International Journal of Computer Sciences and Engineering Systems
Vol. 10 No. 2 (December, 2016)

New Arithmetic Residue to Binary Converters

Amir Sabbagh MOLAHOSSEINI[1] and Keivan NAVI[2]

[1]Department of Computer Engineering, Islamic Azad University, Science and Research Branch Tehran, Iran
E-mail: amir.sabbagh@sr.iau.ac.ir

[2]Faculty of Electrical and Computer Engineering, Shahid Beheshti UniversityTehran, Iran
E-mail: navi@sbu.ac.ir

Abstract: *The residue number system (RNS) is a carry-free number system which can support high-speed and parallel arithmetic. Two major issues in efficient design of RNS systems are the moduli set selection and the residue to binary conversion. In this paper, we present two efficient residue to binary converters for the new three-moduli set $\{2^n, 2^{n+1} + 1, 2^{n+1}–1\}$. This moduli set consists of pairwise relatively prime and balanced moduli, which can offer fast internal RNS processing and efficient implemenatation of the residue to binary converter. The proposed residue to binary converters are memoryless and consist of adders. In comparison with other residue to binary converters for a three-moduli set, the proposed converters have better area-time complexity.*

Keywords: *Residue to binary converter, reverse converter, residue number system (RNS), computer arithmetic*

1. INTRODUCTION

The residue number system (RNS) is a non-weighted number system which speeds up arithmetic operations by dividing them into smaller parallel operations. 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 [1]. RNS is one of the most effective techniques for reducing the power dissipation in VLSI systems design [2]. Also RNS can be efficiently realized in multiple-valued logic (MVL) [3, 4]. Some applications of the RNS are digital signal processing (DSP) [5, 6], the RSA encoding algorithm [7] and digital communication [8]. The architecture of the RNS is naturally fault tolerant and consequently, it is used for error detection, error correction and fault tolerance [9,10]. The complexity as well as the efficiency of residue to binary converter is primarily based on the proper selection of the moduli set and the conversion algorithm. Many different moduli sets have been suggested. Among these, three-moduli sets have been extensively investigated, such as $\{2^n – 1, 2^n, 2^n + 1\}$ [11,12], $\{2^n, 2^n –1, 2^{n–1} –1\}$ [13,14], $\{2^n, 2^n – 1, 2^{n+1} –1\}$ [15], $\{2^{2n} + 1, 2^n + 1, 2^n – 1\}$ [16] and $\{2^n, 2^{2n} – 1, 2^{2n} + 1\}$ [17]. The algorithms of residue to binary conversion are mainly based on Chinese remainder theorem (CRT) [1], mixed-radix conversion (MRC) [1] and new Chinese remainder theorems (New CRTs) [18]. In addition to these, novel conversion algorithms [19] which are designed for some special moduli sets have been proposed. Among these, New CRTs algorithms have simple computations which can be efficiently realized in hardware.

In this paper, firstly we proposed the new three-moduli set $\{2^n, 2^{n+1} + 1, 2^{n+1} – 1\}$. This moduli set contains balanced and well-formed moduli which can result in efficient implementation of the residue to binary converter. Then, we present two efficient designs of the residue to binary converter for these three-moduli set based on New CRT. The proposed converters have better performance, compared to the other residue to binary converters for a three-moduli set with similar dynamic range, where the dynamic range is defined in terms of product of the moduli.

The rest of paper is organized as follows. In section 2 we introduce the necessary background. The residue to binary converters is presented in section 3. Section 4 makes comparisons and section 5 is conclusion.

2. BACKGROUND

A residue number sytem 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 $i \neq j$. A weighted binary number X can be represented as $X = (x_1, x_2,\ldots, x_n)$, where

$$x_i = X \bmod P_i = \left|X\right|_{P_i}, \quad 0 \leq x_i < P_i \tag{1}$$

Such a representation is unique for any integer X in the range $[0, M\text{-}1]$, where $M=P_1P_2\ldots P_n$ is the dynamic range of the moduli set $\{P_1, P2\ldots Pn\}$ [11]. Addition, subtraction and multiplication on residues can be performed in parallel without carry propagation. Hence, by converting the arithmetic of large numbers to a set of the parallel arithmetic of smaller numbers, RNS representation yields significant speed up. Binary to residue conversion [20] is very simple

and can be implemented with modular adders. When binary to residue conversion of the needed operands had finished, arithmetic operations on RNS numbers are performed in parallel without carry-propagation between residue digits. Hence, RNS leads to carry-free, parallel and high-speed arithmetic. It should be noted that each modulo of the moduli set has its own arithmetic processor which is consists of a modulo adder, a modulo subtractor and a modulo multiplier. In order to use the result of arithmetic operations in outside of RNS, the resulted RNS number must be converted into its equivalent weighted binary number. The algorithms of residue to binary conversion are mainly based on CRT, MRC and New CRTs.

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 Mi modulo P_i. Using the MRC, the number X can be computed by the equation

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

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 MRC is a sequential approach and CRT requires large modulo operations which is not suitable for efficient hardware implementation. By New CRT-I [21], the number X is calculated by

$$X = x_1 + P_1 \left| \begin{matrix} k_1(x_2 - x_1) + k_2 P_2 (x_3 - x_2) + \cdots \\ + k_{n-1} P_2 P_3 \cdots P_{n-1}(x_n - x_{n-1}) \end{matrix} \right|_{P_2 P_3 \cdots P_n} \tag{5}$$

Where

$$\left| k_1 \times P_1 \right|_{P_2 P_3 \cdots P_n} = 1$$

$$\left| k_2 \times P_1 \times P_2 \right|_{P_3 \cdots P_n} = 1$$

$$\cdots\cdots$$

$$\left| k_{n-1} \times P_1 \times P_2 \times \cdots \times P_{n-1} \right|_{P_n} = 1 \tag{6}$$

For a three-moduli set $\{P_1, P_2, P_3\}$, the number X can be converted from its residue representation (x_1, x_2, x_3) by New CRT-I as follow

$$X = x_1 + P_1 \left| k_1(x_2 - x_1) + k_2 P_2 (x_3 - x_2) \right|_{P_2 P_3} \tag{7}$$

Where

$$\left| k_1 \times P_1 \right|_{P_2 P_3} = 1 \tag{8}$$

$$\left| k_2 \times P_1 \times P_2 \right|_{P_3} = 1 \tag{9}$$

3. RESIDUE TO BINARY CONVERTER

In this section, New CRT-I is applied to derive an efficient residue to binary conversion algorithm for the new moduli-set $\{2^n, 2^{n+1} + 1, 2^{n+1}–1\}$. First, we must prove that this moduli set includes pairwise relatively prime numbers.

Theorem 1: The numbers 2^n, $2^{n+1} + 1$, $2^{n+1}– 1$ are pairwise relatively prime.

Proof: Baesd on Euclid's Theorem, we have

$$\gcd(a,b)=\gcd(b,a \bmod b) \tag{10}$$

where gcd(a,b) denotes the greatest common divisor of a and b. So we have,

$$\gcd(2^{n+1}+1,\ 2^{n+1}–1)= \gcd(2^{n+1}–1,\ 2) = 1 \tag{11}$$

$$\gcd(2^{n+1}–1,\ 2^n)= \gcd(2^n,\ –1)=1 \tag{12}$$

$$\gcd(2^{n+1}+1,\ 2^n)= \gcd(2^n,\ 1)=1 \tag{13}$$

since all the greatest common divisors are equal to 1, these three numbers are pairwise relatively prime.

Proposition 1: The multiplicative inverse of 2^n modulo $(2^{2n+2} –1)$ is $k_1= 2^{n+2}$.

Proof: by substituting values in (8), we have

$$\begin{aligned} \left| k_1 \times 2^n \right|_{(2^{n+1}+1)(2^{n+1}–1)} &= \left| 2^{n+2} \times 2^n \right|_{2^{2n+2}–1} \\ &= \left| 2^{2n+2} \right|_{2^{2n+2}–1} = 1 \end{aligned} \tag{14}$$

Proposition 2: The multiplicative inverse of $2^n \times (2^{n+1}+1)$ modulo $(2^{n+1} –1)$ is $k_2= 2^{n+1}$.

Proof: Since $\left| 2^{n+1} + 1 \right|_{2^{n+1}–1} = 2$, by substituting values in (9), we have

$$\begin{aligned} &\left| k_2 \times 2^n \times (2^{n+1} + 1) \right|_{2^{n+1}–1} \\ &= \left| 2^{n+1} \times 2^n \times (2^{n+1} + 1) \right|_{2^{n+1}–1} \\ &= \left| 1 \times 2^n \times 2 \right|_{2^{n+1}–1} = \left| 2^{n+1} \right|_{2^{n+1}–1} = 1 \end{aligned} \tag{15}$$

Theorem 2: In the RNS defined by the three-moduli set $\{2^n, 2^{n+1} + 1, 2^{n+1}– 1\}$, the weighted binary number X can be calculated from its corresponding residues (x_1, x_2, x_3) by

$$X = x_1 + 2^n \left| 2^{n+2}(x_2 - x_1) + 2^{n+1}(2^{n+1} + 1)(x_3 - x_2) \right|_{2^{2n+2}–1} \tag{16}$$

Proof: By letting $P_1=2^n$, $P_2=2^{n+1}+1$, $P_3=2^{n+1}–1$ and the values of k_1, k_2 from Propositions 1 and 2 into (7), we have

$$X = x_1 + 2^n \left| 2^{n+2}(x_2 - x_1) + 2^{n+1}(2^{n+1} + 1)(x_3 - x_2) \right|_{2^{2n+2}–1}$$

Example: Given the moduli set $\{2^n, 2^{n+1}+1, 2^{n+1}–1\}$ where $n = 3$, the residue number $(x_1, x_2, x_3) = (2, 5, 7)$ is converted into its equivalent weighted number as follows, by substituting the residues and $n = 3$ into (16), we have

$$X = 2 + 8\left|32(3) + 16(17)(2)\right|_{255} = 1042$$

By using the following properties, Theorem 2 is simplified to reduce the hardware complexity.

Property 1: Modulo (2^p-1) multiplication of a residue number by 2^k, where p and k are positive integers, is equivalent to k bit circular left shifting [22].

Property 2: Modulo (2^p-1) of a negative number is accomplished by subtracting this number from (2^p-1). This is equivalent to taking the one's complement of the number [22].

Suppose that the residues x_1, x_2 and x_3 have binary representation as follow

$$x_1 = (x_{1,n-1}x_{1,n-2}\cdots x_{1,1}x_{1,0}) \tag{17}$$

$$x_2 = (x_{2,n+1}x_{2,n}\cdots x_{2,1}x_{2,0}) \tag{18}$$

$$x_3 = (x_{3,n}x_{3,n-1}\cdots x_{3,1}x_{3,0}) \tag{19}$$

Equation (16) can be rewritten as

$$X = x_1 + 2^n Y \tag{20}$$

Where

$$Y = \left|v_1 + v_2 + v_3\right|_{2^{2n+2}-1} \tag{21}$$

$$v_1 = \left|-2^{n+2}x_1\right|_{2^{2n+2}-1} \tag{22}$$

$$v_2 = \left|(2^{n+2} - 2^{n+1} - 2^{2n+2})x_2\right|_{2^{2n+2}-1} \tag{23}$$

$$v_3 = \left|(2^{2n+2} + 2^{n+1})x_3\right|_{2^{2n+2}-1} \tag{24}$$

With respect to the Properties 1 and 2, we have

$$\begin{aligned} v_1 &= \left|-2^{n+2}x_1\right|_{2^{2n+2}-1} = \left|-(\underbrace{x_{1,n-1}\cdots x_{1,1}x_{1,0}}_{n}\underbrace{0\cdots 00}_{n+2})\right|_{2^{2n+2}-1} \\ &= \underbrace{\bar{x}_{1,n-1}\cdots\bar{x}_{1,1}\bar{x}_{1,0}}_{n}\underbrace{1\cdots 11}_{n+2} \end{aligned} \tag{25}$$

Since $2^{n+2}-2^{n+1}=2^{n+1}$, equation (23) can be rewritten as

$$v_2 = \left|(2^{n+1} - 2^{2n+2})x_2\right|_{2^{2n+2}-1} = \left|v_{21} + v_{22}\right|_{2^{2n+2}-1} \tag{26}$$

where

$$\begin{aligned} v_{21} &= \left|2^{n+1}x_2\right|_{2^{2n+2}-1} = \left|2^{n+1}(\underbrace{0\cdots 00}_{n}\underbrace{x_{2,n+1}\cdots x_{2,1}x_{2,0}}_{n+2})\right|_{2^{2n+2}-1} \\ &= \underbrace{x_{2,n}\cdots x_{2,1}x_{2,0}}_{n+1}\underbrace{0\cdots 00}_{n}x_{2,n+1} \end{aligned} \tag{27}$$

$$\begin{aligned} v_{22} &= \left|-2^{2n+2}x_2\right|_{2^{2n+2}-1} = \left|-x_2\right|_{2^{2n+2}-1} \\ &= \underbrace{1\cdots 11}_{n}\underbrace{\bar{x}_{2,n+1}\cdots\bar{x}_{2,1}\bar{x}_{2,0}}_{n+2} \end{aligned} \tag{28}$$

And v_3 is calculated by

$$\begin{aligned} v_3 &= \left|(2^{2n+2} + 2^{n+1})x_3\right|_{2^{2n+2}-1} = \left|2^{n+1}(2^{n+1}+1)x_3\right|_{2^{2n+2}-1} \\ &= \left|2^{n+1}(\underbrace{x_{3,n}\cdots x_{3,1}x_{3,0}}_{n+1}\underbrace{x_{3,n}\cdots x_{3,1}x_{3,0}}_{n+1})\right|_{2^{2n+2}-1} \\ &= \underbrace{x_{3,n}\cdots x_{3,1}x_{3,0}}_{n+1}\underbrace{x_{3,n}\cdots x_{3,1}x_{3,0}}_{n+1} \end{aligned} \tag{29}$$

Since both least significant $(n+2)$ bits of v_1 in (25) and most significant n bits of v_{22} in (28) are 1's, we can use the following vectors instead of v_1 and v_{22}

$$v_1' = \underbrace{\bar{x}_{1,n-1}\cdots\bar{x}_{1,1}\bar{x}_{1,0}}_{n}\underbrace{\bar{x}_{2,n+1}\cdots\bar{x}_{2,1}\bar{x}_{2,0}}_{n+2} \tag{30}$$

$$v_{22}' = \underbrace{1\cdots 11}_{n}\underbrace{1\cdots 11}_{n+2} \tag{31}$$

we know that,

$$\left|v_{22}'\right|_{2^{2n+2}-1} = \left|\underbrace{1\cdots 11}_{2n+2}\right|_{2^{2n+2}-1} = \left|2^{2n+2}-1\right|_{2^{2n+2}-1} = 0 \tag{32}$$

So, Y in (21) can be calculated by

$$Y = \left|v_1' + v_{21} + v_3\right|_{2^{2n+2}-1} \tag{33}$$

Hardware implementation of the proposed residue to binary converters for the moduli set $\{2^n, 2^{n-1}+1, 2^{n-1}-1\}$ are based on (20) and (30) and consist of one $(2n+2)$-bit carry save adder (CSA) with end around carry (EAC) and a modulo $(2^{2n+2}-1)$ adder. Modulo $(2^{2n+2}-1)$ adder can be implemented with different methods. By using a $(2n+2)$-bit one's complement adder for performing modulo $(2^{2n+2}-1)$ addition, we obtain a cost-efficient (CE) converter. One's complement adder is a carry propagate adder (CPA) with EAC. Instead of using a one's complement adder, we can use the method of [23]. In this method, two $(2n+2)$-bit regular CPA's are work in parallel, one with a zero carry-in and the other with a one carry-in. The correct result is selected by a multiplexer (MUX) based on the carry-out of the adder with zero carry-in. In this case, we obtain a speed-efficient (SE) converter. It should be noted that, Since x_1 is an n-bit number, no computational hardware is needed to compute x_1+2^nY in (20). The desired result is the result of concatenating x_1 with Y. The proposed implementations of the residue to binary converter are shown in Fig. 1, 2.

4. COMPARISONS

In this section, we evaluate the performance of the proposed residue to binary converters in terms of hardware cost and conversion delay. In (33), the three operands are added using a $(2n+2)$-bit CSA with EAC and a modulo $(2^{2n+2}-1)$ adder. Calculation of (27), (29) and (30) rely on simply manipulating the routing of the bits of the residues and only $(2n+2)$ inverter are used for perfoming the inversions of (30). Since (27) has n bits of 0's, n of the full adders (FA's) in

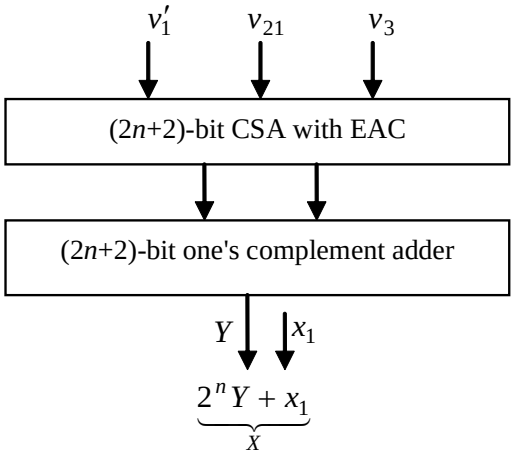

Figure 1: The Proposed Cost-efficient Residue to Binary Converter

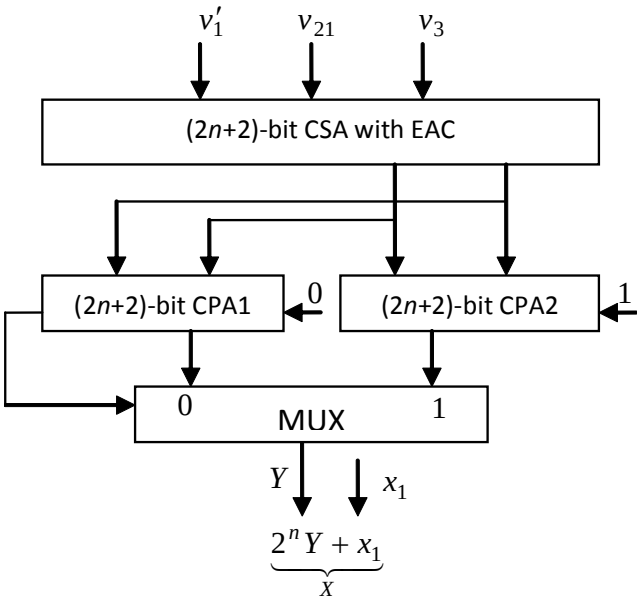

Figure 2: The Proposed Speed-efficient Residue to Binary Converter

CSA are reduced to half adders (HA's). Hence, the CSA with EAC is consists of (n+2) FA's and n HA's. The (2n+2)-bit one's complement adder has complexity of (2n+2) FA's and the delay of $(4n+4)t_{FA}$, where t_{FA} denotes the delay of one FA. Therefore, the total cost of the proposed cost-efficient residue to binary converter is $n+2+2n+2=(3n+4)$ FA's and n HA's. The delay of a CSA is the same as that of an FA. So, the proposed cost-efficient converter has a total delay of $1+4n+4=(4n+5)t_{FA}$. The proposed speed-efficient residue to binary converter used two (2n+2)-bit CPA that work in parallel. Therefore, the total cost of this converter is $n+2+4n+4=(5n+6)$ FA's and n HA's. Also it has the delay of $1+2n+2=(2n+3)t_{FA}$.

To verify the performance of the proposed converters, they have to be compared with other residue to binary converters for a three-moduli set with similar dynamic range. The closest three-moduli set to the proposed moduli set is the moduli set $\{2^n, 2^n-1, 2^{n+1}-1\}$. Three residue to binary converters for this moduli set have been presented in [15]. The first one is based on MRC and the second is based on CRT, both are adder based. But the third converter uses ROM. Table 1 shows the hardware requirements and conversion delays of these converters and also the proposed converters.

It is clear from Table 1 that the proposed speed-efficient converter is faster than all the converters of [15] while it requires less hardware than the converters [15]-CII and [15]-CIII. Also the proposed cost-efficient converter utilizes lower hardware than the converters of [15] and also it is faster than the converter [15]-CI. It should be noted that for a same value of n, the proposed residue to binary converters support larger dynamic range than the residue to binary converters of [15].

5. CONCLUSIONS

In this work, we introduced a new three-moduli set for RNS which can results in efficient residue to binary conversion. Also efficient residue to binary converters for the proposed moduli set based on New CRT-I is presented. Comparison with other residue to binary converters shown that the proposed converters have better performance.

Table 1
Hardware Requirements and Conversion Delays of the Residue to Binary Converters

Converter	*[15]-CI*	*[15]-CII*	*[15]-CIII*	*Proposed-CE*	*Proposed-SE*
FA	4n+3	14n+21	12n+19	3n+4	5n+6
HA	–	2n+3	2n+2	n	n
OR/NOT	n	–	–	2n+2	2n+2
XNOR	n	–	–	–	–
Multiplexer	–	1	1	–	1
ROM	–	–	1	–	–
Delay	$(6n+5)t_{FA}$	$(2n+7)t_{FA} + t_{MUX}$	$(2n+7)t_{FA} + t_{MUX}$	$(4n+5)t_{FA}$	$(2n+3)t_{FA} + t_{MUX}$

REFERENCES

[1] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Design*, Oxford University Press, 2000.

[2] T. Stouratitis and V. Paliouras, "Considering the Alternatives in Lowpower Design", *IEEE Circuits and Devices*, 2001, pp. 23–29.

[3] M. Hosseinzadeh, K. Navi, S. Gorgin, "A New Moduli Set for Residue Number System: {rn-2, rn-1, rn}", in *IEEE International Conference on Electrical Engineering*, 2007, pp. 1-6.

[4] M. Hosseinzadeh and K. Navi, "A New Moduli Set for Residue Number System in Ternary Valued Logic", *Journal of Applied Sciences*, Vol. 7, No. 23, pp. 3729-3735, 2007.

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

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

[7] 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. 52, No. 4, 2003, pp. 461-472.

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

[9] 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.

[10] L. L. Yang, and L. Hanzo, "Redundant Residue Number System Based Error Correction Codes", in *IEEE Vehicular Technology Conference*, 2001, Vol. 3, pp. 1472-1476.

[11] W. K. Jenkins and B. J. Leon, "The Use of Residue Number Systems in the Design of Finite Impulse Response Digital Filters", *IEEE Trans. Circuits Syst.*, vol. CAS-24, 1977, pp. 191–201.

[12] Y. Wang, X. Song, M. Aboulhamid and H. Shen, "Adder Based Residue to Binary Numbers Converters for (2n-1, 2n, 2n+1)", *IEEE Trans. Signal Processing*, Vol. 50, No. 7, 2002, pp. 1772-1779.

[13] A. Hiasat and H. S. Abdel-Aty-Zohdy, "Residue-to-Binary Arithmetic Converter for the Moduli Set (2k, 2k-1, 2k-1-1)", *IEEE Trans. Circuits Syst.*, Vol. 45, 1998, pp. 204–208.

[14] 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, 2000, pp. 1576–1581.

[15] P. V. A. Mohan, "RNS-To-Binary Converter for a New Three-Moduli Set {2n+1–1, 2n, 2n–1}," *IEEE Trans. Circuits Syst.-II*, Vol. 54, No. 9, 2007, pp. 775-779.

[16] F. Pourbigharaz and H. M. Yassine, "A Signed-Digit Architecture for Residue to Binary Transformation", *IEEE Trans. Comput.*, Vol. 46, 1997, pp. 1146–1150.

[17] 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.

[18] Y. Wang, "Residue-to-Binary Converters Based on New Chinese Remainder Theorems", *IEEE Trans. Circuits Syst.-II*, Vol. 47, No. 3, 2000, pp. 197-205.

[19] M. Hosseinzadeh, A. Sabbagh, K. Navi, "A Fully Parallel Reverse Converter," *International Journal of Electrical, Computer, and Systems Engineering*, Vol. 1, No. 3, 2007, pp. 183–187.

[20] B. Guan and E. V. Jones, "Fast Conversion Between Binary and Residue Numbers," *Electronics Letters*, Vol. 24, No. 19, 1988, pp. 1195–1197.

[21] B. Cao, C. H. Chang and T. Srikanthan, "An Efficient Reverse Converter for the 4-Moduli Set {2n-1, 2n, 2n+1, 22n+1} Based on the New Chinese Remainder Theorem", *IEEE Trans. Circuits Syst. I*, Vol. 50, No. 10, 2003, pp. 1296-1303.

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

[23] J. Mathew, D. Radhakrishnan, T. Srikanthan, "Fast Residue-to-Binary Converter Architectures," in *IEEE, 42nd Midwest Symposium on Circuits and Systems*, 2000, pp. 1090–1093.

[24] A. Hariri, K. Navi, R. Rastegar, "A Simplified Modulo 2n–1 Squaring Scheme for Residue Number System", in *IEEE International Conference on Computer as a tool*, 2005, Vol. 1, pp. 615-618.

[25] S. J. Piestrak, "Design of Residue Generators and Multioperand Modular Adders using Carry-Save Adders", *IEEE Trans. Comput.*, Vol. 423, No. 1, 1994, pp. 68-77.

[26] B. Cao, C. H. Chang and T. Srikanthan, "Adder Based Residue to Binary Converters for a New Balanced 4-Mduli Set, " in *3rd International Symposium on Image and Signal Processing and Analysis*, Vol. 2, pp. 820-825, 2003.

[27] A. Sabbagh, K. Navi, "An Improved Residue to Binary Converter for the RNS with Pairs of Conjugate Moduli," in *International Conference on Electrical Engineering and Informatics*, 2007, Vol. 1, pp. 318-320.

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