GSJ: Volume 9, Issue 2, February 2021, Online: ISSN 2320-9186
www.globalscientificjournal.com

# New Algorithm for Reverse Conversion in Residue Number System

Daniel Asiedu[1]
Department of Computer Science
C. K. T. University of Technology and Applied Sciences
Navrongo, Ghana
asiedudaniellll@gmail.com

Abdul-Mumin Salifu[2]
Department of Computer Science
C. K. T. University of Technology and Applied Sciences
Navrongo, Ghana, asalifu@cktutas.edu.gh

**BSTRACT**

Reverse conversion is an important exercise in achieving the properties of Residue Number System (RNS). Current algorithms available for reverse conversion exhibits greater computational overhead in terms of speed and area. In this paper, we have developed a new algorithm for reverse conversion for two-moduli set and three-moduli set that are very simple and with fewer multiplicative inverse operations than there are in the traditional algorithms like the Chinese Remainder Theorem (CRT) and Mixed Radix Conversion (MRC).

**Keywords: Residue Number System, Forward Conversion, Reverse Conversion, CRT, MRC**

## INTRODUCTION

Residue Number System can be traced back to Sun Tzu, which is an old number system. Large binary and decimal numbers are represented in RNS uniquely using a set of smaller residues, that results in carry-free, high speed arithmetic operations. Parallelism is also ensured in RNS. [1][2][3]. Numbers are represented in this system by taking modulus operation where the divider is known as the modulo and the remainder is the residue which represents the number in RNS.

The main advantage of RNS over the conventional number systems is that the standard arithmetic operations can be easily implemented due to the carry propagation inherent in the conventional number system. This ensures the design of high speed digital processors. RNS is very helpful in applications requiring many additions and for that matter many multiplication (multiplication is a repeated form of addition) but with no or fewer number of division and comparisons. The main problems of RNS representation are that division, magnitude comparison, scaling, overflow detection and sign detection are difficult to implement. RNS is suitable in applications such as cryptography, digital signal processing, image processing, speech processing etc [1]-[4].

### Background

The RNS is defined in terms of a set of relatively prime moduli, $P = \{m_1, m_2, \ldots \ldots . m_n\}$, where $GCD\ (m_i, m_j) = 1$ for $i \neq j$. $M = \prod_{i=1}^{n} m_i$ is the dynamic range. Any integer $X$ in the range [0, M) called the legitimate range can be unambiguously represented [5],[6].

To perform an arithmetic operation in the residue number representation to achieve the properties of RNS, raises the need to be able to convert from the conventional representation to RNS and vice versa. The conversion from the conventional numbers to RNS is known as forward conversion and process is very simple and direct operation; divide the given number by each modulus in the moduli set and taking string of the remainders constitute the RNS representation. The conversion form RNS to the conversional number system is referred to as reverse/backward conversion. This operation is very difficult to accomplish and it introduces grave overhead in terms of speed and complexity. The process of reverse/backward conversions are shrouded on the Chinese Remainder Theorem (CRT) or the Mixed-Radix Conversion (MRC) [7]

**Chinese Remainder Theorem (CRT)**

Given a set of pair-wise relatively-prime moduli, $m_1, m_2, \dots\dots. m_N$, and a residue representation $(x_1, x_2, \dots\dots, x_N)$ in that system of some number $X$, i.e. $x_i = |X|_{m_i}$, that number and its residues are related by the Chinese Remainder Theorem that is given as;

$$|X|_M = |\sum_{i=1}^{N} x_i |M_i^{-1}|_{m_i} M_i|_M \dots\dots\dots\dots.. (1)$$

Where $M$ is the product of the $m_i$ $M_i^{-1}$ are the multiplicative inverse of $M_i$ with respect to $m_i$ $M_i = \frac{M}{m_i}$ [8], [9]. In other words, given the moduli set, and a number, $X$ represented in its residue form, $X$ can be computed by equation 1. The main problem of the CRT is the modulo operation of $M$ which introduces more overhead to the reverse conversion process in terms of speed and complexity.

**Mixed Radix Conversion (MRC)**

The Mixed Radix Conversion (MRC) is an algorithm use to convert any number $X$ in RNS representation to its binary/decimal equivalent. MRC is given as follows;

$$X = d_1 + d_2 m_1 + d_3 m_1 m_2 + \cdots + d_n m_1 m_2 m_3 \dots m_{n-1} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots. (2)$$

Where $d_i, i = 1,2, \dots, n$ are the Mixed Radix Digits (MRDs) and computed as follows:

$$d_1 = x_1$$

$$d_2 = \left|(x_2 - d_1)|m_1^{-1}|_{m_2}\right|_{m_2}$$

$$d_3 = \left|\left((x_3 - d_1)|m_1^{-1}|_{m_3} - d_2\right)|m_2^{-1}|_{m_3}\right|_{m_3}$$

$$\vdots$$

$$d_n = \left|\left(\dots\left((x_3 - d_1)|m_1^{-1}|_{m_n} - d_2\right)|m_2^{-1}|_{m_n} - \dots - d_{n-1}\right)|m_{n-1}^{-1}|_{m_n}\right|_{m_n} \dots\dots\dots\dots\dots\dots(2.3)\ [10].$$

That is, $X$ in the interval $[0, M]$ can be uniquely represented. The MRC is serial and an error in the computation of $d_1$ will lead to an error in the subsequent $d_i$.

**The New Algorithms**

The proposed new algorithm is presented below is very simple and have fewer multiplicative inverse operations than there are in the CRT and MRC.

**ALGORITHM FOR TWO MODULI SET:**

Given a two moduli set $m = \{m_1, m_2\}$ and residues $r = (r_1, r_2)$, $m$ and $r$ can be written in a form:

$$X \equiv r_1 \ mod \ m_1 \dots\dots\dots\dots\dots\dots 1$$

$$X \equiv r_2 \ mod \ m_2 \dots\dots\dots\dots\dots\dots 2$$

Equation 1 can also be written as

$$X = m_1 p + r_1 \ \dots\dots\dots\dots\dots\dots 3$$

Now equation 3 is the general form of decimal equivalent that satisfies equation 2 such that:

$$|m_1 p + r_1|_{m_2} = r_2$$

Where p = [0,$r_2$].

Generally, the decimal equivalent for any two moduli set can be computed as follows:

X =
$$|m_1 p + r_1|_{m_2} = r_2$$

Where,

p = [0,$r_2$] that is, any value from the range that satisfies X.

Therefore the value that satisfies X at a chosen

**NUMERICAL ILLUSTRATION WITH TWO MODULI SET {$2^n + 1, 2^n$}**

Example 1: Given the moduli set m = {5, 4} and residues r= (2, 3)

Example 2: Given the moduli set m = {9, 8} and residues r= (7, 7)

Solution:

With two moduli set, we have

Generally, the decimal equivalent for any two moduli set can be computed as follows:

X =
$$|m_1 p + r_1|_{m_2} = r_2$$

Where,

p = [0,$r_2$] that is, any value from the range that satisfies X.

Therefore the value that satisfies X at a chosen number from p, is its decimal equivalent.

From example 1, $m_1 = 5, m_2 = 4, r_1 = 2, r_2 = 3$

X =
$$|5p + 2|_4 = 3$$

p= [0, 1,…3]

When p=1

$|5(1) + 2|_4 = 3$ is satisfied. Since $|7|_4 = 3$ satisfied X at the value of 7 $(5(1) + 2)$, we stop and take that value as its decimal equivalent.

**Therefore the decimal equivalent for example 1 is 7.**

From example 2, $m_1 = 9, m_2 = 8, r_1 = 7, r_2 = 7$

X =
$$|9p + 7|_8 = 7$$

p=[0, 1,…7]

When p=0

$|9(0) + 7|_8 = 7$ is satisfied. Since $|7|_8 = 7$ satisfied X at the value 7 $(9(0) + 7)$, we stop and take that value as is decimal equivalent.

**Therefore the decimal equivalent for example 2 is 7.**

**ALGORITHM FOR THREE MODULI SET:**

Given a moduli set $m = \{m_1, m_2, m_3\}$ and residues $r = (r_1, r_2, r_3)$, $m$ and $r$ can be written in a form:

$$X \equiv r_1 \ mod \ m_1 \dots\dots\dots\dots\dots\dots 1$$

$$X \equiv r_2 \ mod \ m_2 \dots\dots\dots\dots\dots\dots 2$$

$$X \equiv r_3 \ mod \ m_3 \dots\dots\dots\dots\dots\dots 3$$

Equation 1 can also be written as

$$X = m_1 k + r_1 \ \dots\dots\dots\dots\dots\dots 4$$

Equation 4 must satisfy equation 2 such that

$$m_1 k + r_1 \equiv r_2 mod \ m_2$$

$$m_1 k \equiv (r_2 - r_1) mod \ m_2$$

$$k \equiv (r_2 - r_1).m_1^{-1} \ mod \ m_2$$

$$k = m_2 t + |(r_2 - r_1).m_1^{-1}|_{m_2}$$

Putting k into equation 4, we have

$$X = m_1(m_2 t + |(r_2 - r_1)m_1^{-1}|_{m_2}) + r_1$$

$$X = m_1 m_2 t + \left( m_1 |(r_2 - r_1)m_1^{-1}|_{m_2} + r_1 \right) \dots\dots\dots\dots\dots\dots 5$$

Now equation 5 is the general form of decimal equivalent that satisfies equation 3 such that:

$$\left| m_1 m_2 t + \left( m_1 |(r_2 - r_1)m_1^{-1}|_{m_2} + r_1 \right) \right|_{m_3} = r_3$$

Where t = [0,$r_3$]

Generally, the decimal equivalent for any three moduli set can be computed as follows:

X =
$$|m_1 m_2 p + (s)|_{m_3} = r_3$$

Where,

s = $\left( m_1 |(r_2 - r_1) m_1^{-1}|_{m_2} + r_1 \right)$

p = [0,$r_3$] that is, any value from the range that satisfies X after computing the value of *s*.

Therefore the value that satisfies X at a chosen number from p, is its decimal equivalent.

**NUMERICAL ILLUSTRATION WITH THREE MODULI SET {$2^n + 1$, $2^n$, $2^n - 1$}**

Example 3: Given the moduli set m = {5, 4, 3} and residues r= (0, 3, 0)

Example 4: Given the moduli set m = {9, 8, 7} and residues r= (7, 0, 2)

Solution:

With three moduli set, we have

Generally, the decimal equivalent for any three moduli set can be computed as follows:

X =
$$|m_1 m_2 p + (s)|_{m_3} = r_3$$

Where,

s = $\left( m_1 |(r_2 - r_1) m_1^{-1}|_{m_2} + r_1 \right)$

p = [0,$r_3$] that is, any value from the range that satisfies X after computing the value of *s*.

Therefore the value that satisfies X at a chosen number from p, is its decimal equivalent.

From example 3, $m_1 = 5, m_2 = 4, m_3 = 3, r_1 = 0, r_2 = 3, r_3 = 0$

X =
$$|5.4.p + (s)|_3 = 0$$

s = $( 5|(3 - 0)5^{-1}|_4 + 0)$

s = $( 5|(3).1|_4 + 0)$

s = 15

X =
$$|5.4.p + (15)|_3 = 0$$

X =
$$|20p + 15|_3 = 0$$

p= [0, 1, 2]

When p=0

$|20(0) + 15|_3 = 0$ is satisfied. Since $|15|_3 = 0$ satisfied X at the value of 15 $(20(0) + 15)$, we stop and take that value as its decimal equivalent.

**Therefore the decimal equivalent for example 3 is 15.**

From example 4, $m_1 = 9, m_2 = 8, m_3 = 7, r_1 = 7, r_2 = 0, r_3 = 2$

X =
$$|9.8.p + (s)|_7 = 2$$

s = $( 9|(0 - 7)9^{-1}|_8 + 7)$

s = $( 9|(-7).1|_4 + 7)$

s = 16

X =
$$|9.8.p + (16)|_7 = 2$$

X =
$$|72p + 16|_7 = 2$$

p= [0, 1,2,…6]

When p=0

$|72(0) + 16|_7 = 2$ is satisfied. Since $|16|_7 = 2$ satisfied X at the value of 16 $(72(0) + 16)$, we stop and take that value as its decimal equivalent.

**Therefore the decimal equivalent for example 4 is 16.**

## Conclusion

New algorithm for reverse conversion in RNS for two moduli set and three moduli set have been proposed. These algorithms are very simple and straight forward with fewer number multiplicative inverses operations than there are in the traditional CRT and MRC algorithms.

## References

[1]R.I.Tanaka and N.S.Szabo,"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] Y. Ayyavaru Reddy1 , B. Sekhar2, "An Efficient Reverse Converter Design for Five Moduli Set RNS". International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 8, August 2016, pp: 208-212

[4] S.Palnitkar, "Verilog HDL- A Guide to Digital Design and Synthesis," Prentice Hall PTR , ISBN: 0-13-044911-3.

[5] Amir Sabbagh Molahosseini and Keivan Nav: New Arithmetic Residue to Binary Converters, International Journal of Computer Sciences and Engineering Systems, Vol. 1, No. 4, pp. 295-299, October, 2007.

[6] Salifu Abdul-Mumin, Kazeem Alabge Gbolagade. Rivest Shamir Adleman Encryption Scheme Based on the Chinese Remainder Theorem. Advances in Networks. Vol. 6, No. 1, 2018, pp. 40-47. doi: 10.11648/j.net.20180601.14

[7] Hector Pettenghi, Ricardo Chaves, and Leonel Sousa: IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, VOL. 60, NO. 6, JUNE 2013 pp: 1487- 1500

[8] Duc-Minh P., Premkumar A. B., and Madhukumar A. S. (2011). Error Detection and Correction in Communication Channels Using Inverse Gray RSNS Codes, IEEE Transactions on Communications- TCOM, VOL: 59 no. 4 pp: 975-986

[9] Omondi A. and Premkumar B. (2007)*:* Residue Number Systems: Theory and Implementation, *Published by* Imperial College Press 57 Shelton Street Covent Garden London WC2H 9HE.

[10] Chung-Kuan C. (2006): Computer Arithmetic Algorithms and Hardware Design, Lecture notes, University of California, San Diego, La Jolla, CA