Efficient RNS to binary converters for the new 4-moduli set $\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$

Mohammad Esmaeildoust[1a)], **Keivan Navi**[1], **MohammadReza Taheri**[2], **Amir Sabbagh Molahosseini**[3], **and Siavash Khodambashi**[1]

[1] *Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran*

[2] *Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran*

[3] *Department of Computer Engineering, Islamic Azad University, Kerman Branch, Kerman, Iran*

a) *m_doust@sbu.ac.ir*

Abstract: In this paper, we propose efficient designs of residue number system (RNS) to binary converter for the balanced moduli set $\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ where n has even values. This new moduli set is completely free from modulo-(2^k+1)-type which results in high-speed modulo arithmetic channels for RNS. Also, mixed-radix conversion (MRC) algorithm is used to achieve both an arithmetic-based and reduced-complexity two-level RNS to binary converter architectures. The proposed moduli set provides fast arithmetic operation with higher speed of the reverse converter comparing to other five moduli set which is found in literature.

Keywords: computer arithmetic, residue number system (RNS), RNS to binary (reverse) converter, moduli sets

Classification: Integrated circuits

References

[1] K. Navi, A. S. Molahosseini, and M. Esmaeildoust, "How to Teach Residue Number System to Computer Scientists and Engineers," *IEEE Trans. Educ.*, vol. 54, no. 1, pp. 156–163.

[2] J. C. Bajard and L. Imbert, "A Full RNS Implementation of RSA," *IEEE Trans. Comput.*, vol. 53, no. 6, pp. 769–774, 2004.

[3] 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. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 9, pp. 775–779, 2007.

[4] A. S. Molahosseini, C. Dadkhah, K. Navi, and M. Eshghi, "Efficient MRC-Based Residue to Binary Converters for the New Moduli Sets $\{2^{2n}, 2^n-1, 2^{n+1}-1\}$ and $\{2^{2n}, 2^n-1, 2^{n-1}-1\}$," *IEICE Trans. Inf. & Syst.*, vol. E92-D, no. 9, pp. 1628–1638, 2009.

© IEICE 2012
DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012

[5] P. V. A. Mohan and A. B. Premkumar, "RNS-to-Binary Converters for Two Four-Moduli Set $\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$ and $\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 6, pp. 1245–1254, 2007.
[6] P. V. A. Mohan, "New reverse converters for the moduli set $\{2^n-3, 2^n-1, 2^n+1, 2^n+3\}$," *Elsevier Journal of Electronics and Communications (AEU)*, vol. 62, no. 9, pp. 643–658, 2008.
[7] A. S. Molahosseini, K. Navi, C. Dadkhah, O. Kavehei, and S. Timarchi, "Efficient Reverse Converter Designs for the new 4-Moduli Set $\{2^n-1, 2^n, 2^n+1, 2^{2n+1}-1\}$ and $\{2^n-1, 2^n+1, 2^{2n}, 2^{2n}+1\}$ Based on New CRTs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 4, pp. 823–835, 2010.

1 Introduction

The Residue number system (RNS) is an alternative number system in the applications such as Image processing, Digital Signal Processing (DSP), FIR Filters, low power applications, cryptography and other similar applications that require operations like addition, subtraction and multiplication [1, 2]. The RNS mainly consists of three main parts that includes binary to RNS (forward) converter, arithmetic operation and RNS to binary (reverse) converter [1]. Within these three parts, RNS to binary conversion is a difficult process which attracts a lot of attention from researchers. Efficiency of the RNS to binary converter is largely dependent on the form of the moduli set and also the chosen algorithm. The most prominent moduli set was $\{2^n-1, 2^n, 2^n+1\}$ [1]. Operations in moduli 2^n+1 is a time consuming process compared to 2^n and 2^n-1. Therefore, the total speed of RNS arithmetic unit is restricted to this modulo. In order to eliminate modulo 2^n+1, other three moduli sets such as $\{2^n, 2^{n+1}-1, 2^n-1\}$ [3] and $\{2^{2n}, 2^n-1, 2^{n\pm1}-1\}$ [4] have been introduced. The provided dynamic range by these moduli set as well as their degree of parallelism is not sufficient for modern applications. Hence, balanced four moduli sets such as $\{2^n-1, 2^n, 2^n+1, 2^{n+1}\pm1\}$, $\{2^n-3, 2^n-1, 2^n+1, 2^n+3\}$ [5, 6] have been introduced. These moduli sets included moduli in form of 2^k+1 or 2^k+3 which cause to inefficient arithmetic operation. In this paper, we introduce the new 4-moduli set $\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ for even n. This moduli set is free from modulo (2^k+1)-type and also includes balanced moduli, resulting in fast internal modulo arithmetic circuits for RNS. Moreover, an efficient two-level design of RNS to binary converter for this new set is presented.

2 Related background

An integer number X in the range $[0, M]$ in residue number system can be represented as $X = (x_1, x_2, \ldots, x_n)$, defined over relatively prime moduli set $\{P_1, P_2, \ldots, P_n\}$ where $x_i = X \bmod P_i$, $0 \le x < P_i$, and $M = P_1P_2\ldots P_n$ is called dynamic range of the RNS system [1].

© IEICE 2012
DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012

By using a two-channel version of MRC and 2-moduli set $\{P_1, P_2\}$, the weighed number X can be calculated from residues (x_1, x_2) by using

$$X = v_1 + v_2 P_1 \tag{1}$$

Where

$$v_1 = x_1 \tag{2}$$

$$v_2 = \left|(x_2 - v_1)|{P_1}^{-1}|_{P_2}\right|_{P_2} \tag{3}$$

Note that, $|P_1^{-1}|_{P_2}$ shows the multiplicative inverse of P_1 in modulo P_2 [1].

3 The proposed RNS to binary converter

This section describes the proposed RNS to binary converters for the moduli set $\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$. The converters have a two-level hardware architectures. The first level of each one considers the subset $\{2^n, 2^{n+1}-1, 2^n-1\}$ and is based on the efficient implementation of the work reported in [3]. Also, the second level uses the superset $\{2^n(2^{n+1}-1)(2^n-1), 2^{n-1}-1\}$ to achieve final converter.

3.1 The converter design-1

Consider the subset $\{2^n, 2^{n+1}-1, 2^n-1\}$ with corresponding weighted number $Z = (x_1, x_2, x_3)$. The process to calculate Z with its hardware implementation is described in [3]. Next, on the second level and based on MRC Eqs. (1)–(3), we have the following conversion equations for the composite set $\{2^n(2^{n+1}-1)(2^n-1), 2^{n-1}-1\}$ with corresponding weighted number $X = (Z, x_4)$:

$$X = Z + 2^n(2^{n+1}-1)(2^n-1)v_2 \tag{4}$$

Where

$$v_2 = \left|(x_4 - Z)|{P_{123}}^{-1}|_{P_4}\right|_{2^{n-1}-1} \tag{5}$$

Next, Lemma-1 calculates the needed multiplicative inverse.
Lemma-1: The multiplicative inverse of $2^n(2^{n+1}-1)(2^n-1)$ in modulo $2^{n-1}-1$ is as below

$$|{P_{123}}^{-1}|_{P_4} = \left|\frac{2^{n-2}}{3}\right|_{2^{n-1}-1} \tag{6}$$

Where $|{P_{123}}^{-1}|_{P_4}$ is the multiplicative inverses of $2^n(2^{n+1}-1)(2^n-1)$ in modulo $2^{n-1}-1$.
Proof:

$$\begin{aligned}|{P_{123}}^{-1} \times 2^n(2^{n+1}-1)(2^n-1)|_{2^{n-1}-1} &= \left|\frac{2^{n-2}}{3} \times 2(3)(1)\right|_{2^{n-1}-1} \\ &= |2^{n-1}|_{2^{n-1}-1} = 1\end{aligned}$$

Therefore, by substituting the value of multiplicative inverse in Eq. (5), we have

$$v_2 = \left|(x_4 - Z)\frac{2^{n-2}}{3}\right|_{2^{n-1}-1} \tag{7}$$

Division by 3 in Eq. (7), can be eliminated by considering the following

© IEICE 2012
DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012

equation:

$$\left|\frac{1}{3}\right|_{2^{n-1}-1} = \frac{2^n-1}{3} = \sum_{i=0}^{(n/2)-1} 2^{2i} \tag{8}$$

Thus, v_2 can be calculated by expanding Eq. (8) in (7):

$$v_2 = |(x_4 - Z) \times 2^{n-2}(2^0 + 2^2 + \cdots + 2^{n-2})|_{2^{n-1}-1} \tag{9}$$

Since, Z is a $(3n+1)$-bit number, it should be partitioned into $(n-1)$-bit blocks as follows

$$\begin{aligned} v_2 &= \left|(x_4 - \overbrace{\underbrace{00\cdots0}_{n-5\,\text{bits}}Z}^{K}) \times (2^{n-2} + 2^n + \cdots + 2^{2n-4})\right|_{2^{n-1}-1} \\ &= |(x_4 + \bar{K}_1 + \bar{K}_2 + \bar{K}_3 + \bar{K}_4) \times (2^{n-2} + 2^n + \cdots + 2^{2n-4})|_{2^{n-1}-1} \end{aligned} \tag{10}$$

Where

$$\begin{aligned} K_1 &= Z_{n-2}\cdots Z_0 \\ K_2 &= Z_{2n-3}\cdots Z_{n-1} \\ K_3 &= Z_{3n-4}\cdots Z_{2n-2} \\ K_4 &= \underbrace{0\cdots0}_{n-5\,\text{bit}} Z_{3n}\cdots Z_{3n-3} \end{aligned}$$

To achieve an efficient hardware implementation for Eq. (10), first we add the four parts of the inversion of K together with x_4 by using three $(n-1)$-bit carry-save adders (CSAs) with end-around carries (EACs) as shown in Fig. 1-a.

Next, the result of the carry-save addition can be substituted in Eq. (10) as below

$$v_2 = |(S_1 + C_1) \times (2^{n-2} + 2^n + \cdots + 2^{2n-4})|_{2^{n-1}-1} \tag{11}$$

The multiplications required by Eq. (11) can be simply realized using circular left shifting (Property 2). So,

$$v_2 = \left| \begin{array}{l} CLS(S_1, n-2) + \cdots + CLS(S_1, 2n-4) \\ +CLS(C_1, n-2) + \cdots + CLS(C_1, 2n-4) \end{array} \right|_{2^{n-1}-1} \tag{12}$$

Where CLS(Y, k) denotes k-bit circular left shifting of Y. For an instance

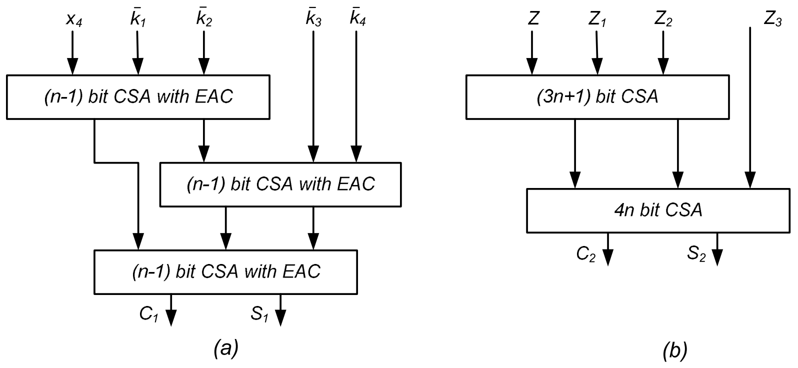

Fig. 1. Hardware schema. (a) calculation of S_1 and C_1, (b) calculation of S_2 and C_2.

© IEICE 2012
DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012

ELectronics EXpress

$\mathrm{CLS}(\mathrm{S}_1, 2n-4)$ denotes $(2n-4)$-bit circular left shift of S_1.

Now, since Z has $(3n+1)$ bits and v_2 has $(n-1)$ bits, we can simplify Eq. (4) as follows

$$\begin{aligned} X &= Z + (2^{3n+1} - 2^{2n+1} - 2^{2n} + 2^n)v_2 \\ &= Z + 2^{3n+1}v_2 - 2^{2n+1}v_2 - 2^{2n}v_2 + 2^n v_2 \\ &= Z + 2^n(2^{2n+1}v_2 + v_2) - 0v_2\underbrace{00\cdots0}_{2n+1} - 00v_2\underbrace{00\cdots0}_{2n} \\ &= Z + v_2\underbrace{00\cdots0}_{n+2}v_2\underbrace{00\cdots0}_{n} + (1\bar{v}_2\underbrace{11\cdots1}_{2n+1}+1) + (11\bar{v}_2\underbrace{11\cdots1}_{2n}+1) \\ &= v_2\underbrace{00\cdots0}_{n+2}v_2\underbrace{00\cdots0}_{n} + Z + 1\bar{v}_2\underbrace{11\cdots1}_{2n+1}+11\bar{v}_2\underbrace{11\cdots1}_{2n}+\underbrace{00\cdots0}_{3n-1\text{ bits}}10 \\ &= v_2\underbrace{00\cdots0}_{n+2}v_2\underbrace{00\cdots0}_{n-2}10 + Z + 1\bar{v}_2\underbrace{11\cdots1}_{2n+1} + 11\bar{v}_2\underbrace{11\cdots1}_{2n} \\ &= Z + Z_1 + Z_2 + Z_3 \end{aligned} \tag{13}$$

Where

$$\begin{aligned} &Z_1 = 1\bar{v}_2\underbrace{11\cdots1}_{2n+1\text{ bits}},\ Z_2 = 11\bar{v}_2\underbrace{11\cdots1}_{2n\text{ bits}}, \\ &Z_3 = v_2\underbrace{00\cdots0}_{n+2}v_2\underbrace{00\cdots0}_{n-2}10. \end{aligned} \tag{14}$$

To implement Eq. (13), first, three $(3n+1)$-bit CSAs are used as shown in Fig. 1-b. Finally, by considering the result of carry-save addition (S_2 and C_2), Eq. (13) can be computed using this equation:

$$X = 2^{3n+1}v_2 + C_2 + S_2 \tag{15}$$

Total Hardware architecture of the proposed RNS to binary converter is shown in Fig. 2-a. The block Operand Preparation Unit 1 (OPU1) performs the negation required in Eq. (10). The structures of the first and second stages are shown in Figure 1. The OPU2 performs circular left shifting operations required by Eq. (12). After that, Modulo $(2^{n-1}-1)$ adder with End around Carry ($\mathrm{MA}(2^{n-1}-1)$ with EAC) is used to calculate v_2. The OPU3 includes some wire and NOT gates to perform the required operations in Eq. (14). In the last step, a $(3n+1)$-bit adder is employed to calculated the $(3n+1)$-bit Least Significant Bits (LSBs) of the weighted number X. With concatenation of this result at the end of the v_2 the final weighted number X will be achieved.

3.2 The converter design-2

In order to decrease the number of CSAs with EAC which are needed in Eq. (12), a $\mathrm{MA}(2^{n-1}-1)$ can be employed to calculate the summation of S_1 and C_1. Therefore, Eq. (11) can be rewritten as

$$\begin{aligned} v_2 &= |R \times (2^{n-2} + 2^n + 2^{n+2} + \cdots + 2^{2n-4})|_{2^{n-1}-1} \\ &= |CLS(R, n-2) + \cdots + CLS(R, 2n-4)|_{2^{n-1}-1} \end{aligned} \tag{16}$$

Where

$$R = |S_1 + C_1|_{2^{n-1}-1} \tag{17}$$

Applying this change in hardware architecture, the number of CSA-tree in-

© IEICE 2012
DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012

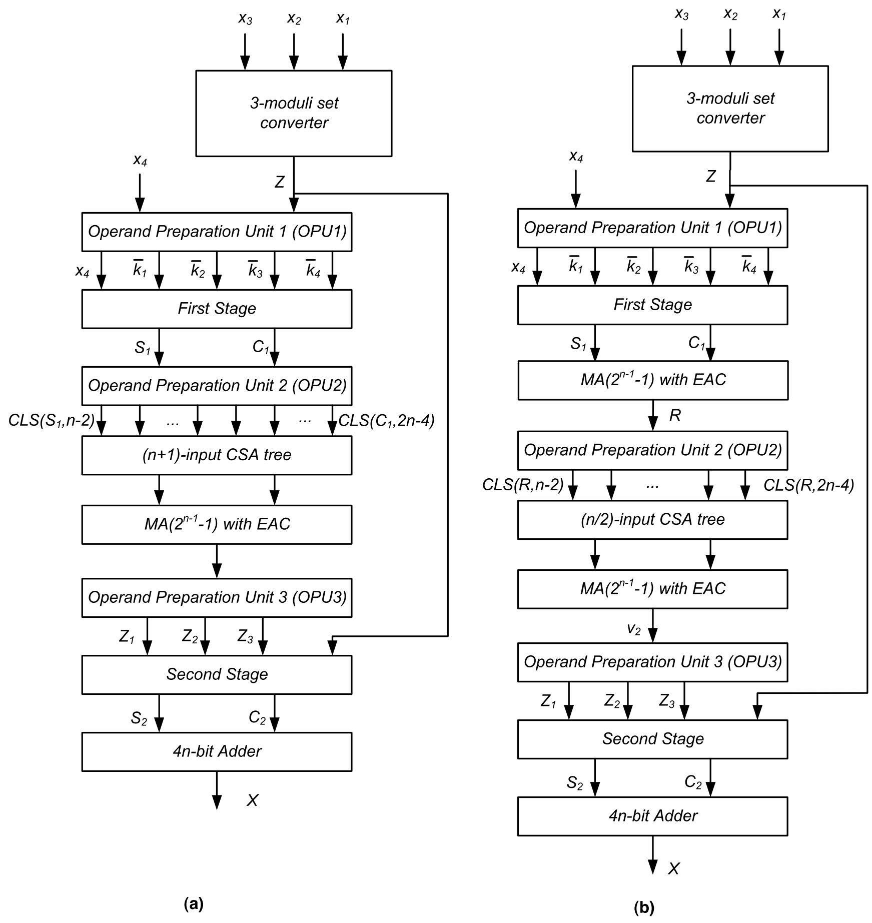

Fig. 2. Hardware schema. (a) calculation of S_1 and C_1, (b) calculation of S_2 and C_2.

puts reduces to $(n/2)$ with delay of only one $\mathrm{MA}(2^{n-1}-1)$. Hardware architecture of the design taking this change into account is shown in Fig. 2-b.

4 Performance comparison

This section compares the performance of the proposed RNS to binary converter architectures for the proposed moduli set with the performance of latest RNS to binary converters for other 4-moduli sets with the same dynamic range class reported in [5, 6]. The comparisons are done in terms of speed of the arithmetic operation, delay and area of the RNS to binary conversion. The assumption for the calculation of the hardware requirements are the same as in [7]. Also, in [3], three different design of the RNS to binary converter for moduli set $\{2^n, 2^{n+1}-1, 2^n-1\}$ are presented which are named C-I, C-II and C-III. Therefore, with use of each of these converters on the first level of the proposed designs, different versions can be achieved. Table I shows the conversion delays and hardware requirements of the proposed converters and other designs. The results show the noticeable improvement in terms of speed of the RNS to binary converter compared to other designs.

Note that, in [6], HS versions of the RNS to binary converters are presented. This high-speed method can be used in any RNS to binary converter

DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012

Table I. Delay and hardware requirement for the different reverse converters.

Moduli Set	Hardware requirements	Delay
$\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$[5]	$(9n + 5 + ((n-4)(n+1)/2))A_{FA} + 2nA_{ex\text{-}Nor} + 2nA_{OR} + (6n+1)A_{INV}$	$(23n+12)/2\ D_{FA}$
$\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$ [5]-Using ROM	$(6n+1)A_{INV} + (8n+4)A_{FA} + 2nA_{ex\text{-}Nor} + 2nA_{OR} + (n+1)2^{n+1}A_{ROM}$	$(9n+6)\ D_{FA}$
$\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$ [5]	$(6n+7)A_{INV}+(n^2+12n+12)A_{FA} + 2nA_{EXNOR} + 2nA_{OR}+(4n+8)A_{2:1MUX}$	$(16n+22)\ D_{FA}$
$\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$ [5]–Using ROM	$(5n+6)A_{INV} + (9n+10)A_{FA} + 2nA_{EXNOR} + 2nA_{OR} + (2n+2)A_{2:1MUX} + (n+2)2^{n+2}A_{ROM}$	$(11n+14)\ D_{FA}$
$\{2^n-3, 2^n-1, 2^n+1, 2^n+3\}$ [6]-C1 CE	$(25.5n + 12 + (5n^2/2))A_{FA} + 5nA_{HA} + 3nA_{EXNOR} + 3nA_{OR}$	$(18n+23)\ D_{FA}$
$\{2^n-3, 2^n-1, 2^n+1, 2^n+3\}$ [6]-C2 CE	$(20n+17)A_{FA} + (3n-4)A_{HA} +2^n(5n+2)A_{ROM}$	$(13n+22)\ D_{FA} + 3\ D_{ROM}$
$\{2^n-3, 2^n-1, 2^n+1, 2^n+3\}$ [6]-C3 CE	$(23n+11)A_{FA} + (2n-2)A_{HA} +(6n+4)2^nA_{ROM}$	$(16n+14)\ D_{FA} + D_{ROM}$
$\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ Design1-Using C-I	$(n^2+16n+6)\ A_{FA} + (n+2)A_{XNOR} + (3n-5)A_{AND} + (n+2)A_{OR} + (3n-5)A_{XOR} + 4n\ A_{INV}$	$(12n+9+q)\ D_{FA}$**
$\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ Design1-Using C-II	$(n^2+24n+24)\ A_{FA}+ (2n+3)A_{HA} + 2A_{XNOR} + (2n-5)A_{AND} + 2A_{OR} + (2n-5)A_{XOR}$ $(2n+1)A_{3:1MUX} + 4n\ A_{INV}$	$(8n+11+q)\ D_{FA}$**
$\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ Design1-Using C-III	$(n^2+22n+22)\ A_{FA} + (2n+2)A_{HA} + 2A_{XNOR} + (2n-5)A_{AND} + 2A_{OR} + (2n-5)A_{XOR} + 10(2n+1)A_{ROM} + (2n+1)A_{2:1MUX}+ 4n\ A_{INV}$	$(8n+11+q)\ D_{FA}$**
$\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ Design2-Using C-I	$(n^2/2 + 7n/2 + 20n + 7)A_{FA} + (n+2)A_{XNOR} + (3n-5)A_{AND} + (n+2)A_{OR} + (3n-5)A_{XOR} + 4n\ A_{INV}$	$(14n+7+p)\ D_{FA}$**
$\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ Design2-Using C-II	$(n^2/2 + 7n/2 + 28n + 23)A_{FA}+ (2n+3)A_{HA} + 2A_{XNOR} + (2n-5)A_{AND} + 2A_{OR} + (2n-5)A_{XOR}$ $(2n+1)A_{3:1MUX} + 4n\ A_{INV}$	$(10n+9+p)\ D_{FA}$**
$\{2^n, 2^{n+1}-1, 2^n-1, 2^{n-1}-1\}$ Design2-Using C-III	$(n^2/2 + 7n/2 + 26n + 21)A_{FA}+ (2n+2)A_{HA} + 2A_{XNOR} + (2n-5)A_{AND} + 2A_{OR} + (2n-5)A_{XOR} + 10(2n+1)A_{ROM} + (2n+1)A_{2:1MUX}+ 4n\ A_{INV}$	$(10n+9+p)\ D_{FA}$**

** p and q are the number of levels in CSA tree of $n/2$ inputs and $(n + 1)$ inputs, respectively.

which includes modulo $(2^k - 1)$ adders. Therefore for ease of comparison, these HS versions are not listed in the Table I. In some cases the proposed designs rely on more hardware. But, as described in introduction, the frequency of performing arithmetic operations in RNS is much more than RNS to binary conversion. Hence, speed of modulo arithmetic operations is very important parameter in RNS. As mentioned before modulo of the kind $(2^k + 1)$ can reduce the total efficiency of the RNS arithmetic unit. The proposed moduli set is free from this type of modulo. Therefore our moduli set can lead to efficient arithmetic operations for RNS and also provides high dynamic range for application like cryptography and DSP [1, 2] with more efficient arithmetic operation comparing to other moduli sets.

5 Conclusion

We have designed efficient RNS to binary converters for the new moduli set $\{2^n, 2^{n+1} - 1, 2^n - 1, 2^{n-1} - 1\}$. This moduli set is free from modulo $(2^k + 1)$-type and can provide fast RNS arithmetic unit, due to its balanced moduli. The proposed converters have a two-level structure and has improved the conversion delay with full adder-based implementations.

DOI: 10.1587/elex.9.1
Received September 26, 2011
Accepted December 02, 2011
Published January 10, 2012