IEICE TRANS. INF. & SYST., VOL.E94–D, NO.2 FEBRUARY 2011

PAPER

A General Reverse Converter Architecture with Low Complexity and High Performance

Keivan NAVI[†a)], ***Member***, **Mohammad ESMAEILDOUST**[†], ***and*** **Amir SABBAGH MOLAHOSSEINI**[††], ***Nonmembers***

SUMMARY This paper presents a general architecture for designing efficient reverse converters based on the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where $\beta < \alpha \leq 2\beta$, by using a parallel implementation of mixed-radix conversion (MRC) algorithm. The moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$ is free from modulo (2^k+1)-type which can result in an efficient arithmetic unit for residue number system (RNS). The values of α and β can be selected to provide the required dynamic range (DR) and also to adjust the desired equilibrium between moduli bit-width. The simple multiplicative inverses of the proposed moduli set and also using novel techniques to simplify conversion equations lead to a low-complexity and high-performance general reverse converter architecture that can be used to support different DRs. Moreover, due to the current importance of the 5n-bit DR moduli sets, we also introduced the moduli set $\{2^{2n}, 2^{2n+1}\text{-}1, 2^{n}\text{-}1\}$ which is a special case of the general set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where α=2n and β=n. The converter for this special set is derived from the presented general architecture with higher speed than the fastest state-of-the-art reverse converter which has been designed for the 5n-bit DR moduli set $\{2^{2n}, 2^{2n+1}\text{-}1, 2^{n}\text{-}1\}$. Furthermore, theoretical and FPGA implementation results show that the proposed reverse converter for moduli set $\{2^{2n}, 2^{2n+1}\text{-}1, 2^{n}\text{-}1\}$ results in considerable improvement in conversion delay with less hardware requirements compared to other works with similar DR.

key words: *residue arithmetic, reverse converter, residue number system (RNS), VLSI architecture*

1. Introduction

One of the most effective ways to achieve parallelism on arithmetic level in VLSI design is using residue number system (RNS) [1]. Because, RNS has an inherent property to perform addition, subtraction and multiplication without carry-propagation between residue digits, this makes RNS a high-performance alternative number system that can lead to reducing power dissipation and considerable speed-up in digital computing systems [2], [3]. The most important applications of RNS have been reported in the digital signal processing (DSP) area including FIR filters, convolutions, DFT and FFT computations [4]–[7]. Furthermore, the advantages of using redundant RNS to provide easy error detection and correction are well documented [8], [9]. However, the difficulties which have existed in implementation of non-modular RNS operations, as well as the overhead incurred by forward and reverse converters, were preventing the usage of RNS in general-purpose processors. But, the recent achievements to perform difficult RNS operations such as sign detection [10], magnitude comparison [11] and scaling [12] promote the increase in applicability of RNS in general-purpose computing systems. The most imperative issue to design efficient RNS systems is appropriate selection of moduli set since the performance of residue arithmetic channels as well as the complexity of forward and reverse converters depends mainly on the form and the number of moduli [13]. The moduli set $\{2^n, 2^n\text{-}1, 2^n\text{+}1\}$ has attracted a large amount of research for many decades primarily because of simple and balanced moduli. However, its dynamic range (DR) is not suitable for current high-performance DSP applications. To overcome this problem, i.e., having large DR together with the advantages of popular set $\{2^n, 2^n\text{-}1, 2^n\text{+}1\}$, Hariri et al. [14] proposed the 5n-bit DR moduli set $\{2^n, 2^{2n}\text{-}1, 2^{2n}\text{+}1\}$ with its high-speed and low-cost reverse converter. Moreover, the moduli set $\{2^{\alpha}, 2^{\beta}\text{-}1, 2^{\beta}\text{+}1\}$, where $\alpha < \beta$, has been introduced by Molahosseini et al. [15] to provide a large dynamic range RNS systems. The reverse converter of [15] relies on a simple and efficient architecture; however, with constraint $\alpha < \beta$, the DR will be concentrated on low-performance moduli 2^{β}+1 and this leads to an increase in the total delay of RNS arithmetic unit. Chavez and Sousa [16] suggested the moduli set $\{2^{\alpha}, 2^{\beta}\text{-}1, 2^{\beta}\text{+}1\}$, where $\alpha > \beta$. They have tried to decrease the inefficiency of modulo 2^{β}+1 by concentrating DR to efficient modulo 2^{α}; at the expense of a lower-performance reverse converter than [15]. The moduli sets $\{2^{n-1}\text{-}1, 2^n\text{-}1, 2^n\}$ [17] and $\{2^n\text{-}1, 2^n, 2^{n+1}\text{-}1\}$ [18] which are free from modulo 2^n+1 have been also introduced to provide fast RNS arithmetic unit but with more complex reverse converters than those for set $\{2^n, 2^n\text{-}1, 2^n\text{+}1\}$. Recently, the moduli sets $\{2^{2n}, 2^n\text{-}1, 2^{n\pm1}\text{-}1\}$ [19] which are the enhanced versions of these classical three-moduli sets are introduced to provide 4n-bit DR with reduced-complexity reverse converters. The demands for more parallelism than three moduli persuaded the researchers to investigate additional number of moduli. Hence, several four and five-moduli sets with different DRs have been proposed for RNS [20]–[28]. The researchers have aimed to introduce large DR moduli sets which can lead to efficient internal RNS arithmetic circuits as well as high-performance reverse converters. However, examination of published papers in this area shows that they have not completely reached this aim. In other words, when the researchers achieved fast arithmetic units, inefficient reverse converter is yield and vice versa. Although, some re-

Manuscript received April 6, 2010.

[†]The authors are with the Faculty of Electrical and Computer Engineering, Shahid Beheshti University, GC, Tehran, Iran.

[††]The author is with theMicroelectronic Laboratory of Shahid Beheshti University, GC, Tehran, Iran.

a) E-mail: navi@sbu.ac.ir

DOI: 10.1587/transinf.E94.D.264

cent works have reported better tradeoffs between performance of the RNS arithmetic unit and reverse converter, there is still a need for moduli sets which can provide high-efficiency in arithmetic unit and reverse converter.

In this paper, we propose the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where $\beta \prec \alpha \preceq 2\beta$ as a basis to provide large dynamic range RNS systems with adjustable DR and to attain fast RNS arithmetic unit as well as low-complexity reverse converters. Next, we present a general reverse converter architecture based on the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$ to achieve high-performance converters. The presented design is obtained using a parallel and adder-based implementation of the mixed-radix conversion (MRC) algorithm, resulting in a VLSI efficient architecture. Thus, the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$ can be regarded as a conversion-friendly as well as arithmetic-friendly moduli set, due to its potential to provide efficiency for all parts of RNS. Finally, we present the reverse converter for the 5n-bit DR special moduli set $\{2^{2n}, 2^{2n+1}\text{-}1, 2^{n}\text{-}1\}$, that is obtained from the general architecture. This converter results in lower conversion delay than the converter design for $\{2^{n}, 2^{2n}\text{-}1, 2^{2n}+1\}$ [14] which is the fastest known reverse converter in the area of 5n-bit DR. Moreover, the proposed converter for moduli set $\{2^{2n}, 2^{2n+1}\text{-}1, 2^{n}\text{-}1\}$ outperforms the best state-of-the-art reverse converters which have been designed for 5n-bit DR moduli sets $\{2^{n}\text{-}1, 2^{n}, 2^{n}+1, 2^{2n}+1\}$, $\{2^{n}, 2^{n}\text{-}1, 2^{n}+1, 2^{n-2(n+1)/2}+1, 2^{n+2(n+1)/2}+1\}$, $\{2^{n}\text{-}1, 2^{n}, 2^{n}+1, 2^{n-1}\text{-}1, 2^{n+1}\text{-}1\}$ and $\{2^{n}\text{-}1, 2^{n}, 2^{n}+1, 2^{2n+1}\text{-}1\}$ [25]–[28]. The remaining sections of the paper are arranged as follows. In Sect. 2, we present the proposed general reverse conversion algorithm with its hardware architecture. The derivation of the reverse converter for the moduli set $\{2^{2n}, 2^{2n+1}\text{-}1, 2^{n}\text{-}1\}$ from the general architecture, evaluation of its performance and comparison with other works are described in Sect. 3. Finally, Sect. 4 concludes the paper.

2. The General Reverse Converter Architecture

First, we apply a three-modulus version of MRC to the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where $\beta \prec \alpha \preceq 2\beta$ to obtain the conversion algorithm. Next, to reduce the hardware complexity, some mathematical properties are utilized to simplify the conversion equations. But, to begin, we provide a brief introduction to RNS and MRC followed by a theorem which shows the efficient multiplicative inverses of the proposed set.

2.1 RNS and MRC

The basis of each RNS system is a moduli set $\{P_1, P_2, \ldots, P_n\}$ which involves pairwise relatively prime numbers. The DR is defined as $M=P_1P2\ldots P_n$, so that the regular weighted number $X<M$ can be represented as $(x_1, x_2, \ldots, x_n)$ where $x_i = X \bmod P_i = |X|_{P_i}$. The following theorem confirms that the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$ can be used for RNS.

Theorem 1: The moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where $\beta \prec \alpha \preceq 2\beta$ consists of pairwise relatively prime numbers.

Proof. Consider Euclid's theorem, i.e., GCD (a, b) = GCD (b, a mod b), where the term GCD stands for the greatest common divisor of a and b. We have

$$GCD(2^{2\beta+1}-1, 2^{\alpha}) = GCD(2^{\alpha}, -1) = 1 \quad (1)$$

$$GCD(2^{\alpha}, 2^{\beta}-1) = GCD(2^{\beta}-1, 2^{\alpha-\beta}) = GCD(2^{\alpha-\beta}, -1) = 1 \quad (2)$$

$$GCD(2^{2\beta+1}-1, 2^{\beta}-1) = GCD(2^{\beta}-1, 1) = 1 \quad (3)$$

Since all the greatest common divisors of these moduli are equal to one, these numbers are pairwise relatively prime. By MRC [3], [19] the reverse conversion (i.e., translating the residue represented number into its equivalent weighted number), can be done using this equation:

$$X = Z_1 + Z_2P_1 + Z_3P_1P_2 + \ldots + P_1P_1\ldots P_{n-1}Z_n \quad (4)$$

Where mixed-radix digits can be computed as follows:

$$Z_1 = x_1 \quad (5)$$

$$Z_2 = \left| (x_2 - Z_1) \left| P_1^{-1} \right|_{P_2} \right|_{P_2} \quad (6)$$

$$Z_3 = \left| ((x_3 - Z_1) \left| P_1^{-1} \right|_{P_3} - Z_2) \left| P_2^{-1} \right|_{P_3} \right|_{P_3} \quad (7)$$

$$\vdots$$

$$Z_n = \left| (((x_n - Z_1) \left| P_1^{-1} \right|_{P_n} - Z_2) \left| P_2^{-1} \right|_{P_n} - \ldots - Z_{n-1}) \left| P_{n-1}^{-1} \right|_{P_n} \right|_{P_n} \quad (8)$$

Where $\left|P_i^{-1}\right|_{P_j}$ is denoting the multiplicative inverse of P_i modulo P_j.

2.2 Multiplicative Inverses

The multiplicative inverses in the form of powers of two can lead to reducing the complexity of the reverse converter, since the required multiplications can be substituted with shift operations. The following lemma introduces the simple multiplicative inverses of the proposed set with their proofs.

Lemma 1: The multiplicative inverses for the moduli set $P_1, P_2, P_3 = \{2^{\alpha}, 2^{2\beta+1}-1, 2^{\beta}-1\}$, where $\beta \prec \alpha \preceq 2\beta$ are $\left|P_1^{-1}\right|_{P_2} = 2^{2\beta-\alpha+1}$, $\left|P_1^{-1}\right|_{P_3} = 2^{2\beta-\alpha}$ *and* $\left|P_2^{-1}\right|_{P_3} = 1$.

Proof: We show that $\left|\left|P_i^{-1}\right|_{P_j} \times P_i\right|_{P_j} = 1$. Hence,

$$\left|\left|P_1^{-1}\right|_{P_2} \times P_1\right|_{P_2} = \left|2^{2\beta-\alpha+1} \times 2^{\alpha}\right|_{2^{2\beta+1}-1} = \left|2^{2\beta+1}\right|_{2^{2\beta+1}-1} = 1 \quad (9)$$

$$\left|\left|P_1^{-1}\right|_{P_3} \times P_1\right|_{P_3} = \left|2^{2\beta-\alpha} \times 2^{\alpha}\right|_{2^{\beta}-1} = 1 \quad (10)$$

$$\left|\left|P_2^{-1}\right|_{P_3} \times P_2\right|_{P_3} = \left|2^{2\beta+1} - 1\right|_{2^{\beta}-1} = 1 \quad (11)$$

2.3 Conversion Algorithm

Considering the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$ with its corresponding RNS representation (x_1, x_2, x_3). These residues can be shown in bit-level as below:

$$x_1 = (\underbrace{x_{1,\alpha-1} \ldots x_{1,1} x_{1,0}}_{\alpha bits})_2 \tag{12}$$

$$x_2 = (\underbrace{x_{2,2\beta} \ldots x_{2,1} x_{2,0}}_{\alpha bits})_2 \tag{13}$$

$$x_3 = (\underbrace{x_{3,\beta-1} \ldots x_{3,1} x_{3,0}}_{\alpha bits})_2 \tag{14}$$

The following theorem and lemmas present the proposed conversion algorithm.

Theorem 2: For the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where $\beta < \alpha \leq 2\beta$, the weighted number X can be achieved from its residues (x_1, x_2, x_3) by

$$X = x_1 + 2^{\alpha} Z_2 + 2^{\alpha}(2^{2\beta+1} - 1) Z_3 \tag{15}$$

Where

$$Z_2 = \left| (x_2 - x_1) \times 2^{2\beta-\alpha+1} \right|_{2^{2\beta+1}-1} \tag{16}$$

$$Z_3 = \left| (x_3 - x_1) \times 2^{2\beta-\alpha} - Z_2 \right|_{2^{\beta}-1} \tag{17}$$

Proof. By substituting the moduli $P_1 = 2^{\alpha}$, $P_2 = 2^{2\beta+1}\text{-}1$ and $P_3 = 2^{\beta}\text{-}1$, together with the values of multiplicative inverses from lemma 1 into MRC formulas (4)–(7), the above equations will be achieved. The following properties can be used to simplify (16) and (17), resulting in a reduction in hardware complexity.

Property 1: The residue of a negative residue number (-v) in modulo $(2^k\text{-}1)$ is the one's complement of v, where $0 \leq v < 2^k - 1$ [14].

Property 2: The multiplication of a residue number v by 2^P in modulo $(2^k\text{-}1)$ is carried out by P bit circular left shift, where P is a natural number [14].

Lemma 2: Z_2 is computed as follows:

$$Z_2 = \left| L_1 + \overline{L}_2 \right|_{2^{2\beta+1}-1} \tag{18}$$

Where

$$L_1 = \underbrace{x_{2,\alpha-1} \ldots x_{2,0}}_{\alpha} \underbrace{x_{2,2\beta} \ldots x_{2,\alpha}}_{2\beta-\alpha+1} \tag{19}$$

$$L_2 = \underbrace{x_{1,\alpha-1} \ldots x_{1,1} x_{1,0}}_{\alpha} \underbrace{0 \ldots 00}_{2\beta-\alpha+1} \tag{20}$$

Proof. The above equations can be obtained by applying properties 1 and 2 to (16). Thus,

$$Z_2 = \left| L_1 - L_2 \right|_{2^{2\beta+1}-1} = \left| L_1 + \overline{L}_2 \right|_{2^{2\beta+1}-1} \tag{21}$$

$$L_1 = \left| 2^{2\beta-\alpha+1} \times x_2 \right|_{2^{2\beta+1}-1}$$

$$= \left| 2^{2\beta-\alpha+1} (\underbrace{x_{2,2\beta} \ldots x_{2,1} x_{2,0}}_{2\beta+1}) \right|_{2^{2\beta+1}-1} \tag{22}$$

$$= \left| 2^{2\beta-\alpha+1} (\underbrace{x_{2,2\beta} \ldots x_{2,\alpha}}_{2\beta-\alpha+1} \underbrace{x_{2,\alpha-1} \ldots x_{2,0}}_{\alpha}) \right|_{2^{2\beta+1}-1}$$

$$= \underbrace{x_{2,\alpha-1} \ldots x_{2,0}}_{\alpha} \underbrace{x_{2,2\beta} \ldots x_{2,\alpha}}_{2\beta-\alpha+1}$$

$$L_2 = \left| 2^{2\beta-\alpha+1} \times x_1 \right|_{2^{2\beta+1}-1}$$

$$= \left| 2^{2\beta-\alpha+1} (x_{1,\alpha-1} \ldots x_{1,1} x_{1,0}) \right|_{2^{2\beta+1}-1} \tag{23}$$

$$= \left| 2^{2\beta-\alpha+1} (\underbrace{0 \ldots 00}_{2\beta-\alpha+1} \underbrace{x_{1,\alpha-1} \ldots x_{1,1} x_{1,0}}_{\alpha}) \right|_{2^{2\beta+1}-1}$$

$$= \underbrace{x_{1,\alpha-1} \ldots x_{1,1}}_{\alpha} \underbrace{0 \ldots 00}_{2\beta-\alpha+1}$$

Lemma 3. Z_3 is calculated as below:

$$Z_3 = \left| L_3 + L_{41} + L_{42} + L_{51} + L_{52} + L_8 \right|_{2^{\beta}-1} \tag{24}$$

Where

$$L_3 = \underbrace{x_{3,\alpha-\beta-1} \ldots x_{3,0}}_{\alpha-\beta} \underbrace{x_{3,\beta-1} \ldots x_{3,\alpha-\beta}}_{2\beta-\alpha} \tag{25}$$

$$L_{41} = \underbrace{x_{1,\alpha-\beta-1} \ldots x_{1,0}}_{\alpha-\beta} \underbrace{x_{1,\beta-1} \ldots x_{1,\alpha-\beta}}_{2\beta-\alpha} \tag{26}$$

$$L_{42} = \underbrace{x_{1,\alpha-1} \ldots x_{1,\beta+1} x_{1,\beta}}_{\alpha-\beta} \underbrace{0 \ldots 00}_{2\beta-\alpha} \tag{27}$$

$$L_{51} = \underbrace{\overline{x}_{2,\alpha-\beta-2} \ldots \overline{x}_{2,0}}_{\alpha-\beta-1} \underbrace{\overline{x}_{2,2\beta} \ldots \overline{x}_{2,\alpha}}_{2\beta-\alpha+1} \tag{28}$$

$$L_{52} = \underbrace{\overline{x}_{2,\alpha-2} \ldots \overline{x}_{2,\alpha-\beta} \overline{x}_{2,\alpha-\beta-1}}_{\beta} \tag{29}$$

$$L_8 = \begin{cases} \underbrace{1 \ldots 11}_{\beta-1} \overline{x}_{2,\alpha-1} & if (L_1 - L_2) \geq 0 \\ \underbrace{0 \ldots 00}_{\beta-2} \overline{x}_{2,\alpha-1} x_{2,\alpha-1} & if (L_1 - L_2) < 0 \end{cases} \tag{30}$$

Proof. First, (17) can be rewritten as

$$Z_3 = \left| 2^{2\beta-\alpha} x_3 - 2^{2\beta-\alpha} x_1 - Z_2 \right|_{2^{\beta}-1} \tag{31}$$

Where, from (21) we have

$$Z_2 = \left| L_1 - L_2 \right|_{2^{2\beta+1}-1} \tag{32}$$

The binary vectors L_1 and L_2 both are $(2\beta+1)$-bit numbers, so the maximum value of each one can be $2^{2\beta+1}\text{-}1$; However, L_2 has $2\beta\text{-}\alpha+1$ bits of zero and also L_1 is composed of the bits of the x_2. We know that at least one of the bits of the x_2 is equal to zero, since the maximum value of x_2 is $2^{2\beta+1}\text{-}2$ (due to the fact that x_2 is a residue in modulo $2^{2\beta+1}\text{-}1$). Therefore, L_1 and L_2 are always less than $2^{2\beta+1}\text{-}1$. As a result, the most positive value of the modular subtraction

of (32) will be less than $2^{2\beta+1}$-1 and consequently the reduction in modulo $2^{2\beta+1}$-1 can be removed. Moreover, the most negative value of (32) is higher than $2^{2\beta+1}$-1. Hence, we only need one corrective addition. Then, (32) can be calculated by

$$Z_2 = \begin{cases} L_1 - L_2 & if L_1 - L_2 \geq 0 \\ L_1 - L_2 + (2^{2\beta+1} - 1) & if L_1 - L_2 < 0 \end{cases} \quad (33)$$

Secondly, careful examination of (23) shows that

$$L_2 = \left| 2^{2\beta-\alpha+1} x_1 \right|_{2^{2\beta+1}} = 2^{2\beta-\alpha+1} x_1 \quad (34)$$

Because x_1 is a α-bit number, so representing it in 2β+1 bits where $\beta < \alpha \leq 2\beta$ requires $2\beta - \alpha + 1$ bits of zero before x_1. Thus, (2β-α + 1)-bit circular left shifting of x_1 will become the same as (2β-α + 1)-bit regular left shifting. Now, substituting (34) in (33) yields

$$Z_2 = \begin{cases} L_1 - 2^{2\beta-\alpha+1} x_1 & if (L_1 - L_2) \geq 0 \\ L_1 - 2^{2\beta-\alpha+1} x_1 + (2^{2\beta+1} - 1) & if (L_1 - L_2) < 0 \end{cases} \quad (35)$$

Next, with considering (35) in case of $L_1 - L_2 \geq 0$, (31) can be evaluated as

$$\begin{aligned} Z_3 &= \left| 2^{2\beta-\alpha} x_3 - 2^{2\beta-\alpha} x_1 - (L_1 - 2^{2\beta-\alpha+1} x_1) \right|_{2^\beta-1} \\ &= \left| 2^{2\beta-\alpha} x_3 + x_1 (2^{2\beta-\alpha+1} - 2^{2\beta-\alpha}) - L_1 \right|_{2^\beta-1} \\ &= \left| 2^{2\beta-\alpha} x_3 + 2^{2\beta-\alpha} x_1 - L_1 \right|_{2^\beta-1} \end{aligned} \quad (36)$$

In a similar way, for $L_1 - L_2 < 0$, we obtain the following:

$$Z_3 = \left| 2^{2\beta-\alpha} x_3 + 2^{2\beta-\alpha} x_1 - L_1 - (2^{2\beta+1} - 1) \right|_{2^\beta-1} \quad (37)$$

Therefore, in general case, we have

$$Z_2 = \begin{cases} \left| \overbrace{2^{2\beta-\alpha} x_3 + 2^{2\beta-\alpha} x_1 - L_1}^{H} \right|_{2^\beta-1} & if L_1 - L_2 \geq 0 \\ \left| H - (2^{2\beta+1} - 1) \right|_{2^\beta-1} & if L_1 - L_2 < 0 \end{cases} \quad (38)$$

Now, we must simplify (38) using properties 1 and 2.
First: Let's consider the case $L_1 - L_2 \geq 0$. Equation (38) can be rewritten as

$$Z_3 = \left| L_3 + L_{41} + L_{42} + L_{51} + L_{52} + L_{53} \right|_{2^\beta-1} \quad (39)$$

Where

$$\begin{aligned} L_3 &= \left| 2^{2\beta-\alpha} x_3 \right|_{2^\beta-1} = \left| 2^{2\beta-\alpha} (\underbrace{x_{3,\beta-1} \ldots x_{3,1} x_{3,0}}_{\beta}) \right|_{2^\beta-1} \\ &= \left| 2^{2\beta-\alpha} (\underbrace{x_{3,\beta-1} \ldots x_{3,\alpha-\beta}}_{2\beta-\alpha} \underbrace{x_{3,\alpha-\beta-1} \ldots x_{3,0}}_{\alpha-\beta}) \right|_{2^\beta-1} \\ &= \underbrace{x_{3,\alpha-\beta-1} \ldots x_{3,0}}_{\alpha-\beta} \underbrace{x_{3,\beta-1} \ldots x_{3,\alpha-\beta}}_{2\beta-\alpha} \end{aligned} \quad (40)$$

$$L_4 = \left| 2^{2\beta-\alpha} x_1 \right|_{2^\beta-1} = \left| 2^{2\beta-\alpha} (\underbrace{x_{1,\alpha-1} \ldots x_{1,1} x_{1,0}}_{\alpha}) \right|_{2^\beta-1}$$

$$\begin{aligned} &= \left| 2^{2\beta-\alpha} (\underbrace{x_{1,\alpha-1} \ldots x_{1,\beta}}_{\alpha-\beta} \underbrace{x_{1,\beta-1} \ldots x_{1,0}}_{\beta}) \right|_{2^\beta-1} \\ &= \left| 2^{2\beta-\alpha} (\underbrace{x_{1,\alpha-1} \ldots x_{1,\beta}}_{\alpha-\beta} \times 2^\beta + \underbrace{x_{1,\beta-1} \ldots x_{1,0}}_{\beta}) \right|_{2^\beta-1} \end{aligned} \quad (41)$$

By splitting (41), we have

$$\begin{aligned} L_{41} &= \left| 2^{2\beta-\alpha} \times 2^\beta \underbrace{00 \ldots 0}_{2\beta-\alpha} \underbrace{x_{1,\alpha-1} \ldots x_{1,\beta}}_{\alpha-\beta} \right|_{2^\beta-1} \\ &= \underbrace{x_{1,\alpha-1} \ldots x_{1,\beta}}_{\alpha-\beta} \underbrace{00 \ldots 0}_{2\beta-\alpha} \end{aligned} \quad (42)$$

$$\begin{aligned} L_{42} &= \left| 2^{2\beta-\alpha} (\underbrace{x_{1,\beta-1} \ldots x_{1,0}}_{\beta}) \right|_{2^\beta-1} \\ &= \left| 2^{2\beta-\alpha} (\underbrace{x_{1,\beta-1} \ldots x_{1,\alpha-\beta}}_{2\beta-\alpha} \underbrace{x_{1,\alpha-\beta-1} \ldots x_{1,0}}_{\alpha-\beta}) \right|_{2^\beta-1} \\ &= \underbrace{x_{1,\alpha-\beta-1} \ldots x_{1,0}}_{\alpha-\beta} \underbrace{x_{1,\beta-1} \ldots x_{1,\alpha-\beta}}_{2\beta-\alpha} \end{aligned} \quad (43)$$

Subsequently, the reduction of -L_1 in modulo 2^β-1 can be performed by considering (19) as follows

$$\begin{aligned} L_5 &= \left| -L_1 \right|_{2^\beta-1} = \left| -(\underbrace{x_{2,\alpha-1} \ldots x_{2,0}}_{\alpha} \underbrace{x_{2,2\beta} \ldots x_{2,\alpha}}_{2\beta-\alpha+1}) \right|_{2^\beta-1} \\ &= \left| -(x_{2,\alpha-1} \times 2^{2\beta} + \underbrace{x_{2,\alpha-2} \ldots x_{2,\alpha-\beta} x_{2,\alpha-\beta-1}}_{\beta} \times 2^\beta) \right. \\ &\quad \left. + \underbrace{x_{2,\alpha-\beta-2} \ldots x_{2,0}}_{\alpha-\beta-1} \underbrace{x_{2,2\beta} \ldots x_{2,\alpha}}_{2\beta-\alpha+1} \right|_{2^\beta-1} \end{aligned} \quad (44)$$

Now, by separating (44) into three parts, we achieve

$$\begin{aligned} L_{51} &= \left| -(\underbrace{x_{2,\alpha-\beta-2} \ldots x_{2,0}}_{\alpha-\beta-1} \underbrace{x_{2,2\beta} \ldots x_{2,\alpha}}_{2\beta-\alpha+1}) \right|_{2^\beta-1} \\ &= \underbrace{\overline{x}_{2,\alpha-\beta-2} \ldots \overline{x}_{2,0}}_{\alpha-\beta-1} \underbrace{\overline{x}_{2,2\beta} \ldots \overline{x}_{2,\alpha}}_{2\beta-\alpha+1} \end{aligned} \quad (45)$$

$$\begin{aligned} L_{52} &= \left| -2^\beta (\underbrace{x_{2,\alpha-2} \ldots x_{2,\alpha-\beta} x_{2,\alpha-\beta-1}}_{\beta}) \right|_{2^\beta-1} \\ &= \underbrace{\overline{x}_{2,\alpha-2} \ldots \overline{x}_{2,\alpha-\beta} \overline{x}_{2,\alpha-\beta-1}}_{\beta} \end{aligned} \quad (46)$$

$$L_{53} = \left| -2^{2\beta} (\underbrace{0 \ldots 00}_{\beta-1} x_{2,\alpha-1}) \right|_{2^\beta-1} = \underbrace{1 \ldots 11}_{\beta-1} \overline{x}_{2,\alpha-1} \quad (47)$$

Secondly: Let's consider the case $L_1 - L_2 < 0$. Equation (38) can be simplified in the same way as before with only one additional vector to taking into account -($2^{2\beta+1}$-1). Thus,

$$Z_3 = \left| L_3 + L_{41} + L_{42} + L_{51} + L_{52} + L_{53} + L_6 \right|_{2^\beta-1} \quad (48)$$

Where

$$L_6 = \left| -(2^{2\beta+1}-1)\right|_{2^\beta-1} = \left| -1\right|_{2^\beta-1} = \underbrace{1\ldots11}_{\beta-1}0 \qquad (49)$$

The other binary vectors are previously obtained (40)–(47). The (47) and (49) both have β-1 bits with constant values. So, we can merge L_{53} and L_6 to achieve one vector as

$$\begin{aligned} L_7 &= \left| L_{53}+L_6\right|_{2^\beta-1} = \left| \underbrace{1\ldots11}_{\beta-1}\overline{x}_{2,\alpha-1}+\underbrace{1\ldots11}_{\beta-1}0\right|_{2^\beta-1} \\ &= \left|10\underbrace{\ldots00}_{\beta-1}\overline{x}_{2,\alpha-1}\right|_{2^\beta-1} = \left|2^\beta+\underbrace{0\ldots00}_{\beta-1}\overline{x}_{2,\alpha-1}\right|_{2^\beta-1} \\ &= \left|1+\underbrace{0\ldots00}_{\beta-1}\overline{x}_{2,\alpha-1}\right|_{2^\beta-1} = \underbrace{0\ldots00}_{\beta-2}\overline{x}_{2,\alpha-1}x_{2,\alpha-1} \end{aligned} \qquad (50)$$

Therefore, seven operands of (48) reduced to six as shown below:

$$Z_3 = \left| L_3 + L_{41} + L_{42} + L_{51} + L_{52} + L_7\right|_{2^\beta-1} \qquad (51)$$

Finally, the following equation can be used instead of (39) and (51) to realize both cases.

$$Z_3 = \left| L_3 + L_{41} + L_{42} + L_{51} + L_{52} + L_8\right|_{2^\beta-1} \qquad (52)$$

Where

$$L_8 = \begin{cases} L_{53} & if(L_1 - L_2) \geq 0 \\ L_7 & if(L_1 - L_2) < 0 \end{cases} \qquad (53)$$

2.4 Hardware Architecture

The hardware architecture of the proposed general reverse converter is depicted in Fig. 1 and is based on theorem 2 and lemmas 2 and 3. To implement the main equation, i.e., (15), we must first realize the mixed-radix coefficients (16) and (17). Lemmas 2 and 3 provide simplified versions of (16) and (17) without direct dependency between Z_2 and Z_3. First of all, the required operands (19), (20), (25)–(30) are prepared by operand preparation unit 1 (OPU 1) with only some NOT gates and wiring. Next, we need a modulo 2^β-1 adder to realize (18). Modular adders can be implemented using different methods; however, this paper considers the carry-propagate adder (CPA) with end-around carry (EAC) [29] to realize modulo of the forms 2^k-1 addition. The CPA with EAC has the same hardware complexity and double delay than the regular CPA. Therefore, realization of (18) relies on a β-bit CPA with EAC. Also, a six-operand modulo $2^{2\beta+1}$-1 adder [30] is employed to implement (24). This multi-operand modular adder can be mechanized using a six inputs carry-save adder (CSA) tree followed by a $(2^{2\beta+1}$-1)-bit CPA with EAC. This CSA tree consists of four $(2\beta+1)$-bit CSAs with EACs as shown in Fig. 2. Some of the full adders (FAs) are reduced to XOR/AND or XNOR/OR pairs, since some inputs of the CSAs have constant value of one or zero. One of the main features of lemma 3 is that it removes the direct dependency to Z_2 which exists in (17);

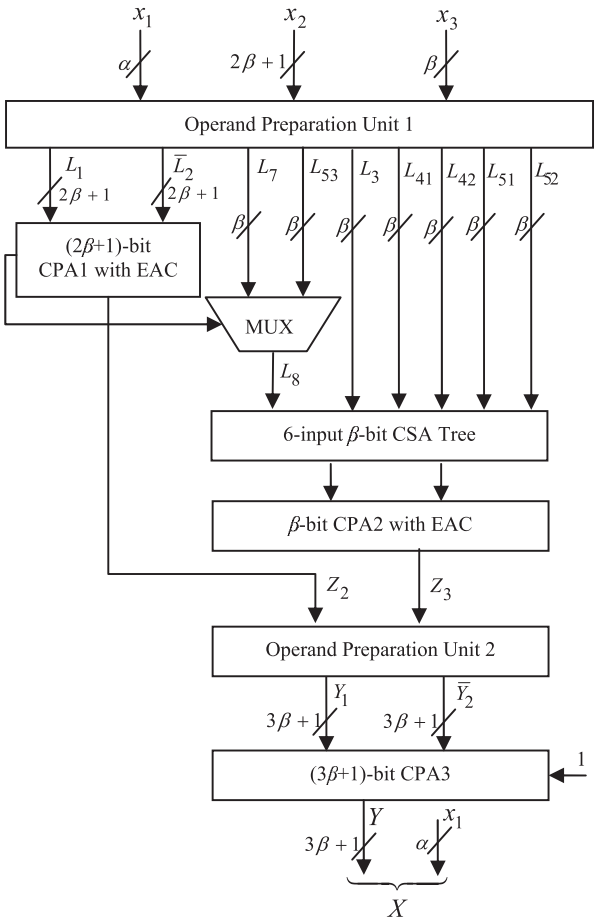

Fig. 1 The proposed general reverse converter architecture.

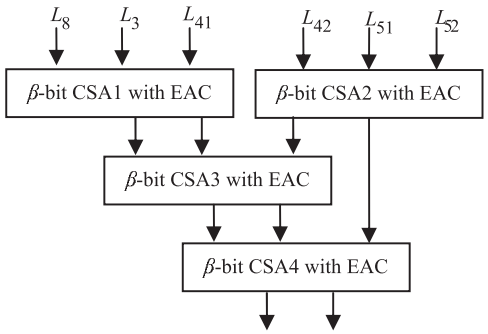

Fig. 2 The 6-input β-bit CSA tree.

however the carry of the first round addition of CPA1 with EAC is needed to achieve (30). In other words, the EAC bit of CPA1 determines the sign of $L_1 - L_2$. Thus, we used a 2×1 β-bit multiplexer (MUX) with inputs of (47) and (50), where the select line is connected to the carry-out of CPA1. Moreover, implementation of (15) can be done using simple concatenations followed by a regular binary addition. Clearly, (15) can be rewritten as

$$X = x_1 + 2^\alpha Y \qquad (54)$$

Where

Table 1 Details of each part of the proposed general architecture.

Parts	FA	NOT	XOR/AND pairs	XNOR/OR pairs	MUX 2×1	Delay
OPU1	-	$2\beta+\alpha+1$	-	-	β	$D_{NOT}+D_{MUX}$
CPA1	α	-	-	$2\beta+\alpha+1$	-	$(4\beta+2)D_{FA}$
CSA1	2	-	$\beta-2$	-	-	D_{FA}
CSA2	$\alpha-\beta$	-	$2\beta-\alpha$	-	-	D_{FA}
CSA3	β	-	-	-	-	D_{FA}
CSA4	β	-	-	-	-	D_{FA}
CPA2	β	-	-	-	-	$2\beta D_{FA}$
OPU2	-	β	-	-	-	D_{NOT}
CPA3	β	-	-	$2\beta+1$	-	$(3\beta+1)D_{FA}$

$$Y=\underbrace{Z_2+2^{2\beta+1}Z_3}_{3\beta+1 bits}-Z_3=Y_1+\overline{Y}_2+1 \tag{55}$$

Preparing the operands of (55) relies on simple concatenations and inversions as shown below

$$Y_1=\underbrace{Z_{3,\beta-1}\ldots Z_{3,1}Z_{3,0}}_{\beta}\underbrace{Z_{2,2\beta}\ldots Z_{2,1}Z_{2,0}}_{2\beta+1} \tag{56}$$

$$\overline{Y}_2=\underbrace{1\ldots11}_{2\beta+1}\underbrace{\overline{Z}_{3,\beta-1}\ldots\overline{Z}_{3,1}\overline{Z}_{3,0}}_{\beta} \tag{57}$$

Therefore, a $(3\beta+1)$-bit regular CPA with '1' carry-in is needed to add (56) and (57), where OPU 2 prepares Y_1 and Y_2. Note that, $2\beta+1$ FAs of CPA3 are reduced to $2\beta+1$ XNOR/OR pairs, since (57) has $2\beta+1$ constant bits with value of one. Finally, due to the fact that x_1 is a α-bit number, (54) can be achieved by a simple concatenation without using any computational hardware. Table 1 describes area and delay specifications for different parts of the converter. It should be mentioned that the numbers of FAs and logic gates in CSA1 depends on (30). If $L_1-L_2\geq 0$ then one FA and $(\beta$-1) XNOR/OR pairs will be used in CSA1. Otherwise if $L_1-L_2<0$, two FAs and $(\beta$-2) XOR/AND pairs will be needed. In Table 1, we consider the second case to derive area details of CSA1; however it may change if $L_1-L_2\geq 0$. Furthermore, although the delay of CPA1 with EAC is $(4\beta+2)D_{FA}$, the carry-out will be available after $(2\beta+1)D_{FA}$. Thus, the critical path delay can be obtained as

$$Delay=(7\beta+4)D_{FA}+D_{MUX}+2D_{NOT} \tag{58}$$

Where D_{FA}, D_{NOT} *and* D_{MUX} are indicating the delay of one FA, NOT gate and MUX, respectively.

3. Reverse Converter for $\{2^{2n}, 2^{2n+1}-1, 2^n-1\}$

In this section a new moduli set with its specialized reverse converter is presented. The motivation to propose this new set as well as the way to derive the reverse converter from general architecture is described in the followings.

3.1 Introducing New Moduli Set

Many considerations are given on the 5n-bit DR residue number systems in recent years and new moduli sets with this DR are introduced. The first moduli set was $\{2^n, 2^n-1, 2^n+1, 2^n-2^{(n+1)/2}+1, 2^n+2^{(n+1)/2}+1\}$ and the best reverse converter for this set presented in [25]. The main drawbacks are the moduli $2^n-2^{(n+1)/2}+1$ and $2^n+2^{(n+1)/2}+1$ that result in decreasing performance of the arithmetic operation. Therefore in [26], the moduli set $\{2^n-1, 2^n, 2^n+1, 2^{2n}+1\}$ was suggested. For the first time, four moduli set are used to provide more than 4n-bit DR. The reverse converter of the above mentioned work has higher performance and also faster arithmetic operations in comparison to [25]. Moreover, the three moduli set $\{2^n, 2^{2n}-1, 2^{2n}+1\}$ [14] has been also proposed with 5n-bit DR and faster reverse converter compared to [25] and [26]. Although, moduli sets reported in [14], [25] and [26] can provide high DR but presence of the moduli 22n+1 caused inefficient arithmetic operations. Hence, the set $\{2^n-1, 2^n, 2^n+1, 2^{n-1}-1, 2^{n+1}-1\}$ [27] with balanced moduli is introduced. However, unfavorable multiplicative inverses of this set lead to noticeable decreases in the reverse converter performance. In the newly reported work [28], the moduli set $\{2^n-1, 2^n, 2^n+1, 2^{2n+1}-1\}$ is proposed to solve the problem of inefficient multiplicative inverses. Comparing to [27], their moduli set provides faster reverse converter with the same speed of the RNS arithmetic unit and also less hardware complexity. However, the delay of the converter of [28] is longer than [14]. Therefore, the lack of a moduli set that can provide better tradeoff between fast arithmetic operations and efficient reverse converter in 5n-bit DR is evident. Hence, we propose the new 5n-bit DR moduli set $\{2^{2n}, 2^{2n+1}-1, 2^n-1\}$. This set can result in a very efficient RNS arithmetic unit, since it is free from modulo 2n+1. Furthermore, the critical modulo of this set is $2^{2n+1}-1$ and as investigated in [28], the moduli $2^{2n+1}-1$ can result in a slightly faster modular addition than 2^n+1. So, it can be concluded that the proposed moduli set is slightly faster than moduli sets of [27] and [28] and quite faster than other moduli sets in 5n-bit DR class which have been introduced in [14], [25] and [26].

3.2 Converter Design

The moduli set $\{2^{2n}, 2^{2n+1}-1, 2^n-1\}$ is a special case of the general set $\{2^\alpha, 2^{2\beta+1}-1, 2^\beta-1\}$, where $\beta\prec\alpha\preceq 2\beta$. Hence, its reverse converter can be derived from the general architecture by substituting α=2n and β=n into the general conversion equations which are described in Sect. 2. First, from lemma 2 formulas (18)–(20), we have

$$Z_2 = \left| L_1 + \overline{L}_2 \right|_{2^{2n+1}-1} \tag{59}$$

Where

$$L_1 = \underbrace{x_{2,2n-1} \ldots x_{2,1} x_{2,0}}_{2n} x_{2,2n} \tag{60}$$

$$L_2 = \underbrace{x_{1,2n-1} \ldots x_{1,1} x_{1,0}}_{2n} 0 \tag{61}$$

Second, from lemma 3 formulas (24)–(30), we have

$$Z_3 = \left| L_3 + L_{41} + L_{42} + L_{51} + L_{52} + L_8 \right|_{2^n-1} \tag{62}$$

Where

$$L_3 = \underbrace{x_{3,n-1} \ldots x_{3,1} x_{3,0}}_{n} \tag{63}$$

$$L_{41} = \underbrace{x_{1,n-1} \ldots x_{1,1} x_{1,0}}_{n} \tag{64}$$

$$L_{42} = \underbrace{x_{1,2n-1} \ldots x_{1,n+1} x_{1,n}}_{n} \tag{65}$$

$$L_{51} = \underbrace{\overline{x}_{2,n-2} \ldots \overline{x}_{2,1} \overline{x}_{2,0}}_{n-1} \overline{x}_{2,2n} \tag{66}$$

$$L_{52} = \underbrace{\overline{x}_{2,2n-2} \ldots \overline{x}_{2,n} \overline{x}_{2,n-1}}_{n} \tag{67}$$

$$L_8 = \begin{cases} \underbrace{1 \ldots 11}_{n-1} \overline{x}_{2,2n-1} & if \quad (L_1 - L_2) \geq 0 \\ \underbrace{0 \ldots 00}_{n-2} \overline{x}_{2,2n-1} x_{2,2n-1} & if \quad (L_1 - L_2) < 0 \end{cases} \tag{68}$$

Third, the final conversion equations can be obtained from the simplified relations of theorem 2, i.e., (54)–(57) as follows

$$X = x_1 + 2^{2n} Y = \underbrace{Y_{3n} \ldots Y_1 Y_0}_{3n+1} \underbrace{x_{1,2n-1} \ldots x_{1,1} x_{1,0}}_{2n} \tag{69}$$

Where

$$Y = \underbrace{Z_2 + 2^{2n+1} Z_3}_{3n+1 bits} - Z_3 = Y_1 + \overline{Y}_2 + 1 \tag{70}$$

$$Y_1 = \underbrace{Z_{3,n-1} \ldots Z_{3,1} Z_{3,0}}_{n} \underbrace{Z_{2,2n} \ldots Z_{2,1} Z_{2,0}}_{2n+1} \tag{71}$$

$$\overline{Y}_2 = \underbrace{1 \ldots 11}_{2n+1} \underbrace{\overline{Z}_{3,n-1} \ldots \overline{Z}_{3,1} \overline{Z}_{3,0}}_{n} \tag{72}$$

The hardware implementation and also components details are the same as shown in Figs. 1–2 and Table 1 with α=2n and β=n.

3.3 Numerical Example

Consider the moduli set {16, 31, 3} which is derived from $\{2^{2n}, 2^{2n+1}-1, 2^n\text{-}1\}$, where n=2. The RNS number (12, 25, 2) can be converted into its equivalent weighted number by doing the following steps:

1) Binary representation of residues (12)–(14):

$$x_1 = x_{1,3} x_{1,2} x_{1,1} x_{1,0} = 1100$$
$$x_2 = x_{2,4} x_{2,3} x_{2,2} x_{2,1} x_{2,0} = 11001$$
$$x_3 = x_{3,1} x_{3,0} = 10$$

2) Obtaining Z_2 (59)–(61):

$$L_1 = x_{2,3} x_{2,2} x_{2,1} x_{2,0} x_{2,4} = 10011$$
$$L_2 = x_{1,0} x_{1,2} x_{1,1} x_{1,0} 0 = 11000$$
$$Z_2 = \left| 10011 + 00111 \right|_{31} = 11010$$

3) Obtaining Z_3 (62)–(68):

$$L_3 = x_{3,1} x_{3,0} = 10, L_{41} = x_{1,1} x_{1,0} = 00$$
$$L_{42} = x_{1,3} x_{1,2} = 11, L_{51} = \overline{x}_{2,0} \overline{x}_{2,4} = 00$$
$$L_{52} = \overline{x}_{2,2} \overline{x}_{2,1} = 11, L_8 = \overline{x}_{2,3} x_{2,3} = 01$$
$$Z_3 = \left| 10 + 11 + 00 + 11 + 00 + 01 \right|_3 = 00$$

4) Calculating X according to (68)–(71):

$$Y_1 = Z_{3,1} Z_{3,3} Z_{2,4} Z_{2,3} Z_{2,2} Z_{2,1} Z_{2,0} = 0011010$$
$$\overline{Y}_2 = 11111 \overline{Z}_{3,1} \overline{Z}_{3,0} = 1111111$$
$$Y = Y_1 + \overline{Y}_2 + 1 = 0011010 + 1111111 + 1 = 0011010$$
$$X = Y_6 Y_5 Y_4 Y_3 Y_2 Y_1 Y_0 x_{1,3} x_{1,2} x_{1,1} x_{1,0} = 00110101100$$

Thus, X=428, and verification can be simply done as

$$x_1 = \left| 428 \right|_{16} = 12$$
$$x_2 = \left| 428 \right|_{31} = 25$$
$$x_3 = \left| 428 \right|_3 = 2$$

3.4 Performance Evaluation

This section evaluates the performance of the proposed reverse converter for the moduli set $\{2^{2n}, 2^{2n+1}-1, 2^n\text{-}1\}$, and compares it with the performance of the best state-of-art reverse converters for moduli sets in the class of 5n-bit DR such as $\{2^n, 2^{2n}-1, 2^{2n}+1\}$ [14], $\{2^n-1, 2^n, 2^n+1, 2^{2n}+1\}$ [26], $\{2^n, 2^n-1, 2^n+1, 2^n-2^{(n+1)/2}+1, 2^n+2^{(n+1)/2}+1\}$ [25], $\{2^n-1, 2^n, 2^n+1, 2^{n-1}-1, 2^{n+1}\text{-}1\}$ [27] and $\{2^n-1, 2^n, 2^n+1, 2^{2n+1}\text{-}1\}$ [28]. Table 2 presents hardware requirements and conversion delays of these reverse converters in terms of logic gates and FAs. Note that all the assumptions used in [28] are considered to obtain the formulas of Table 2, such as using k-bit CPAs with EACs for the implementation of the moduli of the form 2^k-1 adders of all of the converters. Furthermore, the hardware requirements and conversion delay of the proposed reverse converter for the moduli set $\{2^{2n}, 2^{2n+1}-1, 2^n\text{-}1\}$ is derived from Table 1 and (58), respectively, where α=2n and β=n. The results show that efficient tradeoff between hardware requirements and conversion delay is obtained. However, to achieve precise estimations for

Table 2 Hardware requirements and conversion delays of the different reverse converter.

Converter	Moduli set	Hardware requirements	Conversion delay
[25]	$\{2^n-1, 2^n, 2^n+1, 2^n-2^{(n+1)/2}+1, 2^n+2^{(n+1)/2}+1\}$	$(19n)A_{FA}+(7n)A_{XOR}+(7n)A_{AND}$ $+(2n)A_{XNOR}+(2n)A_{OR}+(4n)A_{NOT}$	$(8n+4)D_{FA}+D_{NOT}$
[27]	$\{2^n-1, 2^n, 2^n+1, 2^{n-1}-1, 2^{n+1}-1\}$	$((5n^2+43n+m^*)/6+16n-1)A_{FA}+(6n+1)A_{NOT}$	$(18n+l^*+7)D_{FA}$
[28]	$\{2^n-1, 2^n, 2^n+1, 2^{2n+1}-1\}$	$(8n+2)A_{FA}+(n-1)A_{XOR}+(n-1)A_{AND}$ $+(4n+1)A_{XNOR}+(4n+1)A_{OR}+(7n+1)A_{NOT}+(n)A_{MUX2\times1}$	$(12n+5)D_{FA}+D_{MUX}$ $+3D_{NOT}$
[26]	$\{2^n-1, 2^n, 2^n+1, 2^{2n}+1\}$	$(11n+6)A_{FA}+(2n-1)A_{XOR}+(2n-1)A_{AND}+$ $(4n)A_{XNOR}+(4n)A_{OR}+(5n+3)A_{NOT}$	$(8n+3)D_{FA}+D_{NOT}$
[14]	$\{2^n, 2^{2n}-1, 2^{2n}+1\}$	$(5n+2)A_{FA}+(2n-1)A_{XOR}+(2n-1)A_{AND}$ $+(n-1)A_{XNOR}+(n-1)A_{OR}+(3n+1)A_{NOT}$	$(8n+1)D_{FA}+D_{NOT}$
Proposed	$\{2^{2n}, 2^{2n+1}-1, 2^n-1\}$	$(7n+2)A_{FA}+(n-2)A_{XOR}+(n-2)A_{AND}+(2n+2)A_{XNOR}$ $+(2n+2)A_{OR}+(5n+1)A_{NOT}+(n)A_{MUX2\times1}$	$(7n+4)D_{FA}+D_{MUX}$ $+2D_{NOT}$

* m=n-4, 9n-12 and 5n-8 for n=6k-2, 6k and 6k+2, respectively, and l is the number of the levels of the CSA tree with ((n/2)+1) inputs.

Table 3 Implementation results of the converter on FPGA.

Converter	n=8				n=16				n=24			
	Area		Delay		Area		Delay		Area		Delay	
	(Slices)	Saving	(ns)	Speed-up	(Slices)	Saving	(ns)	Speed-up	(Slices)	Saving	(ns)	Speed-up
[25]*	294	52.3%	26.572	24.2%	550	51.1%	45.737	29%	804	49.2%	64.937	38.5%
[27]	328	57.3%	43.325	53.5%	764	64.8%	86.575	62.5%	1168	68.1%	124.782	68%
[28]	141	0.7%	32.185	37.4%	293	8.2%	60.882	46.6%	451	9.5%	83.464	52.2%
[26]	178	21.3%	22.784	11.6%	354	24%	41.984	22.6%	530	23%	61.184	34.7%
[14]	116	-17.1%	21.673	7.1%	236	-12.2%	38.898	16.5%	356	-12.7%	50.002	20.1%
Proposed	140	-	20.124	-	269	-	32.471	-	408	-	39.927	-

*Implementation for this work is done with n equal to 9, 17 and 25.
Hardware savings and speed-up are calculated based on $\frac{Area_{other}-Area_{proposed}}{MAX(Area_{proposed},Area_{other})}\times 100$ and $\frac{Delay_{other}-Delay_{proposed}}{MAX(Delay_{proposed},Delay_{other})}\times 100$.

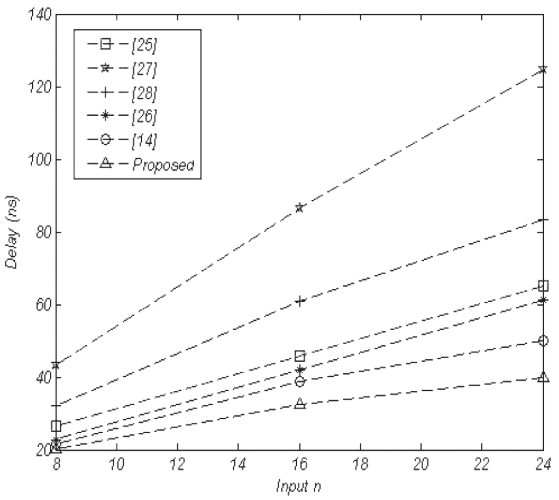

Fig. 3 Comparison the delay of the different converters.

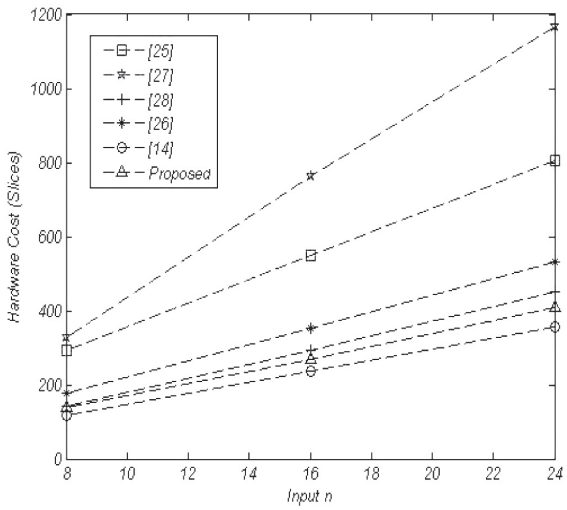

Fig. 4 Comparison the area of the different converters.

area and delay, the proposed design as well as other converters were described in VHDL, and implemented using FPGA technology. The target technology is a Xilinx Virtex-5 FPGA and the area is evaluated by the number of occupied slices. Table 3 compares the area and delay of the converters showing the amount of improvement (%) for different n. As it is expected, delay of the proposed design is the least than the other converters. Comparing to fastest reverse converter which is proposed in [14], 16.5% and 20.1% improvement in terms of speed of the reverse converter when n is equal to 16 and 24 respectively is achieved. Speed and area improvement compared to other converters are higher than these results. In order to ease the comparison, Figs. 3 and 4 are produced to show the practical delay and area comparison for converters based on the result of Table 3. The Fig. 3 confirms that with the growth of n, noticeable reduction in reverse conversion delay will be achieved. Moreover, Fig. 4 shows the noticeable hardware saving compared to other converters. There is only one work reported in [14] that needs less hardware requirement; however the ineffi-

ciency of arithmetic operation due to the moduli 2^{2n}+1 and lower speed of reverse converter forces this moduli set to decrease the total efficiency of RNS.

4. Conclusion

We have presented a simple and efficient general reverse converter architecture which is constructed based on the moduli set $\{2^{\alpha}, 2^{2\beta+1}\text{-}1, 2^{\beta}\text{-}1\}$, where $\beta < \alpha \leq 2\beta$. Due to the absence of the low-performance modulo $(2^{\beta}+1)$ together with simple multiplica1tive inverses, the introduced moduli set is suitable for realizing large DRs, fast modulo arithmetic circuits and efficient forward/reverse converters providing high-performance RNS systems. The general reverse converter architecture has been built using a FA-based implementation of MRC where some novel techniques have been used to eliminate the dependency between mixed-radix coefficients to achieve high-speed. Moreover, the moduli set $\{2^{2n}, 2^{2n+1} - 1, 2^{n}\text{-}1\}$ is suggested with its specialized reverse converter derived from the proposed general architecture with high-speed and low-cost, compared to the best state-of-the-art reverse converters. In any case, modularity and regularity of our design makes it suitable for efficient VLSI implementation.

Acknowledgment

The authors would like to thank Dr. Belmond Yoberd for his literature contribution.

References

[1] M.A. Bayoumi and P. Srinivasan, "Parallel arithmetic: From algebra to architecture," Proc. IEEE Symposium on Circuits and Systems, pp.2630–2633, 1990.
[2] T. Stouratitis and V. Paliouras, "Considering the alternatives in low-power design," IEEE Circuits and Devices, vol.7, pp.23–29, 2001.
[3] K. Navi, A.S. Molahosseini, and M. Esmaeildoust, "How to teach residue number system to computer scientists and engineers," IEEE Trans. Educ., vol.3, no.4, 2010.
[4] M.A. Soderstrand, W.K. Jenkins, G.A. Jullien, and F.J. Taylor, Residue number system arithmetic: modern applications in digital signal processing, IEEE Press, 1986.
[5] R. Chaves and L. Sousa, "RDSP: A RISC DSP based on residue number system," Proc. Euromicro Conference on Digital System Design: Architectures, Methods, and Tools, pp.128–135, 2003.
[6] R. Chokshi, K.S. Berezowski, A. Shrivastava, and S.J. Piestrak, "Exploiting residue number system for power-efficient digital signal processing in embedded processors," Proc. International Conference on Compilers, Architecture and Synthesis for Embedded Systems, pp.19–28, 2009.
[7] J. Ramirez, U. Meyer-Base, and A. Garcia, "Efficient RNS-based design of programmable FIR filters targeting FPL technology," J. Circuits Syst. Comput., vol.14, no.1, pp.165–177, 2005.
[8] V.T. Goh and M.U. Siddiqi, "Multiple error detection and correction based on redundant residue number systems," IEEE Trans. Commun., vol.56, no.3, pp.325–330, 2008.
[9] S. Timarchi and K. Navi, "Arithmetic circuits of redundant SUT-RNS," IEEE Trans. Instrum. Meas., vol.58, no.9, pp.2959–2968, 2009.
[10] T. Tomczak, "Fast sign detection for RNS $(2^n - 1, 2^n, 2^n + 1)$," IEEE Trans. Circuits Syst. I, vol.55, no.6, pp.1502–1511, 2008.
[11] S. Bi and W.J. Gross, "The mixed-radix chinese remainder theorem and its applications to residue comparison," IEEE Trans. Comput., vol.57, no.12, pp.1624–1632, 2008.
[12] M. Dasygenis, K. Mitroglou, D. Soudris, and A. Thanailakis, "A full-adder-based methodology for the design of scaling operation in residue number system," IEEE Trans. Circuits Syst. I, vol.55, no.2, pp.546–558, 2008.
[13] W. Wang, M.N.S. Swamy, and M.O. Ahmad, "Moduli selection in RNS for efficient VLSI implementation," Proc. IEEE Symposium on Circuits and Systems, 2003.
[14] A. Hariri, K. Navi, and R. Rastegar, "A new high dynamic range moduli set with efficient reverse converter," Elsevier Journal of Computers and Mathematics with Applications, vol.55, no.4, pp.660–668, 2008.
[15] A.S. Molahosseini, K. Navi, O. Hashemipour, and A. Jalali, "An efficient architecture for designing reverse converters based on a general three-moduli set," Elsevier Journal of Systems Architecture, vol.54, no.10, pp.929–934, 2008.
[16] R. Chaves and L. Sousa, "Improving residue number system multiplication with more balanced moduli sets and enhanced modular arithmetic structures," IET Comput. Digit. Tech., vol.1, no.5, pp.472–480, 2007.
[17] W. Wang, M.N.S. Swamy, M.O. Ahmad, and Y. Wang, "A high-speed residue-to-binary converter and a scheme of its VLSI implementation," IEEE Trans. Circuits Syst. II, vol.47, no.12, pp.1576–1581, 2000.
[18] P.V.A. Mohan, "RNS-to-binary converter for a new three-moduli set $2^{n+1} - 1, 2^n, 2^n - 1$," IEEE Trans. Circuits Syst. II, vol.54, no.9, pp.775–779, 2007.
[19] A.S. Molahosseini, C. Dadkhah, K. Navi, and M. Eshghi, "Efficient MRC-based residue to binary converters for the new moduli sets $2^{2n}, 2^n - 1, 2^{n+1} - 1$ and $2^{2n}, 2^n - 1, 2^{n-1} - 1$," IEICE Trans. Inf. & Syst., vol.E92-D, no.9, pp.1628–1638, Sept. 2009.
[20] P.V.A. Mohan and A.B. Premkumar, "RNS-to-binary converters for two four-moduli set $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and $\{2^n\text{-}1, 2^n, 2^n\text{+}1, 2^{n+1} + 1\}$," IEEE Trans. Circuits Syst. I, vol.54, no.6, pp.1245–1254, 2007.
[21] B. Cao, T. Srikanthan, and C.H. Chang, "Efficient reverse converters for the four-moduli sets $\{2^n\text{-}1, 2^n, 2^n\text{+}1, 2^{n+1}\text{-}1\}$ and $\{2^n\text{-}1, 2^n, 2^n\text{+}1, 2^{n-1}\text{-}1\}$," IEE Proc. Comput. Digit. Tech., vol.152, pp.687–696, 2005.
[22] M.H. Sheu, S.H. Lin, C. Chen, and S.W. Yang, "An efficient VLSI design for a residue to binary converter for general balance moduli $\{2^n - 3, 2^n + 1, 2^n - 1, 2^n + 3\}$," IEEE Trans. Circuits Syst. II, vol.51, no.2, pp.152–155, 2004.
[23] L.S. Didier and P.Y. Rivaille, "A generalization of a fast RNS conversion for a new 4-modulus base," IEEE Trans. Circuits Syst. II, vol.56, no.1, pp.46–50, 2009.
[24] W. Zhang and P. Siy, "An efficient design of residue to binary converter for four moduli set $\{2^n - 1, 2^n + 1, 2^{2n} - 2, 2^{2n+1} - 3\}$ based on new CRT II," Elsevier Journal of Information Sciences, vol.178, no.1, pp.264–279, 2008.
[25] A.A. Hiasat, "VLSI implementation of new arithmetic residue to binary decoders," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.13, no.1, pp.153–158, 2005.
[26] B. Cao, C.H. Chang, and T. Srikanthan, "An efficient reverse converter for the 4-moduli set $2^n - 1, 2^n, 2^n + 1, 2^{2n} + 1$ based on the new Chinese remainder theorem," IEEE Trans. Circuits Syst. I, vol.50, no.10, pp.1296–1303, 2003.
[27] B. Cao, C.H. Chang, and T. Srikanthan, "A residue-to-binary converter for a new five-moduli set," IEEE Trans. Circuits Syst. I, vol.54, no.5, pp.1041–1049, 2007.
[28] A.S. Molahosseini, K. Navi, C. Dadkhah, O. Kavehei, and S. Timarchi, "Efficient reverse converter designs for the new 4-moduli sets $2^n - 1, 2^n, 2^n + 1, 2^{2n+1} - 1$ and $2^n - 1, 2^n + 1, 2^{2n}, 2^{2n} + 1$ based on new CRTs," IEEE Trans. Circuits Syst. I, vol.57, issue 4, pp.823–835, 2010.

[29] S.J. Piestrak, "Design of residue generators and multioperand modular adders using carry-save adders," IEEE Trans. Comput., vol.423, no.1, pp.68–77, 1994.
[30] S.J. Piestrak, "A high speed realization of a residue to binary converter," IEEE Trans. Circuits Syst. II, vol.42, no.10, pp.661–663, 1995.

Keivan Navi received M.Sc. degrees in Electrical Engineering (Computer Hardware) from Sharif University of Technology, Tehran, Iran, in 1990. He also received the Ph.D. degree in computer architecture from Paris XI University, Paris, France, in 1995. He is currently Associate Professor in faculty of electrical and computer engineering of Beheshti University. His research interests include Residue Number System, Carbon Nano-tube, Single Electron transistors, Reversible Logic Design, Interconnection network and Quantum Computing.

Mohammad Esmaeildoust is Phd candidate in Computer architecture at Shahid Beheshti University of Technology (Tehran, Iran). He received his MSc degree in Computer architecture at Shahid Beheshti University of Technology (Tehran, Iran) in 2008. He received his BSc degree in 2006 from shahed University in Hardware Engineering. He is working on reconfigurable computing and computer arithmetic especially on residue number system.

Amir Sabbagh Molahosseini was born in Kerman, Iran, in 1983. He received the B.Sc. degree from Shahid Bahonar University of Kerman, Iran, in 2005, and the M.Sc. degree (with highest honors) from Islamic Azad University (IAU), Science and Research Branch, Tehran, Iran, in 2007, both in Computer Engineering. He is currently working toward the Ph.D. degree in computer engineering at Science & Research Branch of IAU, Tehran, Iran. His research interests include VLSI design and computer arithmetic with emphasis on residue number system.