Energy-Efficient Residue-to-Binary Conversion Based on a Modulo-Adder-Free Architecture

Energy-Efficient Residue-to-Binary Conversion Based on a Modulo-Adder-Free Architecture

Kamalaldin Mozaffari Majd
Department of Computer Engineering
Azad University of Kerman
Kerman, Iran

Amir Sabbagh Molahosseini
EEECS School
Queen's University Belfast
Belfast, UK

***Abstract*— This paper proposes a novel residue-to-binary converter for residue number system (RNS) based on the moduli set $\{2^{n+k}, 2^{2n+1}-1, 2^n+1, 2^n-1\}$. By adopting the new Chinese Remainder Theorem II (CRT II) and the properties of modulo 2^k-1 arithmetic, the research herein presented shows the possibility of designing RNS residue-to-binary converters free of modular adders. By using exclusively regular binary adders, one can leverage the design of efficient residue-to-binary converters based on the large range of optimizations proposed to those regular arithmetic units. Experimental results show over 37% energy-improvement in comparison to conventional designs, which make use of modulo adders.**

Keywords—Residue Number System (RNS), Residue-to-Binary Converter, Modulo Adder, New CRT-II.

I. INTRODUCTION

The Residue Number System (RNS) [1] has been one of the most promising unconventional number systems for high-speed realization of arithmetic circuits, useful to applications requiring multiple additions and multiplications [2]. Well known applications of RNS are digital signal processing (DSP) [3], cryptography [4], and deep learning [5]

The RNS based platforms include three fundamental units, namely a binary-to-residue (forward) converter, modular arithmetic units and a residue-to-binary (reverse) converter. Forward and reverse converters transform binary to residue representations and vice versa, respectively. The forward converter typically contains parallel multi-operand modular adders, which can be efficiently realized using carry-save adders (CSAs). However, reverse converter requires several computational parts that lead to time-consuming and power-hungry structures, particularly for arithmetic-friendly moduli sets [6]. Thus, efficient RNS reverse converter design has been a big challenge in computer arithmetic.

For the first time, [7] introduced the usage of CSAs in reverse converters, but a final modular carry propagate adder is required. These modular carry propagate adders impose high delay and hardware requirements. Therefore, researchers have been investigated methods and techniques to mitigate this negative impact avoiding as much as it is possible modular adders in reverse converter architectures. On the other hand, the new Chinese Remainder Theorems (New CRTs) have allowed to simplify reverse conversion formulation reaching more efficient hardware implementations [8-12]. Besides, mixed-radix conversion (MRC) is another useful approach for designing reverse converters [1, 2]. The features and number of modulo in the set define the conversion approach, i.e. new CRT or MRC, which achieves most simplification and consequently better hardware implementation. Another research line in reverse converter design is hardware-based optimization. In other words, instead of changing the moduli sets, or using different conversion approaches, optimized hardware components dedicated for designing reverse converter are applied. One of the first works in this direction was [11]. The customized modulo adders to be used in signed reverse converter architectures are proposed in [11]. Moreover, as shown in [8], a reverse converter for conversion-friendly moduli sets only requires one large modular adder while arithmetic-friendly moduli sets such as $\{2^n, 2^{2n+1}-1, 2^n+1, 2^n-1\}$ requires several modular adders, leading to reverse converters with high delay and circuit area. But faster internal arithmetic operations in arithmetic-friendly moduli sets, such as $\{2^n, 2^{2n+1}-1, 2^n+1, 2^n-1\}$, than conversion-friendly moduli sets, such as $\{2^{2n}, 2^{2n}+1, 2^n+1, 2^n-1\}$, justifies the cost of reverse converters for arithmetic-friendly moduli sets.

This paper aims to optimize the performance of reverse converters for the arithmetic-friendly moduli set $\{2^{n+k}, 2^{2n+1}-1, 2^n+1, 2^n-1\}$, by avoiding the costliest unit, modulo 2^k-1 adders. The proposed novel reverse converter is completely free of modulo 2^k-1 adders, only uses CSAs and regular binary carry-propagate adders (CPAs). This results in significant improvements in all circuit's parameters, including delay, circuit area and power-consumption. In the remaining of this paper, a brief review of RNS is described in Section II. Section III presents the proposed reverse conversion formulas, and Section IV presents its hardware architecture, and evaluates its performance. Finally, Section V concludes the paper.

II. RESIDUE NUMBER SYSTEM

RNS relies on a set composed by N moduli pairwise relatively prime integers. The moduli set $\{m_1, m_2, \cdots, m_N\}$ setup a range of $M = \prod_{i=1}^{n} m_i$ of numbers that can be represented by a unique tuple $\{x_1, x_2, \cdots, x_N\}$. The residue x_i is the remainder of the division of X by m_i, usually represented as $|X|_{m_i}$ [1, 2].

The New CRT-II [8] applied to a general four-moduli set $\{m_1, m_2, m_3, m_4\}$ results in the following formulation for the transforming the residues (x_1, x_2, x_3, x_4) to the equivalent weighted number:

$$X = Z + m_1 m_2 |k_1 (Y - Z)|_{m_3 m_4} \quad (1)$$

$$Z = x_1 + m_1 |k_2 (x_2 - x_1)|_{m_2} \quad (2)$$

$$Y = x_3 + m_3 |k_3 (x_4 - x_3)|_{m_4} \quad (3)$$

where the multiplicative inverses k_i can be achieved for the 4-moduli set as follows:

$$|k_1 m_1 m_2|_{m_3 m_4} = 1 \quad (4)$$

$$|k_2 m_1|_{m_2} = 1 \quad (5)$$

$$|k_3 m_3|_{m_4} = 1 \quad (6)$$

III. The Proposed Formulation for Reverse Conversion

Three modulo 2^k-1 adders with different sizes are required in the reverse converter of [8], that two of them are in the critical delay path. Therefore, first, the well-known fundamental modulo 2^n-1 addition formulas are reviewed (*Lemma 1*), and then the proposed formulation based on it (*Lemma 2*), will be provided.

Lemma 1: The modulo 2^n-1 addition of two *n*-bit operands *A* and *B* can be performed as follows [13, 14]:

$$|A+B|_{2^n-1}=\begin{cases}|A+B+1|_{2^n} & if\ \ A+B\geq 2^n-1\\ A+B & if\ \ A+B<2^n-1\end{cases} \quad (7)$$

Alternatively, (7) can be rewritten by enabling two-representation of zero's in output as follows [18]:

$$|A+B|_{2^n-1}=\begin{cases}|A+B+1|_{2^n} & if\ \ A+B\geq 2^n\\ A+B & if\ \ A+B<2^n\end{cases}$$
$$=|A+B+c_{out}|_{2^n} \quad (8)$$

where c_{out} is the carry output of *A*+*B*. But, modulo adders should have single-representation of zero in order to be used in reverse converter. Therefore, usually series of two-input AND gates are usually used at the output of the modulo adders with double-representation zero, in order to correct the output.

Lemma 2: The modulo 2^n-1 addition of two *n*-bit operands *A* and *B* can be performed as follows:

$$|A+B|_{2^n-1}=S-\overline{c_{out}} \quad (9)$$

where *S* and c_{out} are the regular summation result and carry output of the operation *A*+*B*+1, respectively.

Proof: First, (7) can be rewritten as follows:

$$|A+B|_{2^n-1}=\begin{cases}|A+B+1|_{2^n} & if\ \ A+B+1\geq 2^n\\ A+B+1-1 & if\ \ A+B+1<2^n\end{cases} \quad (10)$$

On the other hand, the carry out of the operation *A*+*B*+1 will be 1 if *A*+*B*+1 is equal or greater than 2^n. Therefore, (10) can be rewritten as:

$$|A+B|_{2^n-1}=\begin{cases}|A+B+1-\overline{c_{out}}|_{2^n} & if\ \ A+B+1\geq 2^n\\ A+B+1-\overline{c_{out}} & if\ \ A+B+1<2^n\end{cases} \quad (11)$$

Therefore, (11) can be rewritten in a single equation as follows [15, 16]:

$$|A+B|_{2^n-1}=|A+B+1-\overline{c_{out}}|_{2^n}=|S-\overline{c_{out}}|_{2^n} \quad (12)$$

where *S* and c_{out} are the regular summation result and carry output of the operation *A*+*B*+1, respectively. In other words, carry output is separated from summation results, and *S* is just represented in *n* bits, that is always less than 2^n. Therefore, if *A*+*B*+1 is equal or greater than 2^n, then c_{out} will be one, and $S-\overline{c_{out}}$ becomes *S*-0. On the other hand, if *A*+*B*+1 is less than 2^n, then c_{out} will be zero, and $S-\overline{c_{out}}$ becomes *S*-1. In both of these cases, $S-\overline{c_{out}}$ is always less than 2^n, and therefore we can rewrite (12) as follows:

$$|A+B|_{2^n-1}=|S-\overline{c_{out}}|_{2^n}=S-\overline{c_{out}} \quad (13)$$

QED

The *Lemma 2* is used for designing the proposed modulo adder-free reverse converter. This lemma has two important features: *i*) it does not require any modulo addition, and *ii*) it relies on single representation of zero addition result due to considering *A*+*B*+1 as the main operation for achieving the carry output.

The multiplicative inverses k_i can be achieved for the 4-moduli set $\{2^{n+k}, 2^{2n+1}\text{-}1, 2^n\text{+}1, 2^n\text{-}1\}$ as follows:

$$|k_1m_1m_2|_{m_3m_4}=1\rightarrow\ |k_1 2^{\,n+k}(2^{2n+1}-1)|_{2^{2n}-1}=1$$
$$k_1=2^{n-k} \quad (14)$$
$$|k_2m_1|_{m_2}=1\ \rightarrow|k_2 2^{\,n+k}|_{2^{2n+1}-1}=1\rightarrow$$
$$k_2=2^{n-k+1} \quad (15)$$
$$|k_3m_3|_{m_4}=1\ \rightarrow|k_3(2^n+1)|_{2^n-1}=1\ \rightarrow$$
$$k_3=2^{n-1} \quad (16)$$

Let us apply (14)-(16) to the 4-moduli set $\{2^{n+k}, 2^{2n+1}\text{-}1, 2^n\text{+}1, 2^n\text{-}1\}$, the starting by the bit-level representation of the residues:

$$x_1=\underbrace{x_{1,n+k-1}\dots x_{1,0}}_{n+k\ bits} \quad (17)$$
$$x_2=\underbrace{x_{2n}\dots x_{2,0}}_{2n+1\ bits} \quad (18)$$
$$x_3=\underbrace{x_{3,n}\dots x_{3,0}}_{n+1\ bits} \quad (19)$$
$$x_4=\underbrace{x_{4,n-1}\dots x_{4,0}}_{n\ bits} \quad (20)$$
$$X=Z+2^{n+k}(2^{2n+1}-1)|2^{n-k}(Y-Z)|_{2^{2n}-1}=Z+2^{n+k}(2^{2n+1}-1)T \quad (21)$$

where

$$Z=x_1+2^{n+k}|2^{n-k+1}(x_2-x_1)|_{(2^{2n+1}-1)}$$
$$=x_1+2^{n+k}H \quad (22)$$
$$Y=x_3+(2^n+1)|2^{n-1}(x_4-x_3)|_{(2^n-1)}$$
$$=x_3+(2^n+1)K \quad (23)$$
$$T=|2^{n-k}(Y-Z)|_{2^{2n}-1} \quad (24)$$
$$K=|2^{n-1}(x_4-x_3)|_{(2^n-1)} \quad (25)$$
$$|2^{n-1}|_{(2^n-1)}=2^{-1} \quad (26)$$
$$K=|2^{-1}x_4-2^{-1}x_3|_{(2^n-1)} \quad (27)$$
$$K=|v_4+v_3|_{2^n-1} \quad (28)$$

Where

$$v_3=\left|-2^{-1}\underbrace{(x_{3,n}\dots x_{3,0})}_{n+1}\right|_{2^n-1}=\left|-2^{-1}(00\dots0x_{3,n}+x_{3,n-1}\dots x_{3,0})\right|_{2^n-1}=\left|\bar{x}_{3,n}11\dots1+\bar{x}_{3,0}\bar{x}_{3,n-1}\dots\bar{x}_{3,1}\right|_{2^n-1}$$
$$v_3=\begin{cases}011\dots1 & x_{3,n}=1\\ \bar{x}_{3,0}\bar{x}_{3,n-1}\dots\bar{x}_{3,1} & x_{3,n}=0\end{cases}$$
$$v_3=\left|(\bar{x}_{3,n}\ \&\ \bar{x}_{3,0})\bar{x}_{3,n-1}\dots\bar{x}_{3,1}\right|_{2^n-1} \quad (29)$$

Where most significant bit of v_3 is $\bar{x}_{3,n}\ and\ \bar{x}_{3,0}$.

$$v_4=|2^{-1}x_4|_{2^n-1}=\left|\underbrace{x_{4,0}x_{4,n-1}\dots x_{4,0}x_{4,1}}_{n\ bits}\right|_{2^n-1} \quad (30)$$

$$Z=x_1+2^{n+k}H \quad (31)$$
$$H=|2^{n-k+1}(x_2-x_1)|_{(2^{2n+1}-1)}=|2^{n-k+1}(x_2-x_1)|_{(2^{2n+1}-1)}=|2^{n-k+1}x_2-2^{n-k+1}x_1)|_{(2^{2n+1}-1)} \quad (32)$$

$$H=|v_2+v_1|_{2^{2n+1}-1} \quad (33)$$
$$v_1=|-2^{n-k+1}x_1|_{2^{2n+1}-1}=$$
$$\left|-2^{n-k+1}(\underbrace{00\dots0}_{n-k+1}\underbrace{x_{1,n+k-1}\dots x_{1,0}}_{n+k})\right|_{2^{2n+1}-1}=$$

$$\left|\underbrace{\bar{x}_{1,n+k-1}\dots\bar{x}_{1,0}}_{n+k}\ \underbrace{11\dots1}_{n-k+1}\right|_{(2^{2n+1}-1)} \quad (34)$$

$$v_2 = |2^{n-k+1}x_2|_{2^{2n+1}-1} = \left|2^{n-k+1}(x_{2,2n}\dots x_{2,0})\right|_{2^{2n+1}-1} = \left|x_{2,n+k-1}\dots x_{2,0}x_{2,2n}\dots x_{2,n+k}\right|_{2^{2n+1}-1} \quad (35)$$

Now, we apply *Lemma 2* to the reverse conversion formulas to derive modulo adder free reverse conversion structure. First, (33) can be rewritten as follows based on (8)-(12):

$$H = |v_1 + v_2|_{2^{2n+1}-1} = |v_1 + v_2 + 1 - \overline{c_H}|_{2^{2n+1}} = \acute{H} - \overline{c_H} \quad (36)$$

where $\acute{H}$ and c_H are the regular summation result and carry output of the operation $v_1 + v_2 + 1$, respectively. Note that v_1 and v_2 are presented in (34) and (35). Now, by substituting the bit-level representation of v_1 and v_2 from [12] to (36), we have:

$$H = \underbrace{v_1 + v_2 + 1}_{\acute{H}} - c_H = \underbrace{\bar{x}_{1,n+k-1}\dots\bar{x}_{1,0}}_{n+k}\ \underbrace{11\dots1}_{n-k+1} + \underbrace{x_{2,n+k-1}\dots x_{2,0}}_{n+k}\ \underbrace{x_{2,2n}\dots x_{2,n+k}}_{n-k+1} + 1 - \bar{c}_H \quad (37)$$

The (37) can be rewritten as follows:

$$H = \underbrace{\bar{x}_{1,n+k-1}\dots\bar{x}_{1,0}}_{n+k}\ \underbrace{00\dots0}_{n-k+1} + \underbrace{x_{2,n+k-1}\dots x_{2,0}}_{n+k}\ \underbrace{x_{2,2n}\dots x_{2,n+k}}_{n-k+1} + \underbrace{00\dots0}_{n-1\ bits}\ 1\ \underbrace{00\dots0}_{n+1\ bits} - \bar{c}_H \quad (38)$$

Now, using some bit manipulations and factorizing, we have the following formula instead of (38):

$$\acute{H} = 2^{n-k+1}Q + \underbrace{x_{2,2n}\dots x_{2,n+k}}_{n-k+1}$$

$$v'_{21} = \underbrace{x_{2,2n}\dots x_{2,n+k}}_{n-k+1}$$

$$\acute{H} = Q\&v'_{21} \quad (39)$$

where & denotes concatenation, and Q assumes the value:

$$Q = \underbrace{x_{2,n+k-1}\dots x_{2,0}}_{n+k} + \underbrace{\bar{x}_{1,n+k-1}\dots\bar{x}_{1,0}}_{n+k} + 1$$

$$v'_1 = x_{2,n+k-1}\dots x_{2,0}$$

$$v'_{22} = \bar{x}_{1,n+k-1}\dots\bar{x}_{1,0}$$

$$Q = v'_1 + v'_{22} + 1 \quad (40)$$

Note that c_H is the carry-output produced from addition operation in (40). Now, by substituting (36) in (31), we have:

$$Z = x_1 + 2^{n+k}(\acute{H} - \overline{c_H}) = x_1 + 2^{n+k}\acute{H} - 2^{n+k}\overline{c_H} \quad (41)$$

This formula can be rewritten as follows:

$$Y = x_3 + (2^n + 1)K = x_3 + (2^n + 1)|v_3 + v_4|_{2^n-1} = x_3 + |(2^n + 1)(v_3 + v_4)|_{2^{2n}-1} \quad (42)$$

The (42) is achieved based on well-known residue arithmetic relation $k|A|_P = |k \times A|_{kP}$ [20]. Third, substituting (41) and (42) in (24) results in:

$$T = |2^{n-k}(Y - Z)|_{2^{2n}-1} = \left|2^{n-k}((x_3 + |(2^n + 1)(v_3 + v_4)|_{2^{2n}-1}) - (x_1 + 2^n\acute{H} - 2^n\overline{c_H}))\right|_{2^{2n}-1} \quad (43)$$

(43) becomes:

$$T = |2^{n-k}(Y - Z)|_{2^{2n}-1} = \left|2^{n-k}(x_3 + (2^n + 1)(v_3 + v_4) - x_1 - 2^{n+k}\acute{H} + 2^{n+k}\overline{c_H})\right|_{2^{2n}-1}$$

$$T = \left|2^{n-k}x_3 + 2^{n-k}(2^n + 1)(v_3 + v_4) - 2^{n-k}x_1 - \acute{H} + \overline{c_H}\right|_{2^{2n}-1} = |v_5 + v_6 + v_7 + v_8 + v_9 + \overline{c_H}|_{2^{2n}-1} \quad (44)$$

Therefore, the modulo addition needed in (42) has been removed. Now, similar to [8], the multiplications required in (44) can be simplified using residue arithmetic formulas [8, Properties 1 and 2] as follows:

$$v_5 = \left|2^{n-k}x_3\right|_{2^{2n}-1} = \left|2^{n-k}(\underbrace{00\dots0}_{n-1}\ \underbrace{x_{3,n}\dots x_{3,0}}_{n+1})\right|_{2^{2n}-1} = \begin{cases} \underbrace{x_{3,n-k-1}\dots x_{3,0}}_{n-k\ bits}\ \underbrace{00\dots0}_{n-1\ bits}\ \underbrace{x_{3,n}\dots x_{3,n-k}}_{k+1} & k < 0 \\ \underbrace{x_{3,n-1}\dots x_{3,0}}_{n\ bits}\ \underbrace{00\dots0}_{n-1\ bits}\ x_{3,n} & k = 0 \\ \underbrace{00\dots0}_{k-1}\ \underbrace{x_{3,n}\dots x_{3,0}}_{n+1}\ \underbrace{00\dots0}_{n-k} & n \geq k \geq 1 \end{cases} \quad (45)$$

$$v_6 = |2^{n-k}(2^n + 1)v_3|_{2^{2n}-1} = \underbrace{v_{3,k-1}\dots v_{3,0}}_{k\ bits}\ \underbrace{v_{3,n-1}\dots v_{3,0}}_{n\ bits}\ \underbrace{v_{3,n-1}\dots v_{3,k}}_{n-k\ bits} \quad (46)$$

$$v_7 = |2^{n-k}(2^n + 1)v_4|_{2^{2n}-1} = \underbrace{v_{4,k-1}\dots v_{4,0}}_{k\ bits}\ \underbrace{v_{4,n-1}\dots v_{4,0}}_{n\ bits}\ \underbrace{v_{4,n-1}\dots v_{4,k}}_{k\ bits} \quad (47)$$

$$v_8 = \left|-2^{n-k}x_1\right|_{2^{2n}-1} = \left|-2^{n-k}(\underbrace{00\dots0}_{n-k}\ \underbrace{x_{1,n+k-1}\dots x_{1,0}}_{n+k})\right|_{2^{2n}-1} = \left|\underbrace{\bar{x}_{1,n+k-1}\dots\bar{x}_{1,0}}_{n+k}\ \underbrace{11\dots1}_{n-k}\right|_{2^{2n}-1} \quad (48)$$

$$v_9 = \left|-\acute{H}\right|_{2^{2n}-1} = \left|-(\acute{H}_{2n} \times 2^{2n} + \underbrace{\acute{H}_{2n-1}\dots\acute{H}_0}_{2n\ bits})\right|_{2^{2n}-1} = \left|\underbrace{11\dots1}_{2n-1\ bits}\ \overline{\acute{H}_{2n}} + \underbrace{\overline{\acute{H}_{2n-1}}\dots\overline{\acute{H}_0}}_{2n\ bits}\right|_{2^{2n}-1} \quad (49)$$

Now, applying *Lemma 2*, i.e. (9) to (44), results in:

$$T = |v_5 + v_6 + v_7 + v_8 + v_9 + \overline{c_H}|_{2^{2n}-1} = |v_5 + v_6 + v_7 + v_8 + v_9 + 1 + \overline{c_H} - \overline{c_T}|_{2^{2n}} \quad (50)$$

Next, (50) can be rewritten based on (13) to remove the required modulo addition as follows:

$$T = |v_5 + v_6 + v_7 + v_8 + v_9 + 1 + \overline{c_H} - \overline{c_T}|_{2^{2n}} = \acute{T} - \overline{c_T} \quad (51)$$

where

$$\acute{T} = v_5 + v_6 + v_7 + v_8 + v_9 + 1 + \overline{c_H} \quad (52)$$

Note that c_T is carry output result of the addition of $\acute{T}$. Finally, substituting (41) and (52) in (21) that is the main conversion formula, results in:

$$X = Z + 2^{n+k}(2^{2n+1} - 1)T = x_1 + 2^{n+k}\acute{H} - 2^{n+k}\overline{c_H} + 2^{n+k}(2^{2n+1} - 1)(\acute{T} - \overline{c_T}) \quad (53)$$

The (53) can be rewritten using some factorizations as follows:

$$X = x_1 + 2^{n+k}S = S\ \&\ x_1 \quad (54)$$

Where

$$S = \acute{H} + (2^{2n+1} - 1)\acute{T} - (2^{2n+1} - 1)\overline{c_T} - \overline{c_H} \quad (55)$$

Now, (55) can be calculated using following operations:

$$S = s_1 + s_2 + s_3 + s_4 + s_5 + \overline{c_T} \quad (56)$$

Where

$$s_1 = \acute{H} = \underbrace{00\ldots0}_{2n\ bits}\ \underbrace{\acute{H}_{2n}\ldots\acute{H}_0}_{2n+1\ bits} \quad (57)$$

$$s_2 = 2^{2n+1}\acute{T} = 2^{2n+1}(\underbrace{00\ldots0}_{2n+1\ bits}\ \underbrace{\acute{T}_{2n-1}\ldots\acute{T}_0}_{2n\ bits}) =$$

$$\underbrace{\acute{T}_{2n-1}\ldots\acute{T}_0}_{2n\ bits}\ \underbrace{00\ldots0}_{2n+1\ bits} \quad (58)$$

$$s_3 = -\acute{T} = -\left(\underbrace{00\ldots0}_{2n+1\ bits}\ \underbrace{\acute{T}_{2n-1}\ldots\acute{T}_0}_{2n\ bits}\right) =$$

$$\underbrace{11\ldots1}_{2n+1\ bits}\ \underbrace{\overline{\acute{T}_{2n-1}}\ldots\overline{\acute{T}_0}}_{2n\ bits} + 1 \quad (59)$$

$$s_4 = -2^{2n+1}\overline{c_T} = -2^{2n+1}\left(\underbrace{00\ldots0}_{4n\ bits}\ \overline{c_T}\right) =$$

$$-\left(\underbrace{00\ldots0}_{2n-1\ bits}\ \overline{c_T}\ \underbrace{00\ldots0}_{2n+1\ bits}\right) = \underbrace{11\ldots1}_{2n-1\ bits}\ c_T\ \underbrace{11\ldots1}_{2n+1\ bits} + 1 \quad (60)$$

$$s_5 = -\overline{c_H} \quad (61)$$

Consecutive 0's or 1's in s_i's enable combinations to reduce the number of addition operands. First, (57) and (58) can be simply added without any hardware using concatenation as:

$$s_{12} = s_1 + s_2 = \underbrace{\acute{T}_{2n-1}\ldots\acute{T}_0}_{2n\ bits}\ \underbrace{\acute{H}_{2n}\ldots\acute{H}_0}_{2n+1\ bits} \quad (62)$$

Moreover, (59) to (61) can be combined as follows:

$$s_{345} = s_3 + s_4 + s_5 = \underbrace{11\ldots1}_{2n+1\ bits}\ \underbrace{\overline{\acute{T}_{2n-1}}\ldots\overline{\acute{T}_0}}_{2n\ bits} + 1 +$$

$$\underbrace{11\ldots1}_{2n-1\ bits}\ c_T\ \underbrace{11\ldots1}_{2n+1\ bits} + 1 - \overline{c_H} \quad (63)$$

Now, (63) can be computed using the following relation:

$$s_{345} = s_3 + s_4 + s_5 = \underbrace{11\ldots1}_{2n-1\ bits}\ c_T 1\ \underbrace{\overline{\acute{T}_{2n-1}}\ldots\overline{\acute{T}_0}}_{2n\ bits} + c_H \quad (64)$$

Therefore, (56) becomes:

$$S = s_{12} + s_{345} + \overline{c_T} = \underbrace{\acute{T}_{2n-1}\ldots\acute{T}_0}_{2n\ bits}\ \underbrace{\acute{H}_{2n}\ldots\acute{H}_0}_{2n+1\ bits} +$$

$$\underbrace{11\ldots1}_{2n-1\ bits}\ c_T 1\ \underbrace{\overline{\acute{T}_{2n-1}}\ldots\overline{\acute{T}_0}}_{2n\ bits} + \underbrace{00\ldots0}_{4n\ bits}\ \overline{c_T} + c_H \quad (65)$$

IV. Proposed Architecture and Experimental Results

The hardware architecture of the proposed modulo-adder-free reverse converter for the moduli set $\{2^{n+k}, 2^{2n+1}\text{-}1, 2^n\text{+}1, 2^n\text{-}1\}$ is depicted in Fig. 1. Only CSAs and regular binary adders, i.e. CPAs, are used to implement the converter. First, (40) is implemented using an n-bit CPA to produce Q and c_H. Note that by concatenating Q and $v'_{2,1}$ according to (39), $\acute{H}$ can be achieved. Next, (52) should be computed using three or four $2n$-bit CSAs with end-around carries (EACs) which is depended on k and followed by a regular $2n$-bit CPA to produce $\acute{T}$ and c_T. It should be mentioned that $\overline{c_H}$ which is required in (52), is entered as carry-in to the CPA2. Finally, the produced values of $\acute{H}$ and $\acute{T}$ are used to implement (65) using a $(4n+1)$-bit CSA followed by a $(4n+1)$-bit regular CPA. Note that $4n$ full adders (FAs) of this CSA are reduced to 4n half adders (HAs), since the third operand of CSA have $4n$ 0's according to (65). Besides, the c_H is entered to the last CPA as carry in. Lastly, the concatenation of x_1 with the output of the last CPA forms the final output.

To evaluate the performance of the proposed converter rather than [8], both converters are described in VHDL codes, and verified using ModelSim. Then, they are implemented in application-specific integrated circuits (ASIC) using CMOS 65nm technology. The experimental results are presented in Tables I and II. It can be seen from these Tables that the proposed converter outperforms the converter of [8] in all circuit's parameters. Particularly, the proposed converter results in energy-consumption reduction than [8] about 52.59%, 54.60%, 39.16% and 37.97% for n=4, 8, 12, and 16, respectively. Because [8] requires three modulo adders with a final binary adder; while the proposed design requires three binary adders.

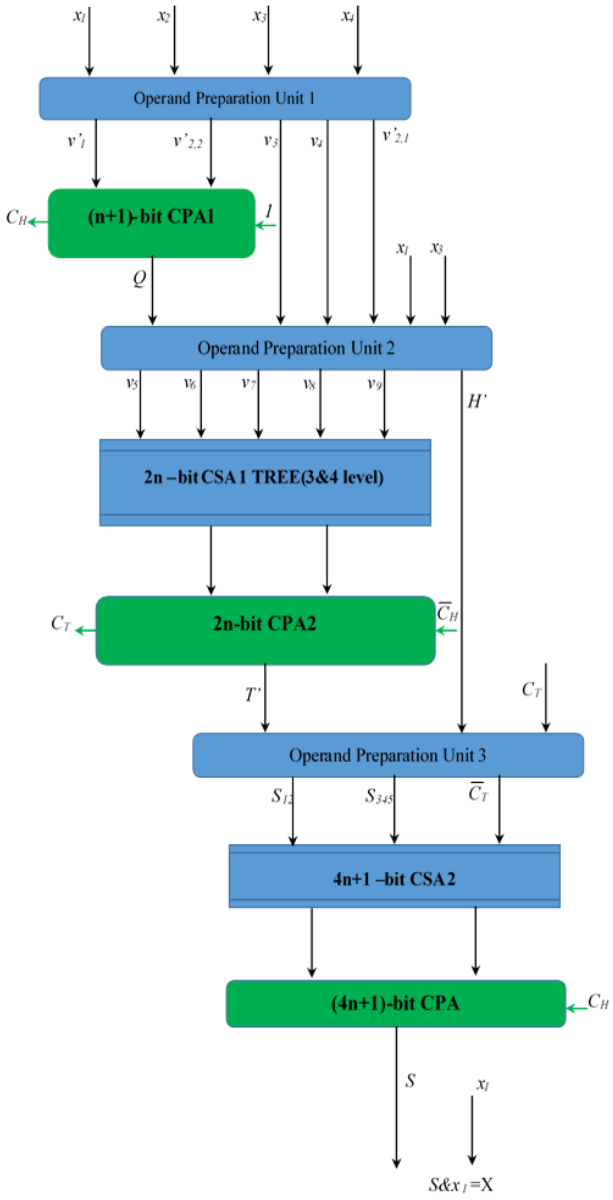

Fig. 1. The proposed modulo-adder-free reverse converter

TABLE I. Performance Comparsion: Delay and Area (ASICimplementation)

n	Delay (ns)		Chip Area (μm²)	
	Proposed	**[8]**	**Proposed**	**[8]**
4	1.74	1.99	2637	3969
8	2.922	3.458	5417	8667
12	4.001	4.873	8207	12187
16	4.963	6.265	10855	15975

TABLE II. PERFORMANCE COMPARSION: POWER AND ENERGY (ASIC IMPLEMENTATION)

n	Power-consumption (mW)		Energy-consumption (pJ)	
	Proposed	**[8]**	**Proposed**	**[8]**
4	0.8426	1.554	1.466124	3.09246
8	1.044	1.943	3.050568	6.718894
12	1.353	1.826	5.413353	8.898098
16	1.541	1.968	7.647983	12.32952

In addition to the ASIC implementation, Field-Programmable Gate Arrays (FPGA) implementation (Vitex7-xc7vx330t-3-ffg1157) of the various reverse converters for the moduli set $\{2^{n+k}, 2^{2n+1}-1, 2^n+1, 2^n-1\}$, where k=-1,0,n have been presented in Tables III and IV in terms of number of look-up tables (LUTs) and delay (ns). It can be seen that the proposed design outperforms all of state-of-the art designs in term of number of required LUTs. Besides, it has higher speed than [8-10].

TABLE III. PERFORMANCE COMPARSION : DELAY (FPGA IMPLEMENTATION)

n	**Delay (ns)**						
	***k* = -1**		***k* = 0**			***k* = *n***	
	Proposed	**[10]**	**Proposed**	**[8]**	**[9]**	**Proposed**	**[9]**
4	3.701	6.85	3.913	6.324	4.286	3.913	5.377
8	4.467	9.123	4.656	8.422	5.908	4.656	7.449
16	5.563	11.13	5.576	10.53	7.092	5.576	9.996

TABLE IV. PERFORMANCE COMPARSION: AREA (FPGA MPLEMENTATION)

n	**Area (Number of LUTs)**						
	***k* = -1**		***k* = 0**			***k* = *n***	
	Proposed	**[10]**	**Proposed**	**[8]**	**[9]**	**Proposed**	**[9]**
4	187	342	209	402	319	209	237
8	433	618	397	753	672	397	578
16	905	1353	933	1540	1600	933	1310

V. CONCLUSIONS

This paper proposed a novel modulo-adder-free architecture to design reverse converters for the arithmetic-friendly moduli set $\{2^{n+k}, 2^{2n+1}-1, 2^n+1, 2^n-1\}$. Experimental evaluation shows that the proposed converter can significantly improve area, delay and power-consumption of the reverse converter. It results in overhead reduction of the reverse converter for using in emerging technologies such as deep learning circuits and systems where energy-efficient implementation of convolution is important for embedded and mobile devices. Moreover, the proposed approach can be used for modulo-adder-free implementation of difficult RNS operations such as sign-detection, magnitude comparison and scaling.

REFERENCES

[1] P.V.A. Mohan, *Residue Number Systems: Theory and Applications*, Basel: Springer Birkhäuser Basel, 2016.

[2] A.S. Molahosseini, L. Sousa and C.H. Chang, *Embedded Systems Design with Special Arithmetic and Number Systems*, New York: Springer Science, 2017.

[3] C.H. Chang, A.S. Molahosseini, A.A. Emrani Zarandi, and T.F. Tay, "Residue Number Systems: A New Paradigm to Datapath Optimization for Low-Power and High-Performance Digital Signal Processing Applications," *IEEE Circuits and Systems Magazine*, vol. 15, no. 4, pp. 26-44, 2015.

[4] L. Sousa, S. Antão, and P. Martins, "Combining Residue Arithmetic to Design Efficient Cryptographic Circuits and Systems," *IEEE Circuits and Systems Magazine*, vol. 16, no. 4, pp. 6-32, 2016.

[5] V. Arrigoni, B. Rossi, P. Fragneto and G. Desoli, "Approximate operations in Convolutional Neural Networks with RNS data representation," In Proc. of . European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 2017.

[6] M.-H. Sheu, Y.-C. Kuo, S.-H. Lin, and S.-M. Siao, "Efficient reverse converter design for new adaptable four-moduli set $\{2^{n+k}, 2^n+1, 2^n-1, 2^{2n}+1\}$," IEICE Trans. Fundam. Electron., Commun. Comput. Sci., vol. E96-A, no. 7, pp. 1571–1578, 2013.

[7] S.J. Piestrak, "A high speed realization of a residue to binary converter," *IEEE Trans. Circuits and Systems-II*, vol. 42, pp. 661-663, 1995.

[8] A.S. Molahosseini, K. Navi, C. Dadkhah, O. Kavehei, 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 and Systems-I*, vol. 57, no. 4, pp. 823-835, 2010.

[9] L. Sousa and S. Antao, "MRC-Based RNS Reverse Converters for the Four-Moduli Sets $\{2^n+1, 2^n-1, 2^n, 2^{2n+1}-1\}$ and $\{2^n+1, 2^n-1, 2^{2n}, 2^{2n+1}-1\}$," *IEEE Transactions on Circuits and Systems II*, vol. 59, no. 4, pp. 244-248, 2012.

[10] R. K. Jaiswal, R. Kumar and R.A. Mishra. "Area Efficient Memoryless Reverse Converter for New Four Moduli Set $\{2^{n-1}, 2^n-1, 2^n+1, 2^{2n+1}-1\}$," *Journal of Circuits, Systems and Computers*, vol. 27, no. 5, pp. 1850075, 2018.

[11] A.A.E. Zarandi, A.S. Molahosseini, M. Hosseinzadeh, S. Sorouri, S.F. Antão and L. Sousa, "Reverse Converter Design via Parallel-Prefix Adders: Novel Components, Methodology and Implementations," *IEEE Transactions on Very Large Scale Integration (VLSI) systems*, vol. 2, no. 374-378, p. 23, 2015.

[12] A Hiasat, "A residue-to-binary converter for the extended four-moduli set $\{2^n-1, 2^n+1, 2^{2n}+1, 2^{2n+p}\}$", *IEEE Transactions on Very Large Scale Integration (VLSI) systems*, vol. 25, no. 7, pp. 2188–2192, 2017.

[13] R. Zimmermann, "Efficient VLSI implementation of modulo $(2^n \pm 1)$ addition and multiplication," In Proc. of IEEE Symposium on Computer Arithmetic, Adelaide, SA, Australia, 1999, pp. 158-167.

[14] J.L. Beuchat, "Some modular adders and multipliers for field programmable gate arrays," In Proc. of International Parallel and Distributed Processing Symposium, Nice, France, 2003, pp. 1-8.

[15] I. Krstic, N. Stamenkovic, M. Petrovic and V. Stojanovic, "Binary to RNS encoder with Modulo 2^n+1 Channel in Diminished-1 Number System," *International Journal of Computational Engineering & Management*, vol. 17, no. 3, pp. 1-9, 2014.

[16] K.H. Rosen, *Elementary Number Theory and Its Application*, Addison-Wesley, 1988.