Reverse conversion architectures for signed-digit residue number systems

Andreas Persson and Lars Bengtsson

N.B.: When citing this work, cite the original article.

©2006 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.


DOI: http://dx.doi.org/10.1109/ISCAS.2006.1693181
Copyright: IEEE

Post-Print available at: Halmstad University DiVA
http://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-2105
Abstract - This paper presents circuits for conversion from radix-2 Signed-Digit Residue Numbers to binary form. Four reverse converters for combined RNS/SD number systems based on different moduli sets are presented. Implementations are compared with respect to timing, area and Area-Delay products. Finite Impulse Response (FIR) filters are used as reference designs in order to evaluate the performance of RNS/SD processing in a typical DSP block using the suggested moduli sets.

1 Introduction

The Residue Number System (RNS) is an integer system capable of supporting high speed concurrent arithmetic. In RNS, an integer is decomposed into a set of residues with shorter binary representations, which can be processed independently and in parallel. The use of the Signed-Digit (SD) number system to represent the residues has been suggested as a way to eliminate carry propagation within RNS arithmetic circuits [1], [2]. The SD number system provides a redundant number representation that facilitates carry-free addition. The use of Signed-Digit representation also implies efficient modulo arithmetic, which helps to simplify crucial RNS operations.

The performance of residue number systems depends heavily on the choice of the moduli set and on the conversion from RNS to binary representation. Many moduli sets and reverse converters have been proposed for RNS systems with residues on 2’s complement form [3], [4], [5], [6]. In this paper, we investigate moduli sets and converters for use with Signed-Digit Residue Number Systems (RNS/SD).

2 Background

2.1 The Residue Number System

The basis of an RNS is a set of pairwise prime integers \(S = \{m_1, m_2, ..., m_L\} \), where \(\gcd(m_i, m_j) = 1 \) for \(i \neq j\). The set \(S\) is the moduli set and the dynamic range of the system is \([0, M)\), where \(M\) is the product of all moduli \(m_i\) in \(S\). Any integer \(X\) within the dynamic range has a unique RNS representation given by an ordered set of residues

\[
X \rightarrow \{x_1, x_2, ..., x_L\}, \quad x_i = [X]_{m_i}, \quad i = 1, 2, ..., L,
\]

where \([X]_{m_i}\) denotes \(X \mod m_i\).

RNS is a non-weighted number system. If integers \(A\) and \(B\) have RNS representations \(\{a_1, a_2, ..., a_L\}\) and \(\{b_1, b_2, ..., b_L\}\) respectively, then the RNS representation of \(C = A+B\) is \(\{c_1, c_2, ..., c_L\}\), where \(c_i = \left[a_i + b_i\right]_{m_i}\) and \(\oplus\) denotes addition, subtraction or multiplication. Since \(c_i\) only depends upon \(a_i\), \(b_i\) and \(m_i\), each \(c_i\) is computed using its own arithmetic unit, often called a channel.

The reconstruction of \(X\) from \(\{x_1, x_2, ..., x_L\}\) is based on the Chinese Remainder Theorem (CRT)

\[
X = \left[\sum_{i=1}^{L} \frac{M_i}{m_i} \cdot x_i \mod M\right],
\]

where \(M = \prod_{i=1}^{L} m_i\), \(M_i = M / m_i\) and \((\hat{x}_i)_{m_i}\) is the multiplicative inverse of \(M_i\) modulo \(m_i\), such that \((\hat{x}_i \cdot \hat{x}_i)^{-1}\)\(\mod m_i\) = \(1\).

2.2 The Signed-Digit Number System

The radix-2 Signed-Digit number system has the digit-set \(\{1, 0, 1\}\), where \(1\) denotes \(-1\). This yields a redundant number representation. For example, \("6"\) can be represented as \[0110\]_{SD}, \[1110\]_{SD}, or \[1010\]_{SD}. Zero is, however, uniquely represented.

To represent an SD digit, \(y\), two binary bits, \(y^+\) and \(y^-\) are required. That is, \(y = [y^+ y^-]\). The value of an \(n\) -digit SD integer \(Y = [y_{n-1} ... y_0]_{SD}\) is

\[
\sum_{i=0}^{n-1} y_i \times 2^i - \sum_{i=0}^{n-1} y_i \times 2^i.
\]

Note that it is possible to represent any integer and its negation with an equal number of digits (unlike the 2’s complement number system). The negation of an integer is a very simple operation in the SD number system. The negation of \(y = [y^+ y^-]\) is \((-y) = [y^- y^+]\), as can be seen by negating Eq. (2). No logic gates are required for this operation.

By exploiting the redundancy of the Signed Digit number representation, carry propagation is limited to one bit position when adding SD numbers [1], [2]. Consequently, addition is performed in constant time, regardless of operand widths.
3 Moduli Selection in RNS/SD

An important consideration when designing residue number systems is the choice of the moduli set $S = \{m_1, m_2, \ldots, m_k\}$. Sets with moduli of the forms $2^n$, $2^n - 1$ and $2^n + 1$, known as low-cost moduli, are of special interest. Such sets facilitate the use of simplified arithmetic units.

The properties of the SD number system, presented in Section 2.2, helps to further simplify modulo arithmetic for the low-cost moduli. Addition modulo $2^n \pm 1$ is performed using SD adders with end-around-carry logic [1]. Because of the limited carry propagation in SD adders, there is no delay penalty for the end-around-carry operation. Unlike the 2’s complement number system, the result of modulo $2^n + 1$ addition is represented using $n$ SD-digits, since results for sums greater than $2^n - 1$ is taken from the negative range.

Modulo multiplication by powers of two relies on shift operations, according to the rules in Eq. 3, where $x = [x_{n-1} \ldots x_0]_{SD}$ is an $n$-digit SD integer. These operations are accomplished by wiring connections appropriately:

$$\begin{align*}
2^n x_{n-1} &= [x_{n-1} \ldots x_0 0 \ldots 0]_{SD} \\
2^n x_{n-1} &= [x_{n-1} \ldots x_0 0 \ldots 0]_{SD} \\
2^n x_{n+1} &= [x_{n-1} \ldots x_0 (-x_{n-1}) \ldots (-x_{n-1})]_{SD}
\end{align*}$$

Eq. Set 3 Rules for multiplication by powers of two.

Five moduli sets, all consisting of low cost moduli have been selected for evaluation in this paper.  

$S_1 = \{2^n - 1, 2^n + 1\}  
\quad S_2 = \{2^n - 1, 2^n, 2^n + 1\}  
\quad S_3 = \{2^n - 1, 2^n, 2^n + 1, 2^{2n} + 1\}  
\quad S_4 = \{2^{n-1}, 2^{n-1} - 1, 2^n - 1, 2^n + 1\}, n \text{ even integer}  
\quad S_5 = \{2^n, 2^n - 1, 2^n - 1, 2^n + 1\}, n \text{ odd integer}

Moduli-set-specific decoders for Signed-Digit residue numbers based on sets $S_1 \ldots S_5$ are presented in Section 4.

4 RNS/SD Decoders

4.1 Decoder for moduli set $S_1$

The proposed architecture for decoding SD residues with respect to the $S_1 = \{m_1, m_2\} = \{2^n - 1, 2^n + 1\}$ moduli set is based on the CRT procedure presented in Section 2.1. The two multiplicative inverses required for evaluation of the CRT in eq (1) are both powers of two:

$$\begin{align*}
\hat{M}_1 &= 2^n + 1, \quad [\hat{M}_1^{-1}]_{m_1} = 2^n - 1, \\
\hat{M}_2 &= 2^n - 1, \quad [\hat{M}_2^{-1}]_{m_2} = 2^n - 1.
\end{align*}$$

Eq. (1) is reduced to

$$X = \left[(2^n + 1)2^{n-1} x_1 + (2^n - 1)2^{n-1} x_2\right]_{2^n - 1}. \quad (4)$$

Since the SD representations of $x_1$ and $x_2$ both have digit-length $n$, Eq. (4) can be computed as the sum of two $n$-digit SD vectors, formed by concatenation and rotation according to the rules in Eq. Set 3

$$X = [A + B]_{2^n - 1}.  
A = \left[(2^{2^n - 1} + 2^{n-1})x_{12} \ldots x_{1n} \right]_{SD} = [x_1 x_1 \ldots x_1 x_1]_{SD}  
B = \left[(2^{2^n - 1} - 2^{n-1})x_{22} \ldots x_{2n} \right]_{SD} = [x_2 x_2 \ldots x_2 x_2]_{SD}
$$

were $x_i$ denotes $(-x_i)$.

$X$ will be in the range $(-M, M)$, due to the fact that SD modulo adders use the negative range as well as the positive. If the output is required to be in the range $[0, M]$, the correct result is obtained by adding $M = [1_{2n-1} \ldots 1]$ to $X$ when $X$ is negative. Adding constant ones to an SD integer is a simple operation, as shown in [1]. Carry-look-ahead (CLA) adders are used to obtain the binary representation of $X$ according to Eq. (2). Both $X$ and $X + M$ are decoded to binary form, using two CLAs operating in parallel. The correct value is selected by examining the carry-out bit of the adder for $X$. The proposed decoder is depicted in Figure 1.

4.2 Decoders for moduli set $S_2$ and $S_3$

The sets $S_2$ and $S_3$ both have moduli products of the form $2^n(2^n - 1)$, where $a = 2n$ for $S_2$ and $a = 4n$ for $S_3$. The decoding of binary-residue number systems based on these sets have previously been studied, for example in [5,6]. The proposed RNSD/SD decoders for these moduli sets are based on the New CRT-1, detailed in [7]

$$X = \hat{a} \sum_{i=0}^{n-1} x_i 2^i, \quad X' = \left[k_1(x_2 - x_1) + k_2 m_2(x_3 - x_2) + \ldots \right. \\
\left. + k_{L-1} m_2 \ldots m_L (x_L - x_{L-1}) \right]_{m_1 \ldots m_L}, \quad (5)$$

where

$$\begin{align*}
[k_1 m_1 m_2 \ldots m_L] &= 1, \\
[k_2 m_1 m_2 \ldots m_L] &= 1, \\
\ldots \\
[k_{L-1} m_1 m_2 \ldots m_L] &= 1.
\end{align*}$$

Eq. (5) is accomplished by adding three $2n$ -digit SD vectors formed by concatenation and rotation.

$$\begin{align*}
M_1 &= 2^n + 1, \quad [M_1^{-1}]_{m_1} = 2^n - 1, \\
M_2 &= 2^n - 1, \quad [M_2^{-1}]_{m_2} = 2^n - 1.
\end{align*}$$

For set $S_2$, let $m_1 = 2^n$, $m_2 = 2^n + 1$ and $m_3 = 2^n - 1$. The multiplicative inverses $k_1$ and $k_2$ from Eq. (5) are

$$\begin{align*}
k_1 &= 2^n, \quad k_2 = 2^{n-1}.
\end{align*}$$

The computation of $X'$ in Eq. (5) is accomplished by adding three $2n$-digit SD vectors formed by concatenation and rotation.
Theorem 1: Let $X = [A + B + C]_{2^{n+1}}$, where

$$A = -2x_1x_2 \cdots x_{n-1},$$

$$B = (2^{2n-1} + 2^{n-1})x_2^2 \cdots x_{n-1},$$

$$C = (2^{2n-1} + 2^{n-1})x_2^2 \cdots x_{n-1}.$$

Similarly, rearranging the elements of $S_1$, such that

$$m_1 = 2^n, m_2 = 2^{n+1}, m_3 = 2^n + 1$$

and $m_4 = 2^n - 1$, yields attractive multiplicative inverses

$$k_1 = 2^{3n}, k_2 = 2^{n-1}, k_3 = 2^{n-2}.$$

For the $S_3$ moduli set, $X'$ in eq (5) is the result of the addition of four $4n$-digit SD vectors

$$X' = [A + B + C + D]_{2^{4n}}.$$

4.3 Decoders for moduli sets $S_4$ and $S_5$

The two four-moduli sets

$$S_4 = \{2^{n-1} - 1, 2^n - 1, 2^n + 1\}, \text{ even integer}$$

$$S_5 = \{2^n, 2^n - 1, 2^n + 1\}, \text{ odd integer}$$

can be expressed on a common form as

$$\{m_1, m_2, m_3, m_4\} = \{2^n, 2^n - 1, 2^n + 1\},$$

where $a = n - 1, b = n$ for $S_4$ and $a = n, b = n - 1$ for $S_5$.

The proposed decoders for these sets are SD implementations of the two-level approach to RNS decoding, detailed in [8]. One of the first-level decoders combines residues $X_1 | m_1$ and $X_3 | m_3$ to form a binary residue $X_1 | m_1, m_3$ according to Eq. (5) in Section 4.2. The other first-level decoder combines $X_3 | m_3$ and $X_4 | m_4$ to form $X_3 | m_3, m_4$. This is the CRT decoder from Section 4.1. The second level is a converter for the binary-residue number system with moduli set

$$\{m_1, m_2, m_3, m_4\} = \{2^n, 2^n - 1, 2^n + 1\},$$

as described in [8].

In the last conversion step, a modulo($2^n + 1$) adder/scaler is required in order to scale $X_1$ and $X_3$ with a precalculated value of the multiplicative inverse $[m_1, m_2] | m_3, m_4$. The decoder for moduli set $S_5$ is depicted in Figure 4.

5 VLSI Implementation results

The presented decoders have been implemented in VHDL and synthesized using Synopsys Design Compiler and a UMC 0.13 μm standard CMOS cell library with 8 metal layers and a core voltage of 1.2 Volts.

In order to evaluate the performance of RNS processing using the presented moduli sets, RNS/SD Finite Impulse Response (FIR) filters have been implemented. For an RNS with moduli set $S = \{m_1, m_2, \ldots, m_t\}$, the corresponding FIR filter is decomposed into $L$ transposed form subfilters working in parallel. The subfilters have eight filter taps, each consisting of a register, an SD modulo multiplier and an SD modulo adder.
When implementing the FIR filters, pipeline stages were added to the RNS/SD converters to maintain a clock cycle that is determined by the critical path of the filter taps. This introduces an additional latency of two clock cycles for the filters with moduli sets $S_1, ..., S_3$ and four clock cycles for filters with moduli sets $S_4$ and $S_5$.

The designs were compiled for typical operating and wire load conditions and synthesised for three different equivalent (binary) wordlengths (16, 32 and 48 bits). For each moduli set, the parameter $n$ was chosen so that the resulting moduli product was as small as possible, but at least equal to the desired dynamic range. The moduli sets used for VLSI implementation are presented in Table I. Table II presents the synthesis results.

As seen in Table II, the decoder for set $S_1$ has the smallest area and, for the 16-bit dynamic range, also the shortest delay. For larger dynamic ranges, the decoders for $S_1$ has the shortest delay, as a result of shorter carry chains in the CLA adders. The decoders for $S_4/S_5$ have significantly longer delay and larger area, due to the constant multiplier in the second stage of the decoder.

When examining the synthesis results for the RNS/SD FIR filters, we see that the filter using modulo set $S_1$ suffers from the large wordlength of the residues. The unbalanced residue wordlengths in the filter using moduli set $S_2$ is also causing degraded performance. The moduli sets $S_4$ and $S_5$ are well balanced and the use of these sets result in shorter delays. The RNS/SD FIR filter with moduli set $S_4/S_5$ shows the best Area-Delay products.

6 Conclusions

This article presents architectures for Signed-Digit Residue Numbers System decoding. Four decoders for RNS/SD systems with different moduli sets have been implemented and comparisons have been made with respect to area and timing. It is shown that, out of the presented moduli sets, $S_1$ and $S_2$ result in reverse converters with the lowest A*D products. However, considering a typical DSP block (FIR filter), the four-moduli sets $S_2$ and $S_3$ result in RNS/SD filters with the best A*D product (at the cost of a higher latency penalty).

References

---

**Table I** Moduli sets used for VLSI implementation.

<table>
<thead>
<tr>
<th>Moduli Set</th>
<th>16 bits</th>
<th>32 bits</th>
<th>48 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n</td>
<td>Value</td>
<td>n</td>
</tr>
<tr>
<td>$S_1$</td>
<td>8</td>
<td>{255, 257}</td>
<td>16</td>
</tr>
<tr>
<td>$S_2$</td>
<td>6</td>
<td>{63, 64, 65}</td>
<td>11</td>
</tr>
<tr>
<td>$S_3$</td>
<td>4</td>
<td>{15, 16, 17, 257}</td>
<td>7</td>
</tr>
<tr>
<td>$S_4/S_5$</td>
<td>5</td>
<td>{32, 31, 15, 17}</td>
<td>9</td>
</tr>
</tbody>
</table>

**Table II** VLSI implementation results.

<table>
<thead>
<tr>
<th>Moduli Set</th>
<th>RNS/SD Decoders</th>
<th>8-tap Transposed Form RNS/SD FIR Filters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay [ns]</td>
<td>Area [mm²]</td>
</tr>
<tr>
<td></td>
<td>16 bits</td>
<td>32 bits</td>
</tr>
<tr>
<td>$S_1$</td>
<td>3.54</td>
<td>5.57</td>
</tr>
<tr>
<td>$S_2$</td>
<td>3.57</td>
<td>4.98</td>
</tr>
<tr>
<td>$S_3$</td>
<td>4.44</td>
<td>5.90</td>
</tr>
<tr>
<td>$S_4/S_5$</td>
<td>7.95</td>
<td>11.62</td>
</tr>
</tbody>
</table>