The Journal of Supercomputing (2023) 79:12656–12670
https://doi.org/10.1007/s11227-023-05162-x

Efficient design of a quantum absolute-value circuit using Clifford+T gates

Francisco Orts[1] · Gloria Ortega[1] · Elías F. Combarro[2] · Ignacio F. Rúa[3] · Antonio M. Puertas[4] · Ester M. Garzón[1]

Accepted: 4 March 2023 / Published online: 18 March 2023

Abstract
Current quantum computers have a limited number of resources and are heavily affected by internal and external noise. Therefore, small, noise-tolerant circuits are of great interest. With regard to circuit size, it is especially important to reduce the number of required qubits. Concerning to fault-tolerance, circuits entirely built with Clifford+T gates allow the use of error correction codes. However, the T-gate has an excessive cost, so circuits with a high number of T-gates should be avoided. This work focuses on optimising in such terms an operation that is widely used in larger circuits and algorithms: the calculation of the absolute-value of two's complement encoded integers. The proposed circuit halves the number of required T gates with respect to the best circuit currently available in the literature. Moreover, our circuit requires at least 2 qubits less than the other circuits for such an operation.

Keywords Reversible gate · Quantum circuit · T-count · Absolute-value · Quantum computing

1 Introduction

Quantum computing [1] is a new computational paradigm that explicitly uses quantum phenomena such as superposition, interference and entanglement to achieve speed-ups over what is possible with classical computers in certain tasks. For instance, the famous algorithm for integer factorization introduced by Peter Shor [2] is exponentially faster than the best classical algorithm for the same task that we have at our disposal. Another provable speed-up is the one provided by Grover's algorithm for black-box search problems [3], achieving a quadratic speed-up over any algorithm that only uses classical resources.

✉ Francisco Orts
francisco.orts@ual.es

Extended author information available on the last page of the article

There are several realizations of quantum computing, but the most popular one — and the one currently implemented in most quantum computers — is the quantum circuit model [1]. This model is analogous to the (classical) digital circuit model, but it uses quantum bits (or qubits) instead of bits, quantum gates (or unitary transformations) instead of digital gates and, in addition, it needs explicit measurements to obtain results.

Quantum algorithms, including Shor's factoring method and Grover's search algorithm, usually depend on circuits that implement arithmetical primitives such as modular addition, multiplication and exponentiation, or, for instance, sign and absolute value computations in two's complement [4]. In principle, these operations can be directly adapted from their classical counterparts with linear overhead, but reducing the size and depth of quantum circuits is extremely important to mitigate the effects of noise and imperfect gate realization.

For these reason, a very active line of research in the quantum computing field [5–8] is devoted to producing quantum circuits for arithmetical primitives that optimize some important metrics [9–11] such as number of gates, T-count, number of two-qubit gates, ancillary qubits and garbage outputs.

In this paper, we follow this line of research in the particular case of the absolute value computation for integer numbers in two's complement representation. Computing the absolute value of such numbers is an essential operation in quantum algorithms with applications in fields as diverse as audio signal processing, protein structure prediction, or stock market indicators [12–14]. In terms of relevance, the calculation of the absolute value can account for from 10% of these algorithms to more than 30%, as in the case of quantum audio signal inversion, addition, or delay operations [12]. The availability of efficient circuits for the absolute value computation, as well as for the rest of the operations involved in these algorithms, will increase their overall efficiency.

In this work, two fundamental objectives are pursued. The first is to achieve a circuit built exclusively from Clifford+T gates to achieve a fault-tolerant implementation. As a second objective, a cost-effective implementation is sought to reduce costs compared to currently available circuits. Combining these two objectives is not trivial, as will be explained in the next section. However, the obtained results will show that the proposed circuit manages to achieve both objectives.

The rest of the paper is organized as follows. Section 2 introduces the basic concepts about quantum circuits, gates, and metrics necessary for the understanding of this work, as well as the methodology used for its development. Section 3 presents the proposed circuit in detail, indicating all the necessary steps for its reconstruction. Section 4 presents the obtained results, as well as a comparison between the proposed circuit and the most relevant circuits available in the state-of-the-art. Finally, the last section presents the conclusions.

2 Background

2.1 Quantum circuits, gates, and metrics

Quantum circuits are built using qubits and quantum gates that modify the state of the qubits. In classical circuits, there are only two logic gates that allow working

on a bit: the NOT gate and the identity gate [15]. At the two-digit level, there are not many more gates available. However, there are infinite quantum gates that allow working on one qubit (and therefore also infinite possibilities for cases with two or more qubits) [16]. Fortunately, there are sets of quantum gates that allow to approximate the infinite possible quantum gates. One of these groups, but not the only one, is the so-called Clifford+T group. The Clifford group, generated by the H, S, and CNOT gates, has a wide variety of applications thanks to its feature of being able to convert any Pauli operation into another Pauli operation. Nevertheless, it is not universal on its own, but it needs the T gate to form the Clifford+T group and, become a universal set [1, 16].

One of the goals of this work is to get a circuit implemented exclusively using Clifford+T gates. However, being a universal group is not the only reason for its choice. Current quantum computers are very sensitive to internal and external noise. It has been shown that building circuits using only Clifford+T gates will make such circuits fault-tolerant thanks to the use of error correcting codes [17, 18]. Nevertheless, there is an associated problem. The cost of the T gate is much higher than that of the Clifford gates, up to 100 times higher. The T-gate is so expensive that a current trend in the quantum circuit literature is to measure the cost of quantum circuits in terms of the T-gate. This metric - the number of T-gates in a circuit - is called T-count. In this work, we have focused not only on building the circuit using Clifford+T gates, but also on reducing the T-count to make the circuit as efficient as possible.

The proposed circuit will be built using only four gates: the CNOT gate, the Toffoli gate, the Temporary logical-AND gate, and the uncomputation gate of the Temporary logical-AND operation. Of this list of gates, only the CNOT gate is a direct member of the Clifford+T group. However, since the Clifford+T group is universal, the remaining gates can be approximated using only gates from this group. The most efficient implementation of the Toffoli gate in terms of T-count is the one proposed by Amy et al. [19] (Fig. 1a). It has a T-count of 7. However, the Temporary logical-AND gate, proposed by Gidney [11], has a T-count of only 4 (Fig. 1b). The Temporary logical-AND gate is similar to the Toffoli gate, but with a disadvantage: it can only act on a target qubit prepared in the state $(\frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{i\pi}{4}}|1\rangle)$. In other words, the Toffoli gate can be applied on any qubit, while the Temporary logical-AND gate can only be applied on a pre-prepared auxiliary qubit. Another important difference between the two gates is that to reverse a Toffoli operation, another Toffoli gate must be applied, thus doubling the T-count. Nevertheless, the uncomputation gate of the Temporary logical-AND gate can be used at no extra cost in terms of T-count thanks to the "measure and fixup" approach shown in Fig. 1c [11].

2.2 State-of-the-art in quantum circuit design for arithmetic operations

The importance of quantum circuits that implement arithmetic operations is due to the need for these operations in larger circuits and quantum algorithms of proven efficiency. An example already mentioned is Shor's algorithm, which

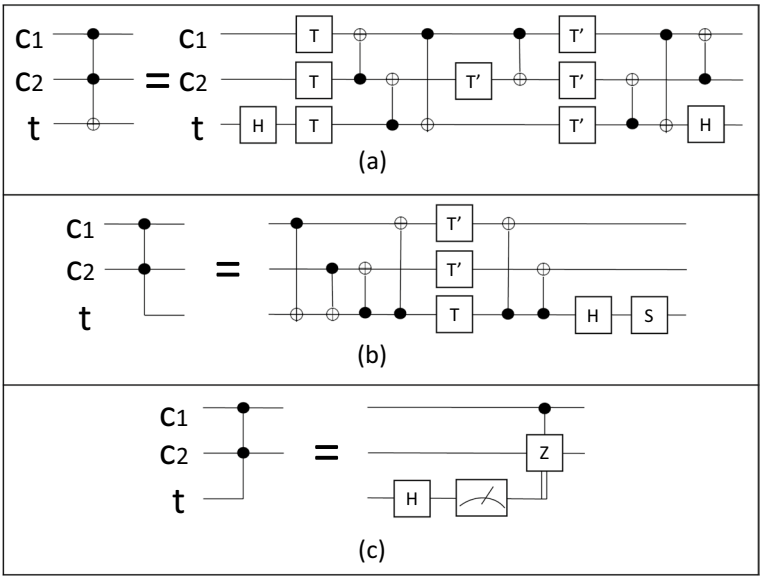

Fig. 1 Symbol and implementation of (**a**) the Toffoli gate proposed by Amy et al. [19], (**b**) the Temporary logical-AND gate proposed by Gidney [11], and (**c**) the uncomputation gate of the Temporary logical-AND gate

requires adders for its implementation. It is precisely because of this algorithm that addition is the arithmetic operation that receives more interest from the scientific community [20]. As a consequence, there are a huge amount of adders for quantum computing published in the literature [5, 20, 21]. In general, such adders can be classified into two main types: ripple-carry adders, and carry-ahead adders. The former are characterized by requiring fewer resources than the latter, while the latter are faster at the cost of requiring more resources. Carry-ahead adders use these extra resources to calculate the carry bits before the sum, thus reducing the total time of the operation. For instance, in the case of adding two N bit numbers, such an adder will calculate the n-th carry (with $n < N$) before the partial $n-1$-th sum finishes. Therefore, carries can be added to these partial sums without waiting for the previous partial sum to finish. Apart from their methodology, the designs are usually focused on optimizing one or more metrics: the quantum cost, the T-count, etc. Likewise, there are designs focused on achieving implementations that meet certain circumstantial conditions, such as only Clifford+T type gates are used in their construction [5, 21].

Although addition is the operation that receives most interest from the research community, the rest of the arithmetic operations are also necessary for various quantum algorithms. In the case of multiplication, there are several different works in the literature that have progressively optimized resources in terms of quantum gates, noise tolerance, or delay cost [22–27]. In the case of division, there are currently also several published works that, as with multiplication, have progressively optimized the use of the scarce resources currently available for quantum computers

[6, 28–32]. The need for these two operations is justified by the fact that they are involved in quantum algorithms focused on maximization of objective functions, quantum image processing, or the evaluation of transcendental functions [33–35].

Subtraction is also represented in the literature [36–38]. However, it is often performed using adders and two's complement representation (as in classical digital electronics) [20]. Given the aforementioned high interest that adders receive in quantum computing, which implies a high availability of this type of circuits, it is not surprising that the second option (using adders and two's complement representation to perform the subtraction) is the most widespread one [20]. Closely related to the latter are circuits called converters, which allow converting a signed binary number into two's complement. This operation can be performed using adders or subtractors. However, using a circuit specifically designed for this operation reduces costs and improves speed and noise tolerance [39–41].

Regarding the operation that is the focus of this paper, the computation of the absolute-value of a number, there are no circuits available in the literature at the time of writing or at least the authors have not been able to find any. The absolute-value is involved, as already mentioned, in existing quantum circuits and algorithms. However, in such works, conditional adders and certain extra operations are used to perform the computation of the absolute value. This methodology is explained in detail in the next subsection. In this work, a circuit specifically dedicated to this operation is proposed to make it more efficient compared to how it is currently performed in the mentioned works.

2.3 Methodology to compute the absolute-value of a number

The absolute-value of a two's complement number can be obtained by reversing its sign or leaving it as it is if the number is negative or positive, respectively. This process is called *taking the two's complement*, and it is widely used in classical digital design [42]. This process begins by identifying the sign of the number, contained in its most-significant digit. If this digit is 0, the number is positive. Otherwise, the number is negative. It can be observed in Table 1 that the absolute value of a number is equal to the number itself in the case of positive numbers. Then, no operations are necessary if the number is positive. However, actions are required in the case of negative numbers. For such cases, all digits of the number must be inverted, and number 1 must be added to the value obtained after the inversions. For instance, Table 1 shows that the value -5 is represented as 1011. By inverting its digits, the value 0100 is obtained. Adding $0100 + 1$ produces the result 0101, which corresponds to the value 5, the absolute value of -5.

It is also possible to obtain the absolute value of negative numbers represented in two's complement using a simple trick. This trick consists of going through the digits of the number starting from the least-significant one, that is, from right to left. Once the first 1 is found, the successive digits (following the direction of displacement to the left) must be inverted, leaving the first 1 and the 0 to its right (if any) invariant. Following the previous example, the absolute-value of 1011 (-5) will be obtained by finding the least significant digit at 1, which in this case coincides with

Table 1 All possible two's complement number using 4 digits

Original number		Absolute-value	
Two's complement	Decimal	Two's complement	Decimal
0000	0	0000	0
0001	1	0001	1
0010	2	0010	2
0011	3	0011	3
0100	4	0100	4
0101	5	0101	5
0110	6	0110	6
0111	7	0111	7
1000	−8	1000	8
1001	−7	0111	7
1010	−6	0110	6
1011	−5	0101	5
1100	−4	0100	4
1101	−3	0011	3
1110	−2	0010	2
1111	−1	0001	1

Decimal values are shown for the sake of clarity

the least significant digit of the number. Next, the digits to the left of that digit are inverted, obtaining the value 0101. Despite being a valuable trick for people, this second methodology is not used in digital electronics because it involves a greater number of operations to separate the subset of bits to be inverted from the bits that must remain invariant. [42, 43]. Therefore we have focused on using the first methodology, which involves inverting all the digits using the sign of the number itself, as well as performing an addition. The first operation can be easily performed using CNOT gates, while the second can be computed using adder circuits, which are the most abundant arithmetic circuits in quantum computing [5, 20, 21].

3 Proposed implementation

In this section we present our proposed quantum absolute-value circuit. The circuit has no garbage outputs, and it involves less qubits and T gates than the circuits in the state-of-the-art. The circuit computes the absolute-value of a integer number B represented using two's complement. The digits of B are stored in a register of qubits $|B_{N-1}\rangle ... |B_1\rangle |B_0\rangle$, being N the number of digits of B. Extra qubits are involved in the operation. At the end of the circuit, the qubits that initially contain B will contain $| B |$ with the exception of the one that contains the sign (B_{N-1}, it maintains its value). The extra qubits, initialized to the specific states that will be detailed in this section, are unchanged after the computation of the circuit.

The proposed circuit is based on the methodology of the conditional adder presented by Coreas and Thapliyal [26]. Figure 2 shows an example of the conditional adder. The circuit of Coreas and Thapliyal performs the addition of two numbers A and B if the control qubit (*Ctrl* in Fig. 2) is set to 1. Otherwise, the circuit just returns the original numbers A and B. This conditional adder, with some minor changes, can be used to compute the absolute-value of a number. It has been explained that the absolute-value of a number B can be obtained keeping B unchanged if it is positive, or flipping its digits and adding 1 to it otherwise. If the digit that contains the sign of B is used as control qubit, that is, $Ctrl = B_{N-1}$, and A is set to 1, the circuit will compute the addition $B + 1$ if $B < 0$. However, B is required to previously flip its digits if it is negative. Such flips can be performed using $N - 1$ CNOT gates, setting B_{N-1} as the control qubit and $B_i (0 \leq i < N - 1)$ as the target ones. The adaptation of the conditional adder to perform the absolute-value of a number B is shown in Fig. 3, for the $N = 7$ case.

The circuit shown in Fig. 3 is fully functional. However, it is possible to subject it to several optimizations and to obtain a more efficient circuit in terms of operations, T-count, and required number of qubits. In the circuit shown in Fig. 3, there are several qubits with constant values (those that contain $A = 1$). Some operations can be simplified as the qubits that contain values $A_j (\forall j > 0)$ are set to 0. At the beginning of the conditional addition, there are $N - 2$ CNOT gates performing the operations $0 \oplus B_i$ $(0 < i < N - 1)$, being 0 the control qubit and B_i the target one. The result of such operations will be always B_i, so these CNOT gates can be removed. Those qubits, the ones that contains the 0 values acting as control qubits, also perform the operations $A_i \oplus A_{i+1}$ $(0 < i < N - 2)$. These operations are now irrelevant, so the $N - 3$ involved CNOT gates can be removed. All these CNOT operations are uncomputed at the end of the circuit, so the CNOT gates dedicated to the uncomputation can also be removed from the circuit. Then, a total of $4N - 10$ CNOT gates can be saved. On the other hand, several extra simplifications can be applied since A_0 is always 1. This qubit only acts as a target qubit in two Toffoli gates. Such

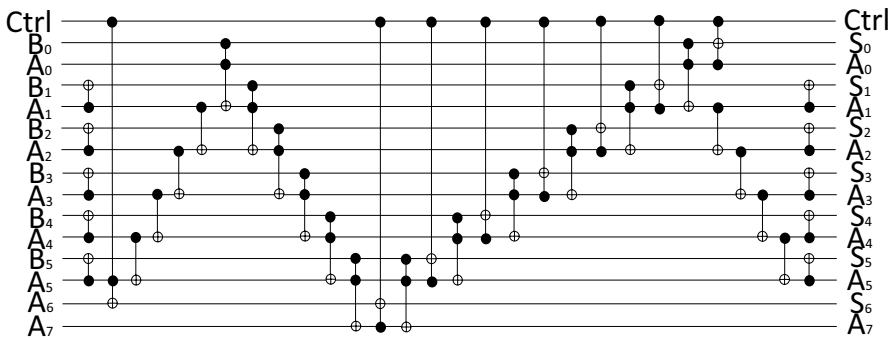

Fig. 2 Conditional adder proposed by Muñoz-Coreas and Thapliyal [26]. If $Ctrl = 1$, the circuit performs the addition between two integer numbers A and B (in this example, A has 8 digits and B 6 digits). Otherwise, it returns A and B without any changes. Therefore, qubits labelled as S will contain B if $Ctrl = 0$, and the addition of A and B if $Ctrl = 1$

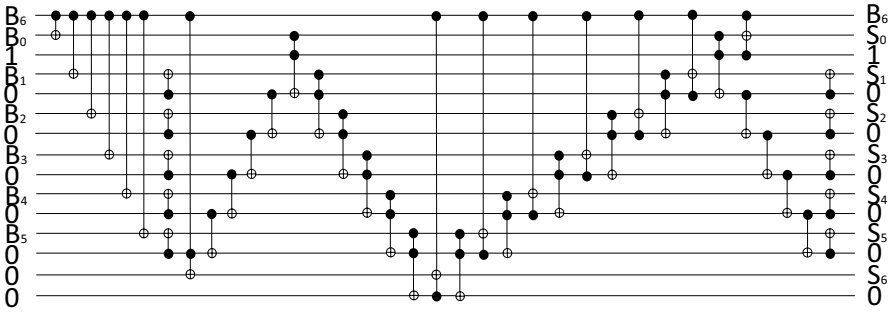

Fig. 3 First version of the proposed absolute-value circuit, for the $N = 7$ case. It is basically the conditional adder of Coreas and Thapliyal [26] (Fig. 3), setting $A = 1$ and being B the number whose absolute-value is to be obtained. The most-significant digit of B, which contains its sign, will be the control qubit that decides if the addition is computed. Previously to the conditional addition, the digits of B should be flipped if the number is negative. This can be easily performed using the control qubit and $N - 1$ CNOT gates. In short, the circuit does nothing if B is positive ($B_6 = 0$), and inverts the digits of B and adds 1 to it if B is negative. Here, S will contain the absolute-value of B. We keep the notation of the original circuit for the sake of clarity

Toffoli gates can be replaced by two CNOT gates. Moreover, the qubit A_0 is no longer needed. Taking into account all the simplifications listed above, the resulting circuit reduces the number of Toffoli gates by 2, the number of CNOT gates by $4N - 8$, and requires one less qubit. An example of this circuit, for the $N = 7$ case, is shown in Fig. 4.

Minor changes can be applied over the circuit shown in Fig. 4. The first ancilla qubit is always initialized to 0 ($A_1 = 0$). This qubit acts as a target qubit of the operation $B_0 \oplus 0$. After the operation, the target qubit will contain B_0, and the qubit will be used as one of the control qubits in a Toffoli operation. This qubit is not involved in other gate until the uncomputation of the mentioned operations. The operation $B_0 \oplus 0$ (that it, the CNOT gate) can be removed, and the qubit that initially contains B_0 can be used in the Toffoli gate as control qubit. Therefore, the

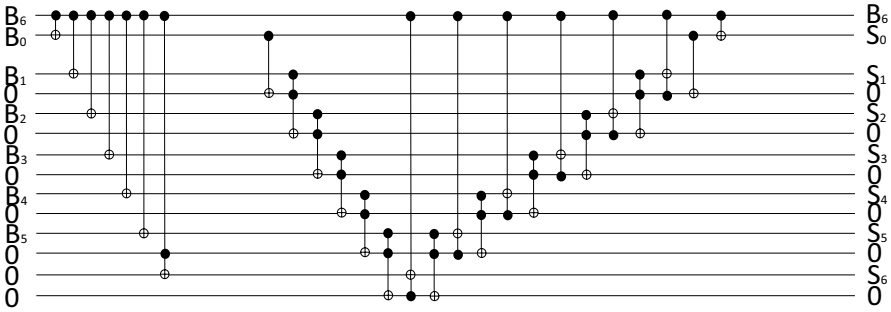

Fig. 4 Second version of the proposed absolute-value circuit, for the $N = 7$ case. Since the least significant digit of A is always 1, several operations can be simplified. The qubit that contains 1 (A_0 in the original circuit) can also be removed

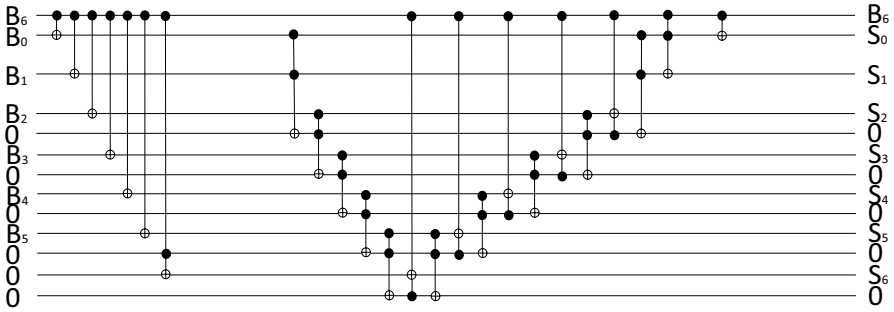

Fig. 5 Example, for the $N = 7$ case, of the third version of the proposed absolute-value circuit. The qubit A_1 is always 0, so again several operations can be simplified and the qubit removed

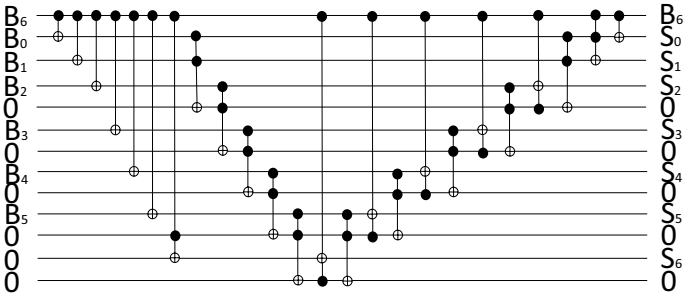

Fig. 6 Well-organised design of the third version for numbers of 7 digits. This circuit is the same circuit as the one shown in Fig. 5, but cleaning up the gaps and spaces between qubits and gates

ancilla qubit is not necessary, with an extra save of two CNOT gates. An example of the resulting circuit, for the $N = 7$ case, is shown in Fig. 5. For the sake of clarity, this figure (and the previous ones) shows the gaps left by the gates and qubits that have been removed. A cleaner, more compact version of the circuit is shown in Fig. 6.

In the circuit of Fig. 6, there are $N - 2$ Toffoli gates simulating a $(N - 1)$-qubit Toffoli gate. Later, the same number of Toffoli gates are applied to uncompute the previous ones. Such gates acts over target qubits initialized to 0 (the uncomputation gates revert them into 0). These Toffoli gates can be replaced by Temporary logical-AND gates (and their corresponding uncomputation gates) without increasing the number of necessary qubits but halving the T-count of the circuit. That is, $N - 2$ Toffoli gates are replaced by $N - 2$ Temporary logical-AND gates, and $N - 2$ Toffoli gates are replaced by $N - 2$ uncomputation gates of the Temporary logical-AND operation. These changes reduces the T-count from $14N - 28$ into $4N - 8$. On the other hand, the first Toffoli gate of the circuit performs the operation $CtrlA_{N-3} \oplus A_{N-2}$ in the original circuit, which is the equivalent to compute now $B_{N-1}0 \oplus 0$. Then, this Toffoli gate can be removed, saving a T-count of 7. As a consequence, the Toffoli gate that computes the most significant digit of

the absolute-value now has a target qubit that is initialized to 0. Therefore, this Toffoli gate can be replaced by a Temporary logical-AND (saving a T-count of 3). The remaining Toffoli gates cannot be replaced as they act over qubits that contain dependent values. An example of the final circuit, for the $N = 7$ case, is shown in Fig. 7.

The described evolution can be summarised as follows:

- Original version (Fig. 2): the circuit proposed by Muñoz-Coreas and Thapliyal.
- Version 1 (Fig. 3): the original version, simply making $A = 1$ and being B the two's complement number whose absolute value is to be calculated. It is necessary to extract the most significant digit from this number, which contains the sign, and to use it as a control qubit of the addition (Ctrl).
- Version 2 (Fig. 4): to eliminate qubit A_0, knowing that it is always set to $|1\rangle$. It only acts as a control qubit in some Toffoli gates, so these Toffoli gates can become CNOT gates dependent on the other control qubit.
- Version 3 (Fig. 5 and 6): to remove qubit A_1 knowing that it is always set to $|0\rangle$. This qubit acts as the target qubit of a CNOT gate, and then acts as the control qubit in a Toffoli gate. The control qubit of the CNOT gate replaces A_1 as the control qubit in the Toffoli gate.
- Version 4 (Fig. 7): Qubits initialized to $|0\rangle$ are replaced by qubits initialized in the state $(\frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{i\pi}{4}}|1\rangle)$. Toffoli gates that had an auxiliary qubit in state $|0\rangle$ as a target qubit are replaced by temporary logical-AND gates. Toffoli gates that reverted to the replaced Toffoli gates are in turn replaced by uncomputation gates of the temporary logical-AND gate.

3.1 Steps of design methodology

1. To prepare N qubits to represent B.
2. To prepare $N-1$ ancilla qubits in the state $\frac{1}{\sqrt{2}}(|0\rangle + e^{\frac{i\pi}{4}}|1\rangle)$.
3. For $i = 0$ to $N-2$, a CNOT gate is applied to perform the operation $B_{N-1} \oplus B_i$. This operation flips the digits of B only if $B < 0$.

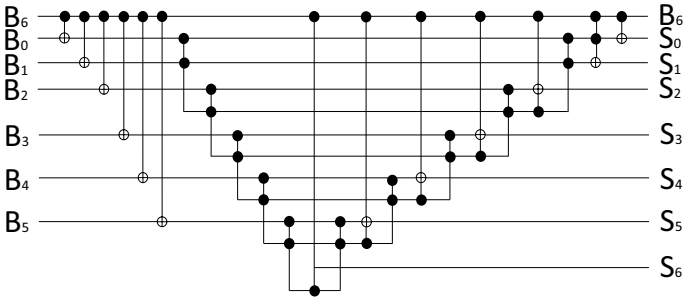

Fig. 7 Final version of the proposed circuits, for the $N = 7$ case. Several Toffoli gates have been replaced by Temporary logical-AND operations, halving the cost of the circuit shown in Fig. 6

4. A Temporary logical-AND is applied to perform the operation B_0B_1. The result of the operation is temporarily stored in an ancilla qubit a_0.
5. For $i = 2$ to $N - 2$, a temporary logical-AND is applied to perform the operation $a_{i-2}B_i$. The result of the operation is temporarily stored in an ancilla qubit a_{i-1}.
6. A Temporary logical-AND gate is used to compute $B_{N-1}a_{N-3}$. The result is saved in an ancilla qubit a_{N-2}, and it is the most significant digit of the absolute-value (excluding the sign, which is always assumed to be positive).
7. For $i = N - 2$ to 2, an uncomputation gate of the Temporary logical-AND operation is applied to uncompute the operation $a_{i-2}B_i$, that is, to uncompute a_{i-1}. Then, a Toffoli gate must be applied to compute $a_{i-2}B_{N-1} \oplus B_i$. The target qubit will contain the i-digit of the absolute-value.
8. An uncomputation gate of the Temporary logical-AND operation is applied to revert the product B_0B_1 (that is, a_0). Then, a Toffoli gate computing $B_6B_0 \oplus B_1$, and a CNOT gate performing $B_6 \oplus B_0$, are applied to compute the least-significant digits of the result.

4 Results

The number of necessary gates to build the proposed circuit can be easily checked by going through the steps described in Sect. 3.1:

- No gates are needed in steps 1 and 2.
- Step 3 involves $N - 1$ CNOT gates.
- Step 4 needs one Temporary logical-AND gate.
- Step 5 computes $N - 3$ Temporary logical-AND gates.
- Step 6 involves one Temporary logical-AND gate.
- Step 7 computes $N - 3$ uncomputation gates of the Temporary logical-AND gate. $N - 3$ Toffoli gates are also applied.
- Step 8 involves one uncomputation gate of the Temporary logical-AND operation. A Toffoli gate and a CNOT gate are also applied.

Overall, the proposed circuit needs N CNOT gates, $N - 2$ Toffoli gates, $N - 1$ Temporary logical-AND gates, and $N - 2$ uncomputation gates of the Temporary logical-AND operation. The original circuit, the conditional-adder of Muñoz-Coreas and Thapliyal [26], involves $5N - 11$ CNOT gates, and $3N - 1$ Toffoli gates (extra costs are added to compute the absolute-value operation using this circuit, as it is shown in Fig. 3). The proposed circuit reduces the number of CNOT gates in $4N - 11$ with respect to the original circuit, and the number of Toffoli gates in $2N + 1$. The saved Toffoli gates are replaced by $N - 1$ extra Temporary logical-AND gates, and $N - 2$ uncomputation gates (of the Temporary logical-AND gate). This will have a very positive impact on the T-count, as explained below.

The T-count of the proposed circuit can be obtained from the number of Toffoli and Temporary logical-AND gates it involves. The remaining gates (CNOT and uncomputation gates) do not contain any T gates. Therefore, the circuit has a T-count of $11N - 18$. On the other hand, it is trivially obtained through steps 1 and

Table 2 Comparison, in terms of T-count and number of qubits, between the proposed circuit and the best designs in the literature

Circuit	T-count	Number of qubits
Lin et al. [46]	$56N$	$4N + 2$
Jayashree et al. [45]	$28N + 7$	$2N + 3$
Markov and Saeedi [44]	$28N + 7$	$2N + 3$
Muñoz-Coreas and Thapliyal [26]	$21N - 7$	$2N + 1$
Proposed circuit	$11N - 18$	$2N - 1$

Such designs are conditional adders adapted for the computation of the absolute value. This adaptation implies extra costs in terms of operations that, however, do not affect either the T-count or the number of qubits

2 of Sect. 3.1 that the circuit involves $2N - 1$ qubits. Table 2 shows a comparison in terms of T-count and number of qubits between the proposed circuit and the best conditional-adders (acting as absolute-value circuits) in the literature. Since there are no specific circuits in the literature for the absolute value of two's complement integers, the comparison only includes such conditional-adders. Our proposal halves the T-count with respect the best circuit in the state-of-the-art: from $21N - 7$ to $11N - 18$. Given the high cost of the T gate, this reduction represents a significant saving. In terms of qubits, our proposal also manages to reduce their number. It needs two fewer qubits than the circuit of Muñoz-Coreas and Thapliyal [26], four fewer qubits than the circuits of Markov and Saeedi [44] and Jayashree et al [45], and half as many qubits as circuit of Lin et al. [46].

5 Conclusions

We have presented a circuit to perform the absolute-value of signed integer numbers (in two's complement representation) of any size in quantum computers. The proposed circuit is optimized in terms of number of operations, T-count, and number of qubits. The circuit is based in the methodology of a conditional-adder. Its design has been described in detail, showing each necessary step, quantum gate, and qubit. It has been built using exclusively Clifford+T gates.

Finally, a comparison has been made in terms of T-count and number of qubits between the proposed circuit and the most important circuits available in the literature. In this comparison, it can be seen that our circuit improves on the others in terms of number of qubits and, what is more, halves the T-count with respect to these circuits.

Author contributions EFC: Orchestration, Idea source, Writing—original draft; IFR: Idea source, Formal analysis, Writing—original draft; AMP: Idea source, Formal analysis, Writing—original draft; EMG: Writing—review an editing, Validation, Visualization; GO: Writing—review and editing, Validation, Visualization; FO: Writing—original draft, Software, Execution of experiments.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This work has been supported by the projects: PID2021-123278OB-I00 and PID2021-127836NB-I00 (funded by MCIN/AEI/10.13039/501100011033/ FEDER "A way to make Europe"); projects PID2020-119082RB-C22 (funded by MCIN/AEI/10.13039/501100011033); AYUD/2021/50994 and FC-GRUPIN-IDI/2018/000193 (funded by Gobierno del Principado de Asturias); MTM-2017-83506-C2-2-P (funded by MINECO); PID2021-123461NB-C22 (funded by MICIIN); and projects P20_00748, UAL2020-TIC-A2101, and UAL18-TIC-A020-B (funded by Junta de Andalucía and the European Regional Development Fund, ERDF). Funding for open access publishing: Universidad de Almería/CBUA.

Data availability Not applicable.

Declarations

Conflict of interest The authors declare that they have no competing interests.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Nielsen MA, Chuang I (2002) Quantum computation and quantum information. American Association of Physics Teachers
2. Shor PW (1994) Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science, pp. 124–134. IEEE
3. Grover LK (1997) Quantum mechanics helps in searching for a needle in a haystack. Phys Rev Lett 79(2):325
4. Gilliam A, Woerner S, Gonciulea C (2021) Grover adaptive search for constrained polynomial binary optimization. Quantum 5:428
5. Thapliyal H, Muñoz-Coreas E, Khalus V (2021) Quantum circuit designs of carry lookahead adder optimized for T-count T-depth and qubits. Sustain Comput Inform Syst 29:100457
6. Gayathri S, Kumar R, Dhanalakshmi S, Dooly G, Duraibabu DB (2021) T-count optimized quantum circuit designs for single-precision floating-point division. Electronics 10(6):703
7. Li H-S, Fan P, Xia H, Long G-L (2022) The circuit design and optimization of quantum multiplier and divider. Sci China Phys Mech Astron 65(6):1–15
8. Asadi M-A, Mosleh M, Haghparast M (2020) An efficient design of reversible ternary full-adder/full-subtractor with low quantum cost. Quant Inform Process 19(7):1–21
9. Mohammadi M, Eshghi M (2009) On figures of merit in reversible and quantum logic designs. Quant Inform Process 8(4):297–318
10. Thapliyal H, Muñoz-Coreas E (2019) Design of quantum computing circuits. IT Prof 21(6):22–26
11. Gidney C (2018) Halving the cost of quantum addition. Quantum 2:74
12. Yan F, Iliyasu AM, Guo Y, Yang H (2018) Flexible representation and manipulation of audio signals on quantum computers. Theor Comput Sci 752:71–85

13. Wong R, Chang W-L (2022) Fast quantum algorithm for protein structure prediction in hydrophobic-hydrophilic model. J Parall Distrib Comput 164:178–190
14. Nakaji K, Uno S, Suzuki Y, Raymond R, Onodera T, Tanaka T, Tezuka H, Mitsuda N, Yamamoto N (2022) Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicators. Phys Rev Res 4(2):023136
15. Hennessy JL, Patterson DA (2011) Computer architecture: a quantitative approach. Elsevier, US
16. Bernhardt C (2019) Quantum computing for everyone. Mit Press, US
17. Paler A, Polian I, Nemoto K, Devitt SJ (2017) Fault-tolerant, high-level quantum circuits: form, compilation and description. Quant Sci Technol 2(2):025003
18. Devitt SJ, Stephens AM, Munro WJ, Nemoto K (2013) Requirements for fault-tolerant factoring on an atom-optics quantum computer. Nat Commun 4(1):1–8
19. Amy M, Maslov D, Mosca M, Roetteler M (2013) A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Trans Comput Aided Des Integr Circ Syst 32(6):818–830
20. Orts F, Ortega G, Combarro EF, Garzón EM (2020) A review on reversible quantum adders. J Netw Comput Appl 170:102810
21. Orts F, Ortega G, Filatovas E, Garz´on ME (2022) Implementation of three efficient 4-digit fault-tolerant quantum carry lookahead adders. J Supercomput 78:1–19
22. Kotiyal S, Thapliyal H, Ranganathan N (2014) Circuit for reversible quantum multiplier based on binary tree optimizing ancilla and garbage bits. In: 2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems, pp. 545–550 https://doi.org/10.1109/VLSID.2014.101
23. Islam MS, Rahman M, Begum Z, Hafiz MZ (2009) Low cost quantum realization of reversible multiplier circuit. Inform Technol J 8(2):208–213
24. Haghparast M, Mohammadi M, Navi K, Eshghi M (2009) Optimized reversible multiplier circuit. J Circ Syst Comput 18(02):311–323
25. Babu HM et al (2017) Cost-efficient design of a quantum multiplier-accumulator unit. Quant Inform Process 16(1):1–38
26. Muñoz-Coreas E, Thapliyal H (2018) Quantum circuit design of a T-count optimized integer multiplier. IEEE Trans Comput 68(5):729–739
27. Sajadimanesh S, Faye, JPL, Atoofian E (2022) Practical approximate quantum multipliers for nisq devices. In: Proceedings of the 19th ACM International Conference on Computing Frontiers, pp. 121–130
28. Dibbo SV, Babu HMH, Jamal L (2016) An efficient design technique of a quantum divider circuit. In: 2016 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2102–2105. IEEE
29. Thapliyal H, Munoz-Coreas E, Varun T, Humble TS (2019) Quantum circuit designs of integer division optimizing t-count and t-depth. IEEE Trans Emerg Top Comput 9(2):1045–1056
30. Khosropour A, Aghababa H, Forouzandeh B (2011) Quantum division circuit based on restoring division algorithm. In: 2011 Eighth International Conference on Information Technology: New Generations, pp. 1037–1040. IEEE
31. Yuan S, Gao S, Wen C, Wang Y, Qu H, Wang Y (2022) A novel fault-tolerant quantum divider and its simulation. Quant Inform Process 21(5):1–15
32. Jamal L, Babu HMH (2013) Efficient approaches to design a reversible floating point divider. In: 2013 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 3004–3007. IEEE
33. Gyongyosi L, Imre S (2019) Quantum circuit design for objective function maximization in gate-model quantum computers. Quant Inform Process 18(7):1–33
34. Zhou R, Wan C (2021) Quantum image scaling based on bilinear interpolation with decimals scaling ratio. Int J Theoret Phys 60(6):2115–2144
35. Wang S, Wang Z, Li W, Fan L, Cui G, Wei Z, Gu Y (2020) Quantum circuits design for evaluating transcendental functions based on a function-value binary expansion method. Quant Inform Process 19(10):1–31
36. Murali K, Sinha N, Mahesh TS, Levitt M, Ramanathan KV, Kumar A (2002) Quantum-information processing by nuclear magnetic resonance: experimental implementation of half-adder and subtractor operations using an oriented spin-7/2 system. Phys Rev A. https://doi.org/10.1103/PhysRevA.66.022313
37. Cheng K-W, Tseng C-C (2002) Quantum full adder and subtractor. Electron Lett 38(22):1343–1344

38. Thapliyal H, Ranganathan N (2011) A new design of the reversible subtractor circuit. In: 2011 11th IEEE International Conference on Nanotechnology, pp. 1430–1435. IEEE
39. Maity H, Biswas A, Pal A, Bhattacharjee AK (2018) Quantum cost optimized design of reversible 2's complement code converter. In: 2018 IEEE Electron Devices Kolkata Conference (EDKCON), pp. 122–125. IEEE
40. Orts F, Ortega G, Garzón E (2019) An optimized quantum circuit for converting from sign-magnitude to two's complement. Quant Inform Process 18(11):332
41. Orts F, Ortega G, Garzon EM (2020) Efficient reversible quantum design of sign-magnitude to two's complement converters. Quant Inform Computat 20(9–10):747–765
42. Harris SL, Harris D (2015) Digital design and computer architecture. Morgan Kaufmann, US
43. Patterson DA, Hennessy JL (2013) Computer organization and design mips edition: the hardware/software interface. Newnes, US
44. Markov IL, Saeedi M (2012) Constant-optimized quantum circuits for modular multiplication and exponentiation. arXiv preprint arXiv:1202.6614
45. Jayashree H, Thapliyal H, Arabnia HR, Agrawal VK (2016) Ancilla-input and garbage-output optimized design of a reversible quantum integer multiplier. J Supercomput 72(4):1477–1493
46. Lin C-C, Chakrabarti A, Jha NK (2014) Qlib: quantum module library. ACM J Emerg Technol Comput Syst (JETC) 11(1):1–20

Authors and Affiliations

Francisco Orts[1] · Gloria Ortega[1] · Elías F. Combarro[2] · Ignacio F. Rúa[3] · Antonio M. Puertas[4] · Ester M. Garzón[1]

Gloria Ortega
gloriaortega@ual.es

Elías F. Combarro
efernandezca@uniovi.es

Ignacio F. Rúa
rua@uniovi.es

Antonio M. Puertas
apuertas@ual.es

Ester M. Garzón
gmartin@ual.es

[1] Informatics Department, Agrifood Campus of International Excellence (ceiA3), University of Almería, Almería, Spain

[2] Informatics Department, University of Oviedo, Oviedo, Spain

[3] Mathematics Department, University of Oviedo, Oviedo, Spain

[4] Department of Chemistry and Physics, University of Almería, Almería, Spain