K. M. Svore, arXiv: quant-ph/0406142.
29 Y. Takahashi, and N. Kunihiro, Quantum Inf. Comput. **8**, 636 (2008).
30 Y. Takahashi, S. Tani, and N. Kunihiro, arXiv: 0910.2530.
31 S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, arXiv: quant-ph/0410184.
32 C. Gidney, arXiv: 1212.5069v1.
33 H. Thapliyal, E. Munoz-Coreas, and V. Khalus, Sustain. Comput. Inf. Sys. **29**, 100457 (2021).
34 H. Thapliyal, and N. Ranganathan, ACM J. Emerg. Tech. Com. **9**, 17 (2013).
35 H. Thapliyal, *Transactions on Computational Science XXVII* (Springer, Berlin, 2016), pp. 16-34.
36 H. Thapliyal, T. S. S. Varun, and E. Munoz-Coreas, arXiv: 1609.01241.
37 Y. Nam, N. J. Ross, Y. Su, A. M. Childs, and D. Maslov, npj Quantum Inf. **4**, 23 (2018), arXiv: 1710.07345.
38 H. V. Jayashree, H. Thapliyal, H. R. Arabnia, and V. K. Agrawal, J. Supercomput. **72**, 1477 (2016).
39 E. Munoz-Coreas, and H. Thapliyal, IEEE Trans. Comput. **68**, 729 (2019).
40 A. Barenco, A. Ekert, K. A. Suominen, and P. Törmä, Phys. Rev. A **54**, 139 (1996), arXiv: quant-ph/9601018.
41 H. S. Li, P. Fan, H. Xia, S. Song, and X. He, Quantum Inf. Process. **17**, 333 (2018).
42 A. Khosropour, H. Aghababa, and B. Forouzandeh, *IEEE Eighth International Conference on Information Technology: New Generations* (IEEE, Las Vegas, 2011), pp. 1037-1040.
43 H. Thapliyal, E. Munoz-Coreas, T. S. S. Varun, and T. S. Humble, IEEE Trans. Emerg. Top. Comput. **9**, 1045 (2021).
44 C. Gidney, https://algassert.com/post/1709 (2017).
45 E. Munoz-Coreas, and H. Thapliyal, in *Proceedings of 9th IEEE International Conference on Computer Vision* (IEEE, Pittsburgh, 2018), pp. 212-219.
46 D. Maslov, Phys. Rev. A **93**, 022311 (2016), arXiv: 1508.03273.
47 H. S. Li, P. Fan, H. Y. Xia, H. Peng, and S. Song, IEEE Trans. Circuits Syst. I **66**, 341 (2019).
48 F. Yan, A. M. Iliyasu, Y. Guo, and H. Yang, Theor. Comput. Sci. **752**, 71 (2018).
49 C. Y. Pang, R. G. Zhou, B. Q. Hu, W. W. Hu, and A. El-Rafei, Inf. Sci. **473**, 121 (2019).
50 H. S. Li, P. Fan, H. Peng, S. Song, and G. L. Long, IEEE Trans. Cybern. 1 (2021).
51 S. Wei, H. Li, and G. L. Long, Research, **2020**, 1486935 (2020).