RNS Overflow Detection by Operands Examination

H. Siewobr and K.A.Gbolagade
Department of Computer Science,
University for Development Studies,
Navrongo, Ghana

ABSTRACT
In this paper, a novel scheme for detecting overflow in Residue Number System (RNS) is presented. A generalized scheme for RNS overflow detection is introduced, followed by a simplified Operands Examination Method for overflow detection for the moduli set $\{2^n-1, 2^n, 2^n+1\}$. The proposed method detects overflow in RNS addition of two numbers without pre-computing their sum .Moreover, when compared with the best known similar state of the art designs, the proposed scheme requires lesser hardware, s the reduce operation sizeand is faster.

General Terms
Residue Number System, Circuits and Systems, Computer Arithmetic, Computer Architecture, Digital Signal Processing.

Keywords
Residue Number System, Overflow Detection, Reverse Converter, Chinese Remainder Theorem, Mixed Radix Conversion.

1. INTRODUCTION
There is performance degradation in computing hardware built based on Weighted Number System (WNS) due to carry propagation phenomenon inherent to WNS [3]. The reduction/elimination of carry chains is the major challenge in improving computer arithmetic performance. Several approaches have been proposed, e.g., carry look ahead, prefix calculations, anticipated calculations, and alternative number representation systems, e.g., (redundant) signed digit systems, or Residue Number Systems (RNS) [3].

RNS is an integer number system which supports parallel, carry-free addition, borrow-free subtraction and single step multiplication without partial product. These features enable RNS utilization in Digital Signal Processing applications, such as digital filtering, convolution, fast Fourier transform and image processing [4]. However, for successful application of RNS, overflow detection must be easy and fast in order not to prevent RNS usage in general purpose computing. Unlike WNS, where overflow can be efficiently handled by rounding, truncating, or saturating arithmetic, overflow detection in RNS involves more complex and time consuming procedures [2]. Overflow is a condition where a number which falls outside the legitimate range of a particular RNS i.e.,$[0, M-1]$, $(M = \prod_{i=1}^{n} m_i)$ is well represented as a legitimate RNS number. For example, the sum of decimal numbers 50 and 15 is 65 .Performing this addition in RNS using the moduli set $\{3,4,5\}$ generates $(2,1,0)_{RNS\langle 3|4|5\rangle}$ as result. $(2,1,0)_{RNS\langle 3|4|5\rangle}$ is the equivalent of decimal number 5.

This is because the sum of 50 and 15 which is 65 falls outside the legitimate range $[0,59]$and hence there is an overflow in the sum. The traditional overflow detection technique utilizes either the Chinese Remainder Theorem (CRT) [9] or the Mixed Radix Conversion (MRC) [1, 4] techniques.

Although many researchers have made considerable efforts to design overflow detection schemes which do not require full reverse conversion, recently proposed RNS overflow detection algorithms still rely on this [2] and other costly and time consuming procedures such as base extension, use of Redundant RNS, group number and sign detections as in [5], [6],[7], [8] and [9].

In this paper, a generalized technique for RNS overflow detection and a simplified technique for the moduli set $\{2^n-1, 2^n, 2^n+1\}$ is proposed. Our proposal completely eliminates all the area and delay intensive computations which characterize best known similar state of the art designs.

The rest of the paper is organized as follows: in Section 2 the necessary background information is presented. Section 3 proposes the Operand Examination Method for RNS overflow detection, followed by the scheme for moduli set $\{2^n-1, 2^n, 2^n+1\}$. In Section 4 the hardware implementation of of the new scheme is explained, while Section 5 proves the importance and efficiency of the proposed scheme by comparing it with current state of the art overflow detection algorithms. Finally, the paper is concluded in Section 6 and references provided in Section 7.

2. BACKGROUND
The MRC can be represented as [3]:

$$X = a_1 + a_2 m_1 + a_3 m_1 m_2 + \cdots + a_n m_1 m_2 m_3 \ldots m_{n-1} \quad)1($$

Where the Mixed Radix Digits (MRDs), $a_i, i = 1, n$ can be computed as follows [1,4]:

$$a_1 = x_1$$

$$a_2 = |(x_2 - a_1)|m_1^{-1}|m_2|_{m_2}$$

$$a_3 = \left|((x_3 - a_1)|m_1^{-1}|_{m_3} - a_2)|m_2^{-1}|_{m_3}\right|_{m_3}$$

$$a_n = \left|((\ldots(x_n - a_1)|m_1^{-1}|_{m_n} - a_2)|m_2^{-1}|_{m_3} - \cdots - a_{n-1})|m_{n-1}^{-1}|_{m_n}\right|_{m_n} \quad (2)$$

Given the MRD $a_i, 0 \le a_i < m_i$, any positive number in the interval $[0, \prod_{i=1}^{n} m_i - 1]$ can be uniquely represented.

One method of obtaining overflow information is MRC [7] .A number X may be expressed in MRC form according to the following equations [7]:

$$X = \sum_{i=1}^{n} a_i w_i \,, \quad (3)$$

where,

$$w_i = \prod_{i=1}^{n-1} m_i, i \ge 2 \quad (4)$$

$w_1 = 1$, for $i = 1$ so that) 1 (holds true .

The terms $w_1, w_2, \dots, w_n$are the bases and are related to the moduli of an associated RNS representation [7] as shown in (4).

MRC based overflow detection schemes have been proposed in [7] and [8]. These schemes employ area and delay intensive computations such as base extension, the use of RRNS, sign detection and a number of reverse conversions.

In the following section, a generalized MRC based overflow detection scheme which eliminates all the expensive computations in [5], [6], [7], [8] and [9] is presented. The resulting overflow scheme uses smaller modulo operations and outperforms best known similar state of the art overflow detection schemes.

3. PROPOSED ALGORITHM

This section presents a new RNS overflow detection scheme called the Operand Examination Method (OEM): This scheme detects overflow by examining the operands only. It eliminates the computation of the modulo M sum of the operands, which is usually required by other known overflow detection schemes.

Property 1: For a particular RNS, (1) represents only numbers within the legitimate range, i.e. $[0, M-1]$.

Assume we have two RNS numbers X and Y represented as $(x_1, x_2, \dots, x_n)$ and $(y_1, y_2, \dots, y_n)$respectively . Let $(\omega_1, \omega_2, \dots, \omega_n)$, $(\varphi_1, \varphi_2, \dots, \varphi_n)$, and $(\rho_1, \rho_2, \rho_3, \dots, \rho_n)$ be the MRDs for X, Y, and Z respectively, where $Z = X + Y$ and $\rho_i = \omega_i + \varphi_i, i = 1, n$ so as to detect overflow.

Theorem 1: Overflow occurs in the sum of X and Y if the following hold true;

$$\rho_n > m_n - 1$$

$$\rho_n = m_n - 1 \text{AND } \rho_{n-1} > m_{n-1} - 1$$

$$\rho_n = m_n - 1 \text{AND } \rho_{n-1} = m_{n-1} - 1 \text{ AND } \rho_{n-2} > m_{n-2} - 1$$

.

.

.

$$\rho_n = m_n - 1 \text{AND } \rho_{n-1} = m_{n-1} - 1 \text{ AND } \rho_{n-2} = m_{n-2} - 1 \text{ AND…AND } \rho_2 = m_2 - 1 \text{ AND } \rho_1 > m_1 - 1 \quad (5)$$

Proof:

Assume (5) holds true;

Then from (4) ρ_n has weight $w_n = \prod_{i=1}^{n-1} m_i$ which implies from (1) that the value of Z could be expressed as;

$$Z \geq \rho_1 + \rho_2 m_1 + \rho_3 m_1 m_2 + \cdots + M \quad (6)$$

Thus from property 1, Z lies outside the legitimate range $[0, M-1]$ which means overflow will occur when computing Z.

3.1 New Overflow Detection Algorithm

Given two RNS numbers $X = (x_1, x_2, \dots, x_n)$ and $Y = (y_1, y_2, \dots, y_n)$the following operations can be performed to detect overflow:

1. Compute the MRDs $(\omega_1, \omega_2, \dots, \omega_n)$ and $(\varphi_1, \varphi_2, \dots, \varphi_n)$of X and Y respectively (from (10))
2. Compute $(\rho_1, \rho_2, \dots, \rho_n)$, where $\rho_i = \omega_i + \varphi_i$ for $i = 1, n$.
3. Overflow occurs in $X + Y$ if the following hold true (from Theorem 1):

$$\rho_n > m_n - 1$$

$$\rho_n = m_n - 1 \text{AND } \rho_{n-1} > m_{n-1} - 1$$

$$\rho_n = m_n - 1 \text{AND } \rho_{n-1} = m_{n-1} - 1 \text{ AND } \rho_{n-2} > m_{n-2} - 1$$

.

.

.

$$\rho_n = m_n - 1 \text{AND } \rho_{n-1} = m_{n-1} - 1 \text{ AND } \rho_{n-2} = m_{n-2} - 1 \text{ AND…AND } \rho_2 = m_2 - 1 \text{ AND } \rho_1 > m_1 - 1$$

In the next sub-section, the proposed scheme is re-present for the moduli set $\{2^n - 1, 2^n, 2^n + 1\}$.

3.1.1 Simplified OEM for Overflow Detection in moduli Set $\{2^n - 1, 2^n, 2^n + 1\}$

This section presents a simplified algorithm for overflow detection for the moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ using the OEM algorithm proposed above.

Theorem 1: Given the moduli set $\{2^n - 1, 2^n, 2^n + 1\}$, where $m_1 = 2^n + 1$, $m_2 = 2^n$ and $m_3 = 2^n - 1$ for every integer $n > 1$, the following hold true:

$$|m_1^{-1}|_{m_2} = 1 \quad (7)$$

$$|m_2^{-1}|_{m_3} = 1 \quad (8)$$

$$|m_1^{-1}|_{m_3} = 2^{n-1} \quad (9)$$

Proof :If it can be demonstrated that $|(2^n + 1)1|_{2^n} = 1$, then 1 is the multiplicative inverse of m_1 with respect to m_2.

$$|(2^n + 1) * 1|_{2^n} = ||2^n|_{2^n} + |1|_{2^n}|_{2^n} = |1|_{2^n} = 1$$

Thus (7)holds true.

Similarly, if it can be demonstrated that $|(2^n) * 1|_{2^n-1} = 1$, then 1 is the multiplicative inverse of $m_2$with respect to m_3.

$$|(2^n) * 1|_{2^n-1} = ||2^n - 1|_{2^n-1} + |1|_{2^n-1}|_{2^n-1} = |1|_{2^n-1} = 1$$

Thus)8 (holds true.

Equation)9 (has been proved in literature]11[

Therefore we can re-write (2) as;

$$a_1 = x_1$$

$$a_2 = |(x_2 - a_1)1|_{2^n} = |x_2 - x_1|_{2^n}$$

$$a_3 = |((x_3 - a_1)2^{n-1} - a_2)1|_{2^n-1} = |(x_3 - a_1)2^{n-1} - a_2|_{2^n-1} = |(x_3 - a_1)2^{n-1} - a_2|_{2^n-1}$$

(10)

3.2 New Overflow Detection Algorithm for moduli set $\{2^n - 1, 2^n, 2^n + 1\}$

Given the RNS numbers $X = (x_1, x_2, x_3)$ and $Y = (y_1, y_2, y_3)$with respect to the moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ the generalized algorithm presented above could be simplified as;

1. Compute the MRDs $(\omega_1, \omega_2, \omega_3)$ and $(\varphi_1, \varphi_2, \varphi_3)$of X and Y respectively (from (10))
2. Compute (ρ_1, ρ_2, ρ_3), where $\rho_i = \omega_i + \varphi_i$ for $i = 1, n$.
3. Overflow will occur in $X + Y$ if the following hold true (from Theorem 1):

$$\rho_3 > 2^n - 2$$

$$\rho_3 = 2^n - 2 \; AND \; \rho_2 > 2^n - 1$$

$$\rho_3 = 2^n - 2 \; AND \; \rho_2 = 2^n - 1 \; AND \; \rho_1 > 2^n$$

4. IMPLEMENTATION

The proposed scheme uses basic and smaller logic units compared to similar state of the art designs. Our major goal is to compute ρ_1, ρ_2, and ρ_3 which are to be transmitted for post processing.

$$\rho_i = \omega_i + \varphi_i$$

Thus;

$$\rho_1 = \omega_1 + \varphi_1 = x_1 + y_1 \quad (11)$$

since $\omega_1 = x_1$ and $\varphi_1 = y_1$

similarly;

$$\rho_2 = \omega_2 + \varphi_2$$

where,

$$\omega_2 = |x_2 - x_1|_{2^n} = ||x_2|_{2^n} - |x_1|_{2^n}|_{2^n} = |x_2 - |x_1|_{2^n}|_{2^n}$$

and

$$\varphi_2 = |y_2 - y_1|_{2^n} = ||y_2|_{2^n} - |y_1|_{2^n}|_{2^n} = |y_2 - |y_1|_{2^n}|_{2^n} \quad (12)$$

since x_2 and y_2 are already modulo 2^n numbers finally;

$$\rho_3 = \omega_3 + \varphi_3$$

where

$$\omega_3 = |(z_1)2^{n-1} - \omega_2|_{2^n-1}$$

$$\varphi_3 = |(z_2)2^{n-1} - \varphi_2|_{2^n-1} \quad (13)$$

Where,

$$z_1 = |(x_3 - x_1)|_{2^n-1}$$

and,

$$z_2 = |(y_3 - y_1)|_{2^n-1}$$

4.1 Numerical Illustrations

This sub-section presents numerical illustrations of the proposed scheme.

4.1.1 Checking overflow in the sum of 49 *and* 21 *using RNS moduli set* $\{5, 4, 3\}$

$$49 = (4,1,1)_{RNS\langle 5|4|3\rangle} = (100,01,01)_{RNS\langle 101|100|11\rangle}$$

$$21 = (1,1,0)_{RNS\langle 5|4|3\rangle} = (001,01,00)_{RNS\langle 101|100|11\rangle}$$

$$= \big((100,01,01) + (001,01,00)\big)_{RNS\langle 101|100|11\rangle}$$

$$= (000,10,01)_{RNS\langle 101|100|11\rangle}$$

RNS to decimal conversion of $(000,10,01)_{RNS\langle 101|100|11\rangle}$ results in decimal number 10. Whilst it is well known that the sum of decimal numbers 49 and 21is 70, a clear sign that overflow has occurred.

Checking for RNS overflow using the proposed algorithm

$$\boldsymbol{\rho_1} = \omega_1 + \varphi_1 = x_1 + y_1 = 100 + 001 = 101$$

$$\boldsymbol{\rho_2} = \omega_2 + \varphi_2 = |x_2 - |x_1|_{2^n}|_{2^n} + |y_2 - |y_1|_{2^n}|_{2^n} = |01 - |100|_{100}|_{100} + |01 - |001|_{100}|_{100}$$

$$= |01 - 00|_{100} + |01 - 01|_{100} = |01|_{100} + |00|_{100} = 01 + 00 = 01$$

$$\boldsymbol{\rho_3} = \omega_3 + \varphi_3 = \omega_3$$

$$= |(z_1)2^{n-1} - \omega_2|_{2^n-1} + |(z_2)2^{n-1} - \varphi_2|_{2^n-1}$$

But $z_1 = |(x_3 - x_1)|_{2^n-1} = |01 - 100|_{11}$

$$= |-011|_{11} = 00$$

and$z_2 = |(y_3 - y_1)|_{2^n-1} = |00 - 001|_{11}$

$$= |-001|_{11} = 10$$

$\therefore \boldsymbol{\rho_3} = |(00)10 - 01|_{11} + |(10)10 - 00|_{11}$

$= |000 - 01|_{11} + |100 - 00|_{11}$

$$= |-001|_{11} + |100|_{11} = 10 + 01 = 11$$

Post Processing

Overflow will occur if the following hold true (from Theorem 1):

$$\rho_3 > 10$$

$$\rho_3 = 10 \; AND \; \rho_2 > 11$$

$$\rho_3 = 10 \; AND \; \rho_2 = 11 \; AND \; \rho_1 > \; 100$$

Examining our values at post processing would reveal that overflow has occurred since $\rho_3 = 11$ which is greater than 10.

4.1.2 Checking for overflow in the sum of 5 *and* 21 *using RNS moduli set* $\{5, 4, 3\}$

$5 = (0,1,2)_{RNS\langle 5|4|3\rangle} = (000,01,10)_{RNS\langle 101|100|11\rangle}$

$21 = (1,1,0)_{RNS\langle 5|4|3\rangle} = (001,01,00)_{RNS\langle 101|100|11\rangle}$

$$= \big((000,01,10) + (001,01,00)\big)_{RNS\langle 101|100|11\rangle}$$

$$= (001,10,10)_{RNS\langle 101|100|11\rangle}$$

RNS to decimal conversion of $(001,10,10)_{RNS\langle 101|100|11\rangle}$ results in decimal number 26 which is the correct result of $5 + 21$.

Checking for RNS overflow using the proposed algorithm

$$\boldsymbol{\rho_1} = \omega_1 + \varphi_1 = x_1 + y_1 = 000 + 001 = 001$$

$\boldsymbol{\rho_2} = \omega_2 + \varphi_2 = |x_2 - |x_1|_{2^n}|_{2^n} + |y_2 - |y_1|_{2^n}|_{2^n} = |01 - |000|_{100}|_{100} + |01 - |001|_{100}|_{100}$

$= |01 - 00|_{100} + |01 - 01|_{100} = |01|_{100} + |00|_{100} = 01 + 00 = 01$

$$\boldsymbol{\rho_3} = \omega_3 + \varphi_3 = \omega_3$$

$$= |(z_1)2^{n-1} - \omega_2|_{2^n-1} + |(z_2)2^{n-1} - \varphi_2|_{2^n-1}$$

But,

$z_1 = |(x_3 - x_1)|_{2^n-1} = |10 - 000|_{11} = |10|_{11} = 10$

and,

$z_2 = |(y_3 - y_1)|_{2^n-1} = |00 - 001|_{11} = |-001|_{11} = 10$

$\therefore \boldsymbol{\rho_3} = |(10)10 - 01|_{11} + |(10)10 - 00|_{11} = |100 - 01|_{11} + |100 - 00|_{11}$

$$= |11|_{11} + |100|_{11} = 00 + 01 = 01$$

Post Processing

Overflow will occur if the following hold true (from Theorem 1):

$$\rho_3 > 10$$

$$\rho_3 = 10 \; AND \; \rho_2 > 11$$

$$\rho_3 = 10 \; AND \; \rho_2 = 11 \; AND \; \rho_1 > \; 100$$

Examining our values at post processing reveals that overflow will not occurred since $\rho_3 = 01$ which is less than 10.

5. PERFORMANCE EVALUATION

In order to evaluate the performance of the proposed overflow detection scheme, it is compared with similar best known state of the art overflow detection schemes.

The proposed scheme requires nine additions whilst [5] and [6] which are the recent state of the art overflow detection schemes in literature require sixteen and fifteen additions respectively, our proposal also uses less modulo adders compared to both [5] and [6].

More specifically, the proposed scheme completely eliminates the pre-computation of the Modulo M sum of X and Y, and reduces the modulo operation from modulo $M = \prod_{i=0}^{n} m_i$ to $M = m_i$. Additionally, the proposed technique is manipulating smaller numbers compared to other techniques. It is well known that the smaller the numbers involved in computation, the faster the arithmetic operations. Thus, the proposed scheme is faster than other techniques.

6. CONCLUSION

In this paper, a general scheme for RNS overflow detection is proposed and based on it a special scheme for the moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ was also proposed. The proposed scheme performs better than current similar state of the art designs in terms of both area cost and delay.

7. REFERENCES

[1] Szabo and R. Tanaka, (1967), "Residue Arithmetic Technology",New York: McGraw Hill.

[2] H. Siewobr, K. A. Gbolagade (2011), "An Efficient RNS Overflow Detection Algorithm", Far East Journal of Electronics and Communications, Vol. (6/2) pp. 83-91.

[3] K. A. Gbolagade and S. D. Cotofana, "Generalized matrix method for efficient residue to decimal conversion," in Proceeding of the 10th IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS '08), pp. 1414–1417, Macao, China, December 2008.

[4] K.A. Gbolagade and S.D. Cotofana, "An O(n) Residue Number System to Mixed Radix Technique", IEEE International Symposium on Circuits and Systems (ISCAS 2009), pp. 521-524, Taipei, Taiwan, China, May, 2009.

[5] M. Rouhifar, M. Hosseinzadeh and M. Teshnehlab,(2011), "A new approach to Overflow detection in moduli set (2^n, 2^n-1, 2^{n-1}-1)", International Journal of Computational Intelligence and Information Security, Vol. 2, No. 3, pp. 35-43.

[6] M. Rouhifar, M. Hosseinzadeh, S. Bahanfar and M. Teshnehlab (2011), "Fast Overflow Detection in Moduli Set $\{2^n - 1, 2^n, 2^n + 1\}$", International Journal of Computer Science Issues, Vol. (8/3), pp. 407-414.

[7] Theodore L. Houk, "Residue Addition Overflow Detection Processor", Boing Company, Seatle, Wash. Appl. No.:414276, Sep. 29, 1989.

[8] Theodore L. Houk, "Method and Apparatus for Pipelined detection of overflow inResidue Arithmetic Multiplication", Boing Company, Seatle, Wash. Appl. No.:472,237, Jan. 30, 1990.

[9] M. Askarzadeh, M. Hosseinzadeh and K. Navi (2009),"A New Approach to Overflow Detection in Moduli Set $\{2^n - 3, 2^n - 1, 2^n + 1, 2^n + 3\}$", Second International Conference on Computer and Electrical Engineering, Vol. 1, pp. 439-442.

[10] K.A. Gbolagade and S.D. Cotofana, "Residue Number System Operands to Decimal Conversion for 3-moduli sets**",** 51st Midwest Symposium on Circuits and Systems, Knoxville, USA, pp. 791-794, August, 2008.

[11] K. Ibrahim and S. Saloum, "An efficient residue to binary converter design", IEEE Trans. on Circuits and Systems, Vol. 35, pp 1156-1158, Sep., 1988.