Fan and Li *EPJ Quantum Technology* (2024) 11:13
https://doi.org/10.1140/epjqt/s40507-024-00222-4

EPJ Quantum Technology
a SpringerOpen Journal

EPJ.org

RESEARCH Open Access

Designs of the divider and special multiplier optimizing T and CNOT gates

Ping Fan[1] and Hai-Sheng Li[2*]

*Correspondence:
lhs1974@mailbox.gxnu.edu.cn
[2]College of Electronic and Information Engineering, Guangxi Normal University, Guilin, 541004, Guangxi, China
Full list of author information is available at the end of the article

Abstract

Quantum circuits for multiplication and division are necessary for scientific computing on quantum computers. Clifford + T circuits are widely used in fault-tolerant realizations. T gates are more expensive than other gates in Clifford + T circuits. But neglecting the cost of CNOT gates may lead to a significant underestimation. Moreover, the small number of qubits available in existing quantum devices is another constraint on quantum circuits. As a result, reducing T-count, T-depth, CNOT-count, CNOT-depth, and circuit width has become the important optimization goal. We use 3-bit Hermitian gates to design basic arithmetic operations. Then, we present a special multiplier and a divider using basic arithmetic operations, where 'special' means that one of the two operands of multiplication is non-zero. Next, we use new rules to optimize the Clifford + T circuits of the special multiplier and divider in terms of T-count, T-depth, CNOT-count, CNOT-depth, and circuit width. Comparative analysis shows that the proposed multiplier and divider have lower T-count, T-depth, CNOT-count, and CNOT-depth than the current works. For instance, the proposed 32-bit divider achieves improvement ratios of 40.41 percent, 31.64 percent, 45.27 percent, and 65.93 percent in terms of T-count, T-depth, CNOT-count, and CNOT-depth compared to the best current work. Further, the circuit widths of the proposed n-bit multiplier and divider are $3n$. I.e., our multiplier and divider reach the minimum width of multipliers and dividers, keeping an operand unchanged.

Keywords: Quantum multiplier; Quantum divider; Quantum arithmetic operators; Clifford + T circuits

1 Introduction

One of the most significant challenges in quantum computing is the realization of quantum computers [1]. The quantum circuit model is a realistic quantum computer model [2] and promotes the efficient implementation of quantum algorithms such as quantum image processing [3, 4], quantum transforms [5, 6], and quantum amplitude estimation [7].

Quantum circuits for arithmetic operations as the vital part of a quantum computer's reversible arithmetic logic unit can be realized by quantum gates [8, 9]. For instance, Nooralahzadeh et al. used elementary quantum gates in the NCV (NOT, CNOT, Controlled-V, and Controlled-V+) library to design quantum multipliers [10–12]. These multipliers

Springer

have the low quantum cost, garbage output, and constant input. The fault-tolerant implementation of quantum gates is needed for robust quantum computing in the presence of noise. Clifford + T circuits are widely accepted solutions to fault-tolerant implementation [13, 14]. An instruction set $\{H, S, S^\dagger, CNOT, T, T^\dagger\}$ can be used to implement quantum gates [15].

The T gates are more expensive than other gates in terms of space and time cost due to their increased tolerance to noise errors [16, 17]. Since the CNOT gate is the only double-qubit gate in the Clifford + T gate set, neglecting the cost of the CNOT gates may lead to a significant underestimation [18, 19]. Therefore, the number of T gates (T-count), the maximum number of T gates in any circuit path (T-depth), CNOT-count, and CNOT-depth are the main performance indicators of Clifford + T circuits.

Toffoli, Fredkin, Peres, and TR gates are typical for the design of quantum arithmetic operations [20–22]. Clifford + T circuits with T-depth 3 and T-count 7 for Toffoli, Fredkin, Peres, and TR gates have been proposed without ancillae [17, 18, 23]. Their CNOT-counts are 7, 9, 6, and 6, respectively. Compared with Peres and TR gates, the Hermitian Toffoli gate has a better symmetry performance. Four 3-bit Hermitian gates were presented in [24], whose T-depth, T-count, and CNOT-count are 3, 7, and 6. Therefore, these Hermitian gates in [24] have better symmetry than Peres or TR gates and the smaller CNOT-count than the Toffoli gate. So, we use these Hermitian gates to design the divider and special multiplier in this paper.

In the past decade, quantum circuit designs for arithmetic operators, including adders, modular adders, and comparators, were actively studied [23, 25–33]. Some arithmetic operations have $O(\log n)$ T-depth but require lots of ancillae. For example, an n-bit adder with the minimal T-count $4n - 4$ was proposed in [30]. But the adder needs $2n - 2$ measurements and $n - 1$ ancillae. In addition, arithmetic operators with the minimum circuit width (0 ancillae) were designed in [23, 32].

Thapliyal et al. used ancillae to realize n-bit multipliers with two unchanged operands, so their circuit width is $4n + 1$ [34, 35]. To reduce the T-count and circuit width, Li et al. utilized Peres and TR gates to design the multiplier with two unchanged operands and the circuit width $4n$ [23]. Two multipliers with an unchanged operand and the circuit width $3n + 1$ were implemented using approximate Toffoli, Peres, and TR gates [36]. They have smaller T-counts, T-depths, and circuit widths than other multipliers proposed in [23, 34, 35]. A quantum divider based on quantum Fourier transform was designed with the circuit width $4n$ [37]. Thapliyal et al. replaced Toffoli gates with the complex quantum Fourier transform to realize two divisions with fewer qubits [38]. The two dividers have circuit widths $3n + 3$ and $3n + 2$. To further reduce circuit width, Li et al. designed a divider with the circuit width $3n$ [36]. The above multipliers and dividers do not consider optimizing CNOT-count and CNOT-depth.

Quantum algorithms may likely be implemented in these noisy intermediate-scale quantum (NISQ) devices [39]. Some algorithms, such as a quantum convolutional neural network, have been proposed for NISQ devices [40, 41]. The large circuit width blocks applications of algorithms in NISQ devices, so the small circuit width is crucial for algorithms applied in NISQ devices. Since an n-bit divider with an unchanged operand needs at least $3n$ qubits to store the n-bit operand and $2n$-bit result, we will use $3n$ qubits to realize a multiplier and a divider, respectively. In this paper, we design some basic arithmetic operations such as the modular adder and controlled modular adder. Then, we propose a

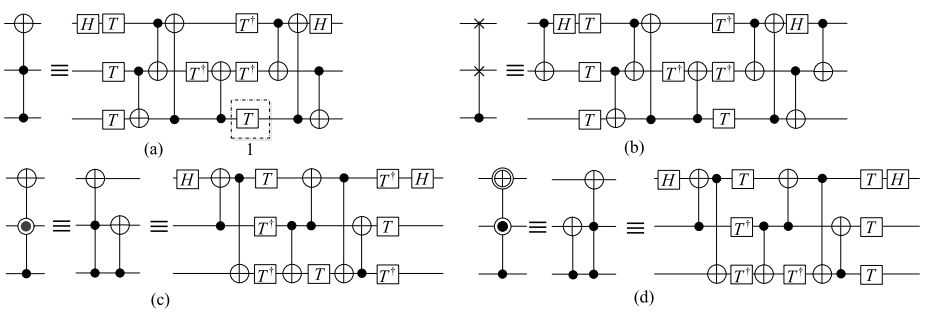

Figure 1 Implementation circuits for (a) the Toffoli gate, (b) the Fredkin gate, (c) the Peres gate, and (d) an inverse-Peres gate named TR. Note: The circuit of the Toffoli gate has an error in [36], so we have modified the error in the dashed box 1 in (a)

special multiplier where 'special' means that one of the two operands of multiplication is not equal to zero, such as the multiplication $s = a \times b$ with $a \neq 0$. Since the division s/a is the inverse operation of the special multiplication $s = a \times b$, we can obtain a divider circuit modifying the proposed special multiplier. Finally, the optimized Clifford + T circuits of the divider and special multiplier are presented.

The rest of this paper is organized as follows. In Sect. 2, we review the background knowledge. Section 3 presents basic arithmetic operations. In Sects. 4 and 5, we propose the special multiplier and divider. Comparative analysis and conclusions are drawn in Sects. 6 and 7.

2 Background

For clarity, we briefly introduce 3-bit Hermitian gates in [24] and approximate Toffoli gates in [36]. The matrix forms of six Clifford + T gates are defined by

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

We use the instruction set $\{H, X, S, S^\dagger, CNOT, T, T^\dagger\}$ to realize the Clifford + T circuits for arithmetic operations in this paper. For instance, the implementations for the Toffoli, Fredkin, Peres, and TR gates are illustrated in Fig. 1 [15, 36].

Figure 1 reveals that Peres and TR gates consist of a Toffoli gate and a CNOT gate, respectively. Similarly, four Hermitian gates are constructed with Toffoli and CNOT gates [24]. The four Hermitian gates are denoted as LI1, LI2, LI3, and LI4 with LI = {LI1, LI2, LI3, LI4}. Their Clifford + T circuits are presented in Fig. 2.

Figure 2 reveals that LI1 and LI2 gates are essentially the same. Since we swap the lines of operands A and B, we can get LI from LI2 swaping the lines of operands A and B. Similarly, we obtain the gate in Fig. 2(e) by swapping the lines of operands C and B for the LI2 gate.

The optimization rule for two LI2 gates is illustrated in Fig. 3.

The implementations for the Toffoli and approximate Toffoli gates are shown in Fig. 4. Compared to the corresponding Toffoli gate in Fig. 4(a), the approximate Tof-

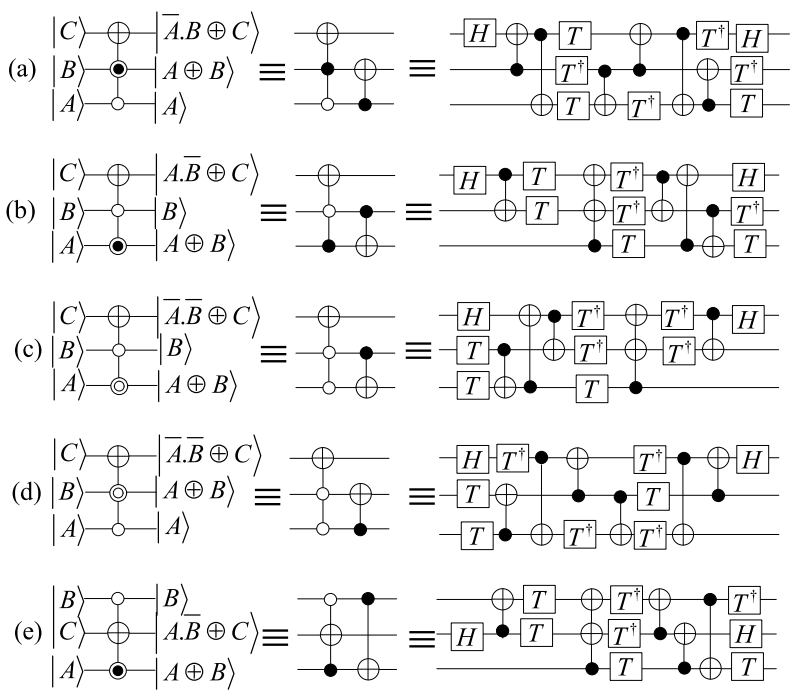

Figure 2 Clifford + T circuits for (a) the LI1 gate, (b) the LI2 gate, (c) the LI3 gate, (d) the LI4 gate, and (e) the variant of the LI2 gate

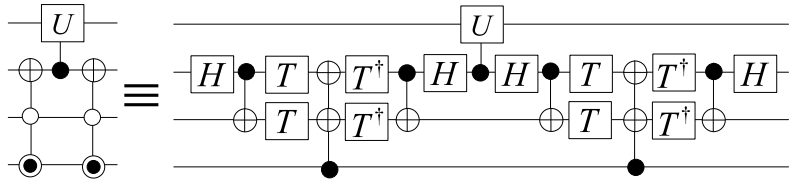

Figure 3 Rule 1 of two LI gates proposed in [36]. *U* is the combination of LI and Clifford gates

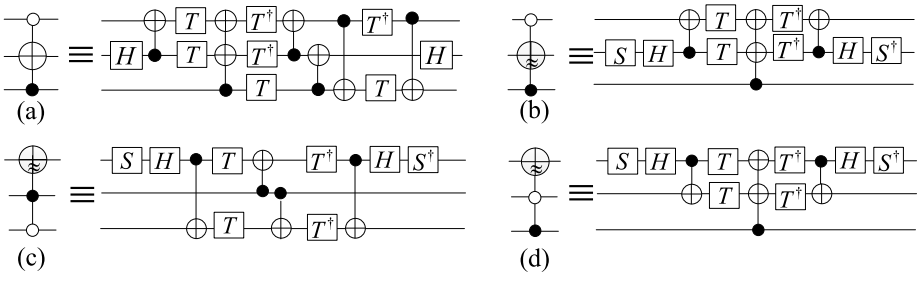

Figure 4 Clifford + T circuits for (a) the Toffoli gate, (b) the approximate Toffoli gate, and the variants of the approximate Toffoli gate in (c) and (d)

foli gate in Fig. 4(b) differs in that it maps $|111\rangle$ to $-|111\rangle$ [36]. In addition, we give Clifford + T circuits for two variants of the approximating Toffoli gate in Fig. 4(c) and (d).

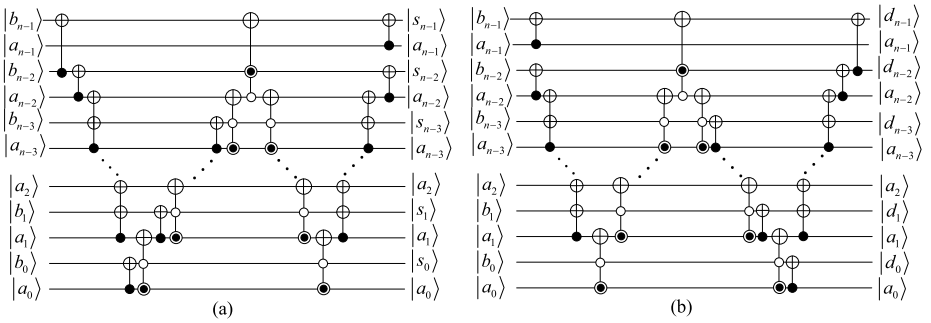

Figure 5 Quantum circuits for (a) the modular adder (MA) and (b) the modular subtractor (MS)

3 Designs of basic arithmetic operations

This section gives Clifford + T circuits for some basic arithmetic operations, including the modular adder (MA), modular subtractor (MS), modular adder-subtractor (MAS), modular subtractor-adder (MSA), controlled modular adder (CMA), controlled modular subtractor (CMS), special modular adder (SMA), special modular subtractor (SMS), special modular adder-subtractor (SMAS), and special modular subtractor-adder (SMSA). MS, MSA, CMS, SMS, and SMSA are inverses of MA, MAS, CMA, SMA, and SMAS, respectively.

3.1 Quantum circuits of basic arithmetic operations

We substitute LI2 gates for Peres and TR gates to realize the modular adder and subtractor in [23]. Their circuits presented in Fig. 5 implement the following n-bit operations:

$$\begin{cases} |s\rangle = |(a+b) \bmod 2^n\rangle, \\ |d\rangle = |(b-a) \bmod 2^n\rangle, \end{cases} \tag{1}$$

with $|b\rangle = |b_{n-1}b_{n-2}\dots b_0\rangle$, $|a\rangle = |a_{n-1}a_{n-2}\dots a_0\rangle$, $|s\rangle = |s_{n-1}\dots s_0\rangle$, $|d\rangle = |d_{n-1}\dots d_0\rangle$, b_k, a_k, $s_k, d_k \in \{0,1\}$, and $k \in \{0,1,\dots,n-1\}$.

Compared to MA, MS consists of the same gates with the inverted order. It reveals that the inverse of basic arithmetic operations based on LI gates can be realized by inverting the circuit order of the corresponding arithmetic operations. We use LI2 gates to design quantum circuits for the modular adder-subtractor (MAS) and the controlled modular adder (CMA) in Fig. 6, which implement

$$\begin{cases} h = (b + (-1)^e a) \bmod 2^n, \\ t = (b + \bar{e}a) \bmod 2^n, \end{cases} \tag{2}$$

with $|h\rangle = |h_{n-1}\dots h_0\rangle$, $|t\rangle = |t_{n-1}\dots t_0\rangle$, $h_k, t_k, e \in \{0,1\}$, and $k \in \{0,1,\dots,n-1\}$.

When $a_{n-1}a_{n-2}\dots a_1a_0$ is equal to $0a_{n-2}\dots a_1a_0$ for MA and MAS in Fig. 5 and Fig. 6, we can omit the most significant bit of $0a_{n-2}\dots a_1a_0$ and design the special modular adder (SMA) and special modular adder-subtractor (SMAS) to reduce circuit width. For instance, SMA can realize $|(b_{n-1}b_{n-2}\dots b_0 + a_{n-2}\dots a_0) \bmod 2^n\rangle$ with $2n-1$ qubits. Their 4-bit examples are proposed in Fig. 7.

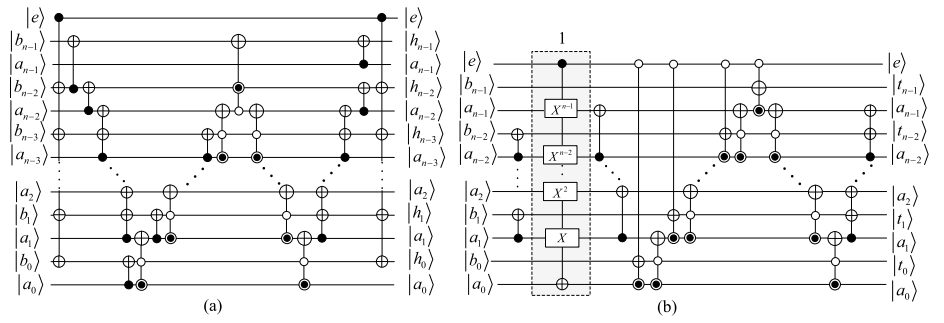

Figure 6 Quantum circuits for (a) the modular adder-subtractor (MAS) and (b) the controlled modular adder (CMA). Due to $X^n = I$ for the even number n and $X^n = X$ for the odd number n, the circuit in box 1 consists of $\lceil n/2 \rceil + 1$ CONT gates, where $\lceil . \rceil$ is a round-up symbol

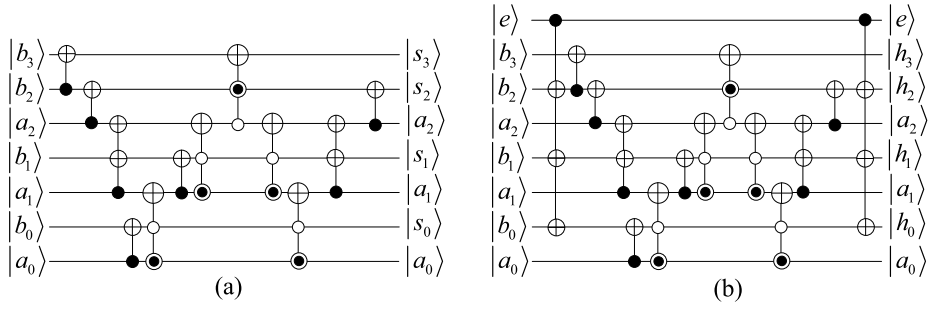

Figure 7 Quantum circuits for (a) the special modular adder (SMA) and (b) the special modular adder-subtractor (SMAS)

Compared to basic arithmetic operations based on Peres and TR gates in [23], the above arithmetic operations have an advantage: these arithmetic operations and their inverse can be realized using the same gates. Compared to basic arithmetic operations based on Toffoli gates in [32, 36, 38], the above arithmetic operations have fewer CNOT-count. For instance, the n-bit CMA in Fig. 6(b) can reduce approximately $2.5n$ CONT gates compared with the controlled modular adders based on Toffoli gates.

3.2 Optimized Clifford + T circuits for basic arithmetic operations

We present four rules for LI gates in Fig. 8 to optimize Clifford + T circuits. T-count, T-depth, CNOT-count, and CNOT-depth are selected as optimization goals. Rule 2 is given in Fig. 8(a) to optimize CNOT gates. Rules 3 and 4 are obtained by modifying rule 1 in Fig. 3. Rule 5 shows that the Clifford + T circuit in box 2 is for the circuit in box 1. Furthermore, for each iteration of the circuit in box 2, the gate in the box will be multiplied by $T^\dagger$. Therefore, the gate in box 3 becomes $(T^\dagger)^n$ after $n + 2$ times iterate the circuit in box 2. We obtain $(T^\dagger)^n \in \{T^\dagger, S^\dagger, S^\dagger T^\dagger, Z, ZT^\dagger, ZS^\dagger, T, I\}$ for any integer n using $(T^\dagger)^2 = S^\dagger$, $(T^\dagger)^3 = S^\dagger T^\dagger$, $(T^\dagger)^4 = Z$, $(T^\dagger)^5 = ZT^\dagger$, $(T^\dagger)^6 = ZS^\dagger$, $(T^\dagger)^7 = T$, and $(T^\dagger)^8 = I$, where the matrix forms of the Clifford gates I and Z are $I = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$, $Z = \left[\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right]$.

We can use rules 3, 4, and 5 to optimize Clifford + T circuits for basic arithmetic operations. For instance, the optimized Clifford + T circuits for the 4-bit SMA,4-bit SMSA, and 3-bit CMA are presented in Fig. 9 and Fig. 10.

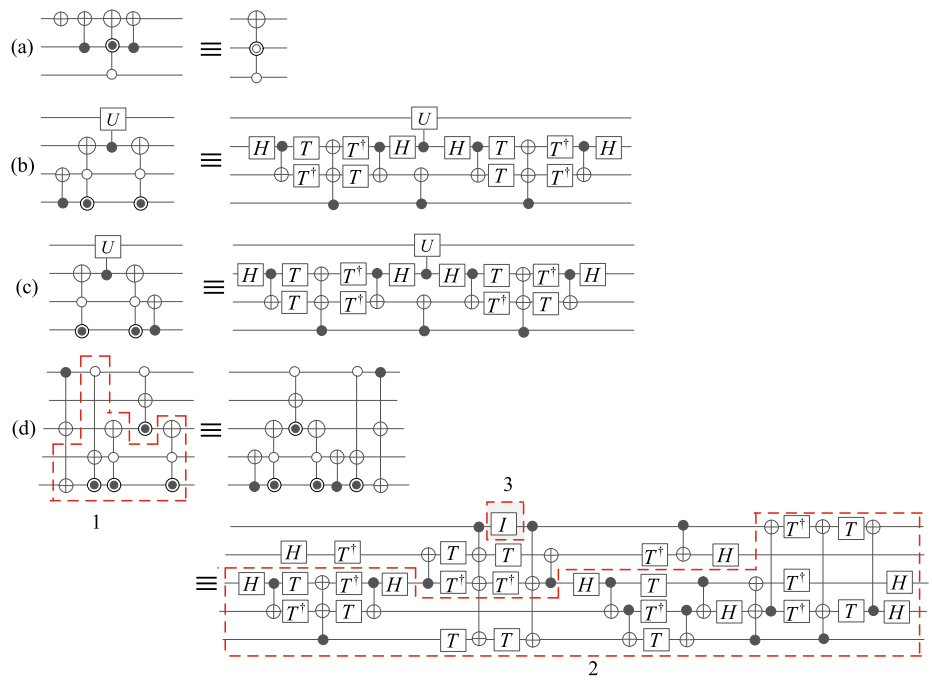

Figure 8 Three rules for LI gates including (a) rule 2, (b) rule 3, (c) rule 4, and (d) rule 5

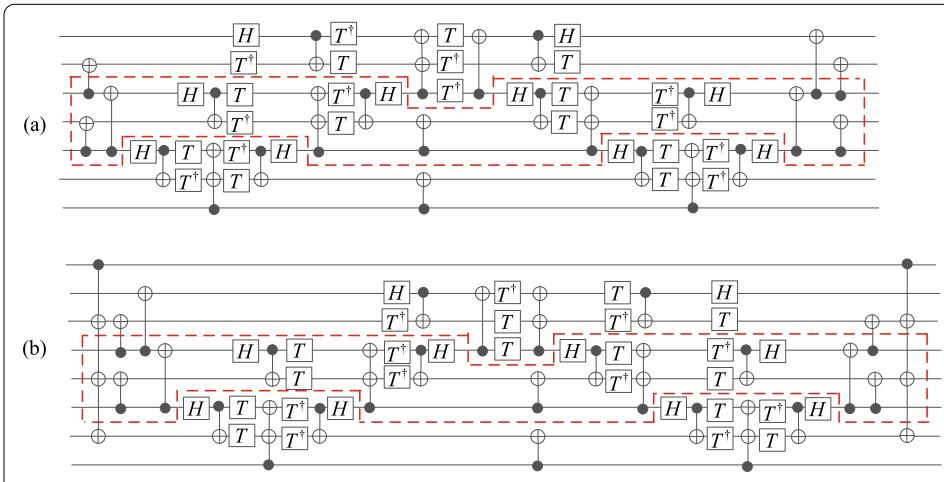

Figure 9 Optimized Clifford + T circuits for the 4-bit SMA and SMSA

Circuits in dashed boxes are iterative circuits of SMA, SMSA, and CMA. Increasing by 1-bit for SMA will increase 8 T-counts, 2 T-depths, 13 CNOT-counts, and 6 CNOT-depths. Therefore, the n-bit SMA has $8(n-3)+8+7 = 8n-9$ T-counts, $2(n-3)+5 = 2n-1$ T-depth, $13(n-3) + 17 = 13n - 22$ CNOT-counts, and $6(n-3) + 10 = 6n - 8$ CNOT-depths. Similarly, we can calculate performance indexes for other basic arithmetic operations listed in Table 1.

4 The design of the special multiplier

In this section, we design a special multiplier using LI and approximate Toffoli gates.

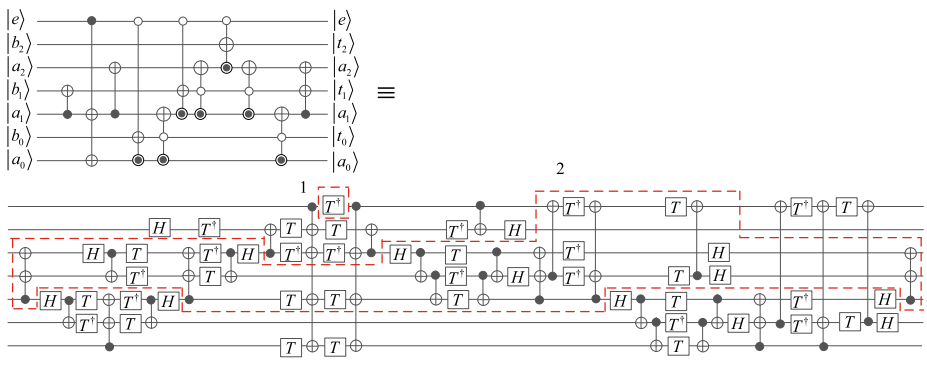

Figure 10 The 3-bit CMA and its optimized Clifford + T circuit

Table 1 Performance indexes for n-bit basic arithmetic operations. HS-count denotes the total number of H and S gates

Operations	T-count	T-depth	CNOT-count	CNOT-depth	HS-count	Width
MA, MS	$8n-9$	$2n-1$	$13n-21$	$6n-7$	$4n-6$	$2n$
SMA, SMS	$8n-9$	$2n-1$	$13n-22$	$6n-8$	$4n-6$	$2n-1$
MAS, MSA	$8n-9$	$2n-1$	$15n-23$	$6n-5$	$4n-6$	$2n+1$
SMAS, SMSA	$8n-9$	$2n-1$	$15n-24$	$6n-6$	$4n-6$	$2n$
CMA, CMS	$\leq 14n-7$	$3n+1$	$20n-18$	$12n-10$	$\leq 6n-3$	$2n+1$

4.1 The circuit for the special multiplier optimizing CNOT gates

The special multiplication $s = ab$ can be expressed by

$$s = ba = \sum_{k=0}^{n-1} b_k 2^k a \tag{3}$$

with $|b\rangle = |b_{n-1}b_{n-2}\ldots b_0\rangle$, $|a\rangle = |0a_{n-1}a_{n-2}\ldots a_0\rangle$, and $a \neq 0$.

Since $a + b = (a + b) \bmod 2^{n+1}$ holds for any two n-bit positive integers, an n-bit addition can be realized by $(n+1)$-bit modular adders. It is the reason that the n-bit integer a in (3) is expressed as $0a_{n-1}a_{n-2}\ldots a_0$.

Equation (3) is rewritten as

$$s = -\overline{b_0}a - \sum_{k=1}^{n-1} (-1)^{b_k} 2^{k-1} a + 2^{n-1}a \tag{4}$$

with $|b\rangle = |b_{n-1}b_{n-2}\ldots b_0\rangle$, $|a\rangle = |0a_{n-1}a_{n-2}\ldots a_0\rangle$, $\overline{b_0} = 1 - b_0$, and $a \neq 0$.

Special arithmetic operations in Fig. 7 and their inverses can be adopted for the operand $0a_{n-1}a_{n-2}\ldots a_0$. I.e., we can omit the most significant bit of $0a_{n-1}a_{n-2}\ldots a_0$ using special arithmetic operations. Thus, we can use a CMS, $(n-1)$ SMSA, and an SMA to realize the special multiplication in (4). The circuits for the 2-bit and 3-bit special multipliers are shown in Fig. 11. Swap gates between basic arithmetic operations are used to shift quantum lines. The output $|s_5s_4s_3s_2s_1s_0\rangle$ equals to $|ba\rangle$.

The CMS in Fig. 11 perfects the operation $(0 - \bar{b}_0 a) \bmod 2^n$, which can be rewritten by

$$(0 - \bar{b}_0 a) \bmod 2^n = (b_0 a - a) \bmod 2^n, \tag{5}$$

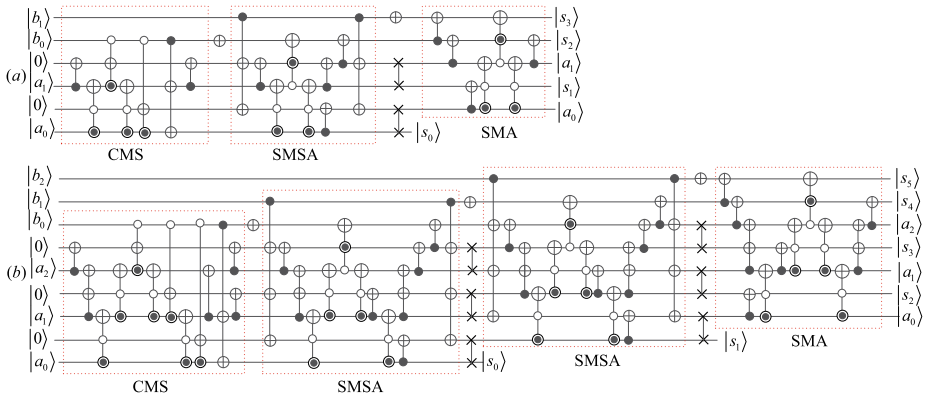

Figure 11 The circuits for (a) a 2-bit special multiplier and (b) a 3-bit special multiplier. The outputs $|s_3s_2s_1s_0\rangle$ and $|s_5s_4s_3s_2s_1s_0\rangle$ equal to $|ba\rangle$, respectively

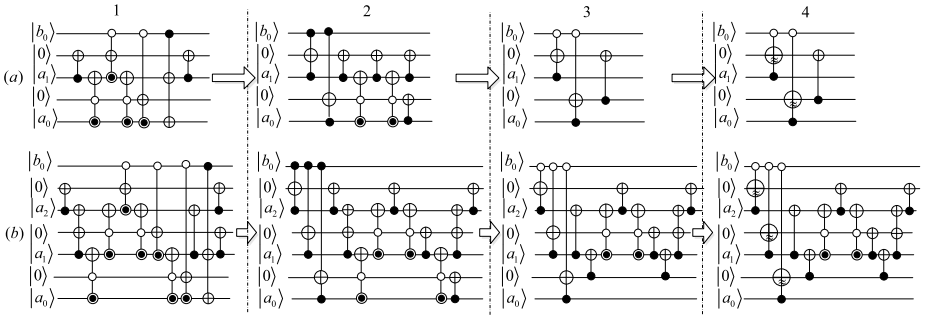

Figure 12 The circuits for the operation $(0-\bar{b}_0a)$ mod 2^n with (a) $n=2$ and (b) $n=3$

with $a = a_{n-1}\ldots a_1a_0$ and $a_{n-1},\ldots,a_1,a_0,b_0 \in \{0,1\}$. The circuits for the operation $(0-\bar{b}_0a)$ mod 2^n with $n=2,3$ are presented in Fig. 12. We replace CMS in the first column with circuits in the second column to realize (5). Since the LI2 gate changes $|a_1\rangle|0\rangle|a_0\rangle$ into $|a_1\oplus a_0\rangle|0\rangle|a_0\rangle$, we substitute CNOT gates for LI2 gates to obtain circuits in the third column. Finally, we give circuits based on LI and approximate Toffoli gates in the fourth column. The circuit for $(0-\bar{b}_0a)$ mod 2^n is presented in Fig. 13.

We give circuits in the left in Fig. 14(b) and the top in Fig. 14(c) by shifting lines directly to replace Swap gates (see Fig. 14(a)), substituting circuits in Fig. 12 for CMS and eliminating some CNOT gates. Then, we obtain the circuits optimizing CNOT gates in Fig. 14 for the 2-bit and 3-bit special multipliers using rule 3.

For clarity, we give the circuit for the 4-bit special multiplier in Fig. 15(a). The circuits in the dashed boxes 1, 2, and 3 are named the first module of the multiplier (FMM), the iterative module of the multiplier (IMM), and the last module of the multiplier (LMM), respectively. Next, we use these modules to design the n-bit special multiplier in Fig. 15(b).

4.2 Clifford + T circuits for the special multiplier

We give the Clifford + T circuit for the 2-bit special multiplier in Fig. 16 using rules 3 and 4. The circuit in dashed box 1 is provided using the Clifford + T circuit for the approximate

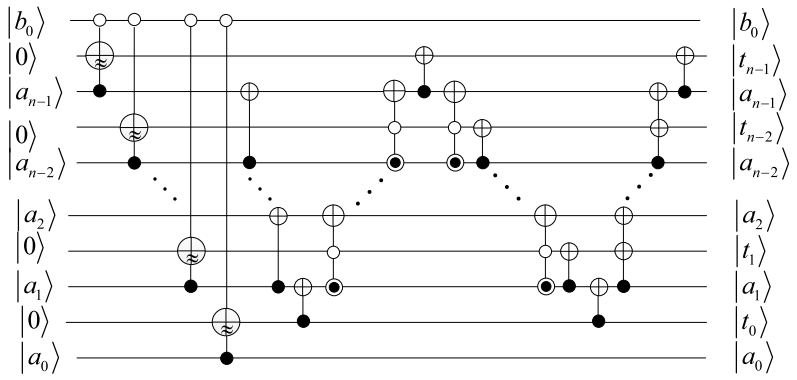

Figure 13 The n-bit circuit for the operation in (5). $|t_{n-1}\ldots t_1 t_0\rangle$ equals to $|(b_0 a - a) \bmod 2^n\rangle$

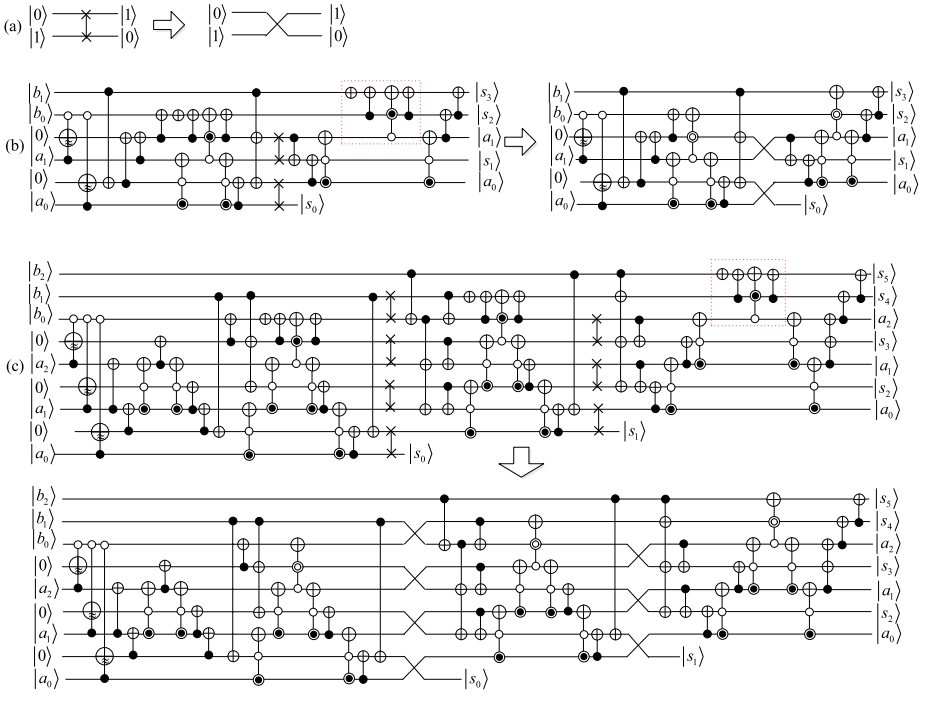

Figure 14 The circuits optimizing CNOT gates for (a) shifting quantum lines to replace the Swap gate, (b) the 2-bit special multiplier and (c) the 3-bit special multiplier. Rule 3 shows that circuits in dashed boxes are equivalent to LI4 gates

Toffoli gate in Fig. 4(b). The circuit in dashed box 2 is another implementation based on Clifford + T gates for the LI4 gate.

Using rule 4, we present the Clifford + T circuit of FMM for the 4-bit special multiplier in Fig. 17. The iterative circuit of FMM is given in the dashed box. Figure 17 reveals that increasing by 1-bit for FMM increases 18 T-counts, 4 T-depths, 21 CNOT-count, and 9 CNOT-depth. Therefore, we obtain performance indexes of the FMM for the n-bit special

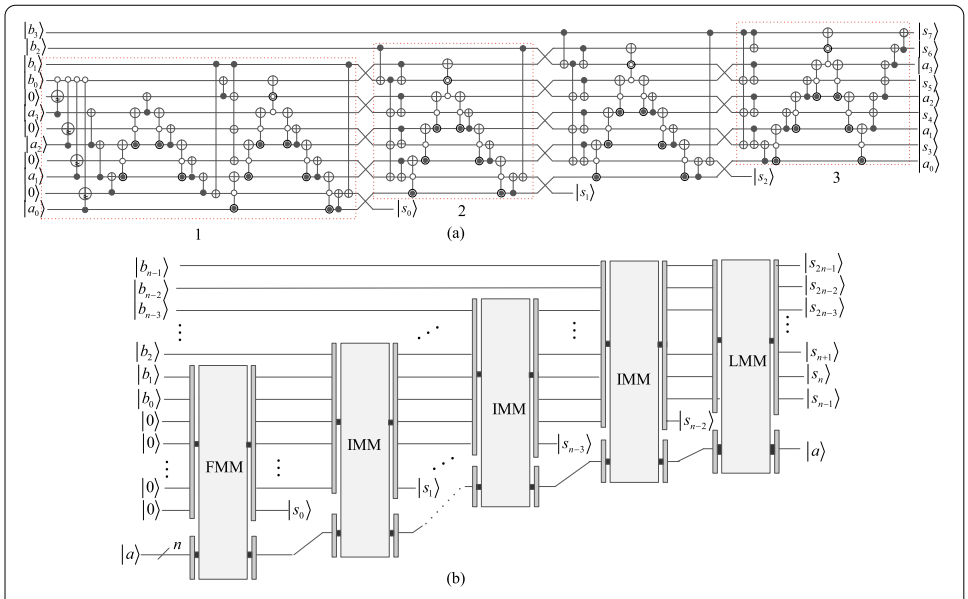

Figure 15 The circuits optimizing CNOT gates for (a) the 4-bit special multiplier and (b) the *n*-bit special multiplier

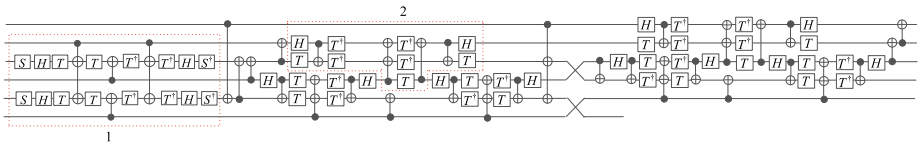

Figure 16 The Clifford + T circuit for the 2-bit special multiplier

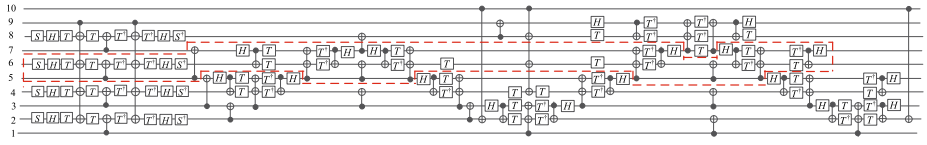

Figure 17 The Clifford + T circuit of FMM for the 4-bit special multiplier

multiplier by calculating

$$\begin{cases} \text{FMM's T-count: } 18(n-2)+23 = 18n-13, \\ \text{FMM's T-depth: } 4(n-2)+10 = 4n+2, \\ \text{FMM's CNOT-count: } 21(n-2)+27 = 21n-15, \\ \text{FMM's CNOT-depth: } 9(n-2)+16 = 9n-2. \end{cases} \tag{6}$$

We propose Clifford + T circuits of the IMM and LMM for the 4-bit special multiplier in Fig. 18 using rules 3 and 4. Analyzing iterative circuits in dashed boxes, we calculate the

Table 2 Performance indexes the special multiplier and its modules with $n \geq 3$

Operations	T-count	T-depth	CNOT-count	CNOT-depth	H-count	S-count	Width
FMM	$18n-13$	$4n+2$	$21n-15$	$9n-2$	$8n-6$	$2n$	$2n+2$
IMM	$8n-1$	$2n+1$	$11n-3$	$4n+4$	$4n-2$	0	$2n+2$
LMM	$8n-1$	$2n+1$	$13n-7$	$5n+2$	$4n-2$	0	$2n+1$
2-bit multiplier	38	14	43	26	16	4	6
n-bit Multiplier	$8n^2+9n-12$	$2n^2+3n+1$	$11n^2+9n-16$	$4n^2+10n-8$	$4n^2+2n-4$	$2n$	$3n$

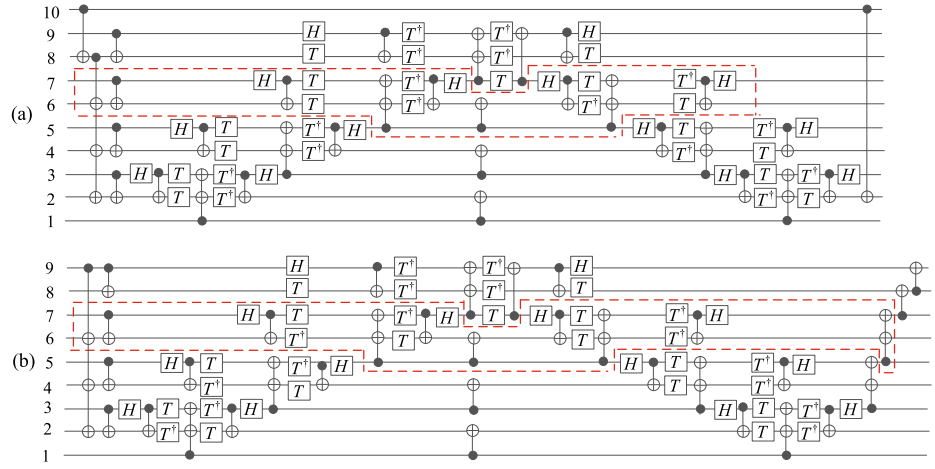

Figure 18 Clifford + T circuits of IMM and LMM for the 4-bit special multiplier including (a) IMM and (b) LMM

performance indexes of the IMM and LMM as follows:

$$\begin{cases} \text{IMM's T-count: } 8(n-2)+15 = 8n-1, \\ \text{IMM's T-depth: } 2(n-2)+5 = 2n+1, \\ \text{IMM's CNOT-count: } 11(n-2)+19 = 11n-3, \\ \text{IMM's CNOT-depth: } 4(n-2)+12 = 4n+4, \end{cases} \tag{7}$$

and

$$\begin{cases} \text{LMM's T-count: } 8(n-2)+15 = 8n-1, \\ \text{LMM's T-depth: } 2(n-2)+5 = 2n+1, \\ \text{LMM's CNOT-count: } 13(n-2)+19 = 13n-7, \\ \text{LMM's CNOT-depth: } 5(n-2)+12 = 5n+2. \end{cases} \tag{8}$$

For clarity, we give performance indexes of the 2-bit special multiplier, n-bit special multiplier ($n \geq 3$), and its modules in Table 2.

5 The design of the divider

In this section, we design a divider using LI and approximate Toffoli gates.

5.1 The circuit for the divider optimizing CNOT gates

The division s/a is the inverse operation of the special multiplication $s = ba$ with $a \neq 0$. We obtain a divider by reversing the circuit order for the special multiplier in Fig. 11. There-

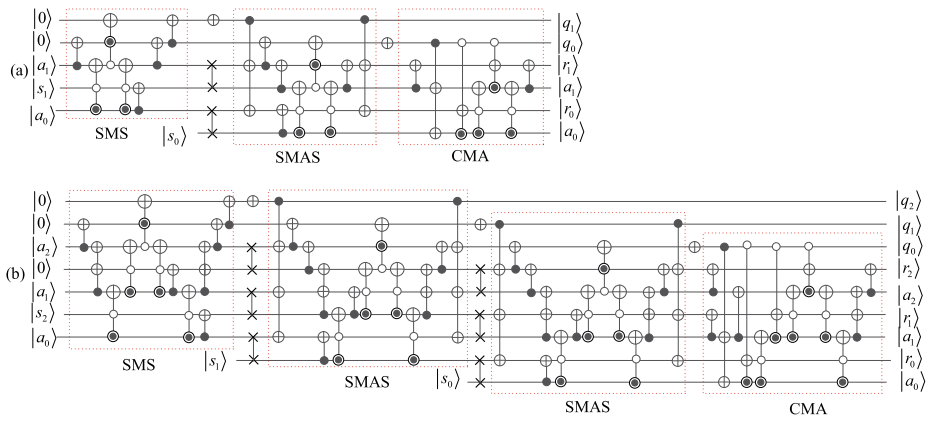

Figure 19 The circuits for (a) a 2-bit divider and (b) a 3-bit divider. The outputs $|q_1q_0\rangle$ and $|q_2q_1q_0\rangle$ are quotients of the division $|s/a\rangle$; $|r_1r_0\rangle$ and $|r_2r_1r_0\rangle$ are remainders of the division $|s/a\rangle$

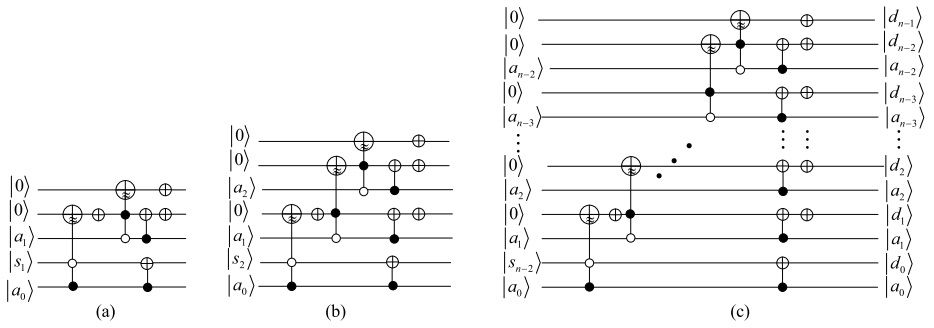

Figure 20 The circuits realizing the special operation in (9) including (a) $n = 3$, (b) $n = 4$, and (c) $n \geq 4$

fore, we can use an SMS, $(n-1)$ SMAS, and a CMA to realize the divider. For instance, the 2-bit and 3-bit dividers are presented in Fig. 19.

The SMS in Fig. 19 perfects the following operation:

$$(s_{n-2} - a) \bmod 2^n, \tag{9}$$

with $a = a_{n-2} \ldots a_1 a_0$ and $a_{n-2}, \ldots, a_1, a_0, s_{n-2} \in \{0, 1\}$.

We design circuits in Fig. 20 for the special operation in (9) using approximate Toffoli gates in Fig. 4(c) and (d).

Firstly, we substitute circuits in Fig. 20 for SMS and eliminate some CNOT gates. Then, we use rule 2 to design circuits of the 2-bit and 3-bit dividers optimizing CNOT gates in Fig. 21(a) and (b). Circuits in dashed boxes in Fig. 21(b) are named the first module of the divider (FMD), the iterative module of the divider (IMD), and the last module of the divider (LMD), respectively. Finally, we use the three modules to design the n-bit divider in Fig. 21(c).

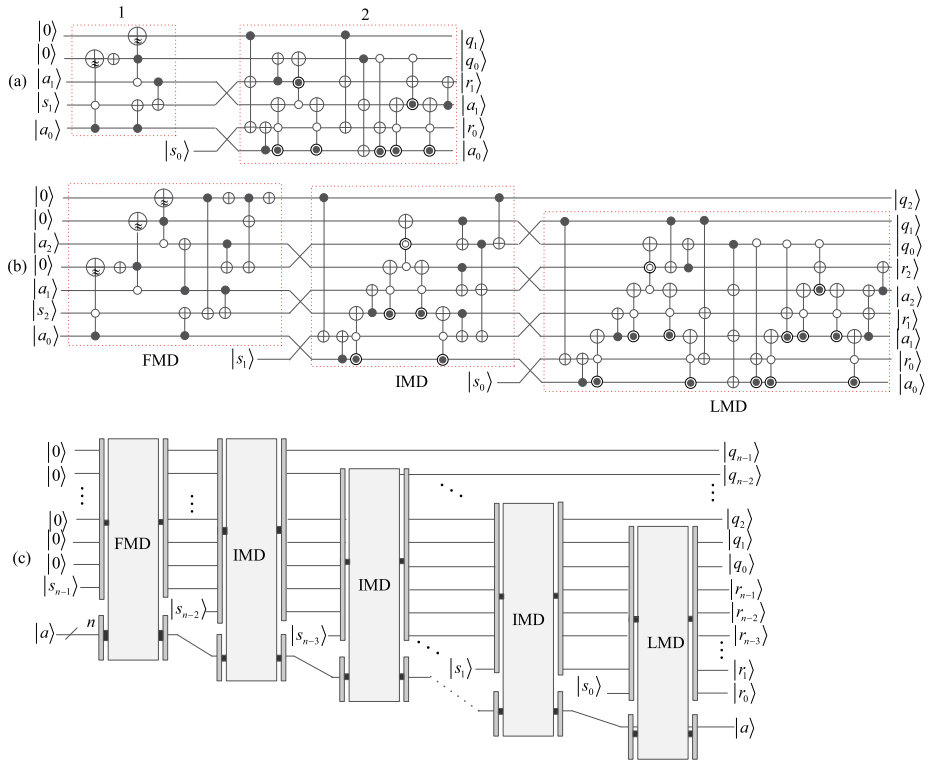

Figure 21 The circuits optimizing CNOT gates for (a) the 2-bit divider, (b) the 3-bit divider, and (c) the n-bit divider ($n \geq 3$)

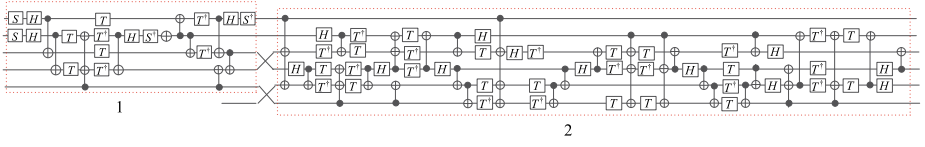

Figure 22 The Clifford + T circuit for the 2-bit divider

Table 3 Performance indexes the divider and its modules with $n \geq 3$

Operations	T-count	T-depth	CNOT-count	CNOT-depth	H-count	S-count	X-count	Width
FMD	$4n$	$n+1$	$7n-2$	$3n+2$	$2n$	$2n$	3	$2n+1$
IMD	$8n-1$	$2n+1$	$11n-3$	$4n+4$	$4n-2$	0	0	$2n+2$
LMD	$20n-8$	$5n$	$26n-14$	$14n-4$	$8n-4$	0	0	$2n+2$
2-bit divider	39	13	48	30	16	4	1	6
n-bit Divider	$8n^2+7n-6$	$2n^2+3n-1$	$11n^2+8n-10$	$4n^2+13n-10$	$4n^2$	$2n$	3	$3n$

5.2 Clifford + T circuits for the divider

We use rule 5 and Clifford + T circuits for approximate Toffoli gates in Fig. 4 to design the Clifford + T circuit of the 2-bit divider in Fig. 22. Clifford + T circuits in dashed boxes 1 and 2 in Fig. 22 correspond to circuits in dashed boxes 1 and 2 in Fig. 21(a).

Similarly, we propose the Clifford + T circuit for the FMD in Fig. 23(a) using approximate Toffoli gates in Fig. 4. The Clifford + T circuit for the IMD is presented in Fig. 23(b) using rule 5.

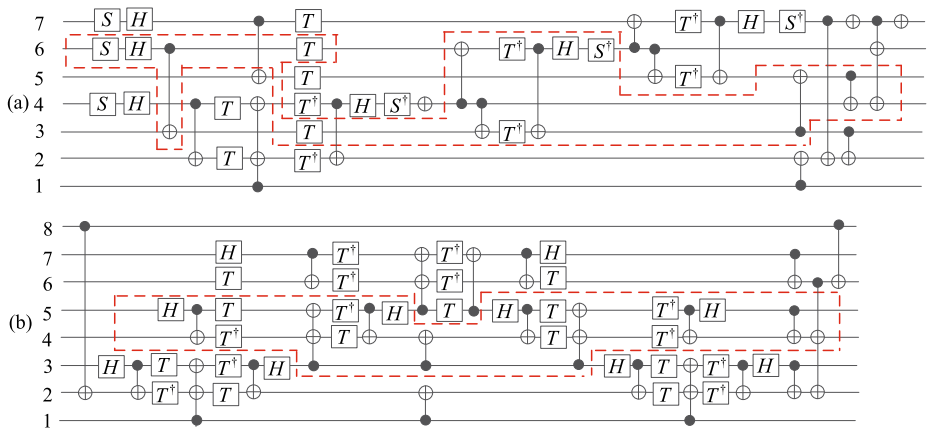

Figure 23 Clifford + T circuits for (a) the FMD and (b) the IMD. Circuits in dashed boxes are iterative circuits of the FMD and IMD, respectively

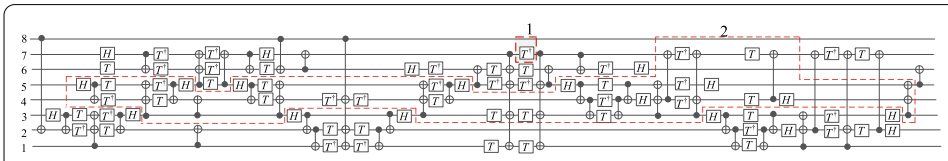

Figure 24 The Clifford + T circuit for the LMD. The circuit in dashed box 2 is the iterative circuit of the LMD. For each iteration of the circuit in box 2, the gate in dashed box 1 will be multiplied by $T^\dagger$

We calculate performance indexes of the FMD and LMD by analyzing iterative circuits in dashed boxes in Fig. 23:

$$\begin{cases} \text{FMD's T-count: } 4(n-2)+8 = 4n, \\ \text{FMD's T-depth: } (n-2)+3 = n+1, \\ \text{FMD's CNOT-count: } 7(n-2)+12 = 7n-2, \\ \text{FMD's CNOT-depth: } 3(n-2)+8 = 3n+2, \end{cases} \tag{10}$$

and

$$\begin{cases} \text{IMD's T-count: } 8(n-2)+15 = 8n-1, \\ \text{IMD's T-depth: } 2(n-2)+5 = 2n+1, \\ \text{IMD's CNOT-count: } 11(n-2)+19 = 11n-3, \\ \text{IMD's CNOT-depth: } 4(n-2)+12 = 4n+4. \end{cases} \tag{11}$$

Table 4 Comparisons of basic arithmetic operators

Operators		T-count	T-depth	CNOT-count	CNOT-depth	Width
Modular adder	Proposed	$8n-9$	$2n-1$	$13n-21$	$6n-7$	$2n$
	[23]	$14n-21$	$6n-9$	$17n-18$	$12n-17$	$2n$
	[29]	$14n-21$	$6n-6$	$19n-28$	$14n-19$	$2n+1$
	[32]	$14n-14$	$6n-6$	$17n-18$	$13n-12$	$2n$
	[36]	$8n-9$	$2n-1$	$13n-21$	$8n-12$	$2n$
Modular adder-subtractor	Proposed	$8n-9$	$2n-1$	$15n-23$	$6n-5$	$2n+1$
	[33]	$14n-14$	$6n-6$	$20n-18$	$13n-10$	$2n+1$
	[36]	$8n-9$	$2n-1$	$15n-23$	$8n-10$	$2n+1$
Controlled modular adder	Proposed	$14n-7$	$3n+1$	$20n-18$	$12n-10$	$2n+1$
	[23]	$21n-14$	$9n-6$	$23n-20$	$18n-13$	$2n+1$
	[36]	$14n-7$	$3n+1$	$20n-17$	$12n-9$	$2n+1$
	[38]	$21n-14$	$9n-6$	$25n-20$	$20n-14$	$2n+1$

The Clifford + T circuit for the LMD is presented in Fig. 24 using rule 5. From Fig. 24, we give performance indexes of the LMD as follows:

$$\begin{cases} \text{LMD's T-count: } \leq 20(n-2)+32 = 20n-8, \\ \text{LMD's T-depth: } 5(n-2)+10 = 5n, \\ \text{LMD's CNOT-count: } 26(n-2)+38 = 26n-14, \\ \text{LMD's CNOT-depth: } 14(n-2)+24 = 14n-4. \end{cases} \tag{12}$$

Furthermore, we give performance indexes of the 2-bit divider, n-bit divider ($n \geq 3$), and its modules in Table 3.

6 Comparative analysis

6.1 Comparisons of basic arithmetic operators

The section Introduction shows that the adder has the best T-count $4n-4$ in [30]. But the adder needs $2n-2$ measurements. It is thus not directly comparable with the T-count. Considering the circuit width, T-count, T-depth, CNOT-count, and CNOT-depth, we compare the proposed works with the rest basic arithmetic operators in [23, 29, 33, 36, 38]. The results are presented in Table 4. Table 4 shows that the proposed basic arithmetic operators are superior to the others.

Objectively, compared with these operators in [36], the proposed basic arithmetic operations have only a slight advantage regarding performance indexes. However, the proposed basic arithmetic operators have a significant advantage: They are more convenient for designs of the multiplier and divider. For instance, the proposed controlled modular adder can be used to realize the LMD of the divider with excellent performance indexes (see Fig. 24).

Note: Thapliyal et al. use the adder and subtractor in [32, 33] to an efficient divider in [38]. We calculate the performance indexes of the divider in [38] because it is one of the main comparison objects of the divider proposed in this paper. Therefore, we describe the computation process of the adder and subtractor in [32, 33] as follows. Circuits of the 4-bit adder and subtractor are shown in Fig. 25. The n-bit adder in Fig. 25(a) is modified to the n-bit subtractor in Fig. 25(b) by adding $3n+1$ X gates. They consist of $4n-5$ CNOT, $n-1$ Toffoli, and n Peres gates, respectively. The adder and subtractor can be modified into the modular adder and subtractor by eliminating the high bit (see Fig. 25(c) and (d)). The

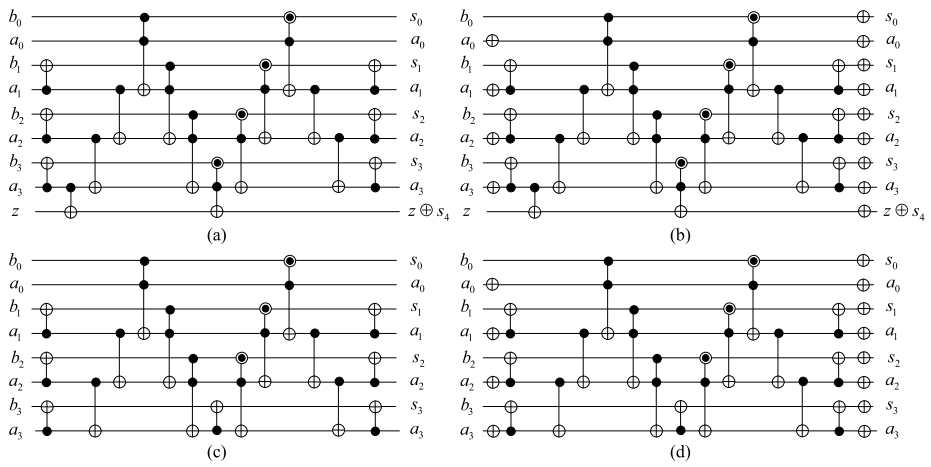

Figure 25 The reversible 4-bit adder, subtractor, modular adder, and modular subtractor for (a) the adder in [32], (b) the subtractor in [33], (c) the modular adder, and the modular subtractor

generated modular subtractor named Subtraction in [38] comprises $4n-5$ CNOT gates, $n-1$ Toffoli gates, and $n-1$ Peres gates. Furthermore, Thapliyal develops a reversible adder-subtractor in [33] by adding $3n+1$ CNOT gates to the n-bit adder in Fig. 25(a). Similarly, a modular adder-subtractor is obtained by eliminating the high bit of the reversible adder-subtractor in [33]. Therefore, the n-bit modular adder-subtractor named Ctrl-AddSub in [38] consists of $7n-5$ CNOT, $n-1$ Toffoli, and $n-1$ Peres gates. Clifford + T circuits of Toffoli gates proposed in [15, 23, 36] have T-count 7, T-depth 3, CNOT-count 7, and CNOT-depth 6. The Clifford + T circuit of the Peres gate with T-count 7, T-depth 3, CNOT-count 6, and CNOT-depth 5 is proposed in [23]. The performance indexes of Subtraction are given by

$$\begin{cases} \text{T-count: } 7(n-1)+7(n-1)=14n-14, \\ \text{T-depth: } 3(n-1)+3(n-1)=6n-6, \\ \text{CNOT-count: } 4n-5+7(n-1)+6(n-1)=17n-18, \\ \text{CNOT-depth: } 2n-1+6(n-1)+5(n-1)=13n-12. \end{cases} \tag{13}$$

Similarly, the modular adder-subtractor (Ctrl-AddSub) has $14n-14$ T-counts, $6n-6$ T-depths, $17n-18+3n=20n-18$ CNOT-counts, and $13n-12+2=13n-10$ CNOT-depths.

6.2 Comparisons of multipliers and dividers

6.2.1 Method comparisons with previous works

The main contributions of this paper are to design the new multiplier and divider. Therefore, we provide a description of the new contributions in this section. Compared to multipliers based on the measure-and-fixup approach [42, 43], our method for the multiplier differs in that it does not require quantum measurements. Method comparisons with previous works [23, 35, 36] are presented in Table 5. Realization formulas of multipliers are

$$\mathrm{s} = -\overline{b_0}a - \sum_{k=1}^{n-1}(-1)^{b_k}2^{k-1}a + 2^{n-1}a, \tag{14}$$

Table 5 Method comparisons of multipliers with previous works. RF, UO, OO, and OR denote realization formulas, unchanged operands, optimization objects, and optimization rules

Operators	RF	UO	OO	OR	Realization of $-\overline{b_0}a$
Proposed	Eq. (14)	One	T and CNOT gates	Based on LI gates	Circuit in Fig. 13
[23]	Eq. (15)	Two	T gates	–	–
[35]	Eq. (15)	Two	T gates	–	–
[36]	Eq. (16)	One	T gates	Based on approximate gates	CMS

$$s = \sum_{k=0}^{n-1} b_k 2^{k-1} a, \tag{15}$$

and

$$s = \left[-\overline{b_0} + (-1)^{\overline{b_1}}\right]a + 2^{n-1}a + \sum_{k=2}^{n-1} (-1)^{\overline{b_k}} 2^{k-1} a, \tag{16}$$

where b_k is equal to $1 - b_k$ with $k \in \{1, 2, \ldots, n-1\}$.

Table 5 shows that the proposed method is different from methods in [23, 35] in terms of realization formulas, unchanged operands, and optimization objects. From realization formulas in Table 5, we obtain that methods in [23, 35] use n controlled modular adders to implement multipliers, respectively. Our method adopts the circuit in Fig. 13, $(n-1)$ special modular subtractor-adders, and a special modular adder to implement the multiplier. Compared to the method in [36], our method used the different realization formula, optimization objects, optimization rules, and realization of $-\overline{b_0}a$. According to the realization formula in [36], the multiplier is implemented by a circuit named SM, $(n-1)$ modular adder-subtractors, and a modular adder, where the SM consists of a controlled modular subtractor and other gates to realize $[-\overline{b_0} + (-1)^{\overline{b_1}}]a$. The proposed multiplier reduces the circuit width, T-count, T-depth, CNOT-count and CNOT-depth, because the method in the paper uses the different realization formula, optimization objects, and optimization rules.

Due to the Hermitian property of LI gates, we can reverse the circuit order for the special multiplier to obtain the divider. The method for the divider in the paper is significantly different from methods of dividers in [36, 38].

6.2.2 Performance comparisons of multipliers

Multipliers based on the measure-and-fixup approach have small T-counts. For instance, T-counts of two multipliers in [42, 43] are $6n^2 + O(n)$ and $8n^2 - 4n$, respectively. But, the two multipliers also require $O(n^2)$ quantum measurements. We compare the proposed multiplier against recent works without quantum measurements [23, 35, 36]. Two multipliers are proposed in [36], but the second multiplier has better performance indexes than the first multiplier. Therefore, we only compare the proposed multiplier with the second multiplier in [36]. The results are presented in Table 6 and Table 7, which illustrate that the proposed multiplier is superior to the others for the five performance indexes. For instance, the CNOT-count of the proposed 32-bit multiplier achieves improvement ratios of 50.20 percent, 54.35 percent, and 26.96 percent compared to the works presented in [23, 35, 36], respectively. The caveat is that the proposed multiplier can only realize the

Table 6 Comparisons of multipliers for $n \geq 3$

Operators	T-count	T-depth	CNOT-count	CNOT-depth	Width
Proposed	$8n^2$+9n-12	$2n^2$+3n+1	$11n^2$+9n-16	$4n^2$+10n-8	3n
[23]	$21n^2$-9n-5	$9n^2$-3n-3	$23n^2$-12n-4	$18n^2$-48n+30	4n
[35]	$21n^2$-14	$9n^2$-3n-3	$25n^2$-10n-8	$20n^2$-10n	$4n+1$
[36]	$8n^2$+15n-8	$2n^2$+4n	$15n^2$+14n-15	$8n^2$+11n-16	$3n+1$

special multiplication ab with $a \neq 0$. The other three multipliers do not have this limitation. The proposed multiplier based on LI gates has another advantage: it can be easily used to design dividers.

Note: Performance indexes of the multiplier in [36] are miscalculated. We recalculate them as follows. The n-bit multiplier consists of an SM module, $n-1$ modular adder-subtractors ($(n+1)$-bit), an $(n+1)$-bit modular adder, and an Aswap module. Performance indexes of the modular adder and modular adder-subtractor in [36] have been listed in Table 4. The SM module has $20n-9$ T-counts, $5n-1$ T-depth, $33n-23$ CNOT-counts, and $21n-21$ CNOT-depths. The Aswap module has $4n$ T-counts, 2 T-depths, $6n$ CNOT-count, and 5 CNOT-depths. Performance indexes of the n-bit multiplier are calculated by

$$\begin{cases} \text{T-count: } 20n-9+[8(n+1)-9](n-2)+8(n+1) \\ \qquad -9+4n=8n^2+15n-8, \\ \text{T-depth: } 5n-1+[2(n+1)-1](n-2)+2(n+1)+1 \\ \quad =2n^2+4n, \\ \text{CNOT-count: } 33n-23+[15(n+1)-23](n-2) \\ \qquad +13(n+1)-21+6n=15n^2+14n-15, \\ \text{CNOT-depth: } 21n-21+[8(n+1)-10](n-2) \\ \qquad +8(n+1)-12+5=8n^2+11n-16. \end{cases}$$

6.2.3 Performance comparisons of dividers

We compare the proposed divider against recent works [36, 38]. Thapliyal et al. design restoring and non-restoring dividers [38]. The non-restoring divider has a smaller T-count and T-depth than the restoring divider, so we only compare the proposed divider with the non-restoring divider. The results in Table 8 and Table 9 illustrate that the proposed divider is superior to the others in terms of T-count, T-depth, CNOT-count, and CNOT-depth. For instance, the proposed 32-bit divider achieves improvement ratios of 40.41 percent, 31.64 percent, 45.27 percent, and 65.93 percent in terms of T-count, T-depth, CNOT-count, and CNOT-depth compared to the work presented in [36]. Meanwhile, the proposed 32-bit divider reduces T-count by 45.54 percent, T-depth by 67.62 percent, CNOT-count by 47.36 percent, and CNOT-count by 68.85 percent when compared with the work in [38]. The n-bit division requires at least $3n$ qubits to store the quotient, remainder, and operand; thus, the proposed divider has the minimum circuit width $3n$ for the n-bit division keeping an operand unchanged.

Note: Thapliyal et al. designed a non-restoring divider to realize the positive 2's complement division [38]. An n-bit positive integer can be changed into the complement number by adding a binary 0 before the high bit. Therefore, a complement operand of the n-bit

Table 7 Comparisons of multipliers by increasing n from 4 to 32. C-count and C-depth denotes CNOT-count and CNOT-depth, respectively

Operators		T-count	T-depth	C-count	C-depth	Width
$n = 4$	Proposed	152	45	196	96	12
	[23]	295	129	316	126	16
	[35]	322	129	352	280	17
	[36]	180	48	281	156	13
$n = 8$	Proposed	572	153	760	328	24
	[23]	1267	549	1372	798	32
	[35]	1330	549	1512	1200	33
	[36]	2280	160	1057	584	25
$n = 16$	Proposed	2180	561	2944	1176	48
	[23]	5227	2253	5692	3870	64
	[35]	5362	2253	6232	4960	65
	[36]	2280	576	4049	2208	49
$n = 32$	Proposed	8468	2145	11536	4408	96
	[23]	21211	9117	23164	16926	128
	[35]	21490	9117	25272	20160	129
	[36]	8664	2176	15793	8528	97
Impr.(%) for $n = 32$	w.r.t. [23]	60.08	76.47	50.20	73.96	25
	w.r.t. [35]	60.60	76.47	54.35	78.13	25.58
	w.r.t. [36]	2.26	1.42	26.96	48.31	1.03

Table 8 Comparisons of dividers for $n \geq 3$

Operators	T-count	T-depth	CNOT-count	CNOT-depth	Width
Proposed	$8n^2+7n-6$	$2n^2+3n-1$	$11n^2+8n-10$	$4n^2+13n-10$	$3n$
[36]	$14n^2-7n+1$	$3n^2+2n-1$	$21n^2-15n+7$	$13n^2-3n-1$	$3n$
[38]	$14n^2+35n-14$	$6n^2+15n-6$	$20n^2+44n-21$	$13n^2+36n-13$	$3n+2$

Table 9 Comparisons of dividers by increasing n from 4 to 32. C-count and C-depth denotes CNOT-count and CNOT-depth, respectively

Operators		T-count	T-depth	C-count	C-depth	Width
$n = 4$	Proposed	150	43	198	106	12
	[36]	197	55	283	195	12
	[38]	350	150	475	339	15
$n = 8$	Proposed	562	151	758	350	24
	[36]	841	207	1231	807	24
	[38]	1162	498	1611	1107	26
$n = 16$	Proposed	2154	559	2934	1222	48
	[36]	3473	799	5143	3279	48
	[38]	4130	1770	5803	3891	50
$n = 32$	Proposed	8410	2143	11510	4502	96
	[36]	14113	3135	21031	13215	96
	[38]	15442	6618	21867	14451	98
Impr.(%) for $n = 32$	w.r.t. [36]	40.41	31.64	45.27	65.93	0
	w.r.t. [38]	45.54	67.62	47.36	68.85	2.04

positive integer division requires $m = n + 1$ bits. The n-bit divider consists of an m-bit modular subtractor named Subtraction, $(m - 1)$ m-bit modular adder-subtractor named Ctrl-AddSub, and an $(m - 1)$-bit controlled modular adder named Ctrl-AddNOP [38]. Performance indexes of the three modules can be found in Table 4. Then, performance

indexes of the n-bit divider are calculated by

$$\begin{cases} \text{T-count:} \\ 14m - 14 + (14m - 14)(m - 1) + 21(m - 1) - 14 \\ \quad = 14m^2 + 7m - 35 \\ \quad = 14n^2 + 35n - 14, \\ \text{T-depth:} \\ 6m - 6 + (6m - 6)(m - 1) + 9(m - 1) - 6 \\ \quad = 6m^2 + 3m - 15 \\ \quad = 6n^2 + 15n - 6, \\ \text{CNOT-count:} \\ 17m - 18 + (20m - 18)(m - 1) + 25(m - 1) - 20 \\ \quad = 20m^2 + 4m - 45 \\ \quad = 20n^2 + 44n - 21, \\ \text{CNOT-depth:} \\ 13m - 12 + (13m - 10)(m - 1) + 20(m - 1) - 14 \\ \quad = 13m^2 + 10m - 36 \\ \quad = 13n^2 + 36n - 13, \\ \text{width: } 3m - 1 = n + 2, \end{cases}$$

with $m = n + 1$.

7 Conclusions and future works

In this paper, we have proposed a special multiplier and a divider based on LI and approximate Toffoli gates. We designed circuits of basic arithmetic operations used in the proposed multiplier and divider, such as the modular adder, modular adder-subtractor, controlled modular adder, special modular adder, special modular adder-subtractor, and their inverses. These basic arithmetic operations based on LI gates have the advantage that their inverse can be realized by inverting the circuit order of the corresponding arithmetic operations. We have proposed new rules of LI gates to design Clifford + T circuits of the proposed multiplier and divider, optimizing T-count, T-depth, CNOT-count, and CNOT-depth. Clifford + T circuits of the proposed multiplier and divider are superior to existing multiplier and dividers in terms of T-count, T-depth, CNOT-count, and CNOT-depth. Furthermore, circuit widths of the proposed n-bit multiplier and divider are $3n$. That is, our multiplier and divider have reached the minimum width of multipliers and dividers, keeping an operand unchanged. As a future work, it will be interesting to apply the proposed multiplier and divider in quantum image processing, such as quantum bilinear interpolation algorithm.

Funding

This work is supported by the National Natural Science Foundation of China under Grant No.62062035, Fujian Provincial Natural Science Foundation Project under Grant No. 2023J011389, and the Science and Technology Project of Guangxi under Grant Nos. 2023JJA170043 and 2020GXNSFDA238023.

Data availability
The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Consent to participate
All authors consent to participate.

Consent for publication
All authors consent for publication.

Competing interests
The authors declare no competing interests.

Author contributions
P. Fan and H.S. Li wrote the main manuscript text. All authors reviewed the manuscript.

Author details
[1]College of Computer and Control Engineering, Minjiang University, Fuzhou, 350108, Fujian, China. [2]College of Electronic and Information Engineering, Guangxi Normal University, Guilin, 541004, Guangxi, China.

Received: 3 February 2024 Accepted: 14 February 2024 Published online: 23 February 2024

References

1. Ladd TD, Jelezko F, Laflamme R, Nakamura Y, Monroe C, O'Brien JL. Quantum computers. Nature. 2010;464:45–53.
2. Nielsen MA, Chuang IL. Quantum computation and quantum information. Cambridge: Cambridge University Press; 2000.
3. Yan F, Iliyasu AM, Guo Y, Yang H. Flexible representation and manipulation of audio signals on quantum computers. Theor Compt Sci. 2013;752:71–85.
4. Li HS, Fan P, Xia HY, Peng H, Song S. Quantum implementation circuits of quantum signal representation and type conversion. IEEE Trans Circuits Syst I, Regul Pap. 2019;66:341–54.
5. Pang CY, Zhou RG, Hu BQ et al. Signal and image compression using quantum discrete cosine transform. Inf Sci. 2019;473:121–41.
6. Li HS, Fan P, Peng H, Song S, Long GL. Multilevel 2-d quantum wavelet transforms. IEEE Trans Cybern. 2022;52:8467–80.
7. Manzano A, Musso D, Leitao A. Real quantum amplitude estimation. EPJ Quantum Technol. 2023;10:2
8. Barenco A, Bennett CH et al. Elementary gates for quantum computation. Phys Rev A. 1995;52:3457–67.
9. Zhou X, Leung DW, Chuang IL. Elementary gates for quantum computation. Phys Rev A. 2000;62:052316.
10. Noorallahzadeh M, Mosleh M, Datta K. A new design of parity-preserving reversible multipliers based on multiple-control Toffoli synthesis targeting emerging quantum circuits. Front Comput Sci. 2024;18(6):186908.
11. Noorallahzadeh M, Mosleh M, Ahmadpour SS, Pal J, Sen B. A new design of parity preserving reversible Vedic multiplier targeting emerging quantum circuits. Int J Numer Model Electron Netw Devices Fields. 2023;36:e3089
12. Noorallahzadeh M, Mosleh M, Misra NK, Mehranzadeh A. A novel design of reversible quantum multiplier based on multiple-control Toffoli synthesis. Quantum Inf Process. 2023;22(4):167
13. Giles B, Selinger P. Exact synthesis of multiqubit Clifford+ T circuits. Phys Rev A. 2013;87(3):032332.
14. Kliuchnikov V, Maslov D, Mosca M. Asymptotically optimal approximation of single qubit unitaries by Clifford and T circuits using a constant number of ancillary qubits. Phys Rev Lett. 2013;110(19):190502.
15. Amy M, Maslov D, Mosca M, Roetteler M. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Trans Comput-Aided Des Integr Circuits Syst. 2013;32(6):818–30.
16. Munoz-Coreas E, Thapliyal H. Quantum circuit design of a T-count optimized integer multiplier. IEEE Trans Comput. 2019;68(5):729–39.
17. 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. 2014;33(10):1476–89.
18. Nam Y, Ross NJ, Su Y et al. Automated optimization of large quantum circuits with continuous parameters. npj Quantum Inf. 2018;4:23.
19. Niemann P, de Almeida AAA, Dueck G, Drechsler R. Design space exploration in the mapping of reversible circuits to ibm quantum computers. In: 2020 23rd euromicro conference on digital system design (DSD). 2020. p. 401–7.
20. Smolin JA, DiVincenzo DP. Five two-bit quantum gates are sufficient to implement the quantum Fredkin gate. Phys Rev A. 1996;53(4):2855–6.
21. Peres A. Reversible logic and quantum computers. Phys Rev A. 1985;32:3266.
22. Thapliyal H, Ranganathan N. Design of efficient reversible binary subtractors based on a new reversible gate. In: IEEE computer society annual symposium on VLSI. Tampa, FL, USA. 2009. p. 229–34.
23. Li HS, Fan P, Xia H, Peng H, Long GL. Efficient quantum arithmetic operation circuits for quantum image processing. Sci China, Phys Mech Astron. 2020;63(8):280311.
24. Li HS. The optimization and application of 3-bit Hermitian gates and multiple control Toffoli gates. IEEE Trans Quantum Eng. 2022;3:3102715.
25. Vedral V, Barenco A, Ekert A. Quantum networks for elementary arithmetic operations. Phys Rev A. 1996;54(1):147–53.
26. Draper TG, Kutin SA, Rains EM et al. A logarithmic-depth quantum carry-lookahead adder. 2004. arXiv preprint. arXiv:quant-ph/0406142.

27. Takahashi Y, Kunihiro N. A linear-size quantum circuit for addition with no ancillary qubits. Quantum Inf Comput. 2005;5(6):440–8.
28. Takahashi Y, Tani S, Kunihiro N. Quantum addition circuits and unbounded fan-out. 2009. arXiv preprint. arXiv:0910.2530.
29. Cuccaro SA, Draper TG, Kutin SA, et al. A new quantum ripple-carry addition circuit. 2004. arXiv preprint. arXiv:quant-ph/0410184.
30. Gidney C. Halving the cost of quantum addition. 2018. arXiv preprint. arXiv:1709.06648v3.
31. Thapliyal H, Munoz-Coreas E, Khalus V. Quantum circuit designs of carry lookahead adder optimized for T-count T-depth and qubits. Sustain Comput Inf Sys. 2021;29:100457.
32. Thapliyal H, Ranganathan N. Design of efficient reversible logic-based binary and BCD adder circuits. ACM J Emerg Tech Comput. 2013;9(3):17.
33. Thapliyal H. Mapping of subtractor and adder-subtractor circuits on reversible quantum gates. In: Transactions on Computational Science XXVII. Berlin: Springer; 2016. p. 16–34.
34. Jayashree HV, Thapliyal H, Arabnia HR et al. Ancilla-input and garbage-output optimized design of a reversible quantum integer multiplier. J Supercomput. 2016;72(4):1477–93.
35. Munoz-Coreas E, Thapliyal H. Quantum circuit design of a T-count optimized integer multiplier. IEEE Trans Comput. 2019;68(5):729–39.
36. Li HS, Fan P, Xia H, Long GL. The circuit design and optimization of quantum multiplier and divider. Sci China, Phys Mech Astron. 2022;65(6):260311.
37. Khosropour A, Aghababa H, Forouzandeh B. Quantum division circuit based on restoring division algorithm. In: Pro. IEEE eighth international conference on information technology: new generations. 2011. p. 1037–40.
38. Thapliyal H, Munoz-Coreas E, Varun TSS et al. Quantum circuit designs of integer division optimizing T-count and T-depth. IEEE Trans Emerg Top Comput. 2021;9(2):1045–56.
39. Mcclean J, Lamat L, Aspuru-Guzik A, Solano E. From transistor to trapped-ion computers for quantum chemistry. Sci Rep. 2014;4:3589.
40. Wen J, Lv D, Yung MH, Long GL. Variational quantum packaged deflation for arbitrary excited states. Quantum Eng. 2021;3(4):e80
41. Wei S, Chen Y, Zhou Z, Long G. A quantum convolutional neural network on NISQ devices. AAPPS Bull. 2022;32:2.
42. Gidney C. 2017. https://algassert.com/post/1709.
43. Munoz-Coreas E, Thapliyal H. T-count optimized quantum circuits for bilinear interpolation. In: Proc. IEEE ninth int. green sust. Comput. Conf. 2018. p. 212–9.

Publisher's Note