An Efficient RNS to Binary Converter Using the Moduli Set $\{2n+1, 2n, 2n-1\}$

Kazeem Alagbe Gbolagade[1,2], Member, IEEE and Sorin Dan Cotofana[1], Senior Member IEEE,
1. Computer Engineering Laboratory, Delft University of Technology,
The Netherlands. E-mail: {gbolagade,sorin}@ce.et.tudelft.nl
2. University for Development Studies, Navrongo, Ghana.

Abstract*—In this paper, we investigate Residue Number System (RNS) to decimal conversion which is an important issue concerning the utilization of RNS numbers in Digital Signal Processing (DSP) applications. We propose a reverse converter using the moduli set $\{2n+1, 2n, 2n-1\}$. First, we show that this converter does not require the computation of multiplicative inverses. Next, we simplify the Chinese Remainder Theorem (CRT) to obtain a low complexity implementation, which does not require explicit use of modulo operation in the conversion process as it is normally the case in the traditional CRT. At the algorithm level the conversion process requires 3 or 4 additions and 4 multiplications thus theoretically speaking its expected performance is similar to that of other state of the art equivalent converters. A critical path analysis however, which takes into consideration hardware implementation details, indicates that the proposed converter outperforms other state of the art converters in terms of speed at the expense of same or lower area.

***Index Terms*—Reverse Conversion, Residue Number System, Multiplicative Inverses, Chinese Remainder Theorem.**

I. INTRODUCTION

Residue Number System (RNS) is an unweighted number system with inherent parallel characteristics, which supports carry-free addition, borrow-free subtraction, and single step multiplication without partial products [8], [2], [3]. While RNS utilization in highly intensive computation has received considerable attention in the past [8], [3] it has not found widespread use in general-purpose signal processor architecture mainly because operations like magnitude comparison, overflow detection, sign detection and division are rather difficult to perform. However, where these operations are not frequently needed, special-purpose Digital Signal Processing (DSP) architectures based on RNS have been proposed [1].

One of the issues that precludes the large scale RNS utilization in computing technology is data conversion. As RNS based functional units have to operate into a binary/decimal environment the input operands provided in either standard binary or decimal format need to be converted to RNS before performing any operation. In a similar way the final results must be represented in the same way as the input operands thus RNS to binary/decimal conversion is also required. This implies that RNS based processors make heavily use of data conversions, which is a slow process. For an RNS processor to compete favourably with a conventional processor those conversions have to be fast such that the RNS speedup is not nullified by the conversion overhead. Up to date several data conversion techniques have been proposed based on either the traditional Chinese Remainder Theorem (CRT) [10], [7], [12], [9] or Mixed Radix Conversion (MRC) [3], [13], which may or may not rely on look up tables. In [11], a CRT that reduced the modulo M was presented. RNS to decimal converters which does not require explicit modulo operations have been presented in [7], [5], [12], [6]. Based on the weight concepts, the decoding of RNS numbers using the moduli sets $\{2n+1, 2n, 2n-1\}$ and $\{2n+2, 2n+1, 2n\}$ have been presented in [7] and [5]. In [12], smaller multipliers were obtained for these same moduli sets and errors in [5] were also pointed out. Also, the corrections to [5] has been presented in [6].

In this paper we present an efficient RNS to decimal converter tailored for the $\{2n+1, 2n, 2n-1\}$ moduli set. It does not involve the computation of the multiplicative inverses and it is a simplification, based on the scheme proposed in [11], of the traditional CRT. We demonstrate that the proposed converter operates on smaller numbers than other state of the art converters for the the same moduli set thus it requires less complex adders and multipliers. From the algorithmic point of view the conversion process requires 3 or 4 additions and 4 multiplications thus its expected performance is similar to that of other state of the art equivalent converters. We also performed a deeper analysis of the required hardware in order to get a more realistic picture of the required components. The critical path analysis, which takes into consideration hardware implementation details, indicates that the proposed converter outperforms other state of the art converters in terms of speed at the expense of same or lower area.

The rest of the article is organised as follows: Section II presents the necessary background. In Section III we describe the proposed algorithm. Section IV presents the hardware realization of the proposed algorithm and a comparison with the state of the art, while the paper is concluded in Section V.

II. Background

RNS is defined in terms of a set of relatively prime moduli set $\{m_i\}_{i=1,n}$ such that $\gcd(m_i, m_j) = 1$ for $i \neq j$, where gcd means the greatest common divisor of m_i and m_j, while $M = \prod_{i=1}^{n} m_i$, is the dynamic range. The residues of a decimal number X can be obtained as $x_i = |X|_{m_i}$ thus X can be represented in RNS as $X = (x_1, x_2, x_3..., x_n)$, $0 \leq x_i < m_i$. This representation is unique for any integer $X \in [0, M-1]$. We note here that in this paper we use $|X|_{m_i}$ to denote the X mod m_i operation and the operator Θ to represent the operation of addition, subtraction, or multiplication. Given any two integer numbers K and L RNS represented by $K = (k_1, k_2, k_3, ..., k_n)$ and $L = (l_1, l_2, l_3, ..., l_n)$, respectively, $W = K \Theta L$, can be calculated as $W = (w_1, w_2, w_3, ..., w_n)$, where $w_i = |k_i \Theta l_i|_{m_i}$, for $i = 1, n$. This means that the complexity of the calculation of the Θ operation is determined by the number of bits required to represent the residues and not by the one required to represent the input operands. This creates the premises for high speed arithmetic and for example it has been proved that RNS based addition can be performed in $O(\log(\log(n)))$ delay for unrestricted moduli and in $O(\log(n))$ for 2^n and $2^n - 1$ moduli [4].

For a moduli set $\{m_i\}_{i=1,n}$ with the dynamic range $M = \prod_{i=1}^{n} m_i$, the residue number $(x_1, x_2, x_3, ..., x_n)$ can be converted into the decimal number X, according to the Chinese Reminder Theorem, as follows [8]:

$$X = \left| \sum_{i=1}^{n} M_i \left| M_i^{-1} x_i \right|_{m_i} \right|_M, \tag{1}$$

where $M = \prod_{i=1}^{n} m_i$, $M_i = \frac{M}{m_i}$, and M_i^{-1} is the multiplicative inverse of M_i with respect to m_i.

This general scheme can be actually simplified when certain moduli sets of interests like $\{2n+1, 2n, 2n-1\}$ are utilized. For this moduli set the following relations have been derived for $(x_1 + x_3)$ even and odd, respectively, in [7]:

$$X = \left| \frac{m_2 m_3}{2} x_1 - m_1 m_3 x_2 + \frac{m_1 m_2}{2} x_3 \right|_M \tag{2}$$

$$X = \left| \frac{M}{2} + \frac{m_2 m_3}{2} x_1 - m_1 m_3 x_2 + \frac{m_1 m_2}{2} x_3 \right|_M \tag{3}$$

For the same moduli set, the following relation, which uses smaller multipliers when compared to Equations (2) and (3) have been presented in [12]:

$$\begin{aligned} X &= x_2 + m_2 \left| \left\lfloor \frac{(x_1 - x_3) + 2z_0 n}{2} \right\rfloor \right. \\ &+ \left. 2n \left\lfloor \frac{(x_1 - 2x_2 + x_3) + 2z_0 n}{2} \right\rfloor \right|_{m_1 m_3}, \end{aligned} \tag{4}$$

where z_0 is the XOR over the least significant bits of x_1 and x_3.

In the following section we assume the same moduli sets $\{2n+1, 2n, 2n-1\}$ and we introduce an RNS to decimal converter based on Equation (1) by eliminating the computation of the required multiplicative inverses. By doing that we obtain exactly the same relations as Equations (2) and (3). Further we simplify these expressions such that we obtain relations that use smaller multipliers and require lesser number of arithmetic operations when compared to Equation (4).

III. Proposed Algorithm

Given the RNS number (x_1, x_2, x_3) with respect to the moduli set $\{m_1, m_2, m_3\}$ in the form $\{2n+1, 2n, 2n-1\}$, the proposed algorithm computes the decimal equivalent of this RNS number based on a further simplification of the well-known traditional CRT. First, we demonstrate that the computation of the multiplicative inverses can be eliminated for this moduli set. We further simplify the resulting CRT to get a low complexity implementation that does not require explicit use of modulo operation in the final stage of the computation.

Theorem 1: Given the moduli set $\{2n+1, 2n, 2n-1\}$ with $m_1 = 2n+1, m_2 = 2n, m_3 = 2n-1$ the following hold true:

$$|(m_1 m_2)^{-1}|_{m_3} = n, \tag{5}$$

$$|(m_2 m_3)^{-1}|_{m_1} = n+1, \tag{6}$$

$$|(m_1 m_3)^{-1}|_{m_2} = 2n-1. \tag{7}$$

Proof: If it can be demonstrated that $|n \times (m_1 m_2)|_{m_3} = 1$, then n is the multiplicative inverse of $(m_1 m_2)$ with respect to m_3. $|n \times (m_1 m_2)|_{m_3}$ is given by: $|(2n+1)(2n)(n)|_{2n-1} = |(4n^2 + 2n)n|_{2n-1} = ||4n^3|_{2n-1} + |2n^2|_{2n-1}|_{2n-1} = |\frac{1}{2} + \frac{1}{2}|_{2n-1} = 1$, thus Equation (5) holds true.

In the same way if $|(n+1) \times (m_2 m_3)|_{m_1} = 1$, then $n+1$ is the multiplicative inverse of $(m_2 m_3)$ with respect to m_1. $|(n+1)\times(m_2m_3)|_{m_1}$ is given by: $|2n(2n-1)(n+1)|_{2n+1} = |2n^2(2n+1)-2n|_{2n+1} = ||2n^2(2n+1)|_{2n+1} + |-2n|_{2n+1}|_{2n+1} = |0+1|_{2n+1} = 1$, thus Equation(6) holds true.

Again, if $|(2n-1)\times(m_1m_3)|_{m_2} = 1$, then $2n-1$ is the multiplicative inverse of (m_1m_3) with respect to m_2. $|(2n-1)\times(m_1m_3)|_{m_2}$ is given by:
$|(2n+1)(2n-1)(2n-1)|_{2n} = |8n^3-4n^2-2n+1|_{2n} = ||2n(4n^2-2n-1)|_{2n} + |1|_{2n}|_{2n} = |0+1|_{2n} = 1$, thus Equation (7) holds true. ■

The following theorem introduces a simplified way to compute the decimal equivalent of the RNS number (x_1, x_2, x_3) with respect to the moduli set $\{m_1, m_2, m_3\}$ in the form $\{2n+1, 2n, 2n-1\}$ by simplifying Equation (1) for the specific case $n = 3$.

Theorem 2: The decimal equivalent of the RNS number (x_1, x_2, x_3) with respect to the moduli set $\{m_1, m_2, m_3\}$ in the form $\{2n+1, 2n, 2n-1\}$ is computed for (x_1+x_3) even and odd, respectively, as follows:

$$X = \left|\frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M \tag{8}$$

$$X = \left|\frac{M}{2} + \frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M \tag{9}$$

Proof: Equation (1) for $n = 3$ is given by:

$$X = \left|\sum_{i=1}^{3} M_i \left|M_i^{-1}x_i\right|_{m_i}\right|_M . \tag{10}$$

By substituting the multiplicative inverse values in Theorem 1 into Equation (10) we obtain the following:

$$\begin{aligned} X &= \Big|(m_2m_3)(\frac{m_2}{2}+1)x_1 + (m_1m_3)(m_3)x_2 \\ &\quad +(m_1m_2)(\frac{m_2}{2}x_3)\Big|_M \\ &= \Big|(m_2m_3)(\frac{m_1+1}{2})x_1 + (m_1m_3)(m_2-1)x_2 \\ &\quad +(m_1m_2)(\frac{m_3+1}{2})x_3\Big|_M \\ &= \Big|(\frac{M}{2})x_1 + (\frac{m_2m_3}{2})x_1 - m_1m_3x_2 \\ &\quad +(\frac{M}{2})x_3 + (\frac{m_1m_2}{2})x_3\Big|_M \\ &= \Big|\Big|(x_1+x_3)\frac{M}{2}\Big|_M + \Big|(\frac{m_2m_3}{2})x_1\Big|_M \\ &\quad -|m_1m_3x_2|_M + \Big|(\frac{m_1m_2}{2})x_3\Big|_M\Big|_M \end{aligned} \tag{11}$$

From Equation (11), the following cases may be considered:

- If (x_1+x_3) is even then $|(x_1+x_3)\frac{M}{2}|_M = 0$.
- If (x_1+x_3) is odd then $|(x_1+x_3)\frac{M}{2}|_M = \frac{M}{2}$.

Given that Equation (11) can be re-written for the first and the second case, respectively, as:

$$X = \left|\frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M \tag{12}$$

$$X = \left|\frac{M}{2} + \frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_M \tag{13}$$

■

Equations (8), (9), and (11), are exactly the same as to ones derived based on a weighting function in [7] and [5]. Thus the hardware realization based on these equations is the same as the one presented in [7], [5] and [12].

We propose to further simplify (8) and (9) using the following theorem:

Theorem 3: Given the RNS number (x_1, x_2, x_3) with respect to the moduli set $\{m_1, m_2, m_3\}$ in the form $\{2n+1, 2n, 2n-1\}$, the decimal equivalent of this RNS number is computed for (x_1+x_3) even and odd, respectively, as follows:

$$X = -m_2x_1 + m_1\left|\frac{m_2}{2}(x_1+x_3) - m_3x_2\right|_{m_2m_3} \tag{14}$$

$$\begin{aligned} X &= -m_2x_1 \\ &\quad +m_1\left|\frac{m_2m_3}{2} + \frac{m_2}{2}(x_1+x_3) - m_3x_2\right|_{m_2m_3} \end{aligned} \tag{15}$$

Proof: To prove this theorem we use the following lemma presented in [11]:

$$|am_1|_{m_1m_2} = m_1\,|a|_{m_2} \tag{16}$$

From Equation (8), we have

$$\begin{aligned} X &= \left|\frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_{m_1m_2m_3} \\ &= \left|\frac{m_2}{2}(m_1-2)x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_{m_1m_2m_3} \\ &= x_1 + \left|\frac{m_1m_2}{2}(x_1+x_3) - m_1x_1 - m_1m_3x_2\right|_{m_1m_2m_3} \end{aligned} \tag{17}$$

Applying Equation (16) we obtain

$$X = x_1 + m_1\left|\frac{m_2}{2}(x_1+x_3) - x_1 - m_3x_2\right|_{m_2m_3}, \tag{18}$$

which can be further simplified as:

$$X = -m_2x_1 + m_1\left|\frac{m_2}{2}(x_1+x_3) - m_3x_2\right|_{m_2m_3} \tag{19}$$

thus Equation (14) holds true.

From Equation (9), we have

$$\begin{aligned} X &= \left| \frac{m_1m_2m_3}{2} + \frac{m_2m_3}{2}x_1 \right. \\ &\quad \left. -m_1m_3x_2 + \frac{m_1m_2}{2}x_3 \right|_{m_1m_2m_3} \\ &= \left| \frac{m_1m_2m_3}{2} + \frac{m_2}{2}(m_1-2)x_1 \right. \\ &\quad \left. -m_1m_3x_2 + \frac{m_1m_2}{2}x_3 \right|_{m_1m_2m_3} \\ &= x_1 + \left| \frac{m_1m_2m_3}{2} + \frac{m_1m_2}{2}(x_1+x_3) \right. \\ &\quad \left. -m_1x_1 - m_1m_3x_2 \right|_{m_1m_2m_3} \end{aligned} \tag{20}$$

Applying Equation (16) we obtain

$$\begin{aligned} X &= x_1 \\ &+ m_1 \left| \frac{m_2m_3}{2} + \frac{m_2}{2}(x_1+x_3) - x_1 - m_3x_2 \right|_{m_2m_3} \end{aligned} \tag{21}$$

which can be further simplified as

$$\begin{aligned} X &= -m_2x_1 + m_1 \left| \frac{m_2m_3}{2} \right. \\ &\quad \left. + \frac{m_2}{2}(x_1+x_3) - m_3x_2 \right|_{m_2m_3} \end{aligned} \tag{22}$$

thus Equation (15) holds also true. ■

IV. Performance Evaluation

The converter described in this paper is processing numbers with smaller magnitude than the one proposed in [7]. This can be easily deduced based on the fact that Equations (14) and (15) use modulo m_2m_3 instead of modulo $m_1m_2m_3$ as required by Equations (2) and (3). The same holds true also when we compare our proposal with the one in [12] due to the fact that for the considered moduli set the modulo (m_2m_3) in Equations (14) and (15) is always smaller than the modulo (m_1m_3) in Equation (4). More precisely the modulo used in [12] is with $2n-1$ larger than the one we utilize. Consequently, the numbers involved in Equations (14) and (15) are smaller than those in Equation (4). This results in a reduced complexity and delay of the associated hardware as the smaller the magnitude of the processed operands the less complex are the required functional units that support the calculations.

The number of arithmetic operations required for our proposal - Equations (14) and (15) - and for the one in [12] - Equation (4) - are summarized in Table I. As one can observe in the Table our proposal requires less additions but more multiplications. This information however is not sufficient to make a comparison as the delay and area of the involved adders and multipliers depend on operand magnitudes and on implementation specific details. Thus a deeper analysis of the required hardware has to be done in order to get a more realistic picture.

Operations	[12]	Proposed Algorithm
Additions	7	3/4
Multiplications	2	4
Reduced M	m_1m_3	m_2m_3

Table I
Performance Comparison

The hardware required for our proposal is depicted in Figure 1 and the one for the converter in [12] in Figure 2. Before we further analyze the two schemes we first prove that modulo (m_2m_3) operations in Equations (14) and (15) do not need to be explicitly implemented and this operation can be done by adder AB in Figure 1 with at most one corrective addition/subraction. For that we analyze all the possible cases as follows:

1) $(x_1+x_3)=0$ and $x_2=2n-1$. This results in the most negative value one may get. In this case Equation (14) reduces to $|-m_3x_2|_{m_2m_3}$. To perform the modulo (m_2m_3) operation we need to do corrective additions. Given that $m_2m_3+(-m_3x_2)=(2n)(2n-1)-((2n-1)(2n-1))=4n^2-2n-4n^2+4n-1=2n-1>0$, for any positive integer n, only one corrective addition with m_2m_3 is required to compute the modulo.
2) (x_1+x_3) is even and has the maximum possible value and x_2 is zero. This is the largest positive value one may get and Equation (14) reduces to $|\frac{m_2}{2}(x_1+x_3)|_{m_2m_3}$. Given that $m_2m_3-\frac{m_2}{2}(x_1+x_3)=4n^2-2n-(n)(2n+2n-2)=4n^2-2n-4n^2+2n=0$ the maximum sum in the modulo adder cannot exceed m_2m_3, thus no correction is required.
3) $(x_1+x_3)=1$ and $x_2=2n-1$. In this case Equation (15) reduces to $|\frac{m_2m_3}{2}+\frac{m_2}{2}-m_3x_2|_{m_2m_3}$. Given that in this case $\frac{m_2m_3}{2}+\frac{m_2}{2}-m_3x_2$ is always negative and that $m_2m_3+\frac{m_2m_3}{2}+\frac{m_2}{2}-m_3x_2=2n(2n-1)+(2n^2-n)+n+(4n^2-4n+1)=2n^2+2n-1>0$, only one corrective addition with m_2m_3 is required to compute the modulo.
4) (x_1+x_3) odd has the maximum, possible value and x_2 is zero. Equation (15) reduces to $|\frac{m_2m_3}{2}+\frac{m_2}{2}(x_1+x_3)|_{m_2m_3}$. Given that $2m_2m_3-(\frac{m_2m_3}{2}+\frac{m_2}{2}(x_1+x_3))=2(4n^2-2n)-(2n^2-n+4n^2-2n)=8n^2-4n-6n^2+3n=2n^2-n>0$ one corrective subtraction of m_2m_3 is required to compute the modulo.

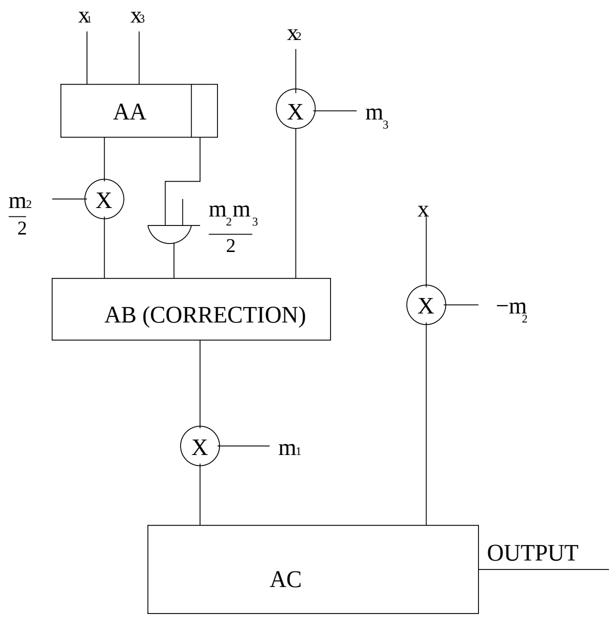

Figure 1. Proposed Data Path.

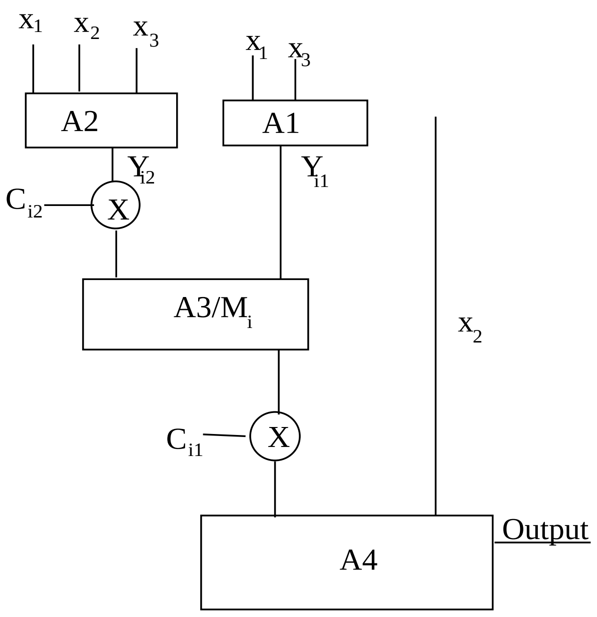

Figure 2. Converter Data Path from [12].

Thus we can conclude that the modulo m_2m_3 operation can be implemented with at most one corrective addition or subtraction. The same holds true also for the modulo m_1m_3 operation that is associated to the A3/Mi Adder in Figure 2.

Roughly speaking the performance of the designs in Figure 2 and in Figure 1 seems to be the same as both of them have three adders and two multipliers on the critical path. However those are not exactly standard components and their actual delays depend on the magnitude of the processed operands.

The Adder A2 in Figure 2 has three inputs thus it cannot be directly implemented with a Carry Propagate Adder (CPA). Actually, according to Equation (4) the term $2z_0n$ has to be also part of its input. Thus A2 has to be a 4:1 adder. To implement such an adder two levels of Carry Save Adders (CSAs) (two full adder delay) are required to reduce the 4 numbers to 2 numbers, followed by a CPA. Given that A2's result can be between $-(2n-1)$ and $3n-1$ the CPA has an additional delay in the order of $log(5n-1)$.

The Adder AA in Figure 1 is a standard CPA thus no CSA is required. Its result is at most $4n-2$ than it has a delay in the order of $log(4n-2)$, thus it is substantially faster than A2.

The next element on the critical path is a multiplier. Its complexity is determined by the magnitude of its inputs and by implication its output value. In Figure 2 it produces a maximum result of $6n^2-2n$ while in Figure 1 it produces a maximum result of $4n^2-2n$, thus the last one is smaller and faster than the first one.

Further the modulo addition has to be performed. This is done in both cases by computing first a tentative result. This result is subsequently compared with m_1m_3 or m_2m_3, respectively, and based on this and on the XOR of the least significant bits of x_1 and x_3, one corrective addition/subtraction is executed on the same adder. Given that m_1m_3 is larger with $2n-1$ than m_2m_3 the modulo adder A3/Mi in Figure 2 is in principle larger and slower than the Adder AB in our proposal. However the Adder AB as depicted in Figure 1 has an extra input and this my change that as it has to perform a 3:1 addition thus an extra CSA is required. In the following we prove that the conditional addition of the $\frac{m_2m_3}{2}$ term can be actually postponed and embedded into the correction step required for the modulo operation without any delay overhead. For that we remove $\frac{m_2m_3}{2}$ as input and revisit the 4 correction cases analyzed in the beginning of this section.

If x_1+x_3 is even the term $\frac{m_2m_3}{2}$ is not part of the calculation and the correction can be done as usual. If x_1+x_3 is odd the tentative sum in the modulo adder is $\frac{m_2}{2}(x_1+x_3)-m_3x_2$ instead of $\frac{m_2m_3}{2}+\frac{m_2}{2}(x_1+x_3)-m_3x_2$ thus it is smaller with $\frac{m_2m_3}{2}$ then it should actually be. Taking that into consideration the correction rules change to:

- If tentative sum is smaller than $-\frac{m_2m_3}{2}$ add $\frac{3m_2m_3}{2}$;
- If tentative sum is larger than $\frac{m_2m_3}{2}$ subtract $\frac{m_2m_3}{2}$;
- Otherwise add $\frac{m_2m_3}{2}$.

In this way the extra input is not contributing any longer to the tentative sum calculation and the AB Adder in Figure 1 outperforms its counterpart in Figure 2 in terms of delay and area.

The last elements on the critical path, a multiplier and an adder, have similar delays in both designs as they deal with operands within the same order of magnitude. Thus our proposal is certainly faster than the one in [12].

V. Conclusions

In this paper, we investigated RNS to decimal conversion, which is an important issue concerning the utilization of RNS numbers in DSP applications. We presented a new algorithm for RNS to decimal conversion in the moduli set $\{2n+1, 2n, 2n-1\}$. First, we eliminated the computation of multiplicative inverses. Next, we further simplified the resulting CRT to obtain a low complexity implementation which does not require the explicit use of modulo operation in the conversion process as it is normally the case in the traditional CRT. The proposed converter operates on smaller numbers than other state of the art converters for the the same moduli set thus it requires less complex adders and multipliers. We demonstrated that from the algorithmic point of view the conversion process requires 3 or 4 additions and 4 multiplications thus its expected performance is similar to that of other state of the art equivalent converters. We also did a critical path analysis, which indicated that the proposed converter outperforms other state of the art converters in terms of speed at the expense of similar or lower area cost.

References

[1] W.K. Jenkins and B.J. Leon. The use of residue number systems in the design of finite impulse response digital filters. *IEEE Trans. On circuitry and systems, vol. 24, pp. 191-200*, 1977.

[2] Mi Lu. *Arithmetic and Logic in Computer Systems*. John Wiley and Sons, New Jersey, 2004.

[3] G.A. Jullien M.A. Soderstrand, W.K. Jenkins and F.J. Taylor. *Residue Number Arithmetic: Modern Application in Digital Signal Processing*. IEEE press, New York, 1986.

[4] B. Parhami. *Arithmetic Algorithms and Hardware Designs*. Oxford University Press, New York, 2000.

[5] A. B. Premkuma. An rns to binary converter in a three moduli set with common factors. *IEEE Trans. on Circuits and Systems-II: Analog and Digital Processing, Vol. 42, No. 4, pp 298-301, April*, 1995.

[6] A. B. Premkuma. Corrections to an rns to binary converter in a three moduli set with common factors. *IEEE Trans. on Circuits and Systems-II: Analog and Digital Processing, Vol. 51, No.1, pp 43, January*, 2004.

[7] A. B. Premkumar. An rns to binary converter in $2n+1$,$2n$, $2n-1$ moduli set. *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing, Vol. 39, No. 7, pp. 480-482, July*, 1992.

[8] N. Szabo and R Tanaka. *Residue Arithmetic and its Application to Computer Technology*. MC-Graw-Hill, New York, 1967.

[9] M. O. Ahmad W. Wang, M.N.S. Swamy and Y. Wang. A study of the residue-to-binary converters for three-moduli sets. *IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications, Vol. 50,No. 2, Feb.,pp 235-243*, 2003.

[10] M.N.S. Swamy W. Wang and M.O. Ahmad. Moduli selection in rns for efficient vlsi implementation. *International Symposium on Circuits and Systems, Vol. 4, pp.512-515*, 2003.

[11] Y. Wang. New chinese remainder theorems. *in Proc. Asilomar Conference, USA, pp. 165-171, Nov.*, 1998.

[12] M.O. Ahmad Y. Wang, M.N.S. Swamy. Residue to binary number converters for three moduli set. *IEEE Trans. Circuits Syst. II, vol. 46, pp. 180-183, Feb.*, 1999.

[13] H.M. Yassine and W.R. Moore. Improved mixed radix conversion for residue number system architectures. *proceedings of IEEE pt-G , Vol. 138, No.1, pp.120-124, Feb.*, 1991.