International Journal of Electrical and Computer Engineering (IJECE)
Vol. 13, No. 2, April 2023, pp. 2009~2022
ISSN: 2088-8708, DOI: 10.11591/ijece.v13i2.pp2009-2022

The use of reversible logic gates in the design of residue number systems

Ailin Asadpour[1], Amir Sabbagh Molahosseini[1], Azadeh Alsadat Emrani Zarandi[2]
[1]Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman, Iran
[2]Department of Computer Engineering, Shahid Bahonar University of Kerman, Kerman, Iran

Article Info

Article history:

Received Feb 10, 2022
Revised Sep 15, 2022
Accepted Oct 13, 2022

Keywords:

Arithmetic
Digital circuits
Digital signal processors
Residue number system
Reversible circuits

ABSTRACT

Reversible computing is an emerging technique to achieve ultra-low-power circuits. Reversible arithmetic circuits allow for achieving energy-efficient high-performance computational systems. Residue number systems (RNS) provide parallel and fault-tolerant additions and multiplications without carry propagation between residue digits. The parallelism and fault-tolerance features of RNS can be leveraged to achieve high-performance reversible computing. This paper proposed RNS full reversible circuits, including forward converters, modular adders and multipliers, and reverse converters used for a class of RNS moduli sets with the composite form {2^k, 2^p-1}. Modulo 2^n-1, 2^n, and 2^n+1 adders and multipliers were designed using reversible gates. Besides, reversible forward and reverse converters for the 3-moduli set {2^n-1, 2^{n+k}, 2^n+1} have been designed. The proposed RNS-based reversible computing approach has been applied for consecutive multiplications with an improvement of above 15% in quantum cost after the twelfth iteration, and above 27% in quantum depth after the ninth iteration. The findings show that the use of the proposed RNS-based reversible computing in convolution results in a significant improvement in quantum depth in comparison to conventional methods based on weighted binary adders and multipliers.

Corresponding Author:

Amir Sabbagh Molahosseini
Department of Computer Engineering, Kerman Branch, Islamic Azad University
Kerman, Iran
Email: sabbagh@iauk.ac.ir

1. INTRODUCTION

As the validity of Moore’s law gradually diminishes, the yearly exponential computer performance improvement gets slower [1]. Therefore, computer architects increasingly investigate alternative methods at different design abstraction levels, including arithmetic circuits. These methods are used to afford high-performance computing for emerging applications, embedded deep learning, [2] and the internet of things (IoT) [3]. At the arithmetic level, the conventional weighted binary number representation, which is based on the primary microprocessor design, is still prevalent. However, alternative number systems such as the residue number system (RNS) [4] have attracted attention in recent years. The RNS has been known as a powerful tool to break the long carry-propagation chain and parallelize the arithmetic operations. It is useful in various applications, including embedded systems and digital signal processing. Besides, redundant RNS (RRNS) [5] has been used in many applications, including DNA arithmetic [6], wireless sensor networks, and fault-tolerant processor design [7]. RNS is useful in applications in which additions and multiplications are dominant [8]. The theoretically minimum possible energy consumption of a logic operation at room temperature is about 4.14 zepto-joules, and the conventional complementary metal oxide semiconductor (CMOS) technology cannot reach it because its limit is thousands of times higher [9]. Reversible computing makes it possible to reach the least energy dissipation by avoiding information loss, resulting in ultra-low challenge is how to design efficient circuits for reversible computing. Researchers have discussed the design power circuits [10]. Besides, reversible logic is the basis of quantum computing. The most important challenge is how to design efficient circuits for reversible computing. Researchers have discussed the design and implementation of reversible logic, including synthesis [11], adders [12], multipliers, dividers [13], and realization based on quantum-dot cellular automata (QCA) [14]. This paper addresses the parallelism of RNS with the low-power feature of reversible logic. Particularly, for demanding calculations on wide operands, the parallelism of RNS can outperform the conventional weighted number representation supported by reversible computation circuits with long carry propagation chains. A challenge for achieving this is to design and tune the RNS components that can be efficiently implemented with reversible logic instead of the traditional CMOS application-specific integrated circuits (ASIC). To this end, the first step is to design the reversible-RNS modulo 2^n-1 adders [12] as the cornerstone of RNS circuits. The main contributions of this paper for the selected wide c-class moduli sets with the composite form {2^k, 2^p-1} [15] are: i) to propose efficient modulo adders and multipliers based on reversible gates (RGs); ii) to design reversible forward and reverse converters, which are based on the suggested reversible modular adders, for the considered class of moduli sets; and iii) to apply the proposed R-RNS approach, as a case study, to consecutive multiplications, and also to evaluate the performance of R-RNS and traditional RNS computational structures.

In this paper, the fundamentals of RNS, the design of RNS circuits for sets of c-class moduli, and reversible computing concepts are briefly introduced in section 2. In section 3, reversible modular adders, multipliers, and forward and reverse converters are discussed, and the proposed approach based on the computation of a sequence of multiplications is evaluated. The last part, section 4, concludes the paper and presents the key findings.

2. METHOD

The following section briefly explains the basic concepts of RNS in c-class moduli set, moduli adders, modular multipliers, and forward and reverse converters. Also, the key concepts in reversible logic including the basic features of reversible circuits were described. Also, the characteristics of the reversible gates used in this article were examined.

2.1. The residue number system

A residue number system (RNS) [4] is planned and designed based on pairwise relatively prime numbers that compose the moduli set of the system {m_1, m_2, …, m_n}. The dynamic range in RNS is defined as (0, M) where:

$$M = m_1 \times m_2 \times \ldots \times m_n \quad (1)$$

After mapping the weighted representation of numbers into equivalent RNS representations, arithmetic operations on residues are performed as (2),

$$X \odot Y \xrightarrow{RNS} (|x_1 \odot y_1|_{m_1}, |x_2 \odot y_2|_{m_2}, \ldots, |x_n \odot y_n|_{m_n}) \quad (2)$$

where $\odot \in \{+, -, \times\}$ indicates that there is not any carry propagation between residues. Finally, in order to convert the result back to the weighted binary system, the Chinese remainder theorem (CRT) or the mixed-radix conversion (MRC) is applied [4]. The architecture of the RNS-based arithmetic system includes three components: i) forward converter, ii) RNS processing units, and iii) reverse converter.

2.2. RNS design for c-class moduli sets

The former class of moduli sets is usually known as c-class [16]. These kinds of moduli set share a modulo in the form of 2^k while the other moduli product presupposes the value 2^P-1. This results in a simple and efficient reverse converter hardware structure that is based on the new CRT-I [17]. In order to simplify the presentation of the circuits, the moduli set {2^n-1, 2^{n+k}, 2^n+1} [18] is selected for the case study.

2.2.1. Modular adders

The formulations for the modular addition of *x* and *y* of the c-class moduli set {2^n-1, 2^{n+k}, 2^n+1} are presented. For the modulo 2^n-1 addition of the residues *x* and *y*.

$$|x + y|_{2^n-1} = (x + x + carry)\, mod\, 2^n \tag{3}$$

Note that the adder in (3) is based on a ripple-carry adder (RCA) with the end-around carry (EAC), adopting a double representation of zero. For removing the double representation of zero, some cascaded AND gates can be used to detect all outputs equal to 1 [15]. For the modulo 2^{n+k} addition of the corresponding residues *x* and *y*:

$$|x + y|_{2^{n+k}} = \left| z_{n+k-1} 2^{n+k} + \underbrace{z_{n+k-1} \dots z_1 z_0}_{n+k bits} \right|_{2^{n+k}} = \underbrace{z_{n+k-1} \dots z_1 z_0}_{n+k bits} \tag{4}$$

where *z* is the regular sum of operands *x* and *y*. Finally, the modulo *2ⁿ+1* addition of residues *x* and *y*, adopting the diminished-one representation (x´=x-1; y´=y-1), is performed [19].

$$\left|x' + y' + 1\right|_{2^n+1} = \begin{cases} x' + y' if x' + y' \geq 2^n \\ (x' + y' + 1)\, mod\, 2^n\, if x' + y' < 2^n \end{cases} = (x' + y' + c_{out})\, mod\, 2^n \tag{5}$$

2.2.2. Modular multipliers

The modulo *2ⁿ-1* multiplication of residues x and y could be performed as in (6)[20].

$$\begin{aligned} |x \times y|_{2^n-1} &= \left|\textstyle\sum_{i=0}^{n-1} 2^i x_i \times (y_{n-1} \dots y_1 y_0)\right|_{2^n-1} \\ &= |\textstyle\sum_{i=0}^{n-1} x_i \times (y_{n-i-1} \dots y_0 y_{n-1} \dots y_{n-i})|_{2^n-1} = |\textstyle\sum_{i=0}^{n-1} PP_i|_{2^n-1} \end{aligned} \tag{6}$$

The details of partial products (PP_is) summations are presented in [20]. According to (6), the PP_is are added to achieve the result. This operation relies on a modulo 2^n-1 multi-operand adder structure that can be formed via some carry-save adders (CSAs) with end-around carry (EAC), with a regular two-operand modulo *2ⁿ-1* adder at the end. Thus, the modulo *2ⁿ* multiplication of residues *x* and *y* could be expressed [17]:

$$|x \times y|_{2^{n+k}} = \left| \underbrace{(c_{2(n+k)-1} \dots c_{n+k})}_{n+k bits} 2^{n+k} + \underbrace{c_{n+k-1}}_{n+k bits} \dots c_0 \right|_{2^{n+k}} = \underbrace{c_{n+k-1}}_{n+k bits} \dots c_0 \tag{7}$$

where *c* is the product of *x* and *y*. The modulo 2^n+1 multiplication of residues *x* and *y* that can be implemented with CSAs with complemented end-around carry (CEAC) is followed by a two-operand modulo 2^n+1 carry-propagate adder (CPA). This can be performed [20]:

$$|x \times y|_{2^n+1} = \left|\textstyle\sum_{i=0}^{n} 2^i x_i \times \sum_{j=0}^{n} 2^j y_j\right|_{2^n+1} = \left|\textstyle\sum_{i=0}^{n} \left(\sum_{j=0}^{n} PP_{i,j} 2^{i+j}\right)\right|_{2^n+1} \tag{8}$$

2.2.3. Forward and reverse converter

The range of c-class moduli sets {2^k, 2^P-1}, which is dynamic, is *k+P* bits. The main forward conversion for the considered moduli set {2^n-1,2^{n+k},2^n+1}, 0≤k≤n is [18]:

$$x_{i=} |X|_{m_i} = |X_{k+P-1} \dots X_0|_{m_i} \tag{9}$$

The reverse converter for the c-class (CRT) relies on the following general relation [15]:

$$X = x_1 + 2^k \left|\textstyle\sum_{i=1}^{k} v_i\right|_{2^P-1} \tag{10}$$

The full set of equations for reverse conversion of the residues x_1, x_2, x_3 to the equivalent weighted number X for the moduli set {2^n-1,2^{n+k},2^n+1} are [18]:

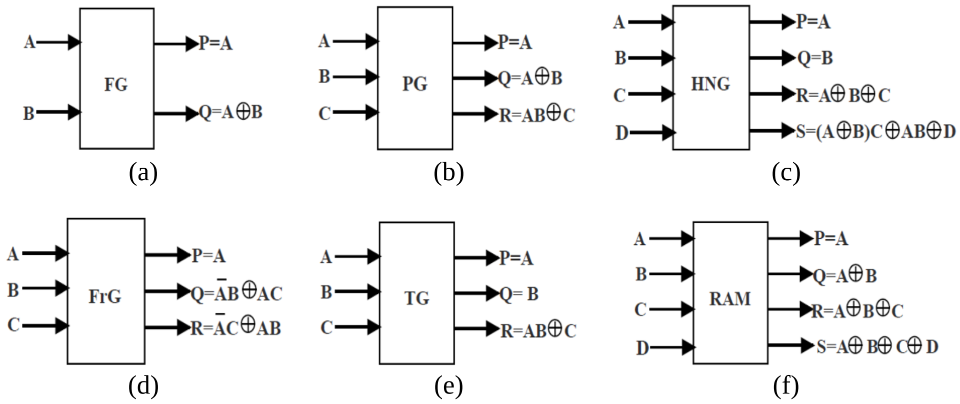
$$X = x_2 + 2^{n+k} Y \tag{11}$$

The hardware realization of Y is the one that requires two *2n*-bit CSAs with EACs and a modulo *2²ⁿ-1* adder. Note that since x_2 is an *n+k*-bit number, a concatenation (&) of x_2 with Y yields the final weighted number.

2.3. Reversible logic

Due to lack of information loss, reversible circuits can lead to ultra-low-power circuits. The number of inputs and outputs in these circuits are the same. Feedback is not allowed, and the fan-out is equal to 1 in reversible circuits. The most important reversible gates (RGs) used in this paper were introduced in Figure 1, including Feynman [10], Peres [21], HNG [12], Fredkin [22], Toffoli [10], and RAM [13], as shown in Figures 1(a) to 1(f).

The Feynman gate (FG) is known as a controlled-NOT gate. By adjusting inputs, the Peres gate (PG) and HNG may be used as a half adder (HA) and full adder (FA), respectively. The Toffoli gate (TG) can be used for copying inputs as well as AND-ing them. Fredkin gate (FrG) can perform OR operations. Finally, RAM [13] consists of multiple FGs, which can be used for copying a signal.



(a) (b) (c)

(d) (e) (f)

Figure 1. Reversible gates of (a) Feynman, (b) Peres, (c) HNG, (d) Fredkin, (e)Toffoli, and (f) RAM

3. RESULTS

3.1. The proposed reversible modular adders

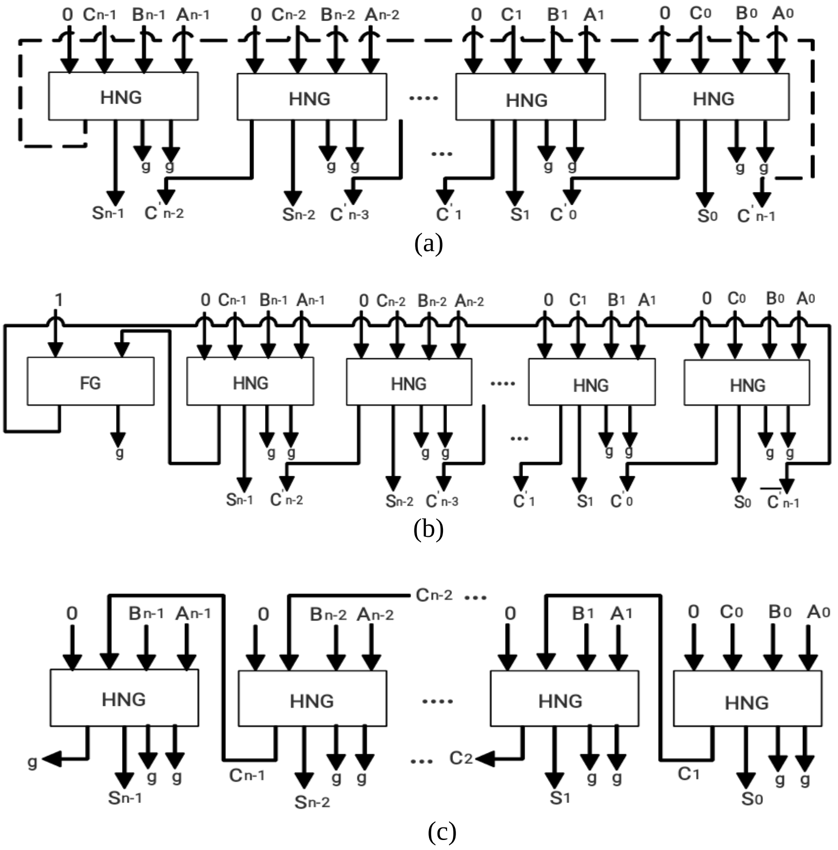
Two main types of adders are required in RNS: i) carry-save based adders, including CSA, CSA with EAC, and CSA with CEAC for multi-operand additions, and ii) CPA, which can be obtained using the ripple-carry approach, including RCA, RCA with EAC, and RCA with CEAC. This section first proposes reversible adders for the c-class moduli set. A new design is then proposed to reduce the number of constant inputs in reversible modulo 2^n adder.

3.1.1. Reversible adders for moduli set {2^n-1, 2^{n+k}, 2^n+1}

HNG [12] RGs are used to implement the FAs required in CSA, as shown in Figure 2(a). The quantum cost and depth of CSA and CSA with EAC for *n*-bit operands are 6n and 5Δ, respectively. Besides, the quantum cost and depth of CSA with CEAC are *6n+1* and 6Δ, respectively. The total constant inputs and garbage outputs in Figure 2(a) are *n* and *2n*, respectively, and those in Figure 2(b) are *n+1* and *2n+1*, respectively. The proposed reversible modulo 2^n adder in Figure 2(c) is based on the HNG-based RCA structure. The total quantum cost for the adder in Figure 2(c) is *6n*.

The quantum depth for the first HNG in an RCA is 5Δ. However, according to RcViewer simulations [23], connecting other series of (*n-1*) HNGs in a ripple carry architecture results in increasing the quantum depth by 3(*n-1*)Δ. The total quantum depth for the modulo 2^n adder in Figure 2(c) is (5+3(n-1))Δ. Moreover, the reversible RCA of Figure 2(c) possesses *n* and *2n+1* constant inputs and garbage outputs, respectively.

The reversible implementation of modulo 2^n*-1* with a single representation of zero relies on three levels: i) in the first level, there is an n-bit RCA adder that can be realized using reversible HNG gates; ii) in the second level, a series of AND gates are used to detect 1s and then ORing the carry out of the RCA, which can be realized using Toffoli gates (TG) structured in a tree and then a Fredkin gate (FrG) to perform OR with the carry-out; and finally, iii) in the third level, an n-bit ripple connected HAs is provided to apply EAC, or one's detector output, to the RCA result. These HAs can be implemented using Peres gates (PG), as shown in Figure 3(a). The quantum cost of the proposed adder in Figure 3(a) is 15n. The quantum depth of the proposed modulo 2^n*-1* adder, which was computed using the RcViewer tool [23], is $(6n + (3 \times \lceil \log_2 n \rceil) + 6)\Delta$. The total constant inputs and garbage outputs of the proposed modulo 2^n-1 adder are 3n and $(5n + 1 - 2\lfloor n/2 \rfloor)$, respectively. Figure 3(b) represents the proposed reversible modulo 2^n+1 adder. The Fredkin and FG gates are used in the middle level to perform the OR and NOT operations. The total quantum cost, quantum depth, garbage outputs, and constant inputs of the proposed modulo 2^n*+1* adder with a diminished-one number system are *10n+6, (6n+8)Δ, 3n+3,* and *2n+2*, respectively.

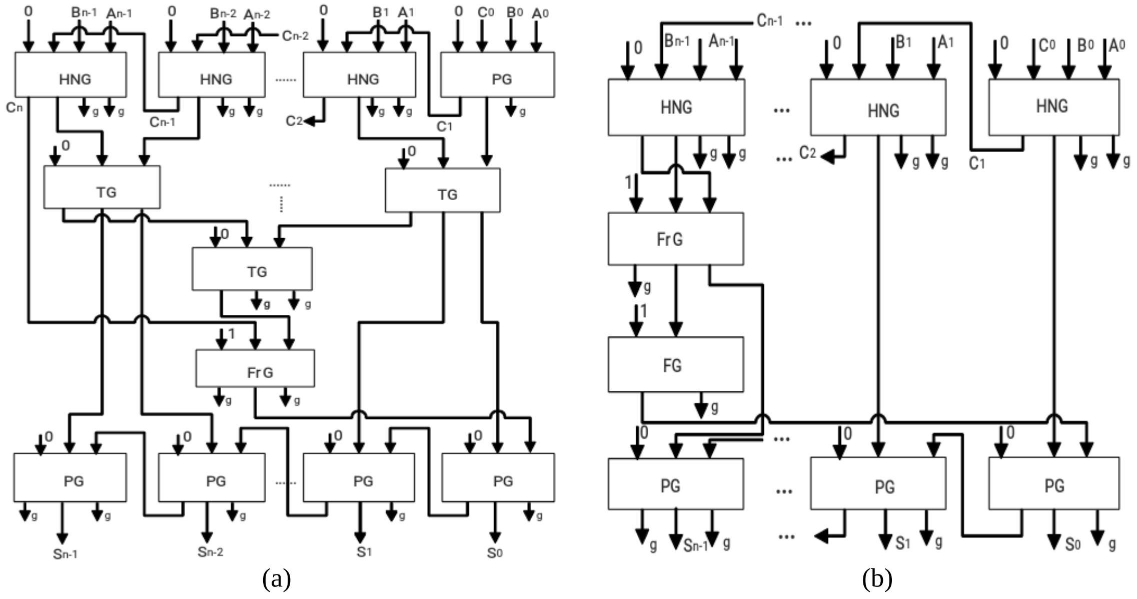


(a)

(b)

(c)

Figure 2. The proposed reversible modular circuits (a) regular CSA and CSA with EAC when the dash-dash connection is introduced, (b) CSA with CEAC, and (c) regular RCA (g means garbage output)



(a)

(b)

Figure 3. The proposed reversible modular adders (a) modulo *2^n-1* adder with one representation of zero, and (b) modulo *2^n+1* adder with diminished-one number system

3.1.2. The design proposed to reduce the number of constant inputs in reversible modulo 2^n adder

As shown in Figure 2(a), in reversible modulo *2^n* adder (regular CSA), the number of constant inputs is equal to the number of bits. However, the carry output is ignored, and using the new design that is shown in Figure 4, the number of constant inputs can be reduced. However, in the new design, the amount of quantum depth will increase slightly.

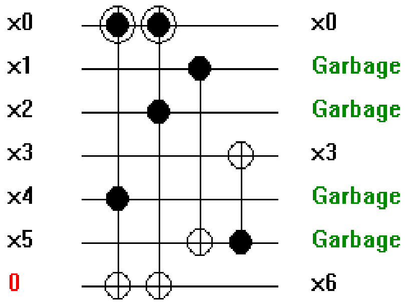


Figure 4. The proposed reversible CSA modulo 2^n for n=2

3.2. The proposed reversible modular multipliers

The multiplier structure has three main parts: i) generation of the partial products (PPis), using bitwise AND/OR gates; ii) Pp_is' summation using CSAs; and iii) a final two-operand CPA. This section first proposes reversible multipliers for the moduli 2^n-1, 2^n, and 2^n+1. the basic parameters of the reversible circuits are then determined. Also, a method to reduce the delay in reversible modular multipliers is then proposed.

3.2.1. Reversible modulo 2^n-1 multiplier

For designing a modulo 2^n-1 multiplier (6), the PP_is should be separately calculated using AND gates. Here, to implement the AND logic operation and the RAM gates for duplicating signals and interconnecting gates, the Peres gate is used. For instance, the PP_is generation for modulo 2^4-1 multiplication is depicted in Figure 5(a) where the general value n relies on $n\times n$ AND operations, which can be realized using $n\times n$ PG and n RAM gates.

The modulo 2^n-1 multiplier requires an n-input RAM gate, which consists of n-1 FGs, leading to a quantum cost of $2(n\times(n-1))$. Therefore, the total quantum cost required for PP_is' generation is $4(n\times n)+2(n\times(n-1))$. Each PG and RAM gate has 1 and n-1 constant inputs, respectively. Also, RAM does not require garbage output while PG needs two. Therefore, the total number of garbage outputs and constant inputs for the PP_i generation unit are $2(n\times n)$ and $(n\times n)+2(n\times(n-1))$, respectively. The RAM gates in the PP_is' unit are operating in parallel, and according to the RcViewer [23] simulation, the total quantum depth for the PP_is generation unit of modulo 2^n-1 multiplier will be $(RAM_{quantum\ depth}+4)\Delta$.

PP_is are added using CSAs with EAC, followed by a modulo 2^n-1 adder with one representation of zero, as shown in Figure 5(b) for $n=4$. The CSA with EAC structure of Figure 2(a) as well as the proposed modulo 2^n-1 adder in Figure 4 were used to obtain the proposed reversible modulo 2^n-1 multiplier in Figure 5(b). In general, $(n-2)\times n$ HNGs are required for the CSAs of the multiplier. Moreover, $(n-1)$ HNGs, $(n-1)$ TGs, one FrG, and $(n+1)$ PGs are required for the RCA with EAC, which is applied to design the reversible modulo 2^n-1 multiplier. The total quantum cost for the CSA and RCA of the proposed multiplier can be calculated as in (12).

$$\begin{aligned} QC_{CSA-RCA} &= 6\times[(n-2)\times n+(n-1)]+4\times[(n+1)]+5\times[(n-1)]+5 \\ &= 3n(2n+1)-2 \end{aligned} \tag{12}$$

For modulo 2^n-1 multiplication, (n-2) levels of CSA-EAC are required [20]. The quantum depth for the CSAs of the modulo 2^n-1 multiplier is $(5+[(n-3)\times3])\Delta$. The RCA in the third row consists of a PG plus serially connected HNGs, and $\lceil\log_2 n\rceil$ levels of TGs are required.

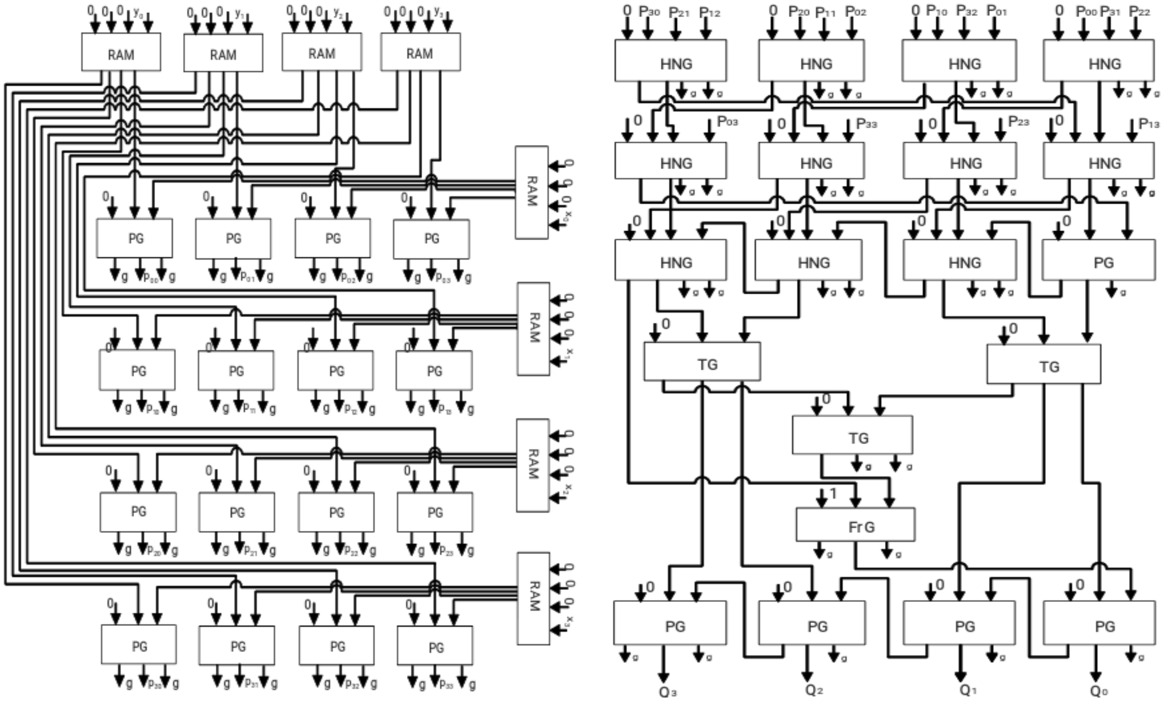
Finally, 1 FrG and n PGs of the multiplier increase the depth by 3Δ and 3nΔ, respectively. Consequently, the total quantum depth of the CSA-RCA part of the proposed reversible modulo 2^n-1 multiplier is:

$$\begin{aligned} quantum\ depth_{CSA-RCA} &= \big(5+[(n-3)\times3]+4n+4+\big(3\times(\lceil\log_2 n\rceil-1)\big)+3+3n\big)\Delta \\ &= (10n+3\times(\lceil\log_2 n\rceil))\Delta \end{aligned} \tag{13}$$

There are *(n-2)*$\times n$ HNGs in the CSA-EAC part of the modulo 2^n*-1* multiplier. The RCA-EAC part includes a PG with one constant input and one garbage output, (n-1) HNGs, (n-1) TGs, (n-1) constant inputs, and $2\times\left(n-1-\left\lfloor\frac{n}{2}\right\rfloor\right)$ garbage outputs. The FrG has one constant input and two garbage outputs. Finally, the last level *n* PGs rely on *n* constant inputs and *n* garbage outputs. Therefore, the total constant inputs and garbage outputs for the CSA-RCA part of the multiplier are:

$$constant\ input_{CSA-RCA} = [(n-2)\times n] + 1 + (n-1) + (n-1) + 1 + n \qquad (14)$$

$$garbage\ output_{CSA-RCA} = 2\times[(n-2)\times n] + 1 + 2\times(n-1) + 2\times\left(n-1-\left\lfloor\frac{n}{2}\right\rfloor\right) + 2 + n + 1 \qquad (15)$$



(a) (b)

Figure 5. The proposed reversible modulo 2^n*-1* multiplier for *n=4* (a) the partial product generation unit of the proposed reversible multiplier and (b) the proposed reversible multiplier

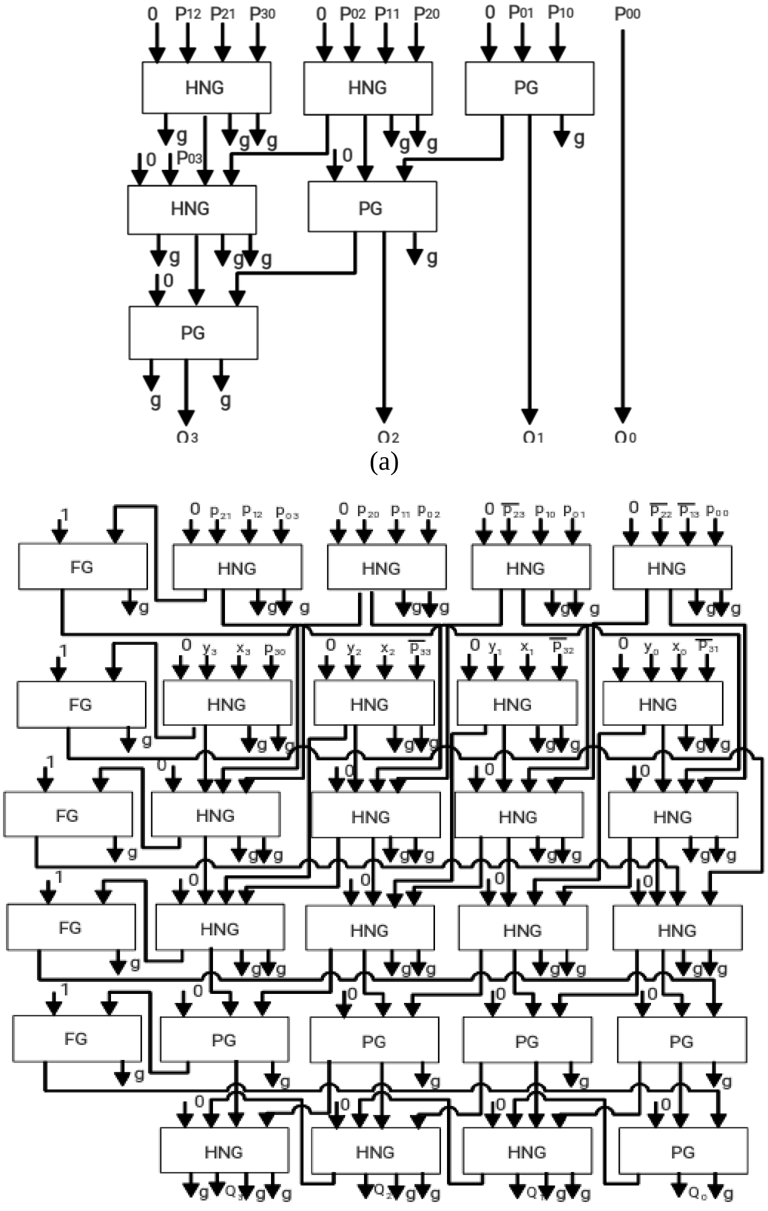
3.2.2. Reversible modulo 2^n multiplier

The modulo 2^n performs a regular $n\times n$ multiplication and only picks up the n LSBs of the result. An efficient reversible modulo 2^n multiplier is obtained by removing the components of the regular reversible $n\times n$ binary multiplier. It produces the necessary partial products, resulting in a significant area reduction. $\sum_{i=1}^{n} i$ Peres gates and *2(n-1)* RAM gates are required to implement PP$_i$s' generation unit of the proposed reversible modulo 2^n multiplier. However, the quantum cost of RAM gates depends on the number of their inputs. Therefore, the total quantum cost of PP$_i$'s generation unit of the proposed modulo 2^n multiplier is $2(\sum_{i=1}^{n-1} i) + 4(\sum_{i=1}^{n} i)$. Besides, its quantum depth estimated by the RcViewer is (quantum depth$_{ram}$+4)Δ. The total number of constant inputs and garbage outputs of the PP$_i$'s generation unit of the proposed modulo 2^n multiplier is $\frac{n(3n-1)}{2}$ and $2(\sum_{i=1}^{n-1} i)$, respectively. The last stages of the hardware architecture of the proposed modulo 2^n multiplier for *n=4* are depicted in Figure 6(a). There are (n-1) PGs and $\sum_{i=1}^{n-2} i$ HNGs in the CSA structure of the proposed modulo 2^n multiplier. Therefore, the total quantum cost of this part of the multiplier is $3n^2-5n+2$. Also, the critical path includes *n-2* HNGs followed by a PG, as shown in Figure 6(a) for *n=4*. Therefore, the total quantum depth will be (5+(3×(n-3))+3)Δ. Considering that each PG and HNG possesses one constant input, the total number of constant inputs is $\sum_{i=1}^{n-2} i + (n-1)$, and the total number of garbage outputs is $2\times(\sum_{i=1}^{n-2} i) + (n-2) + n$.

3.2.3. Reversible modulo 2^n+1 multiplier

The modulo 2^n+1 multiplication relies on PP$_i$'s generation unit, CSAs with CEAC to compress the partial products, and a final modulo 2^n+1 adder, as shown in Figure 6(b). The Peres and FGs are applied to realize the AND, and NOT logic operations, respectively. The PP$_i$s' generation unit in reversible modulo 2^n+1 multiplier is similar to the PP$_i$'s generation unit in reversible modulo 2^n-1 multiplier, except that the FGs are applied to realize NOT logic operations. ($n\times n$) Peres gates, $2n$ RAM gates with n inputs, and $\sum_{i=1}^{n-1} i$ FGs are required to obtain the PP$_i$ unit of the proposed reversible modulo 2^n+1 multiplier. The total quantum cost of the PP$_i$s' unit of the modulo 2^n+1 multiplier is:

$$quantumcost_{PP_i} = 4 \times (n^2 + 2n + 1) + 1 \times \left(\sum_{i=1}^{n-1} i\right) + 2((n-1) \times n) \quad (16)$$



(a)

(b)

Figure 6. The proposed reversible modular multipliers for n=4 (a) the last stages of the modulo 2^n multiplier, and (b) the modulo 2^n+1 multiplier

All the required PGs and FGs in the PP$_i$s unit have just one constant input. Each RAM gate has $n-1$ constant inputs. So, the total number of constant inputs for the PP$_i$ unit is:

$$constantinputs_{PP_i} = (n^2 + 2n + 1) + (\sum_{i=1}^{n-1} i) + 2((n-1) \times n) \quad (17)$$

And the total garbage outputs is:

$$garbageoutputs_{PP_i} = 2n^2 - 2n + \sum_{i=1}^{n-1} i \quad (18)$$

Finally, according to RcViewer simulation, the quantum depth includes *2n* RAM gates with quantum *depth$_{ram}$*. It consists of *n-1* FGs. The total quantum depth for the PP$_i$'s unit of the proposed reversible modulo *2^n+1* adder is equal to *(quantum depth$_{ram}$+4+1)Δ*. Further, according to (8), partial products should be added using CSAs with CEAC followed by a modulo *2^n+1* adder. The CSA-RCA unit of the suggested reversible modulo *2^n+1* multiplier (for *n=4*) is depicted in Figure 6(b). The CSAs of the suggested reversible modulo *2^n+1* multiplier requires $n\times n$*+(n-1)* HNGs, (*n+1*) PGs, and (*n+1*) FGs to realize the FAs, HAs, and NOT gates, respectively. Therefore, regardless of its PP$_i$ generation, the total quantum cost of the proposed modulo *2^n+1* multiplier can be calculated:

$$quantumcost_{CSA-RCA} = 6 \times [n \times n + (n-1)] + 4 \times [(n+1)] + (n+1) \quad (19)$$

The PG in the first and last level of the final modulo adder adds 3Δ, and each (*n-1*) HNG increases the depth by 4Δ. Therefore, the total quantum depth of the suggested modulo *2^n+1* adder will be *(quantum depth$_{CSA-RCA}$+2)Δ*. All RGs in the CSA-RCA part of the suggested modulo *2^n+1* multiplier have one constant input. All FGs and PGs have one garbage output, and each HNG has two garbage outputs. Therefore, the total constant inputs and garbage outputs are:

$$constant\ inputs_{CSA-RCA} = \left((n+1) \times n\right) + n + n + 2 \quad (20)$$

$$garbage\ outputs_{CSA-RCA} = 2 \times \left((n \times n) + (n-1) \times n + n + n + 1 + 1 + 1\right) \quad (21)$$

The design proposed to improve the delay in reversible modular multipliers.

To solve the delay problem in the proposed reversible modular multiplication, a new RAM gate is used. The RAM gate is useful for PP$_i$s' generation unit as a copying circuit that consists of several FGs. If RAM has N inputs, its depth will be N-1 because the depth of each FG is 1. In Figure 7, using the FG and changing the design method, a new design for reversible RAM has been introduced. It is seen that the quantum depth value in the proposed design is lesser than that in the existing approaches. A comparison of the existing and proposed RAM depth is given in Table 1.

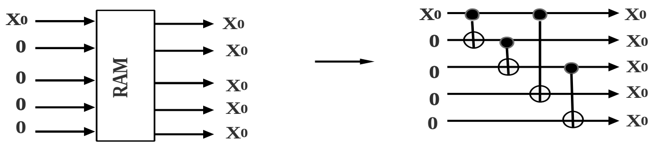


Figure 7. Using a 5-input new RAM gate to achieve 5 copies of X0

Table 1. Comparison of the existing and proposed reversible RAM designs

N. of Input		2	4	7	12	20	33	54	88	143
Quantum depth (Delay)	Ref. [13]	1	3	6	11	19	32	53	87	142
	Proposed	1	2	3	4	5	6	7	8	9

3.3. The proposed reversible forward and reverse converters

The following section presents the reversible design of the forward and reverse converters for the moduli set {2^n-1, 2^{n+k}, 2^n+1}. They are efficiently designed using the proposed reversible modular adders since mainly modular addition is required in the core of the converters. Also, some circuit parameters, including the quantum cost, quantum depth, constant input, and garbage output are calculated.

3.3.1. Forward converters

Figure 8(a) shows the hardware structure of the forward converter for the moduli set {2^n-1, 2^{n+k}, 2^n+1} based on (9), consisting of CSAs and a modulo *2^n±1* adder The reversible CSAs as shown in

Figures 2(a) and 2(b) followed by a reversible modular adder are used to implement (9). A simplified version of the suggested reversible modulo *2^n+1* adder has been used in Figure 8(a) since based on (9), the operands have *n* bits. To improve performance, the reversible modulo *2^n-1* with a double representation of zero [12] has been used as shown in Figure 4. The modulo *2^n-1* with a double representation of zero does not require a series of AND gates for the detection of 1s. However, for the remaining RNS components, the modulo *2^n-1* with a single representation of zero should be used. There are two CSAs with EAC as shown in Figure 2(a) in the modulo *2^n-1* channel, i.e. *x_1*, of the forward converter as shown in Figure 8(a). Therefore, the total quantum cost of the forward converter is equal to *22n-2*. Besides, all the required RGs for the x_1 computation have just one constant input. The modulo adder consists of *n+1* PGs and *n-1* HNGs. Therefore, *4n* and *7n-1* constant inputs and garbage outputs are applied, respectively. The quantum depth of the *2^n-1* channel of the forward converter is *(7n+9)Δ*. Note that the most important path of the forward converter is set by the modulo *2^n+1* channel. Therefore, the quantum depth of the x_3 circuit defines the total quantum depth of the forward converter. The modulo *2^n+1* channel, i.e. x_3, of the forward converter as shown in Figure 8(a), consists of three CSAs with CEAC as shown in Figure 2(b) where each CSA is designed with n HNGs and one FG.

Besides, the final modulo adder includes a PG and *n-1* HNGs for the first level and an FG with n PGs for the second level. Therefore, the total quantum cost for the modulo *2^n+1* channel of the forward converter is *28n+2*. Moreover, *5n+4* constant inputs and *9n+3* garbage outputs are required for the RGs in the modulo *2^n+1* channel of the forward converter. Finally, the total quantum depth of the modulo *2^n+1* channel for the forward converter, which defines its critical path, is *(7n+15)Δ*.

3.3.2. Reverse converters

The proposed reversible reverse converter for the moduli set {2^n-1, 2^{n+k}, 2^n+1} is shown in Figure 8(b). The proposed reversible CSAs with EAC as shown in Figure 2(a) and modulo 2^n-1 adder as shown in Figure 4 for achieving the reversible reverse converter have been used. Due to the use of 2n-bit operands, each CSA requires 2n HNGs. Besides, the modulo adder requires a PG followed by (2n-1) HNGs, (2n-1) TGs, a FrG and 2(n) PGs.

As a result, the total quantum cost of the reverse converter is:

$$quantum\ cost_{Reverse} = 6(2n + 2n) \times 6(2n - 1) + 9 + 4(2n) = 54n - 2 \quad (22)$$

All the HNGs in the proposed reversible reverse converter have one constant input and two garbage outputs. Besides, all PGs have one constant input and one garbage output, except for the last PG, which has two garbage outputs. Also, the TG has one constant input with two garbage outputs, except for the first-level TGs, which have no garbage outputs. Therefore, the total constant inputs and garbage outputs of the reverse converter are:

$$constant\ inputs_{Reverse} = 4n + 1 + 2n - 1 + 2n - 1 + 1 + 2n = 10n \quad (23)$$

$$garbage\ outputs_{Reverse} = 2(4n) + 1 + 2(2n - 1) + 2(2n - 1 - n) + 2 + 2n + 1 = 16n \quad (24)$$

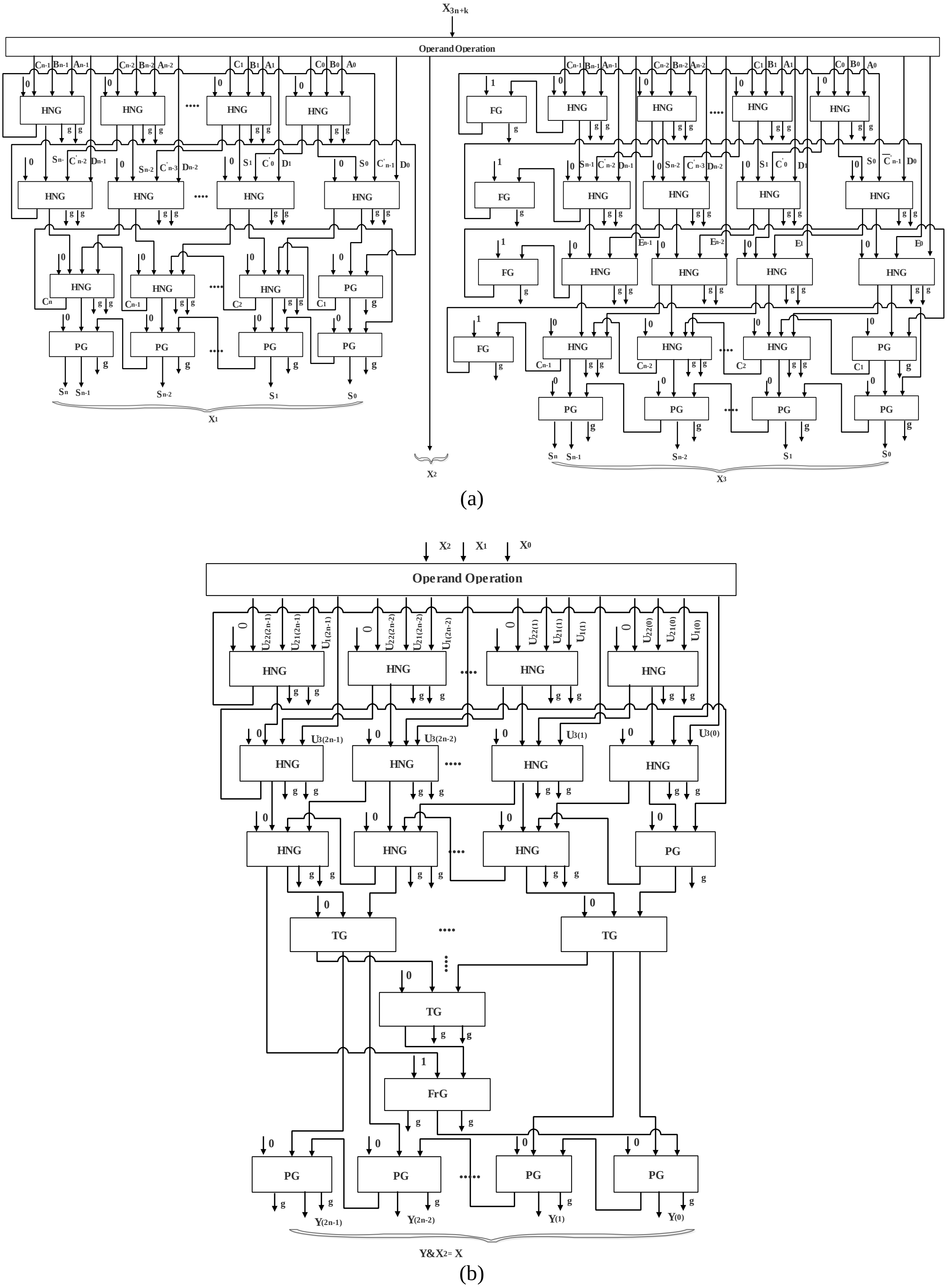
The first and second levels of HNGs in the CSAs add 9Δ to the circuit depth. Besides, the first PG and each of the remaining (2n-1) HNGs add 4Δ to the quantum depth of the reverse converter. Moreover, we have $log_2 2\,n$, TG levels, and the first level of the TG, which increases the depth by 4Δ, and the other levels of TG increase the depth by 3Δ. Each of the next FrGs and the last level PGs add 3Δ to the depth. As a result, the total quantum depth of the reverse converter is (25).

$$quantum\ depth_{Reverse} = (12 + 4 \times (2n - 1) + 4 + 3 \times (\lceil \log_2 2n \rceil - 1) + 3 + 3(2n))\Delta \quad (25)$$

In this section, each component of the suggested RNS reversible system has been evaluated in terms of the quantum cost, quantum depth, and the number of constant inputs and garbage outputs.

3.4. Performance evaluation

This study is the first case study in the literature reporting the implementation of all RNS components using reversible circuits. The performance of the proposed modular reversible circuits is compared with. the performance of conventional binary reversible circuits. Consecutive multiplications are selected due to their importance in several applications. Although consecutive multiplications are modular, i.e., the bit width of the successive operands is constant, the worst-case scenario has been considered, where the bit width doubles once a multiplication is performed. 4-bit operands have been considered for performing consecutive multiplications in both RNS form and the conventional binary representation, as shown in Table 2.



(a)

(b)

Figure 8. The proposed reversible converter for the moduli set $\{2^n-1, 2^{n+k}, 2^n+1\}$ (a) forward converter and (b) reverse converter

Table 2. Reversible implementation of multiple multiplications: regular vs. RNS

Number of Multiplications	Quantum Cost		Quantum Delay		Quantum Cost Improvement	Quantum Delay Improvement	Required Regular Multiplier Size (output bits)	Required RNS Dynamic Range (bits)	
	Regular	RNS	Regular	RNS				*n*	*Total*
2	140	754	41	142	-81.4%	-71.1%	8	3	9
3	428	1127	106	214	-62%	-50.4%	12	4	12
4	864	2056	195	327	-57.9%	-40.3%	16	6	18
5	1448	2611	308	427	-44.5%	-27.8%	20	7	21
6	2180	3229	445	526	-32.4%	-15.3%	24	8	24
7	3060	4646	606	672	-34.1%	-9.8%	28	10	30
8	4088	5446	791	810	-24.9%	-2.3%	32	11	33
9	5264	6307	1000	930	-16.53%	+7%	36	12	36
10	6588	8212	1233	1125	-19.77%	+8.7%	40	14	42
11	8060	9212	1490	1262	-12.5%	+15.3%	44	15	45
12	9680	8212	1771	1291	+15.1%	+27.1%	48	16	48

First, the method of [24] has been considered for multiplying 4-bit operands in the conventional binary number representation. On the other hand, multiplications in the RNS domain have been done using the first translation of 4-bit weighted operands to residue representation by two parallel forward converters. Then, modular multiplications were applied using the residues. Note that the forward converter is considered only once for calculating the total delay since, after conversion of the first two operands to RNS, the remaining numbers' conversions will be done in parallel by modular multiplication of previous operands. Similar to the forward converter, the reverse converter was applied only once at the end to produce the weighted representation of the last product. Note that in this case study, the value of k in the moduli set $\{2^n-1, 2^{n+k}, 2^n+1\}$ is considered zero, and the value of n is selected in such a way that the dynamic range *3n* is enough for the required bit width. For two regular weighted operands with p and q-bit size, the multiplication result will have $p+q$ bits. Therefore, in consecutive multiplications, the first two 4-bit operands yield 8-bit products. Then, another 4-bit operand is multiplied by the previous output result, that is 8 bits, leading to the output result with 12 bits, and this process continues. It can be seen from Table 2 that, as expected, for a small number of operations, the RNS structure cannot lead to improvement mainly due to the overhead of converters. However, by increasing the number of operations, the RNS performs better since the overhead of the required converters is not considered while performing internal modular operations. As indicated in Table 2, after 9 multiplication operations, RNS results in a 7% reduction of the delay, and this improvement continues up to 27% for 12 operations. Also, the performance of the proposed modular reversible circuits in a case-study application was compared with that of the conventional binary reversible circuits. The dot-product operation is selected due to its importance in the convolution operation in a variety of operations from digital signal processing to deep convolutional neural networks. The general dot-product formula of (26) for 20 operands, i.e., *m=10*, and for 18 different operands' bit-width is considered.

$$y(n) = \sum_{i=1}^{m} A_i B_i \tag{26}$$

3.4.1. Reversible regular dot-product calculation

To calculate $A_1B_1+A_2B_2+...$, some multiplications and additions are required. The regular reversible multiplier of [25] is considered to perform the required multiplications. A carry-save adder is used to perform carry-save additions of the multiplication results, and a regular reversible ripple carry adder is used to add the redundant summation outputs of the carry-save adder to achieve the result. The circuit's parameters for multiplying two n-bit numbers, according to the method used in [25], are (27) to (30).

$$quantum\ cost_{multiply} = 3n + (16 \times n) + 16 + 4(n-1) \tag{27}$$

$$quantum\ depth_{multiply} = \big(3 + (2n+2) + \big(6 \times (8 + 3(n-4))\big) + 16\big)\Delta \tag{28}$$

$$constant\ cost_{multiply} = 3n + 4n + 4 + 8 + 3(n-4) \tag{29}$$

$$garbage\ outputs_{multiply} = n \times (n-1) + 2n + 4 + 2(8 + 3(n-4)) \tag{30}$$

3.4.2. Reversible modular dot-product calculation

Here, the proposed RNS circuits can be used to perform the dot-product operation. First, each number should be converted into residues using the forward converter. The forward converter possesses a parallel structure for computing the residues of an operand. Modular arithmetic channels then perform modulo multiplication in residues of two corresponding operands. This operation will be repeated two times more for other operands. Finally, the results of multiplications should be added. This is possible to be done by the use of a modular CSA followed by a modular adder. The result will be converted to its regular binary form using the reverse converter. A comparison between the regular binary and RNS implementation of reversible dot-product calculation for different operand widths is presented in Table 3. The value of n in the moduli set {2^n-1, 2^{n+k}, 2^n+1} is selected in such a way that the relation $3n+k\geq 2q+2$ holds where $3n+k$ is the dynamic range. Note that for q-bit operands, each multiplication result will have 2q bits.

Besides, adding three 2k-bit operands yields the *(2q+2)*-bit final dot products. Thus, the dynamic range should be equal to or greater than *2q+2*. It can be seen from Table 3 that the proposed RNS circuit results in an improvement in *N=15*, and at this stage, we will have a 10.7% improvement in delay. RNS eliminates the need for large multipliers and adders substituting them with small arithmetic circuits working in parallel.

Table 3. Reversible implementation of N-Bit convolution: regular vs. RNS

Bits	Quantum Delay		Quantum Delay Improvement	Required Regular Convolution Size (output bits)	Required RNS Dynamic Range (bits)	
	Regular	RNS			*N*	*Total*
3	530	1032	-48.6%	11	4	12
4	736	1267	-41.9%	13	5	15
5	942	1267	-25.6%	15	5	15
6	1148	1512	-24%	17	6	18
7	1354	1748	-22.5%	19	7	21
8	1560	1748	-10.7%	21	7	21
9	1766	1952	-9.5%	23	8	24
10	1972	2172	-9.2%	25	9	27
11	2178	2172	+0.27%	27	9	27
12	2384	2326	+2.4%	29	10	30
13	2590	2504	+3.3%	31	11	33
14	2796	2504	+10.4%	33	11	33
15	3002	2678	+10.7%	35	12	36

4. CONCLUSION

In this paper, the reversible design of modular adders and multipliers is presented, which is a vital element in computation. As shown, adopting this novel design in reversible modular adders improves the number of constant inputs in the existing circuits. Also, using a new ram gate is likely to reduce the delay in the production of reversible circuits. Reversible forward and reverse converters for the 3-moduli set {2^n-1, 2^{n+k}, 2^n+1} have also been designed. Finally, results showed that the proposed design of modular reversible circuits reduced some circuit parameters, including latency and cost. In the future, reversible parallel prefix multipliers are suggested to be used to make circuits under optimized conditions in a way to be cost-effective in terms of gate cost, delay, garbage, and quantum cost. The parallel prefix multiplier could be used for cases where speed is more important than cost. The disadvantage of using such a method, however, will be its high cost, and in order to solve this problem, a hybrid circuit may be used. The hybrid parallel prefix circuit will be improved further if the parallel prefix Kogge–Stone is implemented because this method has a significantly lower depth compared to other prefix methods. The only defect of the parallel prefix Kogge–Stone circuit is the high cost of the circuit that can be significantly reduced using this hybrid model.

REFERENCES

[1] T. M. Conte, E. P. DeBenedictis, P. A. Gargini, and E. Track, “Rebooting computing: the road ahead,” *Computer*, vol. 50, no. 1, pp. 20–29, Jan. 2017, doi: 10.1109/MC.2017.8.

[2] B. Moons, D. Bankman, and M. Verhelst, *Embedded deep learning*. Cham: Springer International Publishing, 2019.

[3] M. Alioto, Ed., *Enabling the internet of things*. Cham: Springer International Publishing, 2017.

[4] P. V. A. Mohan, “Residue number systems: Theory and applications,” *Basel: Birghauser, Mathematics*, 2016.

[5] T. F.Tay and C.-H. Chang, “A non-iterative multiple residue digit error detection and correction algorithm in RRNS,” *IEEE Transactions on Computers*, vol. 65, no. 2, pp. 396–408, Feb. 2016, doi: 10.1109/TC.2015.2435773.

[6] X. Zheng, B. Wang, C. Zhou, X. Wei, and Q. Zhang, “Parallel DNA arithmetic operation with one error detection based on 3-moduli set,” *IEEE Transactions on NanoBioscience*, vol. 15, no. 5, pp. 499–507, Jul. 2016, doi: 10.1109/TNB.2016.2574359.

[7] B. Deng *et al.*, “Extending Moore’s law via computationally error-tolerant computing,” *ACM Transactions on Architecture and Code Optimization*, vol. 15, no. 1, pp. 1–27, Apr. 2018, doi: 10.1145/3177837.

[8] J. Cong, Z. Fang, M. Huang, P. Wei, D. Wu, and C. H. Yu, “Customizable computing-from single chip to datacenters,” *Proceedings of the IEEE*, vol. 107, no. 1, pp. 185–203, Jan. 2019, doi: 10.1109/JPROC.2018.2876372.

[9] E. P. De Benedictis, J. K. Mee, and M. P. Frank, “The opportunities and controversies of reversible computing,” *Computer*, vol. 50, no. 6, pp. 76–80, 2017, doi: 10.1109/MC.2017.177.

[10] S. M. R. Taha, *Reversible logic synthesis methodologies with application to quantum computing*, vol. 37. Cham: Springer International Publishing, 2016.

[11] A. Zulehner and R. Wille, “One-pass design of reversible circuits: combining embedding and synthesis for reversible logic,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2017, doi: 10.1109/TCAD.2017.2729468.

[12] A. S. Molahosseini, A. Asadpoor, A. A. E. Zarandi, and L. Sousa, “Towards efficient modular adders based on reversible circuits,” in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–5, doi: 10.1109/ISCAS.2018.8351702.

[13] H. G. Rangaraju, A. B. Suresh, and K. N. Muralidhara, “Design of efficient reversible multiplier,” in *Advances in Computing and Information Technology*, Springer Berlin Heidelberg, 2013, pp. 571–579.

[14] H. A. Mousavi, P. Keshavarzian, and A. S. Molahosseini, “A novel fast and small XOR-base full-adder in quantum-dot cellular automata,” *Applied Nanoscience*, vol. 10, no. 11, pp. 4037–4048, Nov. 2020, doi: 10.1007/s13204-020-01511-x.

[15] T. Krishnan, S. Saravanan, P. Anguraj, and A. S. Pillai, “Design and implementation of area efficient EAIC modulo adder,” *Materials Today: Proceedings*, vol. 33, pp. 3751–3756, 2020, doi: 10.1016/j.matpr.2020.06.172.

[16] A. S. Molahosseini, K. Navi, C. Dadkhah, O. Kavehei, and S. Timarchi, “Efficient reverse converter designs for the new 4-noduli sets {2n–1, 2n, 2n+ 1, 22n+ 1–1} and {2n–1, 2n+ 1, 22n, 22n+ 1} based on new CRTs,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 4, pp. 823–835, Apr. 2010, doi: 10.1109/TCSI.2009.2026681.

[17] K. Isupov, “Using floating-point intervals for non-modular computations in residue number system,” *IEEE Access*, vol. 8, pp. 58603–58619, 2020, doi: 10.1109/ACCESS.2020.2982365.

[18] R. Chaves and L. Sousa, “Improving residue number system multiplication with more balanced moduli sets and enhanced modular arithmetic structures,” *IET Computers and Digital Techniques*, vol. 1, no. 5, 2007, doi: 10.1049/iet-cdt:20060059.

[19] J.-L. Beuchat, “Some modular adders and multipliers for field programmable gate arrays,” in *Proceedings International Parallel and Distributed Processing Symposium*, 2003, pp. 1–8, doi: 10.1109/IPDPS.2003.1213353.

[20] T.-B. Juang, C.-T. Kuo, G.-L. Wu, and J.-H. Huang, “Multifunction RNS modulo 2n±1 multipliers,” *Journal of Circuits, Systems and Computers*, vol. 21, no. 4, Jun. 2012, doi: 10.1142/S0218126612500272.

[21] A. Peres, “Reversible logic and quantum computers,” *Physical Review A*, vol. 32, no. 6, pp. 3266–3276, Dec. 1985, doi: 10.1103/PhysRevA.32.3266.

[22] E. Fredkin and T. Toffoli, “Conservative logic,” *International Journal of Theoretical Physics*, vol. 21, no. 3–4, pp. 219–253, Apr. 1982, doi: 10.1007/BF01857727.

[23] D. V Zakablukov, “Application of permutation group theory in reversible logic synthesis,” in *Reversible Computation*, Springer International Publishing, 2016, pp. 223–238.

[24] I. Krstic, N. Stamenkovic, M. Petrovic, and V. Stojanovic, “Binary to RNS encoder with modulo 2n+ 1 channel in diminished-1 number system,” *International Journal of Computational Engineering and Management (IJCEM)*, vol. 17, no. 4, pp. 1–9, 2014.

[25] M. Haghparast, M. Mohammadi, K. Navi, and M. Eshghi, “Optimized reversible multiplier circuit,” *Journal of Circuits, Systems and Computers*, vol. 18, no. 2, pp. 311–323, Apr. 2009, doi: 10.1142/S0218126609005083.

BIOGRAPHIES OF AUTHORS

Ailin Asadpour has gained the Ph.D. in computer engineering of Islamic Azad University, Kerman, Iran. Her research interests include VLSI design and computer arithmetic. She can be contacted at ailinasadpoor@iauk.ac.ir.

Amir Sabbagh Molahosseini received Ph.D. degrees from Science and Research Branch of Islamic Azad University, Tehran, Iran, in computer engineering. He is an assistant professor in Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman, Iran. His research interests include VLSI design and computer arithmetic with emphasis on residue number system. He can be contacted at sabbagh@iauk.ac.ir.

Azadeh Alsadat Emrani Zarandi received her Ph.D. in computer engineering at Science and Research Branch, Islamic Azad University, Tehran, Iran. Her main research interests are residue number systems and wireless sensor networks. She can be contacted at a.emrani@uk.ac.ir.