Memoryless RNS-to-Binary Converters for the $\{2^{n+1}-1, 2^n, 2^n-1\}$ Moduli Set

Kazeem Alagbe Gbolagade[1,2], George Razvan Voicu[1], and Sorin Dan Cotofana[1]
CE Lab, Delft University of Technology, The Netherlands[1] and UDS, Navrongo, Ghana[2].

Abstract

In this paper, we propose two novel memoryless reverse converters for the moduli set $\{2^{n+1}-1, 2^n, 2^n-1\}$. The first proposed converter does not entirely cover the dynamic range while the second proposed converter covers the entire dynamic range. First, we simplify the Chinese Remainder Theorem in order to obtain a reverse converter that utilizes mod-$(2^{n+1}-1)$ operation. Second, we further reduce the resulting architecture to obtain a reverse converter that uses only carry save adders and carry propagate adders. FPGA implementation results indicate that, on average, the proposed limited dynamic range converter achieves about 42% area reduction. However, the second proposed converter provides only 29.48% area reduction when compared with the most effective equivalent state of the art converter. Both of the proposed converters also exhibit a small speed improvement over the state of the art equivalent converter.

***Keywords*-Residue Number System, Reverse Converter, Chinese Remainder Theorem, Memoryless Converter.**

I. Introduction

The Residue Number System (RNS) is a non-weighted number system that utilizes remainders to represent numbers. RNS has received considerable attention in arithmetic computation and Digital Signal Processing (DSP) applications such as digital filtering, Fast Fourier Transform, Discrete Cosine Transform, etc. This is due to the following inherent properties of RNS: parallelism, modularity, fault tolerance, and carry-free operations [1], [2]. Moduli Selection and Data Conversion are the two most important issues for a successful RNS utilization. Data Conversion can be categorized into forward and reverse conversions. The forward conversion involves converting a binary or decimal number into its RNS equivalent while the reverse conversion is the inverse operation, i.e., it involves converting RNS number into binary or decimal. Relatively, reverse conversion is more complex. Many algorithms have been designed for performing the reverse conversion with different choices of moduli sets, e.g., $\{2^n, 2^n-1, 2^n+1\}$ [2], $\{2^n, 2^{n+1}-1, 2^n-1\}$ [3], [4]. Recently, the moduli set $\{2^n, 2^{n+1}-1, 2^n-1\}$ was proposed in [3] by removing the modulus (2^n+1) from the 4-moduli set $\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$ proposed in [5]. This is due to the fact that performing the modulo (2^n+1)-type arithmetic is complex and degrades the entire RNS system performance in terms of both area and delay.

In this paper, two new memoryless residue to binary converters for the $\{2^{n+1}-1, 2^n, 2^n-1\}$ moduli set are proposed. First, we simplify the Chinese Remainder Theorem (CRT) to obtain a reverse converter that uses mod-$(2^{n+1}-1)$ instead of both mod-$(2^{n+1}-1)$ and mod-(2^n-1) required by the converter in [3]. Second, we further simplify the resulting architecture in order to obtain a reverse converter that utilizes only Carry Save Adders (CSAs) and Carry Propagate Adders (CPAs). We resolve the dynamic range limitation problem and obtain another reverse converter, which is practically evaluated to be better than the one in [4]. Theoretically speaking, the proposed converters are faster than the one in [4]. Experimentally, with no delay penalty, the proposed limited dynamic range converter achieves about 42% area reduction, while the second proposed converter provides only 29.48% area reduction.

II. Proposed Algorithm

Given the RNS number (x_1, x_2, x_3) for the moduli set $\{2^{n+1}-1, 2^n, 2^n-1\}$, the proposed algorithm computes the decimal equivalent of this RNS number based on a further simplification of the well-known traditional CRT.

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

$$|(m_1m_2)^{-1}|_{m_3} = 1, \quad (1)$$

$$|(m_1m_3)^{-1}|_{m_2} = 1, \quad (2)$$

$$|(m_2m_3)^{-1}|_{m_1} = -4. \quad (3)$$

Proof: It can be demonstrated by value substitution for

m_1, m_2, m_3 that $|1 \times (m_1 m_2)|_{m_3} = 1$, $|1 \times (m_1 m_3)|_{m_2} = 1$, and $|-4 \times (m_2 m_3)|_{m_1} = 1$. ■

Theorem 2. *The decimal equivalent of the residues* (x_1, x_2, x_3) *for the moduli set* $\{2^{n+1}-1, 2^n, 2^n-1\}$, *assuming* $X \in [0, M-(m_3)^2)$, *can be computed as follows:*

$$X = m_2 \left\lfloor \frac{X}{m_2} \right\rfloor + x_2, \tag{4}$$

$$\left\lfloor \frac{X}{m_2} \right\rfloor = x_3 - x_2 + m_3 \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1}. \tag{5}$$

Proof: Since (4) follows the basic integer division definition in RNS, which is always true, we only need to show the correctness of (5). The traditional CRT [1] for length 3 moduli set is given by:

$$X = \left| \sum_{i=1}^{3} M_i \left| M_i^{-1} x_i \right|_{m_i} \right|_M. \tag{6}$$

By substituting (1), (2), and (3) and applying $m_1 = 2m_2 - 1$ and $m_1 = 2m_3 + 1$ into (6) we obtain:

$$\begin{aligned} X &= |-4m_2m_3x_1 + 2m_2m_3x_2 - m_3x_2 \\ &\quad +2m_2m_3x_3 + m_2x_3|_M. \end{aligned} \tag{7}$$

Applying $|am_1|_{m_1m_2} = m_1 |a|_{m_2}$ [1] and $m_3 = m_2 - 1$, (7) becomes:

$$\begin{aligned} X &= |m_2x_3 - m_2x_2 + x_2 \\ &\quad +m_2m_3 \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1} \Big|_M. \end{aligned} \tag{8}$$

Dividing both sides of the above equation by m_2 and taking the floor, we shall have:

$$\begin{aligned} \left\lfloor \frac{X}{m_2} \right\rfloor &= |x_3 - x_2 \\ &\quad +m_3 \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1} \Big|_{m_1m_3}. \end{aligned} \tag{9}$$

Equation (9) is the general expression of (5) and it holds true for the entire dynamic range. The next stage of the proof is to demonstrate that the corrective addition required for the calculation of the mod-m_1m_3 can be avoided in most of the cases.

By definition of modulus we have:

$$\begin{aligned} &0 \le \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1} \le m_1 - 1 \quad \Big| \cdot m_3 \\ &0 \le m_3 \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1} \le m_3m_1 - m_3. \end{aligned} \tag{10}$$

Using the following inequalities and Equation (10)

$$0 \le x_3 < m_3 \text{ and } 0 \le x_2 < m_2 = m_3 + 1,$$

we have

$$\begin{aligned} -m_3 \le -x_2 &\le x_3 - x_2 + m_3 \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1} < \\ &< m_3m_1 - m_3 + m_3 = m_3m_1. \end{aligned}$$

Thus one corrective addition of m_1m_3 is required in order to obtain the correct result when $x_3 - x_2 + m_3 \left| -4x_1 + 2x_2 + 2x_3 \right|_{m_1} < 0$.

Further, we show that if we slightly restrict the RNS dynamic range, no corrective addition is required. For the numbers that require corrective addition the following hold true:

$$\begin{aligned} -x_2 + m_1m_3 \le &\quad \left\lfloor \frac{X}{m_2} \right\rfloor < m_1m_3 \quad | \cdot m_2 \\ M - m_2x_2 \le &\quad m_2 \left\lfloor \frac{X}{m_2} \right\rfloor < M \quad | + x_2 \\ M - (m_2 - 1)x_2 \le &\quad X < M \\ M - m_3m_3 \le &\quad X < M. \end{aligned}$$

Therefore, the numbers within the interval $[0, M-(m_3)^2)$ require no corrective addition and thus, (5) holds true. ■

The hardware required for the implementation of (5) can be further reduced by using the following properties from [4]:

Property 1: Modulo $(2^s - 1)$ multiplication of a residue number by 2^t, where s and t are positive integers, is equivalent to t-bit circular left shifting.

Property 2: Modulo $(2^s - 1)$ of a negative number is equivalent to the one's complement of the number, which is obtained by subtracting the number from $(2^s - 1)$.

Equation (5) can be directly rewritten as:

$$\left\lfloor \frac{X}{m_2} \right\rfloor = x_3 - x_2 + 2^n A - A, \tag{11}$$

$$A = |u_1 + u_2 + u_3|_{2^{n+1}-1}. \tag{12}$$

For simplicity sake, let us represent (11) by the following:

$$\left\lfloor \frac{X}{m_2} \right\rfloor = B_1 + B_2 + B_3, \tag{13}$$

$$B_1 = -x_2, B_2 = 2^n A + x_3, B_3 = -A. \tag{14}$$

Let the binary representations of the residues be the following:

$$\begin{aligned} x_1 &= (x_{1,n}x_{1,n-1}\cdots x_{1,0}), x_2 = (x_{2,n-1}x_{2,n-2}\cdots x_{2,0}), \\ x_3 &= (x_{3,n-1}x_{3,n-2}\cdots x_{3,0}). \end{aligned}$$

In (12), u_1, u_2, and u_3 are represented as follows:

$$\begin{aligned} u_1 &= \left| -2^2 x_1 \right|_{2^{n+1}-1} = (\underbrace{\overline{x}_{1,n-2}\cdots\overline{x}_{1,0}\overline{x}_{1,n}\overline{x}_{1,n-1}}_{n+1}), \\ u_2 &= |2x_2|_{2^{n+1}-1} = (\underbrace{x_{2,n-1}x_{2,n-2}\cdots x_{2,0}0}_{n+1}), \\ u_3 &= |2x_3|_{2^{n+1}-1} = (\underbrace{x_{3,n-1}x_{3,n-2}\cdots x_{3,0}0}_{n+1}). \end{aligned}$$

Assuming that A has the following binary representation:

$$A = (\underbrace{a_n a_{n-1} \cdots a_1 a_0}_{n+1}),$$

then B_2 will be given by

$$B_2 = (\underbrace{a_n a_{n-1} \cdots a_0 x_{3,n-1} x_{3,n-2} \cdots x_{3,0}}_{2n+1}). \quad (15)$$

B_1 and B_3 must have the same number of bits, i.e., $(2n+1)$-bits, as B_2 and are represented as:

$$B_1 = \underbrace{111\cdots11}_{n+1} \underbrace{\overline{x}_{2,n-1}\overline{x}_{2,n-2}\cdots\overline{x}_{2,0}}_{n}, \quad (16)$$

$$B_3 = \underbrace{111\cdots11}_{n} \underbrace{\overline{a}_n\overline{a}_{n-1}\cdots\overline{a}_0}_{n+1}. \quad (17)$$

III. Handling The Dynamic Range Limitation Problem

In this section, we resolve the dynamic range limitation problem. If (5) produces a negative result, then $|-4x_1 + 2x_2 + 2x_3|_{m_1} = 0$ since $m_3 \geq x_2$. Thus, (5) is negative if and only if $x_2 > x_3$ and $|-4x_1 + 2x_2 + 2x_3|_{m_1} = 0$. For this case, since it has been proved in Section II that only one corrective addition of $m_1 m_3$ is required, (9) can be written as:

$$\left\lfloor \frac{X}{m_2} \right\rfloor = x_3 - x_2 + m_1 m_3$$
$$= x_3 - x_2 + 2^{2n+1} - 2^{n+2} + 2^n + 1. \quad (18)$$

By using the following notations:

$$B_4 = -x_2 = \underbrace{111\cdots11}_{n+2} \underbrace{\overline{x}_{2,n-1}\overline{x}_{2,n-2}\cdots\overline{x}_{2,0}}_{n},$$

$$B_5 = x_3 + 2^{2n+1} - 2^{n+2} + 2^n + 1$$
$$= (\underbrace{100\cdots00}_{n-1})(\underbrace{101 x_{3,n-1} x_{3,n-2} \cdots x_{3,0}}_{n+3}), \quad (19)$$

equation (18) may be simplified as follows:

$$\left\lfloor \frac{X}{m_2} \right\rfloor = B_4 + B_5. \quad (20)$$

IV. Hardware Realization

The hardware implementations of the proposed reverse converter, which does not cover the entire dynamic range, namely CI is based on (12) and (13). In Figure 1, u_1, u_2, and u_3 are added by CSA1 with End Around Carry (EAC) producing s_1 and c_1. Next these must be added modulo $2^{n+1} - 1$ in order to obtain A. To speed up this addition, we utilize anticipated computation. We compute s_1 + c_1

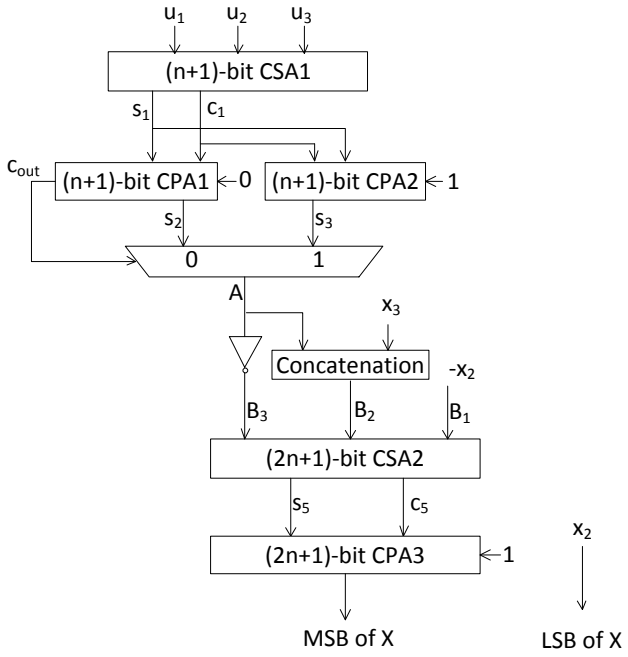

Figure 1. Proposed converter CI

for both $c_{in} = 0$ and $c_{in} = 1$ and we select the right result with a MUX. B_2 is easily obtained by concatenating the operand x_3 with the result of n-bit left shift of A. This concatenation does not require any hardware resources. The three operands B_1, B_2, and B_3 are added using CSA2 with EAC. It should be noted that in order to make B_1 and B_3 $(2n+1)$-bit numbers, 1's are appended to the result of complementations, as given in (16) and (17). Thus, the most significant $(n+1)$-bits from CSA2 are reduced to half adders (HAs). Moreover, since these half adders all have two inputs equal to 1, the final one's complement adder will always generate an EAC. Taking this into consideration the one's complement adder can be reduced to a normal CPA3 with a constant carry-in equal to 1. The final result, which is computed based on (4) is obtained just by a shift and a concatenation operation with no computational hardware. Given that in reality, the numbers that fall outside the range $[0, M - (m_3)^2)$ may be of interest, we resolve the dynamic range limitation problem and propose a second converter namely CII based on (12), (13), and (20). The hardware implementations of CII, which is valid for the entire dynamic range $[0, M-1]$, is depicted in Figure 2.

V. Performance Evaluation

In order to evaluate the performance of the proposed converters, we compare them with the best state of the art equivalent converters proposed in [4]. The result of this

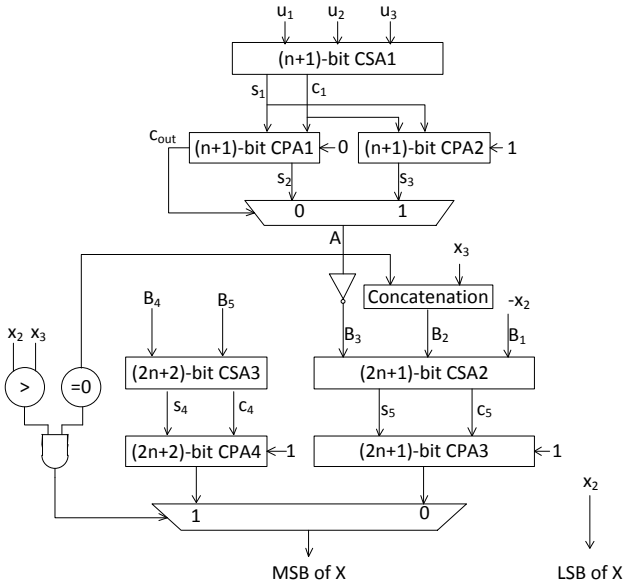

Figure 2. Proposed converter CII

comparison is presented in Table I. In the table, we have the converters CI and CII as the converters proposed in this paper and CIII, the one in [4]. The theoretical results indicate that proposed converter CI outperforms CIII in all terms and Converter CII, which is valid for the entire dynamic range, maintains almost the same lower delay as CIII at an additional hardware cost.

Table I. Area-delay comparisons

Converters	FA	HA	Delay
CIII	$5n+3$	$2n+1$	$(4n+6)t_{FA}+t_{MUX}$
CI	$5n$	$n+4$	$2nt_{FA}+(n+2)t_{HA}+t_{MUX}$
CII	$6n-1$	$n+5$	$2nt_{FA}+(n+2)t_{HA}+2t_{MUX}$

We also carried out an experimental assessment by implementing the proposed converters and CIII using Xilinx ISE 10.1 software on a Xa3s200-4vqg100 FPGA. The results obtained after design place and route are given in terms of the number of FPGA slices and input-to-output propagation delays (in nano seconds). Table II presents the results for various dynamic range requirements (different values of n). Contrary to the theoretical analysis, the results indicate that, on average, the proposed converter CI reduces the area by about 42% when compared with the current most effective CIII converter, with a small improvement in the speed of conversion. However, the proposed full-range converter CII is about 29.48% smaller, still with some speed improvement over CIII, but lower than the one achieved by CI.

Table II. Implementation results: area-delay comparison

n	3	4	5	8
CI Area	22	27	34	57
CII Area	30	35	44	74
CIII Area	44	43	55	94
CI Delay (ns)	18.401	21.276	25.621	33.604
CII Delay (ns)	19.482	21.429	25.273	34.111
CIII Delay (ns)	22.143	22.331	24.126	34.419
CI Area-Delay	404.822	574.452	871.114	1915.428
CII Area-Delay	584.46	750.015	1112.012	2524.214
CIII Area-Delay	974.292	960.233	1326.930	3235.386

VI. Conclusions

In this paper, we proposed two new memoryless residue to binary converters for the moduli set $\{2^{n+1}-1, 2^n, 2^n-1\}$. First, we simplified the CRT to obtain a reverse converter that requires mod-$(2^{n+1}-1)$ instead of both of mod-$(2^{n+1}-1)$ and mod-(2^n-1) required by state of the art converter. Second, we further reduced the resulting architecture in order to obtain a reverse converter that utilizes only CSAs and CPAs. We resolved the dynamic range restriction problem and proposed another converter, which is valid for the entire dynamic range. The two proposed converters have been demonstrated to have lower area cost than the most effective equivalent state of the art converter with no delay penalty.

References

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

[2] M. A. Y. Wang, X. Song 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.

[3] P. 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.

[4] 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.

[5] A. Vinod and A. Premkumar, "A memoryless residue to binary converter for the 4-superset $\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$," *Journal of Circuits, Syst. and Computers, Vol. 10, pp. 85-99,*, 2000.