Reverse Conversion for Residue Number System Realizations of Digital Signal Processing Hardware

Negovan Stamenković , Bojan Jovanović, and Vidosav Stojanović

***Abstract*—One of the fundamental problems with residue arithmetic is the difficulty associated with residue-to-binary conversion. In recently published papers, a hardware implementation of the Mixed-Radix system was proposed for the translation of residue coded outputs into binary number for a digital filter realization. These methods are presented as a block diagram or data flow chart and require integrated circuits which are not commercially available and were to be constructed from a number of basic logic elements. This paper presents a more successful technique based on a mixed-radix conversion process. The residue to binary convertor can be built using commercially available elements. To validate these results, the converters are implemented in a Standard Logic Cell of FPGA technology.**

***Index Terms*—Digital signal processing, mixed radix conversion (MRC), powers of two related moduli set, residue number system (RNS)-to-binary conversion, reverse converters, VLSI architectures.**

I. Introduction

THE residue number system (RNS) is a carry-free system for addition, subtraction and multiplication operations. Hence, a large dynamic range binary system can be partitioned into several small wordlength channels in parallel. Thus, the RNS can result in a high speed operation [1]. Today RNS is one of the most popular technique for reducing the power dissipation and the computation load in VLSI system design. The challenges of the RNS system design lie in the choice of the moduli set and in the residue to binary (R/B) conversion.

The choice of moduli set is very important and necessary for nearly equal delay of the channels. Special moduli sets have been used extensively to reduce the hardware complexity in the implementation of converters and arithmetic operations [2]–[5]. Among which the triple moduli set $\{2^n-1, 2^n, 2^n+1\}$ has some benefits [4]. Because of operand lengths of these moduli, the operation delay of this system is determined by the modulo 2^n+1 channel. The latter means that, if we cut down the time required for modulo 2^n+1 addition, we also cut down the RNS addition time [6].

All available RNS to binary conversion methods are based on two techniques, namely: the Chinese remainder theorem (CRT) [7]–[9], and mixed-radix conversion (MRC) [1], [10]. A direct implementation of the CRT is inefficient since it is based on a large modulo M operation, where M is the dynamic range of the RNS. The MRC is a strictly sequential process and requires a long delay [11]. However, special moduli sets $2^n \pm 1$ and 2^n,have been used extensively to reduce the hardware complexity in the implementation of RNS architectures, especially for the residue-to-binary converters [5], [12]–[17]. Only block diagram and results using VLSI technology for these architecture are given. There is a large gap between the block diagrams and VLSI technology.

Zahvaljujemo se finansijskoj podršci Ministarstvu za nauku u okviru projekta pod brojem 18019.

N. Stamenković is with Faculty ofr Natural Science, 28220 K. Mitrovica, Lole Ribara 29, Serbia; e-mail: negovanstamenkovic@gmail.com.

B. Jovanovic and V. Stojanović are with Faculty of Electronic Engineering, Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia; e-mail: [bojan.jovanovic vidosav.stojanovic]@elfak.ni.ac.yu.

In this paper we propose MRC VHDL realization of reverse converter for the $\{2^n+1, 2^n, 2^n-1\}$ moduli set. Only standard integrated circuits can be used for convertor implementation.

The paper is organized as follows. In section II, we provide a short overview of residue number systems (RNS). Section III provides hardware implementation and section IV is simulation. Section V is Conclusion.

II. Background

Residue number system (RNS) is defined in terms of a set of relatively prime moduli set $\{m_i\}_{i=1,n}$ such that $\gcd(mi;mj)=1$ for $i \neq j$, 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 for unsigned residue number system. For signed residue number system dynamic range is $[-M/2, M/2-1]$. Thus in a signed number system the lower half of the dynamic range is reserved for the negative numbers.

The residues of a decimal number X can be obtained as

$$x_i = \begin{cases} \langle X \rangle_{m_i}, & \text{if} \quad X \geq 0 \\ m_i - \langle |X| \rangle_{m_i}, & \text{otherwise} \end{cases} \tag{1}$$

thus X can be represented in RNS as $X=\{x_1, x_2, x_3, \ldots, x_n\}$, $0 \leq x_i < m_i$. This representation is unique for any integer $X \in [-M/2, M/2-1]$. We note here that in this paper we use $\langle X \rangle_{m_i}$ to denote the X mod m_i operation and the operator $\circ$ to represent the operation of addition, subtraction, or multiplication. Given any two integer numbers X and Y in RNS represented by $X=\{x_1, x_2, x_3, \ldots, x_n\}$ and $Y=\{y_1, y_2, y_3, \ldots, y_n\}$, respectively, $Z=X \circ Y$, can be calculated as $Z=\{z_1, z_2, z_3, \ldots, z_n\}$, where $z_i = x_i \circ y_i$, for $i=1, \ldots, n$. This means that the complexity of the calculation of the $\circ$ operation is determined by the number of bits required to represent the residues and not by the one required to represent the input operands.

A. A residue adder and subtractor

The residue sum of two residue digits, $\langle x_i + y_i \rangle_{m_i}$, is the residue of the sum $x_i + y_i$ with respect to the modulus m_i. The operation may be defined as follows.

$$z_i = \langle x_i + y_i \rangle_{m_i} = \begin{cases} x_i + y_i, & \text{if} \quad x_i + y_i < m_i \\ x_i + y_i - m_i, & \text{if} \quad x_i + y_i \geq m_i \end{cases} \tag{2}$$

with $0 \leq x_i, y_i \leq m_i - 1$.

The subtraction operation may be defined as follows

$$z_i = \langle x_i - y_i \rangle_{m_i} = \begin{cases} x_i - y_i, & \text{if} \quad x_i \geq y_i \\ x_i - y_i + m_i, & \text{if} \quad x_i < y_i \end{cases} \tag{3}$$

with $0 \leq x_i, y_i \leq m_i - 1$.

B. Mixed Radix Conversion

The conversion from RNS to binary using Mixed Radix Conversion (MRC) can be formulated as follows [1]: given an n-digit binary number $X=\{u_1, u_2, u_3, \ldots, u_n\}$ in an RNS with the set of relatively prime integer moduli $\{m_i\}_{i=1,\ldots,n}$ one has to find a set of

digits $\{v_1, v_2, v_3, \ldots, v_n\}$, which are the mixed radix digits (MRD), such that Equation (4) holds true

$$X = v_1 + v_2 m_1 + v_3 m_1 m_2 + \cdots + v_n \prod_{i=1}^{n-1} m_i \tag{4}$$

were are m_1, $m_1 m_2$, …, $\prod_{i=1}^{n-1} m_i$ mixed-radix.

The mixed radix digits v_i, $0 \le v_i < m_i$ can be computed as follows [18]:

$$\begin{aligned} v_1 &= u_1 \\ v_2 &= \big\langle (u_2 - v_1) c_{12} \big\rangle_{m_2} \\ v_3 &= \big\langle ((u_3 - v_1) c_{13} - v_2) c_{23} \big\rangle_{m_3} \\ &\vdots \\ v_n &= \big\langle (((\cdots (u_n - v_1) c_{1n} - v_2) c_{2n} - \ldots - v_{n-1} c_{n-1,n} \big\rangle_{m_n} \end{aligned} \tag{5}$$

where $c_{i,j}$ for $1 \le i \le j < n$ is the multiplicative inverse of m_i modulo m_j, or $\langle c_{ij} \times m_i \rangle_{m_j} = 1$. If the mixed-radix digits are given, any number in the interval $[-M/2, M/2-1]$ can be uniquely represented.

Thus, the MRC is sequential and involves modulo subtractions and modulo multiplication by multiplicative inverses of one modulus with respect to the remaining moduli.

C. Chinese Reminder Theorem

The residue number $\{x_1, x_2, x_3, \ldots, x_n\}$ can be converted into the decimal number X, according to the Chinese Reminder Theorem (CRT), as follows [1]:

$$X = \Big\langle \sum_{i=1}^{n} M_i \langle M_i^{-1} \rangle_{m_i} x_i \Big\rangle_M \tag{6}$$

where $M_i = M/m_i$, and M_i^{-1} is the multiplicative inverse of M_i with respect to m_i.

The CRT requires a large modulo operation of size M, which is not efficient.

III. Hardware Implementation

A. Adder and Subtractor

Bayoumi and Jullien [19] describe a modular adders that utilizes two binary adder and a multiplexer, shown in Fig. 1(a). The first adder ads x and y. The second adder adds the sum to $2^n - m$. The carry bit generated from the second adder indicates whether or not $(x+y)$ is greater than m.

Modulo m substraction can be implemented by two cycles: substraction and addition. The first cycle is substraction two inputs signals x and y, and the second cycle adds a correction factor of m to the difference rom the first cycle. The borrow bit generated from the subtractor indicates whether or not x is greater than y. This implementation is shown in Fig. 1(b).

B. The Reverse Converter

Given the RSN numbers $\{u_1, u_2, u_3\}$ with respect to moduli set $\{2^n+1, 2^n, 2^n-1\}$ the proposed algorithm computes the binary equivalent of this RNS number using mixed radix conversion technique. First, we demonstrate that the computation of the multiplicative inverses can be eliminated for this moduli set.

If it can be demonstrated that $\langle c_{12}(2^n+1)\rangle_{2^n} = 1$, then $c_{12} = 1$ is the multiplicative inverse of 2^n+1 with respect to 2^n

$$\langle c_{12}(2^n+1)\rangle_{2^n} = \langle 2^n+1\rangle_{2^n} = 1$$

thus, $c_{12} = 1$ holds true.

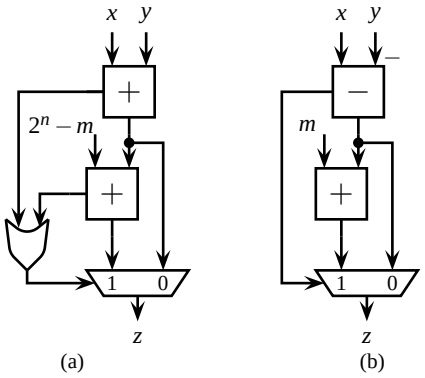

Fig. 1. Modulo adder (a) and modulo subtractor (b) with serial method.

In the same way, if it can be shown that $\langle c_{23} 2^n \rangle_{2^n-1} = 1$, then $c_{23} = 1$ is the multiplicative inverse of 2^n with respect to $2^n - 1$.

$$\langle c_{23} 2^n \rangle_{2^n-1} = \langle 2^n \rangle_{2^n-1} = 1$$

thus, $c_{23} = 1$ holds true.

Again, if it can be proved that $\langle c_{13}(2^n+1)\rangle_{2^n-1} = 1$, then 2^{n-1} is the multiplicative inverse of 2^n+1 with respect to 2^n-1.

$$\begin{aligned} \langle c_{13}(2^n+1)\rangle_{2^n-1} &= \langle 2^{n-1}(2^n+1)\rangle_{2^n-1} \\ &= \langle 2^{n-1}(2^n-1+2)\rangle_{2^n-1} \\ &= \langle 2^n \rangle_{2^n-1} = 1 \end{aligned}$$

thus, $c_{13} = 2^{n-1}$ holds true.

Next, we assume the residue representation of number $X = \{u_1, u_2, u_3\}_{RNS\{m_1,m_2,m_3\}}$, and by substituting $c_{12} = 1$, $c_{13} = 2^2$, and $c_{23} = 1$ in Eq (4), we obtain following expressions

$$Y = v_1 + v_2 m_1 + v_3 m_1 m_2 \tag{7}$$

where

$$\begin{aligned} v_1 &= u_1 \\ v_2 &= \langle u_2 - v_1 \rangle_{m_2} \\ v_3 &= \langle 2^{n-1}(u_3 - v_1) - v_2 \rangle_{m_3} \end{aligned} \tag{8}$$

The hardware realization of MRC algorithm (7) is presented in Fig. 2 for the three moduli set. The first mixed radix digit is u_1. The first step in MRC algorithm is to obtain $v_2 = \langle u_2 - u_1 \rangle_{m_2}$ and $v_{13} = \langle u_3 - u_1 \rangle_{m_3}$. In the next step, v_{13} is multiplied mod m_3 with the multiplicative inverse of m_3 with respect to m_1. This yields the third mixed radix digit v'_{13}. The desired binary number next can be obtained as $\langle v'_{13} - v_2 \rangle_{m_3}$.

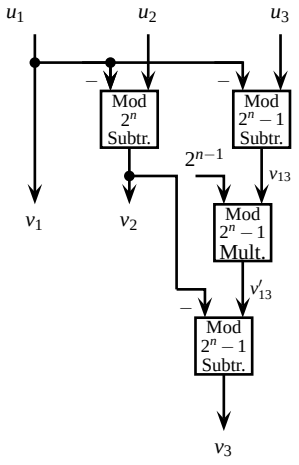

Fig. 2. Mixed-Radix conversion for $\{2^n+1, 2^n, 2^n-1\}$ moduli set.

Block for RNS multiplication v_{13} with 2^{n-1} is very simple. The bits of v_{13} are shifted left (multiplied by 2^{n-1}) to achieve properly result. The length of this register is $2n$ bits. Binary numbers on $2n$ bits can be represented as follows:

$$\begin{aligned} v'_{13} &= \langle k_1 2^n + k_0 \rangle_{2^n-1} \\ &= \langle k_1(2^n - 1 + 1) + k_0 \rangle_{2^n-1} = k_1 + k_0 \end{aligned} \tag{9}$$

where k_1 and k_0 are the corresponding high and low n bits word, respectively. Thus

$$v'_{13} = \langle 2^{n-1} v_{13} \rangle_{2^n-1} = k_1 + k_0$$

We now illustrate this procedure for $n = 6$ and $v_{13} = \langle u_3 - u_1 \rangle_{63} = 10$. The binary representation of v_{13} is $[0001010]$. After multiplication with 2^5 binary number it is $[000101\ 000000]$. Thus last six bits are $k_0 = 000000_2 = 0$ and first six bits are $k_1 = 000101_2 = 5$. Hence $v'_{13} = \langle v_{13} \times 2^5 \rangle_{63} = k_1 + k_0 = 5$.

Distributed arithmetic (DA) is used to compute inner product (7) because of efficiency DA architecture. In a DA architecture, the multiplier is eliminated by employing a memory to store linear combinations of mixed radix digits. Figure 3 shows DA-based implementation of a 3-moduli set residue to binary conversion. The DA architecture consists of four small units: the shift register unit, the DA base unit, the adder/shifter unit, and positive/negative number separation unit.

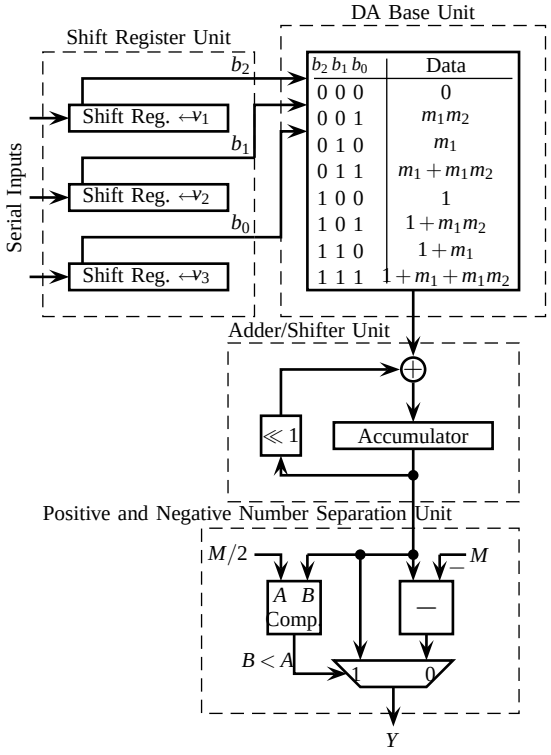

Fig. 3. LUT-based DA implementation for a 3set residue to binary converter. The DA architecture consists of four small units: the shift register unit, the DA base unit, the adder/shifter unit, and positive/negative number separation unit.

The memory addresses are formed by grouping bits in the same bit position from successive input mixed radix digits. The input is shifted in one bit left at a time into registers v_i, $i = 1,2,3$. The output is available from the accumulator once every block $\{b_2\,b_1\,b_0\}$. The size of memory block is 2^3 words. Data is shifted every clock cycle and the LUT outputs are shifted and accumulated.

Adder/Shifter unit contain standard adder, accumulator and shift register. The dynamic range is $M = 2^n(2^{2n} - 1) \simeq 2^{3n}$. Hence size of accumulator is $3n$ (18) bits. This is done m times where m is the length of shift registers v_i. At the end of every $m-1$ clock cycles, the output is tapped at properly value Y.

IV. MATLAB® AND VHDL SIMULATION

The performance of the proposed converter is evaluated by both performing a theoretical analysis using MATLAB® software and experimentally by implementing it on an FPGA chip.

A. MATLAB® simulation

Let the given RNS number be $X = \{30,32,19\}_{RNS\{65,64,63\}}$ in binary form. We convert this number representation into the mixed-radix number representation with v_3, v_2, v_1 using Mixed-Radix convertor shown in Figure 2: $v_1 = 30$, $v_2 = 2$ and $v_3 = 24$, or in binary representation these are

v_1	0011110	b_2
v_2	0000010	b_1
v_3	0011000	b_0

The contents of of the 8-word ROM are shown in Table I. A scaling accumulator multiplier performs multiplication using an iterative shift-add routine.

TABLE I
LOOKUP TABLE FOR $\{65,64,63\}$ MODULI SET

Address	LUT Output	
000	0	0
001	m_1m_2	4160
010	m_1	65
011	$m_1+m_1m_2$	4225
100	1	1
101	$1+m_1m_2$	4161
110	$1+m_1$	66
111	$1+m_1+m_1m_2$	4226

Thus, seven addresses $b_2b_1b_0$ are: 000, 000, 101, 101, 100, 110 and 000. The output result is

$$\begin{aligned} Y =& 0 + (66 + (1 + (4161 + (4161 \\ &+ (0 + 0 \times 2) \times 2) \times 2) \times 2) \times 2) \times 2 = 100\,000 \end{aligned} \tag{10}$$

It is shown, after six shifts and accumulates, the output word Y is finished.

B. VHDL simulation

The proposed implementation is programmed (Described) and implemented using VHDL language which is a Hardware Description Language that was developed by the Institute of Electrical and Electronic Engineers (IEEE) as a standard language for describing the structure and behavior of digital electronic systems. A 18-bit precision reverse converter using 7-bit moduli set $\{65,64,63\}$ was implemented as a VHDL model in 0.35 micron CMOS technology using Synopsys tools.

Each part of residue to binary converter is described in VHDL programming language and implemented in EP2C35F672C6N FPGA chip on Altera DE2 board. VHDL simulation of forward RNS converter is shown in Fig. 4. There are five RNS number $(30,32,19)$, $(54,27,38)$, $(62,48,22)$, $(50,24,62)$ and $(3,25,55)$. According to the Figure 4, the results of VHDL implementation are: 100 000, 77339, -25808, -130600 and 18073, respectively.

The results of implementation performance are shown in Table II.

From Table II can be seen that converter does not require a lot of FPGA resources (less than 1%) and has a maximal throughput of 277.47 MHz.

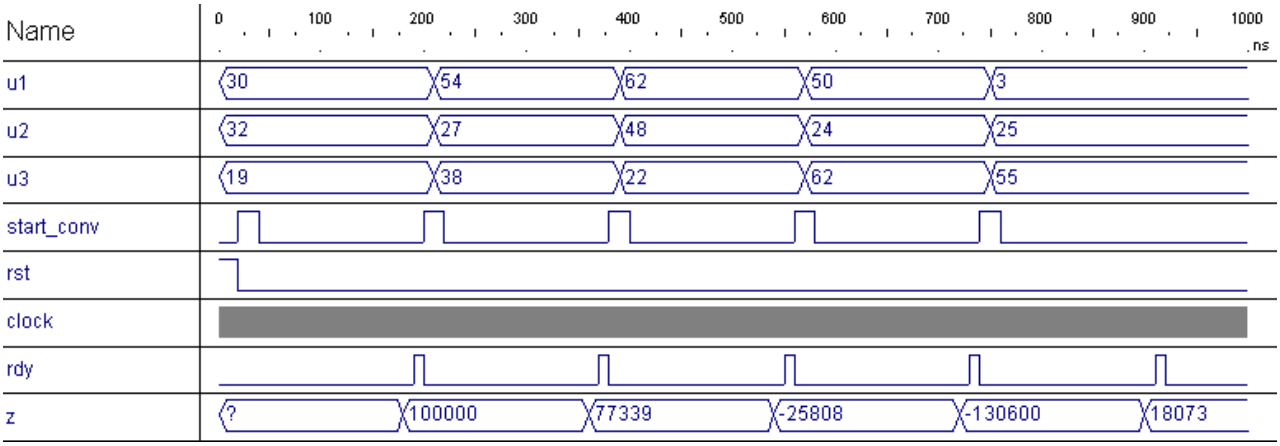

Fig. 4. Residue to binary converter VHDL simulation results.

TABLE II
FPGA IMPLEMENTATION OF REVERSE CONVERTER.

	Logic elem.	% of FPGA	F_{max} [MHz]
Converter	255/33216	< 1 %	277.47

V. CONCLUSION

A VHDL implementation for residue to binary converter using popular moduli set $\{2^n+1, 2^n, 2^n-1\}$ is presented in this paper. We used an MRC technique for efficient RNS to binary conversion. First, we demonstrate that the computation of the required multiplicative inverse can be simplified. Next, corresponding formulas for the conversion process which can be easily implemented using only three subtractors have been derived. It has been shown that the overall conversion process will require 2 levels of subtractors and one level multiplier. Distributed arithmetic for compute for computing inner product value of mixed-radix number is used. Only standard integrated circuits can be used for reverse convertor implementation.

The performance of proposed converter architecture are determined using MATLAB® software and VHDL programming language.

REFERENCES

[1] N. Szabo and R. I. Tanaka, *Residue Arithmetic and its Application to Computer Technology*. New York: McGraw-Hill, 1967.

[2] E. Gholami, R. Farshidi, and M. H. end Keivan Navi, "High speed residue number system comparison for the moduli set $\{2^n-1, 2^n, 2^n+1\}$," *Journal of Communication and Computer*, vol. 6, no. 3, pp. 40–46, 2009.

[3] S. Chen and S. Wei, "A high-speed realization of chinese remainder theorem," in *Proc. of the 2007 WSEAS Int. Conference on Circuits, Systems, Signal and Telecommunications*, Gold Coast, Australia, Jan. 17–19 2007, pp. 97–102.

[4] B. Cao, C.-H. Chang, and T. Srikanthan, "A residue-to-binary converter for a new five-moduli set," *IEEE Trans. on Circuits And SystemsI: Regular Papers*, vol. 54, no. 5, pp. 1041–1049, May 2007.

[5] S. Andraos and H. Ahmad, "A new efficient memoryless residue to binary converter," *IEEE Transactions On Circuits And Systems*, vol. 35, no. 11, pp. 1441–1444, Nov. 1988.

[6] T.-B. Juang, C.-C. Chiu, and M.-Y. Tsai, "Improved area-efficient weighted modulo 2^n+1 adder design with simple correction schemes," *IEEE Transactions on Circuits and SystemsII: Express Briefs*, vol. 57, no. 3, pp. 198–202, Mar. 2010.

[7] R. M. Capocelli and R. Giancarlo, "Efficient vlsl networks for converting an integer from binary system to residue number system and vice versa," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 11, pp. 1425–1430, Nov. 1988.

[8] S. J. Meehan, S. D. O'Neil, and J. J. Vaccaro, "An universal input and output RNS converter," *IEEE Transactions on Circuits and Systems*, vol. 37, no. 6, pp. 799–803, June 1990.

[9] J. Y. Kim, K. H. Park, and H. S. Lee, "Efficient residue-to-binary conversion technique with rounding error compensation," *IEEE Transactions on Circuits and Systems*, vol. 38, no. 3, pp. 315–317, Mar. 1991.

[10] K. A. Gbolagade, "A memoryless MRC technique for RNS-to-binary conversion using the moduly set $(2^n, 2^n-1, 2^{n-1}-1)$," *Int. Journal of Soft Computing*, vol. 4, no. 3, pp. 127–130, 2009.

[11] S. Bi and W. J. Gross, "The mixed-radix chinese remainder theorem and its applications to residue comparison," *IEEE Transactions on Computers*, vol. 57, no. 12, pp. 1624–1632, Dec. 2008.

[12] K. M. Ibrahim and S. N. Saloum, "The digit parallel method for fast rns to weighted number system conversion for specific moduli $(2^k-1, 2^k, 2^k+1)$," *IEEE Transactions on Circuits And Systems*, vol. 35, no. 9, pp. 1156–1158, Sept. 1988.

[13] S. J. Piestrak, "A high-speed realization of a residue to binary number system converter," *IEEE Transactions on Circuits And Systems-II: Analog and Digital Signal Processing*, vol. 42, no. 10, pp. 661–663, Oct. 1995.

[14] W. K. Jenkins and B. Leon, "The use of residue number systems in the design of finite impulse response digital filters," *IEEE Trans. on Circuits and Systems*, vol. CAS-24, no. 4, pp. 191–201, Apr. 1977.

[15] W. Wang, M. Swamy, M. Ahmad, and W. Wang, "A study of residue to binary converters for the three-moduli sets," *IEEE Trans. on Circuits and Syst-Fundamental Theory and Applications*, vol. 50, no. 2, pp. 235–245, 2003.

[16] D. Gallaher, F. E. Petry, and P. Srinivasan, "An efficient residue to binary converter design," *IEEE Transactions on Circuits And Systems-II: Analog and Digital Signal Processing*, vol. 44, no. 1, pp. 53–57, Jan. 1997.

[17] A. Hiasat and A. Zohdy, "Residue-to-binary arithmetic converter for the moduly set $(2^k, 2^k-1, 2^{k-1}-1)$," *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 45, no. 2, pp. 204–209, Feb. 1998.

[18] A. Sabbagh and K. Navi, "Two-level implementation of the residue to binary converter for the 4-moduli set $\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$," in *Proceedings of the International Conference on Electrical Engineering and Informatics*, Gold Coast, Australia, Jan. 17–19 2007, pp. 927–929.

[19] M. A. Bayoumi, G. A. Jullien, and W. C. Miller, "A VLSI implementation of residue adders," *IEEE Transactions on Circuits and Systems*, vol. CAS-34, no. 3, pp. 284–288, Mar. 1987.