Asian Journal of Applied Science and Technology (AJAST)
Volume 1, Issue 1, Pages 255-264, February 2017

An Effective FPGA Design of A High Speed Reverse Converter for the Unrestricted Moduli Set

Dr.G.Prakash[1], Mr.A.Jagadeeswaran[2] and Mr.M.Prakash[3]

Excel Engineering College, Affiliated to Anna University, Namakkal, Tamilnadu, India. Email: gp1gp2@gmail.com[1]

Article Received: 11 February 2017 *Article Accepted: 19 February 2017* *Article Published: 23 February 2017*

ABSTRACT

A modern residue to binary conversion is a very challenging task to design a reverse converter for the various specific DSP and wireless applications. In this paper, a novel modified reverse converter will be designed for the unrestricted moduli set. This uses to investigate the Residue Number System (RNS) to decimal equivalent binary conversion for the utilization of RNS numbers in Digital Signal Processing (DSP) applications. This reverse conversion will be designed based on two approaches, the first one is Core Functional Approach which can be proved theoretically and practically speaking it outperforms state of the art equivalent converters. The second one is New CRT Approach which can be proved theoretically. The proposed converter is implemented on Xilinx Spartan 3 field-programmable gate array. The results indicate that the proposal shows the better performance in conversion time, area cost and power consumption.

Keywords: Residue Number system, Chinese Remainder Theorem, Core Function and FPGA.

1. INTRODUCTION

Residue Number System is an unweighted number system which supports the carry-free addition, borrow free subtraction and single step multiplication [1]. RNS is mostly applied in addition and multiplication dominated DSP application such as digital filtering and convolution. Moduli selection and the data conversion are the two important cases that can determine the RNS Hardware performance [2]. As RNS based functional units have to operate into a binary or decimal environment the input operands provided in either standard binary or decimal format need to be converted to RNS before performing any operation. The process of encoding the input data into RNS representation is called forward conversion, and the process of converting back the output data from RNS to conventional representation is called reverse conversion. The reverse conversion process is based on the Chinese Remainder Theorem (CRT) or Mixed Radix Conversion (MRC) techniques. To determine the RNS hardware performance the selection of moduli and the data conversion are the important cases in DSP applications.

RNS is defined by a set of relatively prime integers called the moduli. The moduli-set is denoted as $\{m_1, m_2 \ldots m_n\}$ where m_i is the i_{th} modulus. Each integer X can be represented as a set of smaller integers called the residues. The residue set is denoted as $\{x_1, x_2 \ldots x_n\}$ where x_i is the i_{th} residue. The residue x_i is defined as the least positive remainder when X divided by the modulus mi. This relation can be written based on congruence. $x_i = X \bmod m_i$ For an RNS processor to compete favorably with a conventional processor those conversions has to be fast such that the RNS speedup is not nullified by the conversion overhead. The conversion stages are very critical in the evaluation of the performance of the overall RNS. Conversion circuitry can be very complex and may introduce latency that offsets the speed gained by the RNS processors. For a full RNS based system, the interaction with the analog world requires conversion from analog to residue and vice versa. Usually, this is done in two steps where

conversion to binary is an intermediate stage. For RNS design to be competitive, the conversion process must be very fast as otherwise the RNS fast arithmetic advantage may be nullified by the slow conversion processes.

2. BACKGROUND

RNS is defined in terms of a set of relatively prime moduli set $\{m_i\}_{i=1,n}$ such that gcd $(m_i, m_j) = 1$ for i≠j, where gcd means the greatest common divisor of m_i and m_j, while $M = \prod_{i=1}^{n} m_i$, is the dynamic range. The residues of a decimal number X can be obtained as $x_i = |X|_{m_i}$ thus X can be represented in RNS as $X = (x_1, x_2 \ldots, x_n), 0 \leq x_i < m_i$. This representation is unique for any integer $X \in [0, M-1]$.

For a moduli set $\{m_i\}_{i=1, n}$ with the dynamic range $M = \prod_{i=1}^{n}$, the residue number (x_1, x_2, x_3 .... , x_n) can be converted into the decimal number X, according to the Chinese Remainder Theorem, as follows

$$X = \left| \sum_{i=1}^{n} M_i \left| M_i^{-1} . x_i \right|_m \right|_M , \qquad (1)$$

In the following section we introduce an RNS to decimal converter for the given moduli set based on the core functional approach and Mixed radix conversion. In core functional approach, the logic is reasonably straight forward. The fundamentals of the RNS representation are introduced in this section. The congruence is explained in details with their properties. These properties form a solid background to understand the process of conversion between the conventional system and the RNS. Basic algebra related to RNS is introduced here. This includes finding the additive and the multiplicative inverses, and some properties of division and scaling which are not easy operations in RNS. The basic definitions are used to understand the logic behind of each its property. It consists addition, subtraction, multiplication, multiplicative inverses, Additive inverses and division.

Prime numbers are considered for the process of unrestricted moduli selection whereas in restricted moduli selection depends on powers of two. RNS architectures are typically composed of three main parts, namely, a binary-to-residue converter, residue arithmetic units, and a residue-to-binary converter. The residue-to-binary converter is the most challenging part of any RNS architecture. For RNS design to be competitive, the conversion process must be very fast as otherwise the RNS fast arithmetic advantage may be nullified by the slow conversion processes.

As those conversions are computation demanding and have crucial influence on the speed and complexity of the RNS architectures, designing efficient converters has been an important task in the realization of different RNS based applications and processors and several proposals have been reported up to date.

3. PROPOSED METHODOLOGY

3.1 Core Functional Approach

Core function is one of the methodologies for the reverse conversion. It provides a logic, which is reasonably straightforward, although there are some minor complications. There are some certain operations are

fundamentally difficult in RNS. The essence of the difficulties is that RNS digits carry no weight information, and therefore it is not easy to determine the magnitude of a given RNS number. It is one of the best efficient methodology to convert from residue to binary number for DSP applications.

There are certain operations are fundamentally difficult in RNS; these include magnitude-comparison, sign-determination, overflow- determination, and division. The essence of the difficulties is that RNS digits carry no weight information, and therefore it is not easy to determine the magnitude of a given RNS number. The core function provides some means that can help deal with the difficult operations; briefly, one may view it as a method for approximating the magnitude of an RNS number. The underlying idea is to map an RNS number $n \in [0, M\text{-}1]$ to a number $C(n) \in [0, C(M)]$, where $C(M) << M$ and M is the product of the moduli $m_1, m_2, \ldots, m_N$ in the RNS.

The Core Function provides a method for reverse conversion that is reasonably straightforward, although there are some minor complications. Generally RNS has some difficulties like magnitude-comparison, sign-determination and overflow determination. These difficulties can be solved with the help of core functional approach.

Let $\{x_1, x_2, \ldots, x_N\}$ be the residue representation of x, with respect to the moduli $m_1, m_2, \ldots, m_N$. Then the core, C(x), of x is defined by

$$C(n) = \sum_{i=1}^{N} \left\lfloor \frac{x}{m_i} \right\rfloor$$

Thus, $C(M)$ should be as large as possible, or the weights should be as small as possible. There are, however, some constraints: To keep hardware simple and balanced, $C(M)$ should be of roughly the same magnitude as the moduli, and these are likely to be rather small. Choices for the weights are also restricted by the solutions to the equations above. One way in which the noisy part can be readily reduced is by replacing the definition above of the Core Function.

$$C(x) = \frac{C(M)}{M} x - \sum_{i=1}^{N} \frac{|w_i x_i|_{m_i}}{m_i}$$

The core of a number can be obtained by using a Chinese Remainder Theorem for Core Functions, but this sometimes produces values that have some ambiguity. This ambiguity can be eliminated by the use of an extra modulus, m_E, that is greater than the difference between the smallest and largest values assumed by the given core function. Reduction, with respect to m_E, of both sides.

$$|C(x)|_{m_E} = \left| \frac{C(M)}{M} x - \sum_{i=1}^{N} \frac{w_i}{m_i} x_i \right|_{m_E}$$

$$= \left| \frac{C(M)}{M} |x|_{m_E} - \sum_{i=1}^{N} \frac{w_i}{m_i} x_i \right|_{m_E} \qquad (2)$$

Ambiguities are then eliminated, because the core is evaluated relative a modulus larger than its range. To avoid complex computations for$|x|_{m_E}$, has instead proposed the use of parity bit. This requires that M be odd. The weight can be found as follows. Let$M_i = \frac{M}{m_i}$, the C (M) is

$$|C(M)|_{m_i} = \left|\sum_{i=1}^{N} w_i M_i\right|_{m_i} \quad (3)$$

$$|C(M)|_{m_i} = |w_i|_{m_i}$$

Since all but one of the terms in the sum is a multiple of m_i. Therefore

$$|w_i|_{m_i} = \left|C(M)\left|M_i^{-1}\right|_{m_i}\right|_{m_i}$$

Where $\left|M_i^{-1}\right|_{m_i}$ is the multiplicative inverse of M_i with respect to m_i. Thus if C (M) is given, then the w_i are obtainable.

In a {2n+1, 2n, 2n-1} moduli set, the weights in the equation $X = \sum_i w_i x_i$ can be calculated with little computational complexity. The weights corresponding to the three moduli m_1= 2n+1, m_2= 2n and m_3=2n-1 may be computed as

$$w_1 = \frac{(m_1+1)m_2m_3}{2}, w_2 = m_1(m_2-1)m_3, w_3 = \frac{m_1m_2(m_3+1)}{2}$$

Where, w_1, w_2 and w_3 are the multiplicative inverses of m_1, m_2 and m_3 respectively.

Once the weights have been computed, the conventional equivalent X, of the residue numbers x_1, x_2, x_3, Where$x_i = |X|_{m_i}$, is obtained as

$$X = \left|\sum_{i=1}^{3} w_i x_i\right|_{m_1m_2m_3} \quad (4)$$

The computed weights w_1, w_2 and w_3 are directly implemented into the above equation in order to start the conversion process.

The weights$w_1 = \frac{(m_1+1)m_2m_3}{2}$, $w_2 = m_1(m_2-1)m_3$, $w_3 = \frac{m_1m_2(m_3+1)}{2}$

$$w_1 = \frac{(M+m_2m_3)}{2}, w_2 = M - m_1m_3, w_3 = \frac{(M+m_1m_2)}{2}$$

The M indicates that the Maximum dynamic range. The dynamic range can be calculated by the product terms of moduli set. The dynamic range causes the difficulties in the structure due to the number of multipliers, such an implementation requires large multipliers to obtain the X. A few modifications are possible that simplify the multipliers.

Substituting the weights in order to find the X value,

$$|X|_M = |A + m_1(m_2-1)m_3x_2 + C|_M \quad (5)$$

Where

$$A = \frac{(m_1+1)m_2m_3}{2}x_1 , C = \frac{m_1m_2(m_3+1)}{2}x_3$$

Substituting A, C and M values in equation (5) and then simplify

$$|X|_M = \left|\frac{(m_1+1)m_2m_3}{2}x_1 + m_1(m_2-1)m_3x_2 + \frac{m_1m_2(m_3+1)}{2}x_3\right|_M$$

$$|X|_M = \left|\frac{M}{2}(x_1+x_3) + \frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M \quad (6)$$

The first term, $x_1 + x_3$, in this equation can be either even or odd. If it is even then

$$\left|\frac{M}{2}(x_1+x_3)\right|_M = 0$$

If it is odd, then

$$\left|\frac{M}{2}(x_1+x_3)\right|_M = \frac{M}{2}$$

So if $(x_1 + x_3)$ is even, and then the equation may be written as

$$|X|_M = \left|\frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M$$

$$|X|_M = |M_1x_1M_2x_2 + M_3x_3|_M \quad (7)$$

And if $(x_1 + x_3)$ is odd, then

$$|X|_M = \left|\frac{M}{2} + \frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M$$

$$|X|_M = \left|\frac{M}{2} + M_1x_1M_2x_2 + M_3x_3\right|_M \quad (8)$$

Equations (7) and (8) are used to construct the hardware structure. Observe that the numbers involved in the multiplications of those equations are smaller than in the original equations. Therefore, in an implementation, the corresponding multipliers will be accordingly smaller.

The core function can be proved by theoretically for the varies inputs. From that we can conclude that the property of the core function is performed well.

3.2 New CRT Approach

This is another approach, which is used to prove the reverse converse theoretically. Comparing the CRT and the modified CRT, it can be noted that the modified CRT reduces the modulo base by m_1. Thus it leads to an efficient converter design. However for the moduli sets with a large size, the modified CRT is still slow. If the modulo base can be further reduced and the delay becomes independent of the size of the moduli sets, then a more efficient converter design can be obtained.

Given the moduli set $\{m_1, m2 \ldots m_n\}$, the residue number is converted into the binary number X by

$$X = x_1 + m_1\left|\sum_{i=1}^{n} w_i x_i'\right|_{m_n \ldots m_2} \quad (9)$$

Where n > 1, $w_1 = \frac{\left(M_2\left|M_2^{-1}\right|_{m_2} - 1\right)}{m_2}$, $w_i = \frac{M_i}{m_i}$, $x_1' = x_1$ and $x_i' = \left|M_i^{-1}\right|_{m_i} x_i$, for i=2, 3...n.

For Instance:

$m_1 = 7, m_2 = 6, m_3 = 5 \quad x_1 = 6, x_2 = 2, x_3 = 0 \quad M = 210$

$M = m_1 m_2 m_3, Mi = \frac{M}{m_i},$

$M1 = 30\,, M2 = 35\,,\ M3 = 42$

To find the multiplicative inverses

$|a^m|_m = |a|_m \ for\ all\ integers\ a$

The multiplicative inverses of $|a|_m$ for $|a|_m \neq 0$ is $|a^{m-2}|_m$

$$w_1 = \frac{\left(30|30^{-1}|_7 - 1\right)}{7};\ w_2 = \frac{\left(35|35^{-1}|_6\right)}{6};\ w_3 = \frac{\left(42|42^{-1}|_5\right)}{5}$$

$w_1 = 17$; $w_2 = 5$; $w_3 = 6$

$x_2' = |5 * 2|_6 = 4\,,\ x_3' = |3 * 0|_5 = 0$

These values are substitute in Equation (9) in order to find out the X value

$X = x_1 + m_1 |\sum_{i=1}^{n} w_i x_i'|_{m_n \ldots m_2}$

$X = 6 + 7|(17 * 6) + (20)|_{30}$

$X = 20$

This is the exact decimal equivalent value for the respective inputs of the moduli set. The New CRT Approach is proved by theoretically. The conversion takes critical calculations especially finding the multiplicative inverses. The weight value might be calculated to find the decimal equivalent value.

4. HARDWARE REALIZATION

A basic architecture for the implementation of Equations (7) and (8) is shown in the Figure. The value $x_1 + x_3$ need not actually be computed, since all that is required is the knowledge of whether it is odd or even; therefore, a half-adder on the least significant bits is sufficient. The sum-output bit of this half-adder is then used to select either 0 or M=2, according to whether that bit is 0 or 1.

Every CSA needs three inputs and two outputs. These two outputs are Sum and Carry. Those outputs will be the input for the next CSA. Initially four CSA Trees are integrated to process the inputs. The AND gate is used to set the condition for the addition of $(x_1 + x_3)$. The last stage is the most important stage, which is used to finalize the reverse conversion. Carry Propagate Adder is used for the last stage. This stage is also called as Mod-M CPA. This operation is the modulo operation and done by the Carry Propagate Adder, since it is called Mod- M CPA. The Carry Propagate adder has the low power consumption property. The inputs of the Mod-M CPA are in reduced form. Those two inputs are added together in order to get the decimal equivalent binary value.

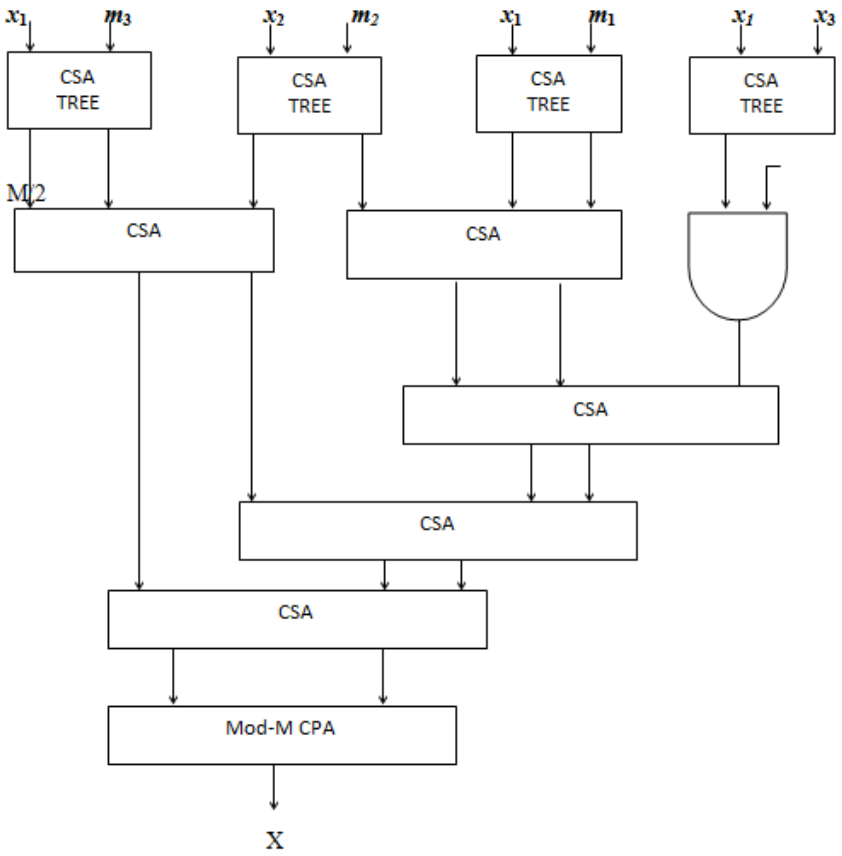

Fig.1 Hardware Structure

5. PERFORMANCE EVALUATION

This proposal requires five 3:1 adders, five 2:1 adders and one AND gate. This converter is less complex and faster than the state of the art equivalent converters. All the logic of each component is coded by using the VHDL language and this was implemented on the Xilinx Spartan 3 xa3s200-4-ftg256 FPGA, with Xilinx ISE 10.1.03. This reverse converter is implemented for a wide range of n, up to an equivalent dynamic range of 32 bits.

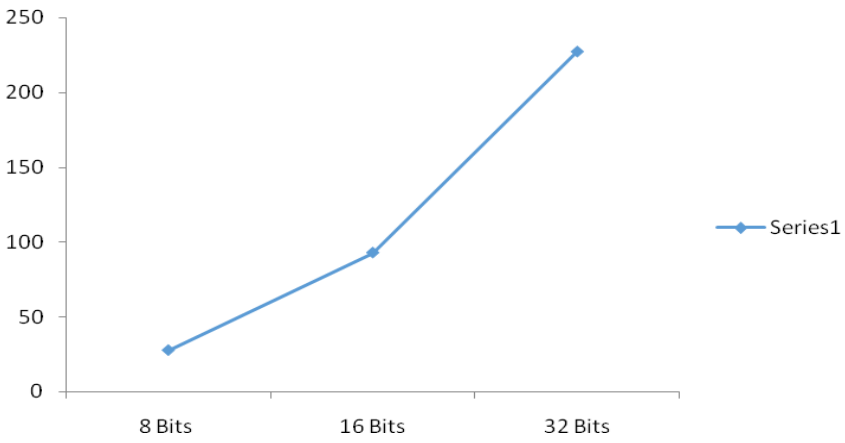

Fig. 2 Area for varies bit range

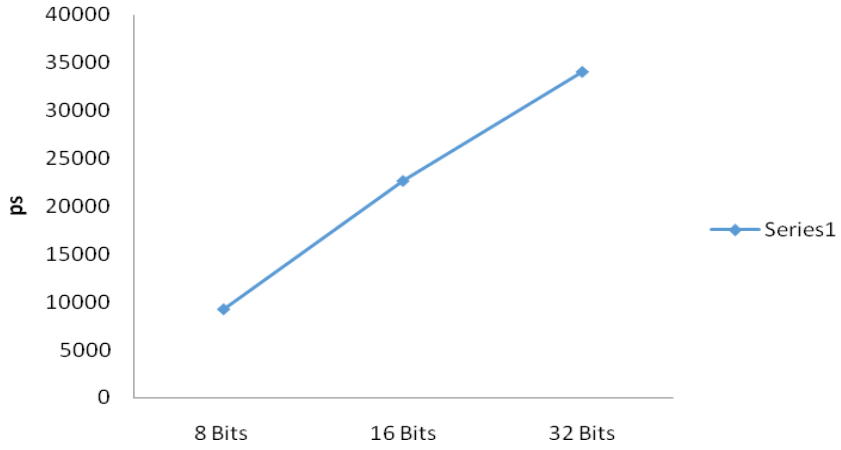

Fig. 3 Speed for varies bit range

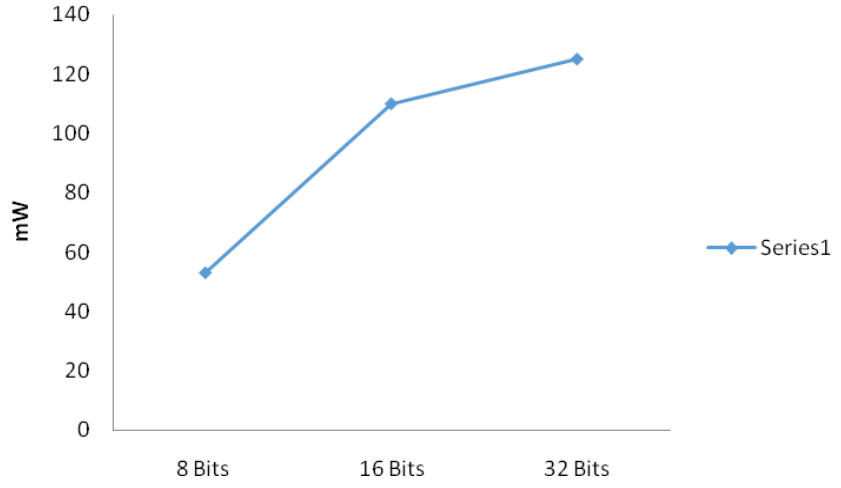

Fig.4 Power consumption for Different Bits

5.1 Comparison

	CRT			Core Functional Approach		
	Area	**Delay(ps)**	**Power(mW)**	**Area**	**Delay(ps)**	**Power(mW)**
2^8	17	14741	143	28	9248	53
2^{16}	69	30758	265	93	22635	110
2^{32}	135	42159	325	227	34015	175

TABLE I: Comparison Results in Terms of Area, Delay and Power

Comparison analysis is used to determine the best approach by comparing with the previous approaches. From the Table.1, We conclude that core functional approach is the best approach in terms of speed and power consumption. But there is a drawback, which occupies more area than the CRT method.

In core functional approach, we used more number of adders than the CRT. This is the reason behind that it occupies more area. But this is the best approach since it has very less delay and power consumption. When compared with area, the number of slices is increased in FPGA. It occupied some 35% of area in FPGA. Whereas in CRT method, it occupied 21% of area. So this is the only one drawback in Core Functional Approach. The power consumption also shows the best result as the speed. This approach consumes 175mW powers for its entire operation.

6. SUMMARY AND CONCLUSION

CSA gives better performance in terms of area, speed and power consumption. This is the reason that CSA dominates in this structure for the implementation. Core functional approach is best approach since it consumes very less time and power consumption. The Modified reverse converter is proposed for the moduli set {2n+1, 2n, 2n-1}. The Core Function, which at first glance appears to be a different approach, is in fact a variant of the CRT approach. This converter proposes the low complexity implementation that does not require the explicit use of modulo operation in the conversion process. The performance of the proposed converter is evaluated theoretically and this was involved in the process of FPGA implementation. The proposed converter outperforms the state of the art with about 86.15ns, 35% and 175mW in terms of conversion time , area cost and power consumption, respectively. The New CRT Approach was not implemented for this moduli set. It proved theoretically with the proper reverse conversion. In future, these moduli set can be implemented by the New CRT methodology, since it proved theoretically.

REFERENCES

[1] Kazeem Alagbe Gbolagade, George RazvanVoicu and Sorin Dan Cotofana, “An Efficient FPGA Design of Residue to Binary Converter” IEEE Trans. Very Large Scale Integration System, vol.19, no.8, August 2011.

[2] R. Conway and J. Nelson, “Improved RNS FIR filter architectures,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 51, no. 1, pp. 26–28, Jan. 2004.

[3] W. Wang, M. Swamy, M. Ahmad, and Y. Wang, “A study of the residue-to-binary converters for the three-moduli sets,” IEEE Trans.Circuits Syst. I, Reg. Papers, vol. 50, no. 2, pp. 235–243, Feb. 2003.

[4] M. Ahmad, Y.Wang, and M. Swamy, “Residue-to-binary number converters for three moduli sets,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 46, no. 7, pp. 180–183, Feb. 1999.

[5] A. Premkumar, “An RNS to binary converter in {2n+1, 2n, 2n-1}moduli set,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 39, no. 7, pp. 480–482, Jul. 1992.

[6] Pemmaraju V. Ananda Mohan, "RNS-to-Binary Converter for a new Three-Moduli set $\{2^{n+1}- 1, 2^n, 2^n-1\}$" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, VOL. 54, NO. 9, SEPTEMBER 2007.

[7] P. V. Ananda Mohan, Fellow, IEEE, and A. B. Premkumar, Senior Member, IEEE "RNS-to-Binary Converters for Two Four-Moduli Sets $\{2^n-1, 2^n, 2^n+1, 2^{n+1} -1\}$ and $\{2^n-1, 2^n, 2^n+1, 2^{n+1} +1\}$" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, VOL. 54, NO. 6, JUNE 2007.

[8] M.Abtahi and P.Siy, " Core Function of an RNS Number with No Ambiguity", ELSEVIER Computers and Mathematics with Applications 50 (2005) 459-470

[9] R. Conway and J. Nelson, "Improved RNS FIR filter architectures," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 51, no. 1, pp. 26– 28,Jan. 2004.

[10] M. Akkal and P. Siy, "A new mixed radix conversion algorithm MRCII," J. Syst. Arch., vol. 53, pp. 577–586, 2007.

