Journal of
Quantum Computing

Tech Science Press

DOI: 10.32604/jqc.2022.034059
Article

Design of a Novel Signed Binary Subtractor Using Quantum Gates

Arindam Banerjee[1,*], Aniruddha Ghosh[2] and Mainuck Das[2]

[1]ICFAI University Tripura, Kamalghat, Agartala, West Tripura, India
[2]JIS College of Engineering, Kalyani, Nadia, West Bengal, 741235, India
*Corresponding Author: Arindam Banerjee. Email: banerjee.arindam1@gmail.com
Received: 05 July 2022; Accepted: 07 April 2023; Published: 03 July 2023

Abstract: In this paper, focus has been given to design and implement signed binary subtraction in quantum logic. Since the type of operand may be positive or negative, therefore a novel algorithm has been developed to detect the type of operand and as per the selection of the type of operands, separate design techniques have been developed to make the circuit compact and work very efficiently. Two separate methods have been shown in the paper to perform the signed subtraction. The results show promising for the second method in respect of ancillary input count and garbage output count but at the cost of quantum cost.

Keywords: Signed subtraction; reversible logic; quantum gates

1 Introduction

From decades ago, computer system works in binary arithmetic. The binary arithmetic, in the computing system, has been implemented using standard CMOS, Pass Transistor and Bi-CMOS technologies. In case of standard CMOS based implementation, speed improvement can not be achieved. After that FinFET has been introduced [1] for faster operation and it provides better sub-threshold swing. FinFET also provides better gain for the amplifiers. But in VLSI technology, the optimization of speed and power is a major challenge because to improve speed, power consumption must be increased. Therefore, some alternate technology to solve this issue has been thought of by the scientists. Reversible logic is considered to be a good solution to this issue because both speed and power consumption can be optimized in reversible logic system [2–5].

In [2], Landauer has explained the heat generation in computational process. In [3–5], Bennett has described the logical process of reversibility. In the year 1980, Toffoli [6] has proposed reversible gates for computation using Boolean logic. In the year 1981, Fredkin et al. [7] described the conservative logic in the reversible system. In 1985, Peres [8] has introduced a new reversible gate to synthesize Boolean functions. After few years, Deutsch [9], in 1985, has described the basic principles of quantum theory. In the year 2000, different quantum computing algorithms have been described by Nielsen et al. [10]. After that, several synthesis techniques for implementing Boolean functions in reversible logic have been reported in [11–18]. In [19], Moraga has described the use of double gates in reversible logic. The works in [12–14] show the arithmetic circuits implementation. In [12], a quantum

ripple carry adder design has been reported. In [13–15], Thapliyal have proposed binary subtraction methods. But those methods are for unsigned binary number system.

In this paper, we propose another subtraction technique but the subtraction technique is for signed number system. The technique is new to the best of the knowledge of the authors and thus beyond the scope of comparison with other reported works.

2 Brief Overview of Basic Quantum Gates used for Arithmetic Operations

Basic quantum gates are associated with qubit operation. As defined in [9], Qubit is basically quantum state which is analogous to bit in binary number system. In binary number system, bit has two values logic 0 and logic 1 but in qubit, specific value may not be assigned since quantum states are probabilistic. Qubit can be written as $|Q>= a|0> +b|1>$ where $|>$ is called the Dirac notation and a and b are complex numbers and $|a|^2 + |b|^2 = 1$ which means that $|0>$ may appear with the probability $|a|^2$ and $|1>$ may appear with the probability $|b|^2$. In matrix form, $|0>= \begin{bmatrix}1\\0\end{bmatrix}$ and $|1>= \begin{bmatrix}0\\1\end{bmatrix}$. Therefore, $|Q>$ can be written as $|Q>= a|0> +b|1>= \begin{bmatrix}a\\b\end{bmatrix}$. The quantum gates which can process the information must have some characteristics matrix. The characteristics matrix must be unitary [9] by nature. A matrix (G) is called unitary if $GG^{\dagger} = I$ where, $G^{\dagger}$ is the conjugate transpose of the matrix G [9]. For example, if a gate G has the characteristics matrix $G = \begin{bmatrix}a+ib & c+id\\ e+if & k+il\end{bmatrix}$ where, $i = \sqrt{-1}$, then its conjugate transpose is $G^{\dagger} = \begin{bmatrix}a-ib & e-if\\ c-id & k-il\end{bmatrix}$. Then, $GG^{\dagger} = \begin{bmatrix}a+ib & c+id\\ e+if & k+il\end{bmatrix}\begin{bmatrix}a-ib & e-if\\ c-id & k-il\end{bmatrix} = \begin{bmatrix}a^2+b^2+c^2+d^2 & (a+ib)(e-if)+(c+id)(k-il)\\ (a-ib)(e+if)+(c-id)(k+il) & e^2+f^2+k^2+l^2\end{bmatrix}$ and if it is equal to the identity matrix $I = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ then, $a^2+b^2+c^2+d^2 = 1, e^2+f^2+k^2+l^2 = 1, (a+ib)(e-if)+(c+id)(k-il) = 0$ and $(a-ib)(e+if)+(c-id)(k+il) = 0$ which are the conditions for the matrix G to be unitary.

Fig. 1 shows the basic quantum gates. The solid circle notation indicates control input and the circled plus notation indicates the target input. The first gate is NOT or Feynman gate. It has the characteristics matrix $NOT = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$. Here for each gate, the column index indicates input combination and the row index indicates output combination. The second gate is CNOT or Controlled NOT gate. It has the characteristics matrix $CNOT = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}$. The third gate is Toffoli gate. A Toffoli gate may have more number of inputs. Here in Fig. 1, the characteristics matrix

for 3 input Toffoli gate is $Toffoli = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$. The fourth gate is Peres gate. To compute its characteristics matrix, at first matrix representation at the output side must be done. It is basically a CNOT gate combining with no operation Identity gate. So the matrix operation at the output side is the Tensor product of the CNOT and Identity matrix i.e., $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$. Next the matrix is multiplied with the Toffoli matrix i.e., $Peres = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$. The next gate is double Peres gate which is the combination of two Peres gates. The next gate is the Fredkin gate which is basically a controlled swap gate. The characteristics matrix of the Fredkin gate is $Fredkin = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$. To compute output, gate matrices are multiplied from the output side to the input side. Now few definitions related to reversible logic are important to mention such as ancillary input, garbage output and quantum cost.

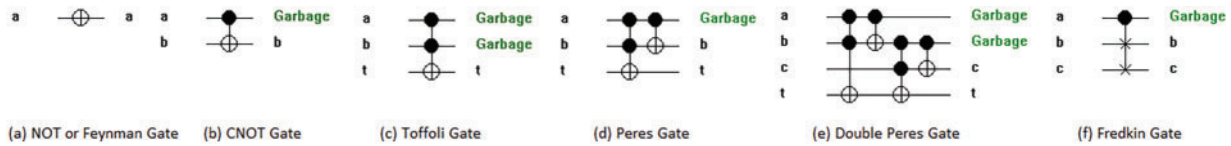

Figure 1: Basic quantum gates

Ancillary Input: It is defined as the extra constant input other than the primary inputs to evaluate the logic function.

Garbage Output: It is defined as the unused output other than the primary outputs to maintain logical reversibility. Garbage outputs are redundant outputs but they must be generated. If p_i is the number of primary inputs, c_i is the number of ancillary inputs and p_o is the number of primary outputs then the number of garbage outputs $g_o = p_i + c_i - p_o$.

Quantum Cost: It is the estimation of computational complexity of the circuit. It is defined as the number of quantum gates involved in the circuit implementation.

Fig. 2 shows a quantum circuit which represents logical OR operation. The first two gates combine to form a single Peres gate. As per the Boolean logic, the output at the third line is equal to $ab \oplus a \oplus b = a \oplus \bar{a}b = \bar{a}b + a\left(\overline{\bar{a}b}\right) = \bar{a}b + a\left(a + \bar{b}\right) = \bar{a}b + a = a(1 + b) + \bar{a}b = a + b(a + \bar{a}) = a + b$ which indicates an OR operation. Here an extra input (0) has been taken which is the ancillary input. The number of garbage outputs is 2. The circuit shown in Fig. 2 can be decomposed into quantum gates as shown in Fig. 3 using the decomposition technique described in [8].

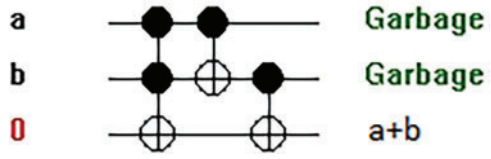

Figure 2: A quantum circuit representing OR operation

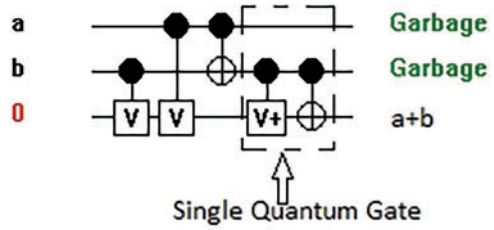

Figure 3: Decomposition of OR gate into unitary quantum gates

In Fig. 3, basically single Peres gate has been decomposed into unitary quantum gates based on the technique shown in [8]. Here few new quantum gates have been used V and V+ gates. V and V+ gates are represented as follows.

$$V = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} \tag{1}$$

$$V^{+} = \frac{1-i}{2}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} \tag{2}$$

where $i = \sqrt{-1}$. V and V+ gates are the square root NOT gate i.e., $V^2 = V^{+2} = NOT$. The proof is given below.

Lemma-1: $V^2 = V^{+2} = NOT$

Proof: Since, $V = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}$ then $V^2 = \left(\frac{1+i}{2}\right)^2 \times \begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}^2 = \frac{i}{2} \times \begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} \times \begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} = \frac{i}{2} \times \begin{bmatrix} 0 & -2i \\ -2i & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = NOT.$

Similarly, Since, $V^{+} = \frac{1-i}{2}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}$ then $V^{+2} = \left(\frac{1-i}{2}\right)^2 \times \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}^2 = -\frac{i}{2} \times \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} \times \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} = -\frac{i}{2} \times \begin{bmatrix} 0 & 2i \\ 2i & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = NOT$. (Thus proved).

Similar proof can be done for controlled V and V+ gates i.e., $CV^2 = CV^{+2} = CNOT$. The controlled V+ gate and the adjacent CNOT gate forms another quantum gate and so the combination is considered to be a single quantum gate as indicated in Fig. 3. This statement can also be proved as shown below.

Lemma-2: One V or V+ gate and one NOT gate can be combined to generate a single quantum gate

Proof: We know that $V = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}$ and $NOT = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. Thus, $NOT \times V = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix} = \begin{bmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix}$. Clearly, it is a unitary matrix and hence represents a quantum gate.

Similarly, $V^{+} = \frac{1-i}{2}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}$. Thus, $NOT \times V^{+} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \frac{1-i}{2}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix} = \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix}$. Now let $G = \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix}$.Thus, $G^{\dagger} = \begin{bmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix}$. Therefore, $GG^{\dagger} = \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix} \begin{bmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$. Hence the gate represents a quantum gate. (Thus proved).

Therefore there are 4 unitary gates and so the quantum cost is 4.

3 Mathematical Description of Signed Subtraction

Consider numbers A and B with extra sign bits (SignA and SignB). A and B can be represented as follows.

$$A = \sum_{i=0}^{n-1} a_i 2^i \tag{3}$$

$$B = \sum_{i=0}^{n-1} b_i 2^i \tag{4}$$

Similarly, (−A) and (−B) can be represented in 2's complement form as follows.

$$-A = 2^n - A \tag{5}$$

$$-B = 2^n - B \tag{6}$$

The following table (Table 1) shows the subtraction operation with sign bits. Here SignA and SignB are the sign bits of the operand A and B.

Table 1: Signed binary operation

SignA (0 indicates positive and 1 indicates negative)	SignB (0 indicates positive and 1 indicates negative)	Operation
0	0	$A - B$
0	1	$A-(2^n - B) = (A + B)-2^n$, (A+B) and discard carry
1	0	$(2^n - A) - B = 2^n - (A + B)$, 2's complement of (A + B)
1	1	$(2^n - A)-(2^n - B) = B-A$, 2's complement of (A − B)

4 Reversible Circuit Implementation for Signed Binary Subtraction

The signed subtraction technique shown in Table 1 has been verified using VHDL programming. The detailed code is shown in Fig. 4 and the verified output is shown in Fig. 5. The first two lines of the code are basically library declaration. Next part is the entity declaration which declares the input and the output signals. In the architecture part, there are two processes as shown in Fig. 4; one for addition or subtraction based on the signals (SignA and SignB); and the second for the 2's complement operation based on the sign signals (SignA and SignB). In the first process of the code shown in Fig. 4, addition or subtraction has been performed based on the signs of the two operands (SignA and SignB). In the second process, the 2's complement operation has been performed. It is a mixed style of the VHDL code because both dataflow and behavioral approaches have been used. For addition or 2's complement operation, Boolean expressions, for half and full adder, have been used. As shown in Table 1, the 2's complement operation has been performed based on the signal (SignA). Though the circuit has been implemented using reversible logic still VHDL implementation has been shown here to confirm the correctness of the procedure so that accurate circuit can be implemented in reversible logic.

```
library ieee;
  use ieee.std_logic_1164.all;
  entity signed_binary_subtraction  is
    port(signa,signb : in std_logic;
       a,b : in std_logic_vector(3 downto 0);
       y : out std_logic_vector(4 downto 0));
  end signed_binary_subtraction;
  architecture behave of signed_binary_subtraction is
    signal val: std_logic_vector(4 downto 0);
    begin
      process(signa,signb,a,b)
        variable sig : std_logic;
        variable moda : std_logic_vector(3 downto 0);
        variable bin: std_logic_vector(4 downto 0);
        begin
          sig := not (signa xor signb);
          for i in 0 to 3 loop
          moda(i) := sig xor b(i);
          end loop;
          bin(0) := sig;
          for i in 0 to 3 loop
          val(i) <= a(i) xor moda(i) xor bin(i);
          bin(i+1) := (a(i) and moda(i)) xor ((a(i) xor moda(i)) and bin(i));
          end loop;
          val(4) <= bin(4);
      end process;
      process(signa,val)
        variable moda : std_logic_vector(3 downto 0);
        variable bin: std_logic_vector(4 downto 0);
        begin
          for i in 0 to 3 loop
          moda(i) := signa xor val(i);
          end loop;
          bin(0) := signa;
          for i in 0 to 3 loop
          y(i) <= moda(i) xor bin(i);
          bin(i+1) := moda(i) and bin(i);
          end loop;
          y(4) <= val(4);
      end process;
  end behave;
```

Figure 4: Detailed VHDL code for binary subtraction (for 4 bit)

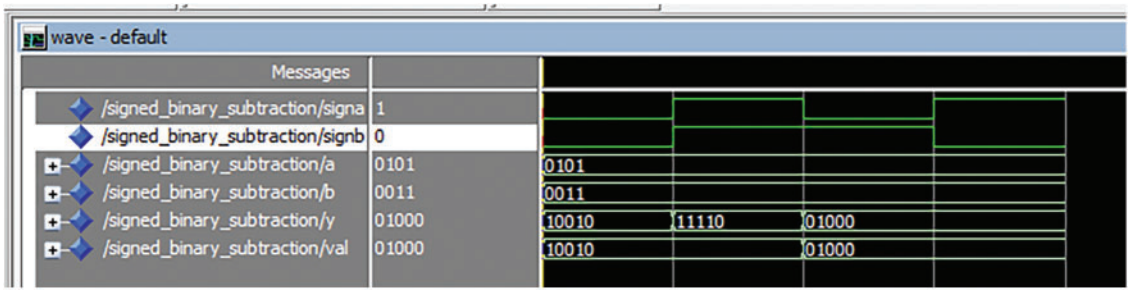

Figure 5: Waveform using Modelsim simulator for verification of the binary subtraction algorithm (for 4 bits)

The implementation scheme has two major parts: (i) addition or subtraction part and (ii) 2's complement part. The first part (addition or subtraction) has been implemented using two types of addition methods. Figs. 6 and 7 show the two methods. Both the methods are generic and can be extended for any bit length. Like Fig. 5, "signa" and "signb" are the two signals which indicate the sign bits for the two operands as shown in Figs. 6 and 7. The outputs "Dif0" to "Dif3" are the results (here 4 bit operation has been shown). The signal "Sign" is the sign bit for the output. For the reversible circuit implementation, shown in Fig. 6, double Peres gates have been used for addition. For the circuit, shown in Fig. 7, the addition technique, as shown in [12], has been used.

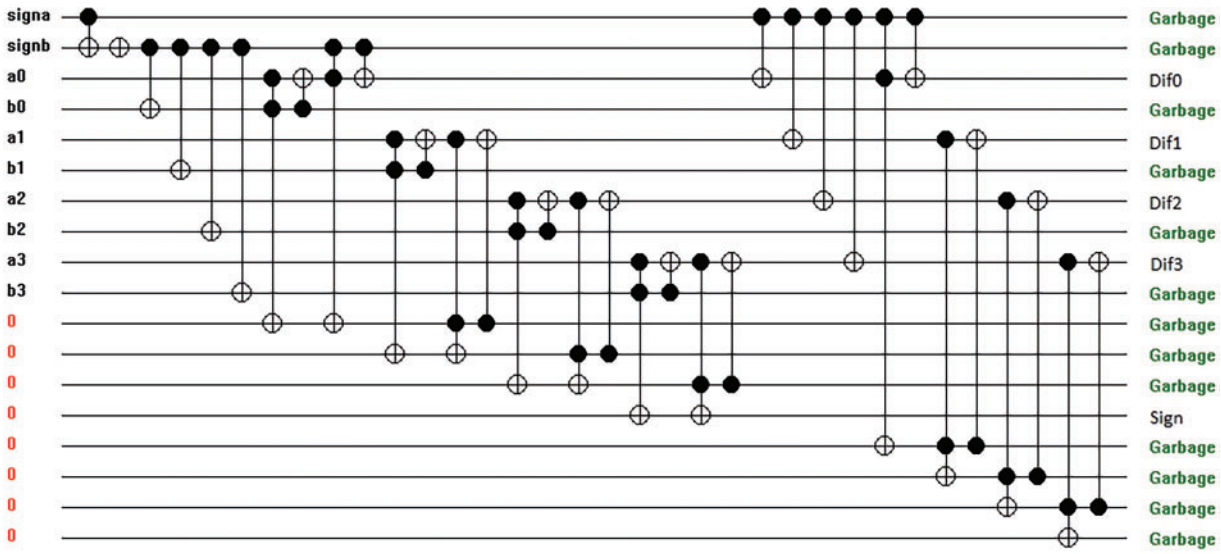

Figure 6: Reversible circuit for signed binary subtraction (4 bit) (First Method)

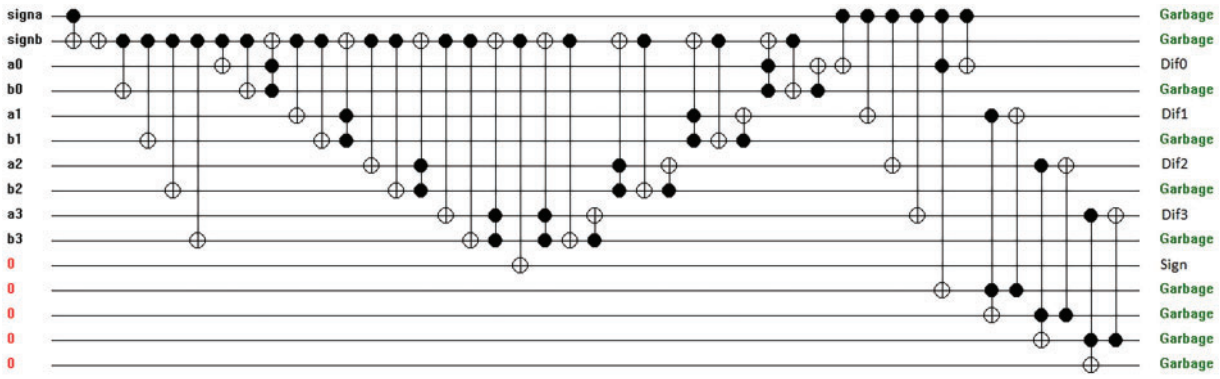

Figure 7: Reversible circuit for signed binary subtraction (4 bit) (Second Method)

5 Result Analysis

Ancillary Input Count:

First Method: Using Fig. 6, a generalized circuit for n-bit subtraction can be designed. The first part of the circuit is the addition or subtraction part. For n-bit operation, n number of ancillary inputs is required for addition or subtraction. The second part is for 2's complement operation. It also requires n number of ancillary inputs. Therefore total $(n + n) = 2n$ number of ancillary inputs is required.

Second Method: In the second method, the first part (addition or subtraction) is replaced by a new adder circuit proposed in [12]. The adder circuit drastically reduces the number of ancillary inputs. Only 1 ancillary input is required for the addition. Therefore total $(n + 1)$ number of ancillary inputs is required.

Garbage Output Count:

First Method: Number of ancillary inputs is $2n$. Number of primary inputs is $2n + 2$. Number of primary outputs is $n + 1$. Therefore, the number of garbage outputs is $2n + 2 + 2n - (n + 1) = 3n + 1$.

Second Method: In the second method, $2n + 2 + (n + 1) - (n + 1) = 2n + 2$ number of garbage outputs is required.

Quantum Cost Count:

First Method: For arithmetic operation, here, single Peres and double Peres gates have been used. Decomposition of Peres gate and double Peres gate into quantum gates is shown in Figs. 8 and 9. The decomposition technique of double Peres gate, shown in Fig. 9, follows the technique in [19] proposed by Moraga. For single Peres gate, the quantum cost is 4 and for double Peres gate, the quantum cost is 6 as shown in Figs. 8 and 9. For the design of half adder, single Peres gates are used which can be decomposed into quantum gates as shown in Fig. 8. For the full adder circuit implementation, double Peres gates are used as shown in Fig. 9. At first, the double Peres gate can be decomposed into quantum gates as shown in the upper part of Fig. 9. The circuit can be further optimized by eliminating redundant gates as shown in the lower part of Fig. 9. In Fig. 9, the combination of one V gate and one V+ gate, indicated by a box, is an identity gate and thus the combination is a redundant gate. This can be proved as given below.

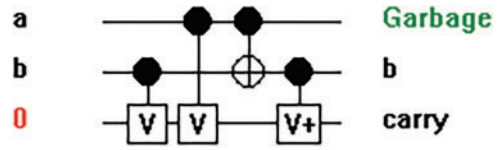

Figure 8: Decomposition of Peres gate into quantum gates

Lemma-3: One V gate and one V+ gate can be combined to generate an identity gate

Proof: We know that $V = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}$ and $V^+ = \frac{1-i}{2}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}$. Identity gate, $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

Thus, $V \times V^+ = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} \times \frac{1-i}{2}\begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix} = \frac{1}{2} \times \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$. (Thus proved).

Similarly, it can be proved that, $CV \times CV^+ = I$. Since an identity gate has zero quantum cost so the two gates indicated in the box in Fig. 9 can be eliminated.

In the circuit shown in Fig. 6, in general $2n$ CNOT gates have been used for control operation. Total n number of double Peres gates has been used for addition or subtraction purpose. Total n number of single Peres gates has been used for 2's complement operation. Therefore, the quantum cost is $2 + 2n + 6n + 4n = 12n + 2$. For 4 bit operation, the quantum cost is $12 \times 4 + 2 = 50$.

Second Method: For the second method, the adder proposed in [12] has been used. The single bit adder circuit can be decomposed into several quantum gates shown in Fig. 10. The two circuits indicated in square boxes in Fig. 10 are two quantum gates because they are unitary matrices. It can be proved in the following Lemma.

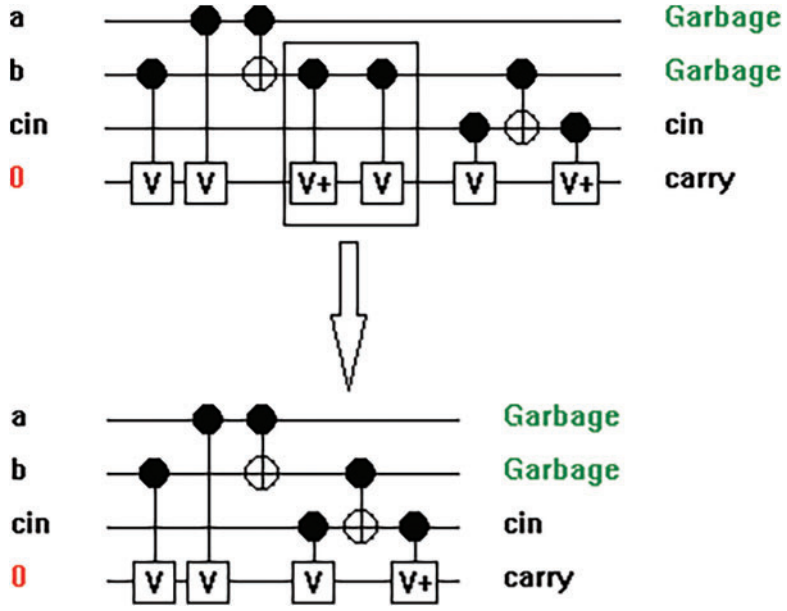

Figure 9: Decomposition of double Peres gate into quantum gates

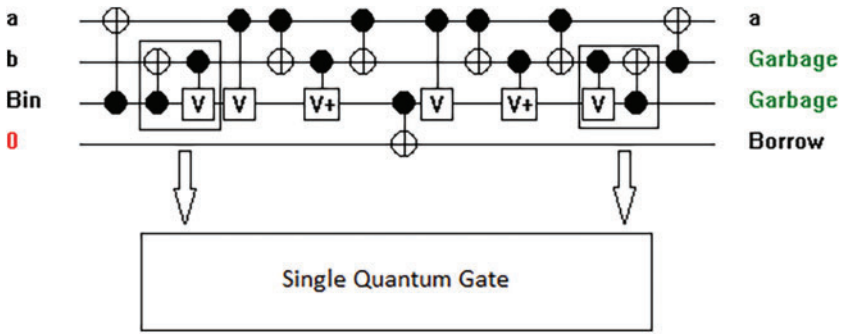

Figure 10: Decomposition of the new adder circuit into quantum gates

Lemma-4: The combination of one NOT and one V gate forms a unitary gate

Proof: As we know that $V = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}$ and $NOT = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. Therefore, $V \times NOT = \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix}$ which can be proved to be a unitary gate. Similarly, $NOT \times V = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \frac{1+i}{2}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} \frac{1+i}{2} & \frac{1-i}{2} \\ \frac{1-i}{2} & \frac{1+i}{2} \end{bmatrix} = \begin{bmatrix} \frac{1-i}{2} & \frac{1+i}{2} \\ \frac{1+i}{2} & \frac{1-i}{2} \end{bmatrix} = V \times NOT$. It can be proved that the output matrix is a unitary matrix, thus, it is a quantum gate. (Thus proved).

One single bit adder circuit has the quantum cost 13 as counted from Fig. 9. Thus for n-bit operation, the total quantum cost is $2 + 2n + 13n + 4n = 19n + 2$. For 4 bit operation, the quantum cost is $19 \times 4 + 2 = 78$.

As shown in Table 2, for the second method, the ancillary input count is reduced by almost $\frac{\mathrm{n}-1}{2n} \times 100 \cong 50\%$, the garbage output count is reduced by almost $\frac{n-1}{3n+1} \times 100 \cong 33\%$ and the quantum cost is increased by almost $\frac{7n}{19n+2} \times 100 \cong 36\%$. In [13–15], subtractor design schemes have been shown but the designs are for unsigned numbers whereas in this paper, the designs for signed numbers have been focused. Therefore, the results in [13–15] are beyond the scope of comparison with the achieved results of this paper.

Table 2: Parametric comparison of two signed subtraction methods

Parameters	First method	Second method
Ancillary input count	2n	n + 1
Garbage output count	3n + 1	2n + 2
Quantum cost count	12n + 2	19n + 2

6 Conclusion

A novel signed subtraction technique was shown in the paper. The technique is unique in the sense it used four types of operands (positive–positive, positive–negative, negative–positive and negative–negative) and for the four types of operands, the algorithm worked efficiently. The algorithm was verified using HDL programming and then it was implemented in reversible logic. Here two different addition techniques were shown for the subtraction. Finally the reversible circuit was implemented using the quantum gates. It could be concluded that the second method was better in respect of fabrication cost whereas the first method was better in respect of the computational complexity.

7 Future Scope of the Research

In future, the work can be extended to the design optimization of other arithmetic circuits like, signed multiplier, squarer, exponential computation etc. using quantum gates.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Marshal, *Circuit Design Using a FinFET Process*. Dallas: Texas Instruments Incorporated, 2006. [Online]. Available at: https://ewh.ieee.org/soc/cas/dallas/documents/Sem-012506-Andrew_FinFET.pdf
[2] R. Landauer, "Irreversibility and heat generation in the computational process," *IBM Journal of Research and Development*, vol. 5, no. 3, pp. 183–191, 1961.
[3] C. H. Bennett, "Logical reversibility of computation," *IBM Journal of Research and Development*, vol. 17, no. 6, pp. 525–532, 1973.
[4] C. H. Bennett, "The thermodynamics of computation—A review," *IBM Journal of Research and Development*, vol. 21, pp. 905–940, 1981.
[5] C. H. Bennett, "Notes on the history of reversible computation," *IBM Journal of Research and Development*, vol. 32, no. 1, pp. 16–23, 1998.
[6] T. Toffoli, *Reversible Computing*. Cambridge, New York, USA: Technical Memo MIT/LCS/TM-151, MIT Lab for Computer Science, pp. 1–37, 1980.
[7] E. Fredkin and T. Toffoli, "Conservative logic," *International Journal of Theoretical Physics*, vol. 21, pp. 219–253, 1981.
[8] A. Peres, "Reversible logic and quantum computers," *Physical Review A*, vol. 32, pp. 3266–3276, 1985.
[9] D. Deutsch, "Quantum theory, the Church-Turing principle and the universal quantum computer," *Proceedings of the Royal Society of London*, vol. A400, pp. 97–117, 1985.

[10] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, New York, USA: Cambridge University Press, 2000.
[11] R. Wille and R. Drechsler, *Towards a Design Flow for Reversible Logic*. Germany: Springer, 2010. (https://doi.org/10.1007/978-90-481-9579-4)
[12] S. Cuccaro, T. Draper, D. Moulton and S. Kutin, "A new quantum ripple-carry addition circuit," in *Proc. of the 8th Workshop on Quantum Information Processing*, Cambridge, pp. 1–9, 2005.
[13] H. Thapliyal and N. Ranganathan, "Design of efficient reversible binary subtractors based on a new reversible gate," in *IEEE Proc. of the Computer Society Annual Symp. on VLSI*, Tampa, FL, USA, pp. 229–234, 2009.
[14] H. Thapliyal and N. Ranganathan, "A new design of the reversible subtractor circuit," in *Proc. of IEEE Int. Conf. on Nanotechnology*, Portland, pp. 1430–1435, 2011.
[15] H. Thapliyal, "Mapping of subtractor and adder-subtractor circuits on reversible quantum gates," in *Transactions on Computational Science XXVII*, vol. 9570. Berlin, Heidelberg: LNCS, Springer, pp. 10–34, 2016.
[16] R. Wille, O. Keszocze, L. Othmer, M. K. Thomsen and R. Drechsler, "Generating and checking control logic in the HDL-based design of reversible circuits," in *Int. Symp. on Embedded Computing and System Design*, Patna, India, pp. 7–12, 2016.
[17] M. Soeken, R. Wille, O. Keszöcze, D. M. Miller and R. Drechsler, "Embedding of large boolean functions for reversible logic," *ACM Journal on Emerging Technologies in Computing System*, vol. 12, no. 4, pp. 41(1)–41(26), 2016.
[18] A. Zulehner and R. Wille, "Improving synthesis of reversible circuits: Exploiting redundancies in paths and nodes of QMDDs," in *Reversible Computation*, Kolkata, India, pp. 232–247, 2017.
[19] C. Moraga and F. Z. Hadjam, "On double gates for reversible computing circuits," in *Proc. Int. Workshop on Boolean Problems*, Freiberg University of Mining and Technology, Germany, 2012.