A New Algorithm for RNS Magnitude Comparison Based on New Chinese Remainder Theorem II

Yuke Wang
Department of Electrical and Computer Engineering
Concordia University
1455 de Maisonneuve Blvd. W. H961-33
Montreal, Quebec
Canada H3G 1M8

Xiaoyu Song, Mostapha Aboulhamid
Departement d'informatique et de recherche operationnelle
Universite de Montreal
Montreal, Quebec
Canada H3C 3J7

Abstract

The number comparison is a difficult and fundamental operation for residue number systems (RNS). Previous algorithms use either some redundant modulus or big modulo operations. In this paper, based on the New Chinese Remainder Theorem II, we present a new comparison algorithm using smaller modulo operations and no redundant modulus.

1. Introduction

There has been interest in residue number arithmetic as a basis for computational hardware since the 1950's [1] [2]. During the past decade, the residue number system (RNS) has received a considerable interest in arithmetic computation and signal processing applications, such as fast Fourier transforms, digital filtering and image processing. The main reasons for the wide spread use of the RNS are its inherent properties such as parallelism, modularity, fault tolerance, and carry free operations [3] resulting in fast additions and multiplications. However, due to the non-position nature of the RNS, the comparison between RNS numbers is not as simple as in the weighted number systems. So is for overflow, sign detection, and division. Because of those difficulties, the residue arithmetic has not been used widely for general purpose computation [1] [2] [5][7].

The traditional techniques for RNS number comparison use the celebrated Chinese Remainder Theorem (CRT) or the Mixed Radix Conversion (MRC) [2]. A direct implementation of the CRT is unprofitable since it is based on a modulo M operation, where M is the large dynamic range of the RNS [5]. The MRC is a sequential process requiring long delay. Some techniques are based on the "core" functions [9] or the "diagonal functions" [5]. Other techniques use redundant modulus and assume special conditions on the moduli sets [4].

In [10], we proposed a new residue-to-binary conversion algorithm called the New Chinese Remainder Theorem II, which improves the celebrated CRT in many aspects. To convert a residue number to its decimal correspondence, the New Chinese Remainder Theorem II uses modulo multipliers of size less than the square root of the range M. Based on the New Chinese Remainder Theorem II, we propose a new RNS comparison algorithm in this paper. Compared with previous algorithms in the literature, the new algorithm uses smallest modulo operations; assumes no special conditions on the moduli sets; uses no redundant modulus. The delay of the new algorithm is $O(\log n)$, where n is the size of the moduli set.

2. Background Materials

In this section, we introduce the background materials. For any two numbers X and P_i, $x_i = X \bmod P_i$ is defined

0-7695-0104-4/99 $10.00

as $X = x_i + bP_i$ for some integer b such that $0 \le x_i < P_i$. $X \bmod P_i$ is denoted as X_{P_i}.

A Residue Number System is defined in terms of a set of relatively prime moduli set $(P_1, P_2, ..., P_n)$, i.e., the $(P_i, P_j) = 1$ for $i \neq j$. A binary number $X \in [0, M-1]$ has a unique residue number as $X = (x_1, x_2, ..., x_n)$, where $x_i = X \bmod P_i$, $M = P_1 ... P_n$ is the *dynamic range* of the moduli $(P_1, P_2, ..., P_n)$.

To convert a residue number $(x_1, x_2, ..., x_n)$ into its binary representation X, the Chinese Remainder Theorem is 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 $P_i \bmod P_j$.

Chinese Remainder Theorem (CRT) The number X can be computed as

$$X = [\Sigma_{i=1}^{n} s_i \, |x_i / s_i|_{P_i}] \bmod M$$

where $s_i = M / P_i$, $|1/s_i|_{P_i}$ is the multiplicative inverse of $s_i \bmod P_i$.

The CRT requires large modulo operation of size M. In the following, we introduce the New Chinese Remainder Theorem - II which reduces the size of the modulo operations. We first consider the moduli set of two numbers $P = (P_1, P_2)$, where $P_1 < P_2$.

Proposition 1 The corresponding decimal number for (x_1, x_2) can be found using the formula $X = x_2 + [k_0 (x_1 - x_2)]_{P_1} P_2$, where k_0 is a positive integer which satisfies the condition that $k_0 * P_2 = 1 \bmod P_1$. Since $GCD(P_1, P_2) = 1$, 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 to use big modulo $M = P_2 P_1$ adders; instead the smaller modulo P_1 multiplier is used, $P_1 < \sqrt{M}$.

New Chinese Remainder Theorem II Suppose the general moduli set to be $(P_1, P_2, ..., P_n)$ with $P_1 < P_2 < ... < P_n$. The algorithm, *translate*, finds the correct decimal representation of the RNS number $X = (x_1, x_2, ..., x_n)$.

Algorithm *translate*$((x_1, x_2, ..., x_n), (P_1, ..., P_n), X)$

(1) if $n>2$, let $t = \lfloor n/2 \rfloor$, then

translate$((x_1, ..., x_t), (P_1, ..., P_t), N_1)$, $M_1 = P_1 ... P_t$,

translate$((x_{t+1}, ..., x_n), (P_{t+1}, ..., P_n), N_2)$, $M_2 = P_{t+1} ... P_n$,

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

(2) if $n=2$, then *findno*(x_1, x_2, P_1, P_2, X).

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)

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

The difference between the New Chinese Remainder Theorem II and the CRT is obvious. The size of the modulo multipliers in the New CRT-II is bounded by $\sqrt{M}$. In the literature, similar divide-and-conquer approach has been used for implementing the CRT. However, they all have modulo $M = P_1 P_2 ... P_n$ operations.

Example 1 Find the number X=(1, 2, 3, 4) with the moduli (3,5,7,11).

Using the CRT, the computation is done as follows.

$M = 3*5*7*11 = 1155$

$s_1 = 385, \quad s_2 = 231, \quad s_3 = 165, \quad s_4 = 105,$

$|1/s_1|_{P_1} = 1, \quad |1/s_2|_{P_2} = 1, \quad |1/s_3|_{P_3} = 2, \quad |1/s_4|_{P_4} = 2$

$X = [385*1*x_1 + 231*1*x_2 + 165*2*x_3 + 105*2*x_4]_{1155}$ (3)

For $(x_1, x_2, x_3, x_4) = (1,2,3,4)$,

$X = [385 + 462 + 990 + 840]_{1155} = 2677_{1155} = 367$.

Applying the New Chinese Remainder Theorem II, the computation is done as follows.

$5k_1 = 1 \bmod 3; \quad 11k_2 = 1 \bmod 7; \quad 77k_3 = 1 \bmod 15$

$k_1 = 2; \quad k_2 = 2; \quad k_3 = 8$

$N_1 = x_2 + [k_1 (x_1 - x_2)]_{P_1} P_2 = 2 + [2*(-1)]_3 5 = 7$

$N_2 = x_4 + [k_2 (x_3 - x_2)]_{P_3} P_4 = 4 + [2*(-1)]_7 11 = 4 + 5*11 = 59$

$X = N_2 + [k_3 (N_1 - N_2)]_{P_1 P_2} P_3 P_4 = 59 + [8(7-59)]_{15} * 77$

$= 59 + [8*8]_{15} * 77 = 59 + 308 = 367$

The above moduli (3, 5, 7, 11) can represent 10-bit binary numbers. Instead of using modulo 1155 adder, the computation based on the New Chinese Remainder Theorem II needs only modulo 3, 7, and 15 multipliers. Such small residue multipliers can be easily implemented by ROM look-up table.

3. A New RNS Number Comparison Algorithm

In this section, we propose a new RNS number comparison algorithm based on the New Chinese Remainder Theorem II. In the following, the moduli set is $(P_1, P_2, ..., P_n)$, $(P_i, P_j) = 1$, $M = P_1 * P_2 ... * P_n$. Two binary numbers are $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$. If the moduli set is the set (P_1, P_2), then $X = (x_1, x_2)$, $Y = (y_1, y_2)$.

Algorithm *RNS_Compare*$((x_1, x_2, ..., x_n), (y_1, y_2, ..., y_n))$

(1) if $n>2$, let $t = \lfloor n/2 \rfloor$, then

translate$((x_1, ..., x_t), (P_1, ..., P_t), X_1)$,

translate$((x_{t+1}, ..., x_n), (P_{t+1}, ..., P_n), X_2)$,

translate$((y_1, ..., y_t), (P_1, ..., P_t), Y_1)$,

translate$((y_{t+1}, ..., y_n), (P_{t+1}, ..., P_n), Y_2)$,

$M_1 = P_1 ... P_t$, $M_2 = P_{t+1} ... P_n$,

RNS_Compare_size2$((X_1, X_2), (Y_1, Y_2), (M_1, M_2))$.

(2) if $n=2$, then

RNS_Compare_size2$((x_1,x_2),(y_1,y_2),(P_1,P_2))$.

Procedure

RNS_Compare_size2$((x_1,x_2),(y_1,y_2),(P_1,P_2))$

(1) find a k_0 such that $k_0 * P_2 = 1 \bmod P_1$,

(2) if $x_1 = y_1$ and $x_2 = y_2$, then $X = Y$,

(3) if $[k_0(x_1 - x_2)]_{P_1} > [k_0(y_1 - y_2)]_{P_1}$, then

$X = (x_1,x_2) > Y = (y_1,y_2)$,

else if $[k_0(x_1 - x_2)]_{P_1} < [k_0(y_1 - y_2)]_{P_1}$, then

$X = (x_1,x_2) < Y = (y_1,y_2)$,

else if $[k_0(x_1 - x_2)]_{P_1} = [k_0(y_1 - y_2)]_{P_1}$, then

if $x_1 > y_1$, then $X = (x_1,x_2) > Y = (y_1,y_2)$,

else if $x_1 < y_1$, then $X = (x_1,x_2) < Y = (y_1,y_2)$.

It is not hard to verify the above algorithm gives correct answers for RNS number comparison. In the following, we show an example. Example 2 is from [5], where the most recent RNS number comparison algorithm was given.

Example 2 The moduli set is $(P_1,P_2,P_3,P_4,P_4) = (5,11,14,17,9)$, $M = 117810$, we want to compare the size of two RNS numbers $X = (3,5,11,8,7)$, $Y = (0,0,10,1,2)$.

Using the algorithm in [5], we have the following computations:

$Q_i = M / P_i, i = 1,2,3,4,5, \quad SQ = Q_1 + Q_2 + Q_3 + Q_4 + Q_5$

$s_i * P_i = 1 \bmod SQ, \quad i = 1,2,3,4,5,$

$Q_1 = 23562, \quad Q_2 = 10710, \quad Q_3 = 8415, \quad Q_4 = 6930, \quad Q_5 = 13090$

$SQ = Q_1 + Q_2 + Q_3 + Q_4 + Q_5 = 62707$

$s_1 = 37624, \quad s_2 = 45605, \quad s_3 = 4479, \quad s_4 = 51641, \quad s_5 = 48772$

$D(X) = |s_1x_1 + s_2x_2 + s_3x_3 + s_4x_4 + s_5x_5|_{SQ}$

$= |37624 * 3 + 45605 * 5 + 4479 * 11 + 51641 * 8 + 48772 * 7|_{62707}$

$= 15972$

$D(Y) = |s_1y_1 + s_2y_2 + s_3y_3 + s_4y_4 + s_5y_5|_{SQ}$

$= |37624 * 0 + 45605 * 0 + 4479 * 10 + 51641 * 1 + 48772 * 2|_{62707}$

$= 5854$

$D(X) > D(Y) \Rightarrow X > Y$

Using the algorithm *RNS_Compare,* we need the following constants.

$t = \lfloor 5/2 \rfloor = 2, \quad M_1 = P_1P_2 = 5 * 11 = 55, \quad M_2 = P_3P_4P_5 = 14 * 17 * 9 = 2142$

$k_1 * P_2 = 1 \bmod P_1, \quad k_2 * P_4 = 1 \bmod P_5, \quad k_3 * P_4P_5 = 1 \bmod P_3, \quad k_4 * P_3P_4P_5 = 1 \bmod P_1P_2$

$k_1 = 1, \quad k_2 = -1, \quad k_3 = -1, \quad k_4 = 18$

Given the above constants, the comparison is done as follows.

$X_1 = x_2 + [k_1(x_2 - x_1)]_{P_1}P_2 = 5 + [1 * (3 - 5)]_5 11 = 38$

$Y_1 = y_2 + [k_1(y_2 - y_1)]_{P_1}P_2 = 0 + 0 * 11 = 0$

$X_{21} = x_4 + [k_2(x_5 - x_4)]_{P_5}P_4 = 8 + [(-1) * (7 - 8)]_9 17 = 25$

$X_2 = X_{21} + [k_3(x_3 - X_{21})]_{P_3}P_4P_5 = 25 + [(-1)(11 - 25)]_{14} 17 * 9 = 25$

$Y_{21} = y_4 + [k_2(y_5 - y_4)]_{P_5}P_4 = 1 + [(-1) * (2 - 1)]_9 17 = 137$

$Y_2 = Y_{21} + [k_3(y_3 - Y_{21})]_{P_3}P_4P_5 = 137 + [(-1)(10 - 137)]_{14} 17 * 9 = 290$

$[k_4(X_1 - X_2)]_{P_1P_2} = [18(38 - 25)]_{55} = 14, \quad [k_4(Y_1 - Y_2)]_{P_1P_2} = [18(0 - 290)]_{55} = 5$

$[k_4(X_1 - X_2)]_{P_1P_2} > [k_4(Y_1 - Y_2)]_{P_1P_2} \Rightarrow X > Y$

It is shown that using our new algorithm, we need modulo operations of size P_1, P_5, P_3, P_1P_2. The maximum size is $P_1P_2 = 55$, which is to be compared with the modulo $SQ = 62707$. The number SQ is 1000 times bigger.

4. Performance Evaluation

In this section, we present the performance evaluation of our new algorithm and compare it with previous related algorithms.

The delay of the algorithm *RNS_Compare* is $O(\log n)$, where n is the size of the moduli set. This is the same as all other algorithms based on CRT.

Modulo operations are needed for the algorithm *RNS_Compare.* The maximum size modulo operations for the operation *translate*$((x_1,...,x_t),X_1)$ and *translate*$((x_{t+1},...,x_n),X_2)$ are of size less than $\sqrt{M_1}$ and $\sqrt{M_2}$; while the modulo operation for the operation *RNS_Compare_size2*$((X_1,X_2),(Y_1,Y_2)(M_1,M_2))$ is of size M_1. Therefore the maximum size modulo operation for the algorithm *RNS_Compare* is of size M_1 which is less than $\sqrt{M}$.

The following Table 1 is the summary of the performances of previous proposed RNS number comparison algorithms. The "operation size" refers to the largest modulo operations or the largest integers involved in the operation. It is easy to see that our new algorithm is the best RNS number comparison algorithm based on the criteria listed in Table 1.

5. Conclusion

The number comparison is a difficult and fundamental operation for residue number systems. Because of difficulties related to the number comparison, the residue arithmetic has not been used widely for general purpose computation.

We have presented a new comparison algorithm based on the New Chinese Remainder Theorem II. Compared with previous RNS number comparison algorithms, the new algorithm uses smallest modulo operation and no redundant modulus. Moreover, it assumes no special conditions on the moduli sets.

The number comparison algorithm presented here is an application of the New Chinese Remainder Theorems developed in [10]. The New Chinese Remainder Theorems offer many other advantages compared to the traditional CRT and the MRC method. It is expected that they will play an important role in residue arithmetic.

Table 1. Performance Evaluation of Different Algorithms

	Delay	Operation size	Special conditions	Redundant Moduli
[4]	$O(\log n)$	$\log nM$ - bit	yes	yes
[5]	$O(\log n)$	modulo $\sum_{i=1}^{n} \frac{M}{P_i}$	no	no
[7]	$O(n)$	$\sum_{i=1}^{n} w_i[\frac{M}{P_i}]$	no	no
[9]	$O(\log n)$	$\sum_{i=1}^{n} w_i[\frac{M}{P_i}]$	no	yes
New Algorithm	$O(\log n)$	modulo $\prod_{i=1}^{n/2} P_i$	no	no

References

[1] H. L. Garner, "The residue number system", IRE Trans. Electronic Computers, Vol. EC-8, pp. 140-147, June 1959.

[2] N. S. Szabo and R. I. Tanaka, "Residue Arithmetic and Its Applications to Computer Technlogy", New York, McGraw-Hill, 1967.

[3] Fred J. Taylor "Residue Arithmetic: A Tutorial with Examples", IEEE Computer, vol. 17, No. 5, May 1984, pp. 50-62.

[4] M. Lu and J. Chiang, "A novel division algorithm for the residue number system", IEEE Transactions on computers, Vol. 41, No. 8, August 1992.

[5] G. Dimauro, S. Impedovo, and G. Pirlo, "A New Technique for Fast Number Comparison in Residue Number System", IEEE Transactions on Comptuers, Vol. 42, No. 5, May 1993, pp. 608-612.

[6] T. Cormen, C. Leiserson, and R. Rivest, "Introduction to Algorithms", the MIT Press, 1990.

[7] D. D. Miller, J. N. Polky, and J. R. King, "A survey of Soviet development in residue number theory applied to digital filtering", Proc. 26th Midwest Symposimu. Circuits and Systems., Aug. 1983.

[8] Thu Van Vu, "Efficient Implementations of the Chinese Reminder Theorem for Sign Detection and Residue Decoding" , IEEE Transactions on Computer, vol. c-34, No. 7, July, 1985, pp. 646-651.

[9] D. D. Miller, R. E. Altschul, J. R. King and J. N. Polky, "Analysis of the residue class core function of Akushskii, Burcev and Pak", in *Residue Number System Arithmetic, Modren Applications in Digital Signal Processing*, M. A. Soderstrand, W. C. Jenkins, G. A. Jullien, and F. J. Taylor, eds. New York, IEEE Press, paper 7-2, 1985, pp. 390-401.

[10] Yuke Wang, "New Chinese Remainder Theorem with Applications for DSP", the Thirty second annual Asilomar Conference on Signals, Systems, and Computers, November 1 - November 4, 1998, USA.

[11] Yuke Wang and M. Abd-el-Barr, "A new Algorithm for RNS Decoding", IEEE Transacitons on Circuits and Systems, Part I, Vol. 43, No. 12, December 1996.