An Efficient FPGA Design of Reverse Converter for the Moduli Set $\{2n+2, 2n+1, 2n\}$

Kazeem Alagbe Gbolagade[1,2], George Razvan Voicu[1], and Sorin Dan Cotofana[1]
Computer Engineering Lab., Delft University of Technology, Delft, The Netherlands[1]
and
University for Development Studies, Navrongo, Ghana[2].

***Abstract*—This paper points out error in earlier literature and then proposes a novel reverse converter for the moduli set $\{2n+2, 2n+1, 2n\}$. A previously proposed scheme is simplified in order to obtain a reverse converter that uses mod-n operations. Next, a low complexity implementation that does not require the explicit use of modulo operation in the conversion process is presented. We implement the proposed converter and the best equivalent state of the art converters on Xilinx Spartan 3 FPGA. The results indicate that, on average, our proposal is about 2% and 15% better in terms of area costs and conversion time, respectively, when compared to the best known equivalent state of the art converter.**

***Index Terms*—Residue Number System, Reverse Converter, Chinese Remainder Theorem.**

I. Introduction

Residue Number Systems (RNS) is an unweighted number system, which is usually employed in addition and multiplication dominated intensive applications such as fast Fourier transform, discrete Fourier transform, image processing, cryptography, digital filtering, and video coding [3], [10]. This is due to the RNS inherent features, such as carry free operations, parallelism, modularity, and fault tolerance. However, despite all these advantages, RNS have not found a widespread usage in general purpose processors since sign detection, magnitude comparison, overflow detection, and division are rather difficult to perform. Several solutions for these problems, which rely heavily on RNS to binary conversion, have been proposed [10].

RNS based calculation requires reverse and forward conversions, which must be as fast as possible not to nullify the RNS advantages. In the past, several converters have been proposed for different moduli sets, e.g., $\{2^n, 2^n-1, 2^n+1\}$ [2], [11], [13], $\{2^n, 2^{n+1}-1, 2^n-1\}$ [6], [7], $\{2n+2, 2n+1, 2n\}$ [4], [8], $\{2n+2, 2n+1, 2n\}$ [1], [5], [9]. The advantages of utilizing the moduli set $\{2n+2, 2n+1, 2n\}$ over the $\{2^n, 2^n-1, 2^n+1\}$ moduli set have been discussed extensively in literature [1], [9].

In this paper, we point out error in [5] and then propose a novel reverse converter for the moduli set $\{2n+2, 2n+1, 2n\}$, which has a common factor of 2. First, we simplify a previously proposed scheme, which is based on the traditional Chinese Remainder Theorem (CRT) in order to obtain a reverse converter that uses mod-n operations. Next, a low complexity implementation that does not require the explicit use of modulo operation in the conversion process is presented. We implement the proposed converter and the converters [1] and [9] on Xilinx Spartan 3 FPGA. The results indicate that, on average, our proposal performs reverse conversion 15% and 13% faster than the converters in [1] and [9], respectively. Additionally, on average, the proposed scheme requires 2% and 66% lesser hardware resources than the converters in [1] and [9], respectively.

The rest of this paper is organized as follows. In Section II, the new algorithm for reverse conversion is proposed. Section III presents the hardware realization of the proposed reverse converter, while the paper is concluded in Section IV.

II. Proposed Algorithm

The RNS to binary converter proposed in [5] is not entirely valid. Given the RNS number (x_1, x_2, x_3) for the moduli set $\{2n+2, 2n+1, 2n\}$, we first correct the error in [5] and then present an efficient RNS to binary conversion technique based on the proposed conversion scheme in [5]. The proposed scheme does not require explicit use of the modulo operation at the final stage of computation.

The following theorem was presented in [5].

Theorem 1: The decimal equivalent of the RNS number (x_1, x_2, x_3) with respect to the moduli set $\{m_1 = 2n+2, m_2 = 2n+1, m_3 = 2n\}$, for any even integer $n > 0$ is computed as follows:

$$X = (x_2-x_1)m_1+x_1 + m_1m_2\left|\frac{(x_1+x_3)}{2}-x_2\right|_{\frac{m_3}{2}} \quad (1)$$

(1) is erroneous due to the following reasons:
If the condition

$$x_1 > x_2 \quad (2)$$

$$\left|\frac{(x_1+x_3)}{2}-x_2\right|_{\frac{m_3}{2}} = 0 \quad (3)$$

holds true, (1) produces a negative result, which is erroneous. In order to obtain correct results also if this type of situation arises, we modify Theorem 1 as follows:

Theorem 2: The decimal equivalent of the RNS number (x_1,x_2,x_3) for the moduli set $\{2n+2,2n+1,2n\}$ is computed as follows:

$$\begin{cases} X = y, & \text{y} \geq 0 \\ X = y+M_L, & \text{y}<0 \end{cases} \quad (4)$$

where

$$y = m_1(x_2-x_1)+x_1 + m_1m_2\left|\frac{(x_1+x_3)}{2}-x_2\right|_{\frac{m_3}{2}} \quad (5)$$

and $M_L = \frac{m_1m_2m_3}{2}$

Proof: To prove this theorem, we use the following lemma presented in [12]:

$$|am_1|_{m_1m_2} = m_1\,|a|_{m_2}. \quad (6)$$

We utilize the equation given below, which has been proved in [5] in order to prove the above theorem.

$$X = \left|\frac{m_2m_3}{2}x_1 - m_1m_3x_2 + \frac{m_1m_2}{2}x_3\right|_{M_L}$$

Putting $m_3 = m_2-1$ in the above equation, we obtain:

$$\begin{aligned} X &= \left|\frac{m_2m_3}{2}x_1 - m_1x_2(m_2-1) + \frac{m_1m_2}{2}x_3\right|_{M_L} \\ &= |m_1x_2 + \left|\frac{m_2m_3}{2}x_1 - m_1m_2x_2 + \frac{m_1m_2}{2}x_3\right|_{\frac{m_1m_2m_3}{2}}\Big|_{M_L} \end{aligned}$$

Applying (6) we have:

$$X = |m_1x_2 + m_2\left|\frac{m_3}{2}x_1 - m_1x_2 + \frac{m_1}{2}x_3\right|_{\frac{m_1m_3}{2}}\Big|_{M_L} \quad (7)$$

Putting $m_3 = m_1-2$ in the above equation, we obtain:

$$\begin{aligned} &= |m_1x_2 + m_2\left|\frac{(m_1-2)}{2}x_1 - m_1x_2 + \frac{m_1}{2}x_3\right|_{\frac{m_1m_3}{2}}\Big|_{M_L} \\ &= |m_1x_2-m_2x_1 + m_2\left|m_1\frac{(x_1+x_3)}{2} - m_1x_2\right|_{\frac{m_1m_3}{2}}\Big|_{M_L} \end{aligned}$$

Applying (6) we obtain:

$$X = |m_1x_2 - x_1(m_1-1) + m_1m_2\left|\frac{(x_1+x_3)}{2}-x_2\right|_{\frac{m_3}{2}}\Big|_{M_L}$$

Further simplifications give:

$$X = \left|(x_2-x_1)m_1+x_1+m_1m_2\left|\frac{x_1+x_3}{2}-x_2\right|_{\frac{m_3}{2}}\right|_{M_L} = |y|_{M_L} \quad (8)$$

(8) is the general expression of (5), valid for both y positive and negative.

Next we demonstrate that at most one M_L corrective addition is required. By definition of modulus we have:

$$\begin{aligned} 0 \leq \quad & \left|\tfrac{x_1+x_3}{2}-x_2\right|_{\frac{m_3}{2}} \leq \frac{m_3}{2}-1 \quad \Big|\cdot m_1m_2 \\ 0 \leq \quad & m_1m_2\left|\tfrac{x_1+x_3}{2}-x_2\right|_{\frac{m_3}{2}} \leq \frac{m_1m_2m_3}{2}-m_1m_2. \end{aligned}$$

Adding to this double inequality the following inequalities: $0 \leq m_1x_2 < m_1m_2$ and $-m_1m_2 < -m_2x_1 \leq 0$, we have

$$-m_1m_2 < m_1x_2 - m_2x_1 + m_1m_2\left|\frac{x_1+x_3}{2}-x_2\right|_{\frac{m_3}{2}} < \frac{m_1m_2m_3}{2}$$

Thus one corrective addition of M_L is required in order to obtain the correct result when $y<0$, and (5) holds true. ∎

For the case when $y<0$, the correct result is computed as follows:

$$\begin{aligned} X &= m_2(x_2-x_1)+x_2 + m_1m_2\left|\frac{(x_1+x_3)}{2}-x_2\right|_{\frac{m_3}{2}} + M_L \\ &= m_2(x_2-x_1)+x_2 + m_1m_2\left(\left|\frac{(x_1+x_3)}{2}-x_2\right|_{\frac{m_3}{2}} + \frac{m_3}{2}\right) \end{aligned} \quad (9)$$

III. Hardware Realization of Algorithm-2

The hardware realization of the proposed scheme is depicted by Figure 1. The implementation follows Equation (5) but the following should be noted. At a first glance, D is a 3:1 adder. However, the extra input x_2 can be embedded into the partial product matrix of the m_1 multiplier according to the merged arithmetic principle. Furthermore, the modulo-$\frac{m_3}{2}$ operation associated with the adder C does not have to be explicitly computed. It can be replaced by at most one corrective addition.

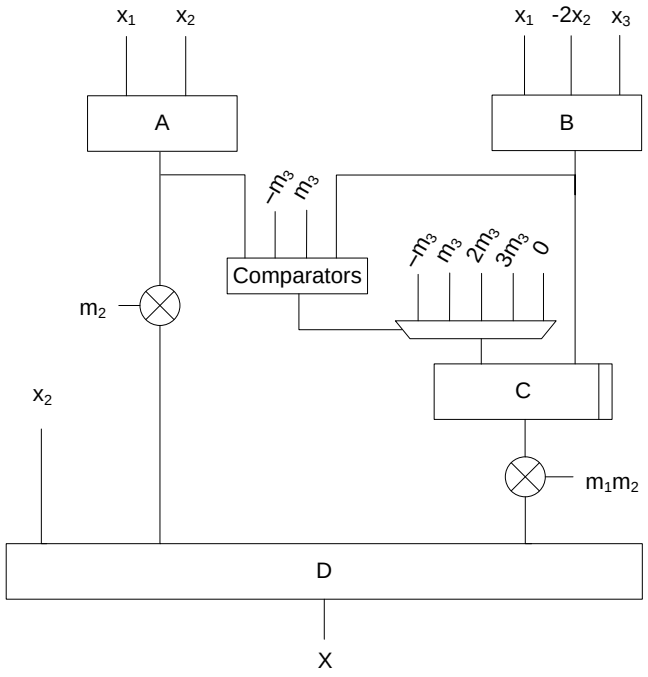

Figure 1. Block diagram of our proposal

In order to demonstrate that no explicit modulo operation is required by the proposed converter, we analyze the two possible extreme cases as follows:

Case 1: $(x_1 + x_3) = 0$ and $x_2 = 2n$. This results in the most negative value one may get. In this case Equation (5) reduces to $|-x_2|_{\frac{m_3}{2}}$. To perform the modulo $\frac{m_3}{2}$ operation, we need to do corrective additions. Given that $m_3 + (-x_2) = (2n) + (-2n) = 2n - 2n = 0$, for any positive even integer n, only one corrective addition with m_3 is required to compute the modulo.

Case 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 (5) reduces to $|\frac{(x_1+x_3)}{2}|_{\frac{m_3}{2}}$. Given that $m_3 - \frac{(x_1+x_3)}{2} = (2n) - \frac{(2n+1+2n-1)}{2} = 2n - 2n = 0$ the maximum sum in the modulo adder cannot exceed $\frac{m_3}{2}$, thus only one correction is required.

This means that the modulo $\frac{m_3}{2}$ operation can be implemented with at most one corrective addition.

As demonstrated by (5), the final modulo-M also does not require explicit implementation as in (9).

The scheme is simplified by moving the M_L corrective addition before the $m_1 m_2$ multiplication, hence transforming it into a corrective $\frac{m_3}{2}$ addition. As mentioned before, this correction must be applied when both of the following statements hold true:

$$x_1 > x_2 \tag{10}$$

$$\left| \frac{(x_1 + x_3)}{2} - x_2 \right|_{\frac{m_3}{2}} = 0 \tag{11}$$

We now combine the modulo-$\frac{m_3}{2}$ operations by revisiting the correction rules:

- if tentative sum is smaller than -$\frac{m_3}{2}$ add m_3;
- if tentative sum is greater than or equal to -$\frac{m_3}{2}$ add $\frac{m_3}{2}$; Equation (11) holds true when the tentative sum is equal to $-m_3$, but we can see from *Case 1* that Equation (10) does not hold true, hence the extra $\frac{m_3}{2}$ addition is not needed;
- if tentative sum is zero and Equation (10) is true (the sign bit of adder A is 1) add $\frac{m_3}{2}$;
- otherwise do nothing;

In this way all modulo operations have been replaced by a single corrective addition or subtraction greatly reducing the complexity of the converter. The theoretical analysis result of our proposal is almost the same as the one already presented in [5]. To avoid duplication, this theoretical analysis result is omitted in this paper. We structurally described our proposal, the ones in [1] and [9], in VHDL. We implement the proposed converter and the converters in [1] and [9] using Xilinx ISE 10.1 software on a Xa3s2004vqg100 FPGA. The implementation result is presented in Table I. The results indicate that, on average, our proposal performs reverse conversion 15% and 13% faster than the converters in [1] and [9], respectively. Additionally, on average, the proposed scheme requires 2% and 66% lesser hardware resources than the converters in [1] and [9], respectively.

IV. Conclusions

In this paper, we pointed out error in [5] and then proposed a novel reverse converter for the moduli set $\{2n+2, 2n+1, 2n\}$, which has a common factor of 2. First, we simplified a previously proposed scheme, which is based on the traditional CRT in order to obtain a reverse converter that uses mod-n operations. Next, a low complexity implementation that does not require

Table I
IMPLEMENTATION RESULTS

n	Proposed Area (FPGA Slices)	Area [1] (FPGA Slices)	Area [9] (FPGA Slices)	Proposed Delay (ns)	Delay [1] (ns)	Delay [9] (ns)
5	48	58	161	33.16	41.15	40.09
6	55	58	156	29.84	40.68	38.81
7	39	58	142	29.89	40.95	41.64
8	36	48	113	22.25	29.26	30.47
9	58	74	186	31.39	43.07	38.06
10	69	73	190	34.30	42.54	37.08
15	45	73	186	37.66	42.97	43.05
16	43	57	135	23.05	27.75	33.85
20	78	85	218	40.92	46.05	38.86
30	112	85	320	45.23	45.93	45.72
40	87	102	243	37.16	44.13	41.84
50	114	102	318	40.74	45.06	45.53
60	125	102	358	44.63	44.72	46.27
70	129	116	373	41.55	46.66	45.24
80	97	115	274	36.71	47.23	41.33
90	150	115	434	44.75	46.05	48.66
100	125	115	352	39.136	44.80	43.97

the explicit use of modulo operation in the conversion process is presented. We implemented the proposed converter and the converters [1] and [9] on Xilinx Spartan 3 FPGA. The results indicate that, on average, our proposal performs reverse conversion 15% and 13% faster than the converters in [1] and [9], respectively. Additionally, on average, the proposed scheme requires 2% and 66% lesser hardware resources than the converters in [1] and [9], respectively.

REFERENCES

[1] M.O. Ahmad, Y. Wang, and M.N.S. Swamy. Residue to binary number converters for three moduli set. *IEEE Trans. on Circuits and Systems-II, Vol. 46, No.7, pp. 180-183, Feb.*, 1999.

[2] R. Chaves and L. Sousa. Improving residue number system multiplication with more balanced moduli sets and enhanced modular arithmetic structures. *IET Comp. Digital Tech., Vol. 5, No.1, pp.472-480, Sept.*, 2007.

[3] R. Conway and J. Nelson. Improved RNS fir filter architectures. *IEEE Trans. on Circuits and Systems-II: Express briefs, Vol. 51, No.1, pp. 26-28, January*, 2004.

[4] K.A. Gbolagade and S.D. Cotofana. Residue number system operands to decimal conversion for 3-moduli sets. *Proceedings of 51st IEEE Midwest Symposium on Circuits and Systems, pp.791-794, Knoxville, USA, August*, 2008.

[5] K.A. Gbolagade and S.D. Cotofana. A residue to binary converter for the $\{2n+2, 2n+1, 2n\}$ moduli set. *Proceedings of 42nd Asilomar Conference on Signals, Systems, and Computers, pp. 1785-1789, California, USA, October*, 2008.

[6] S. Lin, M. Sheu, and C. Wang. Efficient VLSI design of residue to binary converter for the moduli set $\{2^n, 2^{n+1}-1, 2^n-1\}$. *IEICE Trans. INF. and SYST., Vol. E91-D, No.7, pp. 2058-2060, July*, 2008.

[7] P.V.A. Mohan. Rns-to-binary converter for a new three-moduli set $\{2^{n+1}-1, 2^n, 2^n-1\}$. *IEEE Trans. on Circuits and Systems-II: Express briefs, Vol. 54, No.9, pp. 775-779, September*, 2007.

[8] 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.

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

[10] L. Sousa. Efficient method for magnitude comparison in RNS based on two pairs of conjugate moduli. *Proceedings of the 18th IEEE Symposium on Computer Arithmetic*, 2007.

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

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

[13] Y. Wang, X. Song, M. Aboulhamid, and H. Shen. Adder based residue to binary number converters for $\{2^n+1, 2^n, 2^n-1\}$. *IEEE Trans. on Signal Processing, Vol. 50, pp.1772-1779, July*, 2002.