A SINGLE BIT ERROR DETECTION AND CORRECTION BASED ON THEMRC AND THE MP TECHNIQUES IN RRNS ARCHITECTURE

Yaw Afriyie  
Department of Accountancy and Commerce  
School of Business and Law  
University for Development Studies  
Wa, Ghana.

M.I. Daabo  
Department of Computer Science  
Faculty of Mathematical Sciences  
University of Development Studies  
Navrongo, Ghana.

Abstract: This paper presents some results on single error detection and correction based on the Redundant Residue Number System (RRNS). The proposed technique utilizes the Mixed Radix Conversion (MRC) and the Modulus Projection (MP) algorithms that significantly simplifies the error correction process for integers. The MP considerably reduces the computational steps and hardware architecture and further improve the processing speed. This results in a considerable improvement in the speed by 97% and tends to require about 96% less hardware resources in the proposed scheme when compared with the existing scheme used in this work. The proposed scheme is built on simple adders in the design of the architecture which saw a considerable improvement in both area and speed in as compared to the work by Yangyang et al. [6] which used ROMs and latches for the design of their architecture.

Keywords: MP, MRC, Mixed Radix Digits, Residue Number System, Redundant Residue Number System (RRNS)

I. INTRODUCTION

The increasing demand for speed and accuracy in digital communication has led to the introduction of parallel computing. RNS is a form of parallel computing that was first introduced by Garner [1]. RNS provides a very fast arithmetic due to its capability of performing the carry-free operations, i.e. addition, subtraction and multiplication. RNS also possesses parallel and fault tolerant features, which are seen to be helpful for hardware implementation. Barsi and Maestri [2] in their work posited that RNS offers a great speed as a result of its carry-free nature. Because of this, these have led to the increase in the development of a number of error detection and error correction algorithms based on RNS. When some redundant residues are added, the RNS has the possibility of error detection and correction, hence, the term Redundant Residue Number System (RRNS). Redundancy is achieved through various schemes in RNS. Behrouz [3] indicated that the ratio of the redundant bits to information bit is important to any scheme. There are significant works by Mandelbaum [4] and Szabo [5] concerning RRNS in the detection and correction of errors. Some concepts such as legitimate and illegitimate range for consistency checking related to error techniques are also studied. An algorithm that is based on the detection and correction of single bit errors by Szabo and Tanaka [4] allowed for scaling by a product moduli from the RNS base on n clock cycles. The main difficulty in their work was that the scaling reduced the processing speed. The clock cycles denote the time required for the element operation. In the work by Yangyang et al. [6], a discussion of a single bit error correction algorithm that implements ROMs and latches were used. The cost of ROMs and latches are however expensive to build affecting both the area and delay of the architecture. Some useful investigations were conducted by Jenkins et al. [7] to detect and correct single bit errors based on the Mixed Radix Conversion (MRC) and the Base Extension (BEX) techniques to RRNS application in digital filters and residue number error checkers due to efficient pipeline architectures. Goldreich et al. [8], proposed a performance evaluation of Residue number based on the Chinese Remainder Theorem (CRT) in the detection and correction of errors in RRNS. The resulting effects of the schemes in [5]-[8] when compared to the MRC and the MP proposed in this work offers a low complexity and detects and corrects single bit errors faster. To this effect, the proposed scheme offers a great advantage in terms of area cost, delay and would be able to detect and correct single bit errors faster.

In this paper, we propose a new scheme that will show the effectiveness of the RRNS based on MRC and the MP that will detect and correct bit single errors.

II. RESIDUE ARITHMETIC FUNDAMENTALS

RNS is characterized by a set of k pairwise relatively prime positive integers, i.e. the greatest common divisor gcd(m_i, m_j) =1 for each i ≠ j, m_1, m_2, …, m_k called the moduli, that is formed in increasing, i.e., m_1 < m_2 < … < m_{k-1} < m_k. The products represent the interval (0, M) called the legitimate range that defines the useful computational range of the number system, that is,

\[ M = \prod_{i=1}^{N} m_i \quad \ldots \quad \ldots \quad \ldots \quad (1) \]

To represent positive and negative numbers, the dynamic range is defined as \([-(M-1)/2, (M-1)/2]\) if M is odd and M/2 if M is even. Every natural integer X in the legitimate range can be represented by a set of residues r_1, r_2, …, r_{k-1}, r_k where

\[ r_i \equiv X \pmod{m_i} \quad \ldots \quad \ldots \quad \ldots \quad (2) \]

With \( i \in \{1, k\} \) and \( |X|/m_i \) denotes X modulo \( m_i \). Due to the carry-free property, the three operations namely addition, subtraction and multiplication can be operated with respect to the moduli independently, i.e.,

\[ x_1 + x_2 + \ldots + x_k \equiv y_1 + y_2 + \ldots + y_k \equiv z_1 + z_2 + \ldots + z_k \equiv [x_1 + y_1] \pmod{m_1} \]

With * denotes the three operations. Consequently, RNS is able to provide a fast arithmetic.
III. CONVERSION

It is well known that MRC and CRT are approaches that are often applied in conversion. This can be seen in the work of Mandelbaum [4]. This study will be limited to the MRC and the MP techniques because the real-time implementation of the CRT involves a modular operation with a large integer $M$ which results in large complexities. Daabo [9] indicated that to prevent the computations with such larger $M$, the CRT satisfies the real-time signal processing time due to its parallel means of computation and there is a constant limit to this approach. The process of converting from conventional representations to RNS is known as forward conversion whilst converting from the RNS to the conventional representations is known as the reverse conversion. The residue to conversion whilst converting from the RNS to the conventional representations is known as the reverse conversion. The residue to conventional number representation is done mainly by the MRC or the CRT as seen in the work of Mohaboseini [10]. The MRC is carried out by a weighted approach. The MRC is expressed by the following equations:

$$X = a_1 + a_2m_1 + a_3m_1m_2 + a_4m_1m_2m_3 \ldots m_{k-1}$$(4)

where $a_{i,j=1, k}$ is the Mixed Radix Digits (MRDs) can be computed as:

$$a_2 = \left( [x_2 - a_1]m_1^{-1}m_2 \right) \mod m_2$$

$$a_k = \left( \left( [x_k - a_1]m_1^{-1}m_2 \cdots m_{k-2}]m_2 \mod m_k \right) \right)$$

This paper presents an efficient algorithm for detection and correction of single bit errors for the moduli set $\{2^n - 1, 2^n, 2^n + 1, 22n-3, 22n+1\}.$

The rest of this paper is organized as follows: Section 4 presents the proposed method. In Section 5, the hardware implementation of the proposed scheme is presented, a simplified algorithm with numerical illustrations are also presented. The performance of the proposed scheme is evaluated in Section 6 whilst the paper is concluded in Section 7.

IV. PROPOSED METHOD

This section provides a new method for detecting and correcting single bit errors in RRNS in the given moduli set.

a. Proposed Algorithm

The algorithm for the proposed scheme is given below:

1. Compute the integer message $X$ using the MRC.
2. Perform iterations using $C_i = \frac{n}{i}$ by discarding a residue at time
3. An error occurs if the integer message $X$ falls within the illegitimate range but not found within the legitimate range.
4. Declare the error in the residue digit

In the course of computing the MP into integers, the decoding algorithm is used. The algorithm is premised on the MP and the MRC. For the MP, we have:

$$X_i = X \mod m_i = M_i$$

(6)

For the given moduli set $S = \{2^n - 1, 2^n, 2^n + 1, 22n - 3, 22n + 1\}$ and $m_1 = 2n - 1, m_2 = 2n, m_3 = 2n + 1, m_4 = 22n - 3$ and $M_i = 2^n + 1$ and the respective $m_i - projections$ are:

$$M_1 = (2^n)^2 + 1) \mod (2^n + 1)$$

(7)

$$M_2 = (2^n - 1)(2^n + 1) \mod (2^n + 1)$$

(8)

$$M_3 = (2^n - 1)(2^n + 3) \mod (2^n + 1)$$

(9)

$$M_4 = (2^n - 1)(2^n + 3) \mod (2^n + 1)$$

(10)

$$M_i = (2^n - 1)(2^n + 3) \mod (2^n + 1)$$

(11)

The projections for the respective moduli are given as:

$$X_1 = \left| X \mod (2^{n+1})(2^{n+3})(2^{n+1}) \right|$$

(12)

$$X_2 = \left| X \mod (2^{n+1})(2^{n+3})(2^{n+1}) \right|$$

(13)

$$X_3 = \left| X \mod (2^{n+1})(2^{n+3})(2^{n+1}) \right|$$

(14)

$$X_4 = \left| X \mod (2^{n+1})(2^{n+3})(2^{n+1}) \right|$$

(15)

$$X_5 = \left| X \mod (2^{n+1})(2^{n+3})(2^{n+1}) \right|$$

(16)

The multiplicative inverses for the MRC based on the same moduli set are computed as follows:

$$\text{mod}\ 1 = \left( 2^n - 1 \right) \mod -1$$

(17)

$$\text{mod}\ 2 = \left( 2^n - 1 \right) \mod -1$$

(18)

$$\text{mod}\ 3 = \left( 2^n - 1 \right) \mod -1$$

(19)

$$\text{mod}\ 4 = \left( 2^n - 1 \right) \mod -1$$

(20)

Now the $a_y$ can be computed using the MRC as follows:

$$a_1 = a_1$$

(21)

$$a_2 = \left( (x_2 - a_1)m_1m_2 \right) \mod m_2$$

(22)

$$a_3 = \left( (x_3 - a_1)m_1m_3 \right) \mod m_3$$

(23)

The decimal equivalent for the non-redundant part is thus:

$$X = a_1 + a_2m_1 + a_3m_1m_2$$

(24)

The redundant part is:

$$a_3 = a_4$$

(25)

$$a_3 = a_4$$

(26)

$$a_3 = a_4$$

(27)

The decimal equivalent for the redundant part is thus:

$$X = a_1 + a_2m_1 + a_3m_1m_2$$

(28)

Thus by the MRC technique, we have:

$$a_1 = x_1$$

(35)

$$a_2 = [x_1 + x_2] \mod 2^n$$

(36)

$$a_3 = [x_1, x_2] \mod 2^n$$

(37)

Implementation of equations (30) – (37) gives the correct output of $a_3$ whatever an error occurs in the non-redundant part. The decimal representation of the architecture for the non-redundant from equations (28) and (29) is:

$$X = \left( x_1 + 2^n \cdot a_2 \right) \mod 2^n$$

(38)

$$= 2^n \cdot a_3$$

(39)

$$= 2^n \cdot a_3$$

(40)

where

© 2015-19, IARCS All Rights Reserved
\[ \bar{r} = \rho - 2^\nu a_3 - a_2 \]
\[ \rho_{2n-1} \ldots \rho_0 + \bar{a}_{3, n-1} \ldots \bar{a}_2 + 111 \ldots 1 + \]
\[ a_{2, n-1}, a_{2, n-2} \ldots a_{2, 0} \]
\[ \frac{2n}{n} \text{bit} \]
\[ \frac{n}{n} \text{bit} \]
\[ \frac{1}{1} \]
\[ \frac{2}{2} \]
\[ \frac{3}{3} \]
\[ \frac{4}{4} \]
\[ \frac{5}{5} \]
\[ \frac{6}{6} \]
\[ \frac{7}{7} \]
\[ \frac{8}{8} \]
\[ \frac{9}{9} \]
\[ \frac{10}{10} \]
\[ \frac{11}{11} \]

VI. Proposed Architecture

The residue number is converted to the Mixed Radix System (MRS) in parallel with the computation of the MRS, which detects and corrects primarily based on the non-redundant part. In the event of an error in any of the channels, the redundant part will be employed in the detection and correction of the residue digit error. The Mixed Radix Digits (MRDs) are computed according to equation (24) where all the MRDs \( a_1, a_2 \) and \( a_3 \) are computed individually in equations (21) to (23). The \( a_{\nu} \) which are the MRDs for the non-redundant part are computed separately which is seen in Figure 1. As shown in Figure 1, \( a_3 \) is computed using Carry Save Adders (CSAs) 1, 2 and 3 and two regular \((n+1)\) bit Carry Propagate Adders (CPAs) 1 and 2 respectively. All the CSAs require an area of \((n+1)\Delta_{FA}\) each whilst CPAs 1, 2 and 3 require an area of \(n\) each. In order to obtain the MRD \( a_3 \) will require a total area of \((11n+4)\Delta_{FA}\). Regarding the delay, CSA (i.e. CSAs 1, 2 and 3) impose a delay of \(D_{FA}\) each in the reverse convertor. CPAs 1 in the reverse convertor require a delay of \((4n+2)D_{FA}\), CPAs 2 and 3 in the reverse convertor require a delay of \((6n+20)D_{FA}\) each. The reverse convertor for the MRC also has one CSA that also impose a delay on the system. The total delay needed for the proposed scheme is \((10n+7)D_{FA}\). The schematic diagram for the proposed scheme is shown in Figure 1.

The schematic diagrams for the proposed scheme are shown below.

Figure 1: Block Diagram of RC for the non-redundant part

VII. Numerical Results

Let us now consider some numerical illustrations with the proposed scheme.

Consider an \((n, k)\) code where \(n\) is the length of the code and \(k\) is the dimension of the code with the moduli set \(m_1, m_2, m_3, m_4, m_5 = (3, 4, 5, 13, 17)\) where \(m_1, m_2\) and \(m_3\) are non-redundant moduli, \(m_4\) and \(m_5\) are the redundant moduli. We consider the integer message \(X=57\), for its residue digits are \(x_i = (0, 1, 2, 5, 6)\). The legitimate range = \(M_R = 3^4\times5^6 = 60\) and the illegitimate range = \(M_I = 13\times17 = 221\). Assume that during storage or computation, an error occurs in the second residue digit such that \(x_3 = 3\). Therefore, the received codevector will be \(\bar{x}_1 = (0, 3, 2, 5, 6)\).

\[ 6687_3 = 6687_{(42)} = 2267 \]
\[ 6687_4 = 6687_{(3115)} = 57^* \]
\[ 6687_5 = 6687_{(2652)} = 1383 \]
The ability to detect and correct errors in a digital system improves its reliability and integrity. In this work, the MRC decoding technique is compared with the MP. It is found that the MRC decoding processes result in more computational steps in detecting and correcting errors than in the proposed implementation. A theoretical analysis shows that the MP recovers integer messages faster and improves computational speed than the MRC decoding processes as shown in Figure 2.

To evaluate the performance of the proposed scheme, the work is premised on both the Modulus Projection and the Mixed Radix Conversion. The MP considerably reduced the iterative steps and improved the speed of the architecture. Generally, the proposed scheme was built using simple adders instead of ROMs and latches that are associated with high cost of implementation. An area/delay analyses showed that there is a considerable improvement in the speed by 97% and tends to require about 96% less hardware resources in the proposed technique.

Table 1: Area, Delay Comparison

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Area($\Delta_{FA}$)</th>
<th>Delay($D_{FA}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yangyang et. al [6]</td>
<td>$40n^2 + 20n$</td>
<td>$72n^2 + 40n$</td>
</tr>
<tr>
<td>Proposed</td>
<td>$11n + 4$</td>
<td>$10n + 7$</td>
</tr>
</tbody>
</table>

Figures 3 and 4 show the graphical illustrations of the area and delay comparisons.

VIII. PERFORMANCE EVALUATION

Table 1: Area, Delay Comparison

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Area($\Delta_{FA}$)</th>
<th>Delay($D_{FA}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yangyang et. al [6]</td>
<td>$40n^2 + 20n$</td>
<td>$72n^2 + 40n$</td>
</tr>
<tr>
<td>Proposed</td>
<td>$11n + 4$</td>
<td>$10n + 7$</td>
</tr>
</tbody>
</table>

REFERENCES


© 2015-19, IJARCS All Rights Reserved


