# An Efficient FPGA Design of Residue-to-Binary Converter for the Moduli Set $\{2n+1, 2n, 2n-1\}$

Kazeem Alagbe Gbolagade, George Razvan Voicu, and Sorin Dan Cotofana

***Abstract*—In this paper, we propose a novel reverse converter for the moduli set $\{2n+1, 2n, 2n-1\}$. First, we simplify the Chinese Remainder Theorem in order to obtain a reverse converter that uses mod-$(2n-1)$ operations. Next, we present a low complexity implementation that does not require the explicit use of modulo operation in the conversion process and we prove that theoretically speaking it outperforms state of the art equivalent converters. We also implemented 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 14%, 21%, and 8% better in terms of conversion time, area cost, and power consumption, respectively.**

***Index Terms*—Code converters, Field programmable gate arrays, Residue Arithmetic**

## I. Introduction

The attractive carry-free property of Residue Number Systems (RNS) gives room for RNS implementations in a variety of specialised high-performance Digital Signal Processing (DSP) applications. RNS is mostly applied in addition and multiplication dominated DSP applications such as Digital Filtering and Convolutions [1]. Moduli selection and data conversion are the two most important issues that determine the RNS hardware performance and may limit the utilization of RNS in DSP applications [2]. Moduli sets of length three have been extensively studied [2]. While the most popular length three moduli set is $\{2^n+1, 2^n, 2^n-1\}$, the advantages of utilizing the moduli set $\{2n+1, 2n, 2n-1\}$ over the $\{2^n+1, 2^n, 2^n-1\}$ moduli set have been extensively discussed in [3], [4], [5].

Several data conversion techniques have been proposed based on either the Chinese Remainder Theorem (CRT) [6], [7], [5], [8], or on the Mixed Radix Conversion (MRC) [9]. The major CRT problem is the complex and slow modulo-$M$ operation ($M = m_1 m_2 m_3$ being the system dynamic range, thus a rather large constant).

In this paper, a novel reverse converter for the moduli set $\{2n+1, 2n, 2n-1\}$ is proposed. First, we simplify the CRT to obtain a reverse converter that utilizes mod-$(2n-1)$ operations instead of mod-$(2n+1)(2n-1)$ and $(2n)(2n-1)$ operations required by the converters in [3] and [6], respectively. Next, we present a low complexity implementation that does not require the explicit use of the modulo operation in the conversion process as it is normally the case in the traditional CRT and some other state of the art equivalent converters.

## II. Proposed Algorithm

For the sake of completeness of this work, we briefly re-state without proof the following theorem, which has been presented in [6] before introducing our approach.

Manuscript received November 8, 2009; revised March 14, 2010; accepted May 03, 2010.

K. A. Gbolagade is with Computer Engineering Laboratory, Delft University of Technology, Mekelweg 4, 2628 CD, Delft (+31), The Netherlands, and also with University for Development Studies, P.O.Box 24, Navrongo (+233), Ghana (e-mail: gbolagade@ce.et.tudelft.nl).

G. R. Voicu and S. D. Cotofana are with Computer Engineering Laboratory, Delft University of Technology, Mekelweg 4, 2628 CD, Delft (+31) The Netherlands (e-mail: g.r.voicu@tudelft.nl; sorin@ce.et.tudelft.nl).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2010.2050608

*Theorem 1:* 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:

$$X = -m_2 x_1 + m_1 \left| \frac{m_2}{2}(x_1 + x_3) - m_3 x_2 \right|_{m_2 m_3}, \quad (1)$$

$$X = -m_2 x_1 + m_1 \left| \frac{m_2 m_3}{2} + \frac{m_2}{2}(x_1 + x_3) - m_3 x_2 \right|_{m_2 m_3}. \quad (2)$$

Next, we propose to simplify (1) and (2) in order to obtain a converter that only utilizes modulo-$(2n-1)$.

*Theorem 2:* 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 as follows:

$$\begin{cases} X = y, & \text{y} \geq 0 \\ X = y + M, & \text{y} < 0 \end{cases} \quad (3)$$

where

$$y = m_2(x_2 - x_1) + x_2 + m_1 m_2 \left| \frac{(x_1 + x_3)}{2} - x_2 \right|_{m_3}. \quad (4)$$

*Proof:* To prove this theorem we use the following lemma presented in [10]:

$$|a m_1|_{m_1 m_2} = m_1 |a|_{m_2}. \quad (5)$$

To be more accurate and since (1) may occassionally produce negative result, $X$ is represented as

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

Substituting $m_3 = m_2 - 1$ and applying (5) we get

$$X = \left| m_2(x_2 - x_1) + x_2 + m_1 m_2 \left| \frac{(x_1 + x_3)}{2} - x_2 \right|_{m_3} \right|_M. \quad (6)$$

Equation (6) is the general expression of (4), valid for both $y$ positive and negative. The next stage of the proof is to demonstrate that at most one corrective addition is required for the calculation of the mod-$M$. We demonstrate that by considering the most positive value one may get in (6).

- *Most positive value*: in order to get the most positive value in (6), the following must hold true: $\left| \frac{(x_1+x_3)}{2} - x_2 \right|_{m_3} = m_3 - 1$, $x_1 = m_1 - 1$, $x_2 = 1$, $x_3 = m_3 - 1$. Substituting all these values in (6), we obtain

$$X = |M - 2m_1 m_2 + 2m_2 + 1|_M. \quad (7)$$

  Since $0 < M - 2m_1 m_2 + 2m_2 + 1 < M$, no corrective addition of $M$ is required in order to obtain the desired result.

On the other hand, for $y < 0$, the following must hold true: $\left| \frac{(x_1+x_3)}{2} - x_2 \right|_{m_3} = 0$, $x_1 > x_2$. We demonstrate that only one corrective addition is required in order to compute the correct result. This is achieved by computing the most negative result one may have in (6).

- *Most negative value*: in order to get the most negative value in (6), we substitute $x_1 = m_1 - 1$, $x_2 = 0$, and $\left| \frac{(x_1+x_3)}{2} - x_2 \right|_{m_3} = 0$ in (6), obtaining

$$X = |-m_1 m_2 + 1|_M. \quad (8)$$

Since $0 < -m_1m_2+1+M < M$, only one corrective addition is therefore required in order to obtain the correct result if $y < 0$.

Thus, (3) holds true. ■

Thus, given that $M = m_1m_2m_3$, for the case when $y < 0$, the correct result can be computed as follows:

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

*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 $(x_1 + x_3)$ odd as follows:

$$\begin{cases} X = y, & \text{y} \geq 0 \\ X = y + M, & \text{y} < 0 \end{cases} \quad (10)$$

where

$$y = m_2(x_2 - x_1) + x_2 + m_1m_2\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3}. \quad (11)$$

*Proof:* Similarly, in order to be more accurate and also since (2) may occassionally produce negative result, $X$ is represented as

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

By substituting $m_3 = m_2 - 1$ and applying (5) we get

$$X = \left|m_2(x_2 - x_1) + x_2 + m_1m_2\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3}\right|_M. \quad (12)$$

Again, from (11), it can be seen easily that (12) is the same as $|y|_M$. Just as earlier described, we need to demonstrate that at most one corrective addition is required for the calculation of mod-$M$. We demonstrate that by considering the most positive value one may get in (12).

- *Most positive value*: in order to get the most positive value in (12), the following must hold true: $\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3} = m_3 - 1$, $x_1 = 1$, $x_2 = 1$, $x_3 = m_3 - 1$.
  Substituting the above values in (12), we obtain
  $$X = |M - m_1m_2 + 1|_M. \quad (13)$$
  Since $0 < M - m_1m_2 + 1 < M$, no corrective addition is required for the calculation of mod-$M$.

Again, for $y < 0$, the following must hold true: $\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3} = 0$, $x_1 > x_2$. We demonstrate that only one corrective addition is needed in order to obtain the correct result in (12). This is achieved by considering the most negative value one may get in (12).

- *Most negative value*: in order to get the most negative value in (12), the following must hold true: given that $\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3} = 0$, $x_1 = 1$, $x_2 = 0$, and $x_3 = m_3 - 1$. Substituting these values in (12), we obtain
  $$X = |-m_2 + 1|_M. \quad (14)$$
  Since $0 < -m_2 + 1 + M < M$, only one corrective addition is required in order to obtain correct result.

■

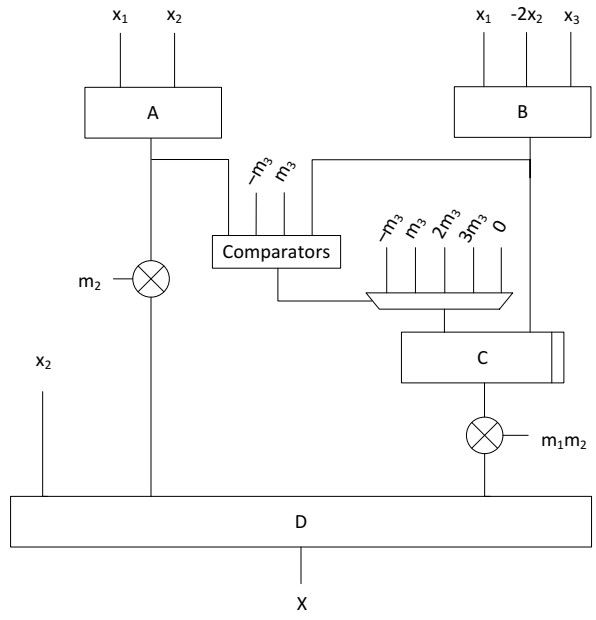


Figure 1. Hardware Structure of Our Proposal

Thus, for the case when $y < 0$, the correct result can be computed as follows:

$$X = m_2(x_2 - x_1) + x_2 + m_1m_2\left(\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3} + m_3\right). \quad (15)$$

## III. Hardware Realization

The hardware realization of the proposed scheme, depicted in Fig. 1 is based on the equations in Theorems 2 and 3. In adder A, residue $x_1$ is subtracted from $x_2$ and next the result is multiplied by $m_2$. The 3-input adder B computes $(\frac{x_1+x_3}{2} - x_2)$ and can be implemented as a 3:2 Carry Save Adder (CSA) followed by a Carry Propagate Adder (CPA). We prove later in this section that only one corrective addition or subtraction is required to compute the modulo-$m_3$ operation and this can be combined with the possible additions of $\frac{m_3}{2}$ and $m_3$ terms. The above mentioned operations are implemented by adder C with a selectable input. The hardware implementation removes the fractions by shifting left all the operands involved in adder B and C, thereby extending the two adders with one bit. Finally, the output of adder C, without the rightmost bit to account for the previous shift, is multiplied by $m_1m_2$ and the result is summed together by adder D with the one from the multiplier $m_2$. The extra input $x_2$ for adder D can be embedded in the $m_2$ multiplier according to the principle of merged arithmetic, thus D can be actually implemented as a standard 2:1 adder.

Next, we demonstrate that no explicit mod-$m_3$ operation is required for the computation of $\left|\frac{(x_1+x_3)}{2} - x_2\right|_{m_3}$ and $\left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3}$ by analyzing the four possible extreme cases as follows:

*Case 1:* $(x_1 + x_3) = 0$ and $x_2 = 2n - 1$. This results in the most negative value one may get. In this case the modulus in (4) reduces to $|-x_2|_{m_3}$. To perform the modulo $m_3$ operation we need to do corrective additions. Given that $m_3 + (-x_2) = (2n-1) - (2n-1) = 2n - 1 - 2n + 1 = 0$, for any positive 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 the modulus in (4) reduces to $|\frac{(x_1+x_3)}{2}|_{m_3}$. Given that $m_3 - \frac{(x_1+x_3)}{2} = (2n-1) - \frac{(2n+2n-2)}{2} = 2n-1-2n+1 = 0$ the maximum sum in the modulo adder cannot exceed $m_3$, thus one subtraction with $m_3$ is required.

*Case 3:* $(x_1+x_3) = 1$ and $x_2 = 2n-1$. In this case the modulus from (11) reduces to $|\frac{m_3}{2} + \frac{1}{2} - x_2|_{m_3}$. Given that in this case $\frac{m_3}{2} + \frac{1}{2} - x_2$ is always negative and that $m_3 + \frac{m_3}{2} + \frac{1}{2} - x_2 = 2n-1+n-\frac{1}{2}+\frac{1}{2}-2n+1 = n > 0$, for any integer $n$, only one corrective addition with $m_3$ is required to compute the modulo.

*Case 4:* $(x_1+x_3)$ odd has the maximum possible value and $x_2$ is zero. The modulus from (11) reduces to $|\frac{m_3}{2} + \frac{(x_1+x_3)}{2}|_{m_3}$. Given that $2m_3 - (\frac{m_3}{2} + \frac{(x_1+x_3)}{2}) = 2(2n-1) - (\frac{2n-1}{2} + \frac{(2n+2n-2)-1}{2}) = 4n-2-3n+2 = n > 0$, for any positive integer $n$, one corrective subtraction of $m_3$ is required to compute the modulo.

This means that the modulo $m_3$ operation can be implemented with at most one corrective addition or subtraction. In the following, we prove that the addition of the $\frac{m_3}{2}$ term can be actually postponed and embedded into the correction step required for the modulo-$m_3$ operation without any delay overhead. Thus, we remove $\frac{m_3}{2}$ as adder C input and revisit the 4 correction cases analyzed above.

If $(x_1+x_3)$ is even, the term $\frac{m_3}{2}$ is not part of the calculation and the correction can be done as usual. If $(x_1+x_3)$ is odd, the tentative sum at the output of adder B is $\frac{(x_1+x_3)}{2} - x_2$ instead of $\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2$, thus it is smaller with $\frac{m_3}{2}$ than it should actually be. Taking that into consideration, the correction rules change to:

1) $(x_1+x_3)$ even:
   - if tentative sum is smaller than 0 add $m_3$;
   - if tentative sum is equal or larger than $m_3$ subtract $m_3$;
   - otherwise do nothing;
2) $(x_1+x_3)$ odd:
   - if tentative sum is smaller than $-\frac{m_3}{2}$ add $\frac{3m_3}{2}$;
   - if tentative sum is greater than or equal to $\frac{m_3}{2}$ subtract $\frac{m_3}{2}$;
   - otherwise add $\frac{m_3}{2}$.

As indicated by (9) and (15), the final modulo-$M$ also does not require explicit implementation. The scheme is simplified by moving this addition before the $m_1m_2$ multiplication, hence transforming it into a corrective $m_3$ addition. As mentioned in the previous section, this correction must be applied when both of the following statements hold true:

$$x_1 > x_2 \tag{16}$$

$$\begin{cases} \left|\frac{(x_1+x_3)}{2} - x_2\right|_{m_3} = 0 & (x_3+x_1) \text{ even} \\ \left|\frac{m_3}{2} + \frac{(x_1+x_3)}{2} - x_2\right|_{m_3} = 0 & (x_3+x_1) \text{ odd} \end{cases} \tag{17}$$

We now combine the modulo-$m_3$ operations by revisiting the correction rules:

1) $(x_1+x_3)$ even:
   - if tentative sum is smaller than 0 add $m_3$;
     (17) holds true when the tentative sum is equal to $-m_3$, but we can see from *Case 1* that (16) does not hold true, hence the extra $m_3$ addition is not needed;
   - if tentative sum is zero and (16) is true (the sign bit of adder A is 1) add $m_3$;
   - otherwise do nothing;
     (17) holds true when the tentative sum is equal to $m_3$ and we can see from *Case 2* that this happens when $x_2$ is zero and $x_1 = 2n$, hence (16) also holds true. But the required $m_3$ addition cancels out the previous $m_3$ corrective subtraction for the modular-$m_3$ adder;
2) $(x_1+x_3)$ odd:
   - if tentative sum is smaller than $-\frac{m_3}{2}$ add $\frac{3m_3}{2}$;
     From *Case 3*, it can be seen that the minimum value for the tentative sum is $\frac{1}{2} - x_2 < -\frac{3m_3}{2}$, so (17) does not hold true. Thus, no extra $m_3$ addition is required;
   - if tentative sum is larger than $\frac{m_3}{2}$ subtract $\frac{m_3}{2}$;
     (17) holds true when the tentative sum is equal to $\frac{m_3}{2}$. Following this, $x_1 + x_3 - 2x_2 = 2m_3 \Rightarrow x_1 - x_2 = 2m_3 - x_3 + x_2$. Since $2m_3 - x_3 > 0$ (16) also holds true, thus the correction becomes $-\frac{m_3}{2} + m_3 = \frac{m_3}{2}$;
   - otherwise add $\frac{m_3}{2}$.

In this way all modulo operations have been replaced by a single corrective addition or subtraction greatly reducing the complexity of the converter.

## IV. Performance Evaluation

The converter in [3] for the moduli set under consideration has been shown to outperform the one in [4]. Recently, the converter in [6] was also shown to be better than the one in [3] through a detailed critical path analysis, which takes the hardware implementation details into consideration. In view of that, we compare our proposal with the equivalent converters in [6] and [3].

On the critical path, our proposal requires one 3:1 adder, two 2:1 adders, one multiplier, one comparator and one multiplexer; converter [3] requires one 4:1 adder, two 2:1 adders, and two multipliers, while converter [6] requires one 3:1 adder, two 2:1 adders, and two multipliers. Consequently, the newly introduced converter is less complex and it is faster than state of the art equivalent converters.

Additionally, we also carried out experimental comparison by describing the proposed converter, the ones [6] and [3] in VHDL, and implementing them on Xilinx Spartan 3 xa3s200-4-ftg256 FPGA, with Xilinx ISE 10.1.03. In order to properly evaluate the relations between these converters, several reverse converters were implemented for a wide range of values of $n$, up to an equivalent dynamic range of 64 bits. The converters performance evaluated in terms of area (expressed as number of FPGA slices), and delay (ns obtained by post-place & route static timing analysis) is depicted in Fig. 2 and Fig. 3, respectively.

As expected, since all the three converters have somehow similar architectures, all of them present roughly the same area and delay profile. The downward spikes that all the converters present in both area and delay graphs occur when $n$ is a power of 2. For this special values of $n$ the logic synthesizer optimizes the design by replacing the multipliers with simple shifters, thereby greatly reducing both the occupied area and the time of conversion. Apart from these spikes, we can observe some non-monotonic area and delay variations which are also induced by logic optimizations made by the synthesizer, for certain $n$ values.

The obtained values suggest that, on average, the proposed converter is capable of performing the reverse conversion 15.8% and 14% faster with 21% and 27% area decrease, when compared to the reverse converters in [3] and [6], respectively.

Exact values of number of occupied FPGA slices, conversion time, and average power consumption are presented in Table I, for some common dynamic ranges. For this particular cases, our converter consumes on average 8% and 18% less power than the one in [3] and [6], respectively.

## V. Conclusions

In this paper, we proposed a new reverse converter for the moduli set $\{2n+1, 2n, 2n-1\}$. First, we simplified the traditional CRT in order to obtain a reverse converter that utilizes mod-$(2n-1)$

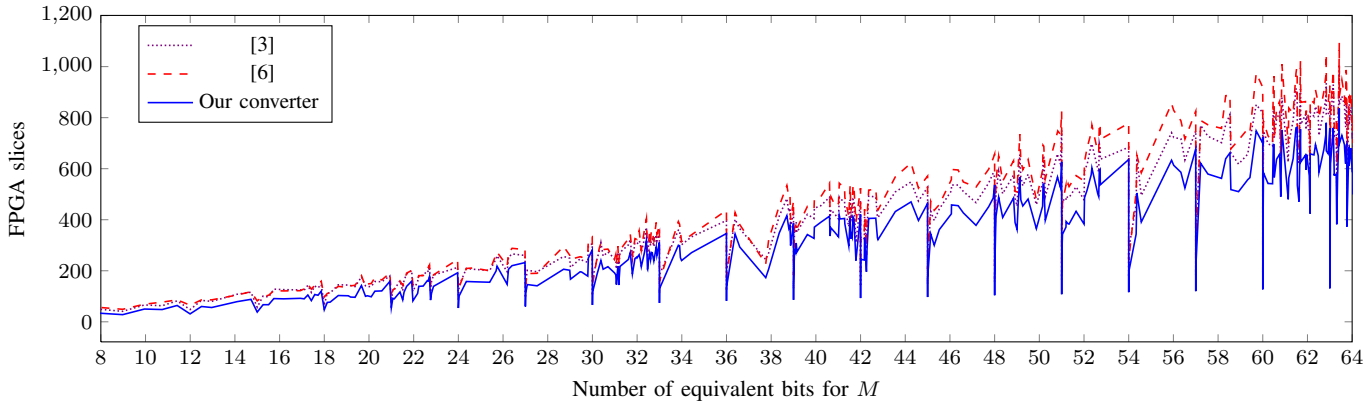


Figure 2. Area comparison

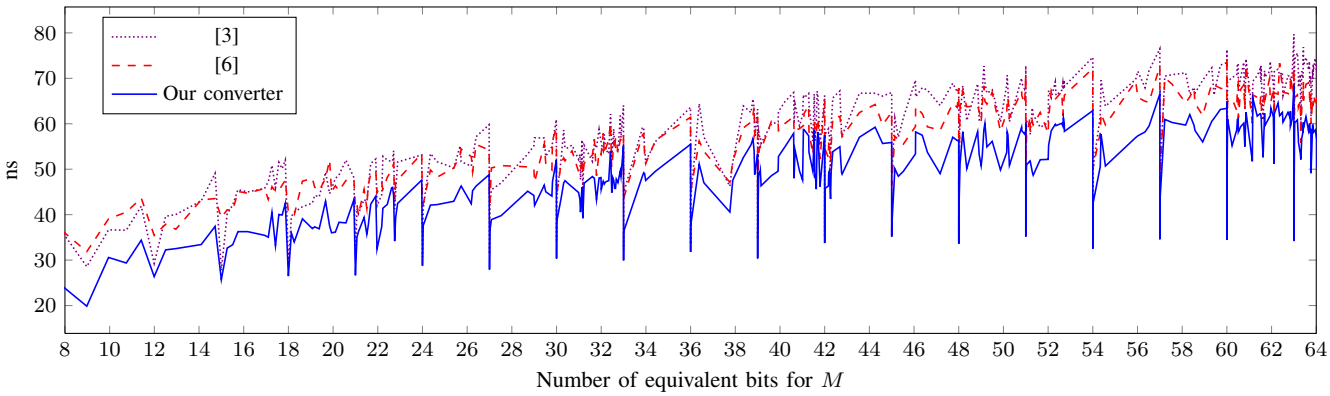


Figure 3. Delay comparison

Table I
IMPLEMENTATION RESULTS: AREA, DELAY, POWER

| M | n | Our Converter | | | [6] | | | [3] | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Area (slices) | Delay (ps) | Power (mW) | Area (slices) | Delay (ps) | Power (mW) | Area (slices) | Delay (ps) | Power (mW) |
| $2^8$ | 4 | 28 | 19857 | 15 | 49 | 31865 | 16 | 41 | 28575 | 15 |
| $2^{16}$ | 21 | 90 | 36262 | 19 | 121 | 44718 | 20 | 126 | 44999 | 22 |
| $2^{24}$ | 129 | 101 | 37599 | 19 | 136 | 42329 | 22 | 141 | 41312 | 24 |
| $2^{32}$ | 813 | 251 | 45543 | 33 | 328 | 53556 | 37 | 305 | 55609 | 42 |
| $2^{48}$ | 32769 | 177 | 39256 | 25 | 218 | 44896 | 28 | 253 | 47955 | 34 |
| $2^{64}$ | 1321123 | 589 | 57261 | 65 | 749 | 65295 | 68 | 735 | 71930 | 95 |

operations instead of mod-$(2n+1)(2n-1)$ and $(2n)(2n-1)$ operations required by the converters in [3] and [6], respectively. Next, we transformed the needed corrective addition of $M$ into a corrective $m_3$ addition. We then presented a novel low complexity implementation that does not require the explicit use of modulo operation in the conversion process.

The performance of the proposed converter is evaluated both theoretically, in terms of the required number of arithmetic operations and experimentally by FPGA implementation. On average, the proposed converter outperforms the state of the art with about 14%, 21%, and 8% in terms of conversion time, area cost and power consumption, respectively.

## REFERENCES


[1] R. Conway and J. Nelson, "Improved RNS FIR filter architectures," *IEEE Trans. Circuits Syst. II*, vol. 51, no. 1, pp. 26–28, Jan. 2004.

[2] W. Wang, M. Swamy, M. Ahmad, and Y. Wang, "A study of the residue-to-binary converters for the three-moduli sets," *IEEE Trans. Circuits Syst. I*, vol. 50, no. 2, pp. 235–243, Feb. 2003.

[3] M. Ahmad, Y. Wang, and M. Swamy, "Residue-to-binary number converters for three moduli sets," *IEEE Trans. Circuits Syst. II*, vol. 46, no. 7, pp. 180–183, Feb 1999.

[4] A. Premkumar, "An RNS to binary converter in $\{2n+1, 2n, 2n-1\}$ moduli set," *IEEE Trans. Circuits Syst. II*, vol. 39, no. 7, pp. 480–482, July 1992.

[5] ——, "An RNS to binary converter in a three moduli set with common factors," *IEEE Trans. Circuits Syst. II*, vol. 42, no. 4, pp. 298–301, Apr 1995.

[6] K. Gbolagade and S. Cotofana, "An efficient RNS to binary converter using the moduli set $\{2n+1, 2n, 2n-1\}$," in *XXIII Conf. Design of Circuits and Integrated Systems (DCIS 2008)*, Grenoble, France, Nov. 2008.

[7] ——, "A residue to binary converter for the $\{2n+2, 2n+1, 2n\}$ moduli set," in *Proc. 42nd Asilomar Conf. on Signals, Systems, and Computers (ACSSC 2007)*, California, USA, Oct. 2008, pp. 1785–1789.

[8] B. Vinnakota and V. Rao, "Fast conversion techniques for binary-residue number systems," *IEEE Trans. Circuits Syst. I*, vol. 41, pp. 927–929, Dec. 1994.

[9] M. Akkal and P. Siy, "A new mixed radix conversion algorithm MRC-II," *J. Syst. Archit.*, vol. 53, pp. 577–586, 2007.

[10] Y. Wang, "Residue-to-binary converters based on new chinese remainder theorems," *IEEE Trans. Circuits Syst. II*, vol. 47, no. 3, pp. 197–205, Mar. 2000.