International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:3, 2007

A Fully Parallel Reverse Converter

Mehdi Hosseinzadeh, Amir Sabbagh Molahosseini, Keivan Navi

***Abstract*—**The residue number system (RNS) is popular in high performance computation applications because of its carry-free nature. The challenges of RNS systems design lie in the moduli set selection and in the reverse conversion from residue representation to weighted representation. In this paper, we proposed a fully parallel reverse conversion algorithm for the moduli set $\{r^n - 2, r^n - 1, r^n\}$, based on simple mathematical relationships. Also an efficient hardware realization of this algorithm is presented. Our proposed converter is very faster and results to hardware savings, compared to the other reverse converters.

***Keywords*—**Reverse converter, residue to weighted converter, residue number system, multiple-valued logic, computer arithmetic.

I. Introduction

THE conventional arithmetic carries propagation based on a weighted number system is the reason for performance degradation in hardware computing systems. The residue number system is a non-weighted number system which speeds up arithmetic operations by dividing them into smaller parallel operations [1],[2]. Since the arithmetic operations in each moduli are independent of the others, there is no carry propagation among them and so RNS leads to carry-free addition, multiplication and borrow-free subtraction [3]. RNS is one of the most popular techniques for reducing the power dissipation and the computation load in VLSI systems design [4]. Some applications of the RNS are real-time processing, digital filters [5],[7], digital signal processing (DSP) [8],[9], the RSA encoding algorithm [12] and digital communication [13]. The architecture of the RNS is naturally fault tolerant and consequently, it is used to error detection, error correction and fault tolerance [15].

Despite binary logic in which logical levels are restricted to two possible states, there exists an alternative named multiple-valued logic. In MVL, the number of discrete signal values or logic states extends beyond two. Arithmetic units implemented with MVL achieve more efficient use of silicon resource and circuit interconnections [20]. There is a clear mathematical attraction of using multiple-valued number representation in RNS. The modular arithmetic that is inherent in MVL can be match with modular arithmetic needed in RNS.

The concept of MVL-RNS was introduced by Soderstrand et al. [17] to design a high speed FIR digital filter. The reverse converter proposed in [17] is based on chinese remainder theorem (CRT) and implemented with read-only memories (ROM's). This converter is practical to implement small and medium RNS dynamic ranges and it is not appropriate for large dynamic ranges. In [18], new RNS systems based on the moduli of forms r^a, r^b–1 and r^c+1 are presented. Abdallah et al. in [18] developed a systematic framework utilizing high-radix arithmetic for efficient MVL-RNS implementations and proposed many radix-r moduli sets. This moduli sets are not pairwise relatively prime that resulting in reduced dynamic ranges and unbalanced moduli. The reverse converter presented in [18] is based on CRT and because of the scale-down factors that used to makes moduli pairwise relatively prime, conversion delay and cost are increased. In [19], a new moduli set $\{r^n - 2, r^n - 1, r^n\}$ where $r=2k+1$, $k=1,2,\ldots$ for MVL-RNS was proposed. This moduli set includes pairwise relatively prime and balanced moduli that offers large dynamic range and simple realization of related circuits. The reverse converter presented in [19] is based on CRT and requires many multiplications and reductions, so its area and delay complexities have been increased.

In this paper, a fully parallel reverse conversion algorithm dedicated to the moduli set $\{r^n - 2, r^n - 1, r^n\}$ where $r=2k+1$, $k=1,2,\ldots$ and an efficient hardware realization are presented. Our proposed converter is faster and has lower hardware cost than the other MVL-based reverse converters for three-modulus sets.

The rest of paper is organized as follows. In section II we introduce the necessary background. The conversion algorithm and its hardware realization are presented in section III. Section IV makes comparison and section V is conclusion.

II. Background

Residue Number System: A residue number system is defined in terms of a relatively-prime moduli set $\{P_1,P_2, \ldots,P_n\}$ that is $\gcd(P_i,P_j)=1$ for $i \neq j$ and $i, j = 1,2,\ldots,n$. A weighted number X can be represented as $X=(x_1,x_2, \ldots ,x_n)$, where

$$x_i = X \bmod P_i = \left|X\right|_{P_i}, 0 \le x_i < P_i \quad (1)$$

Such a representation is unique for any integer X in the range $[0,M-1]$, where $M=P_1P_2\ldots P_n$ is the dynamic range of the moduli set $\{P_1,P_2, \ldots,P_n\}$.

Addition, subtraction and multiplication on residues can be performed in parallel without any carry propagation among the residue digits. Hence, by converting the arithmetic of large numbers to a set of the parallel arithmetic of smaller numbers, the RNS representation yields significant speed up.

The challenges of the RNS system design lie in the moduli set selection and in the conversion of the residues to the

Manuscript received September 9, 2007.

M. Hosseinzadeh and A. Sabbagh are with the Department of Computer Engineering, Islamic Azad University Science and Research Branch, Tehran, Iran, (emails: hosseinzadeh@sr.iau.ac.ir ; amir.sabbagh@sr.iau.ac.ir).

K. Navi is with the Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran, (email: navi@sbu.ac.ir).

equivalent weighted number. The importance of the moduli selection is due to the fact that the dynamic range, the speed as well as the hardware complexity of RNS systems depend on the forms as well as the number of the moduli chosen. Transforming a weighted integer number to the residue representation is called forward conversion and getting back to the weighted representation is referred to as reverse conversion. Forward conversion is conceivably simple and consists of several modulo adders but the reverse conversion involves considerable degree of complexity and includes a lot of modulo operations. The algorithms of reverse conversion are based primarily on chinese remainder theorem and mixed-radix conversion (MRC).

Chinese Remainder Theorem: by CRT, the number X is calculated from residues by

$$X = \left| \sum_{i=1}^{n} \left| x_i N_i \right|_{P_i} M_i \right|_M \tag{2}$$

where $M_i = M/P_i$ and $N_i = |M_i^{-1}|_{P_i}$ is the multiplicative inverse of M_i modulo P_i.

Mixed-Radix Conversion: the number X can be computed by the formula

$$X = a_n \prod_{i=1}^{n} P_i + \ldots + a_3 P_2 P_1 + a_2 P_1 + a_1 \tag{3}$$

where a_is are called the mixed-radix coefficients and they can be obtained from the residues by

$$a_n = \left| (((x_n - a_1)\left|P_1^{-1}\right|_{P_n} - a_2)\left|P_2^{-1}\right|_{P_n} - \cdots - a_{n-1})\left|P_{n-1}^{-1}\right|_{P_n} \right|_{P_n} \tag{4}$$

where n>1 and a_1=x_1.

The CRT requires modular additions and multiplications and it is not efficient for the implementation. The MRC is a sequential algorithm and requires modular multiplications and subtractions that is not suitable for efficient hardware realization. In the next section we proposed a novel conversion algorithm which is depends on simple mathematical relationships without using CRT or MRC.

The RNS with Moduli Set {r^n – 2, r^n – 1, r^n}: In [19], a new moduli set {r^n – 2, r^n – 1, r^n} where r=2k+1, k=1,2,… was introduced for RNS. This moduli set contains pairwise relatively prime moduli for all different values of n, so the dynamic range is greater than the dynamic range of similar moduli sets in [19]. Because of using of high radix (r >2), this RNS can be simply realized in MVL and can provides very large dynamic range for different radix-r values.

Forward conversion in this RNS is performed as follows. Suppose a 3n-digit radix-r number Y

$$Y = (y_{3n-1} y_{3n-2} \ldots y_1 y_0) \tag{5}$$

We must compute the residues (x_3, x_2, x_1) corresponding to moduli set {r^n – 2, r^n – 1, r^n}. First we calculate the residue in moduli r^n. So we have

$$\begin{aligned} x_1 &= Y \bmod r^n = (y_{3n-1} \ldots y_1 y_0) \bmod r^n \\ &= ((y_{3n-1} \ldots y_{n+1} y_n) r^n + (y_{n-1} \ldots y_1 y_0)) \bmod r^n \\ &= (y_{n-1} \ldots y_1 y_0) \end{aligned} \tag{6}$$

Therefore, it is enough to consider the right most n digits and the rest of digits will be ignored as they are multiplies of r^n. Now we investigate the moduli r^n – 1. The residue of Y in moduli r^n – 1 can be calculated as follow

$$\begin{aligned} x_2 &= Y \bmod r^n - 1 = (y_{3n-1} \ldots y_1 y_0) \bmod r^n - 1 \\ &= ((y_{3n-1} \ldots y_{2n+1} y_{2n}) r^{2n} + (y_{2n-1} \ldots y_{n+1} y_n) r^n \\ &\quad + (y_{n-1} \ldots y_1 y_0)) \bmod r^n - 1 \\ &= \left| (y_{3n-1} \ldots y_{2n}) + (y_{2n-1} \ldots y_n) + (y_{n-1} \ldots y_0) \right|_{r^n - 1} \end{aligned} \tag{7}$$

So, the number Y is partitioned into consecutive n-digit blocks and then we must sum these blocks by a modular adder. The residue of Y in moduli r^n – 2 is obtained as follow

$$\begin{aligned} x_3 &= Y \bmod r^n - 2 = (y_{3n-1} \ldots y_1 y_0) \bmod r^n - 2 \\ &= ((y_{3n-1} \ldots y_{2n+1} y_{2n}) r^{2n} + (y_{2n-1} \ldots y_{n+1} y_n) r^n \\ &\quad + (y_{n-1} \ldots y_1 y_0)) \bmod r^n - 2 \\ &= \left| 4(y_{3n-1} \ldots y_{2n}) + 2(y_{2n-1} \ldots y_n) + (y_{n-1} \ldots y_0) \right|_{r^n - 2} \end{aligned} \tag{8}$$

Thus, after partitioning Y into consecutive n-digit blocks, we should add the least significant block with two times of the next block and with four times of the most significant block. Then, the result must be reduced in moduli r^n – 2. The details of the hardware realization of forward converter are presented in [19].

III. Parallel reverse converter

We now propose a novel conversion algorithm which converts residue number into its equivalent weighted number. The conversion method is based on simple mathematical relationships without using CRT or MRC.

Theorem: Given the moduli set {r^n – 2, r^n – 1, r^n}, the residue number (x_3, x_2, x_1) is converted into the radix-r weighted number by

$$X = \left| (c' \times r^{2n}) + (b' \times r^n) + x_1 \right|_M \tag{9}$$

where

$$c' = \begin{cases} \dfrac{x_1 + x_3 - 2x_2 + 3r^n - 10}{2} & \text{if } (x_1 + x_3) \text{ is odd} \\ \dfrac{x_1 + x_3 - 2x_2 + 2(r^n - 4)}{2} & \text{if } (x_1 + x_3) \text{ is even} \end{cases} \tag{10}$$

$$b' = \begin{cases} \dfrac{-3x_1 + 4x_2 - x_3 + r^n + 6}{2} & \text{if } (x_1 + x_3) \text{ is odd} \\ \dfrac{-3x_1 + 4x_2 - x_3 + 2(r^n + 2)}{2} & \text{if } (x_1 + x_3) \text{ is even} \end{cases} \tag{11}$$

$$M = (r^n - 2)(r^n - 1) r^n \tag{12}$$

Proof: Suppose the 3n-digit radix-r number X that is partitioned into consecutive n-digit blocks as

$$X = \underbrace{y_{3n-1} \ldots y_{2n+1} y_{2n}}_{c} \underbrace{y_{2n-1} \ldots y_{n+1} y_n}_{b} \underbrace{y_{n-1} \ldots y_1 y_0}_{a} \tag{13}$$

We know from the previous section that forward conversion is performed by the following equations

$$x_1 = a \tag{14}$$

$$x_2 = \left|a+b+c\right|_{r^n-1} \tag{15}$$

$$x_3 = \left|a+b+c\right|_{r^n-2} \tag{16}$$

Equations (15) and (16) can be rewritten as

$$x_2 = a+b+c-(\alpha\times(r^n-1)) \tag{17}$$

$$x_3 = a+b+c-(\beta\times(r^n-2)) \tag{18}$$

By substituting the value of a from (14), we have

$$b+c = x_2 - x_1 + (\alpha\times(r^n-1)) \tag{19}$$

$$2b+4c = x_3 - x_1 + (\beta\times(r^n-2)) \tag{20}$$

Therefore, we have

$$c = \frac{x_3 + x_1 - 2x_2 + (\beta\times(r^n-2)) - (\alpha\times 2(r^n-1))}{2} \tag{21}$$

$$b = \frac{-x_3 - 3x_1 + 4x_2 - (\beta\times(r^n-2)) + (\alpha\times 4(r^n-1))}{2} \tag{22}$$

Considering the maximum value of residues, the values of α and β are obtained as below

$$\begin{cases} \alpha = 2, \beta = 7 & if\ \ (x_1+x_3)\ \ is\ \ odd \\ \alpha = 2, \beta = 6 & if\ \ (x_1+x_3)\ \ is\ \ even \end{cases} \tag{23}$$

By substituting the values of α and β in (21) and (22), we obtained the following equations

$$c' = \begin{cases} \dfrac{x_1+x_3-2x_2+3r^n-10}{2} & if\ (x_1+x_3)\ is\ odd \\ \dfrac{x_1+x_3-2x_2+2(r^n-4)}{2} & if\ (x_1+x_3)\ is\ even \end{cases} \tag{24}$$

$$b' = \begin{cases} \dfrac{-3x_1+4x_2-x_3+r^n+6}{2} & if\ (x_1+x_3)\ is\ odd \\ \dfrac{-3x_1+4x_2-x_3+2(r^n+2)}{2} & if\ (x_1+x_3)\ is\ even \end{cases} \tag{25}$$

Thus, the values of b' and c' are calculated and then the result of concatenating x_1, b' and c' should be reduced in order to placed in the dynamic range. So

$$X = \left|(c'\times r^{2n}) + (b'\times r^n) + x_1\right|_M \tag{26}$$

The following example clarifies the conversion method.

Example: Given the moduli set $\{r^n-2, r^n-1, r^n\}$ where r=3 and n=3, the residue number (x_3, x_2, x_1)=(0,12,14) is converted to the weighted number X by this way

Moduli Set: $\{3^3-2, 3^3-1, 3^3\}$={25, 26, 27}

M=25×26×27=17550

(x_1+x_3) is even, so

$$c' = \frac{14+0-24+46}{2} = 18$$

and

$$b' = \frac{-42+48-0+58}{2} = 32$$

Therefore,

$$X = \left|18\times 3^6 + 32\times 3^3 + 14\right|_{17550} = 14000$$

It is easy to see that Theorem 1 is very simpler than CRT or MRC. Also Theorem 1 enables us to implement a fully parallel reverse converter for the moduli set $\{r^n-2, r^n-1, r^n\}$. Hardware realization of the proposed reverse converter is based on equations (9)-(11). The computation of (10) requires an n-digit radix-r carry save adder (CSA_r) followed by two n-digit radix-r adder and a multiplexer. we check whether (x_1+x_3) is odd or even by using radix-r XOR gates. Instead of direct division by 2, we used the multiplication by 2^{-1} and as noted in [18], this multiplication can be performed by existing radix-r multiplier belonging to the RNS processing hardware. Equation (11) has a similar realization and can be implemented with a CSA tree, two adders, one multiplexer and one multiplier for performing multiplication by 2^{-1}. Since x_1 is an n-digit radix-r number, no additional hardware is needed to compute $(r^n b'+x_1)$. The desired result is the concatenation of x_1 and b'. Finally, a 3n-digit radix-r modular adder is used to perform the addition of $r^{2n}c'$ and $(r^n b'+x_1)$. Fig. 1 shows the hardware architecture of the proposed reverse converter.

IV. PERFORMANCE EVALUATION

The reverse converter proposed in this paper is a novel converter dedicated to the moduli set $\{r^n-2, r^n-1, r^n\}$. Therefore, to verify the performance of this converter, it has to be compared with other proposed converters which can convert residue numbers to their equivalent radix-r (r > 2) weighted representation. Such reverse converters are presented in [19] and [18]. In [19] a CRT-based reverse converter for the moduli set $\{r^n-2, r^n-1, r^n\}$ was presented. In [18] four three-moduli sets $s_1=\{r^{n-2}+1, r^{n-1}+1, r^n+1\}$, $s_2=\{r^{n-3}+1, r^{n-1}+1, r^n+1\}$, $s_3=\{r^{n-4}-1, r^{n-2}-1, r^n-1\}$ and $s_4=\{r^{n-5}-1, r^{n-3}-1, r^{n-1}-1\}$ are proposed and also a conversion algorithm based on CRT with scale-down factors is presented. Since the hardware requirements and conversion speed for the moduli sets s_1 and s_3 are the same as the moduli sets s_2 and s_4, respectively, we confine our comparison to to the reverse converter for moduli sets s_1 and s_3. All the hardwares listed in Table I, are radix-r hardware. An overview of MVL circuits' implementation can be found in [20], [21]. It should be noted the radix-r full adder (FA) cell is a r-input adder cell. Therefore, a radix-r CSA tree includes r-to-2 CSA's. Since the CSA tree used in Fig. 1 has 8 inputs, the maximum levels of r-to-2 CSA in CSA tree is two. So, the total delay of the CSA tree used in Fig. 1 is the delay of two radix-r FA. Also for r=3, this CSA tree consists of three 4-to-2 radix-3 CSA and for greater values of r (r>3), it consists of two r-to-2 CSA. For comparison with other converters, we consider the worst case (r=3) for CSA tree. The hardware requirements and conversion delays of the reverse converters listed in Table I and II, respectively. It should be noted that we used some radix-r NOT and XOR gates for calculating negative numbers and checking the value of (x_1+x_3) for odd or even, respectively. Since these gates are not on the critical delay path and their cost are small, we don't taking into

account these gates in Table I and II.

It has been assumed that T_{FAr}, T_{MUXr}, T_{MULr} and T_{MOAr} refer to the delays of the radix-r full adder, multiplexer, multiplier and mod M modular adder, respectively.

It is clear from Table I and II that the area and delay of the proposed converter are considerably better than the other converters. Our proposed converter is very faster and results to hardware savings when compared with the converters of [18] and [19]. It must be noted that for a specified dynamic range, the value of n for our reverse converter is smaller than the n for reverse converters of [18].

V. CONCLUSION

In this paper, we proposed a fully parallel reverse conversion algorithm for the moduli set $\{r^n - 2, r^n - 1, r^n\}$, based on simple mathematical relationships. Our proposed conversion algorithm is very simple and resulted to an efficient hardware realization of the reverse converter. In comparison with other reverse converters for MVL based RNS systems, the presented converter has a superior area-time complexity.

REFERENCES

[1] M. A. Soderstrand and et al. Eds, *Residue number system arithmetic: modern applications in digital signal processing*, New York: IEEE Press, 1986.

[2] N. Szabo and R. Tanaka, *Residue arithmetic and its applications to computer technology*, New York: McGraw-Hill, 1967.

[3] B. Parhami, *Computer arithmetic: algorithms and hardware designs*, Oxford, 2001.

[4] T. Stouratitis and V. Paliouras, "Considering the alternatives in lowpower design," *IEEE Circuits and Devices*, pp. 23–29, 2001.

[5] R. Conway and J. Nelson, "Improved RNS FIR Filter Architectures," *IEEE Transactions On Circuits and Systems II*, Vol. 51, No. 1, pp. 26-28, 2004.

[6] P. G. Fernandez, et al., "A RNS-Based Matrix-Vector-Multiply FCT Architecture for DCT Computation," *Proc. of 43rd IEEE Midwest Symposium on Circuits and Systems*, pp. 350-353, 2000.

[7] A. D. Re, A. Nannareli and M. Re, "A Tools for Arithmetic Generation of RTL-Level VHDL Description of RNS FIR Filters," *IEEE Proceeding of the Design, Automation and Test in Europe Conference and Exhibition*, pp. 686-687, 2004.

[8] W. L. Freking and K. K. Parhi, "Low-power FIR digital filters using residue arithmetic," *Proc. Of 31st Asilomar Conference on Signals, Systems, and Computers*, vol. 1, pp. 739–43, 1997.

[9] F. Taylor, "A Single Modulus ALU for Signal Processing," *IEEE Transactions on Acoustics, Speech, Signal Processing*, vol. 33, pp. 1302-1315, 1985.

[10] S. Yen, S. Kim, S. Lim and S. Moon, "RSA Speedup with Chinese Remainder Theorem Immune against Hardware Fault Cryptanalysis," *IEEE Transactions On Computers*, Vol. XX, No. Y, pp. 461-472, 2003.

[11] J. Ramirez, et al., "Fast RNS FPL-Based Communications Receiver Design and Implementation," *Proc. 12th Int'l Conf. Field Programmable Logic*, pp. 472-481, 2002.

[12] B. Parhami , "RNS Representation with Redundant Residues," *Proc. of the 35th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, pp. 1651-1655, 2001.

[13] E. Kinoshita and K. Lee, "A Residue Arithmetic Extension for Reliable Scientific Computation," *IEEE Transactions. On Computers*, Vol. 46, No. 2, pp. 129-138, 1997.

[14] V. Paliouras and T. Stouraitis, "Novel High-Radix Residue Number System Architectures," *IEEE Transactions On Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 47, No. 10, pp. 1059-1073, 2000.

[15] L. L. Yang, and L. Hanzo, "Redundant Residue Number System Based Error Correction Codes," *Proc. of VTC'2001*, Atlantic City, USA. pp. 1472-1476, 2001.

[16] Y. Wang, X. Song, M. Aboulhamid and H. Shen: "Adder based residue to binary numbers converters for $(2^n\text{-}1, 2^n, 2^n\text{+}1)$," *IEEE Trans. Signal Processing*, Vol. 50, No. 7, pp. 1772-1779, 2002.

[17] M. A. Soderstrand and R. A. Escott, "VLSI implementation in multiple-valued logic of an FIR digital filter using residue number system arithmetic," *IEEE Trans. Circuits Syst.*, vol. CAS -33, no. 1, pp. 5–25l , 1986.

[18] M. Abdallah and A. Skavantzos, "On MultiModuli Residue Number Systems With Moduli of Forms r^a, r^b-1, r^c+1," *IEEE Transactions Circuits System I*: Regular Paper, Vol. 52, No. 7, pp. 1253-1266, 2005.

[19] M. Hosseinzadeh, K. Navi, S. Gorgin, "A New Moduli Set for Residue Number System: $\{r^n\text{-}2, r^n\text{-}1, r^n\}$," *IEEE International Conference on Electrical Engineering*, 2007.

[20] E. Dubrova, "Multiple-Valued logic in VLSI: Challenges and opportunities," *34th IEEE International Symposium on Multiple-Valued Logic*, 2004.

[21] E. Kinvi-Boh, M. Aline, O. Sentieys, and E. D. Olson, "MVL circuit design and characterization at the transistor level using SUS-LOC," in *Proc. 33rd Int. Symp. Multiple-Valued Logic*, May 16–19, pp. 105–110, 2003.

[22] A. Hiasat and H. S. Abdel-Aty-Zohdy, "Residue-to-binary arithmetic converter for the moduli set $(2^k, 2^k\text{-}1, 2^{k\text{-}1}\text{-}1)$," *IEEE Trans. Circuits Syst.*, vol. 45, pp. 204–208, 1998.

[23] 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, pp. 1576–1581, 2000.

[24] Y. Wang, X. Song, M. Aboulhamid and H. Shen: "Adder based residue to binary numbers converters for $(2^n\text{-}1, 2^n, 2^n\text{+}1)$," *IEEE Trans. Signal Processing*, Vol. 50, No. 7, pp. 1772-1779, 2002.

[25] 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, pp. 1576–1581, 2000.

[26] S. L. Hurst, "Multiple-Valued Logic – Its status and its future," *IEEE Transaction on Computers*, pp. 1160-1179, 1984.

[27] A.F. Gonzalez, and P. Mazumdar, "Redundant Arithmetic, Algorithms and Implementations," *Integration: The VLSI Journal,* Vol. 30, No. 1, pp. 13-53, 2000.

[28] A. K. Jain, R. J. Bolton, and M. H. Abd-El-Barr, "CMOS multiplevalued logic design—Part I: Circuit implementation," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 40, no. 8, pp. 503–514, 1993.

[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] A. A. Hiasat, "VLSI implementation of New Arithmetic Residue to Binary Decoders," *IEEE Trans. VLSI Systems*, Vol.13, pp. 153-158, 2005.

[31] A. Hariri, K. Navi, R. Rastegar, "A Simplified Modulo $(2^n - 1)$ Squaring Scheme for Residue Number System," *Proc. IEEE International Conference on Computer as a tool* , 2005.

[32] S. Timarchi, K. Navi and M. Hosseinzadeh, "New Design of RNS Subtractor for modulo $(2^n + 1)$," *Proc. 2th IEEE International Conference on Information & Communication Technologies: From Theory To Applications*, 2006.

[33] M. Hosseinzadeh, K. Navi and S. Timarchi, "Design of Residue Number System Circuits in Current mode," *Proc. 14th Iranian Conference of Electrical Engineering*, 2006.

[34] A. Sabbagh, K. Navi, "An Improved Residue to Binary Converter for the RNS with Pairs of Conjugate Moduli," *Proc. International Conference on Electrical Engineering and Informatics*, Indonesia, 2007.

[35] M. Hosseinzadeh, K. Navi and S. Timarchi, "New Design of 4-3 Compressor," *Proc.* 11th *International CSI Computer Conference of Iran*, 2006.

[36] A. Hariri, K. Navi, R. Rastegar, "A new high dynamic range moduli set with efficient reverse converter," *International Elsevier Journal of Computers and Mathematics with Applications*, doi:10.1016/j.camwa.2007.04.028, 2007.

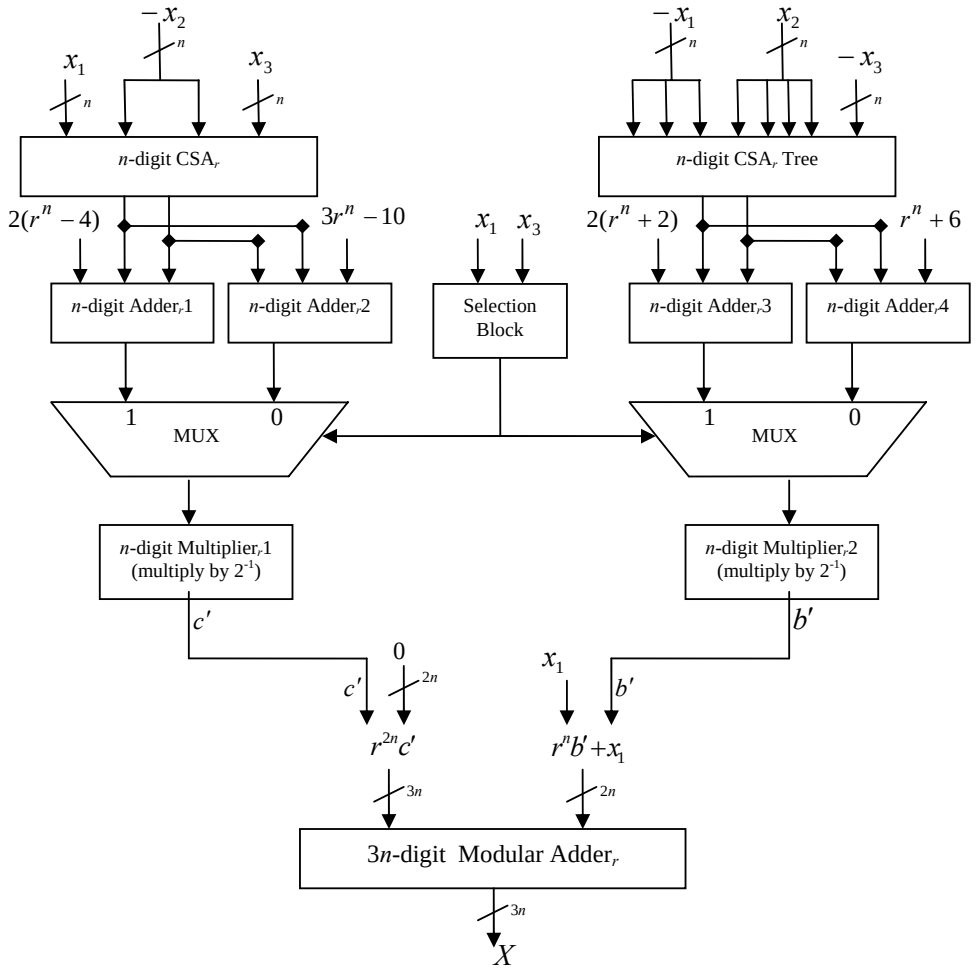

Fig. 1 Block diagram of the proposed reverse converter.

TABLE I COMPARISON OF HARDWARE REQUIREMENTS

converter	Moduli Set	Regular Adders and Subtractors						Total FA's	Modular Adder (3n)	Multiplier (n)	Multi plexer
		CSA	Adder		Subtractor						
		n	n	3n	n	2n	3n				
[19]	$\{r^n-2, r^n-1, r^n\}$	--	1	1	-	1	2	12n	1	3	--
[18]	$\{r^{n-2}+1, r^{n-1}+1, r^n+1\}$	--	1	4	-	--	--	13n	1	3	--
[18]	$\{r^{n-5}-1, r^{n-3}-1, r^{n-1}-1\}$	--	--	3	3	--	3	21n	1	3	--
Proposed	$\{r^n-2, r^n-1, r^n\}$	4	4	--	-	--	--	8n	1	2	2

TABLE II COMPARISON OF CONVERSION DELAYS

converter	Moduli Set	Conversion Delay
[19]	$\{r^n-2, r^n-1, r^n\}$	$(5n)T_{FA_r}+T_{MUL_r}+T_{MOA_r}$
[18]	$\{r^{n-2}+1, r^{n-1}+1, r^n+1\}$	$(6n)T_{FA_r}+T_{MUL_r}+T_{MOA_r}$
[18]	$\{r^{n-5}-1, r^{n-3}-1, r^{n-1}-1\}$	$(6n)T_{FA_r}+T_{MUL_r}+T_{MOA_r}$
Proposed	$\{r^n-2, r^n-1, r^n\}$	$(n+2)T_{FA_r}+T_{MUL_r}+T_{MOA_r}+T_{MUX_r}$