A Novel Error Detection And Correction Technique for RNS based FIR Filters

S. Pontarelli‡, G.C. Cardarilli†, M. Re†, A. Salsano†
{pontarelli, salsano}@ing.uniroma2.it,
{marco.re, g.cardarilli}@ieee.org
†University of Rome "Tor Vergata", Department of Electronic Engineering
Via del Politecnico 1, 00191, Rome, ITALY
‡(ASI) Italian Space Agency, Viale Liegi, 26 00198 Rome, ITALY

Abstract

In this paper a novel technique for detecting and correcting errors in the RNS representation is presented. It is based on the selection of a particular subset of the legitimate range of the RNS representation characterized by the property that each element is a multiple of a suitable integer number m. This method allows to detect and correct any single error in the modular processors of the RNS based computational unit. This subset of the legitimate range can be used to perform addition and multiplication in the RNS domain allowing the design of complex arithmetic structures like FIR filters. In the paper, the architecture of a FIR filter with error detection and correction capabilities is presented showing the advantages with respect to filters in which the error detection and correction are obtained by using the traditional RNS technique.

I. Introduction

Nowadays digital systems are very often used to implement high complexity Digital Signal Processing (DSP) algorithms working in real time. The requirements in terms of speed and circuit complexity of these applications are stringent, and the mandatory use of technologies with the best available feature size increases the probability of the occurrence of faults.

To face these problems much research work has been published on fault detection in DSP architectures. In particular, with respect to the basic arithmetic operations, self-checking adders based on residue codes [1], [2], parity codes [3], or Berger codes [4] have been proposed. With respect to basic DSP building blocks, the RRNS (Redundant Residue Number System) representation has been used in the implementation of FIR filters [5], [6], [7] allowing fault detection and correction.

The use of RRNS gives to the designer, advantages both in terms of error detection and correction capabilities and in terms of maximum operating frequency. In fact, the operations on each residue digit are independent and so the addition, subtraction, and multiplication on the full dynamic range are split in different channels and performed in parallel on each of the moduli over a smaller dynamic range. The error detection in RRNS is based on the following consideration: the representation range is divided in two intervals: the *legitimate range* and the *illegitimate range*. An error in a single module is detected if, after the conversion from the RNS to the two's complement representation, the result belongs to the illegitimate range.

In this work the RNS representation range defined by the chosen moduli set is divided in two subsets: the legitimate subset is composed by any element that is exactly divisible by an additional modulus, i.e. the integer number m, while the other elements belong to the illegitimate subset. The paper shows how elements of the legitimate subset can be used to perform addition and multiplication and the architecture of a FIR filter with fault detection and correction capabilities is presented.

The paper is organized as follows: in Section II a background on RNS and RRNS arithmetic is given. In Section III, the new method to obtain error detection is presented, and the implementation of a FIR filter based on this method is described. In Section IV the extension of this method to achieve error correction is presented, while in Section V the implementation of a set of example FIR filter with error correction capabilities is shown, and the advantages with respect to the traditional RRNS representation are discussed. The Conclusions are drawn in Section VI.

II. RNS-RRNS Background

In this section a short background on RNS and RRNS is presented, and the traditional techniques to implement FIR filters with error detection and correction capabilities based on the RRNS are shown.

A. Background on Residue Number System

A Residue Number System (RNS) is defined by a set of relatively prime integers $\{m_1, m_2, \cdots, m_P\}$.

The dynamic range of the system is given by the product of the moduli m_i:

$$M = \prod_{i=1}^{P} m_i$$

1550-5774/08 $25.00 © 2008 IEEE
DOI 10.1109/DFT.2008.32

Any integer $X \in [0, M-1]$ has a unique RNS representation given by

$$X \xrightarrow{RNS} (< X >_{m_1}, < X >_{m_2}, \cdots, < X >_{m_P}) \tag{1}$$

where $< X >_{m_i} = X \bmod m_i$

A comprehensive description of the RNS theory and its applications to computer systems and DSP can be found in [8], [9], and [10]. In RNS, operations such as addition and multiplication, are executed in parallel as shown in the following formula

$$Z = X \text{ op } Y \xrightarrow{\text{RNS}} \begin{cases} < Z >_{m_1} = < X_{m_1} \text{ op } Y_{m_1} >_{m_1} \\ \cdots \\ < Z >_{m_P} = < X_{m_P} \text{ op } Y_{m_P} >_{m_P} \end{cases} \tag{2}$$

As a consequence, arithmetic operations are split into several modular operations with reduced word length.

The conversion of the RNS representation of Z is accomplished by using the Chinese Remainder Theorem (CRT)

$$Z = CRT(Z_{m_1}, \cdots, Z_{m_P}) = \left\langle \sum_{i=1}^{P} < Z_{m_i} \cdot k_i >_{m_i} \cdot M_i \right\rangle_M \tag{3}$$

where $M_i = \frac{M}{m_i}$ and k_i are obtained by $< M_i \cdot k_i >_{m_i} = 1$, *i.e.* they are the multiplicative inverse of M_i modulo m_i.

Clearly, the input and output conversions, constitute a significant overhead in systems implemented in RNS. However, efficient methods to perform those conversions are presented in [11], [12], and [13].

B. Background on Redundant Residue Number System for error detection and correction

A Redundant Residue Number System (RRNS) is defined as a residue number system added with r additional moduli. The first k moduli form a set of non redundant moduli, and their product represents the legitimate range, M that is,

$$M = \prod_{i=1}^{k} m_i$$

The remaining $P - k = r$ moduli form the set of redundant moduli that allows error detection and correction where M_R is defined as

$$M_R = \prod_{i=k+1}^{P} m_i$$

Given a residue vector $(x_{m_1}, \ldots, x_{m_P})$, the corresponding integer X belongs to the interval $[0, M_T - 1]$, where $M_T = M \cdot M_R$.

This interval, usually called total range, can be divided into two adjacent intervals by considering the ranges defined by the non redundant and redundant moduli. The interval $[0, M-1]$ is called the *legitimate range* and the interval $[M, M_T - 1]$ is the *illegitimate range*.

In RRNS error detection and correction is obtained by constraining the operands and the results in the legitimate range. This restriction defines the dynamic range of the system. The m_i-projection of X, denoted X_i, is defined as the residue vector $(x_{m_1}, \cdots, x_{m_{i-1}}, x_{m_{i+1}}, \cdots, x_{m_P})$, *i.e.* the representation of X with the i-th residue digit deleted.

A single error occurs when a legitimate vector $(x_{m_1}, \cdots, x_{m_i}, \cdots, x_{m_P})$ is changed into a different residue vector, $(x_{m_1}, \cdots, \overline{x}_{m_i}, \cdots, x_{m_P})$ by the occurrence of an error in the i-th digit, the number corresponding to this vector is $\overline{X}$.

In [6] has been proved that, under the hypothesis of ordered moduli (i.e. $m_i < m_{i+1}\ \forall i$), in a RRNS representation with $r = 2$ any error in a single module produces an illegitimate number $\overline{X}$. Moreover, $\overline{X}_i$ is legitimate, where i is the residue affected by an error, while the other projections $\overline{X}_j$, for $j \neq i, i = 1, \ldots, P$ are all illegitimate. From these considerations is straightforward to detect and correct an error in the RRNS representation. The erroneous module is that characterized by his m_i-projection belonging to the legitimate range, while the correct value of the integer can be obtained by performing the reverse conversion of the X_i projection.

C. Implementation of FIR Filters in RNS

A N taps FIR filter is expressed by

$$y(n) = \sum_{k=0}^{N-1} a_k x(n-k) \tag{4}$$

The RNS implementation of the FIR filter is a direct consequence of equation (2),

$$y(n) = \sum_{k=0}^{N-1} a_k x(n-k) \xrightarrow{RNS} \begin{cases} Y_{m_1}(n) = \left\langle \sum_{k=0}^{N-1} \langle < a_k >_{m_1} \cdot x_{m_1}(n-k) \rangle_{m_1} \right\rangle_{m_1} \\ \cdots \\ Y_{m_P}(n) = \left\langle \sum_{k=0}^{N-1} \langle < a_k >_{m_P} \cdot x_{m_P}(n-k) \rangle_{m_P} \right\rangle_{m_P} \end{cases} \quad (5)$$

Adding error detection capabilities to the FIR filter requires the use of an additional modulus m_{P+1} and a comparator after the RNS to binary converter. A well know architecture of a RNS FIR filter capable of error detection using the RRNS representation is given in Fig. 1 ([6]).

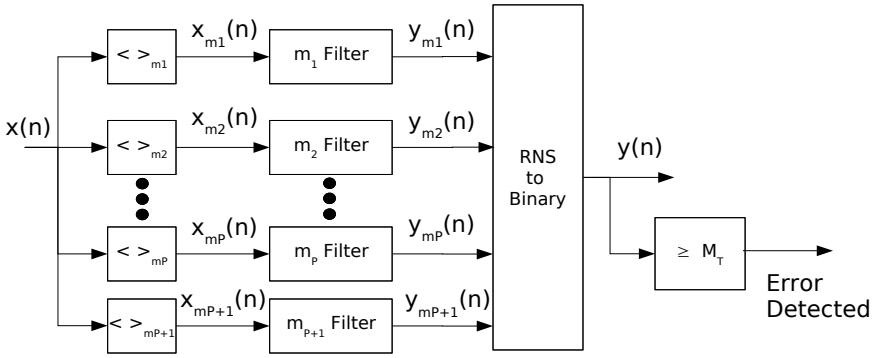

Fig. 1. RNS implementation of a FIR filter with error detection

The binary to RNS conversion is performed by reducing modulo m_i the input sequence $x(n)$, providing the residue digits x_{m_i}. The parallel filters mod m_i compute the residues Y_{m_i} (eq. (5)), while the the standard binary representation of the result $y(n)$ is obtained by the RNS to the binary conversion block. An error inside one of the modulo m_i filters produces a result belonging in the *illegitimate range* and the comparator after the RNS to the binary converter allows the detection this error.

An RRNS FIR filter with error correction capabilities is shown in Fig. 2 in the case of P=3.

It requires two additional moduli with respect to the RNS representation, a reverse converter for each m_i projection, and a block choosing the value of the m_i projection that is in the legitimate range. The blocks called CRT (Chinese Remainder Theorem) performs the reverse conversion for the m_i projections, while the block called "choose legitimate" selects which input belong in the legitimate range.

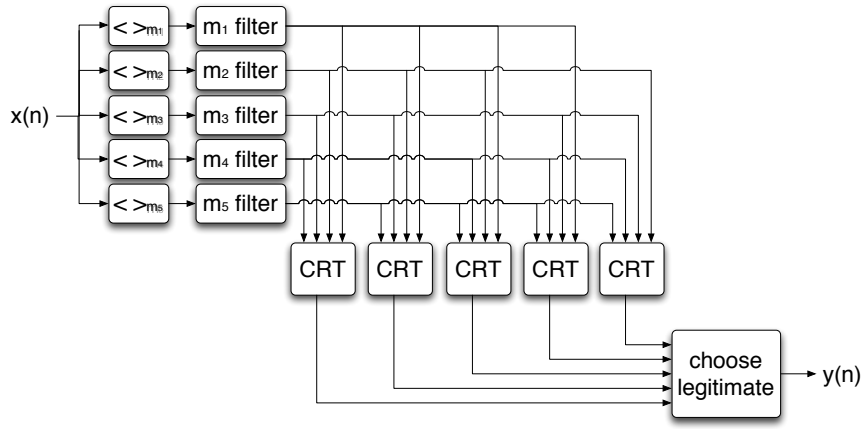

Fig. 2. RNS implementation of a FIR filter with error correction

The main overhead of this architecture is related to the implementation of a CRT block for each module. For a RNS with a moduli set of i element we need i CRT blocks. Each CRT block is composed by:

1) $i-1$ modulo m_i constant multipliers to compute $< Z_{m_i} \cdot k_i >_{m_i}$
2) $i-1$ constant multipliers for M_i
3) $i-2$ modulo M adders

Therefore, the overall overhead due to the i CRTs grows quadratically with the number of element i of the moduli set.

III. Error detection in RNS by using scaled values

To define the method for error detection in RNS using scaled values we restrict the discussion to a moduli set composed only by prime numbers. The choice is not reductive because in RNS the use of prime moduli gives to the designer the possibility to exploit the isomorphism technique [8] to perform multiplication modulo a prime avoiding the use of multipliers.

Given the moduli set $\{m_1, m_2, \cdots, m_P\}$ the range of the traditional RNS representation is $[0, M-1]$, with $M = \prod_{i=1}^{P} m_i$. Without loss of generality, we suppose that the moduli set is ordered. The subset $\mathcal{C}$ of the range constitute the legitimate subset and is formed by the numbers $x \in [0, M-1]$ with $m \mid x$ (*i.e.* x is exactly divisible by m), with the constrains that m is relatively prime with respect to each element of the moduli set and $m > m_P$. The illegitimate subset is composed by all the numbers not belonging to $\mathcal{C}$. The mapping between the integer representation and the Scaled RNS (SRNS) representation is defined by

$$X \xrightarrow{Scaled\ RNS} (< m \cdot X >_{m_1}, < m \cdot X >_{m_2}, \cdots, < m \cdot X >_{m_P}) \tag{6}$$

The SRNS representation range is $[0, M_s - 1]$, with $M_s = \lfloor \prod_{i=1}^{n} m_i / m \rfloor$. It must be noticed that if $m \approx m_P$, the range representable with this mapping is similar to the range of an RRNS representation with $\{m_1, m_2, \cdots, m_P\}$ as the set of non redundant moduli and $m = m_{P+1}$ as the redundant modulus.

For the SRNS representation a relationship similar to equation (2) can be used for the addition operation, i.e.

$$Z = X + Y \xrightarrow{Scaled\ RNS} \begin{cases} < m \cdot Z >_{m_1} = << m \cdot X >_{m_1} + < m \cdot Y >_{m_1} >_{m_1} \\ \cdots \\ < m \cdot Z >_{m_P} = << m \cdot X >_{m_P} + < m \cdot Y >_{m_P} >_{m_P} \end{cases} \tag{7}$$

that is valid until the operand and the results belong to the range $[0, M_s - 1]$.

Instead, for multiplication the following relationship holds:

$$Z = X \cdot Y \xrightarrow{Scaled\ RNS} \begin{cases} < m \cdot Z >_{m_1} = << X >_{m_1} \cdot < m \cdot Y >_{m_1} >_{m_1} \\ \cdots \\ < m \cdot Z >_{m_P} = << X >_{m_P} \cdot < m \cdot Y >_{m_P} >_{m_P} \end{cases} \tag{8}$$

that is valid until the operand and the results belong to the range $[0, M_s - 1]$. We invite the reader to notice that, differently from addition, multiplication is performing scaling only one of the two operands.

Now let us suppose that a single module error occurs to a number represented in the scaled RNS. If the original value is $X = (x_{m_1}, \cdots, x_{m_i}, \cdots, x_{m_P})_{SRNS}$, the corresponding erroneous value can be expressed as $\overline{X} = (x_{m_1}, \cdots, \overline{x}_{m_i}, \cdots, x_{m_P})_{SRNS}$ and the value of the single module error is defined as $E = (0, \cdots, e_i, \cdots, 0)$, with $e_i = x_{m_i} - \overline{x}_{m_i}$ and $e_i \in [0, m_i - 1]$. The error is detected if the value $\overline{X}$ is an element of the illegitimate range, i.e. $m \nmid X$ (m not divide X). This condition occurs if the integer representation of the error E in not divisible by m.

Now, we define the condition under which all possible single module errors E respect the condition $m \nmid E$.

Applying the Chinese Remainder Theorem to E we obtains

$$E = CRT((0, \cdots, e_i, \cdots, 0)) = < e_i \cdot k_i >_{m_i} \cdot \prod_{j=1, j \neq i}^{n} m_j$$

The constrain that m is relative prime with respect to m_i have as a consequence that

$$m \mid E = e_i \cdot k_i \cdot \prod_{j=1, j \neq i}^{n} m_j \iff m \mid < e_i \cdot k_i >_{m_i} \tag{9}$$

where k_i are the multiplicative inverse of M_i modulo m_i.

The value of $< e_i \cdot k_i >_{m_i}$ is less than m_i and therefore a sufficient condition for equation (9) is:

$$m > m_i \ \forall i \tag{10}$$

For the set of moduli that respect the condition expressed in eq. (9) the single module error detection can be performed easily by checking the remainder modulo m of the result after the RNS to binary conversion. From the above discussion is possible to detect an error in the scaled RNS representation computing $< E >_m$. If no errors occur $< E >_m = 0$, otherwise $< E >_m \neq 0$.

scaled RNS value	integer value	RNS representation
0	0	$(0,0,0)$
1	11	$(2,1,4)$
2	22	$(1,2,1)$
3	33	$(0,3,5)$
4	44	$(2,4,2)$
5	55	$(1,0,6)$
6	66	$(0,1,3)$
7	77	$(2,2,0)$
8	88	$(1,3,4)$
9	99	$(0,4,1)$

TABLE I

RNS REPRESENTATIONS FOR THE CODEWORD DIVISIBLE BY 11

A. A numerical example

Given the set of moduli $3,\ 5,\ 7$, $M = 105$, if $m = 11$ is chosen, the codeword values in the RNS range are {0,11,22,33,44, 55,66,77,88,99}. For the chosen moduli set the k_i values are $2,\ 1,\ 1$ and equation (9) is respected for all the possible e_i values. The correct codewords represented in RNS are shown in Table I. In the first column the value used in the proposed coding is reported, while in the second column the corresponding integer value is shown.

All the values that a single module error can assume are reported in Table II. It is easy to see that for all the possible errors, the residue modulo 11 is always different from zero, allowing the detection of any possible single module error.

error RNS coordinates	Integer value	value mod 11
$(1,0,0)$	70	4
$(2,0,0)$	35	2
$(0,1,0)$	21	10
$(0,2,0)$	42	9
$(0,3,0)$	63	8
$(0,4,0)$	84	7
$(0,0,1)$	15	4
$(0,0,2)$	30	8
$(0,0,3)$	45	1
$(0,0,4)$	60	5
$(0,0,5)$	75	9
$(0,0,6)$	90	2

TABLE II

VALUE OF ERRORS IN AN RNS MODULE

An arithmetic computation in the SRNS representation is illustrated in the following example. Given the expression $2 \cdot 3 + 1 = 7$, using the SRNS representation the operation without error can be expressed as

$$(2,2,2) \cdot (0,3,5) + (2,1,4) = (2,2,0)$$

The multiplication step is performed between the scled value of 3, $i.e.$ (0,3,5) and the unscaled value of 2, $i.e.$ (2,2,2). Now we suppose that an error occurs in the second residue changing the result from 2 to 4 (the error is $(0,2,0)$). The erroneous result is $(2,4,0)$, that corresponds to the integer value 14, that is not divisible by 11.

B. FIR filter with error detection capability

Using the result presented in the previous section the design of a FIR filter with error detection capability that exploits the characteristic of the SRNS representation is obtained by using the following additional hardware resources:

1) a constant multiplier before the conversion of the input sequence into the RNS representation,
2) a constant divider after the conversion of the output sequence from the RNS representation to the binary one,
3) a modulo m reduction block to check if the result is a multiple of m.

The architecture of the FIR filter with error detection is shown in Fig. 3.

The $< a_k >_{m_i}$ coefficients of the FIR filter are computed in standard RNS representation to respect the equivalence defined in equation (8) for multiplication in scaled RNS. The overhead of this schema is strictly dependent by the choice of the scaling factor m. With the right choice this overhead can be drastically reduced. For example, if we choose $m = 2^i$, the multiplication is performed by shifting left the input sequence samples, the division by shifting right discarding the i less significant bits of the result after the RNS to binary conversion, the modulo m reduction of the result consists in taking the i less significant bits of the result, and the congruence to zero is checked by the logic OR of the i less significant bits. In this case, the overhead of the additional hardware resources, is negligible. In fact, the additional hardware resources for $m = 2^i$ are:

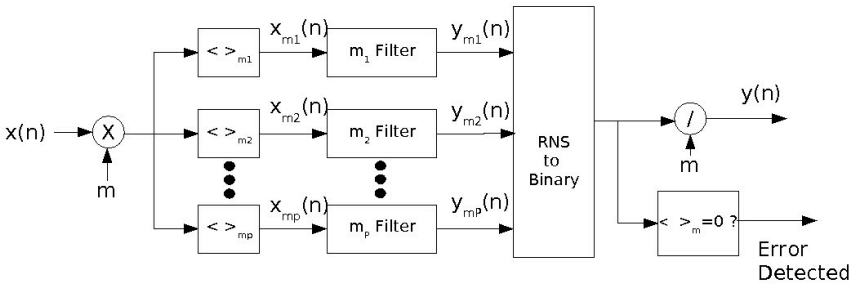

Fig. 3. SRNS implementation of a FIR filter with error detection capability

1) the constant multiplier simply is a shift of the input bits,
2) the result of the division by 2^i is obtained discarding the i less significant bits.
3) the bit composing the result of the modulo 2^ioperation are the i less significant bits of the RNS to binary conversion.
4) the block for error detection is an i-input OR.

IV. Error correction in RNS by using scaled values

To add error correction capabilities to the scaled RNS representation the sufficient and necessary condition is that two different errors correspond to two different values of $< E >_m$. We call this condition as no aliasing condition. The no aliasing condition allows to define a one to one map between the value of $< E >_m$ and the value of E that is the value of the error occurred in the scaled RNS representation. Subtracting E to the output of the RNS to binary conversion of Fig. 3 we can obtain the correct value of the output. To identify the relationship between the moduli set $\{m_1, m_2, \cdots, m_P\}$ and m that satisfy the no aliasing condition we take two different errors $E' = (0, \cdots, e'_i, \cdots, 0)$ and $E'' = (0, \cdots, e''_j, \cdots, 0)$.

First of all we define the no aliasing condition $< E' >_m \neq < E'' >_m$ as $< E' - E'' >_m \neq 0$.

If $i = j$ the errors E' and E'' correspond to errors occurred in the same m_i module. In this case $E' - E'' = (0, \cdots, e'_i - e''_i, \cdots, 0)$ and the no aliasing condition correspond to the condition defined in the previous section for error detection. Instead, if the errors occur in different moduli the value $E' - E''$ is:

$$E' - E'' = (0, \cdots, e'_i, 0, \cdots 0, e''_j, \cdots, 0)$$

To show the condition that must be satisfied in this case, we remark that the moduli set $\{m_1, \cdots, m_i, \cdots, m_j, \cdots, m_P\}$ is equivalent to the moduli set $\{m_1, \cdots, m^*, \cdots, m_P\}$, where $m^* = m_i \cdot m_j$ substitute the two moduli m_i and m_j. Using this moduli set equivalence the value of $E' - E''$ can be expressed as:

$$E' - E'' = (0, \cdots, e^*, \cdots, 0)$$

where $e^* = CRT(e'_i, e''_j)$ and the CRT is computed on the moduli set composed by m_i and m_j. After this manipulation the case of $i \neq j$ is re-conducted to the first case and the no aliasing condition is the same one defined for error detection. The condition $< E' - E'' >_m \neq 0$ is therefore equivalent to $m^* < m$. This condition is verified for all the value of i and j if

$$m > m_i \cdot m_j \ \forall i, j \tag{11}$$

If the condition expressed in eq. (11) is satisfied all possible errors occurring in a single module correspond to one and only one value of $< E >_m$ and therefore from the value of $< E >_m$ we can derive the error value E.

Let us suppose that an error e_i occurs in the i module changing the value of the output from Y (the correct value) to $\bar{Y} = Y + E$. The value of E is:

$$E = e_i \cdot M_i \tag{12}$$

where e_i is in the range $[-m_i + 1, m_i - 1]$.

The corresponding $< E >_m$ value is:

$$< E >_m = < e_i \cdot M_i >_m$$

Now we can multiply modulo m for the inverse modulo m of M_i. This inverse, M_i^{-1} exist because m is relative prime with respect to all the m_i, and therefore is prime with respect to M_i. After this multiplication the equation becomes:

$$< e_i >_m = < M_i^{-1} \cdot E >_m = << M_i^{-1} >_m \cdot < E >_m >_m$$

Defining

$$\bar{e}_i = << M_i^{-1} >_m \cdot < E >_m >_m$$

and using the inequality $m_i < m$ the following equation holds:

$$e_i = \begin{cases} \bar{e}_i & if \quad \bar{e}_i < m_i \\ \bar{e}_i - m & if \quad \bar{e}_i > m_i \end{cases} \tag{13}$$

The e_i values computed from equation (13) can be used to compute some candidate erroneous values E_i defined as $E_i = e_i \cdot M_i$.

The correct value E is one of the E_i candidate values. The choice of the right E_i value can be performed computing $< \bar{Y} - E_i >_M$. The only value of $< \bar{Y} - E_i >_M$ that is exactly divisible by m is the corrected value of output. The uniqueness and the existence of an E_i that satisfy the condition $< \bar{Y} - E_i >_M | \ m$ is assured by the no aliasing condition previously discussed.

A. A numerical example

Now we give a numerical example of the scaled RNS representation with error correction capabilities. The moduli set is chosen as $\{11,\ 13,\ 15\}$, $M = 2145$, and $m = 256$.

The M_i values for the chosen moduli set are $M_1 = 195$, $M_2 = 165$, $M_3 = 143$, while $k_1 = 7$, $k_2 = 3$, $k_3 = 2$.

The values of $< M_i^{-1} >_{256}$ are $< M_1^{-1} >_{256} = 235$, $< M_2^{-1} >_{256} = 45$ and $< M_3^{-1} >_{256} = 111$.

The codeword values in the RNS range are $0, 256, 512, 768, 1024, 1280, 1536, 1792, 2048$ that are all divisible by 256. The correct codewords represented in RNS are shown in Table III. In the first column the value used in the proposed coding is reported, in the second column the corresponding integer value is shown and in the last column the RNS representation is given.

scaled RNS value	integer value	RNS representation
0	0	$(0, 0, 0)$
1	256	$(3, 9, 1)$
2	512	$(6, 5, 2)$
3	768	$(9, 1, 3)$
4	1024	$(1, 10, 4)$
5	1280	$(4, 6, 5)$
6	1536	$(7, 2, 6)$
7	1792	$(10, 11, 7)$
8	2048	$(2, 7, 8)$

TABLE III

ERROR CORRECTING SCALED RNS REPRESENTATIONS CODEWORDS

All the values that a single module error can assume are reported in Table IV. It is easy to see that for all the possible errors, the residue modulo 256 is always different from zero. The no aliasing condition is verified, and all modulo 256 values the are different.

error RNS coordinates	Integer value	value mod 256	error RNS coordinates	Integer value	value mod 256	error RNS coordinates	Integer value	value mod 256	error RNS coordinates	Integer value	value mod 256
$(1, 0, 0)$	1365	85	$(10, 0, 0)$	780	12	$(0, 9, 0)$	165	165	$(0, 0, 6)$	1716	180
$(2, 0, 0)$	585	73	$(0, 1, 0)$	495	239	$(0, 10, 0)$	660	148	$(0, 0, 7)$	2002	210
$(3, 0, 0)$	1950	158	$(0, 2, 0)$	990	222	$(0, 11, 0)$	1155	131	$(0, 0, 8)$	143	143
$(4, 0, 0)$	1170	146	$(0, 3, 0)$	1485	205	$(0, 12, 0)$	1650	114	$(0, 0, 9)$	429	173
$(5, 0, 0)$	390	134	$(0, 4, 0)$	1980	188	$(0, 0, 1)$	286	30	$(0, 0, 10)$	715	203
$(6, 0, 0)$	1755	219	$(0, 5, 0)$	330	74	$(0, 0, 2)$	572	60	$(0, 0, 11)$	1001	233
$(7, 0, 0)$	975	207	$(0, 6, 0)$	825	57	$(0, 0, 3)$	858	90	$(0, 0, 12)$	1287	7
$(8, 0, 0)$	195	195	$(0, 7, 0)$	1320	40	$(0, 0, 4)$	1144	120	$(0, 0, 13)$	1573	37
$(9, 0, 0)$	1560	24	$(0, 8, 0)$	1815	23	$(0, 0, 5)$	1430	150	$(0, 0, 14)$	1859	67

TABLE IV

VALUE OF ERRORS IN AN RNS MODULE

In the following example, an arithmetic computation in the SRNS representation is illustrated. Given the expression $2 \cdot 3 + 1 = 7$, using the SRNS representation the operation without error can be expressed as

$$(2, 2, 2) \cdot (9, 1, 3) + (3, 9, 1) = (10, 11, 7)$$

Now we suppose that an error occurs in the third residue changing the result from 7 to 8 (i.e. the error is $(0, 0, 1)$). The erroneous result is $(10, 11, 8)$, that corresponds to the integer value 2078, that is not divisible by 256. Computing the values

of $\bar{e}_i$ from the value $< E >_{256}= 30$ we obtain $\bar{e}_1 =< 30 \cdot 235 >_{256}= 138$, $\bar{e}_2 =< 30 \cdot 45 >_{256}= 70$, $\bar{e}_3 =< 30 \cdot 111 >_{256}= 2$. e_1 and e_2 are out of the range $[-m_i + 1, m_i - 1]$ and therefore do not represent a possible candidate value.

However, the candidate values for E_i are: $E_1 = 138 \cdot 195 = 26910$, $E_2 = 70 \cdot 165 = 11550$ and $E_3 = 2 \cdot 143 = 286$. The value of $< \bar{Y} - E_i >_M$ are:

$$< \bar{Y} - E_1 >_M =< 2078 - 26910 >_M = 908$$

$$< \bar{Y} - E_2 >_M =< 2078 - 11550 >_M = 1253$$

$$< \bar{Y} - E_3 >_M =< 2078 - 286 >_M = 1792$$

$< \bar{Y} - E_3 >_M$ identify the error in the third digit, and the correct value is 1792, that is the value corresponding to the SRNS value of 7.

B. FIR filter with error correction capability

Now we present the architecture of a FIR filter that use scaled RNS representation to achieve error correction. Like in the case of error detection we need some additional hardware resources. For the case of a SRNS representation with a moduli set of i elements we have:

1) a constant multiplier before the conversion of the input sequence into the RNS representation,
2) a constant divider after the conversion of the output sequence from the RNS representation to the binary one
3) a modulo m reduction block to obtain $< E >_m$
4) i modulo m constant multiplier to obtain $\bar{e}_i$
5) i constant multiplier to obtain the different candidate value of E_i
6) i comparator between $\bar{e}_i$ and m_i
7) i modulo M adder to compute the candidate correct value
8) i modulo m reduction block to select the correct value
9) a mux selecting the right E_i value

The architecture of the FIR filter with error correction capabilities is shown in Fig. 4

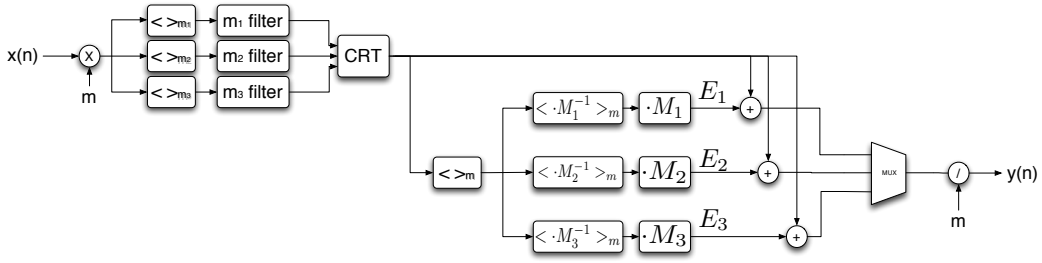

Fig. 4. SRNS implementation of a FIR filter with error correction capability

Also in this case the overhead of this schema is strictly dependent by the choice of the scaling factor m and the use of $m = 2^n$ allows to reduce the resource occupation of the additional blocks.

V. Implementation of Scaled RNS Filters

In this section some FIR filter has been implemented to carry out a comparison between our technique and the standard RRNS one.

The dynamic range of a filter depends on the dynamic range of coefficient, input data and number of taps, therefore the choice of moduli sets is dependent by these parameters. We choose two cases, one with 8 bits input data and coefficients and another with 12 bits input data and coefficients. The number of taps used to implement the filters vary from 32 to 128 taps and we restricted our analysis to constant coefficient filters. The implementation has been carried out on the Xilinx Virtex V FPGA and the moduli has been chosen to satisfy the relationship $m_i \leq 2^6$. In table V the different parameters the dynamic range of the filter and the chosen moduli set are presented. The moduli set has been defined for the RNS representation, without error detection and correction capability, for the standard RRNS and for the proposed scaled RNS.

For dynamic ranges up to 23 bits the unredundant representation use 4 moduli, the RRNS use six moduli, while the SRNS use six or seven moduli depending on the dynamic range of the filter. For the dynamic range between 29 and 31 both the RRNS and the SRNS require 8 moduli. The dynamic range of the SRNS is computed as $M_s = \lfloor log_2(\prod_{i=1}^{n} m_i/m) \rfloor$, where $m = 2^{12}$ is chosen to satisfy equation (11). The table shows how for most of the example filter the number of moduli of the RRNS is the same of the SRNS representation. The resource usages of the filters taken into account are summarized in table

Name	input/coefficient length	number of taps	dynamic range	unredundant moduli set	RRNS moduli set	SRNS moduli set
FIR1	8	32	21	61,59,53,47	64,63,61,59,53,47	63,61,59,53,47,43
FIR2	8	64	22	61,59,53,47	64,63,61,59,53,47	63,61,59,53,47,43
FIR3	8	128	23	61,59,53,47	64,63,61,59,53,47	63,61,59,53,47,43,41
FIR4	12	32	29	61,59,53,47,43,41	64,63,61,59,53,47,43,41	63,61,59,53,47,43,41,37
FIR5	12	64	30	61,59,53,47,43,41	64,63,61,59,53,47,43,41	63,61,59,53,47,43,41,37
FIR6	12	128	31	61,59,53,47,43,41	64,63,61,59,53,47,43,41	63,61,59,53,47,43,41,37

TABLE V

PARAMETERS AND DYNAMIC RANGE OF THE EXAMPLES FILTERS

VI both for RRNS and for SRNS. In the last column of table VI the ratio between resource occupation of RRNS and SRNS is given. The resource usage has been divided in three parts:

1) resource occupation for the forward converters (modulo m_i reduction)
2) resource occupation for all the modulo m_i filters
3) resource occupation for the CRT and the error correction blocks

	RRNS				SRNS				ratio between
Filter Name	forward (#LUTs)	m_i filters (#LUTs)	CRT and error correction (#LUTs)	total usage (#LUTs)	forward (#LUTs)	m_i filters (#LUTs)	CRT and error correction (#LUTs)	total usage (#LUTs)	RRNS and SRNS
FIR1	70	1760	1344	**3174**	84	1920	650	**2654**	83%
FIR2	70	3520	1344	**4934**	84	3840	650	**4574**	92%
FIR3	70	7040	1344	**8454**	98	8960	737	**9795**	115%
FIR4	98	2400	2472	**4970**	112	2560	853	**3525**	71%
FIR5	98	4800	2472	**7370**	112	5120	853	**6085**	82%
FIR6	98	9600	2472	**12170**	112	10240	853	**11205**	92%

TABLE VI

RESOURCE OCCUPATION OF EXAMPLES FILTERS

It can be noticed that, when the area overhead due to the error correction of RRNS a significant part of the resource occupation, the use of the SRNS allows to reduce this overhead up to 30% with respect to the RRNS method. Instead, when the number of moduli used for SRNS is greater than the one used for RRNS the RRNS is more convenient than SRNS.

VI. CONCLUSIONS

In this paper a novel error detection and correction technique for RNS representation is proposed. It is based on the use of a subset of the RNS legitimate range in which each element of the subset is a multiple of a number m with the constrain that $m > m_i$, where m_i are the elements of the set of moduli used in the RNS representation. This method allows to detect and correct any single error in the modular processors of the RNS based computational unit. The paper also shows how this method is suitable for the design of FIR filters with error detection and correction capabilities. Comparing the proposed technique with respect to the traditional one the use of scaled RNS allows reducing overhead avoiding the use of different CRT modules for error correction.

REFERENCES

[1] W. W. Peterson, "On Checking an Adder", I.B.M. J. Res. Develop., vol 2, pp. 166-168, Apr 1958.
[2] D. Nikolos, A.M. Paschalis, G. Philokyprou, "Efficient Design of Totally Self-Checking Checkers for All Low-cost Arithmetic Codes", IEEE Transactions on Computers, vol. 37, N. 7, pp. 807 - 814, July 1988.
[3] M. Nicolaidis, "Carry Checking/Parity Prediction Adders and ALUs", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 11, N. 1, Feb 2003 pp. 121-128.
[4] J.-C. Lo, S. Thanawastien, T. R. N. Rao, M. Nicolaidis, "An SFS Berger Check Prediction ALU and its Application to Self-Checking Processors Design", IEEE Transactions on Computer Aided Design, pp. 525-540, Mar. 1992.
[5] S. Bandyopadhyay, G.A. Jullien, A. Sengupta, "A Systolic Array for Fault Tolerant Digital Signal Processing using a Residue Number System Approach" , Proceedings of the International Conference on Systolic Arrays, 25-27 May 1988, Page(s):577 - 586
[6] Mark H. Etzel and W. K. Jenkins, "Redundant Residue Number Systems for Error Detection and Correction in Digital Filters", IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASS-28, No 5, pp. 538-544, October 1980.
[7] W. K. Jenkins, "The Design of Error Checkers for Self-Checking Residue Number Arithmetic", IEEE Transactions on Computers, Volume C-32, Issue 4, Apr 1983 Page(s):388 - 396
[8] I. Vinogradov, "An Introduction to the Theory of Numbers", New York: Pergamon Press, 1955.
[9] N. Szabo and R. Tanaka, "Residue Arithmetic and its Applications in Computer Technology", New York: McGraw-Hill, 1967.
[10] M. Sodestrand, W. Jenkins, G. A. Jullien, and F. J. Taylor, "Residue Number System Arithmetic: Modern Applications in Digital Signal Processing", New York: IEEE Press, 1986.
[11] T. V. Vu, "Efficient Implementation of the Chinese Remainder Theorem for Sign Detection and Residue Decoding", IEEE Transactions Circuits Systems-I, vol. 45, pp. 667-669, June 1985.
[12] S.Piestrak, "A High-Speed Realization of a Residue to Binary Number System Converter", IEEE Transactions Circuits Systems-II Analog and Digital Signal Processing, vol. 42, pp. 661-663, Oct. 1995.
[13] G. Cardarilli, M. Re, and R. Lojacono, "A Residue to Binary Conversion Algorithm for Signed Numbers", European Conference on Circuit Theory and Design (ECCTD97), vol. 3, pp. 1456-1459, 1997.
[14] F. Barsi and P. Maestrini, "Error Correcting Properties of Redundant Residue Number Systems", IEEE Transactions Compututer, vol. C-22, pp. 307-315, Mar. 1973.