A residue-to-binary converter for a new five-moduli set

Cao, Bin; Chang, Chip Hong; Srikanthan, Thambipillai

2007

Cao, B., Chang, C. H., & Srikanthan, T. (2007). A residue-to-binary converter for a new five-moduli set. *IEEE Transactions on Circuits and Systems-I: Regular Papers*, 54(5), 1041-1049.

<https://hdl.handle.net/10356/91531>

<https://doi.org/10.1109/TCSI.2007.890623>

IEEE Transactions on Circuits and Systems-I: Regular Papers © copyright 2007 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE. This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder. <http://www.ieee.org/portal/site>.

A Residue-to-Binary Converter for a New Five-Moduli Set

Bin Cao, Chip-Hong Chang, *Senior Member, IEEE*, and Thambipillai Srikanthan, *Senior Member, IEEE*

Abstract—The efficiency of the residue number system (RNS) depends not only on the residue-to-binary converters but also the operand sizes and the modulus in each residue channel. Due to their special number theoretic properties, RNS with a moduli set consisting of moduli in the form of $2^n \pm 1$ is more attractive than those with other forms of moduli. In this paper, a new five-moduli set RNS $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1, 2^{n-1} - 1\}$ for even n is proposed. The new moduli set has a dynamic range of $(5n - 1)$ bits. It incorporates two additional moduli to the celebrated three-moduli set, $\{2^n - 1, 2^n, 2^n + 1\}$ with VLSI efficient implementations for both the binary-to-residue conversion and the residue arithmetic units. This extension increases the parallelism and reduces the size of each residue channel for a given dynamic range. The proposed residue-to-binary converter relies on the properties of an efficient residue-to-binary conversion algorithm for $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and the mixed-radix conversion (MRC) technique for the two-moduli set RNS. The hardware implementation of the proposed residue-to-binary converter employs adders as the primitive operators. Besides, it can be easily pipelined to attain a high throughput rate.

Index Terms—Mixed-radix conversion (MRC), residue arithmetic, residue number system (RNS), residue-to-binary converter, VLSI.

I. INTRODUCTION

THE inherent carry-free operations, parallelism, and fault-tolerant properties of the residue number system (RNS) have made it an important candidate for high-performance and fault-tolerant applications [1], [2]. During the past decade, the RNS has received considerable attention in computationally intensive applications where the key operations required are addition, subtraction and multiplication, such as digital filters, correlators, fast Fourier transform (FFT), image processing, and digital communications [3]–[8].

To interface the highly parallel residue arithmetic based processing with the prevailing binary number system, a typical RNS-based processor must possess at least two components: the binary-to-residue converter which converts the binary data into their residue representations, and the residue-to-binary converter, which converts the RNS-represented results into their binary weighted representations. Besides these two overhead components, a typical RNS should include the residue arithmetic units (RAUs) which perform the necessary arithmetic operations required by the applications. The degree of RNS's

complexity involved in the hardware implementation of each component depends largely on the moduli set. The architectures of the RNS for general moduli sets have been widely studied. Well-balanced residue channels with low wastage on dynamic ranges are readily obtained from general moduli sets. However, due to the lack of special number theoretic properties of general moduli sets, the residue-to-binary converters and RAUs for the general moduli set RNS are usually implemented with large number of adders and ROMs, which are area intensive and computationally inefficient, particularly for ASIC implementation of RNS involving large dynamic range [1], [2], [9]–[13]. Although the cost of memory has been driven low nowadays, the number of ROMs and the access time incurred by the need to read these ROMs iteratively make their implementations in application-specific integrated circuit (ASIC) implementations unfavorable.

Special moduli sets have been used extensively to reduce the hardware complexity in the implementation of RNS architectures, especially for the residue-to-binary converters [14]–[32]. Among the special moduli sets, those employ moduli of the forms 2^n and $2^n \pm 1$ are the most popular choices. This type of moduli not only eases the design of efficient binary-to-residue and residue-to-binary converters, but the modular arithmetic operators in their RAUs can also be readily realized with highly efficient architectures, such as the modular adders and modular multipliers in [33]–[42]. The three-moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ has gained unprecedented popularity by virtue of its inherent number theoretic properties in Chinese Remainder Theorem (CRT), and several very efficient memoryless residue-to-binary converters have been proposed for this triple moduli set [14], [16], [21]–[23].

In some high-performance and fault-tolerant signal processing applications [7], [43], [44], the level of parallelism provided by the three-moduli set is compromised by the increased dynamic range. The demand for the increased granularity of parallelism and dynamic range calls for the necessity to expand the three-moduli set. Moduli sets obtained by adding moduli in the form of $2^n \pm 1$ to the celebrated three-moduli set are called supersets. Four-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} + 1\}$ was proposed by Bhardwaj *et al.* in [24], but two of the moduli are in the form of $2^n + 1$, causing the excess of the dynamic range comparatively larger. Vinod *et al.* proposed a more efficient four-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ [25], and its residue-to-binary converters are improved by Cao *et al.* [26] using the efficient conversion algorithm for the three-moduli set $\{2^n - 1, 2^n, 2^n + 1\}$. To minimize the waste of dynamic range, the same authors also proposed a new supplementary four-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ in [26] with compatible conversion efficiency in its residue-to-binary converter.

Manuscript received January 28, 2004; revised November 13, 2004, and November 15, 2005. This paper was recommended by Associate Editor S.-G. Chen.

B. Cao is with Seagate Technology International, Singapore 569877 (e-mail: caobinec@yahoo.com.cn).

C. H. Chang and T. Srikanthan are with Nanyang Technological University, Singapore 639798 (e-mail: echchang@ntu.edu.sg; astsrikan@ntu.edu.sg).

Digital Object Identifier 10.1109/TCSI.2007.890623

Skavantzios *et al.* proposed a class of conjugate moduli sets in [20]. Although these are high-cardinality sets, the moduli are not pairwise relatively prime, resulting in reduced dynamic ranges, asymmetric moduli channel length and long conversion delay. Skavantzios also proposed a five-moduli set $\{2^n - 1, 2^n, 2^n + 1, 2^n + 2^{(n+1)/2} + 1, 2^n - 2^{(n+1)/2} + 1\}$ [28]. As the two extended moduli are not in the form of 2^n or $2^n \pm 1$, it is difficult to design VLSI efficient architectures for the binary-to-residue conversion and the modular arithmetic operators. To minimize the hardware cost, reuse of the modular multipliers from the RNS arithmetic processing unit has been exploited. In [29], Mathew *et al.* proposed a five-moduli set $\{2^{3n}, 2^{3n} - 1, 2^{3n} + 1, 2^{3n} - 2^{(3n+1)/2} + 1, 2^{3n} + 2^{(3n+1)/2} + 1\}$. It has the same disadvantages as [28]. Hiasat [32] proposed a similar five-moduli set, where the residue-to-binary converter is more efficient than those for [28] and [29]. In [31], Skavantzios and Stouraitis proposed a new five-moduli set $\{2^{n+1}, 2^n - 1, 2^n + 1, 2^{n+1} - 1, 2^{n+1} + 1\}$, where all the moduli are in the form of 2^n or $2^n \pm 1$. Therefore, its RAU is efficient than those of [28], [29], [32]. The disadvantage is that it is not co-prime for any value of n , resulting in the reduced dynamic range and increased complexity of the residue-to-binary conversion algorithm. There are two moduli in the form of $2^n + 1$. According to [33]–[41], the operations modulo $2^n - 1$ are more efficient than those for modulo $2^n + 1$, therefore, it is better to reduce the number of moduli in the form of $2^n + 1$.

In this paper, we propose a new five-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1, 2^{n+1} + 1\}$, which is valid for even values of n . This moduli set is efficient for both the binary-to-residue converters and for the modular operations, as the moduli are in the form of 2^n or $2^n \pm 1$. Furthermore, there is only one modulus in the form of $2^n + 1$. Our proposed algorithm for its residue-to-binary converter is based on the mixed-radix conversion (MRC) technique and the conversion algorithm for the four-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ [26], [27] wherein an efficient residue-to-binary converter for the popular three-moduli set proposed in either [22] or [23] has been adopted. The fundamental idea of decomposing the moduli set is borrowed from the pioneering work of New Chinese Remainder Theorem first introduced in [15]. The derived residue-to-binary converter is based on adders. Such adder-based design has the advantage of being design automation friendly and can be readily pipelined to suit the throughput rate constrained by the application.

II. BACKGROUND

In a RNS, an integer X can be represented by an n -tuple of residues, $(x_1, x_2, \dots, x_n)$, defined over a set of pairwise relatively prime moduli $\{P_1, P_2, \dots, P_n\}$. The residue x_i is a smaller weighted binary number $x_i = |X|_{P_i}$, where the operation $|x|_m$ denotes x modulo m , i.e., the remainder of the integer x divided by the integer m . The residue-to-binary conversion can be performed using the CRT [1]

$$X = \left| \sum_{i=1}^n |1/\hat{M}|_{P_i} \cdot x_i \cdot \hat{M}_i \right|_M \quad (1)$$

where $M = \prod_{i=1}^n P_i$, $\hat{M}_i = M/P_i$, and $|1/\hat{M}|_{P_i}$ is the multiplicative inverse of $\hat{M}_i$ modulo P_i .

MRC can also be used for the calculation of X [1]. For a simple two-moduli set $\{P_1, P_2\}$, the integer X can be converted from its residue representation (x_1, x_2) by MRC as follows:

$$X = a_1 + a_2 P_1 = x_1 + P_1 \cdot |(x_2 - x_1)|1/P_1|_{P_2}|_{P_2} \quad (2)$$

where $|1/P_1|_{P_2}$ is the multiplicative inverse of P_1 modulo P_2 , and the coefficients a_1 and a_2 are the mixed-radix digits of X .

If the integers X and Y have RNS representations $(x_1, x_2, \dots, x_n)$ and $(y_1, y_2, \dots, y_n)$ respectively, then the RNS representation of $Z = X \diamond Y$ is given by $(z_1, z_2, \dots, z_n)$, where $z_i = |x_i \diamond y_i|_{P_i}$, and $\diamond$ denotes one of the operations of addition, subtraction or multiplication [1]. This means that arithmetic operation on large numbers can be performed by a collection of smaller arithmetic operations, and these operations can be performed in each residue channel concurrently and independently without inter-channel carry propagation.

III. RESIDUE-TO-BINARY CONVERTER FOR THE PROPOSED RNS

The aim of this section is to establish the number theoretic framework for the efficient conversion of the residue number represented in the proposed superset to its binary equivalent. We decompose the superset S into two subsets, $S_1 = \{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and $S_2 = \{2^n(2^{2n} - 1)(2^{n+1} - 1), 2^{n-1} - 1\}$. Being a new moduli set, we shall first prove that it is pairwise prime. It is well acknowledged that the moduli of $S_1 = \{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ are pairwise prime when n is even [25], [26]. Therefore, we shall prove that the elements of S_1 are all relative prime to the fifth element of S , $2^{n-1} - 1$ for even value of n . As 2^n is an even number, it is apparent that it is relative prime to $2^{n-1} - 1$. Assume that $2^n - 1$ is not relative prime to $2^{n-1} - 1$, and let $\gcd(2^n - 1, 2^{n-1} - 1) = k$ for some integer $k > 1$. Then we have $(2^n - 1) - (2^{n-1} - 1) = m \cdot k$, where m is an integer. As both $2^n - 1$ and $2^{n-1} - 1$ are odd numbers, their common divisor cannot be an even number. Therefore, k must be odd and $m \cdot k$ can never be a power-of-two term. However, $(2^n - 1) - (2^{n-1} - 1) = 2^{n-1}$ which contradicts the hypothesis that $(2^n - 1) - (2^{n-1} - 1) = m \cdot k$ is not a power-of-two term. Therefore, the assumption of $2^n - 1$ is not pairwise prime to $2^{n-1} - 1$ is incorrect. This proves that $2^n - 1$ is pairwise prime to $2^{n-1} - 1$. The pairwise primality of other moduli of S_1 with $2^{n-1} - 1$ can be proven in a similar manner.

As the superset S can be decomposed into S_1 and S_2 , a residue-to-binary converter of the four moduli superset, $S_1 = \{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ can be used to recover an interim integer $X^{(2)}$ from the residues x_1, x_2, x_3, x_4 of S_1 [26]. The final binary equivalent, X of $(x_1, x_2, x_3, x_4, x_5)$ in S can then be calculated from the residues $(X^{(2)}, x_5)$ correspond to the two-moduli set $S_2 = \{2^n(2^{2n} - 1)(2^{n+1} - 1), 2^{n+1} - 1\}$ by (2).

According to the conversion algorithm of [26], the interim integer $X^{(2)} = (x_1, x_2, x_3, x_4)$ can be calculated by

$$X^{(2)} = x_2 + 2^n(2^{2n}Z + 2^nY_2 + Y_1 - Z) = x_2 + 2^n \cdot T \quad (3)$$

where Y_1, Y_2 are the output signals of the residue-to-binary converter for the triple moduli set $\{2^n - 1, 2^n, 2^n + 1\}$, Z is the output of the residue-to-binary converter for the four-moduli superset S_1 . Therefore, T is a $(3n + 1)$ -bit integer.

By applying (2) to the resultant S_2 , the binary equivalent X of the proposed superset can be obtained from its residues by

$$X = X^{(2)} + 2^n(2^{2n} - 1)(2^{n+1} - 1)|k_0(x_5 - X^{(2)})|_{2^{n-1}-1} \quad (4)$$

where k_0 is the multiplicative inverse of $2^n(2^{2n} - 1)(2^{n+1} - 1)$ modulo $2^{n-1} - 1$, thus

$$|k_0 2^n(2^{2n} - 1)(2^{n+1} - 1)|_{2^{n-1}-1} = 1. \quad (5)$$

Since $|2^n|_{2^{n-1}-1} = |2(2^{n-1} - 1) + 2|_{2^{n-1}-1} = |2|_{2^{n-1}-1}$, $|2^{2n} - 1|_{2^{n-1}-1} = |2^{n+1}(2^{n-1} - 1) + 2 \cdot 2^n - 1|_{2^{n-1}-1} = |3|_{2^{n-1}-1}$ and $|2^{n+1} - 1|_{2^{n-1}-1} = |2^2(2^{n-1} - 1) + 3|_{2^{n-1}-1} = |3|_{2^{n-1}-1}$, the left-hand side (LHS) of (5) can be simplified as follows:

$$|k_0 2^n(2^{2n} - 1)(2^{n+1} - 1)|_{2^{n-1}-1} = |k_0 \cdot 2 \cdot 3 \cdot 3|_{2^{n-1}-1} = |18k_0|_{2^{n-1}-1}. \quad (6)$$

The solution to (5) is provided by the following Lemma.

Lemma 1: For any even number $n(n > 2)$, the solution of the modular equation

$$|18 \cdot k_0|_{2^{n-1}-1} = 1 \quad (7)$$

is given by the following.

When $n = 6k - 2, k = 1, 2, 3, \dots$

$$k_0 = 2^{n-3} + \frac{2}{9}(2^{n-4} - 1) = 2^{6k-5} + \frac{2}{9}(2^{6k-6} - 1). \quad (8)$$

When $n = 6k, k = 1, 2, 3, \dots$

$$k_0 = 2^{n-2} + \frac{1}{9}(2^{n-1} - 5) = 2^{6k-2} + \frac{1}{9}(2^{6k-1} - 5). \quad (9)$$

When $n = 6k + 2, k = 1, 2, 3, \dots$

$$k_0 = 2^{n-2} + \frac{8}{9}(2^{n-2} - 1) = 2^{6k} + \frac{8}{9}(2^{6k} - 1). \quad (10)$$

Proof of (8): When $n = 6k - 2$, LHS of (7) becomes

$$\begin{aligned} & \left| 18 \cdot \left(2^{6k-5} + \frac{2}{9}(2^{6k-6} - 1) \right) \right|_{2^{n-1}-1} \\ &= |5 \cdot 6^{6k-3} - 4|_{2^{6k-3}-1} \\ &= |5 \cdot (6^{6k-3} - 1) + 1|_{2^{6k-3}-1} \\ &= 1. \end{aligned}$$

Proof of (9): When $n = 6k$, LHS of (7) becomes

$$\begin{aligned} & \left| 18 \cdot \left(2^{6k-2} + \frac{1}{9}(2^{6k-1} - 5) \right) \right|_{2^{n-1}-1} \\ &= |11 \cdot 6^{6k-1} - 10|_{2^{6k-1}-1} \\ &= |11 \cdot (6^{6k-1} - 1) + 1|_{2^{6k-1}-1} \\ &= 1. \end{aligned}$$

Proof of (10): When $n = 6k + 2$, LHS of (7) becomes

$$\begin{aligned} & \left| 18 \cdot \left(2^{6k} + \frac{8}{9}(2^{6k} - 1) \right) \right|_{2^{n-1}-1} \\ &= |17 \cdot 6^{6k+1} - 16|_{2^{6k+1}-1} \\ &= |17 \cdot (6^{6k+1} - 1) + 1|_{2^{6k+1}-1} \\ &= 1. \end{aligned}$$

This completes the proof.

The closed form expressions provided by Lemma 1 in their original forms are not amenable to hardware realization. Two special properties of modulo $2^n - 1$ arithmetic, proposed by Szabo and Tanaka [1] and used extensively in [26], are exploited to simplify the implementations of the solutions. These properties are introduced here as Properties 1 and 2.

Property 1: Multiplying an n -bit binary number x by r power of 2 in modulo $2^n - 1$ is equivalent to a circular left shift operation

$$|x \cdot 2^r|_{2^n-1} = \text{CLS}_n(x, r) \quad (11)$$

where $\text{CLS}_n(x, r)$ denotes a circular shift of the n -bit binary number x by r bits to the left.

If the multiplicand is negative, we have the following.

Property 2:

$$\begin{aligned} |-x \cdot 2^r|_{2^n-1} &= |(2^n - 1 - x) \cdot 2^r|_{2^n-1} \\ &= |\bar{x} \cdot 2^r|_{2^n-1} \\ &= \text{CLS}_n(\bar{x}, r) \end{aligned} \quad (12)$$

where $\bar{x}$ is the 1's complement of the n -bit binary number x .

Properties 1 and 2 can be utilized to eliminate the logic circuits needed to implement the modulo $2^n - 1$ multiplication by powers of 2. Only re-wiring of bits is required which incurs virtually no hardware cost and delay. In order to take advantage of Properties 1 and 2 for efficient hardware implementation, the expressions of k_0 under the three different ranges of n are expanded into differences of geometrical series.

For the case of $n = 6k - 2 (k \geq 2)$

$$\begin{aligned} k_0 &= 2^{6k-5} + \frac{2}{9}(2^{6k-6} - 1) \\ &= 2^{6k-5} + \sum_{i=0}^{k-2} (2^{6i+4} - 2^{6i+1}) \\ &= 2^{6k-5} + (2^4 + 2^{6 \cdot 1 + 4} + \dots + 2^{6(k-2)+4}) \\ &\quad - (2^1 + 2^{6 \cdot 1 + 1} + \dots + 2^{6(k-2)+1}) \\ &= 2^{n-3} + (2^4 + 2^{6 \cdot 1 + 4} + \dots + 2^{n-6}) \\ &\quad - (2^1 + 2^{6 \cdot 1 + 1} + \dots + 2^{n-9}). \end{aligned} \quad (13)$$

When $n = 6k (k \geq 2)$

$$\begin{aligned} k_0 &= 2^{6k-2} + \frac{1}{9}(2^{6k-1} - 5) \\ &= 2^{6k-2} + \sum_{i=0}^{k-1} 2^{6i+2} - 1 - \sum_{i=0}^{k-2} 2^{6i+5} \\ &= 2^{6k-2} + (2^2 + 2^{6 \cdot 1 + 2} + \dots + 2^{6(k-1)+2}) \\ &\quad - (2^0 + 2^5 + 2^{6 \cdot 1 + 5} + \dots + 2^{6(k-2)+5}) \\ &= 2^{n-2} + (2^2 + 2^{6 \cdot 1 + 2} + \dots + 2^{n-4}) \\ &\quad - (2^0 + 2^5 + 2^{6 \cdot 1 + 5} + \dots + 2^{n-7}). \end{aligned} \quad (14)$$

When $n = 6k + 2$

$$\begin{aligned}
 k_0 &= 2^{6k} + \frac{8}{9}(2^{6k} - 1) \\
 &= 2^{6k} + \sum_{i=1}^k (2^{6i} - 2^{6i-3}) \\
 &= 2^{6k} + (2^{6 \cdot 1} + 2^{6 \cdot 2} + \dots + 2^{6k}) \\
 &\quad - (2^{6 \cdot 1-3} + 2^{6 \cdot 2-3} + \dots + 2^{6k-3}) \\
 &= 2^{n-2} + (2^{6 \cdot 1} + 2^{6 \cdot 2} + \dots + 2^{n-2}) \\
 &\quad - (2^{6 \cdot 1-3} + 2^{6 \cdot 2-3} + \dots + 2^{n-5}). \quad (15)
 \end{aligned}$$

If we let

$$L = \left| (x_5 - X^{(2)}) \right|_{2^{n-1}-1} \quad (16)$$

$$R = \left| k_0 (x_5 - X^{(2)}) \right|_{2^{n-1}-1} = |k_0 \cdot L|_{2^{n-1}-1}. \quad (17)$$

Then (4) can be expressed as follows:

$$X = X^{(2)} + 2^n(2^{2n} - 1)(2^{n+1} - 1)R. \quad (18)$$

By substituting $X^{(2)}$ of (3) into (16), L can be calculated by

$$\begin{aligned}
 L &= |x_5 - x_2 - 2^n \cdot T|_{2^{n-1}-1} = |x_5 - x_2 - 2 \cdot T|_{2^{n-1}-1} \\
 &= |x_5 - x_2 - 2((2^{2n} - 1)Z + 2^n Y_2 + Y_1)|_{2^{n-1}-1} \\
 &= |x_5 - x_2 - 2(2^{2n} - 1)Z - 2 \cdot 2^n Y_2 - 2Y_1|_{2^{n-1}-1} \\
 &= |x_5 - x_2 - 6Z - 4Y_2 - 2Y_1|_{2^{n-1}-1}. \quad (19)
 \end{aligned}$$

For ease of addressing bit level manipulations, the residues are also represented in their binary forms as follows: $x_1 = (X_{1,n-1}X_{1,n-2} \dots X_{1,0})_2$, $x_2 = (X_{2,n-1}X_{2,n-2} \dots X_{2,0})_2$, $x_3 = (X_{3,n-1}X_{3,n-2} \dots X_{3,0})_2$, $x_4 = (X_{4,n-1}X_{4,n-2} \dots X_{4,0})_2$, and $x_5 = (X_{5,n-1}X_{5,n-2} \dots X_{5,0})_2$. Y_1, Y_2 , and Z are similarly expressed in their binary forms as follows: $Y_1 = (Y_{1,n-1}Y_{1,n-2} \dots Y_{1,0})_2$, $Y_2 = (Y_{2,n-1}Y_{2,n-2} \dots Y_{2,0})_2$, $Z = (Z_{n-1}Z_{n-2} \dots Z_0)_2$. Furthermore, let $x_2^d = x_2 - 2^{n-1}X_{2,n-1}$, $Z^d = Z - (2^n Z_n + 2^{n-1}Z_{n-1})$, $Y_2^d = Y_2 - 2^{n-1}Y_{2,n-1}$, and $Y_1^d = Y_1 - 2^{n-1}Y_{1,n-1}$, where x_2^d, Z^d, Y_2^d , and Y_1^d are the lowest $n-1$ bits of x_2, Z, Y_2 , and Y_1 , respectively. By substituting these terms into the expression of L in (19), we have

$$L = \left| \begin{array}{l} x_5 - x_2^d - 2Z^d - 2^2 Z^d - 2^2 Y_2 - 2Y_1 - 2^{n-1}X_{2,n-1} \\ -2^{n+2}Z_n - 2^{n+1}Z_n - 2^{n+1}Z_{n-1} - 2^n Z_{n-1} \\ -2^{n+1}Y_{2,n-1} - 2^n Y_{1,n-1} \end{array} \right|_{2^{n-1}-1} \quad (20)$$

Properties 1 and 2 can then be used to simplify (20) to

$$L = \left| \sum_{i=1}^{13} L_i \right|_{2^{n-1}-1} \quad (21)$$

where

$$L_1 = (X_{5,n-2}X_{5,n-3} \dots X_{5,1}X_{5,0})_2 \quad (22a)$$

$$L_2 = (\bar{X}_{2,n-2}\bar{X}_{2,n-3} \dots \bar{X}_{2,1}\bar{X}_{2,0})_2 \quad (22b)$$

$$L_3 = \text{CLS}_{n-1}(\bar{Z}^d, 1) = (\bar{Z}_{n-3} \dots \bar{Z}_1 \bar{Z}_0 \bar{Z}_{n-2})_2 \quad (22c)$$

$$L_4 = \text{CLS}_{n-1}(\bar{Z}^d, 2) = (\bar{Z}_{n-4} \dots \bar{Z}_0 \bar{Z}_{n-2} \bar{Z}_{n-3})_2 \quad (22d)$$

Fig. 1. Architectures for calculation of M , modular summation of L_7 to L_{13} . (a) Original calculation of M . (b) Simplified calculation of M .

$$L_5 = \text{CLS}_{n-1}(\bar{Y}_2^d, 2) = (\bar{Y}_{2,n-4} \dots \bar{Y}_{2,0} \bar{Y}_{2,n-2} \bar{Y}_{2,n-3})_2 \quad (22e)$$

$$L_6 = \text{CLS}_{n-1}(\bar{Y}_1^d, 1) = (\bar{Y}_{1,n-3} \dots \bar{Y}_{1,1} \bar{Y}_{1,0} \bar{Y}_{1,n-2})_2 \quad (22f)$$

$$L_7 = ((1)^{(n-2)} \bar{X}_{2,n-1})_2 \quad (22g)$$

$$L_8 = ((1)^{(n-5)} \bar{Z}_n(1)^{(3)})_2 \quad (22h)$$

$$L_9 = ((1)^{(n-4)} \bar{Z}_n(1)^{(2)})_2 \quad (22i)$$

$$L_{10} = ((1)^{(n-4)} \bar{Z}_{n-1}(1)^{(2)})_2 \quad (22j)$$

$$L_{11} = ((1)^{(n-3)} \bar{Z}_{n-1}(1)^{(1)})_2 \quad (22k)$$

$$L_{12} = ((1)^{(n-4)} \bar{Y}_{2,n-1}(1)^{(2)})_2 \quad (22l)$$

$$L_{13} = ((1)^{(n-3)} \bar{Y}_{1,n-1}(1)^{(1)})_2. \quad (22m)$$

In (22g) to (22m), the notation, $(c)^{(k)}$ is used to represent a constant string where the constant, c is repeated k times [28]. There are some embedded constant strings of '1's in the expressions of L_7 to L_{13} . Therefore, the modular summation, M of L_7 to L_{13} , can be simplified substantially. Fig. 1(a) shows the architecture of the modular summation M , where C_M and S_M are the $(n-1)$ -bit carry and sum outputs of M . One 7-input carry save adder (CSA) with end-around carry (EAC) [45] tree is used. After the logic simplification, C_M and S_M can be expressed by

$$\begin{aligned}
 C_M &= ((1)^{(n-5)} C_{M,3}(1)^{(1)} C_{M,1}(1)^{(1)})_2 \\
 S_M &= ((1)^{(n-6)} S_{M,4} S_{M,3} S_{M,2} S_{M,1} \bar{X}_{2,n-1})_2 \quad (23)
 \end{aligned}$$

where $C_{M,1}, C_{M,3}$, and $S_{M,1}$ to $S_{M,4}$ can be calculated by

$$\begin{aligned}
 C_{M,1} &= \overline{Z_{n-1} \cdot Y_{1,n-1}} \\
 C_{M,3} &= \overline{Z_n(Z_{n-1} \oplus Y_{2,n-1})} \\
 S_{M,1} &= \overline{Z_{n-1} \oplus Y_{1,n-1}} \\
 S_{M,2} &= \overline{Z_n \oplus Z_{n-1} \oplus Y_{2,n-1}} \\
 S_{M,3} &= \overline{\bar{Z}_n \bar{Z}_{n-1} Y_{2,n-1} + Z_n \bar{Z}_{n-1} \bar{Y}_{2,n-1}} \\
 S_{M,4} &= \overline{Z_n \cdot Z_{n-1} \cdot Y_{2,n-1}}. \quad (24)
 \end{aligned}$$

Fig. 1(b) shows the implementation of (24). The value of L in (20) can be calculated from C_M and S_M of M as follows:

$$L = \left| \sum_{i=1}^6 L_i + 2 \times C_M + S_M \right|_{2^{n-1}-1}. \quad (25)$$

Fig. 2. Calculation of L .

Thus, only one $(8, 2^{n+1} - 1)$ multi-operand modular adder (MOMA) [16], [21] is required. The architecture is shown in Fig. 2.

Once L has been obtained, R can be calculated by (17) for the three different cases of k_0 . When $n = 6k - 2$, the expanded form of k_0 given by (13) is substituted into (17) and with the aid of Property 1 and Property 2, we have

$$R = \left\{ 2^{n-3} + (2^4 + 2^{6 \cdot 1 + 4} + \dots + 2^{n-6}) - (2^1 + 2^{6 \cdot 1 + 1} + \dots + 2^{n-9}) \right\} \cdot L|_{2^{n-1}-1}$$

$$= \left\{ \begin{array}{l} \text{CLS}_{n-1}(L, n-3) \\ + \text{CLS}_{n-1}(L, 4) + \text{CLS}_{n-1}(L, 10) \\ + \dots + \text{CLS}_{n-1}(L, n-6) + \text{CLS}_{n-1}(\bar{L}, 1) \\ + \text{CLS}_{n-1}(\bar{L}, 7) \\ + \text{CLS}_{n-1}(\bar{L}, 13) + \dots + \text{CLS}_{n-1}(\bar{L}, n-9) \end{array} \right\}_{2^{n-1}-1}. \quad (26)$$

When $n = 6k$, substituting k_0 of (14) into (17), we have

$$R = \left\{ 2^{n-2} + (2^2 + 2^{6 \cdot 1 + 2} + \dots + 2^{n-4}) - (2^0 + 2^5 + 2^{6 \cdot 1 + 5} + \dots + 2^{n-7}) \right\} \cdot L|_{2^{n-1}-1}$$

$$= \left\{ \begin{array}{l} \text{CLS}_{n-1}(L, n-2) \\ + \text{CLS}_{n-1}(L, 2) + \text{CLS}_{n-1}(L, 8) \\ + \dots + \text{CLS}_{n-1}(L, n-4) \\ + \bar{L} + \text{CLS}_{n-1}(\bar{L}, 5) \\ + \text{CLS}_{n-1}(\bar{L}, 11) + \dots + \text{CLS}_{n-1}(\bar{L}, n-7) \end{array} \right\}_{2^{n-1}-1}. \quad (27)$$

When $n = 6k + 2$, substituting k_0 of (15) into (17), we have

$$R = \left\{ 2^{n-2} + (2^{6 \cdot 1} + 2^{6 \cdot 2} + \dots + 2^{n-2}) - (2^{6 \cdot 1 - 3} + 2^{6 \cdot 2 - 3} + \dots + 2^{n-5}) \right\} \cdot L|_{2^{n-1}-1}$$

$$= \left\{ \begin{array}{l} \text{CLS}_{n-1}(L, n-2) \\ + \text{CLS}_{n-1}(L, 6) + \text{CLS}_{n-1}(L, 12) \\ + \dots + \text{CLS}_{n-1}(L, n-2) \\ + \text{CLS}_{n-1}(\bar{L}, 3) + \text{CLS}_{n-1}(\bar{L}, 9) + \dots \\ + \text{CLS}_{n-1}(\bar{L}, n-5) \end{array} \right\}_{2^{n-1}-1}. \quad (28)$$

Therefore, one $(2k - 1, 2^{n-1} - 1)$ MOMA is required for $n = 6k - 2$, and one $(2k + 1, 2^{n-1} - 1)$ MOMA is required for $n = 6k$ or $n = 6k + 2$. Fig. 3 shows the architecture for the calculation of R when $k = 3$ and $n = 6k - 2 = 16$, where only one $(5, 2^{15} - 1)$ MOMA is required.

After the $(n - 1)$ -bit R has been obtained, the final value of X can be calculated by (18). Let the binary representation of R be $(R_{n-2}R_{n-3} \dots R_0)_2$. First, let

$$V = (2^{2n} - 1)(2^{n+1} - 1)R. \quad (29)$$

Fig. 3. Calculation of L when $n = 16$.

Fig. 4. Architecture for the RNS converter for the proposed five-moduli set.

By expanding the equation of V , we have

$$V = 2^{n+1}(2^{2n}R - (2^{n-1}R + R)) + R = 2^{n+1} \cdot U + R \quad (30)$$

where U is defined as follows:

$$U = 2^{2n}R - (2^{n-1}R + R)$$

$$= \left((R_{n-2}R_{n-3} \dots R_1R_0)^{(1)}(0)^{(2n)} \right)_2$$

$$- \left((R_{n-2}R_{n-3} \dots R_1R_0)^{(2)} \right)_2. \quad (31)$$

In (31), the notation $(R_{n-2}R_{n-3} \dots R_0)^{(k)}$ denotes k repetitions of the binary string $R_{n-2}R_{n-3} \dots R_0$ [28]. Only a $(3n - 1)$ -bit binary subtractor is needed for the calculation of U , and the resultant $(3n - 1)$ -bit U can be concatenated to the left of R to form the $4n$ -bit string V by (30). By substituting the computed values of $X^{(2)}$ from (3) and V from (29) into (18), we have

$$X = X^{(2)} + 2^n \cdot V = x_2 + 2^n \cdot (T + V). \quad (32)$$

A $4n$ -bit binary adder is required to sum the values of T and V in (32) and the sum is concatenated to the left of the residue x_2 to obtain X . Fig. 4 shows the final architecture of the residue-to-binary converter for the proposed five-moduli superset, where “&” denotes the concatenation operation of two numbers. The above algorithm is based on the decomposition of the five-moduli set into $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and $\{2^{n-1} - 1\}$. The second possible decomposition is $\{2^n - 1, 2^n, 2^n + 1, 2^{n-1} - 1\}$ and $\{2^{n+1} - 1\}$, but this will cause the MOMAs for the calculation of L and R to become $(n + 1)$ -bit wide as opposed to $(n - 1)$ -bit wide in the above algorithm.

TABLE I
HARDWARE COSTS OF THE PROPOSED RESIDUE-TO-BINARY CONVERTER

Resources	$n = 6k - 2$	$n = 6k$	$n = 6k + 2$
FA	$(5n^2 + 44n - 4)/6$	$(5n^2 + 52n - 12)/6$	$(5n^2 + 48n - 8)/6$
mod $2^{n-1} - 1$ add	2	2	2
mod $2^{n+1} - 1$ add	2	2	2
mod $2^{2n} - 1$ add	1	1	1
$(3n-1)$ -bit sub	1	1	1
$(3n+1)$ -bit sub	1	1	1
$4n$ -bit add	1	1	1
Inverter	$6n+1$	$6n+1$	$6n+1$

Example 1: For $n = 6k - 2$ and $k = 2$, then $n = 10$. The proposed five-moduli superset S is $\{1023, 1024, 1025, 2047, 511\}$. Let the RNS representation of X be $(100, 200, 100, 30, 6)_{10}$. From the residue-to-binary converter for the three-moduli set $\{1023, 1024, 1025\}$, we obtain $Y_1 = (11111111)_2$ and $Y_2 = (1110011011)_2$. From the residue-to-binary converter for the four-moduli superset, $S_1 = \{1023, 1024, 1025, 2047\}$, $Z = (10000101100)_2$, $X^{(2)} = (10B39AF4CC8)_{16}$, and $T = (42CE6BD3)_{16}$ are obtained. From (22a) to (22f), we have $L_1 = (000000110)_2$, $L_2 = (100110111)_2$, $L_3 = (110100111)_2$, $L_4 = (101001111)_2$, $L_5 = (110010000)_2$, and $L_6 = (000000000)_2$. From (24), $C_{M,1} = 1$, $C_{M,3} = 0$, $S_{M,1} = 0$, $S_{M,2} = 1$, $S_{M,3} = 1$, and $S_{M,4} = 1$. Thus, according to (23), $C_M = (111110111)_2$ and $S_M = (111111101)_2$. Based on (25) and (26), $L = (11011011)_2$, $R = (11100\ 0010)_2$. Substituting the value of R into (31), we obtain $U = (1C1C7A3E)_{16}$ and (30) yields $V = (E0E3D1F1C2)_{16}$. Finally, by (32), $X = (3849A817654C8)_{16}$.

IV. PERFORMANCE EVALUATION AND COMPARISONS

The performances of our proposed residue-to-binary converter for the new five moduli superset are evaluated in terms of area, delay and power consumption. It is then compared with the residue-to-binary converters for other five-moduli sets.

According to Fig. 4, the proposed residue-to-binary converter consists of one four-moduli set converter, an arithmetic unit for the calculation of R (including the calculation of L), one $(3n - 1)$ -bit binary subtractor for the calculation of U , one $4n$ -bit binary adder and some inverters. The calculation of L consists of one $(8, 2^{n+1} - 1)$ MOMA shown in Fig. 2. The logic gates required for the calculation of C_M and S_M shown in Fig. 1(b) approximate to two full adders (FAs). The calculation of R consists of one $(2k - 1, 2^{n-1} - 1)$ MOMA for $n = 6k - 2$, and one $(2k + 1, 2^{n-1} - 1)$ MOMA for $n = 6k$ or $n = 6k + 2$. When the 4-stage residue-to-binary converter for the four-moduli set $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ is used [26], the estimated hardware costs in terms of primitive logic elements such as FA, binary adder (add) and subtractor (sub), and modular adder (mod add), are shown in Table I. The total conversion delay is the sum of the delays of the residue-to-binary converter for the three-moduli set $\{2^n - 1, 2^n, 2^n + 1\}$, the calculation of Z , the calculation of R , the $(3n - 1)$ -bit subtractor, and the $4n$ -bit adder. To improve the throughput rate, pipelining is usually applied in real implementation.

In order to obtain more realistic results, the proposed converter is described by structural VHDL. Six-stage pipelining is used to achieve a high throughput. Synopsys Design Compiler is used with the Avant! Libra-Passport V2.6 digital library (3.3-V,

TABLE II
PERFORMANCES WITH AREA OPTIMIZED SYNTHESIS

n	DR (bits)	Area (gates)	Total Delay (ns)	Pipelined Delay (ns)	Power (mW)
4	19	1620.8	18.00	5.57	12.68
8	39	3718	28.91	7.93	31.9
12	59	6110	34.74	10.49	51.9
16	79	8555	36.41	10.61	80.9
20	99	11479	39.09	11.02	110.6
24	119	17477	41.73	11.32	144.8
28	139	21136	43.13	11.67	174.8

TABLE III
PERFORMANCES WITH TIMING OPTIMIZED SYNTHESIS

n	DR (bits)	Area (gates)	Total Delay (ns)	Pipelined Delay (ns)	Power (mW)
4	19	3588	12.52	4.16	37.1
8	39	8332	17.32	5.48	87.71
12	59	13829	19.99	6.66	147.0
16	79	19899	20.99	7.06	211.7
20	99	26154	22.35	7.41	278.2
24	119	33477	25.10	7.78	339.5
28	139	41936	27.86	7.92	392.1

0.35- μm , 4 metal layers) to synthesize the design for different dynamic ranges (DRs). Two optimization options are analyzed. First, the design is area constrained to obtain a minimum area design. Second, increasingly stringent timing constraints are applied to each design progressively until the verge of timing closure. Table II shows the performances of the proposed design optimized for area and Table III shows the results of delay driven optimization. The area cost is measured in terms of the number of equivalent two-input NAND gates of the circuit synthesized by the Design Compiler. The total area cost includes both the cell area and the net interconnection area. For completeness, the default power simulation results invoked by Synopsys Design Compiler are included. The dynamic power reported by the Synopsys power compiler models the switching activity in terms of static probability and toggle rate. This power analysis assumed 50% static probability and a toggle rate of 50% of the fastest clock in the design. From the simulation results, extreme timing driven optimization almost doubles the area costs of the designs in comparison with those optimized for area. In real applications, if the desired delay has already been achieved, the excess timing margin can be traded for the area cost of the design.

Since the proposed five-moduli set is an entirely new five-moduli set, it is difficult to make a judicial comparison of its residue-to-binary converter against the converters of other existing five-moduli sets, such as those proposed in [28], [29], [31], [32]. The disadvantages of these moduli sets have been discussed in the introduction section. In [28] the author assumes that the existing modular multipliers in the arithmetic units can be used for the residue-to-binary conversion. According to the comments from [29], this may severely affect the performance of the system as a whole. The authors in [29] proposed a conversion algorithm for a similar five-moduli set by simplifying the modular multipliers used in [28] to some modular additions. Careful analysis shows that the result of this simplification is trivial, as the required MOMAs with nonspecial moduli and the modular reductions still require similar amount of hardware costs as, if not higher than, those required for the constant multipliers. Hiasat [32] proposed a more efficient algorithm for

yet another similar five-moduli set, where the modular multipliers are eliminated, and the number of the CSAs is reduced from 9 in [28] and [29] to 6. All these existing special moduli sets are valid for odd n . In view of the lack of efficient modular adders and multipliers for the two extended moduli that are not of the form $2^n \pm 1$ [33], [38], these moduli sets are envisaged to have inferior performance compared to our proposed five-moduli superset for the binary-to-residue conversion and the residue arithmetic operations. Hence, the most appropriate comparison is to benchmark the performance against another five moduli superset, such as the latest most efficient five moduli superset proposed in [31]. Nevertheless, for the purpose of evaluation, we will still provide an impartial comparison of the efficient implementation of the residue-to-binary converters for all the above four five-moduli sets. Following the same basis of comparison adopted in [32], both residue-to-binary converters of [28] and [32] were implemented as a stand alone entity and synthesized using the same synthesis tools and digital library as before. For generality, reuse of hardware resources from the RNS processing channels as noted in [28] has not considered here to avoid the intricate trade-off on performance due to degree of reusability and the accompanying control and timing circuitries. Results generated by both timing and area constrained optimizations are analyzed. The implementations of the converters of [31] and [32] are obtained directly from the algorithms and structures proposed in these papers. Using similar techniques as [29] and [32], we simplify the modular multipliers for the calculation of A, B, C , and D of the converter of [28] into constant multiplications. The simplification is described as follows.

The architecture proposed in [28] consists of two parts. The first part refers to the calculation of A, B, C , and D , where four modular multipliers with constants are required. After A, B, C , and D have been calculated, Z_1 to Z_{11} can simply be obtained by re-wiring. The second part consists of one $(11, 2^{4n} - 1)$ -MOMA. As the implementation of the modular multipliers in the first part was not given in [28], we propose the improved architecture which has been similarly performed by [29] for the calculation of A, B, C , and D for a fair comparison. According to [28], $A = |x_2 \cdot N_2^*|_{2^n-1}$, where N_2^* is the multiplicative inverse of $(2^n + 1)(2^{2n} + 1)$ modulo $2^n - 1$. By solving the modular equation, we obtain $N_2^* = 2^{n-2}$. Applying **Property 1** to the calculation of A gives $A = |x_2 \cdot 2^{n-2}|_{2^n-1} = \text{CLS}_n(x_2, n-2)$. This simplification leads to the substitution of the more complex modulo $2^n - 1$ multiplication with constant by a simpler circular shift operation, which can be achieved by re-routing the signals.

According to [28], $B = |x_3 \cdot N_3^*|_{2^{n+1}}$, $C = |x_4 \cdot N_4^*|_{2^{n+2(n+1)/2+1}}$ and $D = |x_5 \cdot N_5^*|_{2^{n+2(n+1)/2+1}}$, where N_3^* is the multiplicative inverse of $(2^n - 1)(2^{2n} + 1)$ modulo $(2^n + 1)$, N_4^* is the multiplicative inverse of $(2^{2n} - 1)(2^n - 2^{(n+1)/2} + 1)$ modulo $(2^n + 2^{(n+1)/2} + 1)$, and N_5^* is the multiplicative inverse of $(2^{2n} - 1)(2^n + 2^{(n+1)/2} + 1)$ modulo $(2^n - 2^{(n+1)/2} + 1)$. By solving the modular equations, we obtain the multiplicative inverses $N_3^* = 2^{n-2}$, $N_4^* = 2^{n-2} + 2^{(n-5)/2}$, and $N_5^* = 2^{n-2} - 2^{(n-5)/2}$. By exploiting the special formats of the constants, the modular multipliers for the calculation of B, C and D can be further optimized to modular additions but not completely eliminated as in the calculation of A . Having

Fig. 5. Comparison of area complexity of residue-to-binary converters.

Fig. 6. Comparison of worst case delay of residue-to-binary converters.

Fig. 7. Comparison of power consumption of residue-to-binary converters.

proposed notable improvements to the architecture for the residue-to-binary converter of [28], we proceed to examine its implications on area, delay and power.

Fig. 5 shows the comparison of the costs of silicon area usage of the residue-to-binary converters of our five-moduli superset, Skavantzios' [28], another five-moduli superset [31] and Hiasat's [32], simulated at various dynamic ranges and optimized for minimum silicon area. Figs. 6 and 7 show the

total conversion delays and the average power consumptions of these four residue-to-binary converters, respectively. The results show that our proposed converter is intermediate between [32] and [31] and much better than [28] for the same dynamic range for all performance metrics except for the conversion delay where a crossing point was found at the dynamic range of 45-bit. Below the crossing point, our converter has similar speed as that of [31] but inferior to those of [28] and [32]. Similar trend is observed for the results obtained by timing optimization. For a low dynamic range, the differences in the reverse converter architectures for different five moduli sets cause the critical paths of some converters to deviate in the two-level CLA configurations. It is noted that the area, delay and power of the converter of [28] increase faster with dynamic range than those of the other converters. The main reason for the inferior performance of the residue-to-binary converter of [28] is that the hardware resources required by the modular multipliers cannot be totally eliminated. Due to the higher Hamming weight of the multiplicative inverses for the MRC calculation of [31], the corresponding MOMAs have more inputs than our algorithm, resulting in an escalation of hardware costs with increase in dynamic ranges. For example, for our proposed algorithm, when $n = 6k$, according to (14) and (27), one $(n/3 + 1, 2^{n-1} - 1)$ MOMA is required, whereas for the converter of [31], when $n = 6k - 1$, one $(2n - 2, 2^{2n} - 1)$ MOMA is required. Not only does the size of the MOMA of [31] doubled, but its number of inputs is rapidly expanded to nearly six times more than ours.

Although three out of five moduli composing the RNS of [32] are efficient for the RAU, it is the worst-case residue channel that dictates the overall performance of an arithmetic operation in RNS. The speed and cost of modulo operation depend not only on the width of the residue channel but also on the modulus. The critical modulo operations of our proposed RNS and that of [32] are modulo $2^n + 1$ and modulo $2^n + 2^{(n+1)/2} + 1$, respectively. Based on the unit gate model, which assumes that each two-input logic gate except the exclusive-OR gate has a delay of unity and the exclusive-OR gate has a delay of 2, the execution latency of addition in modulo $2^n + 2^{(n+1)/2} + 1$ is approximately $4\log_2 n + 7$ designed with the method of [41] as opposed to $2\log_2 n + 6$ for modulo $2^n + 1$ adder of [40]. Even for the same channel width, arithmetic in modulo $2^n \pm 1$ is much faster than arithmetic in modulo $2^n + 2^{(n+1)/2} + 1$. In fact, the execution latencies of the fastest reported modulo 2^n and modulo $2^n - 1$ adders are equal to $2\log_2 n + 3$ [40], making the execution speed in each channel of our proposed moduli set well balanced. Recently, Conway and Nelson [42] has demonstrated an implementation of a 16-tap RNS filter which outperformed an equivalent two's complement design for dynamic ranges of between 20 and 40 bits, using moduli of the form $2^n \pm 1$ based on an extensive cost evaluation of all possible moduli sets for the dynamic range of interest. It is conjecture that applications involving inner product processor for large dynamic range will greatly benefit from our balanced five moduli set whose moduli are all in the modulo arithmetic friendly forms.

V. CONCLUSION

In this paper, we have proposed a new five-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1, 2^{n-1} - 1\}$, valid for even values of n . It retains the properties of the popular three-moduli set

$\{2^n - 1, 2^n, 2^n + 1\}$ to provide for increased parallelism and high-speed residue-to-binary conversions. When compared with the existing nonsuperset five-moduli sets, the advantages are obvious because the superset consists of moduli in form of $2^n \pm 1$, which are proven in the literature to be more efficient for the RAUs. Since the modulo operations in the RAU are performed more frequently than the residue-to-binary conversion, supersets are preferred over nonsuperset moduli sets. Comparing with the existing non co-prime five-moduli superset, our residue-to-binary converter uses less hardware resource due to the elegant multiplicative inverses which have smaller Hamming weights.

REFERENCES

- [1] N. S. Szabo and R. I. Tanaka, *Residue Arithmetic and its Applications to Computer Technology*. New York: McGraw Hill, 1967.
- [2] M. A. Soderstrand, W. K. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. New York: IEEE Press, 1986.
- [3] F. J. Taylor, "An RNS discrete Fourier transform implementation," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 8, pp. 1386–1394, Aug. 1990.
- [4] J. D. Mellott, M. Lewis, F. Taylor, and P. Coffield, "ASAP-A 2-D DFT VLSI processor and architecture," in *Proc. IEEE Int. Symp. Circuits Syst.*, 1996, vol. 2, pp. 261–264.
- [5] K. P. Lim and A. B. Premkumar, "A modular approach to the computation of convolution sum using distributed arithmetic principles," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 46, no. 1, pp. 92–96, Jan. 1999.
- [6] B. Rejeb, H. Henkelmann, and W. Anheier, "Real-time implementation of fractal image encoder using residue number system," in *Proc. 10th Medit. Electrotech. Conf.*, May 2000, vol. 2, pp. 612–615.
- [7] H. Krishna and J. Sun, "On theory and fast algorithms for error correction in residue number system product codes," *IEEE Trans. Comput.*, vol. 42, no. 7, pp. 840–853, Jul. 1993.
- [8] T. Keller, T. H. Liew, and L. Hanzo, "Adaptive redundant residue number system coded multicarrier modulation," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 11, pp. 2292–2301, Nov. 2000.
- [9] G. Alia and E. Martinelli, "A VLSI algorithm for direct and reverse conversion from weighted binary number system to residue number system," *Inf. Process. Lett.*, vol. 18, pp. 141–145, 1984.
- [10] C. H. Huang, "A fully parallel mixed-radix conversion algorithm for residue number applications," *IEEE Trans. Comput.*, vol. C-32, no. 4, pp. 398–402, Apr. 1983.
- [11] R. M. Capocelli and R. Giancarlo, "Efficient VLSI networks for converting an integer from binary system to residue number system and vice versa," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 35, no. 11, pp. 1425–1430, Nov. 1988.
- [12] K. M. Elleithy and M. A. Bayoumi, "Fast and flexible architectures for RNS arithmetic decoding," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 39, no. 4, pp. 226–235, Apr. 1992.
- [13] T. Srikanthan, M. Bhardwaj, and C. T. Clarke, "Area-time-efficient VLSI residue-to-binary converters," *Proc. IEE Comput. Digit. Tech.*, vol. 145, no. 3, pp. 229–235, May 1998.
- [14] S. Andraos and H. Ahmad, "A new efficient memoryless residue to binary converter," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 35, no. 11, pp. 1441–1444, Nov. 1988.
- [15] Y. Wang, "Residue-to-binary converters based on new chinese remainder theorems," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 3, pp. 192–206, Mar. 2000.
- [16] S. J. Piestrak, "A high speed realization of residue to binary number conversion," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 42, no. 10, pp. 661–663, Oct. 1995.
- [17] A. A. Hiasat and H. S. Abdel-Aty-Zohdy, "Residue-to-binary arithmetic converter for the moduli set $(2^k, 2^k - 1, 2^{k-1} - 1)$," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 45, no. 2, pp. 204–209, Feb. 1998.
- [18] G. C. Cardarilli, M. Re, and R. Lojaccono, "RNS-to-binary conversion for efficient VLSI implementation," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 45, no. 6, pp. 667–669, Jun. 1998.
- [19] A. B. Premkumar, M. Bhardwaj, and T. Srikanthan, "High-speed and low-cost reverse converters for the $\{2n - 1, 2n, 2n + 1\}$ moduli set," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 45, no. 7, pp. 903–908, Jul. 1998.

- [20] A. Skavantzios and M. Abdallah, "Implementation issues of the two-level residue number system with pairs of conjugate moduli," *IEEE Trans. Signal Process.*, vol. 47, no. 3, pp. 826–838, Mar. 1999.
- [21] M. Bhardwaj, A. B. Premkumar, and T. Srikanthan, "Breaking the 2n-bit carry propagation barrier in residue to binary conversion for the $\{2^n - 1, 2^n, 2^n + 1\}$ moduli set," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 45, no. 9, pp. 998–1002, Sep. 1998.
- [22] Z. Wang, G. A. Jullien, and W. C. Miller, "An improved residue-to-binary converter," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 47, no. 9, pp. 1437–1440, Sep. 2000.
- [23] Y. Wang, X. Song, M. Aboulhamid, and H. Shen, "Adder based residue to binary number converters for $(2^n - 1, 2^n, 2^n + 1)$," *IEEE Trans. Signal Process.*, vol. 50, no. 7, pp. 1772–1779, Jul. 2002.
- [24] M. Bhardwaj, T. Srikanthan, and C. T. Clarke, "A reverse converter for the 4-moduli superset $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} + 1\}$," in *Proc. 14th IEEE Symp. Computer Arithmetic*, Adelaide, Australia, Apr. 1999, pp. 168–175.
- [25] A. P. Vinod and A. B. Premkumar, "A memoryless reverse converter for the 4-moduli superset," *J. Circuits, Syst. and Comput.*, vol. 10, no. 1&2, pp. 85–100, Apr. 2000.
- [26] B. Cao, T. Srikanthan, and C. H. Chang, "Efficient reverse converters for the four-moduli sets $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and $\{2^n - 1, 2^n, 2^n + 1, 2^{n-1} - 1\}$," *Proc. IEEE Comput. Digit. Tech.*, vol. 152, no. 5, pp. 687–696, Sep. 2005.
- [27] B. Cao, T. Srikanthan, and C. H. Chang, "Design of residue-to-binary converter for a new 5-moduli superset residue number system," in *Proc. IEEE Int. Symp. Circuits Syst.*, Vancouver, Canada, May 2004, vol. 2, pp. 841–844.
- [28] A. Skavantzios, "An efficient residue to weighted converter for a new residue number system," in *Proc. 8th Great Lakes Symp. VLSI*, New Orleans, LA, Feb. 1998, no. 9, pp. 185–191.
- [29] J. Mathew, D. Radhakrishnan, and T. Srikanthan, "Fast residue-to-binary converter architectures," in *Proc. 4th Midwest Symp. Circuits Syst.*, Aug. 1999, vol. 2, pp. 1090–1093.
- [30] B. Cao, C. H. Chang, and T. Srikanthan, "An efficient reverse converter for the 4-moduli set $\{2^n - 1, 2^n, 2^n + 1, 2^{2n} + 1\}$ based on the new chinese remainder theorem," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 50, no. 10, pp. 1296–1303, Oct. 2003.
- [31] A. Skavantzios and T. Stouraitis, "Grouped-moduli residue number systems for fast signal processing," in *Proc. Int. Symp. Circuits Syst.*, Jun. 1999, vol. 3, pp. 478–483.
- [32] A. Hiasat, "Efficient residue to binary converter," *Proc. IEEE Comput. Digit. Tech.*, vol. 150, no. 1, pp. 11–16, Jan. 2003.
- [33] A. V. Curiger, H. Bonnenberg, and H. Kaeslin, "Regular VLSI architectures for multiplication modulo $(2^n + 1)$," *IEEE J. Solid-State Circuits*, vol. 26, no. 7, Jul. 1991.
- [34] A. Skavantzios and P. B. Rao, "New multiplier modulo $2^n - 1$," *IEEE Trans. Comput.*, vol. 41, no. 8, pp. 957–961, Aug. 1992.
- [35] C. Efstathiou, D. Nikolos, and J. Kalamatianos, "Area-time efficient modulo $2^n - 1$ adder design," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 41, no. 7, pp. 463–467, Jul. 1994.
- [36] Y. Ma, "A simplified architecture for modulo $(2^n + 1)$ multiplication," *IEEE Trans. Comput.*, vol. 47, no. 3, pp. 333–337, Mar. 1998.
- [37] A. A. Hiasat, "High-speed and reduced-area modular adder structures for RNS," *IEEE Trans. Comput.*, vol. 51, no. 1, pp. 84–89, Jan. 2002.
- [38] H. T. Vergos, C. Efstathiou, and D. Nikolos, "Diminished-one modulo $2^n + 1$ adder design," *IEEE Trans. Comput.*, vol. 51, no. 12, pp. 1389–1399, Dec. 2002.
- [39] T. Stouraitis, S. W. Kim, and A. Skavantzios, "Full adder-based arithmetic units for finite integer rings," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 40, no. 11, pp. 740–745, Nov. 1993.
- [40] C. Efstathiou, H. T. Vergos, and D. Nikolos, "Fast parallel-prefix modulo $2^n + 1$ adder," *IEEE Trans. Comput.*, vol. 53, no. 9, pp. 1211–1216, Sep. 2004.
- [41] A. A. Hiasat, "High-speed and reduced-area modular adder structures for RNS," *IEEE Trans. Comput.*, vol. 51, no. 1, pp. 84–89, Jan. 2002.

- [42] 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.
- [43] L. Imbert, V. S. Dimitrov, and G. A. Jullien, "Fault-tolerant computations over replicated finite rings," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 50, no. 7, pp. 858–864, Jul. 2003.
- [44] A. Garg, I. Steiner, G. A. Jullien, J. W. Haslett, and G. H. McGibney, "A high speed complex adaptive filter for an asymmetric wireless LAN using a new quantized polynomial representation," in *Proc. IEEE Int. Symp. Circuits Syst.*, Bangkok, Thailand, May 2003, vol. II, pp. 157–160.
- [45] S. J. Piestrak, "Design of residue generators and multioperand modular adders using carry-save adders," *IEEE Trans. Comput.*, vol. 43, pp. 68–77, Jan. 1994.

Bin Cao received the B.Eng. degree from Wuhan Institute of Technology, Wuhan, China in 1991, the M.Eng. degree from Zhejiang University, Hangzhou, China in 1996, and the Ph.D. degree from Nanyang Technological University, Singapore in 2006.

From July 1996 to October 2001, he was with Eastern Communications Company, Hangzhou, China. From May 2005, he was with Seagate Technology International, Singapore. His current research interests include computer arithmetic and cryptosystems.

Chip-Hong Chang (S'92–M'98–SM'03) received the B.Eng. (Hons.) from National University of Singapore, Singapore, in 1989, and his M.Eng. and Ph.D. degrees from Nanyang Technological University (NTU), Singapore, in 1993 and 1998, respectively.

He joined the School of Electrical and Electronic Engineering, NTU in 1999, where he is now an Associate Professor. He holds joint appointments as Deputy Director of the Centre for High Performance Embedded Systems (CHiPES) since 2000, and Program Director of the Centre for Integrated Circuits and Systems (CICS) since 2003. His current research interests include low power arithmetic circuits, algorithms and architectures for digital signal processing, and digital watermarking for IP protection. He has authored three book chapters and more than 120 research papers in international refereed journals and conferences.

Thambipillai Srikanthan (SM'92) received the B.Sc. degree (Hons.) in computer and control systems and the Ph.D. degree in system modeling and information systems engineering from Coventry University, Coventry, U.K.

He joined Nanyang Technological University (NTU) in June 1991 and he now holds joint appointments as Professor and Director of the Centre for High Performance Embedded Systems (CHiPES) and Intelligent Devices and Systems (IDeAS) Cluster. His research interests include system integration methodologies for embedded systems, architectural translations of compute intensive algorithms, high-speed techniques for image processing and dynamic-routing. He has authored more than 200 technical papers and has served on a number of administrative and consultative roles during his academic career.

Dr. Srikanthan is a Corporate Member of the Institution of Electrical Engineers, U.K.