Residue-to-Binary Converters Based on New Chinese Remainder Theorems

Yuke Wang

***Abstract*—The speed of arithmetic operations depends on the size of the numbers involved. Smaller numbers have faster operations. That is exactly the reason why residue number systems are attractive in computer arithmetic. However, the conversion from residue to binary numbers involves a large number modulo operations. Several residue-to-binary converters are proposed in this paper. The converters are based on the New Chinese Remainder Theorems (CRT's) I and II, which represent our work. The New CRT's improve the celebrated CRT. The new algorithms use no big size modulo adders. The numbers involved are much smaller compared to the numbers in the CRT and its alternative, the Mixed Radix Conversion method. Given a moduli set as $(P_1, P_2, \cdots, P_n)$, to convert a residue number $(x_1, x_2, \cdots, x_n)$ to its decimal correspondence, a matrix of numbers bounded by P_i is needed for the New CRT I compared to the large numbers M/P_i for the CRT, where $M = P_1 P_2 \cdots P_n$. The New CRT II uses modulo multipliers of size less than $\sqrt{M}$. If the condition $P_{i+1} > P_1 + P_2 + \cdots + P_i$ is satisfied, only one modulo operation of size P_n is needed for the conversion. Residue-to-binary conversion based on New CRT's presented here will have a significant impact on many algorithms which currently use the CRT, particularly in computer arithmetic such as residue number systems.**

***Index Terms*—Algorithm, arithmetic, chinese remainder theorem, circuits and systems, residue number system.**

I. Introduction

THERE has been interest in Residue Number Systems (RNS) arithmetic as a basis for computational hardware since the 1950's [10], [11], [46]. The conventional weighted number systems such as the binary number system and the decimal number system have a carry chain which is often limiting the performance. Arithmetic based on the residue number system (RNS) reduces N-bit arithmetic to $\log N$-bit binary arithmetic [7]. During the past decade, the RNS has received a considerable attention in arithmetic computation and signal processing applications, such as fast Fourier transforms, digital filtering and image processing [4], [5], [11], [12]. The main reasons for the wide spread use are the inherent properties of RNS, such as parallelism, modularity, fault tolerance, and carry-free operations.

The conversion from residue to binary numbers is the crucial step for any successful RNS application. In recent years, the conversion process has been studied very intensively [15]–[45]. For general moduli sets, the residue to binary conversions are based on the Chinese Remainder Theorem (CRT) or Mixed-Radix Conversion (MRC). As is pointed out in [1], one of the most useful and delightful entities in number theory is the CRT. However, a direct implementation of CRT is unprofitable since it is based on a modulo M operation, where M is large [13]. The MRC is a strictly sequential process which requires $O(n)$ times, where n is the size of the moduli set. The arithmetic calculation for MRC is complicated and needs a lot of stored tables [14].

In this paper, several new residue-to-binary converters are proposed. They are not called the new implementation of the CRT, but rather the implementation of the New CRT's, since the converters are based on two general fast conversion theorems which are substantially different from the MRC and the CRT approaches. We consider them to be a major development since the conclusive work of Szabo and Tanaka [10]. Therefore, we name them the New CRT's I and II. Compared to the CRT and the MRC approaches, the converters based the New CRT's require no big size modulo adders. In many cases, only one modulo operation is needed. The numbers involved in the conversion are smaller than the numbers in the CRT. This will gain speed, since binary arithmetic speed is often bounded by the size of the numbers. For example, the addition of N k-bit numbers need time $O(\log k + \log N)$ [5].

Due to the above-mentioned advantages of the New CRT's, it is expected that they will replace both the CRT and the MRC methods. Considering the significant importance of the CRT, we expect many applications in digital signal processing, communication, and computer arithmetic will benefit from the results presented here. Current techniques for those hard RNS operations, such as divisions and scaling, are based on either the CRT or MRC approach. They will definitely benefit from the algorithms presented here. Therefore, the conclusion about RNS-based arithmetic will be reasonably challenged.

The rest of this paper is organized as follows. In Section II, we introduce the background and notations which covers the RNS and the CRT. In Section III, we introduce the new residue-to-binary converters based on the New CRT's and their implementations. In Section IV, we give a brief summary of the literature and compare the performance. In Section V, we conclude the paper.

II. Background

For any two numbers X and $P_i, x_i = X \bmod P_i$ is defined as $X = x_i + bP_i$ for some integer b such that $0 \le x_i < P_i$. $X \bmod P_i$ can be written as X_{P_i}.

A residue number system is defined in terms of a set of relatively prime moduli set $(P_1, P_2, \cdots, P_n)$, that is, the $\mathrm{GCD}(P_i, P_j) = 1$ for $i \neq j$. A binary number X can be represented as $X = (x_1, x_2, \cdots, x_n)$, where $x_i = X \bmod P_i$,

Manuscript received May 1998; revised October 1999. This paper was recommended by Associate Editor E. Friedman.

The author is with the Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431 USA.

Publisher Item Identifier S 1057-7130(00)02391-0.

$0 \le x_i < P_i$. Such a representation is unique for any integer $X \in [0, M-1]$, where $M = \prod_{1 \le i \le n} P_i$.

To convert a residue number $(x_1, x_2, \cdots, x_n)$ into its binary representation X, the CRT and MRC are generally used. We define $|1/P_i|_{P_j}$ as $|1/P_i|_{P_j} * P_i = 1 \bmod P_j$, and call it the multiplicative inverse of $P_i \bmod P_j$.

1) CRT: Using the CRT, we compute the number X by

$$X = \left[\sum_{i=1}^{n} s_i \left| \frac{x_i}{s_i} \right|_{P_i} \right] \bmod M \tag{1}$$

where $s_i = (M/P_i)$, and $|1/s_i|_{P_i}$ is the multiplicative inverse of $s_i \bmod P_i$. The number $s_i|1/s_i|_{P_i}$ satisfies the condition that $[s_i|1/s_i|_{P_i}] \bmod P_i = 1, [s_i|1/s_i|_{P_i}] \bmod P_j = 0$ if $j \ne i$.

To implement (1) in hardware operations, the addition of the n numbers in (1) can be done in $O(\log n)$ time using standard techniques. However, the numbers $s_i|x_i/s_i|_{P_i}$ satisfy the condition $s_i|x_i/s_i|_{P_i} \ge (M/P_i)$, which are much larger compared to the numbers used in other new approaches proposed in this paper.

2) MRC: Using the MRC method, we compute the number X by

$$X = \left(\sum_{i=1}^{n} v_i a_i \right) \tag{2}$$

where $v_i = \prod_{j=1}^{i-1} P_j$ for $2 \le i \le n$ and $v_1 = 1$; the a_i's, called the mixed radix digits, are computed by the following [9]:

$$Y_1 = X, \quad Y_i = (Y_{i-1} - a_{i-1}) \left| \frac{1}{P_{i-1}} \right|, \quad a_i = Y_i \bmod P_i.$$

We list a_1, a_2, a_3 as

$$\begin{aligned} a_1 &= x_1 \\ a_2 &= (x_2 - a_1) \left| \frac{1}{P_1} \right| \bmod P_2 \\ a_3 &= \left[(x_3 - a_1) \left| \frac{1}{P_1} \right| - a_2 \right] \left| \frac{1}{P_2} \right| \bmod P_3. \end{aligned}$$

From the process, one can see that the MRC is a sequential process. The coefficient a_i can not be computed if a_1 to a_{i-1} are not known. The delay of the MRC approach is $O(n)$. For high-speed arithmetic where the CRT is intend to be used, $O(n)$ delay is not desirable.

Example 1: Find the decimal number for the RNS number $X = (1, 2, 3, 4)$ given the moduli set as (3, 5, 7, 11).

Using the CRT, the computation is done as follows:

$$\begin{aligned} M &= 3 * 5 * 7 * 11 = 1155 \\ s_1 &= 385, \quad s_2 = 231, \quad s_3 = 165, \quad s_4 = 105 \\ |1/s_1|_{P_1} &= 1, \quad |1/s_2|_{P_2} = 1, \quad |1/s_3|_{P_3} = 2 \\ |1/s_4|_{P_4} &= 2 \\ X &= [385 * 1 * x_1 + 231 * 1 * x_2 + 165 * 2 * x_3 \\ &\quad + 105 * 2 * x_4]_{1155}. \end{aligned} \tag{3}$$

For $(x_1, x_2, x_3, x_4) = (1, 2, 3, 4)$, $X = [385 + 462 + 990 + 840]_{1155} = 2677_{1155} = 367$.

Using the MRC, the computation is done as shown in the equation at the bottom of the page.

III. New Residue-to-Binary Converters

In this section, we propose two new conversion methods and call them New CRT's. Their implementations are also given. In the following, the moduli set is $(P_1, P_2, \cdots, P_n)$. A binary number X has a residue representation as $X = (x_1, x_2, \cdots, x_n)$, $M = P_1 * P_2 \cdots * P_n$.

A. Converters Based on New CRT I

The following simple propositions are needed for our new theorems. Those propositions are given without proofs, since the proofs are not hard to find.

Proposition 1: $a = 1 \bmod P_1 P_2 \Leftrightarrow a = 1 \bmod P_1$, and $a = 1 \bmod P_2$.

Corollary 1: $a = 1 \bmod P_1 P_2 \cdots P_k \Leftrightarrow a = 1 \bmod P_1, \cdots, a = 1 \bmod P_k$.

Proposition 2: $[aP_1]_{P_1 P_2} = [a]_{P_2} * P_1$.

Proposition 3: $[a_{P_1 P_2}]_{P_1} = [a]_{P_1}$.

Proposition 4: For any $y \in [0, M-1]$, there is a unique mixed radix representation as follows, where y_i satisfies the condition that $0 \le y_i < P_{i+1}$.

$$y = y_0 + y_1 P_1 + y_2 P_1 P_2 + \cdots + y_{n-1} P_1 P_2 \cdots P_{n-1}$$

$$\begin{aligned} X &= (a_1 + a_2 P_1 + a_3 P_1 P_2 + a_4 P_1 P_2 P_3) \\ |1/P_1|_{P_2} &= 2, \quad |1/P_1|_{P_3} = 5, \quad |1/P_2|_{P_3} = 3 \\ |1/P_1|_{P_4} &= 4, \quad |1/P_2|_{P_4} = 9, \quad |1/P_3|_{P_4} = 8 \\ a_1 &= x_1 = 1 \\ a_2 &= (2-1) * |1/P_1|_{P_2} = 2 \\ a_3 &= [(x_3 - a_1)|1/P_1|_{P_3} - a_2]|1/P_2|_{P_3} \bmod P_3 \\ &= [(3-1) * 5 - 2] * \bmod P_3 = 8 * 3 \bmod 7 = 3 \\ a_4 &= \{[(x_4 - a_1)|1/P_1|_{P_4} - a_2]|1/P_2|_{P_4} - a_3\}|1/P_3|_{P_4} \bmod P_4 \\ &= \{[(4-1) * 4 - 2] * 9 - 3\} * 8 \bmod P_4 \\ &= 87 * 8 \bmod 11 = 3 \\ X &= (1 + 2 * 3 + 3 * 3 * 5 + 3 * 3 * 5 * 7) = 367 \end{aligned}$$

Moreover, $P_1P_2\cdots P_{n-1}P_n - 1 = (P_1-1) + (P_2-1)P_1 + (P_3-1)P_1P_2 + \cdots + (P_n-1)P_1P_2\cdots P_n$.

The number y_i is called the i-th digit of the mixed radix representation. y_{n-1} is the most significant digit, y_0 is the least significant digit. It is easy to see that the mixed radix representation is a weighted number system.

Theorem 1: Given the residue number $(x_1, x_2, \cdots, x_n)$, the corresponding decimal number X can be computed using the following:

$$X = [x_1 + k_1P_1(x_2 - x_1) + k_2P_1P_2(x_3 - x_2) + \cdots + k_{(n-1)}P_1P_2\cdots P_{n-1}(x_n - x_{n-1})]_{P_1P_2\cdots P_{n-1}P_n} \quad (4)$$

where $k_1P_1 = 1 \bmod P_2P_3\cdots P_n, k_2P_1P_2 = 1 \bmod P_3\cdots P_n, \cdots, k_{(n-1)}P_1P_2\cdots P_{n-1} = 1 \bmod P_n$.

Proof: Based on Corollary 1, we know the following is true:

$$\begin{aligned} &k_1P_1 = 1 \bmod P_2 \\ &k_1P_1 = 1 \bmod P_3, \quad k_2P_1P_2 = 1 \bmod P_3 \\ &\quad\cdots \qquad\qquad \cdots \qquad\qquad \cdots \\ &k_1P_1 = 1 \bmod P_n, \quad k_2P_1P_2 = 1 \bmod P_n, \quad \cdots \\ &k_{(n-1)}P_1P_2\cdots P_{n-1} = 1 \bmod P_n \end{aligned}$$

Therefore we have the following proof:

$$\begin{aligned} &X = x_1 \bmod P_1 \\ &X = \{[x_1]_{P_2} + [k_1P_1]_{P_2} * (x_2 - x_1)\} = x_2 \bmod P_2 \\ &X = \{[x_1]_{P_3} + [k_1P_1]_{P_3} * (x_2 - x_1) + [k_2P_1P_2]_{P_3} \\ &\quad * (x_3 - x_2)\} = x_3 \bmod P_3 \\ &\quad\cdots \\ &X = \{[x_1]_{P_n} + [k_1P_1]_{P_n} * (x_2 - x_1) + \cdots \\ &\quad + [k_{n-1}P_1P_2\cdots P_{n-1}]_{P_n} * (x_n - x_{n-1})\}_{P_n} \\ &\quad = \{x_1 + (x_2 - x_1) + (x_3 - x_2) + \cdots \\ &\quad + (x_n - x_{n-1})\}_{P_n} = x_n \end{aligned}$$

Equation (4) is different from the CRT (1) and the MRC (2). It is a mixed radix representation. However, we do not need the sequential process to find out the mixed radix digits; instead, the digits are given based on the coordinates x_i as in the CRT. Therefore, (4) has the advantage of both the CRT and the MRC. Theorem 1 is the base for the New CRT I. □

Example 2: Using Theorem 1, decode the RNS number (1, 2, 3, 4) given the moduli set as (3, 5, 7, 11)

$$\begin{aligned} &k_1 * 3 = 1 \bmod 5*7*11, \quad k_2 * 3*5 = 1 \bmod 7*11, \\ &k_3 * 3*5*7 = 1 \bmod 11 \\ &\quad k_1 = 257, \quad k_2 = 36, \quad k_3 = 2 \\ &\quad X = [x_1 + k_1P_1(x_2 - x_1) + k_2P_1P_2(x_3 - x_2) \\ &\qquad + k_3P_1P_2P_3(x_4 - x_3)]_{P_1P_2P_3P_4} \\ &\quad = [1 + 257*3 + 36*15 + 2*15*7]_{3*5*7*11} \\ &\quad = [1 + 771 + 540 + 210]_{1155} = 367. \end{aligned}$$

In the following, we investigate how to compute X using (4). Assume that

$$\begin{aligned} Y &= [x_1 + k_1P_1(x_2 - x_1) + k_2P_1P_2(x_3 - x_2) + \cdots \\ &\quad + k_{(n-1)}P_1P_2\cdots P_{n-1}(x_n - x_{n-1})] \\ &= [(1 - k_1P_1)x_1 + (k_1 - k_2P_2)x_2P_1 + \cdots \\ &\quad + \left(k_{(n-2)} - k_{(n-1)}P_{n-1}\right)x_{n-1}P_1P_2\cdots P_{n-2} \\ &\quad + k_{(n-1)}x_nP_1P_2\cdots P_{n-1}]. \end{aligned}$$

Therefore, we have $X = Y_M$. Assume

$$\begin{aligned} a_0 &= [1 - k_1p_1]_{P_1P_2\cdots P_{n-1}P_n} \\ a_1 &= [k_1 - k_2P_2]_{P_2\cdots P_{n-1}P_n} \\ &\cdots \\ a_{n-2} &= \left[k_{(n-2)} - k_{(n-1)}P_{n-1}\right]_{P_{n-1}P_n} \\ a_{n-1} &= \left[k_{(n-1)}\right]_{P_n}. \end{aligned}$$

Given the set of moduli $(P_1, P_2, \cdots, P_{n-1}, P_n)$, the set $\{a_0, a_1, \cdots, a_{n-1}\}$ is uniquely defined. Moreover, we have the following mixed radix representations:

$$\begin{aligned} &a_0 = a_{0,0} + a_{0,1}P_1 + \cdots + a_{0,n-1}P_1P_2\cdots P_{n-1}, \\ &\qquad\qquad 0 \le a_{0,i} < P_{i+1}, \\ &a_1P_1 = a_{1,1}P_1 + \cdots + a_{1,n-1}P_1P_2\cdots P_{n-1}, \\ &\qquad\qquad 0 \le a_{1,i} < P_{i+1} \\ &\cdots \\ &a_{n-2}P_1P_2\cdots P_{n-1} \\ &\quad = a_{n-2,n-2}P_1P_2\cdots P_{n-2} + a_{n-2,n-1}P_1P_2\cdots P_{n-1}, \\ &\qquad\qquad 0 \le a_{n-2,i} < P_{i+1} \\ &a_{n-1}P_1P_2\cdots P_{n-1} = a_{n-1,n-1}P_1P_2\cdots P_{n-1}, \\ &\qquad\qquad 0 \le a_{n-1,n-1} < P_n \end{aligned}$$

Definition 1: The matrix

$$A = \begin{pmatrix} a_{0,0}, & 0, & \cdots & & 0 \\ a_{0,1}, & a_{1,1}, & \cdots & & 0 \\ \cdots & & & & \\ a_{0,n-2}, & a_{1,n-2}, & \cdots & a_{n-2,n-2}, & 0 \\ a_{0,n-1}, & a_{1,n-1}, & \cdots & a_{n-2,n-1}, & a_{n-1,n-1} \end{pmatrix}$$

is called the *characteristic matrix* of the moduli $(P_1, P_2, \cdots, P_{n-1}, P_n)$, where $a_{j,i} < P_{i+1}$. Given the RNS number $X = (x_1, x_2, \cdots, x_{n-1}, x_n)$, we define the vector $B = A \cdot X'$, i.e.,

$$\begin{aligned} B &= \begin{pmatrix} B_0 \\ B_1 \\ \vdots \\ B_{n-2} \\ B_{n-1} \end{pmatrix} \\ &= \begin{pmatrix} a_{0,0}, & 0, & \cdots & & 0 \\ a_{0,1}, & a_{1,1}, & \cdots & & 0 \\ \cdots & & & & \\ a_{0,n-2}, & a_{1,n-2}, & \cdots & a_{n-2,n-2}, & 0 \\ a_{0,n-1}, & a_{1,n-1}, & \cdots & a_{n-2,n-1}, & a_{n-1,n-1} \end{pmatrix} \\ &\quad \times \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}. \end{aligned}$$

We call the vector B the *first-order radix* of the RNS number $(x_1, x_2, \cdots, x_{n-1}, x_n)$. The number $B_i = a_{0,i}x_1 + a_{1,i}x_2 + \cdots + a_{i,i}x_{i+1}$ is the i-th pseudodigit $i = 0, 1, \cdots, n-1$.

The characteristic matrix of the moduli $(P_1, P_2, \cdots, P_{n-1}, P_n)$ can be pre-computed. Every number in the characteristic matrix is bounded by the numbers P_i . Therefore, they are all small numbers. Transforming the RNS number $(x_1, x_2, \cdots, x_{n-1}, x_n)$ to the first-order radix $(B_0, B_1, \cdots, B_{n-1})$ is done by a set of adders and multipliers of small ranges as indicated by Fig. 1. When the range is small, some adders and multipliers can be replaced by ROM.

Proposition 5: The pseudodigits satisfy the condition

$$B_i < P_{i+1}[(P_1 + P_2 + \cdots + P_{i+1}) - (i+1)] \tag{5}$$

Proof: Based on the mixed radix representation, we have the following proof.

$$\begin{aligned} B_i &< P_{i+1}(x_1 + x_2 + \cdots + x_{i+1}) \\ &\le P_{i+1}[(P_1 - 1) + (P_2 - 1) + \cdots + (P_{i+1} - 1)] \\ &\le P_{i+1}[(P_1 + P_2 + \cdots + P_{i+1}) - (i+1)]. \end{aligned}$$

The first-order radix of a RNS number $(x_1, x_2, \cdots, x_{n-1}, x_n)$ directly leads to the decimal correspondence using the following New CRT I. □

1) New CRT I : Given the residue number $(x_1, x_2, \cdots, x_n)$, the decimal number X can be computed by the following:

$$X = [B_0 + B_1P_1 + B_2P_1P_2 + \cdots + B_{n-1}P_1P_2\cdots P_{n-1}]_{P_1P_2\cdots P_{n-1}P_n}. \tag{6}$$

Proof: The following is the proof:

$$\begin{aligned} X = Y_M &= [a_0x_1 + a_1P_1x_2 + \cdots \\ &\quad + a_{n-2}P_1P_2\cdots P_{n-2}x_{n-1} \\ &\quad + a_{n-1}P_1P_2\cdots P_{n-1}x_n]_{P_1P_2\cdots P_{n-1}P_n} \\ &= [(a_{0,0} + a_{0,1}P_1 + \cdots + a_{0,n-1}P_1P_2\cdots P_{n-1})x_1 \\ &\quad + (a_{1,1}P_1 + \cdots + a_{1,n-1}P_1P_2\cdots P_{n-1})x_2 + \cdots \\ &\quad + (a_{n-2,n-2}P_1P_2\cdots P_{n-2} \\ &\quad + a_{n-2,n-1}P_1P_2\cdots P_{n-1})x_{n-1} \\ &\quad + a_{n-1,n-1}P_1P_2\cdots P_{n-1}x_n]_{P_1P_2\cdots P_{n-1}P_n} \\ &= [a_{0,0}x_1 + (a_{0,1}x_1 + a_{1,1}x_2)P_1 + (a_{0,2}x_1 + a_{1,2}x_2 \\ &\quad + a_{2,2}x_3)P_1P_2 + \cdots \\ &\quad + (a_{0,n-1}x_1 + a_{1,n-1}x_2 + \cdots + a_{n-2,n-2}x_{n-1} \\ &\quad + a_{n-1,n-1}x_n)P_1P_2\cdots P_{n-1}]_{P_1P_2\cdots P_{n-1}P_n} \end{aligned}$$ □

Corollary 2: Equation (6) can be further reduced as the following using Proposition 2:

$$\begin{aligned} X = [&B_0 + (B_1)_{P_2\cdots P_{n-1}P_n}P_1 + (B_2)_{P_3\cdots P_{n-1}P_n}P_1P_2 \\ &+ \cdots + (B_{n-1})_{P_n}P_1P_2\cdots P_{n-1}]_{P_1P_2\cdots P_{n-1}P_n}. \end{aligned} \tag{7}$$

The difference between the New CRT I and the CRT is obvious. Here, we use the characteristic matrix consisting of small numbers instead of the big numbers s_i for the CRT. Moreover, the New CRT I is in the mixed radix representation with the first-order radix as the coefficients. Finally, the numbers involved in the addition are smaller in the New CRT I.

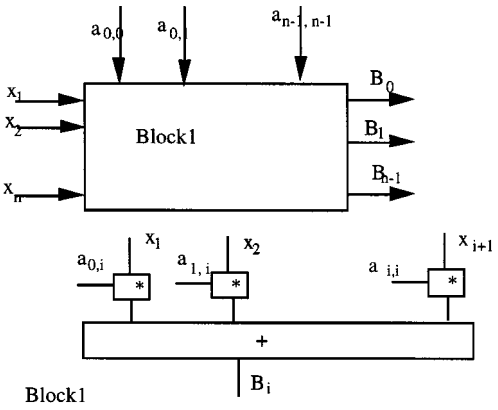

Fig. 1. The calculation of the first-order radix B.

Example 3: Use New CRT I to decode the number (1, 2, 3, 4) given the moduli (3, 5, 7, 11)

$$\begin{aligned} &k_1 * 3 = 1 \bmod 5*7*11, \quad k_2 * 3 * 5 = 1 \bmod 7*11, \\ &k_3 * 5 * 7 = 1 \bmod 11 \\ &\quad k_2 = 257, \quad k_2 = 36, \quad k_3 = 2 \end{aligned}$$

The characteristic matrix is as follows:

$$\begin{pmatrix} a_{0,0} & 0 & 0 & 0 \\ a_{0,1} & a_{1,1} & 0 & 0 \\ a_{0,2} & a_{1,2} & a_{2,2} & 0 \\ a_{0,3} & a_{1,3} & a_{2,3} & a_{3,3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 3 & 2 & 0 & 0 \\ 4 & 1 & 1 & 0 \\ 3 & 2 & 3 & 2 \end{pmatrix}$$

$$\begin{aligned} X = [&x_1 + (3x_1 + 2x_2)P_1 + (4x_1 + x_2 + x_3)P_1P_2 \\ &+ (3x_1 + 2x_2 + 3x_3 + 2x_4)P_1P_2P_3]_{P_1P_2P_3P_4} \end{aligned} \tag{8}$$

It is not hard to see that (8) is very different from (3) in Example 1.

For the number (1, 2, 3, 4), we have the following:

$$\begin{aligned} X &= [1 + (3+4)*3 + (4+2+3)*15 + (3+4+9+8) \\ &\quad * 105]_{1155} \\ &= [1 + 2*3 + (1+2)*15 + (1+24)_7 * 105]_{1155} \\ &= [1 + 2*3 + 3*15 + 3*105] \\ &= 367. \end{aligned}$$

Both (6) and (7) can be implemented as residue-to-binary converters. The two equations share the same mixed radix representation. The difference is that (7) has some extra modulo operations. The summation in (6) can be represented in the following way, where $k = \lfloor n/2 \rfloor$:

$$\begin{aligned} Y &= B_0 + B_1P_1 + B_2P_1P_2 + \cdots + B_{n-1}P_1P_2\cdots P_{n-1} \\ &= (B_0 + B_1P_1 + \cdots + B_{k-1}P_1P_2\cdots P_{k-1}) \\ &\quad + P_1P_2\cdots P_k(B_k + B_{k+1}P_{k+1} + \cdots \\ &\quad + B_{n-1}P_{k+1}P_{k+2}\cdots P_{n-1}). \end{aligned}$$

The terms $B_0 + B_1P_1 + \cdots + B_{k-1}P_1P_2\cdots P_{k-1}$ and $(B_k + B_{k+1}P_{k+1} + \cdots + B_{n-1}P_{k+1}P_{k+2}\cdots P_{n-1})$ can be further divided into two parts to reduce the number of multiplications.

Fig. 2 is a residue-to-binary converter based on the New CRT I ((6)) for a moduli set with eight elements. The first-order radix are computed using the circuit in Fig. 1. Given the first-order

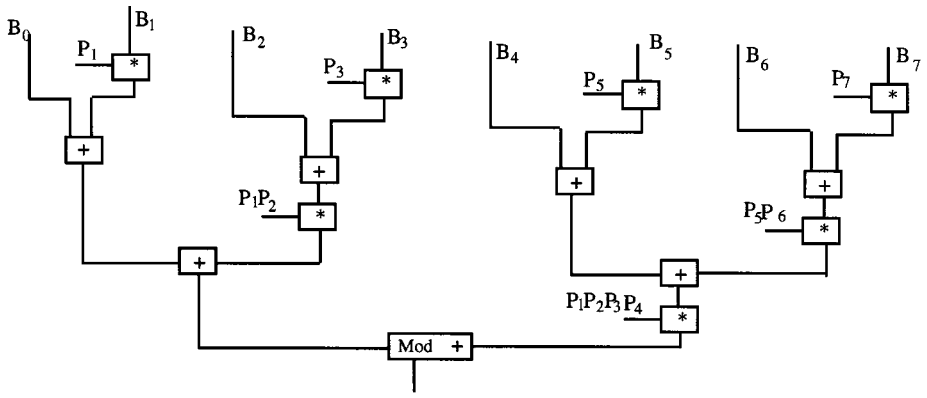

Fig. 2. General RNS converter based on New CRT I.

radix, the conversion needs to add eight numbers together. This can be achieved by using an adder tree of depth $\log n = 3$. The operations are optimized for small size operations. Due to the mixed radix representation of the New CRT I, the addition can be simplified into shifting of numbers at each stage $(P_1P_2, P_5P_6, P_1P_2P_3P_4)$ if we choose the moduli sets right. If one chooses to implement (7), then some extra modulo operations are needed for the first-order radix.

In the following, we assume that the moduli set $(P_1, P_2, \cdots, P_{n-1}, P_n)$ satisfies the condition that $P_i > P_1 + P_2 + \cdots + P_{i-1}$. This condition is satisfied if we have $P_i > 2P_{i-1}$. For such moduli sets, we show that the converter in Fig. 2 can be implemented without modulo M operations.

If $P_i > P_1+P_2+\cdots+P_{i-1}$, the i-th pseudo-digit B_i satisfies the condition

$$B_i < P_{i+1}[(P_1+P_2+\cdots+P_{i+1})-(i+1)] < P_{i+1}P_{i+2} \quad (9)$$

For $i = n-1$, some extra attention is needed since the number P_{n+1} is not defined. However, the following proof process can be easily verified to be valid. Therefore we do not treat this case separately for simplicity reason.

Using the mixed radix representation, we have

$$B_i = b_{i,i} + b_{i,i+1}P_{i+1}, \quad 0 \le b_{i,i} \le P_{i+1} - 1$$

$0 \le b_{i,i+1} < (P_1 + P_2 + \cdots + P_{i+1}) - (i+1)$, based on (9). We define the i-th partial sum as

$$Y_i = b_{0,0} + (b_{0,1} + b_{1,1})P_1 + (b_{1,2} + b_{2,2})P_1P_2 + \cdots + (b_{i-1,i} + b_{i,i})P_1 \cdots P_i.$$

When $i = n$, we denote

$$\begin{aligned} Y &= b_{0,0} + (b_{0,1} + b_{1,1})P_1 + (b_{1,2} + b_{2,2})P_1P_2 + \cdots \\ &\quad + (b_{n-2,n-1} + b_{n-1,n-1})P_1 \cdots P_{n-1} \\ &= B_0 + B_1P_1 + B_2P_1P_2 + \cdots + B_{n-2}P_1P_2 \cdots P_{n-2} \\ &\quad + (B_{n-1})_{P_n}P_1P_2 \cdots P_{n-1} \end{aligned} \quad (10)$$

Proposition 6: The following conditions are true:

$$(b_{i-1,i} + b_{i,i}) < 2P_{i+1} - (i+1) \quad (11)$$

and

$$Y_i < 2P_1 \cdots P_iP_{i+1}.$$

Proof:

$$\begin{aligned} b_{i-1,i}+b_{i,i} &< [(P_1 + P_2 + \cdots + P_i) - i] + (P_{i+1} - 1) \\ &= (P_1 + P_2 + \cdots + P_i + P_{i+1}) - (i+1) \\ &< P_{i+1} + [P_1 + P_2 + \cdots + P_i - (i+1)] \\ &< 2P_{i+1} - (i+1) \end{aligned}$$

$$\begin{aligned} Y_i &= b_{0,0}+(b_{0,1}+b_{1,1})P_1+(b_{1,2}+b_{2,2})P_1P_2+\cdots \\ &\quad + (b_{i-1,i} + b_{i,i})P_1 \cdots P_i \\ &< P_1 + (2P_2 - 2)P_1 + (2P_3 - 3)P_1P_2 + \cdots \\ &\quad + (2P_{i+1} - (i+1))P_1 \cdots P_i \\ &= (P_1 + 2P_1P_2 + 2P_1P_2P_3 + \cdots + 2P_1 \cdots P_i) \\ &\quad + 2P_1 \cdots P_iP_{i+1} - (2P_1 + 3P_1P_2 + \cdots \\ &\quad + (i+1)P_1 \cdots P_i) \\ &= 2P_1 \cdots P_iP_{i+1} - (P_1 + P_1P_2 + \cdots \\ &\quad + (i-1)P_1 \cdots P_i) \\ &< 2P_1 \cdots P_iP_{i+1}. \end{aligned}$$

Corollary 3: When $i = n$, we have $Y < 2P_1 \cdots P_{n-1}P_n$.

Proposition 7: Under the above conditions and (10), the number X in (6) can be computed by the following:

$$\begin{aligned} X &= Y, \quad Y < P_1P_2 \cdots P_n \\ &= Y - P_1P_2 \cdots P_n, \quad Y \ge P_1P_2 \cdots P_n. \end{aligned}$$

This is obvious since $X = Y_M$ according to the New CRT I. Therefore there is no need of modulo operations using the New CRT I to convert RNS numbers into decimal numbers.

Fig. 3 shows the block diagram of the converter circuit for this kind of moduli sets. The only modulo operation needed is to convert B_n to $b_{n-1,n-1}$. The addition of block 2 can be done similarly using the tree structure as in Fig. 2 with $O(\log n)$ delay. Moreover, each addition in Fig. 2 is, at most, a simple carry operation due to Proposition 6.

B. Converters Based on New CRT II

In the above section, we presented the residue-to-binary converters based on the New CRT I. It was shown that if $P_{i+1} > P_1 + P_2 + \cdots + P_i$, there is no need for modulo adders in the conversion process. In this section, we present a converter based on the New CRT II, which requires no big size modulo

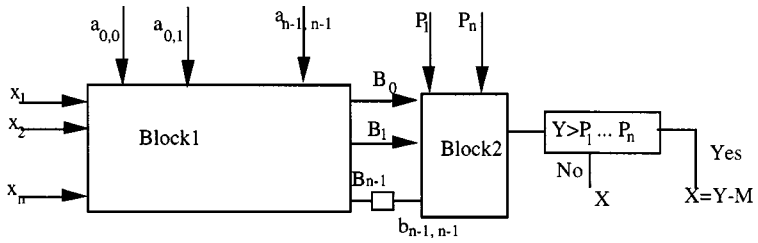

Fig. 3. Special converter based on the New CRT I.

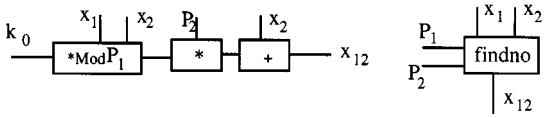

Fig. 4. Implementation of *findno*.

adders for any moduli sets. The algorithm is designed using divide-and-conquer approach.

We first consider the case when the moduli set consists of only two numbers. Let $P = \{P_1, P_2\}$ be the moduli set, where $P_1 < P_2$. Given a RNS representation (x_1, x_2), we want to find X such that $X = x_1 \bmod P_1$, $X = x_2 \bmod P_2$ and $X < P_1 * P_2$. We have the following proposition.

Proposition 8: The corresponding decimal number for (x_1, x_2) can be found using the following:

$$X = x_2 + [k_0(x_1 - x_2)]_{P_1} P_2 \tag{12}$$

where k_0 is a positive integer which satisfies the condition that $k_0 * P_2 = 1 \bmod P_1$. Since $\mathrm{GCD}(P_1, P_2) = 1$, such k_0 always exists.

Proof: From the condition that $X = x_2 \bmod P_2$, we know that $X = x_2 + k * P_2$ for some integer k. Therefore $(x_2 + k * P_2) = x_1 \bmod P_1$. Equivalently, $k * P_2 = x_1 - x_2 \bmod P_1$. Since $k_0 * P_2 = 1 \bmod P_1$, then $k_0 * (x_1 - x_2) * P_2 = x_1 - x_2 \bmod P_1$. Therefore, we have

$$x = [x_2 + k_0(x_1 - x_2)P_2]_{P_1 P_2} = x_2 + [k_0(x_1 - x_2)]_{P_1} P_2.$$

This proposition shows that to decode in two moduli sets, we can avoid using big modulo $M = P_2 P_1$ adders; instead, the smaller modulo P_1 multiplier is used, where $P_1 < \sqrt{M}$. The algorithm for finding the corresponding decimal number for the RNS number (x_1, x_2) is as follows, where we have $X = x_1 \bmod P_1$, $X = x_2 \bmod P_2$. □

Procedure: $findno(x_1, x_2, P_1, P_2, X)$:

1) find a k_0 such that $k_0 * P_2 = 1 \bmod P_1$;
2) $X = x_2 + [k_0(x_1 - x_2)]_{P_1} P_2$.

Fig. 4 shows the block diagram and the implementation of the *findno* procedure. It uses one multiplier of size P_2, one modulo multiplier of size P_1, and one adder. The size of the adder is bounded by P_2 since we have the relation $x_2 < P_2$.

Now we consider the general case with the moduli set as $\{P_1, P_2, \cdots, P_n\}$ and $P_1 < P_2 < \cdots < P_n$. The following conversion algorithm decodes $X = (x_1, x_2, \cdots, x_n)$.

Algorithm: $translate((x_1, x_2, \cdots, x_n), X)$

1) if $n = 2t > 2$ (n is an even number greater than 2), then

$$translate((x_1, \cdots, x_t), N_1), \quad M_1 = \prod_{i=1}^{t} P_i$$

$$translate((x_{t+1}, \cdots, x_n), N_2), \quad M_2 = \prod_{i=t+1}^{n} P_i$$

$$findno(N_1, N_2, M_1, M_2, X).$$

2) if $n = 2t + 1 > 2$ (n is an odd number greater than 2), then

$$translate((x_1, \cdots, x_t), N_1), \quad M_1 = \prod_{i=1}^{t} P_i$$

$$translate((x_{t+1}, \cdots, x_n), N_2), \quad M_2 = \prod_{i=t+1}^{n} P_i$$

$$findno(N_1, N_2, M_1, M_2, X).$$

3) if $n = 2$, then *findno*(x_1, x_2, P_1, P_2, X).
4) if $n = 1$, then $X = x_1 \bmod P_1$.

New CRT II: The above algorithm, *translate*, finds the correct decimal representation of the RNS number $X = (x_1, x_2, \cdots, x_n)$.

Proof: We prove the correctness of the algorithm by induction.

1) If $n = 1$ (the size of the moduli set is 1), then (4) of *translate* is correct by definition.
   If $n = 2$, then by Proposition 8, (3) of *translate* is correct.
2) Induction step: Suppose that for $n < N_0$, the algorithm *translate* is correct.
3) For $n = N_0 + 1$, without loss of generality, we assume that $n = N_0 + 1 = 2m$, then we translate $(x_1, \cdots, x_m) = N_1 \bmod M_1 = \prod_{i=1}^{m} P_i$ and translate $(x_{m+1}, \cdots, x_n) = N_2 \bmod M_2 = \prod_{i=m+1}^{n} P_i$.

By the induction, we know that

$$N_1 = x_i \bmod P_i, \quad \text{for } i = 1 \text{ to } m \tag{i}$$

$$N_2 = x_j \bmod P_j, \quad \text{for } j = m+1 \text{ to } n = 2m. \tag{ii}$$

The procedure $findno(N_1, N_2, M_1, M_2, X)$ ensures that $X = N_1 \bmod M_1$ and $X = N_2 \bmod M_2$, i.e., $X = N_1 + a_1 * M_1 = N_2 + a_2 * M_2$, where a_1 and a_2 are some integers.

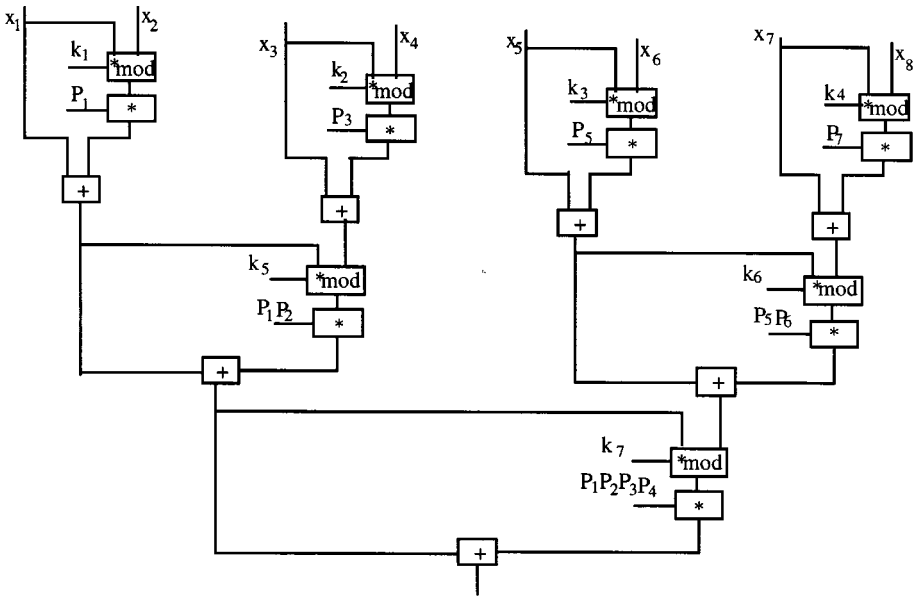

Fig. 5. A converter based on New CRT II for 8-element moduli sets.

Therefore, for $i \le m$

$$\begin{aligned} X \bmod P_i &= (N_1 + a_1 * M_1) \bmod P_i \\ &= N_1 \bmod P_i, \text{ since } M_1 \bmod P_i = 0 \\ &= x_i, \text{ by induction (i).} \end{aligned}$$

For $m < i \le 2m$

$$\begin{aligned} X \bmod P_i &= (N_2 + a_2 * M_2) \bmod P_i \\ &= N_2 \bmod P_i, \text{ since } N_2 \bmod P_i = 0 \\ &= x_i, \text{ by induction (ii).} \end{aligned}$$

Therefore X is the correct number. □

The difference between the New CRT II and the CRT is obvious. No big modulo operations are needed for the New CRT II. The modulo multipliers in the New CRT II is bounded by size $\sqrt{M}$. In the literature, similar divide-and-conquer approach has been used for implementing the CRT [19]. In [19], a divided and conquer based algorithm has been presented. However, the algorithm is based on the recursive application of the CRT itself rather than the recursive application of (12) as in the New CRT II. As the result, the algorithm in [19] still requires the modulo $M = P_1P_2 \cdots P_n$ operations while the New CRT II does not.

Example 4: Using the New CRT II, find the decimal number for the RNS number (1, 2, 3, 4) given the moduli set (3, 5, 7, 11)

$$\begin{aligned} &5k_1 = 1 \bmod 3; \quad 11k_2 = 1 \bmod 7; \quad 77k_3 = 1 \bmod 15 \\ &k_1 = 2; \quad k_2 = 2; \quad k_3 = 8 \\ &N_1 = x_2 + [k_1(x_1 - x_2)]_{P_1} P_2 = 2 + [2 * (-1)]_3 5 = 7 \\ &N_2 = x_4 + [k_2(x_3 - x_2)]_{P_3} P_4 = 4 + [2 * (-1)]_7 11 \\ &\quad = 4 + 5 * 11 = 59 \\ &X = N_2 + [k_3(N_1 - N_2)]_{P_1P_2} P_3P_4 \\ &\quad = 59 + [8(7-59)]_{15} * 77 = 59 + [8 * 8]_{15} * 77 = 59 + 308 \\ &\quad = 367. \end{aligned}$$

The above moduli (3, 5, 7, 11) can represent 10-bit binary numbers. Instead of using modulo 1155 adder, we only need modulo 3, 7, and 15 multipliers. Therefore, the decoding process will be much more efficient.

A converter for 8-element moduli sets based on the New CRT II is shown in Fig. 5. Comparing Fig. 5 to Fig. 2, which is the implementation for the New CRT I, we can see that the two are very similar. They share the structure of multiples P_1P_2, P_5P_6, and $P_1P_2P_3P_4$. They differ in the input and modulo operations. Fig. 2 of New CRT I has input as the first-order radix while Fig. 5 has the input as the coordinates x_i. Fig. 2 requires fewer or no modulo operations while there are modulo operations in Fig. 5.

A general converter based on the New CRT II implements the algorithm *translate*. A block diagram of the converter for size $n = 2^k$ moduli sets is shown in Fig. 6. All cells in Fig. 6 perform the *findno* algorithm. The RNS numbers together with the moduli are paired up as inputs to each cell. At the first level, output x_{ij} $(j = i + 1)$ is produced by the cell whose inputs are x_i, x_j, P_i, P_j such that $x_{ij} = x_i \bmod P_i, x_{ij} = x_j \bmod P_j$, and $x_{ij} < P_{ij} = P_i * P_j$. Outputs from the first level cells are fed to the cells in the second level. This pattern continues until there is only one output from the final level, which is the number we want to find.

It is easy to see that the first level of the tree has n cells, the second level has $\lceil n/2 \rceil$ cells, and so on. The tree is of height $\log_2 n$. Therefore the time delay to decode the RNS number $(x_1, \cdots, x_n)$ is of $\log_2 n$. For $findno(x_1, x_2, P_1, P_2, X)$, the value k_0 is an integer such that $k_0 * P_2 = 1 \bmod P_1$ and is stored in a ROM. For algorithm *translate*, the values of $\prod_{i=k}^{j} P_i$ are also stored in a ROM. The total ROM area required is $n + n/2 + n/4 + \cdots + 1$, which is of $O(n)$. It is interesting to note that the ROM area needed in [27] is $P_1 + P_2 + \cdots + P_n > 1 + 2 + 3 + \cdots + n = O(n^2)$. In addition, no modulo M adder is used.

IV. Literature Summary

We cite a large number of references at the end of this paper [15]–[45]. Those papers were published in many different jour-

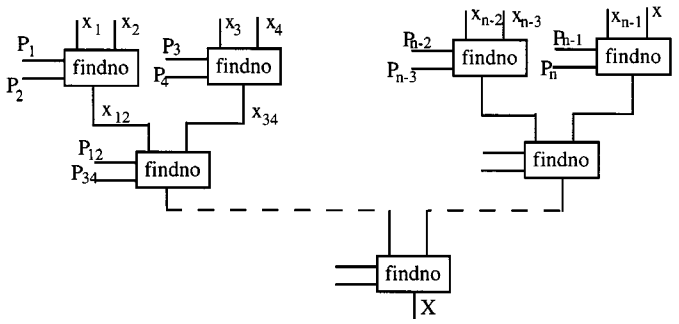

Fig. 6. A general converter based on the New CRT II.

TABLE I
SUMMARY OF PERFORMANCE COMPARISON

algorithms	time	ROM space	special condition	modulo adder size
[27]	$O(\log n)$	$>n^2$	*no*	$n*M$
[38]	$O(\log n)$	$>n^2$	*yes*	$n*M+2^k$
[19]	$O(\log n)$	$O(n)$	*no*	$P_i*...*P_j \le M$
New CRT-II	$O(\log n)$	$O(n)$	*no*	$P_i*...*P_j < \sqrt{M}$
New-CRT-I	$O(\log n)$	*no*	*no*	1 mod$M$
New-CRT-I-special	$O(\log n)$	*no*	yes	0

nals. The list is by no means a complete one. However, it is the longest one compared to the references list in previous papers.

Except for some special moduli sets, almost all previous implementations are based on either the CRT or the MRC approach. The CRT based implementations include the following 13 publications: [23], [25], [27]–[31], [35], [38]–[41], [45]. The MRC approaches include three publications [37], [42], [44]. In recent years, some major effort has been spent on decoding special moduli sets [15]–[18], [20]–[22], [26], [32]–[34], [43]. The most popular special moduli set is the moduli set $\{2^n - 1, 2^n, 2^n+1\}$. Moreover, there are two papers which are general alternatives [19], [36]. The approach in [36] is a table-look-up based approach which can not be applied to general big moduli sets. The algorithm published in [19] is an early version of the New CRT II, which had the best performance at that time.

Table I for performance comparison is a modified version of Table I in [19] to include the New CRT's. In Table I, under the column marked "modulo adder size," $n*M$ means there are n adders used and all of them are modulo $M, n*M+2^k$ means there is an additional modulo 2^k adders besides $n*M$ adders. From Table I, New CRT I and II have the best performance for the general conversion.

V. CONCLUSION

The CRT has many applications in computer arithmetic, digital signal processing, and communications. This paper presents two improved algorithms of the celebrated CRT. Compared to the CRT and its alternative the MRC method, the New CRT's presented here have many advantages which have been demonstrated throughout the paper using examples and analysis.

During the last 20 years, the CRT and the RNS have received intensive attention. The New CRT's presented in this paper will open up many doors for further RNS research. It is expected more efficient arithmetic algorithms can be developed based on the results presented in this paper.

REFERENCES

[1] M. R. Schroeder, *Number Theory in Science and Communication*. Berlin, Germany: Springer-Verlag, 1984.
[2] H. Krishna, B. Krishna, K. Y. Lin, and J. D. Sun, *Computational Number Theory and Digital Signal Processing: Fast Algorithms and Error Control Techniques*. Boca Raton, FL: CRC , 1994.
[3] K. Y. Lin, B. Krishna, and H. Krishna, "Rings, fields, the chinese remainder theorem and an extension—Part I: Theory," *IEEE Trans. Circuits Syst. II* , vol. 41, pp. 641–655, Oct. 1994.
[4] P. W. Beame, S. A. Cook, and H. J. Hoover, "Log depth circuits for division and related problems," *SIAM J. Comput.*, vol. 15, pp. 994–1003, 1986.
[5] T. Leighton, *Introduction to Parallel Algorithms and Architectures: Array, Trees, Hypercubes*. San Mateo, CA: Morgan Kaufmann, 1992.
[6] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, pp. 120–126, 1978.
[7] C. Chiang and L. Johnsson, "Residue arithmetic and VLSI," *Proc. IEEE ICCD*, pp. 80–83, 1983.
[8] D. K. Banerji, "On the use of residue arithmetic for computation," *IEEE Trans. Comput.*, vol. C-21, pp. 1315–1317, Dec. 1974.
[9] I. Koren, *Computer Arithmetic Algorithms*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[10] N. S. Szabo and R. I. Tanaka, *Residue Arithmetic and Its Application to Computer Technology*. New York: McGraw-Hill, 1967.
[11] F. J. Taylor, "Residue arithmetic: A tutorial with examples," *IEEE Comput.*, vol. C-17, pp. 50–62, May 1984.
[12] M. A. Soderstrand, W. K. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Signal Processing*. New York: IEEE Press, 1986.
[13] G. Dimauro, S. Impdedovo, and G. Pirlo, "A new technique for fast number comparison in the residue number systems," *IEEE Trans. Comput.*, vol. 42, pp. 608–612, May 1993.
[14] M. Lu and J. Chiang, "A novel division algorithm for the residue number system," *IEEE Trans. Comput.*, vol. 41, pp. 1026–1032, Aug. 1992.
[15] Y. Wang, X. Song, and M. Aboulhamid, "Near-optimal residue to binary converter for the moduli," presented at the 8th Great Lakes Symp. VLSI, Lafayette, LA, Feb. 1998.
[16] Y. Wang, M. N. Swamy, and O. Ahmad, "Three number moduli sets based residue number systems," presented at the 1998 IEEE Int. Symp. Circuits and Systems, Monterey, CA, May 31–June 3 1998.
[17] F. Pourbigharaz and H. Yassine, "A signed-digit architecture for residue to binary transformation," *IEEE Trans. Comput.*, vol. 46, pp. 1146–1150, Oct. 1997.
[18] D. Gallaher, F. Petry, and P. Srinivasan, "The digital parallel method for fast RNS to weighted number system conversion for specific moduli," *IEEE Trans. Circuits Syst. II*, vol. 44, pp. 53–57, Jan. 1997.
[19] Y. Wang and M. Abd-el-barr, "A new algorithm for RNS decoding," *IEEE Trans. Circuits Syst. I*, vol. 43, pp. 998–1001, Dec. 1996.
[20] S. Piestrak, "A high-speed realization of a residue to binary number system converter," *IEEE Trans. Circuits Syst. II*, vol. 42, pp. 661–663, Oct. 1995.
[21] A. Premkumar, "An RNS to binary converter in a three moduli set with common factors," *IEEE Trans. Circuits Syst. II*, vol. 42, pp. 298–301, Apr. 1995.
[22] B. Vinnakota and V. V. Rao, "Fast conversion techniques for binary-residue number systems," *IEEE Trans. Circuits Syst. I*, vol. 41, pp. 927–929, Dec. 1994.
[23] F. Barsi and M. Pinotti, "A fully parallel algorithm for residue to binary conversion," *IPL*, pp. 1–8, 1994.
[24] G. Alia and E. Martinelli, "On the lower bound to the VLSI complexity of number conversion from weighted to residue representation," *IEEE Trans. Comput.*, vol. 42, pp. 962–967, 1993.
[25] C. N. Zhang, B. Shirazi, and D. Yun, "An efficient algorithm and parallel implementations for binary and residue number systems," *J. Symbolic Computation*, vol. 15, pp. 451–462, 1993.
[26] A. Premkumar, "An RNS to binary converter in $2n+1, 2n, 2n-1$ moduli set," *IEEE Trans. Circuits Syst.*, vol. 39, pp. 480–482, 1992.
[27] K. Elleithy and M. Bayoumi, "Fast and flexible architectures for RNS arithmetic decoding," *IEEE Trans. Circuits Syst.*, vol. 39, pp. 226–235, Apr. 1992.
[28] J. Kim, K. Park, and H. Lee, "Efficient residue-to-binary conversion technique with rounding error compensation," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 315–317, Mar. 1991.
[29] S. Meehan, S. O'neil, and J. Vaccaro, "An universal input and output RNS converter," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 799–803, 1990.

[30] G. Alia and E. Martinelli, "VLSI binary-to-residue converters for pipelined processing," *The Comput. J.*, vol. 33, no. 5, pp. 473–475, 1990.
[31] C. N. Zhang, B. Shirazi, and D. Yun, "Parallel designs for chinese remainder coversion," in *Proc. 16th Int. Conf. Parallel Processing*, Aug. 1989.
[32] A. Shenoy and R. Kumaresan, "Residue to binary conversion for RNS arithmetic using only modular look-up tables," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1158–1162, Sept. 1988.
[33] K. Ibrahim and S. Saloum, "An efficient residue to binary converter design," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1156–1158, Sept. 1988.
[34] S. Andraos and H. Ahmad, "A new efficient memoryless residue to binary converter," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1441–1444, Nov. 1988.
[35] R. M. Capocelli and R. Gian Carlo, "Efficient VLSI networks for converting an integer from binary system to residue number system and vice versa," *IEEE Trans. Circuits Syst.*, vol. 35, no. 11, pp. 1425–1430, Nov. 1988.
[36] R. Thun, "On residue number system decoding," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, no. 5, pp. 1346–1347, Oct. 1986.
[37] N. Chakraborti, J. Soundararajan, and A. Reddy, "An implementation of mixed-radix conversion for residue number applications," *IEEE Trans. Comput.*, vol. C-35, pp. 762–764, Aug. 1986.
[38] T. Van Vu, "Efficient implementations of the Chinese reminder theorem for sign detection and residue decoding," *IEEE Trans. Comput.*, vol. C-34, pp. 646–651, July 1985.
[39] F. Taylor and W. Dirr Jr., "A new residue to decimal converter," *Proc. IEEE*, vol. 73, pp. 378–340, Feb. 1985.
[40] G. Alia and E. Martinelli, "A VLSI algorithm for direct and reverse conversion from weighted binary number system to residue number system," *IEEE Trans. Circuits Syst.*, vol. CAS-31, pp. 1033–1039, Dec. 1984.
[41] G. Alia, F. Barsi, and E. Martinelli, "A fast VLSI conversion between binary and residue systems," *IPL*, vol. 18, pp. 141–145, 1984.
[42] C. Huang, "A fully parallel mixed-radix conversion algorithm for residue number applications," *IEEE Trans. Comput.*, vol. C-32, pp. 398–402, Apr. 1983.
[43] F. Taylor and A. S. Ramnarayanan, "An efficient residue-to-decimal converter," *IEEE Trans. Circuits Syst.*, vol. CAS-28, pp. 1164–1169, Dec. 1981.
[44] A. Baraniecka and G. Jullien, "On decoding techniques for residue number system realizations of digital signal proceesing hardware," *IEEE Trans. Circuits Syst.*, vol. CAS-25, pp. 935–936, Nov. 1978.
[45] D. Banerji and J. Brzozowski, "On translation algorithms in residue number systems," *IEEE Trans. Comput.*, vol. C-21, pp. 1281–1285, Dec. 1972.
[46] H. L. Garner, "The residue number system," *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 140–147, June 1959.
[47] Y. Wang, "New Chinese remainder theorems," in *Proc. 32nd Asilomar Conf. Signals, Systems, Computers*, vol. 1, 1998, pp. 165–171.

Yuke Wang received the B.Sc. degree from the University of Science and Technology of China, Hefei, China, in 1989, and the M.Sc. and the Ph.D. degrees from the University of Saskatchewan, Saskatoon, Canada, in 1992 and 1996, repectively.

From September 1995 to May 1996, he was a postdoctoral Fellow at the University of Montreal, Montreal, Canada. From June 1996 to May 1999, he was an Assistant Professor at Concordia University, Montreal, Canada. He was a Visiting Assistant Professor in the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis St. Paul, during the summer of 1999. Since August 1999, he has been with the Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, where he is an Assistant Professor. His research interests include VLSI ASIC design of digital signal processing and communication applications, computer arithmetic, and computer-aided design tools for all levels of VLSI design.