INCOMPLETELY SPECIFIED NUMBERS IN THE RESIDUE NUMBER SYSTEM - DEFINITION AND APPLICATIONS

Dragan Gamberger

Ruđer Bošković Institute
41000 Zagreb P.O.B. 1016
Yugoslavia

ABSTRACT

Incompletely specified numbers in the residue number system (RNS) are defined with object to enable multiplicative inverse computation of a number regardless to its magnitude. The introduced incompletely specified RNS represents the general RNS model in which completely specified numbers are the special case. Two efficient algorithms for transformation of incompletely to completely specified RNS numbers are shown. Examples of their application in divisibility testing and integer matrix inversion are described.

1. INTRODUCTION

The residue number system (RNS) is a nonweighted number system in which the presentation of a number takes the form of a N-tuple $X = (x_1, x_2, \ldots, x_N)$ where $x_i = X$ modulo m_i is the i-th residue digit and m_i is the i-th modulus. If all m_i are relatively prime numbers, there is a unique representation for each positive integer in the range $X < M$ where $M = \prod_{i=m}^{N} m_i$. For simplicity, in this paper it is supposed that all m_i are prime numbers. The main characteristic of the RNS is that from $Z = X \circ Y$, where $\circ$ denotes addition, subtraction or multiplication, follows $z_i = \langle x_i \circ y_i \rangle$ modulo m_i. That means that these operations can be done independently for each modulus and that fast parallel computations can be realized [1]. When the moduli are sufficiently small, it is possible to realize residue operations with look-up tables stored in ROM's. Addition, subtraction and multiplication are called elementary operations because they can be executed in only one parallel memory access [2]. But this property is not valid for integer division while RNS is not closed under this operation. Moreover, integer division is necessary iterative operation and for its realization much more hardware is required than for all other operations together, and its execution time is incomparably longer [1],[5].

Besides integer division, in the RNS exists division that can be executed as multiplication by the multiplicative inverse element of the divisor. Such division, named direct division in this paper, represents the fourth elementary operation in the RNS. The result of the direct division is equal to the result of the integer division if and only if it is division with zero remainder [1].

The main restriction on the extensive use of the direct division is the fact that the multiplicative inverse element of a RNS number is not known for the moduli in which the number has value "0". Because of that, direct division can be executed only if divisor is relatively prime to M. In this paper a RNS allowing direct division to be executed always regardless of the divisor's magnitude will be introduced. Such system will be called incompletely specified RNS (ISRNS) and the known RNS will be called completely specified RNS to point out the difference.

In the section 2 of this paper the definition of the ISRNS is given. In the section 3 and 4 algorithms for the transformation of ISRNS numbers to the completely specified ones are presented and finally in the section 5 some examples for the ISRNS utilization are described.

2. DEFINITION OF INCOMPLETELY SPECIFIED RNS NUMBERS

Let us define that besides m_i "normal" states for the i-th residue coding $0,1,2, \ldots, m_i-2, m_i-1$, there is an additional (m_i+1)-th state "u" which denotes an unspecified value of the residue digit. A RNS number is said to be incompletely specified if it can have unspecified values for some of its digits and accordingly the ISRNS is defined as a system in which it is possible to represent incompletely specified RNS numbers.

In the ISRNS there is a unique representation for each positive integer in the range $X < M/M_u$, or each positive and negative integer in the range $X \leq \lfloor (M/M_u - 1)/2 \rfloor$ where M_u is the product of moduli in which the number has unspecified value and where $\lfloor z \rfloor$ denotes the greatest integer less or equal to z. M_u can be defined by

$$M_u = \prod_{i=1}^{N} (m_i)^{\delta_i} \qquad (1)$$

where δ_i is a binary variable denoting whether the residue digit corresponding to the modulus m_i is specified ($\delta_i = 1$) or not ($\delta_i = 0$).

Elementary residue operations on ISRNS numbers are executed in the same way as on completely specified RNS numbers except that the result takes unspecified value in the moduli in which any of operands has unspecified value. In the ISRNS direct division can be always executed because it is pos-

sible to assign the unspecified state "u" to the multiplicative inverse element of "0". (Multiplicative inverse element of "u" is not "0" but "u" itself) . After multiplication with such multiplicative inverse element, the final result of the direct division becomes the incompletely specified number with "u" digits in all positions in which divisor has value "0". Direct division is the only arithmetic operation which can generate the ISRNS numbers from the completely specified operands.

Another way of generating ISRNS numbers is possible in the case when transmissional or computational errors can influence residue digits [4]. In this case all illegitimate residue digit codes can be treated as unspecified state "u", resulting in a legitimate ISRNS number and allowing to continue normal computations in spite of encountered errors.

The use of the ISRNS numbers can be of great practical value in division with zero remainder, in error detection and correction systems and some nonelementary residue operations if there exists efficient way for transformation of the incompletely specified results into the completely specified ones. In the next section a general algorithm will be presented which enables transformation of the incompletely specified numbers to the completely specified RNS. In the section 4 an algorithm will be presented enabling faster transformation in the case when the starting and the resulting systems have relatively prime moduli.

3. SPECIFICATION OF INCOMPLETELY SPECIFIED RNS NUMBERS

The transformation of an ISRNS positive number to the completely specified one, can always be done by the well known base extension algorithm [1],[2], [4]. This is not a suitable solution for the systems with great number of moduli because each possible combination of unspecified digits requires distinct base extension hardware. Here is described an algorithm for the transformation of the ISRNS positive numbers to the completely specified ones, using the same hardware regardless of the combination of unspecified digits.

The idea of the known base extension algorithm is to convert the representation of a number to the associated mixed radix system (MRS) and from it then to compute the value of digits in additional moduli. Number X is in a MRS represented by

$$X = a_1 + \sum_{i=2}^{N} a_i \cdot T_i \qquad T_i = \prod_{j=1}^{i-1} r_j \qquad (2)$$

and the MRS is associated to the RNS if for all N radices it holds $r_i = m_i$. Any RNS digit x_k can be computed from the MRS representation in N-1 steps of modulo operations realized by look-up tables with two variables per table

$x_k = \langle\langle\langle a_1 + a_2 \cdot T_2 \rangle \bmod m_k + a_3 \cdot T_3 \rangle \bmod m_k + \ldots + a_N \cdot T_N \rangle \bmod m_k$. Conversion from a RNS to the associated MRS is done by the known algorithm [1]

$$a_1 = x_1 \qquad (3)$$

$$a_i = \left\langle \left\lfloor \frac{X}{m_1 \cdot m_2 \cdot \ldots \cdot m_{i-1}} \right\rfloor \right\rangle \bmod m_i \qquad \text{for } i=2,\ldots,N .$$

As the computation of a_i takes only i-1 steps of modulo operations, the transformation from MRS to the new RNS can be done in parallel, with one step delay, with the RNS to the MRS transformation. Because of that, the hole base extension algorithm takes N steps of modulo operations.

Let us now define that the MRS is incompletely specified if some of its digits can have unspecified value. Some digit a_i has unspecified value if its corresponding digit x_i in the associated RNS has unspecified value. The value of a number X represented in a incompletely specified MRS (ISMRS) is defined by

$$X = \delta_1 \cdot a_1 + \sum_{i=2}^{N} \delta_i \cdot a_i \cdot Y_i , \qquad Y_i = \prod_{j=1}^{i-1} (m_j)^{\delta_j} . \qquad (4)$$

Conversion from the ISRNS to the ISMRS can be done again in N-1 steps of modulo operations and is defined by following algorithm

$$a_1 = x_1 \qquad (5)$$

$$a_i = \left\langle \left\lfloor \frac{X}{(m_1)^{\delta_1} \cdot (m_2)^{\delta_2} \cdot \ldots \cdot (m_{i-1})^{\delta_{i-1}}} \right\rfloor \right\rangle \bmod m_i \qquad \text{for } i=2,\ldots,N .$$

This algorithm is shown in the upper part of figure 1 with look-up tables marked by * . Functions realized by the look-up tables of the j-th step (j= 1,...,N-1) and for moduli m_i (i = j+1,...,N) are:

$$x_{i,j} = \left\langle \frac{x_{i,j-1} - x_{j,j-1}}{m_j} \right\rangle \bmod m_i \qquad (6)$$

if $x_{j,j-1} \neq u$ and $x_{i,j-1} \neq u$, else

$$x_{i,j} = x_{i,j-1}$$

where $x_{i,0} = x_i$ is the starting RNS representation of the number X . Digits of the number X represented in ISMRS are:

$$a_i = x_{i,i-1} \qquad \text{for } i=1,\ldots,N .$$

For the realization of this algorithm one needs the same quantity of look-up tables as for the original RNS to MRS transformation algorithm. For m > 2 , introduction of the additional state for the unspecified state will not require additional coding lines, so that the algorithm for the transformation of ISRNS to the ISMRS is of the same hardware complexity as the corresponding algorithm for completely specified systems.

Such statement is not true for the MRS to RNS conversion if it is done in the same way as in the original algorithm

$$x_i = \langle\langle\langle \delta_1 \cdot a_1 + \delta_2 \cdot a_2 \cdot Y_2 \rangle \bmod m_i + \delta_3 \cdot a_3 \cdot Y_3 \rangle \bmod m_i + \ldots + \delta_N \cdot a_N \cdot Y_N \rangle \bmod m_i , \qquad (7)$$

because expressions for Y_i are variables and their value depends on all δ_j ($j < i$). Direct implementation of relation (7) would require additional inputs to look-up tables and would result in growth of the necessary memory space.

Relation (4) can also be written as:

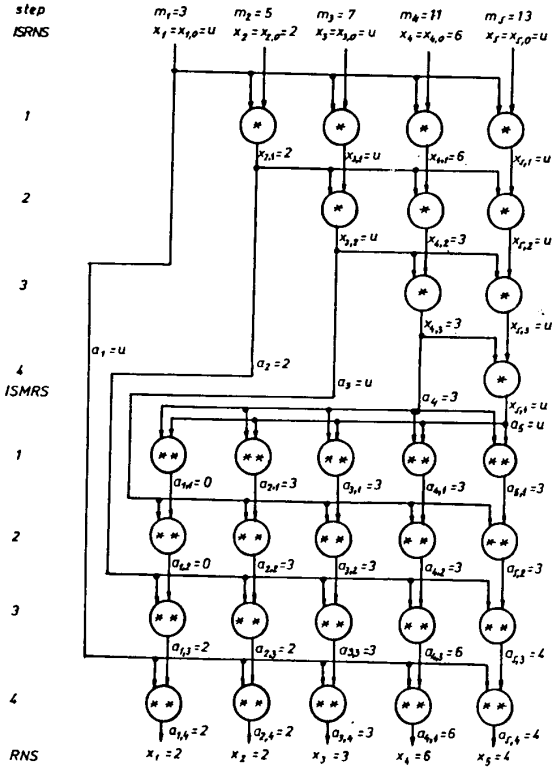

Fig.1. Specification of incompletely specified numbers

$$X = \delta_1 \cdot a_1 + (m_1)^{\delta_1} \cdot (\delta_2 \cdot a_2 + (m_2)^{\delta_2} \cdot (\delta_3 \cdot a_3 + \ldots + (m_{N-1})^{\delta_{N-1}} \delta_N \cdot a_N)) \quad (8)$$

This relation can also be realized in N-1 steps but in this realization additional inputs are not necessary, because information about δ_i is required only in the step when calculations with a_i are done. So, using relation (8), transformation from ISRNS to any completely specified RNS can be done with the hardware of the same complexity as the corresponding algorithm for the completely specified MRS. The only difference is that now transformation starts with the digit a_N so that RNS to MRS and MRS to RNS transformations can not be done in parallel. It means that complete base extension algorithm of an ISRNS takes 2N-2 steps of modulo operations.

Conversion from the ISMRS to a completely specified RNS is shown in the lower part of figure 1 with look-up tables marked ** . Functions realized by look-up tables for moduli m_i (i = 1,...,N) are: for the first step

$$a_{i,1} = \langle a_{N-1} + m_{N-1} \cdot a_N \rangle \bmod m_i \quad \text{if } a_{N-1} \neq u \text{ and } a_N \neq u$$
$$a_{i,1} = \langle a_N \rangle \bmod m_i \quad \text{if } a_{N-1} = u \text{ and } a_N \neq u$$
$$a_{i,1} = \langle a_{N-1} \rangle \bmod m_i \quad \text{if } a_{N-1} \neq u \text{ and } a_N = u$$
$$a_{i,1} = u \quad \text{if } a_{N-1} = u \text{ and } a_N = u$$

for the j-th step (j=2,...,N-1)

$$a_{i,j} = \langle a_{N-j} + m_{N-j} \cdot a_{i,j-1} \rangle \bmod m_i \quad \text{if } a_{N-j} \neq u \text{ and } a_{i,j-1} \neq u$$
$$a_{i,j} = \langle a_{i,j-1} \rangle \bmod m_i \quad \text{if } a_{N-j} = u \text{ and } a_{i,j-1} \neq u$$
$$a_{i,j} = \langle a_{N-j} \rangle \bmod m_i \quad \text{if } a_{N-j} \neq u \text{ and } a_{i,j-1} = u$$
$$a_{i,j} = u \quad \text{if } a_{N-j} = u \text{ and } a_{i,j-1} = u$$

Digits of the number X represented in the completely specified RNS are:

$$x_i = a_{i,N-1} \quad \text{for all } i = 1,\ldots,N .$$

Obviously, the target RNS can be equal to the starting RNS. If a number is uniquely defined in the ISRNS then the described algorithm presents a way for specifying its unspecified digits regardless of their number and combination.

Example 1:

In the RNS with moduli 3,5,7,11,13 the incompletely specified positive number X = (u,2,u,6,u) = 17 is given and it should be transformed to the completely specified one. The application of the described algorithm is shown in figure 1 with partial results after each step.

4. BASE EXTENSION OF INCOMPLETELY SPECIFIED RNS NUMBERS

If a positive ISRNS number should be transformed to the completely specified number in an auxiliary RNS defined by moduli $p_1,p_2,\ldots,p_R$, $P = \prod_{j=1}^{N} p_j$ where M and P are relatively prime numbers, then the algorithm presented in section 3 could be used again. But there is another way in which conversion could be done in only N+1 steps of modulo operations.

Suppose that we have already transformed a number X from the ISRNS to the associated ISMRS and that instead of X we want to compute the value $X \cdot M_u$ in the completely specified auxiliary RNS:

$$X \cdot M_u = (\delta_1 \cdot a_1 + \sum_{i=2}^{N} \delta_i \cdot a_i \cdot Y_i) \cdot (\prod_{j=1}^{N} (m_j)^{\overline{\delta}_j}) \quad (9)$$

$$X \cdot M_u = \delta_1 \cdot a_1 \prod_{j=1}^{N} (m_j)^{\overline{\delta}_j} + \sum_{i=2}^{N} \delta_i \cdot a_i \cdot W_i$$

$$W_i = T_i \prod_{j=i}^{N} (m_j)^{\overline{\delta}_j} .$$

Each digit of the number $X \cdot M_u$ in the auxiliary can be computed by

$$\langle X \cdot M_u \rangle \bmod p_i = \langle\langle\langle \delta_1 \cdot a_1 \cdot (m_2)^{\overline{\delta}_2} + \quad (10)$$
$$+ \delta_2 \cdot a_2 \cdot T_2 \rangle \bmod p_i \cdot (m_3)^{\overline{\delta}_3} +$$
$$+ \delta_3 \cdot a_3 \cdot T_3 \rangle \bmod p_i \cdot (m_4)^{\overline{\delta}_4} + \ldots$$
$$+ \delta_N \cdot a_N \cdot T_N \rangle \bmod p_i ,$$

and from this relation we see that computation starts again with the digit a_i as in the original base extension algorithm and that additional inputs of the look-up tables are not required because in-

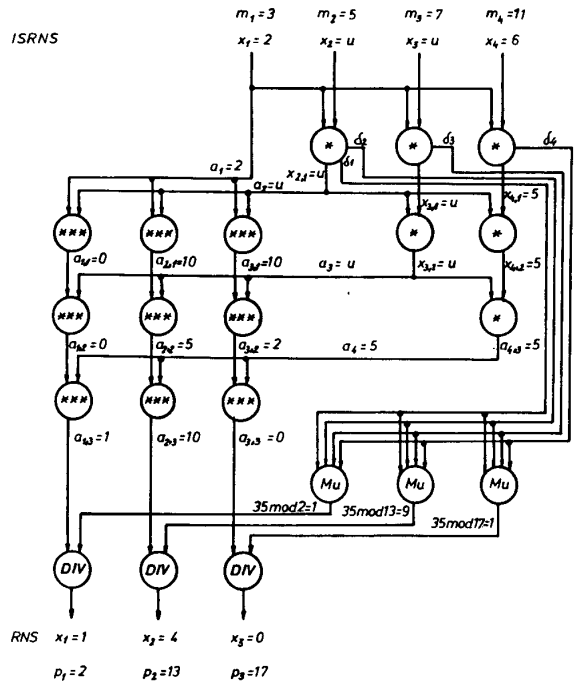

Fig. 2. Base extension of incompletely specified numbers

formation about δ_i is necessary only in the step when computation with a_i is done. It means that the described algorithm is of the same hardware complexity and executable in only N steps as the original base extension algorithm. The only difference is that result in the auxiliary RNS is multiplied by M_u . For completely specified RNS $M_u=1$ and the described algorithm is a real generalization of the known base extension algorithm.

Value of the number X in the auxiliary RNS can be obtained by division of the transformation result $X \cdot M_u$ by M_u . This division is executable in only one modular step because it is division with zero remainder and because M_u is always relatively prime with P. The number M_u represented in the auxiliary RNS can be obtained from look-up tables in which all possible values of M_u are stored. Address inputs of the look-up tables are δ_i lines generated as the additional outputs of the first step of the RNS to MRS transformation. This algorithm is shown in the left part of figure 2 with look-up tables marked by ***. Functions realized by look-up tables for moduli p_i (i=1,...R) are:

for the first step

$a_{i,1} = \langle a_1 + a_2 \cdot T_2 \rangle \bmod p_i$ if $a_1 \neq u$ and $a_2 \neq u$

$a_{i,1} = \langle a_2 \cdot T_2 \rangle \bmod p_i$ if $a_1 = u$ and $a_2 \neq u$

$a_{i,1} = \langle a_1 \cdot m_2 \rangle \bmod p_i$ if $a_1 \neq u$ and $a_2 = u$

$a_{i,1} = u$ if $a_1 = u$ and $a_2 = u$

j-th modular step (j = 2,...,N-1)

$a_{i,j} = \langle a_{i,j-1} + a_j \cdot T_j \rangle \bmod p_i$ if $a_{i,j-1} \neq u$ and $a_j \neq u$

$a_{i,j} = \langle a_j \cdot T_j \bmod p_i \rangle$ if $a_{i,j-1} = u$ and $a_j \neq u$

$a_{i,j} = \langle a_{i,j-1} \cdot m_j \rangle \bmod p_i$ if $a_{i,j-1} = u$ and $a_j = u$

$a_{i,j} = u$ if $a_{i,j-1} = u$ and $a_j = u$.

Digits of the number X represented in a completely specified RNS are:

$$x_i = a_{i,N-1} \quad \text{for all} \quad i = 1,\ldots,R .$$

Example 2:

The incompletely specified number X = (0,u,u,4) = = 15 from the RNS with the moduli 3,5,7,11 should be transformed to the completely specified RNS number in the auxiliary RNS with the moduli 2,13,17. The application of the described algorithm is shown in figure 2 with partial results after each step.

5. APPLICATIONS

There are some obvious applications of the ISRNS in division with zero remainder and in some error detecting and correcting algorithms. In this section its applications for divisibility testing and general integer computations will be shown.

5.1. Divisibility testing

The task of the algorithm is to test if a positive number X is divisible by a positive number Y without remainder, where X and Y are completely specified numbers in the main RNS. The testing will be done in the auxiliary RNS which satisfy conditions: P > M and P is relatively prime to M .

By employing the base extension algorithm, the numbers X and Y can be represented in the auxiliary RNS. For every modulus m_i of the main and the auxiliary RNS, one g_i line is generated indicating that division in this modulus is possible ($g_i = 1$). Namely, in the case of $y_i = 0$ and $x_i \neq 0$, one can immediately conclude that $Y \nmid X$, what is indicated by setting $g_i = 0$.

If all g_i lines are equal 1 then direct divisions X by Y in the main and in the auxiliary RNS are done. The results of the division in the main and in the auxiliary RNS, Z_M and Z_P respectively, are generally incompletely specified numbers defined by

$$\begin{aligned} Z_M \cdot Y &= X + k_M \cdot M \\ Z_P \cdot Y &= X + k_P \cdot P \end{aligned} \qquad (11)$$

where k_M and k_P are smallest integers satisfying inequality

$$0 \leqslant k_M , k_P < Y . \qquad (12)$$

From (11) , (12) and the condition $Y < M < P$ we can conclude that $k_M = k_P = 0$ and $Z_M = Z_P$ follows from $Y \mid X$, as well as $k_M = k_P = 0$ and $Y \mid X$ from $Z_M = Z_P$. It means that Y divides X without remainder if and only if the result of direct division in the main RNS equals the result of direct division in the auxiliary RNS.

Figure 3 shows the described algorithm for the divisibility testing executable in N+3 steps of modulo operations. With the use of the standard base extension algorithm both X and Y are simultaneously converted to the auxiliary RNS (N steps) and then divided (1 step). The result is an incompletely specified number Z_P, represented in the auxiliary RNS, and g_i lines (i=1,...,R). In

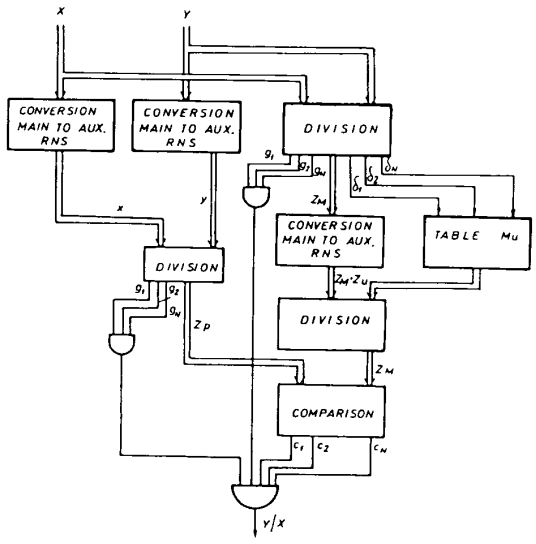

Fig. 3. Divisibility testing

parallel to this operations, X is divided by Y in the main RNS (1 step) and the result is an incompletely specified number Z_M, represented in the main RNS, and the corresponding g_i lines ($i=1,\ldots,N$). The number Z_M can be represented in the completely specified auxiliary RNS by the use of the algorithm described in part 4 (N+1 steps). In the last step values for Z_M and Z_p are compared. The result of the comparison are c_i lines ($i=1,\ldots,R$) which are set ($c_i = 1$) if $(z_M)_i = (z_p)_i$ or if $(z_p)_i = u$. If all c_i and g_i lines are equal 1 then the divisibility of X by Y is signalled.

Example 3:

Given the numbers X = (0,6,2) = 90 and Y = (0,1,4) = 15 in the RNS with moduli 5, 7 and 11, test if Y divides X without remainder using the auxiliary RNS defined by moduli 3, 13 and 17.

The result of direct division in the main RNS is $Z_M = X/Y = (0,6,2)/(0,1,4) = (u,6,6)$ and its representation in the auxiliary RNS is $Z_M = (0,6,6)$. Simultaneously, X and Y are converted into the auxiliary RNS and then direct division in that system is performed

$$Z_p = X/Y = (0,12,5)/(0,2,15) = (u,6,6) .$$

Because all g lines in both direct divisions equal 1 and because Z_p equals Z_M in all the moduli in which Z_p is specified, it follows that Y divides X without remainder.

5.2. General integer computations

The property of distinguishing RNS integer arithmetic from weighted integer arithmetic is the possibility to compute the expressions like $X=X_1+X_2+X_3-X_4-X_5$ correctly if $X \leqslant \lfloor (M-1)/2 \rfloor$, regardless of the sequence of additions and subtractions and regardless of the magnitude of intermediate results. Similarly, it also holds for multiplication and division or any combination of all elementary operations if final result is integer and if division is treated as direct division as defined in section 2. If a series of operations includes direct division, as for example in expression $X = \frac{X_1 \cdot X_2 \cdot X_3}{X_4 \cdot X_5}$ the result will be correct, regardless of the sequence of operations, if X is integer satisfying the inequality

$$X \leqslant \left\lfloor \left(\frac{M}{GCD(X_4 \cdot X_5, M)} - 1 \right)/2 \right\rfloor , \qquad \text{where GCD(a,b)}$$

denotes greatest common divisor of a and b. The result will be an incompletely specified number if $GCD(X_4 \cdot X_5, M) > 1$ and then the described algorithms must be used to transform it into the completely specified number.

It means that RNS integer arithmetic allows unbounded magnitudes of the intermediate results that can even be rational numbers if it is known in advance that the results are integers uniquely defined in the RNS. This property could be of great importance in the various integer computations. The result correctness is guaranteed and fast computations are insured due to the substitution of integer division by direct division. One example of such computations is integer matrix inversion process, in which all elements of the resultant matrix are integers, when multiplied by the determinant D of the starting matrix. Integer matrix inversion process in the ISRNS, using Gaussian method, is shown in example 4.

The algorithm for the transformation of the results to the completely specified RNS presented in part 3 should be used in this application. But the results can be also negative numbers and in the case of implicit sign representation, according to [3], value of M/M_u should be subtracted from the negative results after the transformation. Additional sign determination hardware and look-up tables of all possible values of M/M_u are necessary for this operation.

One of the problems occurring in all RNS applications is the selection of the proper system which will insure the uniqueness of the results. This problem is especially difficult in the case of integer matrix inversion because introduction of digits of unspecified value can significantly reduce the range of the uniquely defined results. The solution to this problem is out of scope of this paper.

Example 4:

Given the matrix $A = \begin{bmatrix} 3 & 3 & 4 \\ 5 & 6 & 7 \\ 6 & 5 & 4 \end{bmatrix}$ in the RNS with moduli 3,5,7,11 and 13 compute the inverse matrix A^{-1}. First, the computation in the rational decade arithmetic will be illustrated and then in the RNS arithmetic.

Start D = 1

$$\begin{bmatrix} 3 & 3 & 4 \\ 5 & 6 & 7 \\ 6 & 5 & 4 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1. step D = 3

$$\begin{bmatrix} 1 & 1 & 4/3 \\ 0 & 1 & 1/3 \\ 0 & -1 & -4 \end{bmatrix} \Rightarrow \begin{bmatrix} 1/3 & 0 & 0 \\ -5/3 & 1 & 0 \\ -2 & 0 & 1 \end{bmatrix}$$

2. step D = 3

$$\begin{bmatrix} 1 & 1 & 4/3 \\ 0 & 1 & 1/3 \\ 0 & 0 & -11/3 \end{bmatrix} \Rightarrow \begin{bmatrix} 1/3 & 0 & 0 \\ -5/3 & 1 & 0 \\ -11/3 & 1 & 1 \end{bmatrix}$$

3. step D = -11

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 & 4/11 & 4/11 \\ -2 & 12/11 & 1/11 \\ 1 & -3/11 & -3/11 \end{bmatrix}$$

4. step D = -11

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & -8/11 & 3/11 \\ -2 & 12/11 & 1/11 \\ 1 & -3/11 & -3/11 \end{bmatrix} = A^{-1}$$

Start D = (1,1,1,1,1)

$$\begin{bmatrix} (0,3,3,3,3) & (0,3,3,3,3) & (1,4,4,4,4) \\ (2,0,5,5,5) & (0,1,6,6,6) & (1,2,0,7,7) \\ (0,1,6,6,6) & (2,0,5,5,5) & (1,4,4,4,4) \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} (1,1,1,1,1) & (0,0,0,0,0) & (0,0,0,0,0) \\ (0,0,0,0,0) & (1,1,1,1,1) & (0,0,0,0,0) \\ (0,0,0,0,0) & (0,0,0,0,0) & (1,1,1,1,1) \end{bmatrix}$$

1. step D = (0,3,3,3,3)

$$\begin{bmatrix} (u,1,1,1,1) & (u,1,1,1,1) & (u,3,6,5,10) \\ (u,0,0,0,0) & (u,1,1,1,1) & (u,2,5,4,9) \\ (u,0,0,0,0) & (u,4,6,10,12) & (u,1,3,7,9) \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} (u,2,5,4,9) & (u,0,0,0,0) & (u,0,0,0,0) \\ (u,0,3,2,7) & (u,1,1,1,1) & (u,0,0,0,0) \\ (u,3,5,9,11) & (u,0,0,0,0) & (u,1,1,1,1) \end{bmatrix}$$

2. step D = (u,3,3,3,3)

$$\begin{bmatrix} (u,1,1,1,1) & (u,1,1,1,1) & (u,3,6,5,10) \\ (u,0,0,0,0) & (u,1,1,1,1) & (u,2,5,4,9) \\ (u,0,0,0,0) & (u,0,0,0,0) & (u,3,1,0,5) \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} (u,2,5,4,9) & (u,0,0,0,0) & (u,0,0,0,0) \\ (u,0,3,2,7) & (u,1,1,1,1) & (u,0,0,0,0) \\ (u,3,1,0,5) & (u,1,1,1,1) & (u,1,1,1,1) \end{bmatrix}$$

3. step D = (u,4,3,0,2)

$$\begin{bmatrix} (u,1,1,u,1) & (u,1,1,u,1) & (u,0,0,u,0) \\ (u,0,0,u,0) & (u,1,1,u,1) & (u,0,0,u,0) \\ (u,0,0,u,0) & (u,0,0,u,0) & (u,1,1,u,1) \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} (u,4,6,u,12) & (u,4,1,u,11) & (u,4,1,u,11) \\ (u,3,5,u,11) & (u,2,3,u,7) & (u,1,2,u,6) \\ (u,1,1,u,1) & (u,2,1,u,8) & (u,2,1,u,8) \end{bmatrix}$$

4. step D = (u,4,3,0,2)

$$\begin{bmatrix} (u,1,1,u,1) & (u,0,0,u,0) & (u,0,0,u,0) \\ (u,0,0,u,0) & (u,1,1,u,1) & (u,0,0,u,0) \\ (u,0,0,u,0) & (u,0,0,u,0) & (u,1,1,u,1) \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} (u,1,1,u,1) & (u,2,5,u,4) & (u,3,6,u,5) \\ (u,3,5,u,11) & (u,2,3,u,7) & (u,1,2,u,6) \\ (u,1,1,u,1) & (u,2,1,u,8) & (u,2,1,u,8) \end{bmatrix}$$

After multiplication A-1 by D and transformation to the completely specified RNS, the obtained result is:

D = (1,4,3,0,2)

$$A^{-1}.D = \begin{bmatrix} (1,4,3,0,2) & (2,3,1,8,8) & (0,2,4,8,10) \\ (1,2,1,0,9) & (0,3,2,10,1) & (2,4,6,10,12) \\ (1,4,3,0,2) & (0,3,3,3,3) & (0,3,3,3,3) \end{bmatrix}$$

6. CONCLUSION

It is shown that introduction of ISRNS might be useful in some RNS applications and that it is possible to realize efficient algorithms for transformation of incompletely to completely specified numbers. Accordingly, the ISRNS should not be treated as the unusual modification of the RNS but as the general RNS in which completely specified numbers are the special case.

Although in this paper it is supposed that all moduli are prime numbers, the presented algorithms can also be used, without change, if the moduli are not prime numbers. But in that case there is the possibility that a residue digit is only partially unspecified and the introduction of more than one unspecified state per modulus might be necessary.

REFERENCES:

[1]. N.S.Szabo,R.I.Tanaka: Residue Arithmetic and Its Applications to Computer Technology, McGraw-Hill, New York, 1967.

[2]. G.A.Jullien: Residue Number Scaling and Other Operations Using ROM Arrays, IEEE Trans. Comp., Vol. C-27, April 1978, pp. 325-336.

[3]. W.K.Jenkins, B.J.Leon: The Use of Residue Number System in the Design of Finite Impulse Response Digital Filters, IEEE Trans. Circuits and Systems, Vol. CAS-24, April 1977, pp. 191-200.

[4]. W.K.Jenkins: The Design of Error Checkers for Self-checking Residue Number Arithmetic, IEEE Trans. Comp., Vol. C-32, April 1983, pp. 388-396.

[5]. F.J.Taylor: Residue Arithmetic: A Tutorial with Examples, IEEE Computer, Vol. 17, May 1984, pp. 50-62.

[6]. N.B.Chakraborti,J.S.Soudararajan,A.N.Reddy: An Implementation of Mixed-radix Conversion for Residue Number Applications, IEEE Trans. Comp., Vol. C-35, August 1986, pp. 762-764.