Design of Efficient Reversible Logic-Based Binary and BCD Adder Circuits

HIMANSHU THAPLIYAL and NAGARAJAN RANGANATHAN,
University of South Florida, Tampa

Reversible logic is gaining significance in the context of emerging technologies such as quantum computing since reversible circuits do not lose information during computation and there is one-to-one mapping between the inputs and outputs. In this work, we present a class of new designs for reversible binary and BCD adder circuits. The proposed designs are primarily optimized for the number of ancilla inputs and the number of garbage outputs and are designed for possible best values for the quantum cost and delay. In reversible circuits, in addition to the primary inputs, some constant input bits are used to realize different logic functions which are referred to as ancilla inputs and are overheads that need to be reduced. Further, the garbage outputs which do not contribute to any useful computations but are needed to maintain reversibility are also overheads that need to be reduced in reversible designs. First, we propose two new designs for the reversible ripple carry adder: (i) one with no input carry c_0 and no ancilla input bits, and (ii) one with input carry c_0 and no ancilla input bits. The proposed reversible ripple carry adder designs with no ancilla input bits have less quantum cost and logic depth (delay) compared to their existing counterparts in the literature. In these designs, the quantum cost and delay are reduced by deriving designs based on the reversible Peres gate and the TR gate. Next, four new designs for the reversible BCD adder are presented based on the following two approaches: (i) the addition is performed in binary mode and correction is applied to convert to BCD when required through detection and correction, and (ii) the addition is performed in binary mode and the result is always converted using a binary to BCD converter. The proposed reversible binary and BCD adders can be applied in a wide variety of digital signal processing applications and constitute important design components of reversible computing.

Categories and Subject Descriptors: B.2.1 [**Arithmetic and Logic Structures**]: Design Styles

General Terms: Design

Additional Key Words and Phrases: Reversible arithmetic, Peres gate, TR gate, ripple carry adders

ACM Reference Format:
Thapliyal, H. and Ranganathan, N. 2013. Design of efficient reversible logic-based binary and BCD adder circuits. ACM J. Emerg. Technol. Comput. Syst. 9, 3, Article 17 (September 2013), 31 pages.
DOI: http://dx.doi.org/10.1145/2491682

1. INTRODUCTION

In the hardware design, binary computing is preferred over decimal computing because of ease in building hardware based on binary number system [Parhami 2010]. In spite of ease in building binary hardware, most of the fractional decimal numbers such as 0.110 cannot be exactly represented in binary, thus their approximate values are used for performing computations in binary hardware. Because the financial, commercial, and Internet-based applications cannot tolerate errors generated by conversion

Authors' addresses: H. Thapliyal and N. Ranganathan, Department of Computer Science and Engineering, University of South Florida at Tampa, 4202 E. Fowler Avenue, ENB-118, Tampa, FL 33620; email: {hthapliy, ranganat}@cse.usf.edu.

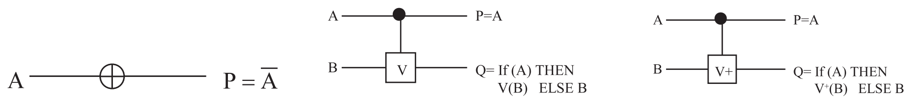
DOI: http://dx.doi.org/10.1145/2491682

between decimal and binary formats, the decimal arithmetic is receiving significant attention and efforts are being accelerated to build dedicated hardware based on decimal arithmetic [Wang et al. 2010]. As commercial databases tend to contain more decimal than binary data, the use of binary hardware requires the conversion of decimal to binary and vice versa which is an overhead [Cowlishaw 2003; Bayrakci and Akkas 2007]. Recently, software libraries that include conversion capabilities have become available so that the computations appear to be in decimal, making it transparent to the programmer. But the software implementation of decimal arithmetic is usually 100 to 1000 times slower than implementing in hardware [Cowlishaw 2010].

Among the emerging computing paradigms, reversible logic appears to be promising due to its wide applications in emerging technologies such as quantum computing, quantum dot cellular automata, optical computing, etc. [Nielsen and Chuang 2000; Ma et al. 2008, 2009; Taraphdara et al. 2010; Parhami 2006]. Reversible logic is also being investigated for its promising applications in power-efficient nanocomputing [Chang et al. 2010; Frank 2005]. Reversible circuits are those circuits that do not lose information during computation and reversible computation in a system can be performed only when the system is comprised of reversible gates. These circuits can generate a unique output vector from each input vector, and vice versa, that is, there is a one-to-one mapping between the input and the output vectors.

A quantum computer will be viewed as a quantum network (or a family of quantum networks) composed of quantum logic gates, each gate performing an elementary unitary operation on one, two, or more two-state quantum systems called qubits. Each qubit represents an elementary unit of information, corresponding to the classical bit values 0 and 1. Any unitary operation is reversible and hence quantum networks must be built from reversible logical components [Nielsen and Chuang 2000; Vedral et al. 1996]. Quantum computers of many qubits are extremely difficult to realize, thus the number of qubits in the quantum circuits needs to be minimized [Takahashi 2010; Takahashi and Kunihiro 2005]. This sets the major objective of optimizing the number of ancilla input qubits and the number of the garbage outputs in the reversible logic-based quantum circuits. The constant input in the reversible quantum circuit is called the ancilla input qubit (ancilla input bit), while the garbage output refers to the output that exists in the circuit just to maintain one-to-one mapping but is neither one of the primary inputs nor a useful output. Thus, the inputs regenerated at the outputs are not considered as garbage outputs [Fredkin and Toffoli 1982].

The proposed work focuses on the design of reversible binary and the BCD adder circuits primarily optimized for number of ancilla input bits and the garbage outputs. As the optimization of ancilla input bits and the garbage outputs may impact the design in terms of the quantum cost and the delay, thus quantum cost and the delay parameters are also considered for optimization with primary focus towards the optimization of number of ancilla input bits and the garbage outputs. To the best of our knowledge this is the first attempt in the literature that explores reversible BCD adder designs with the goal of optimizing the number of ancilla input bits and the garbage outputs. First, we propose two new designs for the reversible ripple carry adder: (i) one with no input carry c_0 and no ancilla input bits, and (ii) one with input carry c_0 and no ancilla input bits. The proposed reversible ripple carry adder designs with no ancilla input bits have less quantum cost and logic depth (delay) compared to their existing counterparts in the literature. In these designs, the quantum cost and delay are reduced by deriving designs based on the reversible Peres gate and the TR gate. Next, four new designs for the reversible BCD adder are presented based on the following two approaches: (i) the addition is performed in binary mode and correction is applied to convert to BCD when required through detection and correction, and (ii) the addition is performed in binary mode and the result is always converted using a binary to BCD converter. The various



(a) NOT gate (b) Controlled-V gate (c) Controlled-V^+ gate

Fig. 1. The Controlled-V and Controlled-V^+ gates.

reversible components needed in the BCD adder design are optimized in parameters of number of ancilla input bits/qubits and the number of garbage outputs and explore the possible best values for the quantum cost and delay. The comparison of the proposed designs with existing designs is also illustrated.

The article is organized as follows: The background of the reversible logic and the details of existing works are presented in Section 2, and the improved design of the TR gate is illustrated in Section 3. The design methodologies of the proposed reversible ripple carry adder with no input carry and with input carry are discussed in Sections 4 and 5, respectively. The designs of the reversible BCD adder are addressed in Section 6. The simulation and the verification of the proposed designs using Verilog HDL are presented in Section 7, while the conclusions are provided in Section 8.

2. BACKGROUND

The most popular reversible gates are the Fredkin gate [Fredkin and Toffoli 1982], the Toffoli gate [Toffoli 1980], and the Peres gate [Peres 1985]. Each reversible gate has an associated implementation cost called the quantum cost [Smolin and DiVincenzo 1996]. The quantum cost of a reversible gate is the number of 1×1 and 2×2 reversible gates or quantum logic gates required in its design. The quantum costs of all reversible 1×1 and 2×2 gates are taken as unity [Smolin and DiVincenzo 1996; Hung et al. 2006; Maslov and Miller 2006]. The 3×3 reversible gates are realized using 1×1 NOT gate, and 2×2 reversible gates such as Controlled-V and Controlled-V^+ (V is a square-root-of NOT gate and V^+ is its hermitian) and the Feynman gate which is also known as the Controlled NOT gate (CNOT). The quantum cost of a reversible gate can be calculated by counting the numbers of NOT, Controlled-V, Controlled-V^+, and CNOT gates required in its implementation.

2.1. The NOT Gate

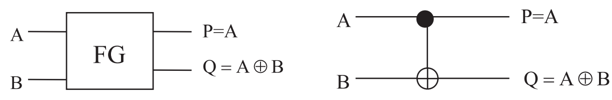
A NOT gate is a 1×1 gate represented as shown in Figure 1(a). Since it is a 1×1 gate, its quantum cost is unity.

2.2. The Controlled-V and Controlled-V^+ Gates

The Controlled-V gate is shown in Figure 1(b). In the Controlled-V gate, when the control signal A = 0 then the qubit B will pass through the controlled part unchanged, that is, we will have Q = B. When A = 1 then the unitary operation $V = \frac{i+1}{2}\begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}$ is applied to the input B, that is, Q = V(B). The Controlled-V^+ gate is shown in Figure 1(c). In the Controlled-V^+ gate when the control signal A = 0 then the qubit B will pass through the controlled part unchanged, that is, we will have Q = B. When A = 1 then the unitary operation $V^+ = V^{-1}$ is applied to the input B, that is, Q = V^+(B).

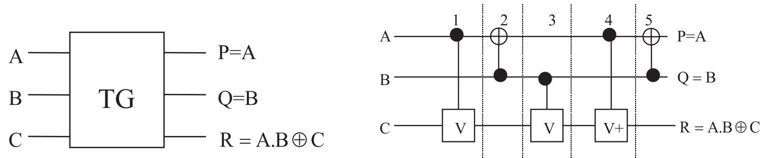
The V and V^+ quantum gates have the following properties.

$$
\begin{aligned}
V \times V &= NOT \\
V \times V^+ = V^+ \times V &= I \\
V^+ \times V^+ &= NOT
\end{aligned}
$$



(a) CNOT gate (b) quantum representation

Fig. 2. The CNOT gate and its quantum representation.



(a) Toffoli gate (b) quantum implementation of Toffoli gate

Fig. 3. The Toffoli gate and its quantum implementation.

The preceding properties show that when two V gates are in series they will behave as a NOT gate. Similarly, two V^+ gates in series also function as a NOT gate. A V gate in series with V^+ gate, and vice versa, is an identity. For more details of the V and V^+ gates, the reader is referred to Nielsen and Chuang [2000], Hung et al. [2006], and Maslov and Miller [2006].

2.3. The Feynman Gate (CNOT Gate)

The Feynman Gate (FG) or the Controlled NOT gate (CNOT) is a 2-inputs 2-outputs reversible gate having the mapping (A, B) to (P = A, Q = A $\oplus$ B) where A, B are the inputs and P, Q are the outputs, respectively. Since it is a 2$\times$2 gate, it has a quantum cost of 1. Figures 2(a) and 2(b) show the block diagrams and quantum representation of the Feynman gate.

2.4. The Toffoli Gate

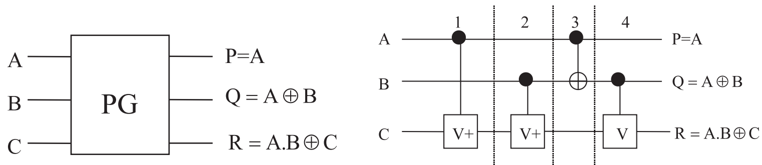
The Toffoli Gate (TG) is a 3$\times$3 two-through reversible gate as shown in Figure 3(a). Two-through means two of its outputs are the same as the inputs with the mapping (A, B, C) to (P = A, Q = B, R = A $\cdot$ B $\oplus$ C), where A, B, C are inputs and P, Q, R are outputs, respectively. The Toffoli gate is one of the most popular reversible gates and has the quantum cost of 5 as shown in Figure 3(b) [Toffoli 1980]. The quantum cost of a Toffoli gate is 5 as it needs 2V gates, 1 V^+ gate, and 2 CNOT gates to implement it.

2.5. The Peres Gate

The Peres gate is a 3-inputs 3-outputs (3x3) reversible gate having the mapping (A, B, C) to (P = A, Q = A $\oplus$ B, R = A $\cdot$ B $\oplus$ C), where A, B, C are the inputs and P, Q, R are the outputs, respectively [Peres 1985]. Figure 4(a) shows the Peres gate and Figure 4(b) shows the quantum implementation of the Peres Gate (PG) with quantum cost of 4 [Hung et al. 2006]. The quantum cost of a Peres gate is 4 since it requires 2 V^+ gates, 1 V gate, and 1 CNOT gate in its design. In the existing literature, among the 3x3 reversible gates, the Peres gate has the minimum quantum cost.

2.6. Delays of the Reversible Gates

Delay is another important parameter that can indicate the efficiency of the reversible circuits. Here, delay represents the critical delay of the circuit. In our delay calculations, we use the logical depth as the measure of the delay [Mohammadi and Eshghi 2009]. The delays of all 1$\times$1 gates and 2$\times$2 reversible gates are taken as unit delay called Δ.



(a) Peres gate

(b) quantum implementation of Peres gate

Fig. 4. The Peres gate and its quantum implementation.

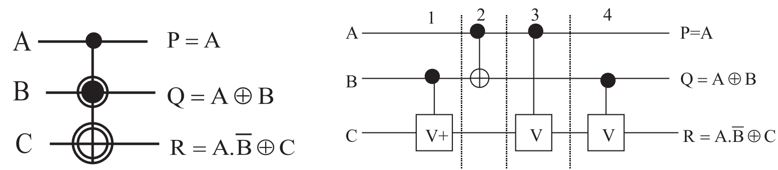
Any 3×3 reversible gate can be designed from 1×1 reversible gates and 2×2 reversible gates, such as the CNOT gate, the Controlled-V, and the Controlled-V^+ gates. Thus the delay of a 3×3 reversible gate can be computed by calculating its logical depth when it is designed from smaller 1×1 and 2×2 reversible gates. Figure 3(b) shows the logic depth in the quantum implementation of a Toffoli gate. Thus, it can be seen that the Toffoli gate has the delay of 5Δ. Each 2×2 reversible gate in the logic depth contributes to 1Δ delay. Similarly, the Peres gate shown in Figure 4(b) has the logic depth of 4 that results in its delay as 4Δ.

2.7. Prior Works

Research on reversible logic is expanding towards both design and synthesis. In the synthesis of reversible logic circuits there has been several interesting attempts in the literature such as in Gupta et al. [2006], Shende et al. [2003], Maslov and Dueck [2004], Yang et al. [2008], and Prasad et al. [2006]. The researchers have addressed the optimization of reversible logic circuits from the perspective of quantum cost and the number of garbage outputs. Recently, in Große et al. [2008, 2009] interesting contributions are made toward deriving exact minimal elementary quantum gate realization of reversible combinational circuits. Thus, in synthesis of reversible logic circuits the optimization in terms of number of ancilla input bits and also the delay are not yet addressed except the recent work in Wille et al. [2010] that discusses the post-synthesis method for reducing the number of lines (qubits) in reversible circuits. The designs of reversible sequential circuits are also addressed in literature in which various latches, flip-flops, etc., are designed [Rice 2008; Chuang and Wang 2008; Sastry et al. 2006; Thapliyal and Ranganathan 2010a, 2010b].

Reversible arithmetic units such as adders, subtractors, and multipliers that form the essential component of a computing system have also been designed in binary as well as ternary logic such as in Desoete and Vos [2002], Haghparast et al. [2008], Biswas et al. [2008], Bruce et al. [2002], Khan [2002], and Khan and Perkowski [2007]. In Cuccaro et al. [2004], researchers have designed the quantum ripple carry adder having no input carry with one ancilla input bit. In Takahashi and Kunihiro [2005] and Takahashi et al. [2009], the researchers have investigated new designs of the quantum ripple carry adder with no ancilla input bit and improved delay. In Trisetyarso and Van Meter [2009], the measurement-based design of a carry look-ahead adder is presented while in Van Meter et al. [2009] the concept of arithmetic on a distributed-memory quantum multicomputer is introduced. A comprehensive survey of quantum arithmetic circuits can be found in Takahashi [2010].

The design of BCD adders and subtractors has also been attempted. The researchers have investigated the design of BCD adders and subtractors in which parameters such as the number of reversible gates, number of garbage outputs, quantum cost, number of transistors, etc., are considered for optimization [Babu and Chowdhury 2006; Biswas et al. 2008; Mohammadi et al. 2008, 2009; Thomsen and Glück 2008; James et al. 2008].



(a) quantum symbol of the TR gate (b) quantum realization of the TR gate

Fig. 5. TR gate and its improved quantum implementation.

Thus to the best of our knowledge researchers have not yet addressed the design of the BCD arithmetic units primarily focusing on optimizing the number of ancilla input bits and the garbage outputs. In this work, we present a class of new designs for reversible binary and *BCD adder circuits*. The proposed designs are primarily optimized for the number of ancilla inputs and the number of garbage outputs and are designed for possible best values for the quantum cost and delay.

3. PROPOSED DESIGN OF THE TR GATE

The reversible TR gate is a 3-inputs 3-outputs gate having inputs to outputs mapping as (P = A, Q = A ⊕ B, $R = A \cdot \bar{B} \oplus C$ [Thapliyal and Ranganathan 2009]. We present the graphical notation of the TR gate in Figure 5(a) along with its new quantum implementation with 2×2 quantum gates in Figure 5(b). The TR gate is designed from 1 Controlled-V gate, 1 CNOT gate, and 2 Controlled-V^+ gates resulting in its quantum cost as 4. Further, the logic depth of the quantum implementation of the TR gate is 4 resulting in its propagation delay as 4Δ. The quantum cost and the delay of the TR gate was earlier estimated as 6 and 6Δ, respectively [Thapliyal and Ranganathan 2009]. The TR gate can realize the Boolean functions $A \cdot \bar{B} \oplus C$ and A ⊕ B with only one gate. Further, it can implement the functions such as $A \cdot \bar{B}$ when its input C is tied to 0. These properties of the TR gate make it very useful in designing reversible arithmetic units [Thapliyal and Ranganathan 2011; Thapliyal et al. 2010].

4. DESIGN METHODOLOGY OF PROPOSED REVERSIBLE RIPPLE CARRY ADDER WITH NO INPUT CARRY

We present the design of reversible ripple carry adder with no input carry(c_0) and designed without any ancilla inputs and the garbage outputs. The proposed method improves the quantum cost and the delay of the reversible ripple carry adder compared to the existing design approaches which have optimized the adder design in terms of number of ancilla inputs. Consider the addition of two n-bit numbers a_i and b_i stored at memory locations A_i and B_i, respectively, where $0 \le i \le n-1$. Further, consider that memory location A_n is initialized with z ∈ {0, 1}. At the end of the computation, the memory location B_i will have s_i, while the location A_i keeps the value a_i. The additional location A_n that initially stores the value z will have the value $z \oplus s_n$ at the end of the computation. Thus A_n will have the value of s_n when z = 0. Here, s_i is the sum bit produced and is defined as

$$s_i = \begin{cases} a_i \oplus b_i \oplus c_i & \text{if } 0 \le i \le n-1 \\ c_n & \text{if } i = n \end{cases}$$

where c_i is the carry bit and is defined as

$$c_i = \begin{cases} 0 & \text{if } i = 0 \\ a_{i-1}b_{i-1} \oplus b_{i-1}c_{i-1} \oplus c_{i-1}a_{i-1} & \text{if } 1 \le i \le n. \end{cases}$$

The proposed design methodology of generating the reversible ripple carry adder with no input carry minimizes the garbage outputs by producing the carry bits c_i based on the inputs a_{i-1}, b_{i-1} and the carry bit c_{i-1} from the previous stage. Once all the carry bits c_i are generated they are stored at memory location A_{i-1} which was initially used for storing the input a_{i-1} for $0 \leq i \leq n-1$. After the generated carry bits are used for further computation, the location A_i is restored to the value a_i while the location B_i stores the sum bit s_i for $0 \leq i \leq n-1$. Thus restoring of location A_i to the value a_i helps in minimizing the garbage outputs. Since no constant input having the value as 0 is needed in the proposed approach, it saves the ancilla inputs. The proposed methodology of generating the reversible ripple adder circuit without input carry is referred as methodology 1 in this work. The proposed methodology is generic in nature and can design the reversible ripple carry adder circuit with no input carry of any size. The steps involved in the proposed methodology are explained for addition of two n-bit numbers a_i and b_i, where $0 \leq i \leq n-1$. An illustrative example of generation of a reversible ripple carry adder circuit that can perform the addition of two 8-bit numbers $a = a_0 \ldots a_7$ and $b{=}b_0 \ldots b_7$ is also shown.

4.1. Steps of Methodology 1 (Reversible Adder Circuit With No Input Carry)

(1) For i = 1 to n-1. At pair of locations A_i and B_i apply the CNOT gate such that the location A_i will maintain the same value, while location B_i transforms to $({}^*A_i \oplus {}^*B_i)$, where ${}^*A_i$ and ${}^*B_i$ represent the values stored at location A_i and B_i. Step 1 is shown for a reversible ripple carry adder circuit that can perform the addition of two 8-bit numbers in Figure 6(a).
(2) For i = n-1 to 1. At pair of locations A_i and A_{i+1} apply the CNOT gate such that the location A_i will maintain the same value, while the location A_{i+1} transforms to $({}^*A_i \oplus {}^*A_{i+1})$. Step 2 is shown for a reversible 8-bit adder circuit in Figure 6(b).
(3) For i = 0 to n-2. At locations B_i, A_i, and A_{i+1} apply the Toffoli gate such that B_i, A_i, and and A_{i+1} are passed to the inputs A, B, C, respectively, of the Toffoli gate. Step 3 is shown for a reversible 8-bit adder circuit in Figure 6(c).
(4) For i = n-1 to 0. At locations A_i, B_i, and and A_{i+1} apply the Peres gate such that A_i, B_i, and and A_{i+1} are passed to the inputs A, B, C, respectively, of the Peres gate. Step 4 is shown for a reversible 8-bit adder circuit in Figure 7(a).
(5) For i = 1 to n-2. At pair of locations A_i and A_{i+1} apply the CNOT gate such that the location A_i will maintain the same value, while location B_i transforms to the value $({}^*A_i \oplus {}^*B_i)$. Step 5 is shown for a reversible 8-bit adder circuit in Figure 7(b).
(6) For i = 1 to n-1. At pair of locations B_i and A_i apply the CNOT gate such that the location A_i will maintain the same value, while location B_i transforms to the value $({}^*A_i \oplus {}^*b_i)$. This final step will result in a reversible adder circuit that can perform the addition of two n-bit numbers. For a reversible 8-bit adder circuit, the design is shown in Figure 7(c).

Thus, the proposed methodology implements the reversible ripple carry adder with no input carry, without any ancilla input bit. Since in the design of the reversible BCD adder, the 4-bit reversible ripple carry adder will be used, thus its design is also illustrated in Figure 8.

THEOREM 1. *Let a and b be two n-bit binary numbers represented as a_i and b_i and $z \in \{0, 1\}$ is another 1-bit input, where $0 \leq i \leq n-1$, then the proposed design steps of methodology 1 result in the ripple carry adder circuit that works correctly. The proposed design methodology designs an n-bit adder circuit that produces the sum output s_i at the memory location where b_i is stored, while it restores the location where a_i is initially stored to the value a_i for $0 \leq i \leq n-1$. Further, the proposed design*

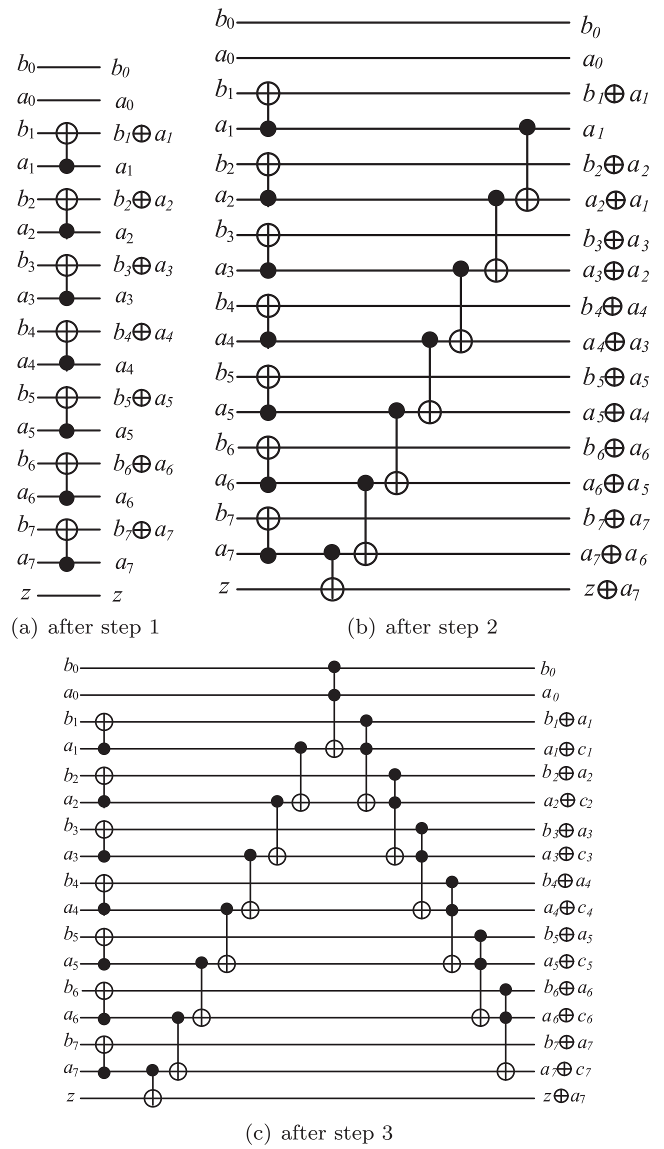


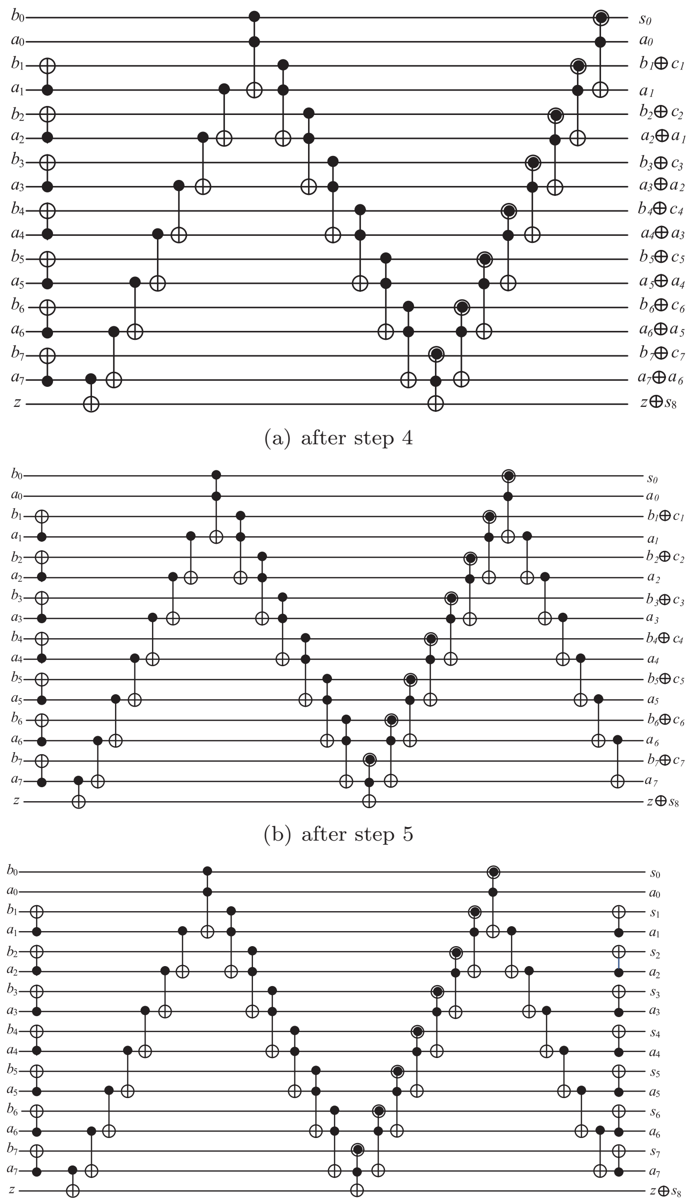
Fig. 6. Circuit generation of a reversible 8-bit adder with no input carry: steps 1–3.

methodology transforms the memory location where z is initially stored to $z \oplus s_n$, and restores the memory location where the input carry c_0 is initially stored to the value c_0.

PROOF. The proposed approach will make the following changes on the inputs that are illustrated as follows.

(1) Step 1. Step 1 of the proposed approach transforms the input states to

$$|b_0\rangle\,|a_0\rangle \left(\bigotimes_{i=1}^{n-1} |b_i \oplus a_i\rangle\,|a_i\rangle\right) |z\rangle.$$



(a) after step 4

(b) after step 5

(c) after step 6

Fig. 7. Circuit generation of a reversible 8-bit adder with no input carry: steps 4–6.

An example of the transformation of the input states after step 1 is illustrated for an 8-bit reversible ripple carry adder circuit in Figure 6(a).

(2) Step 2. Step 2 of the proposed approach transforms the input states to

$$|b_0\rangle\,|a_0\rangle\,|b_1 \oplus a_1\rangle\,|a_1\rangle \left(\bigotimes_{i=2}^{n-1} |b_i \oplus a_i\rangle\,|a_i \oplus a_{i-1}\rangle \right) |z \oplus a_{n-1}\rangle.$$

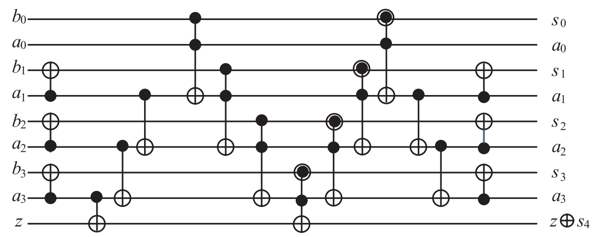


Fig. 8. Proposed reversible 4-bit adder without input carry.

An example of the transformation of the input states after step 2 is illustrated for an 8-bit reversible ripple carry adder circuit in Figure 6(b).

(3) Step 3. Step 3 has n-1 Toffoli gates. The first Toffoli gate takes the inputs as b_0,a_0, and a_1 and produce the output as b_0, a_0, and $a_1 \oplus c_1$. The third output of the Toffoli gate produces $a_1 \oplus c_1$ because $c_1 = a_0 \cdot b_0$ where c_1 represents the generated output carry after addition of a_0 and b_0. The remaining n-2 Toffoli gates take the inputs as $b_i \oplus a_i$, $a_i \oplus c_i$, $a_i \oplus a_{i+1}$ and produce the outputs as $b_i \oplus a_i$, $a_i \oplus c_i$, $a_{i+1} \oplus c_{i+1}$ where $1 \leq i \leq n-1$. Thus, after step 3, the input states are transformed to

$$|b_0\rangle\, |a_0\rangle \left(\bigotimes_{i=1}^{n-1} |b_i \oplus a_i\rangle\, |a_i \oplus c_i\rangle \right) |z \oplus a_{n-1}\rangle.$$

An example of the transformation of the input states after step 3 is illustrated for an 8-bit reversible ripple carry adder circuit in Figure 6(c).

(4) Step 4. Step 4 has n Peres gates. The n-1 Peres gate takes the inputs as $a_i \oplus c_i$, $b_i \oplus a_i$, $a_{i+1} \oplus c_{i+1}$ to produce the outputs as $a_i \oplus c_i$, $b_i \oplus c_i$, $a_i \oplus a_{i+1}$. The third output of the Peres gate is $a_i \oplus a_{i+1}$ because it realizes the function $A \cdot B \oplus C$ where A, B, and C are the inputs of the Peres gate. Hence the Peres gates will have the third outputs as $a_i \oplus c_i \cdot b_i \oplus c_i \oplus a_i \oplus a_{i+1} = a_i \oplus a_{i+1}$. The nth Peres gate takes the inputs as a_0, b_0, $a_1 \oplus c_1$ to produce the outputs as a_0, $a_0 \oplus b_0$, a_1. Please note that $s_0 = a_0 \oplus b_0$. Thus step 4 transforms the input states to

$$|s_0\rangle\, |a_0\rangle\, |b_1 \oplus c_1\rangle\, |a_1\rangle \left(\bigotimes_{i=2}^{n-1} |b_i \oplus c_i\rangle\, |a_i \oplus a_{i-1}\rangle \right) |z \oplus s_n\rangle\, .$$

An example of the transformation of the input states after step 4 is illustrated for an 8-bit reversible ripple carry adder circuit in Figure 7(a).

(5) Step 5. Step 5 of the proposed approach transforms the input states to

$$|s_0\rangle\, |a_0\rangle \left(\bigotimes_{i=1}^{n-1} |b_i \oplus c_i\rangle\, |a_i\rangle \right) |z \oplus s_n\rangle\, .$$

An example of the transformation of the input states after step 5 is illustrated for an 8-bit reversible ripple carry adder circuit in Figure 7(b).

(6) Step 6. Step 6 of the proposed approach transforms the input states to

$$\left(\bigotimes_{i=0}^{n-1} |s_i\rangle\, |a_i\rangle \right) |z \oplus s_n\rangle\, .$$

An example of the transformation of the input states after step 6 is illustrated for an 8-bit reversible ripple carry adder circuit in Figure 7(c). □

Table I. A Comparison of Reversible Ripple Carry Adder with No Input Carry

	1	2	3	Proposed
Ancilla Inputs	1	0	0	0
Garbage Outputs	1	0	0	0
Quantum Cost	17n-12	26n-29	15n-9	13n-8
Delay Δ	10n	24n-27	13n-7	11n-4

1 is the design in [Cuccaro et al. 2004]
2 is the design in [Takahashi and Kunihiro 2005]
3 is the design in [Takahashi et al. 2009]

Thus, the proposed six steps transform the memory location where b_i is initially stored to the sum output s_i, while the location where a_i is initially stored will be restored to the value a_i for $0 \leq i \leq n-1$ after the generation of the output carries and their subsequent use to produce the sum outputs. The memory location where z is stored will have $z \oplus s_n$ and the memory location where the input carry c_0 was stored initially will be restored to the value c_0. In summary, the proposed design methodology 1 generates the n-bit reversible ripple carry adder that is functionally correct.

Delay and Quantum Cost

—Step 1 of the proposed methodology needs $n-1$ CNOT gates working in parallel thus this step has the quantum cost of $n-1$ and delay of 1Δ.
—Step 2 of the proposed methodology needs n CNOT gates working in series thus this step has the quantum cost of n and delay of $n\Delta$.
—Step 3 needs $n-1$ Toffoli gates working in series thus this step has the quantum cost of $5(n-1)$ and delay of $5(n-1)\Delta$.
—Step 4 needs n Peres gates working in series thus this step has the quantum cost of $4n$ and delay of $4n\Delta$.
—Step 5 needs $n-1$ CNOT gates working in series thus this step has the quantum cost of $n-1$ and delay of $(n-1)\Delta$.
—Step 6 needs $n-1$ CNOT gates working in parallel thus this step has the quantum cost of $n-1$ and delay of 1Δ.

Thus, the total quantum cost of an n-bit reversible ripple carry adder is $n-1+n+5(n-1)+4n+n-1+n-1 = 13n-8$. The propagation delay will be $1\Delta + n\Delta + 5(n-1)\Delta + 4n\Delta + (n-1)\Delta + 1\Delta = (11n-4)\Delta$. A comparison of the proposed design with the existing designs is illustrated in Table I. In Table I, for Takahashi and Kunihiro [2005] the quantum cost and the delay values are valid for $n \geq 2$, and for n=1 the design has the quantum cost of 8 and delay of 8Δ. Among the existing designs of the reversible ripple carry adder with no input carry, the designs in Takahashi and Kunihiro [2005] and Takahashi et al. [2009] are designed with no ancilla input bits and the garbage outputs, while the design presented in Cuccaro et al. [2004] has 1 ancilla input and 1 garbage output. In this work we have compared our proposed design of the reversible carry adder with the designs in Cuccaro et al. [2004], Takahashi and Kunihiro [2005], and Takahashi et al. [2009] for values of n varying from 8 bits to 512 bits. Table II shows the comparison in terms of quantum cost which shows that the proposed design of the reversible carry adder with no input carry achieves improvement ratios ranging from 22.5% to 23.51%, 46.36% to 49.95%, and 13.37% to 13.33% compared to the design presented in Cuccaro et al. [2004], Takahashi and Kunihiro [2005], and Takahashi et al. [2009], respectively. From Table III, it can be seen that the proposed design of reversible ripple carry adder achieves improvement ratios ranging from 49.09% to 54.09%, and 13.40% to 15.35% in terms of delay compared to the designs presented in Takahashi and

Table II. Quantum Cost Comparison of Reversible Ripple Carry Adders (no input carry)

Bits	1	2	3	Proposed	% Impr. w.r.t 1	% Impr. w.r.t 2	% Impr. w.r.t 3
8	124	179	111	96	22.5	46.36	13.51
16	260	387	231	200	23	48.32	13.41
32	532	803	471	408	23.3	49.19	13.37
64	1076	1635	951	824	23.42	49.6	13.35
128	2164	3299	1911	1656	23.47	49.8	13.34
256	4340	6627	3831	3320	23.5	49.9	13.33
512	8692	13283	7671	6648	23.51	49.95	13.33

1 is the design in [Cuccaro et al. 2004]
2 is the design in [Takahashi and Kunihiro 2005]
3 is the design in [Takahashi et al. 2009]

Table III. Delay(in $\triangle$) Comparison of Reversible Ripple Carry Adders (no input carry)

Bits	1	2	3	Proposed	% Impr. w.r.t 1	% Impr. w.r.t 2	% Impr. w.r.t 3
8	80	165	97	84	-	49.09	13.40
16	160	357	201	172	-	51.8	14.42
32	320	741	409	348	-	53.03	14.91
64	640	1509	825	700	-	53.61	15.15
128	1280	3045	1657	1404	-	53.89	15.26
256	2560	6117	3321	2812	-	54.02	15.32
512	5120	12261	6649	5628	-	54.09	15.35

1 is the design in [Cuccaro et al. 2004]
2 is the design in [Takahashi and Kunihiro 2005]
3 is the design in [Takahashi et al. 2009]

Kunihiro [2005] and Takahashi et al. [2009], respectively, while the design presented in Cuccaro et al. [2004] is faster than the proposed design by 4.7% to 9.02%.

5. DESIGN METHODOLOGY OF PROPOSED REVERSIBLE RIPPLE CARRY ADDER WITH INPUT CARRY

The reversible ripple carry adder with input carry(c_0) is designed without any ancilla inputs and the garbage outputs, and with less quantum cost and reduced delay compared to the existing design approaches which have optimized the adder design in terms of number of ancilla inputs. Consider the addition of two n-bit numbers a_i and b_i stored at memory locations A_i and B_i, respectively, where $0 \leq i \leq n-1$. The input carry c_0 is stored at memory location A_{-1}. Further, consider that memory location A_n is initialized with z $\in \{0, 1\}$. At the end of the computation, the memory location B_i will have s_i, while the location A_i keeps the value a_i for $0 \leq i \leq n-1$. Further, at the end of the computation, the additional location A_n that initially stores the value z will have the value $z \oplus s_n$, and the memory location A_{-1} keeps the input carry c_0. Thus A_n will have the value of s_n when z = 0. Here, s_i is the sum bit produced and is defined as

$$s_i = \begin{cases} a_i \oplus b_i \oplus c_i & \text{if } 0 \leq i \leq n-1, \\ c_n & \text{if } i = n \end{cases}$$

where c_i is the carry bit and is defined as

$$c_i = \begin{cases} c_0 & \text{if } i = 0 \\ a_{i-1}b_{i-1} \oplus b_{i-1}c_{i-1} \oplus c_{i-1}a_{i-1} & \text{if } 1 \leq i \leq n. \end{cases}$$

As shown before, c_i is the carry bit and is generated by using a_{i-1}, b_{i-1} and c_{i-1}. In our proposed approach, first all the carry bits are generated and are saved in the memory location A_{i-1} which was initially used for storing a_{i-1} for $1 \leq i \leq n-1$. Once the generated carry bits are used, the location A_i is restored to the value a_i while the location B_i will have s_i for $0 \leq i \leq n-1$. Thus restoring of location A_i to the value a_i helps in minimizing the garbage outputs. Since no constant input having the value as 0 is needed in the proposed approach, it saves the ancilla inputs. The details of the proposed approach to minimize the garbage outputs and the ancilla inputs can be understood by following the steps of the proposed design methodology.

The proposed method improves the delay and the quantum cost by selectively using the Peres gate and the TR gate at the appropriate places. The generalized methodology of designing the n-bit reversible ripple carry adder with input carry is explained shortly along with an illustrative example of an 8-bit reversible ripple carry adder. The illustrative example of 8-bit reversible ripple carry adder is shown in Figures 9 and 10 that can perform the addition of two 8-bit numbers $a = a_0 \ldots a_7$ and b=$b_0 \ldots b_7$, and has the input carry c_0. The proposed methodology will be referred as methodology 2 further in this work and is explained in the following steps.

5.1. Steps of Proposed Methodology 2 (Reversible Adder with Input Carry)

(1) For i = 0 to n-1. At pair of locations A_i and B_i apply the CNOT gate such that the location A_i will maintain the same value, while location B_i transforms to the value $^*A_i \oplus {}^*B_i$, where $^*A_i$ and $^*B_i$ represent the values stored at location A_i and B_i. For illustrative purposes, the circuit of the reversible ripple carry adder with input carry c_0 after step 1 is shown for addition of 8-bit numbers in Figure 9(a).

(2) For i = -1 to n-2. At pair of locations A_{i+1} and A_i apply the CNOT gate such that the location A_{i+1} will maintain the same value, while the value at location A_i transforms to $^*A_{i+1} \oplus {}^*A_i$. Further, apply a CNOT gate at pair of locations A_{n-1} and A_n such that the value at location A_{n-1} will remain same, while the value at location A_n transforms to $^*A_{n-1} \oplus {}^*A_n$. The reversible 8-bit adder circuit after the step 2 is illustrated in Figure 9(b).

(3) Step 3 has the following substeps.
   (a) For i = 0 to n-2. At locations A_{i-1}, B_i, and A_i apply the Toffoli gate such that A_{i-1}, B_i, and A_i are passed to the inputs A, B, C, respectively, of the Toffoli gate. Apply a Peres gate at location A_{n-2}, B_{n-1}, and A_n such that A_{n-2}, B_{n-1}, and A_n are passed to the inputs A, B, C, respectively, of the Peres gate.
   (b) For i = 0 to n-2. Apply a NOT gate at location B_i.
   The reversible 8-bit adder circuit based on the proposed design methodology after the step 3 is illustrated in Figure 9(c).

(4) Step 4 has the following two substeps.
   (a) For i = n-2 to 0. At locations A_{i-1}, B_i, and A_i apply the TR gate such that A_{i-1}, B_i, and A_i are passed to the inputs A, B, C, respectively, of the TR gate.
   (b) For i = 0 to n-2. Apply a NOT gate at location B_i.
   The reversible 8-bit adder circuit based on the proposed methodology after step 4 is illustrated in Figure 10(a).

(5) For i = n-1 to 0. At pair of locations A_i and A_{i-1} apply the CNOT gate such that the location A_i will maintain the same value, while value at location A_{i-1} transforms to the value $^*A_i \oplus {}^*A_{i-1}$. The reversible 8-bit adder circuit after the step 5 is shown in Figure 10(b).

(6) For i = 0 to n-1. At pair of locations A_i and B_i apply the CNOT gate such that the location A_i will maintain the same value, while the value at location B_i transforms to the value $^*A_i \oplus {}^*B_i$. After this step we will have the complete working design of

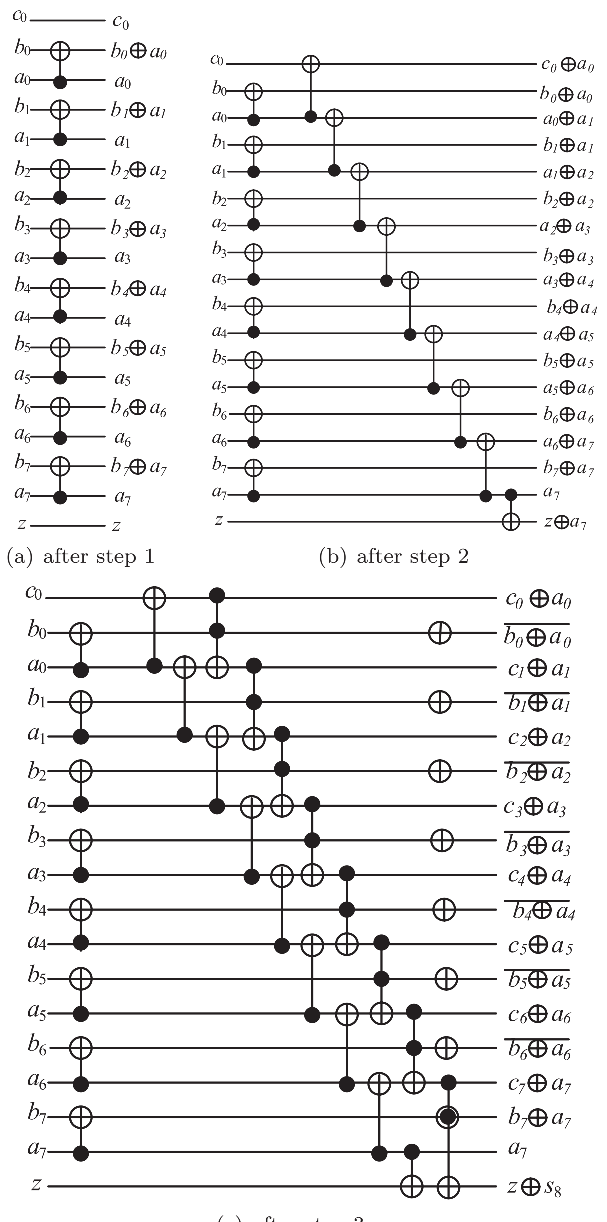
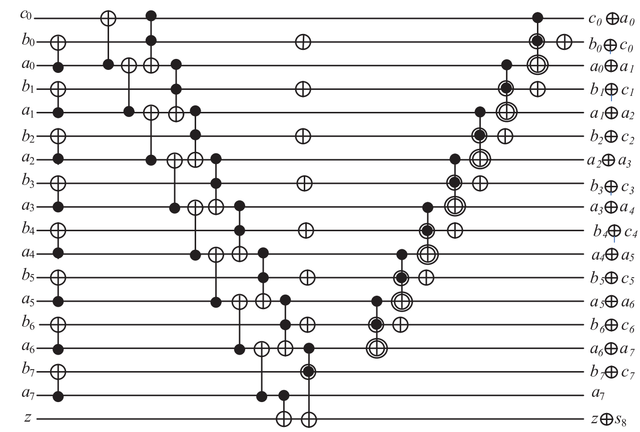


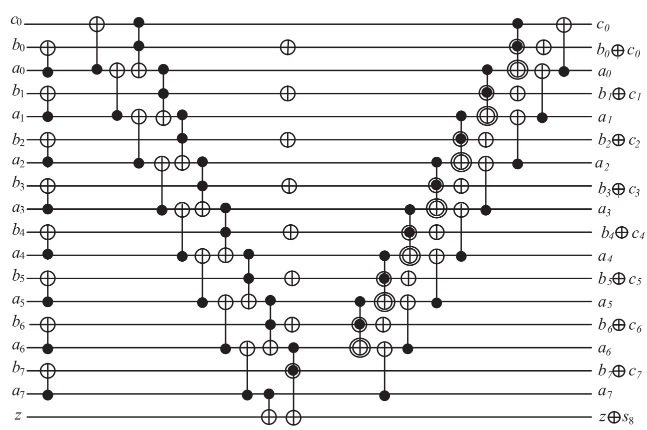
Fig. 9. Circuit generation of reversible 8-bit adder with input carry: steps 1–3.

the reversible adder, an example of which is shown for addition of 8-bit numbers in Figure 10(c).

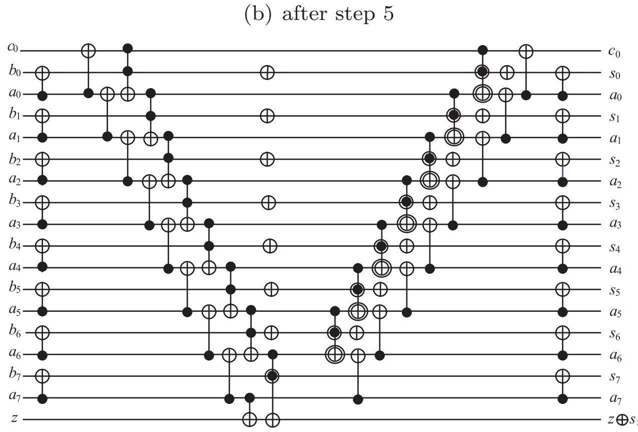
Thus, the proposed methodology is able to design the reversible ripple carry adder with an input carry without any ancilla and garbage bits. As in the design of the reversible BCD adder, the 4-bit reversible ripple carry adder will be used, thus its design is also illustrated in Figure 11.



(a) after step 4



(b) after step 5



(c) after step 6

Fig. 10. Circuit generation of reversible 8-bit adder with input carry: steps 4–6.

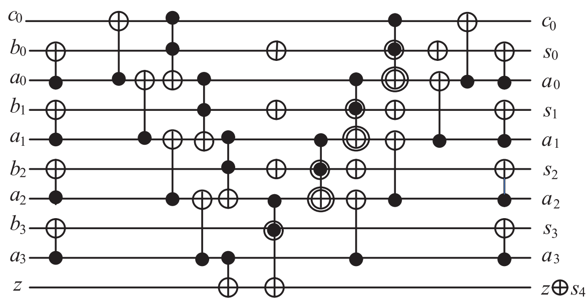


Fig. 11. Proposed reversible 4-bit adder with input carry.

THEOREM 2. *Let a and b be two n-bit binary numbers represented as a_i and b_i, c_0 is the input carry (c_0), and $z \in \{0, 1\}$ is the another 1-bit input, where $0 \le i \le n-1$, then the proposed design steps of methodology 2 result in the ripple carry adder circuit that works correctly. The proposed design methodology designs an n-bit adder circuit that produces the sum output s_i at the memory location where b_i is initially stored, while the location where a_i is initially stored is restored to the value a_i for $0 \le i \le n-1$. Further, the memory location where z is initially stored transforms to $z \oplus s_n$, and the memory location where the input carry c_0 is initially stored is restored to the value c_0.*

PROOF. The proposed approach will make the following changes on the inputs that are illustrated as follows.

(1) Step 1. Step 1 of the proposed approach transforms the input states to

$$|c_0\rangle \left(\bigotimes_{i=0}^{n-1} |b_i \oplus a_i\rangle \, |a_i\rangle \right) |z\rangle.$$

For illustrative purposes, the transformation of the input states of an 8-bit reversible adder circuit after step 1 is shown in Figure 9(a).

(2) Step 2. Step 2 of the proposed approach transforms the input states to

$$|c_0 \oplus a_0\rangle \left(\bigotimes_{i=0}^{n-2} |b_i \oplus a_i\rangle \, |a_i \oplus a_{i+1}\rangle \right) |b_{n-1} \oplus a_{n-1}\rangle \, |a_{n-1}\rangle \, |z \oplus a_{n-1}\rangle.$$

For illustrative purposes, the transformation of the input states of an 8-bit reversible adder circuit after step 2 is shown in Figure 9(b).

(3) Step 3. Step 3 has two substeps. Step 3(a) has the Toffoli gates which take the inputs as $c_i \oplus a_i$, $b_i \oplus a_i$, and $a_i \oplus a_{i+1}$ for $0 \le i \le n-2$. The Toffoli gates will produce the outputs as $c_i \oplus a_i$, $b_i \oplus a_i$, and $c_{i+1} \oplus a_{i+1}$. The third outputs of the Toffoli gates are $c_{i+1} \oplus a_{i+1}$ because of the fact that the Toffoli gate has the logic equation as $A \cdot B \oplus C$, where A, B, and C are the inputs of a Toffoli gate, thus will produce the output as $c_i \oplus a_i \cdot b_i \oplus a_i \oplus a_i \oplus a_{i+1}= c_{i+1} \oplus a_{i+1}$. Finally we have a Peres gate having the inputs $c_{n-1} \oplus a_{n-1}$, $b_{n-1} \oplus a_{n-1}$, and $a_{n-1} \oplus z$ which produces the outputs as $c_{n-1} \oplus a_{n-1}$, $c_{n-1} \oplus b_{n-1}$, and $s_n \oplus z$. Thus after the step 3(a) the input states are transformed to

$$|c_0 \oplus a_0\rangle \left(\bigotimes_{i=0}^{n-2} |b_i \oplus a_i\rangle \, |c_{i+1} \oplus a_{i+1}\rangle \right) |b_{n-1} \oplus c_{n-1}\rangle \, |a_{n-1}\rangle \, |z \oplus s_n\rangle.$$

Step 3(b) applies the NOT operation to the location B_i having the value as $b_i \oplus a_i$ for $0 \le i \le n-2$, thus the input states are transformed to

$$|c_0 \oplus a_0\rangle \left(\bigotimes_{i=0}^{n-2} \overline{|b_i \oplus a_i\rangle}\, |c_{i+1} \oplus a_{i+1}\rangle \right) |b_{n-1} \oplus c_{n-1}\rangle\, |a_{n-1}\rangle\, |z \oplus s_n\rangle.$$

For illustrative purposes, the transformation of the input states of an 8-bit reversible adder circuit after step 3 is shown in Figure 9(c).

(4) Step 4. Step 4 has the TR gates which take the inputs as $c_i \oplus a_i$, $\overline{b_i \oplus a_i}$, and $c_{i+1} \oplus a_{i+1}$ for i = n-2 to 0. The TR gates will produce the outputs as $c_i \oplus a_i$, $\overline{b_i \oplus c_i}$, and $a_i \oplus a_{i+1}$ for i=n-2 to 0. Thus after the application of TR gates the input states transform to

$$|c_0 \oplus a_0\rangle \left(\bigotimes_{i=0}^{n-2} |\overline{b_i \oplus c_i}\rangle\, |a_i \oplus a_{i+1}\rangle \right) |b_{n-1} \oplus c_{n-1}\rangle\, |a_{n-1}\rangle\, |z \oplus s_n\rangle.$$

Next, the NOT gates are applied to the TR gate outputs $\overline{b_i \oplus c_i}$ for i = n-2 to 1. Thus step 4 of the proposed approach transforms the input states to

$$|c_0 \oplus a_0\rangle \left(\bigotimes_{i=0}^{n-2} |b_i \oplus c_i\rangle\, |a_i \oplus a_{i+1}\rangle \right) |b_{n-1} \oplus c_{n-1}\rangle\, |a_{n-1}\rangle\, |z \oplus s_n\rangle.$$

For illustrative purposes, the transformation of the input states of an 8-bit reversible adder circuit after step 4 is shown in Figure 10(a).

(5) Step 5. Step 5 of the proposed approach transforms the input states to

$$|c_0\rangle \left(\bigotimes_{i=0}^{n-2} |b_i \oplus c_i\rangle\, |a_i\rangle \right) |b_{n-1} \oplus c_{n-1}\rangle\, |a_{n-1}\rangle\, |z \oplus s_n\rangle.$$

For illustrative purposes, the transformation of the input states of an 8-bit reversible adder circuit after step 5 is shown in Figure 10(c).

(6) Step 6. Step 6 of the proposed approach transforms the input states to

$$|c_0\rangle \left(\bigotimes_{i=0}^{n-1} |s_i\rangle\, |a_i\rangle \right) |z \oplus s_n\rangle.$$

For illustrative purposes, the transformation of the input states of an 8-bit reversible adder circuit after step 6 is shown in Figure 10(c). □

Thus we can see that the proposed six steps will produce the sum output s_i at the memory location where b_i is stored initially, while the location where a_i is stored initially will be restored to the value a_i for $0 \le i \le n-1$. The memory location where z is stored will have $z \oplus s_n$ and the memory location where the input carry c_0 was stored initially will be restored to the value c_0. This proves the correctness of the proposed methodology of designing the reversible ripple carry adder with input carry.

Delay and Quantum Cost

—Step 1 of the proposed methodology needs n CNOT gates working in parallel thus this step has the quantum cost of n and delay of 1Δ.

—Step 2 of the proposed methodology needs $n+1$ CNOT gates working in series thus this step has the quantum cost of $n+1$. The delay of this stage will be only 2Δ as

Table IV. A Comparison of Reversible Ripple Carry Adder with Input Carry

	1	2	Proposed
Ancilla Inputs	0	0	0
Garbage Outputs	0	0	0
Quantum Cost	17n-6	17n-22	15n-6
DelayΔ	10n+2	10n-8	9n+1
1 is the design 1 in [Cuccaro et al. 2004]			
2 is the design 2 in [Cuccaro et al. 2004]			

Table V. Quantum Cost Comparison of Reversible Ripple Carry Adders (with input carry)

Bits	1	2	Proposed	% Impr. w.r.t 1	% Impr. w.r.t 2
8	130	114	114	12.30	-
16	266	250	234	12.03	6.4
32	538	522	474	11.89	9.19
64	1082	1066	954	11.82	10.50
128	2179	2154	1914	11.79	11.14
256	4346	4330	3834	11.78	11.45
512	8698	8682	7674	11.77	11.61
1 is the design 1 in [Cuccaro et al. 2004]					
2 is the design 2 in [Cuccaro et al. 2004]					

it has $n-1$ CNOT gates working in parallel with the Toffoli gates of the next stage thus only 2 CNOT gates contribute to the delay.

—Step 3 needs $n-1$ Toffoli gates working in series thus contributing to the quantum cost of $5(n-1)$ and delay of $5(n-1)\Delta$. There is a Peres gate contributing to the quantum cost of 4 and delay of 4Δ. There are $n-1$ NOT gates working in parallel with the Peres gate thus contributing to quantum cost of $n-1$ and zero delay. The total quantum cost of this stage is $5(n-1)+4+n-1$ while the delay contribution of this stage is $5(n-1)\Delta+4\Delta$.

—Step 4 needs $n-1$ TR gates working in series thus contributing to the quantum cost by $4(n-1)$ and delay of $4(n-1)\Delta$. Further, there are $n-1$ NOT gates, which all work in parallel with the TR gates except the last NOT gate. Thus, it contributes to quantum cost of $n-1$ and delay of 1Δ. Thus this step has the quantum cost of $4(n-1)+n-1$ and the delay of $4(n-1)\Delta$ +1 Δ.

—Step 5 needs n CNOT gates working in parallel with the TR gates and the NOT gates, except the last one. Thus this step has the quantum cost of n and delay of 1Δ.

—Step 6 needs n CNOT gates working in parallel thus this step has the quantum cost of n and delay of 1Δ.

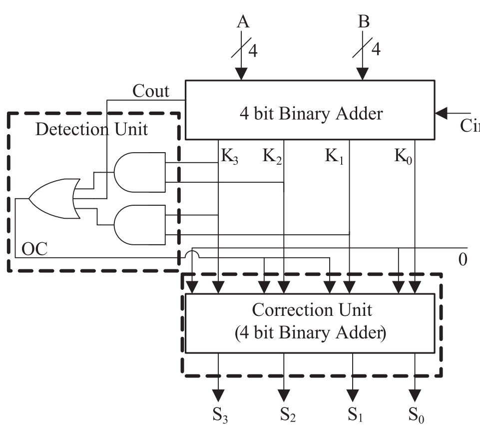
Thus the total quantum cost of an n-bit reversible ripple carry adder is $n+n+1+5(n-1)+4+n-1+4(n-1)+n-1+n+n=15n-6$. The propagation delay will be $1\Delta+2\Delta+5(n-1)\Delta+4\Delta+4(n-1)\Delta+1\Delta+1\Delta+1\Delta=(9n+1)\Delta$.

A comparison of the proposed design with existing designs is illustrated in Table IV which shows that the proposed design of a reversible ripple carry adder with input carry is designed with no ancilla input bit and has less quantum cost and delay compared to its existing counterparts. Table V shows the comparison in terms of quantum cost which shows that the proposed design of the reversible carry adder with input carry achieves the improvement ratios ranging from 12.3% to 11.77% and 0% to 11.61% compared to the designs presented in Cuccaro et al. [2004]. From Table VI, it can be

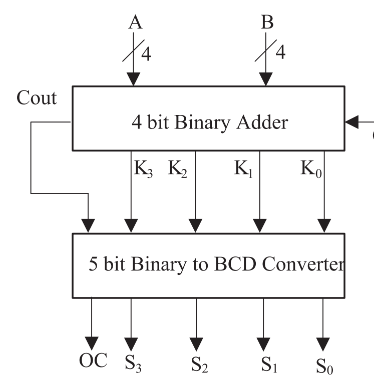
Table VI. Delay (inΔ) Comparison of Reversible Ripple Carry Adders (with input carry)

Bits	1	2	Proposed	% Impr. w.r.t 1	% Impr. w.r.t 2
8	82	72	73	10.97	-
16	162	152	145	10.49	4.6
32	322	312	289	10.24	7.37
64	642	632	577	10.12	8.7
128	1282	1272	1153	10.06	9.35
256	2562	2552	2305	10.03	9.67
512	5122	5112	4609	10.01	9.83

1 is the design 1 in [Cuccaro et al. 2004]
2 is the design 2 in [Cuccaro et al. 2004]



(a) approach 1 for design of conventional 1-digit BCD adder



(b) approach 2 for design of conventional 1-digit BCD adder

Fig. 12. Design approaches of conventional BCD adder.

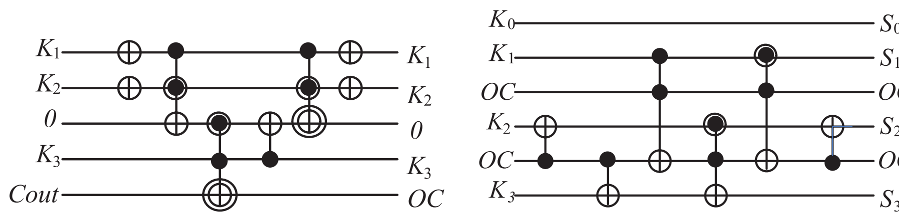
seen that the proposed design of a reversible ripple carry adder achieves improvement ratios ranging from 10.97% to 10.01%, and 0% to 9.83% in terms of delay compared to the designs presented in Cuccaro et al. [2004], respectively.

6. DESIGN OF REVERSIBLE BCD ADDER

A BCD adder is a circuit that adds two BCD digits in parallel and produces a sum digit, also in BCD. We are illustrating two different approaches of designing the conventional BCD adder.

6.1. Basics

Figure 12(a) shows the first approach of designing the 1-digit conventional BCD adder, which also includes the detection and the correction logic in its internal construction. The two decimal digits A and B, together with the input carry Cin, are first added in the top 4-bit binary adder to produce the 4-bit binary sum (K_3 to K_0) and the carry out (Cout). In the BCD addition, when the binary sum of A and B is less than 1001, the BCD number is same as the binary number thus no conversion is needed. But when the binary sum of A and B is greater than 1001, 0110 is added to convert the binary number into an equivalent BCD number. The condition that summation of numbers A, B, and C is greater than 1001 is detected through a detection unit. The



(a) proposed reversible detection unit with one ancilla input bit. $K1$, $K2$, $K3$ and Cout here are the outputs of the top 4-bit binary adder needed in the design of 1-digit BCD adder. OC represents the output carry. The details of $K1$, $K2$, $K3$, Cout and OC signals can be found in Figure 12(a)

(b) proposed reversible correction unit with no ancilla input bit. $K0$ here represents the least significant sum bit generated by the 4-bit binary adder in the BCD adder. $K1$, $K2$, $K3$ and OC are the outputs of the reversible detection unit. The details of these signals can also be found in Figure 12(a)

Fig. 13. Proposed detection and correction unit of reversible BCD adder.

detection unit works on the condition that can be expressed by the Boolean function $OC = Cout + K_3 \cdot K_2 + K_3 \cdot K_1$. When OC (output carry) is equal to zero, nothing is added to the binary sum. When it is equal to one, binary 0110 is added to the binary sum using the correction unit (another 4-bit binary adder).

As illustrated in Figure 12(a), the binary adder produces a result that may not be in correct BCD format and needs to be converted to BCD format through the use of a detection and correction unit. Instead of using the detection and the correction unit to convert the result of summation to BCD format, the outputs of the binary adder can be passed to a binary to BCD converter to have the result of the binary addition in BCD format [Mohammadi et al. 2009]. This approach is illustrated in Figure 12(b) where the 5-bit binary to BCD converter produces the desired output in BCD format. In this work, we have proposed the equivalent reversible design of these two approaches to design the reversible BCD adder optimized for the number of ancilla input bits and the number of garbage outputs.

6.2. Design of Reversible BCD Adder Based on Approach 1

We present two designs of reversible BCD adders based on approach 1 with and without input carry c_0 (the conventional irreversible design of the 1-digit BCD adder based on approach 1 is illustrated in Figure 12(a)).

6.2.1. Design 1 of Reversible BCD Adder with Input Carry. As can be observed from Figure 12(a), in order to have this design, we need the design of a 4-bit reversible adder with input carry, the design of which is already illustrated in Figure 11. Next we present the new reversible design of the detection unit. As illustrated before, the detection unit uses the Boolean function $OC = Cout + K_3 \cdot K_2 + K_3 \cdot K_1$ as the checking condition. It can be written as $OC = Cout + K_3(K_2 + K_1)$. On careful observation it can be reduced to $OC = Cout \oplus K_3(K_2 + K_1)$ as Cout and $K_3(K_2 + K_1)$ cannot be true at the same time [Biswas et al. 2008]. We have designed the reversible detection unit based on the modified Boolean equation $OC = Cout \oplus K_3(K_2 + K_1)$ using NOT, CNOT, the Peres gate, and the TR gate. Before explaining the reversible design of the detection unit, we would like to emphasize a very useful property of the TR gate in relation to the popular Peres gate. We derive the inverse of the TR gate since a reversible gate can be combined with its inverse reversible gate to minimize the garbage outputs [Fredkin and Toffoli 1982]. In order to derive the logic equations of the inverse TR gate, we performed the reverse mapping of the TR gate outputs working as inputs to generate the inputs of the TR gate. We observe that the inverse of the TR gate is the

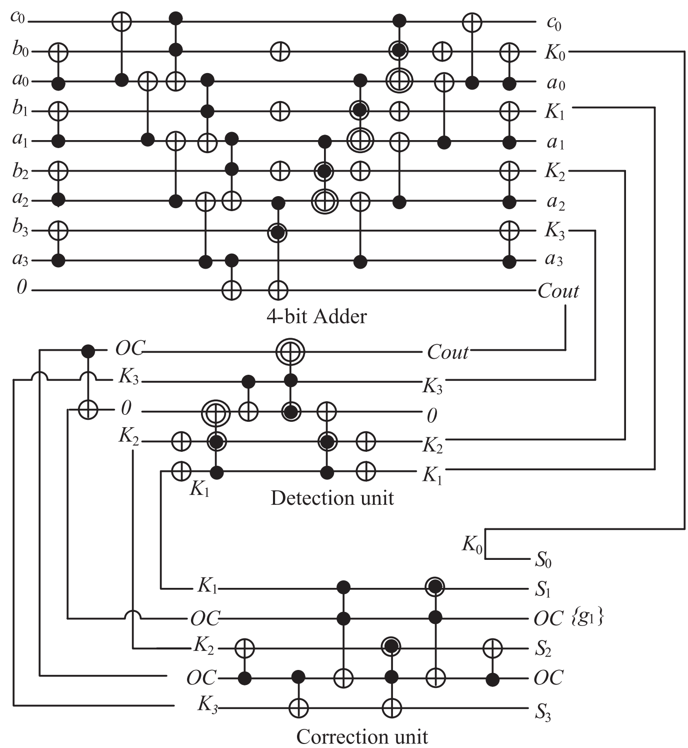


Fig. 14. Proposed design of 1-digit reversible BCD adder with input carry (RBCD-1) based on approach 1. The design consists of a 4-bit reversible binary adder with input carry illustrated in Figure 11, reversible detection unit illustrated in Figure 13(a), and reversible correction unit illustrated in Figure 13(a). g_1 which is the extra copy of the OC output represents the only garbage output. There are two ancilla inputs with constant value of 0. The copy of the OC signal is made through a Feynman gate as fanout is not allowed in reversible logic.

same as the existing Peres gate having inputs to outputs mapping as (P = A, Q = A $\oplus$ B, $R = A \cdot B \oplus C$. Thus, the TR gate and the Peres gate are inverse of each other.

The design of the reversible detection unit is illustrated in Figure 13(a) in which by using the TR gate as the inverse of the Peres gate we are able to regenerate K_1, K_2, K_3 to be used further in the correction unit. Further, the ancilla input bit having the constant value as "0" is regenerated to be used further in the correction unit. In the design, first with the help of the NOT gate and the Peres gate the output $\bar{K_1} \cdot \bar{K_2}$ is generated which is passed to the TR gate to generate the output $OC = Cout \oplus K_3(K_2 + K_1)$. Then with the help of the CNOT gate cascaded at the inputs A and B of the TR gate, the function $\bar{K_1} \cdot \bar{K_2}$ is regenerated. Finally, the TR gate combined with NOT gates is used to regenerate K_1, K_2, 0 outputs (here the final TR gate works as the inverse of the Peres gate). The reversible detection unit has the quantum cost of 17 and delay of 15Δ.

The design of the reversible correction unit is illustrated in Figure 13(b). The design of the correction unit is a 2-bit binary adder based on the methodology of the proposed reversible ripple carry adder without input carry as illustrated earlier in Section 4. In the design the 2-bit inputs of the adders are $a_0 = K_1, b_0 = OC$, $a_1 = K2, b_1 = OC$. In order to generate S_3 we have passed OC at the location z (a_3) of the reversible ripple carry adder as S3 can be generated as $K_3 \oplus C_3$. Since 0 needs to be added to K_0 to produce S_0, thus S_0 will be same as K_0 and hence a wire connection. The proposed design of the reversible correction unit does not need any ancilla input bit.

Table VII. A Comparison of n-Digit Reversible BCD Adders

	Ancilla input	Garbage outputs	Quantum cost	Delay Δ
[Babu and Chowdhury 2006]	17n	18n	110n	-
[Biswas et al. 2008]	7n	6n	55n	-
[Thomsen and Glück 2008]	4n	4n	169n	-
[Mohammadi et al. 2008]	14n	16n	84n	-
[Mohammadi et al. 2009] (Design 3*)	2n	6n	103n	-
This work 1 (with input carry)	2n	2n-1	88n	73n
This work 2 (without input carry)	2n	2n-1	88n-18	73n-1
This work 3 (with input carry)	n	n-1	70n	57n
This work 4 (without input carry)	n	n-1	70n-8	57n-3

1 represents proposed Design 1 of Reversible BCD adder with input carry

2 represents proposed Design 2 of Reversible BCD adder without input carry

3 represents proposed Design 3 of Reversible BCD adder without input carry

4 represents proposed Design 4 of Reversible BCD adder without input carry

*In [Mohammadi et al. 2009], 6 designs of the BCD adders are proposed varying in parameters of the number of ancilla inputs, garbage outputs, quantum cost and the delay. Among 6 designs, design 3 has the minimum number of ancilla inputs and the garbage outputs, thus we have compared to our work with design 3 of the [Mohammadi et al. 2009].

The reversible correction unit has the quantum cost of 16 and delay of 16Δ. The modules designed earlier can be integrated together to design the 1-digit reversible BCD adder as illustrated in Figure 14. The design of Figure 14 will be used with name RBCD-1 further in this work. The proposed design contains the 4-bit reversible adder with input carry, reversible detection unit, and reversible correction unit. A Feynman gate is used to avoid the fanout of OC signal as fanout is not allowed in reversible logic. It can be observed that the proposed reversible BCD adder design uses two ancilla input bits, and generates 1 garbage output labeled as g1 in Figure 14 which is the copy of the Output Carry (OC) that will not be used further in the computation (the inputs regenerated at the outputs are not considered as garbage outputs). The design has the quantum cost of 88 which is the summation of the quantum cost of the 4-bit reversible adder with input carry, reversible detection unit, 1 Feynman gate, and reversible correction unit. Further, the design has the delay of 73Δ which is the summation of the propagation delay of the 4-bit reversible adder with input carry, reversible detection unit, 1 Feynman gate, and reversible correction unit.

Once we have designed the 1-digit reversible BCD adder, the n-digit reversible BCD adder with input carry can be designed by cascading of the 1-digit reversible BCD adder (RBCD-1) in ripple carry fashion as illustrated in Figure 16(a). Thus, the design 1 of n-digit reversible BCD adder with input carry has $2n$ ancilla input bits, $2n-1$ garbage outputs, quantum cost of $88n$, and delay of $73n\Delta$. As shown in Figure 16(a) the design 1 of the n-digit reversible BCD adder with input carry has $2n-1$ garbage outputs because the first 1-digit BCD adder will have 1 garbage output (extra OC output), while the remaining $n-1$ 1-digit BCD adders each will have two garbage outputs. The extra one garbage output in each $n-1$ 1-digit reversible BCD adder is from the ripple carry regenerated at the outputs, for example the second 1-digit BCD adder in Figure 16(a) has two garbage outputs labeled as g2 and g3, the extra garbage output g2 is the output carry OC1 of the first 1-digit BCD adder that is passed to the second 1-digit BCD adder as input carry and is regenerated at one of its outputs.

Table VII illustrates that the proposed design is better than the existing design in terms of ancilla input bits and garbage outputs while also being efficient in terms of the quantum cost and the delay.

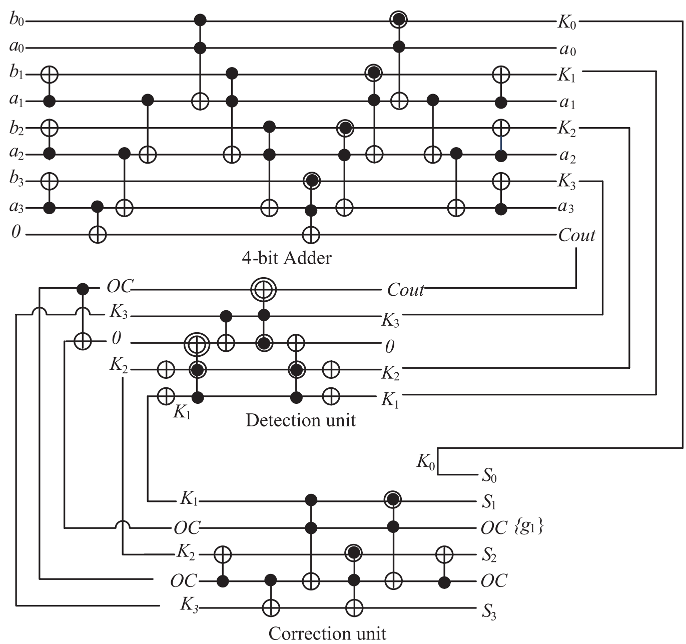
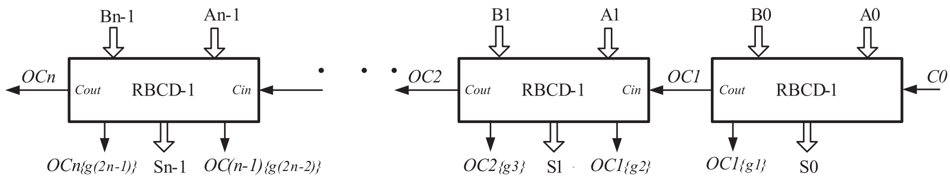


Fig. 15. Proposed design of 1-digit reversible BCD adder with no input carry (RBCD-2) based on approach 1. The design consists of a 4-bit reversible binary adder without input carry illustrated in Figure 8, reversible detection unit illustrated in Figure 13(a), and reversible correction unit illustrated in Figure 13(b). There are two ancilla inputs with constant value of 0 and g_1 represents the only garbage output. The copy of the OC signal is made through a Feynman gate as fanout is not allowed in reversible logic.

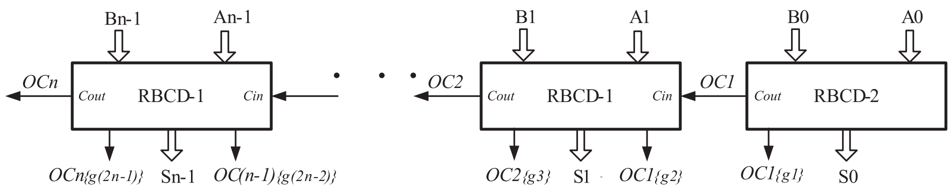
6.2.2. Design 2 of Reversible BCD Adder with No Input Carry. We have also designed a 1-digit reversible BCD adder based on approach 1 having no input carry. For achieving the design efficiently in terms of number of ancilla input bits and the garbage outputs, we have used the 4-bit reversible input adder without any input carry based on the methodology proposed in this work (the design can be seen in Figure 8). The rest of the design is the same as the design of the reversible BCD adder with input carry. The complete design of the 1-digit reversible BCD adder with no input carry is illustrated in Figure 15, and will be used with name RBCD-2 further in this work. The design has only 2 ancilla input bits and needs 1 garbage output. The design has the quantum cost of 80 and delay of 80Δ. Once we have designed the 1-digit reversible BCD adder with no input carry (RBCD-2), the n-digit reversible BCD adder with no input carry can be designed by utilizing the 1-digit reversible BCD adder with no input carry (RBCD-2) to add the least significant digit and then cascading $n-1$ 1-digit reversible BCD adder with input carry (RBCD-1) in ripple carry fashion. The design of an n-digit reversible BCD adder with no input carry is illustrated in Figure 16(b). Thus, the design 2 of an n-digit reversible BCD adder without input carry has $2n$ ancilla input bits, $2n-1$ garbage outputs, quantum cost of $88n-18$, and delay of $73n-1\Delta$. Table VII illustrates that the proposed design is better than the existing design in terms of number of ancilla input bits and the garbage outputs while also being efficient in terms of the quantum cost and delay.

6.3. Design of Reversible BCD Adder Based on Approach 2

In order to design the reversible BCD adder based on approach 2, we need a 4-bit reversible adder and a 5-bit reversible binary to BCD converter as illustrated in Figure 12(b).



(a) Proposed design 1 of n digit reversible BCD adder with input carry



(b) Proposed design 2 of n digit reversible BCD adder with no input carry

Fig. 16. Proposed designs of n-digit reversible BCD adder based on approach 1.

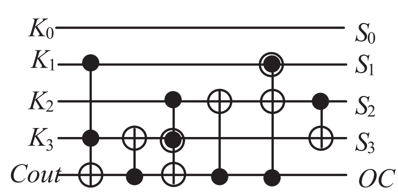


Fig. 17. Design of binary to BCD converter [Mohammadi et al. 2009].

6.3.1. Design 3 of Reversible BCD Adder with Input Carry. We first present the design of the 1-digit reversible BCD adder with input carry. The 4-bit reversible ripple carry adder with input carry shown in Figure 11 is used in the design. Recently, an efficient reversible binary to BCD converter without any ancilla bit is proposed in Mohammadi et al. [2009] which we have used in our design. The reversible binary to BCD converter proposed in Mohammadi et al. [2009] is illustrated in Figure 17 and has the quantum cost of 16 and delay of 16Δ. The proposed design of the 1-digit reversible BCD adder with input carry (c_0) is shown in Figure 18 and will be used with name RBCD-3 in this work. The design has 1 ancilla input bit and zero garbage outputs. The quantum cost of the proposed reversible BCD adder with input carry is 70 while the delay is 57Δ. Once we have designed the 1-digit reversible BCD adder (RBCD-3), the n-digit reversible BCD adder with input carry can be designed by cascading of the 1-digit reversible BCD adder in ripple carry fashion as illustrated in Figure 20(a). Thus, the design 3 of the n-digit reversible BCD adder with input carry has n ancilla input bits, $n-1$ garbage outputs, quantum cost of $70n$ and delay of $57n\Delta$. As shown in Figure 20(a) the design 3 of the n-digit reversible BCD adder with input carry has $n-1$ garbage outputs as the first 1-digit reversible BCD adder will have no garbage output while the remaining $n-1$ 1-digit reversible BCD adders each will have 1 garbage output. This is because the output carry of a BCD adder will work as the input carry to the next one and will be regenerated at the outputs. These regenerated input carries at the outputs will not be used further in the computation and hence will form garbage bits. Table VII illustrates that the proposed design is better than the existing design in terms of number of ancilla input bits and garbage outputs while also being efficient in terms of the quantum cost and delay.

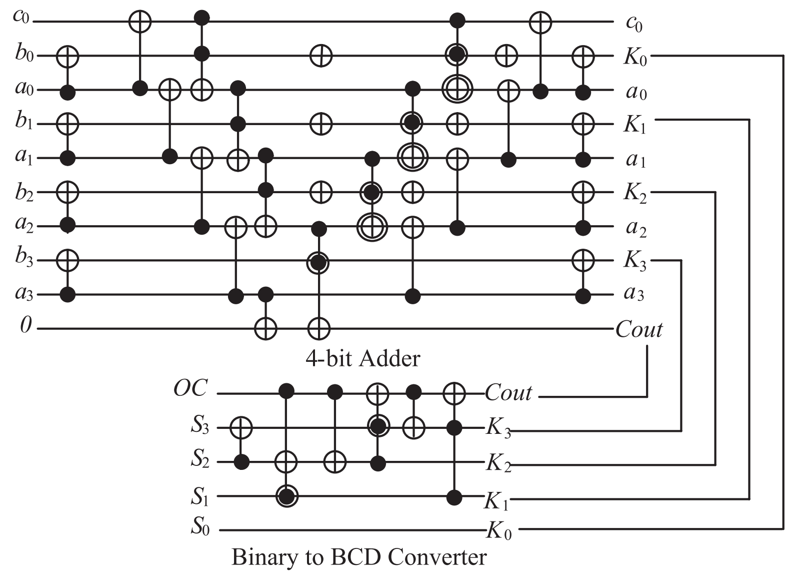


Fig. 18. Proposed design of reversible BCD adder with input carry (RBCD-3) based on approach 2. The design consists of 4-bit reversible binary adder with input carry illustrated in Figure 11 and reversible binary to BCD converter illustrated in Figure 17. $g_1, g_2, \ldots g_5$ represent the 5 garbage outputs. There is one ancilla input with constant value of 0.

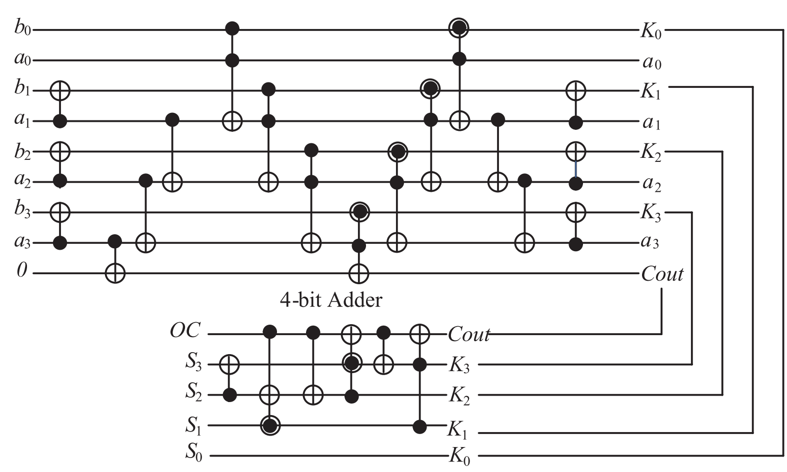
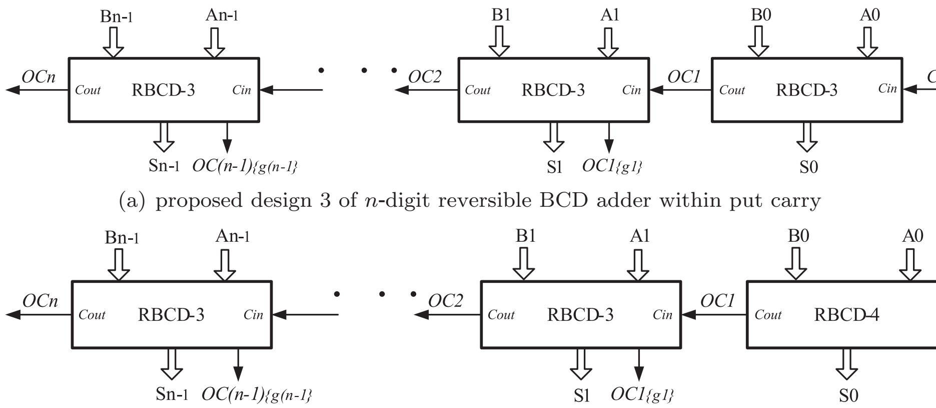


Fig. 19. Proposed design of reversible BCD adder with no input carry (RBCD-4) based on approach 2. The design consists of 4-bit reversible binary adder with no input carry illustrated in Figure 8 and reversible binary to BCD converter illustrated in Figure 17. $g_1, g_2, \ldots g_4$ represent the 4 garbage outputs. There is one ancilla input with constant value of 0.



(a) proposed design 3 of n-digit reversible BCD adder within put carry

(b) proposed design 4 of n-digit reversible BCD adder with no input carry (RBCD-4)

Fig. 20. Proposed designs of n-digit reversible BCD adder based on approach 2.

6.3.2. Design 4 of Reversible BCD Adder with No Input Carry. We are also proposing another design of the n-digit reversible BCD adder that has no input carry. We first design the 1-digit reversible BCD adder with no input carry which is shown in Figure 19, and will be used with name RBCD-4 further in this work. The design uses the 4-bit reversible ripple carry adder with no input carry illustrated in Figure 8 along with the design of the reversible binary to BCD converter illustrated in Figure 17. The proposed design of a 1-digit reversible BCD adder has 1 ancilla input bit and has zero garbage outputs. The quantum cost of the design is 62 while the propagation delay is 54Δ. Once we have designed the 1-digit reversible BCD adder with no input carry, the n-digit reversible BCD adder with no input carry can be designed by utilizing the 1-digit reversible BCD adder with no input carry (RBCD-4) to add the least significant digit and then cascading $n-1$ 1-digit reversible BCD adder with input carry (RBCD-3) in ripple carry fashion. The design of an n-digit reversible BCD adder with no input carry is illustrated in Figure 20(b). Thus, the design 4 of an n-digit reversible BCD adder without input carry has n ancilla inputs bits, $n-1$ garbage outputs, quantum cost of $70n-8$, and delay of $57n-3\Delta$. Table VII illustrates that the proposed design is better than the existing design in terms of number of ancilla input bits and the garbage outputs while also being efficient in terms of the quantum cost and delay.

6.4. Comparison of n-Digit Reversible BCD Adders

All the existing designs of the reversible BCD adders are with input carry c_0. Among our proposed design of an n-digit reversible BCD adder with input carry, design 3 has the minimum number of ancilla input bits, garbage outputs, quantum cost, and delay. The results of all the existing works and our proposed work are summarized in Table VII. Among the existing works shown in Table VII for the design of an n-digit reversible BCD adder, the design presented in Thomsen and Glück [2008] has the minimum number of garbage outputs, while design 3 presented in Mohammadi et al. [2009] has the minimum number of ancilla inputs. Thus we have shown the comparison of our proposed work with the design presented in Thomsen and Glück [2008] and Mohammadi et al. [2009] in Tables VIII, IX, and X in terms of number of ancilla inputs, number of garbage outputs, and the quantum cost, respectively, for values of n ranging from n=8 digits to n=512 digits. It is to be observed that delays of Thomsen and Glück [2008] and design 3 of Mohammadi et al. [2009] are not known, thus we are not able to compare our design with these designs in terms of delay. From Table VIII, it can

Table VIII. Ancilla Inputs Comparison of n-Digit Reversible BCD Adders

Digits	1	2	Proposed*	% Impr. w.r.t 1	% Impr. w.r.t 2
8	32	16	8	75	50
16	64	32	16	75	50
32	128	64	32	75	50
64	256	128	64	75	50
128	512	256	128	75	50
256	1024	512	256	75	50
512	2048	1024	512	75	50

1 is the design in [Thomsen and Glück 2008]

2 is the design 3 in [Mohammadi et al. 2009]

* is our design 3 proposed in this work. In improvement calculation all the existing designs are compared with the proposed Design 3 of the proposed reversible BCD adder as among the proposed design it has minimal number of anicilla inputs , garbage outputs, quantum cost and the delay.

Table IX. Garbage Outputs Comparison of n-Digit Reversible BCD Adders

Digits	1	2	Proposed*	% Impr. w.r.t 1	% Impr. w.r.t 2
8	32	48	7	78.12	85.4
16	64	96	15	76.56	84.37
32	128	192	31	75.78	83.85
64	256	384	63	75.39	83.59
128	512	768	127	75.19	83.46
256	1024	1536	255	75.09	83.39
512	2048	3072	511	75.04	83.36

1 is the design in [Thomsen and Glück 2008]

2 is the design 3 in [Mohammadi et al. 2009]

* is our design 3 proposed in this work. In improvement calculation all the existing designs are compared with the proposed Design 3 of the proposed reversible BCD adder as among the proposed design it has minimal number of anicilla inputs , garbage outputs, quantum cost and the delay.

be observed that in terms of number of ancilla inputs, the proposed design 3 achieves improvement ratios of 75% and 50% compared to the design presented in Thomsen and Glück [2008] and Mohammadi et al. [2009], respectively. The improvement ratios in terms of number of garbage outputs range from 78.12% to 75.04%, and 85.4% to 83.36% compared to the design presented in Thomsen and Glück [2008] and Mohammadi et al. [2009], respectively, where the details are illustrated in Table IX. As illustrated in Table X, the improvement ratios in terms of quantum cost are 58.57%, and range from 34.88% to 32.03% compared to the design presented in Thomsen and Glück [2008] and Mohammadi et al. [2009], respectively. Thus the proposed designs of reversible BCD adders are efficient in terms of number of ancilla inputs, garbage outputs, quantum cost, and delay compared to the existing designs in literature.

7. SIMULATION AND VERIFICATION

The proposed reversible ripple carry adder designs, reversible detection unit, reversible correction units, reversible binary to BCD converter, and the complete working designs of the reversible BCD adders are functionally verified through simulations. The simulation is performed by creating a library of reversible gates in Verilog Hardware

Table X. Quantum Cost Comparison of n-Digit Reversible BCD Adders

Digits	1	2	Proposed*	% Impr. w.r.t 1	% Impr. w.r.t 2
8	1352	824	560	58.57	34.88
16	2704	1648	1120	58.57	32.03
32	5408	3296	2240	58.57	32.03
64	10816	6592	4480	58.57	32.03
128	21632	13184	8960	58.57	32.03
256	43264	26368	17920	58.57	32.03
512	86528	52736	35840	58.57	32.03

1 is the design in [Thomsen and Glück 2008]

2 is the design 3 in [Mohammadi et al. 2009]

* is our design 3 proposed in this work. In improvement calculation all the existing designs are compared with the proposed Design 3 of the proposed reversible BCD adder as among the proposed design it has minimal number of anicilla inputs , garbage outputs, quantum cost and the delay.

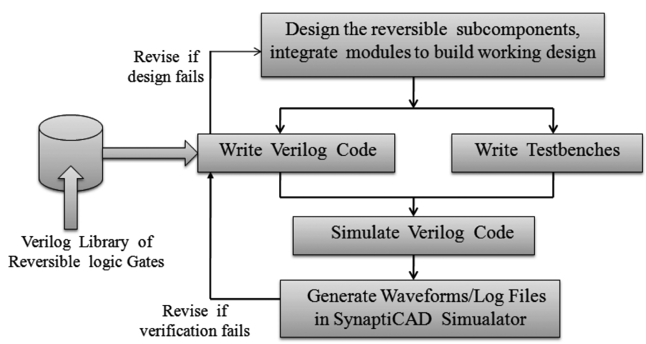


Fig. 21. Simulation flow of reversible circuits using Verilog HDL.

Description Language and is used to code the proposed reversible designs. The Verilog library contains the Verilog codes of reversible gates such as the Fredkin gate, the Toffoli gate, the Peres gate, the TR gate, the Feynman gate, etc. All the reversible designs of the adders and the subcomponents are coded in Verilog HDL by utilizing the reversible gates from the Verilog library of reversible gates. The testbenches are created for every reversible circuit proposed in this work and for 4-bit and 8-bit reversible binary adders, and 1-digit reversible BCD adders where exhaustive simulations are done to verify the correctness. The simulation flow used in this work is illustrated in Figure 21. The ModelSim and the SynaptiCAD simulators are used for the functional verification of the Verilog HDL codes. The waveforms are generated using the SynaptiCAD Verilog simulator.

8. CONCLUSIONS

In this work, we have presented efficient designs of reversible ripple carry binary and BCD adders primarily optimizing the parameters of number of ancilla input bits and the garbage outputs. The optimization of the quantum cost and the delay are also considered. The reversible designs of subcomponents used in the BCD adder design such as detection unit, correction unit, and the binary to BCD converter are also illustrated. The proposed reversible binary and BCD adders designs are shown to be better than the existing designs in terms of the number of ancilla input bits and the garbage outputs while maintaining the lower quantum cost and the delay. We conclude that the use of the specific reversible gates for a particular combinational function can be very beneficial in minimizing the number of ancilla input bits, garbage outputs, quantum cost, and the delay. All the proposed reversible designs are functionally verified at the logical level by using the Verilog hardware description language and the HDL simulators. The proposed efficient designs of reversible binary and BCD adders will find applications in quantum/reversible computing requiring BCD arithmetic units.

ACKNOWLEDGMENTS

We would like to express our sincere thanks to the anonymous reviewers for their critical suggestions which helped in improving the manuscript.

REFERENCES

BABU, H. M. AND CHOWDHURY, A. 2006. Design of a compact reversible binary coded decimal adder circuit. *Elsevier J. Syst. Archit. 52*, 272–282.

BAYRAKCI, A. AND AKKAS, A. 2007. Reduced delay bcd adder. http://www.csie.ncue.edu.tw/csie/ic/Seminar_new/9701/971020/971020_02.pdf.

BISWAS, A. K., HASAN, M. M., CHOWDHURY, A. R., AND HASAN BABU, H. M. 2008. Efficient approaches for designing reversible binary coded decimal adders. *Microelectron. J. 39,* 12, 1693–1703.

BRUCE, J. W., THORNTON, M. A., SHIVAKUMARAIAH, L., KOKATE, P. S., AND LI, X. 2002. Efficient adder circuits based on a conservative reversible logic gate. In *Proceedings of the IEEE Symposium on VLSI.* 83–88.

CHANG, L., FRANK, D. J., MONTOYE, R. K., KOESTER, S. J., JI, B. L., COTEUS, P. W., DENNARD, R. H., AND HAENSCH, W. 2010. Practical strategies for power-efficient computing technologies. *Proc. IEEE 98,* 2, 215–236.

CHUANG, M.-L. AND WANG, C.-Y. 2008. Synthesis of reversible sequential elements. *J. Emerg. Technol. Comput. Syst. 3,* 4, 1–19.

COWLISHAW, M. 2003. Decimal floating-point: Algorism for computers. In *Proceedings of the IEEE Symposium on Computer Arithmetic*. 104–111.

COWLISHAW, M. 2010. Decimal arithmetic faq part 3 hardware questions. http://speleotrove.com/decimal/decifaq3.html.

CUCCARO, S. A., DRAPER, T. G., KUTIN, S. A., AND MOULTON, D. P. 2004. A new quantum ripple-carry addition circuit. http://arXiv.org/quant-ph/0410184.

DESOETE, B. AND VOS, A. D. 2002. A reversible carry-look-ahead adder using control gates. *Integr. VLSI J. 33,* 1, 89–104.

FRANK, M. 2005. Approaching the physical limits of computing. In *Proceedings of the 35th International Symposium on Multiple-Valued Logic (ISMVL'05).* 168–185.

FREDKIN, E. AND TOFFOLI, T. 1982. Conservative logic. *Int. J. Theor. Phys. 21*, 219–253.

GROSSE, D., WILLE, R., DUECK, G., AND DRECHSLER, R. 2009. Exact multiple control toffoli network synthesis with sat techniques. *IEEE Trans. Comput.-Aid. Des. 28,* 5, 703–715.

GROSSE, D., WILLE, R., DUECK, G. W., AND DRECHSLER, R. 2008. Exact synthesis of elementary quantum gate circuits for reversible functions with dont cares. In *Proceedings of the International Symposium on Multi-Valued Logic*. 214–219.

GUPTA, P., AGARWAL, A., AND JHA, N. K. 2006. An algorithm for synthesis of reversible logic circuits. *IEEE Trans. Comput.-Aid. Des. 25,* 11, 2317–2330.

HAGHPARAST, M., JASSBI, S., NAVI, K., AND HASHEMIPOUR, O. 2008. Design of a novel reversible multiplier circuit using hng gate in nanotechnology. *World Appl. Sci. J. 3,* 6, 974–978.

HARI, S. K. S., SHROFF, H. S., MAHAMMAD, S. N., AND KAMAKOTI, V. 2006. Efficient building blocks for reversible sequential circuit design. In *Proceedings of the 49th IEEE International Midwest Symposium on Circuits and Systems.* 437–441.

HUNG, W. N., SONG, X., YANG, G., YANG, J., AND PERKOWSKI, M. 2006. Optimal synthesis of multiple output boolean functions using a set of quantum gates by symbolic reachability analysis. *IEEE Trans. Comput.-Aid. Des. 25,* 9, 1652–1663.

JAMES, R. K., JACOB, K. P., AND SASI, S. 2008. Reversible binary coded decimal adders using toffoli gates. In *Advances in Computational Algorithms and Data Analysis*, Lecture Notes in Electrical Engineering, vol. 14, Springer, 117–131.

KHAN, M. 2002. Design of full-adder with reversible gates. In *Proceedings of the International Conference on Computer and Information Technology*. 515–519.

KHAN, M. H. A. AND PERKOWSKI, M. A. 2007. Quantum ternary parallel adder/subtractor with partially-look-ahead carry. *J. Syst. Archit. 53,* 7, 453–464.

MA, X., HUANG, J., METRA, C., AND LOMBARDI, F. 2008. Reversible gates and testability of one dimensional arrays of molecular qca. *J. Electron. Test. Theory Appl 24,* 1–3, 1244–1245.

MA, X., HUANG, J., METRA, C., AND LOMBARDI, F. 2009. Detecting multiple faults in one-dimensional arrays of reversible qca gates. *J. Electron. Test. Theory Appl. 25,* 1, 39–54.

MASLOV, D. AND DUECK, G. W. 2004. Reversible cascades with minimal garbage. *IEEE Trans. Comput.-Aid. Des. 23,* 11, 1497–1509.

MASLOV, D. AND MILLER, D. M. 2006. Comparison of the cost metrics for reversible and quantum logic synthesis. http://arxiv.org/abs/quant-ph/0511008.

VAN METER, R., MUNRO, W., NEMOTO, K., AND ITOH, K. M. 2009. Arithmetic on a distributed memory quantum multicomputer. http://arxiv.org/abs/quant-ph/0607160.

MOHAMMADI, M. AND ESHGHI, M. 2009. On figures of merit in reversible and quantum logic designs. *Quantum Inf. Process. 8,* 4, 297–318.

MOHAMMADI, M., ESHGHI, M., HAGHPARAST, M., AND BAHROLOLOOM, A. 2008. Design and optimization of reversible bcd adder/subtractor circuit for quantum and nanotechnology based systems. *World Appl. Sci. J. 4,* 6, 787–792.

MOHAMMADI, M., HAGHPARAST, M., ESHGHI, M., AND NAVI, K. 2009. Minimization optimization of reversible bcd-full adder/subtractor using genetic algorithm and don't care concept. *Int. J. Quantum Inf. 7,* 5, 969–989.

NIELSEN, M. A. AND CHUANG, I. L. 2000. *Quantum Computation and Quantum Information*. Cambridge University Press, London, UK.

PARHAMI, B. 2006. Fault-tolerant reversible circuits. In *Proceedings of the 40th Asilomar Conference on Signals, Systems, and Computers*. 1726–1729.

PARHAMI, B. 2010. *Computer Arithmetic: Algorithms and Hardware Designs* 2nd Ed. Oxford University Press, Oxford, UK.

PERES, A. 1985. Reversible logic and quantum computers. *Phys. Rev. A, Gen. Phys. 32,* 6, 3266–3276.

PRASAD, A. K., SHENDE, V., MARKOV, I., HAYES, J., AND PATEL, K. N. 2006. Data structures and algorithms for simplifying reversible circuits. *ACM J. Emerg. Technol. Comput. Syst. 2,* 4, 277–293.

RICE, J. E. 2008. An introduction to reversible latches. *Comput. J. 51,* 6, 700–709.

SHENDE, V. V., PRASAD, A., MARKOV, I., AND HAYES, J. 2003. Synthesis of reversible logic circuits. *IEEE Trans. Comput.-Aid. Des. Autom. Des. Electron. Circ. 22*, 6, 710–722.

SMOLIN, J. A. AND DI VINCENZO, D. P. 1996. Five two-bit quantum gates are sufficient to implement the quantum fredkin gate. *Phys. Rev. A 53*, 4, 2855–2856.

TAKAHASHI, Y. 2010. Quantum arithmetic circuits: A survey. *IEICE Trans. Fundam. E92-A,* 5, 276–1283.

TAKAHASHI, Y. AND KUNIHIRO, N. 2005. A linear-size quantum circuit for addition with no ancillary qubits. *Quantum Inf. Comput. 5,* 6, 440448.

TAKAHASHI, Y., TANI, S., AND KUNIHIRO, N. 2009. Quantum addition circuits and unbounded fan-out. http://arxiv.org/abs/0910.2530.

TARAPHDARA, C., CHATTOPADHYAY, T., AND ROY, J. 2010. Machzehnder interferometer-based all-optical reversible logic gate. *Optics Laser Technol. 42,* 2, 249–259.

THAPLIYAL, H. AND RANGANATHAN, N. 2009. Design of efficient reversible binary subtractors based on a new reversible gate. In *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*. 229–234.

THAPLIYAL, H. AND RANGANATHAN, N. 2010a. Design of reversible sequential circuits optimizing quantum cost, delay and garbage outputs. *ACM J. Emerg. Technol. Comput. Syst. 6,* 4, 14:1–14:35.

THAPLIYAL, H. AND RANGANATHAN, N. 2010b. Reversible logic-based concurrently testable latches for molecular qca. *IEEE Trans. Nanotechnol. 9,* 1, 62–69.

THAPLIYAL, H. AND RANGANATHAN, N. 2011. A new reversible design of bcd adder. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'11)*.

THAPLIYAL, H., RANGANATHAN, N., AND FERREIRA, R. 2010. Design of a comparator tree based on reversible logic. In *Proceedings of the 10th IEEE International Conference on Nanotechnology*. 1113–1116.

THOMSEN, M. AND GLUCK, R. 2008. Optimized reversible binary-coded decimal adders. *J. Syst. Archit. 54,* 7, 697–706.

TOFFOLI, T. 1980. Reversible computing. Tech. memo MIT/LCS/TM-151, MIT Lab for Computer Science.

TRISETYARSO, A. AND VAN METER, R. 2009. Circuit design for a measurement-based quantum carry-lookahead adder. http://arxiv.org/abs/0903.0748.

VEDRAL, V., BARENCO, A., AND EKERT, A. 1996. Quantum networks for elementary arithmetic operations. *Phys. Rev. A 54,* 1, 147–153.

WANG, L., ERLE, M., TSEN, C., SCHWARZ, E. M., AND SCHULTE, M. J. 2010. A survey of hardware designs for decimal arithmetic. *IBM J. Res. Devel. 54,* 2, 8:1–8:15.

WILLE, R., SOEKEN, M., AND DRECHSLER, R. 2010. Reducing the number of lines in reversible circuits. In *Proceedings of the 47th Design Automation Conference (DAC)*. 647–652.

YANG, G., SONG, X., HUNG, W. N., AND PERKOWSKI, M. 2008. Bi-directional synthesis of 4-bit reversible circuits. *Comput. J. 51,* 2, 207–215.

Received October 2010; revised February 2011; accepted April 2011