where n is the stage gain, f is the clock frequency, V_{DD} is the supply voltage, C_{CLK} is the clock capacitance, N is the number of flip-flops loading the tree, C_{FF} is the flip-flop loading capacitance, α is the factor by which this capacitance is reflected to the driver side, P_{FF_sq} is the power consumption of the flip-flop with the 1-V supply, R_{CLK} is the resistance of the clock wires, V_{OH} and V_{OL} are the highest and lowest peaks of the IPCDN signals, respectively, and P_{FF_IPCDN} is the power consumption of the flip-flop with the IPCDN and shared buffer. Note that the factor 2 at the beginning of (6) is due to the two differential networks present in IPCDN.

The number of flip-flops and the size of the network are dependent on each system. Fig. 9 is a 3-D plot of the percentage reduction in power achieved with IPCDN compared to that achieved with the square-wave CDN as a function of the clock capacitance (C_{CLK}) and the number of flip-flops (N). As illustrated in the figure, with a heavily loaded network, IPCDN achieves around 20% reduction in power. When the network capacitance is dominating, IPCDN can achieve up to 50% reduction in power.

IV. Conclusion

We proposed an IPCDN in which the power and clock networks were combined into one network. Simulation results showed correct operation of the LC differential clock driver and clock buffer with TSMC 65-nm CMOS technology at a frequency of 5 GHz. In order to satisfy high current requirements, several LC drivers should be distributed to drive different blocks. This would increase area overhead associated with the inductors. Implementing inductors on top of active metal layers can reduce the area overhead with a penalty of reduced quality factor. Magnetic inductors are another alternative to reduce area.

Comparing IPCDN to a buffered square-wave CDN illustrates that IPCDN achieves around 20% reduction in power when the network is heavily loaded.

References

[1] *International Technology Roadmap for Semiconductors*. (2011) [Online]. Available: http://public.itrs.net

[2] S. D. Naffziger and G. Hammond, "The implementation of the next-generation 64 b Itanium^{TM} microprocessor," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2002, pp. 344–472.

[3] V. F. Pavlidis, I. Savidis, and E. G. Friedman, "Clock distribution networks in 3-D integrated systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 12, pp. 2256–2266, Dec. 2011.

[4] R. Jakushokas and E. G. Friedman, "Globally integrated power and clock distribution network," in *Proc. IEEE Int. Symp. Circuits Syst.*, Mar. 2010, pp. 1751–1754.

[5] S. E. Esmaeili, A. J. Al-Khalili, and G. E. R. Cowan, "Estimating required driver strength in the resonant clock generator," in *Proc. IEEE Asia Pacific Conf. Circuits Syst.*, Dec. 2010, pp. 927–930.

[6] E. Talpes and D. Marculescu, "Toward a multiple clock/voltage island design style for power-aware processors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 5, pp. 591–603, May 2005.

[7] S. M. Kang and Y. Leblebici, *CMOS Digital Integrated Circuits*, 3rd ed. New York: McGraw-Hill, 2003, pp. 260–263.

[8] A. J. Drake, K. J. Nowka, T. Y. Nguyen, J. L. Burns, and R. B. Brown, "Resonant clocking using distributed parasitic capacitance," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Jul. 2004, pp. 1520–1528.

On the Design of RNS Reverse Converters for the Four-Moduli Set $\{2^n + 1, 2^n - 1, 2^n, 2^{n+1} + 1\}$

Leonel Sousa, Samuel Antão, and Ricardo Chaves

***Abstract*— In this brief, we propose a method to design efficient adder-based converters for the four-moduli set $\{2^n + 1, 2^n - 1, 2^n, 2^{n+1} + 1\}$ with n odd, which provides a dynamic range of $4n + 1$ bits for the residue number system (RNS). This method hierarchically applies the mixed radix approach to balanced pairs of residues in two levels. With the proposed method, only simple binary and modulo $2^k - 1$ additions are required, fully avoiding the usage of modulo $2^k + 1$ arithmetic operations, which is a significant advantage over the currently available RNS reverse converters for this type of moduli set. Experimental results show that the delay of the proposed converters is significantly reduced when compared with the related state of the art; for example, for a 65-nm CMOS ASIC technology and a dynamic range of 21 bits, the conversion time and the circuit area are reduced by about 44% and 30%, respectively, while the conversion time is reduced by 34% for a dynamic range of 37 bits with the circuit area increasing only by 25%. Moreover, the proposed reverse converters outperform the related state of the art for any value of n by up to 70%, according to the figure-of-merit energy per conversion.**

***Index Terms*— Application-specific integrated circuit (ASIC), digital hardware design, field-programmable gate array (FPGA), residue number system (RNS), reverse conversion.**

I. Introduction

The interest in residue number systems (RNS) has grown considerably over the last several years [1]. These systems have been proposed as an alternative to binary systems, namely, for multiplying and adding/subtracting large numbers, with direct applications in, for example, signal processing [2] including cryptography [3]. However, the direct computation of the residues and the reverse conversion from the RNS back to positional notation are computationally intensive operations, particularly the latter one. The use of specific power-of-two related pairwise relatively prime moduli sets makes these conversions simpler by taking advantage of well-known arithmetic units for the binary system [4]–[7]. The traditional three-moduli set $\{2^n - 1, 2^n, 2^n + 1\}$ has interesting properties and therefore has been extensively investigated. However, extensions to this moduli set have been proposed to increase the dynamic range, such as the balanced four-moduli sets [8]–[11], five-moduli sets [12], [13], and the ones based on the utilization of a fractional representation [14].

The residue to binary (reverse) conversion is based on the chinese remainder theorem (CRT) [15] or on the mixed-radix conversion (MRC) method [16].

Considering $\{x_1, \ldots, x_k\}$ the RNS residues of X, the MRC (1) avoids the usage of the computational demanding modulo M operations, despite the computations of the mixed radix digits (a_i) being done in a serial manner. Still, it is usually adopted when the moduli set $\{m_1, \ldots, m_k\}$ grows and the dynamic range becomes larger

$$X = a_1 + a_2 \times m_1 + a_3 \times m_1 m_2 + \cdots + a_k \times \prod_{i=1}^{k-1} m_i$$

Manuscript received January 4, 2012; revised June 20, 2012; accepted August 31, 2012. Date of publication November 16, 2012; date of current version September 9, 2013. This work was supported by the National Funds through the Fundação Para a Ciência e a Tecnologia (FCT), under Project PEst-OE/EEI/LA0021/2011.

The authors are with the Instituto Superior Técnico, Universidade Técnica de Lisboa, and INESC-ID, Lisboa 1000-029, Portugal (e-mail: las@inesc-id.pt; sfan@sips.inesc-id.pt; ricardo.chaves@inesc-id.pt).

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.2012.2219564

$$a_1 = x_1;\ \ a_2 = \left|(x_2 - a_1)\left|m_1^{-1}\right|_{m_2}\right|_{m_2};\ \ \dots;$$
$$a_k = \left|((\cdots(x_k - a_1)\left|m_1^{-1}\right|_{m_k} - a_2)\left|m_2^{-1}\right|_{m_k} - \cdots - a_{k-1})\left|m_{k-1}^{-1}\right|_{m_k}\right|_{m_k}. \quad (1)$$

This brief is focused on the design of reverse converters for the balanced four-moduli set $S = \{2^n+1, 2^n-1, 2^n, 2^{n+1}+1\}$, which is a pairwise relatively prime moduli set for n odd [10] with a dynamic range of $4n+1$ bits. The proposed method for designing the converters applies the MRC technique to two balanced two-moduli sets and relies on the properties of the power-of-two related moduli, which can be efficiently exploited when the moduli are organized in proper small sets of only two elements. With this approach, it is easy to automate the design of residue-to-binary converters based on adders, and pipeline its operation to increase their throughput.

This brief is organized as follows. The proposed method for performing reverse conversion from RNS to binary is presented in Section II. In Section III, this method is applied to design efficient VLSI reverse converters, whose relative performance is also assessed. Section IV experimentally evaluates the efficiency of the proposed and corresponding state-of-the-art converters to assess their relative performance. Section V concludes this brief.

II. Proposed Design Method for the Conversion

The following notation, based on [17], is adopted throughout the rest of this brief.

1) For an n-bit value γ, bits are referred from the most significant bit (MSB) to the least significant bit (LSB) as $\gamma[n-1], \dots, \gamma[0]$.
2) $\gamma^l[k]$ refers to an l-bit number such that
$$\gamma^l[k] = \gamma[k+l-1]2^{l-1} + \cdots + \gamma[k+1]2 + \gamma[k].$$
3) $\mathcal{O}$ and $\mathcal{Z}$ refer to a number whose binary representation is an all-1 and all-0 string, respectively (note that with the introduced notation $\mathcal{O}^l[k] = \mathcal{O}^l[k']$ and $\mathcal{Z}^l[k] = \mathcal{Z}^l[k']$ for any value of k and k').
4) The symbol $\bowtie$ operates the concatenation of the binary representation of two numbers.

The method for designing efficient reverse converters starts with partitioning the original moduli set into two-moduli subsets: $S_{12} = \{m_1, m_2\} = \{2^n+1, 2^n-1\}$ and $S_{43} = \{m_4, m_3\} = \{2^{n+1}+1, 2^n\}$. Applying the MRC (1) to S_{12} ($\{x_1, x_2\}$ denotes the residues for this subset), and to S_{43} ($\{x_4, x_3\}$ denotes the residues for this subset) results in

$$X_{12} = x_1 + \omega_{12} \times (2^n + 1) \quad (2)$$
$$X_{43} = x_4 + \omega_{43} \times (2^{n+1} + 1) \quad (3)$$

with

$$\omega_{12} = \left|(x_2 - x_1) \times \left|(2^n+1)^{-1}\right|_{2^n-1}\right|_{2^n-1} \quad (4)$$
$$\omega_{43} = \left|(x_3 - x_4) \times \left|(2^{n+1}+1)^{-1}\right|_{2^n}\right|_{2^n}. \quad (5)$$

The final binary representation (X) can then be computed by considering $\{X_{43}, X_{12}\}$ the representation of X in the moduli sets $\{m_{43}, m_{12}\}$, with $m_{43} = m_4 \times m_3$ and $m_{12} = m_1 \times m_2$, which results in $\{m_{43}, m_{12}\} = \{2^{2n+1} + 2^n, 2^{2n} - 1\}$. Therefore, by applying (1), the binary representation X can be obtained as

$$X = X_{43} + \Omega \times (2^{2n+1} + 2^n) \quad (6)$$

TABLE I
Numerical Example for $n = 5$ and $X = 1692112$ of the Adopted Conversion Method

Variable	Decimal val.	Binary val.
x_1, x_2, x_3, x_4	4, 8, 16, 32	000100, 01000, 10000, 0100000
X_{43} (10), ω_{43} (9)	1072, 16	010000110000, 10000
T_1 (13), T_2 (14)	961, 132	1111000001, 0010000100
θ (16), φ (15)	331, 21	0101001011, 0000010101
Ω (7)	813	1100101101
X (21)	1692112	11001110100011110100000

with

$$\Omega = \left||X_{12} - X_{43}|_{2^{2n}-1} \times \left|(2^{2n+1} + 2^n)^{-1}\right|_{2^{2n}-1}\right|_{2^{2n}-1}. \quad (7)$$

It can be noticed that the value X_{12} is not directly applied to compute X, but only indirectly through Ω. To compute ω_{43} in (5), it can be easily shown that

$$\left|(2^{n+1}+1)^{-1}\right|_{2^n} = 1. \quad (8)$$

Consequently, applying (8) to (5) results in

$$\omega_{43} = \left|x_3 + \overline{x_4^n[0]} + 1\right|_{2^n} \quad (9)$$

and by applying (9) to (3) we obtain

$$X_{43} = x_4 + \omega_{43} \bowtie \mathcal{Z}[0] \bowtie \omega_{43}. \quad (10)$$

The value Ω in (7) is computed with arithmetic modulo $2^{2n} - 1$. Therefore, it is convenient to manipulate (2) in order to express X_{12} in this modulo. The following equality is required to compute ω_{12}:

$$\left|(2^n+1)^{-1}\right|_{2^n-1} = 2^{n-1} \quad (11)$$

since

$$\left|(2^n+1) \times 2^{n-1}\right|_{2^n-1} = \left|2 \times 2^{n-1}\right|_{2^n-1} = \left|2^n\right|_{2^n-1} = 1.$$

Given that $X_{12} < 2^{2n} - 1$, applying the identity $|k \times y|_{k \times z} = k \times |y|_z$ [18] to (2) results in

$$\begin{aligned} X_{12} &= x_1 + (2^n+1) \times \left|(x_2 - x_1) \times 2^{n-1}\right|_{2^n-1} \\ &= \left|x_1 + (x_2 - x_1) \times 2^{n-1} \times (2^n+1)\right|_{2^{2n}-1} \\ &= \left|x_2 \times (2^{2n-1} + 2^{n-1}) - x_1 \times (2^{2n-1} + 2^{n-1} - 1)\right|_{2^{2n}-1} \\ &= \left|x_2 \times (2^{2n-1} + 2^{n-1}) - x_1 \times (2^{2n-1} + 2^{n-1} - 2^{2n})\right|_{2^{2n}-1} \\ &= \left|\underbrace{\left|x_2 \times (2^{2n-1} + 2^{n-1})\right|_{2^{2n}-1}}_{T_2} + \underbrace{\left|x_1 \times (2^{2n-1} - 2^{n-1})\right|_{2^{2n}-1}}_{T_1}\right|_{2^{2n}-1} \end{aligned} \quad (12)$$

Given that a modulo $2^n - 1$ multiplication of an integer γ, represented with n-bits, by 2^s can be accomplished by circularly shifting γ by s bits to the left (note that $\left|2^n\right|_{2^n-1} = 1$), and that $|-\gamma|_{2^n-1}$ corresponds to the one's complement of γ ($\overline{\gamma}$), the terms T_1 and T_2 in (12) can be obtained as

$$\begin{aligned} T_1 &= \left|(2^{2n-1} - 2^{n-1}) \times (x_1^n[0] + x_1[n] \times 2^n)\right|_{2^{2n}-1} \\ &= \left|x_1[0] \bowtie \overline{x_1^n[0]} \bowtie x_1^{n-1}[1] + \overline{x_1[n]} \bowtie \mathcal{Z}^{n-1}[n] \bowtie x_1[n] \bowtie O^{n-1}[0]\right|_{2^{2n}-1} \end{aligned} \quad (13)$$

$$T_2 = x_2[0] \bowtie x_2 \bowtie x_2^{n-1}[1]. \quad (14)$$

To obtain (13), the following identities are considered:

$$\left|2^{2n-1}x_1^n[0]\right|_{2^{2n}-1} = x_1[0] \rtimes \mathcal{Z}^n[n-1] \rtimes x_1^{n-1}[1]$$
$$\left|-2^{n-1}x_1^n[0]\right|_{2^{2n}-1} = \mathcal{O}[2n-1] \rtimes \overline{x_1^n[0]} \rtimes \mathcal{O}^{n-1}[0]$$
$$\left|x_1[n]2^n(2^{2n-1}-2^{n-1})\right|_{2^{2n}-1} = x_1[n]\left(\mathcal{O}[2n-1] \rtimes \mathcal{Z}^n[n-1] \rtimes \mathcal{O}^{n-1}[0]\right).$$

To compute $\varphi = |X_{12} - X_{43}|_{2^{2n}-1}$, we can directly use (12), along with (13), (14), and (3)

$$\begin{aligned}\varphi &= |X_{12} - x_4 - (\omega_{43} \rtimes \mathcal{Z}[n] \rtimes \omega_{43})|_{2^{2n}-1} \\ &= \left|X_{12} + \mathcal{O}^{n-2}[n+2] \rtimes \overline{x_4} + \overline{\omega_{43}}^{n-1}[0] \rtimes \mathcal{O}[n] \rtimes \overline{\omega_{43}} + \right. \\ &\quad \left. + \mathcal{O}^{2n-1}[1] \rtimes \overline{\omega_{43}}[n-1]\right|_{2^{2n}-1} \\ &= \left|T_1 + T_2 + \overline{x_4} + \overline{\omega_{43}}^{n-1}[0] \rtimes \mathcal{O}[n] \rtimes \overline{\omega_{43}} + \overline{\omega_{43}}[n-1] + \right. \\ &\quad \left. + \mathcal{O}^{n-3}[n+3] \rtimes \mathcal{Z}[n+2] \rtimes \mathcal{O}^{n+2}[0]\right|_{2^{2n}-1}. \end{aligned} \quad (15)$$

The multiplicative inverse $\theta = |(2^{2n+1}+2^n)^{-1}|_{2^{2n}-1}$ in (7) can be computed as

$$\theta = \left|(2^{2n+1}+2^n)^{-1}\right|_{2^{2n}-1} = \left|\frac{1}{3} \times (2^{2n+2}-2^n-2)\right|_{2^{2n}-1} \quad (16)$$

$$\frac{1}{3} \times (2^{2n+2}-2^n-2) = \sum_{i=0}^{\frac{n-3}{2}} 2^{2i+1} + \sum_{i=\frac{n-1}{2}}^{n-1} 2^{2i+2}. \quad (17)$$

For the proof of (17), we compute the sum of n terms of a geometric series with common ratio r by using

$$\sum_{i=m}^{n} r^i = \frac{r^m - r^{n+1}}{1-r}. \quad (18)$$

Therefore

$$\begin{aligned}\sum_{i=0}^{\frac{n-3}{2}} 2^{2i+1} + \sum_{i=\frac{n-1}{2}}^{n-1} 2^{2i+2} &= \frac{1}{3}(2^n-2) + \frac{1}{3}(2^{2n+2}-2^{n+1}) \\ &= \frac{1}{3} \times (2^{2n+2}-2^n-2) \end{aligned} \quad (19)$$

and

$$\begin{aligned}&\left|\frac{1}{3} \times (2^{2n+2}-2^n-2) \times (2^{2n+1}+2^n)\right|_{2^{2n}-1} \\ &= \left|\frac{1}{3} \times (2^{4n+3}+2^{3n+2}-2^{3n+1}-2^{2n}-2^{2n+2}-2^{n+1})\right|_{2^{2n}-1} \\ &= \left|\frac{1}{3} \times (8+2^{n+2}-5-2^{n+2})\right|_{2^{2n}-1} = 1. \end{aligned} \quad (20)$$

Finally, since $X < 2^{4n+1}$, X can be computed by applying (7) to (6)

$$\begin{aligned}X &= \left|X_{43} + \Omega \times (2^{2n+1}+2^n)\right|_{2^{4n+1}} \\ &= \left|X_{43} + 2^{2n+1}\Omega + 2^n\Omega\right|_{2^{4n+1}}. \end{aligned} \quad (21)$$

In Table I, a numerical example is presented that summarizes the values computed with the adopted conversion approach. In the next sections, we present reverse converters for the moduli set $\{2^n+1, 2^n-1, 2^n, 2^{n+1}+1\}$, which were designed by applying the method proposed in this section, and evaluate their performance when compared with related state-of-the-art circuits.

III. Proposed Architecture

In this section, we present the architecture of the converter and theoretically evaluate its performance. To achieve such goal, we consider the total number of full adders (FAs) for evaluating the circuit area (A_{FA} is the area required by a FA), and the number of FAs in the critical path to assess the delay (D_{FA} is the delay imposed by a FA). Furthermore, we consider that the area and delay of a half adder (HA) are $A_{\mathrm{HA}} = 1/2A_{\mathrm{FA}}$ and $D_{\mathrm{HA}} = 1/2D_{\mathrm{FA}}$, respectively. The bitwise operations are ignored for area and delay analysis, as they are expected to be negligible regarding the FAs and HAs.

The block diagram in Fig. 1 represents the proposed reverse converter for the moduli set $\{2^n+1, 2^n-1, 2^n, 2^{n+1}+1\}$, based on the equations derived in the previous section. Although fast parallel-prefix modulo 2^k-1 adders with a delay proportional to $\log_2(k)$ but area proportional to $k\log_2(k)$ have been proposed [19], we consider for this analysis, as in [11], that a k-bit carry-propagate adder (CPA) with end-around carry (EAC) has twice the delay of a normal k-bit CPA, but the same area. The EAC approach is an efficient method to compute modulo 2^k-1 addition, which consists in redirecting the resulting carry-out of an addition into the carry-in.

The proposed converter is composed of four main building blocks: 1) for obtaining ω_{43} using (9); 2) for obtaining Ω using (7), (15) to compute φ and the multiplicative inverse in (16); 3) for obtaining X_{43} using (10) in parallel with the computation of Ω; and 4) for achieving X using (21).

The computation of ω_{43} is accomplished in Fig. 1(a) with a n-bit CPA requiring nA_{FA} and imposing a delay of nD_{FA}. Obtaining Ω requires the computation of φ as in (15). The implementation of (15) can be simplified by merging the term $\overline{\omega_{43}}[n-1]$ with the other two terms containing arrays of zeros and ones, such that a single term is obtained with bitwise operations which only depend on $\overline{\omega_{43}}[n-1]$ and $x_1[n]$. Therefore, the value φ is obtained with the sum of five terms, of which only two terms depend on ω_{43}. This allows removing one of the carry-save adders (CSAs) from the critical path, as depicted in Fig. 1(b), which results in a total delay of $(4n+2)D_{FA}$ for computing φ. Given that one of the CSAs with EAC has a $(n+2)$-bit operand, the resources required to obtain φ are $(7n+2)A_{FA}$ and $(n-2)A_{HA}$, totaling $(7.5n+1)A_{FA}$. In order to obtain Ω, φ is multiplied modulo $2^{2n}-1$ by the constant multiplicative inverse in (16). This constant contains n bits different from zero. Therefore, the multiplication can be obtained by performing n modular additions of $2n$-bit terms, obtained by properly left-shifting (rotating) φ. A Wallace adder tree [20] can be used to implement these multioperand modular additions. This type of tree requires approximately $\log_{1.5} k \approx 2\log_2 k$ levels for k operands, with a delay D_{FA} per level. The final carry and save vectors obtained from the Wallace adder tree are added in a $2n$-bit CPA with EAC. Summarizing, $2(n^2-n)A_{FA}$ are required to compute Ω from φ with a delay of $(2\log_2 n + 4n)D_{FA}$.

The computation of X_{43} in Fig. 1(c) with (10) requires an $(n+2)$-bit CPA and an $(n-1)$-bit CPA which conditionally increments its input depending on the carry output of the $(n+2)$-bit CPA. Hence, $n+2$ FAs and $n-1$ HAs are required for the computation of X_{43}, which results in a delay and area of $1.5(n+1)D_{FA}$ and $1.5(n+1)A_{FA}$, respectively. Note that the delay for computing X_{43} is smaller than the delay for computing φ, and thus the block that computes X_{43} is not in the critical path. Finally, the value X is obtained with a $(n+1)$-bit CPA, a 2n-bit CPA, and a $2n$-bit CSA, which are used to sum the properly aligned terms in (21). Note that the CSA only requires $0.5nA_{FA}$, due to the size of its inputs. Furthermore, since the $n+1$ MSBs of the carry output vector of the CSA are zero, the $2n$-bit CPA only requires $(1.5n-0.5)A_{FA}$ with a delay of $(1.5n-0.5)D_{FA}$. Summing up, the total resources and delay for computing

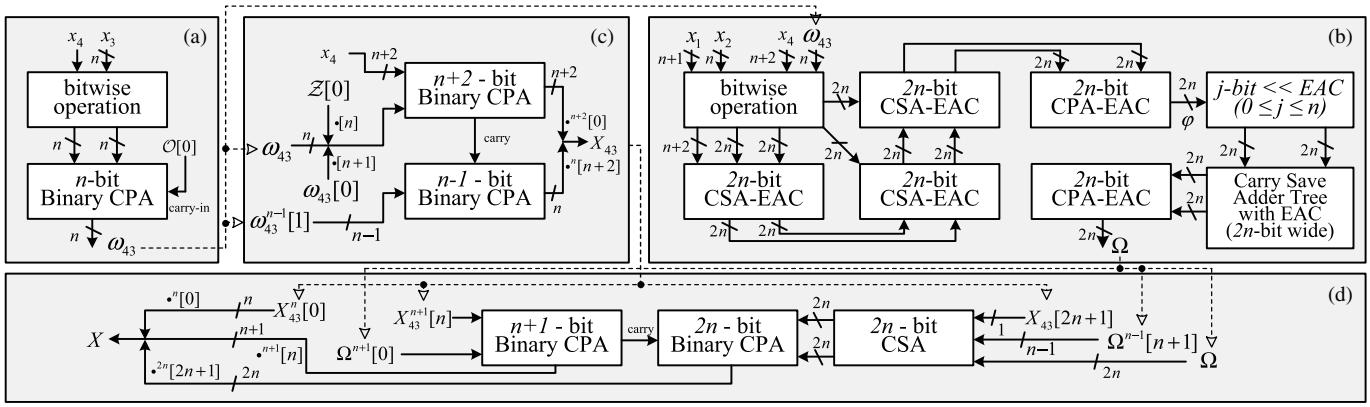

Fig. 1. Block diagram of the propose converter. (a) w_{43} is computed from x_3 and x_4 using (9). (b) From x_1, x_2, x_4, and w_{43}, computing φ using (15) and the multiplicative inverse using (16), Ω is achieved using (7). (c) X_{43} is computed using (10) from x_4 and w_{43} and in parallel with Ω. (d) X is computed from Ω and X_{43} using (21).

TABLE II
COMPARISON OF AREA AND TIME REQUIREMENTS OF THE PROPOSED CONVERTERS AND THE RELATED STATE-OF-THE-ART CONVERTERS

Converter	Area (A_{FA})	Delay (D_{FA})
Proposed	$2n^2 + 11n + 3$	$11.5n + 2\log_2 n + 2.5$
[11]	$n^2 + 14n + 15$	$15n + 2\log_2(n+2) + 22$

X from X_{43} and Ω are $(3n + 0.5)A_{FA}$ and $(2.5n + 0.5)D_{FA}$, respectively.

Table II estimates the performance results of both the proposed and related state-of-the-art converters [11]. The values estimated for the delay suggest a significant reduction with the proposed converters, with a penalty in the circuit area, when compared with the related state-of-the-art converters [11]. For example, supposing $n = 9$, the proposed converter imposes approximately 46% less delay than the converter in [11]. It should be noted that the main difference in the circuit area arises from the Wallace tree adder, which is quite regular for VLSI implementations and, therefore, may result in less circuit area than the one obtained from the simplified analysis in Table II. Therefore, we designed specific circuits to implement the converters so as to accurately assess their real performance. Experimental results are provided in the next section.

IV. EXPERIMENTAL RESULTS

The circuits of the proposed and related state-of-the-art converters were described in synthesizable VHDL. A well-known library of arithmetic units [21], also written with synthesizable VHDL, was used. This library contains a structural specification of components, namely optimized prefix adders, which were employed in the converters' description. Using these HDL specifications, implementations targeting FPGA and ASIC were accomplished. A Xilinx Virtex 5 (part xc5vlx220ff1760-2) FPGA was targeted using the Synopsys Synplicity Premier tools (version F-2011.09-SP1) for the synthesis procedure and the Xilinx ISE tools (version 13.1) for placing and routing. For the ASIC implementation, the design was supported on a TSMC 65-nm general-purpose standard cell library (TCBN65GPLUS, version 200A) tailored for the TSMC 65-nm CMOS logic salicide process (1-poly, 9-metal), using the Cadence RTL Compiler tools (version v09.10-s242_1) for synthesizing the design and the Cadence Encounter and NanoRoute tools (versions v09.12-s159 and v09.12-s013, respectively) for placing and routing. For both the FPGA and ASIC technologies, no manual optimizations were introduced. The synthesis tools were set to target minimum delay, allowing the tool to use unconstrained resources and power consumption. The presented energy-per-conversion estimated values were obtained from the placed and routed circuit specifications for a switching activity corresponding to 10 000 random inputs vectors under typical operating conditions. The Cadence Encounter built-in power reporting tool and the Xilinx power analyzer were employed to compute the required power estimations for the target ASIC and FPGA technologies, respectively. For FPGA, only the dynamic power is addressed since the quiescent power (estimated in 2185 mW) is implementation independent (depends only on the device).

We made the HDL specification of the proposed and related art converters publicly available at http://sips.inesc-id.pt/~sfan/prototypes/rnsrevconv/.

Table III presents the obtained delay (D), area (A), and energy per conversion (E/C) results for the proposed converters and the ones in [11] for two different values of n.

Though both Virtex 5 and ASIC implementations correspond to 65-nm technologies, Table III suggests that the delay of the FPGA-based implementation of the proposed RNS reverse converters is more than an order of magnitude greater than the corresponding ASIC implementations, for all configurations. These results confirm that the proposed converters can take advantage of the ASIC technologies' characteristics, namely: 1) the regularity of the main components of the converters and 2) the CSAs, for which efficient placement and routing can be obtained while the FPGAs are optimized for CPAs (i.e., fast carry chains).

The results in Table III also show that throughputs of up to 92 and 42 mega conversions per second (MC/s) can be observed for the FPGA technology, for a dynamic range of approximately $4n = 20$ and $4n = 68$, respectively. The same performance metrics boost to 1585 and 1109 MC/s for the ASIC technology. While the dynamic range increases by a factor higher than three, the throughput only decreases by a factor that is around two or less for both technologies. These results suggest that the proposed converters provide a more interesting scaling behavior with the RNS channel's width n than would be expected from the theoretical delay evaluation presented in Table II, which predicts an approximately linear delay increase. Hence, it can be expected that, for larger converters, more advantage can be taken of the device's characteristics, due to the regularity of the proposed circuits.

Fig. 2 depicts the delay, area and power improvements of the proposed converter with respect to [11]. This figure also includes the theoretical behavior predicted in Table II. In order to properly evaluate the area and energy per conversion, new implementations of the proposed converters were obtained with relaxed timing constraints so as to obtain implementations whose delay is similar to the one obtained in [11]. Therefore, the area and energy per conversion

TABLE III
PERFORMANCE FIGURES OF THE CONVERTERS FOR THE MODULI SET $\{2^n+1, 2^n-1, 2^n, 2^{n+1}+1\}$ FOR $n = 5$ AND $n = 17$

	65 nm ASIC						Virtex 5 FPGA					
	D (ns)		A (Kμm^2)		E/C (pJ)		D (ns)		A (slices)		E/C (nJ)	
n	5	17	5	17	5	17	5	17	5	17	5	17
prop.	0.63	0.90	2.7	17.9	8.9	71.8	10.9	23.6	107	650	1.5	8.6
[11]	0.91	1.21	3.1	11.0	10.8	57.7	13.9	29.7	98	482	1.6	8.1

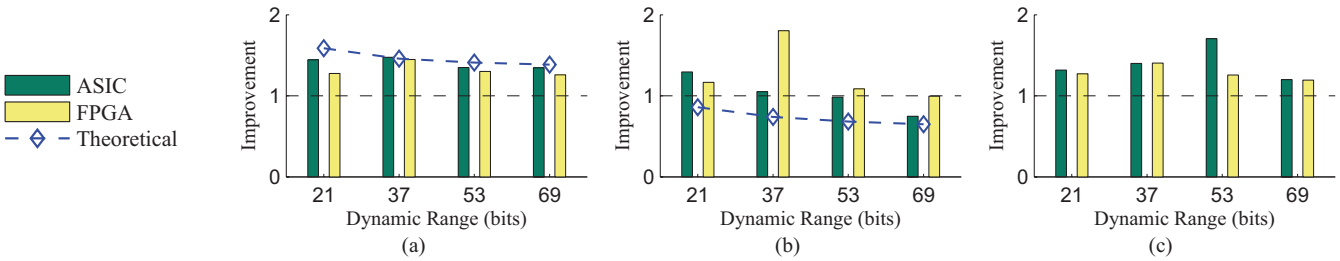

Fig. 2. Comparison with the state of the art. (a) Delay. (b) Circuit area. (c) Energy per conversion. An improvement metric greater than 1 means that the proposed converter is more efficient than the related state-of-the-art converters.

analysis in Fig. 2 is roughly equivalent to the area and power consumption ratios between the converter herein proposed and the one in [11] for the same conversion delay

$$\text{Improvement} = \frac{\text{Delay/Area/Energy}_{[11]}}{\text{Delay/Area/Energy}_{\text{proposed}}}.$$

The presented results suggest that the improvements for the two considered technologies follow the same trend. Improvements in the delay above 25% can be identified for all values of n. On the other hand, the required area for the same delay comparatively increases with n for the proposed converter, thus a slight penalty is observed for large n, as theoretically expected. Concerning the energy per conversion, the proposed converter presents improvements of more than 19% for all values of n.

V. CONCLUSION

In this brief, we proposed a new method to design hardware reverse converters for the moduli set $\{2^n+1, 2^n-1, 2^n, 2^{n+1}+1\}$. Based on this method, an efficient architecture was defined and hardware converters were designed for the target moduli set. The hardware for these converters and for the ones in the related state of the art were described in VHDL and synthesized for both FPGA and ASIC technologies. The obtained experimental results suggest that the delay of the conversion can be reduced by up to 48%. Moreover, the obtained results also suggest a competitive figure of merit for the area and energy per conversion, achieving improvements of up to 30% and 70% with respect to the related state of the art for the same delay, respectively. Therefore, the proposed converter not only provides higher throughput, but also has a competitive utilization of the area and power resources, which is very important for embedded systems.

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, Feb. 2011.

[2] M. A. Soderstrand, W. K. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. Piscataway, NJ: IEEE Press, 1986.

[3] S. Antão, J.-C. Bajard, and L. Sousa, "Elliptic curve point multiplication on GPUs," in *Proc. IEEE Int. Conf. Appl.-Specific Syst. Arch. Process.*, Apr. 2010, pp. 192–199.

[4] L. Sousa and R. Chaves, "A universal architecture for designing efficient modulo 2^n+1 multipliers," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 6, pp. 1166–1178, Jun. 2005.

[5] R. Muralidharan and C.-H. Chang, "Radix-8 booth encoded modulo 2^n-1 multipliers with adaptive delay for high dynamic range residue number system," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 5, pp. 982–993, May 2011.

[6] D. Bakalis and H. T. Vergos, "Shifter circuits for $\{2^n+1, 2^n, 2^n-1\}$ RNS," *Electron. Lett.*, vol. 45, no. 1, pp. 27–29, 2009.

[7] I. Kouretas and V. Paliouras, "A low-complexity high-radix RNS multiplier," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 11, pp. 2449–2462, Nov. 2009.

[8] L.-S. Didier and P.-Y. Rivaille, "A generalization of a fast RNS conversion for a new 4-modulus base," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 56, no. 1, pp. 46–50, Jan. 2009.

[9] A. P. Vinod and A. B. Premkumar, "A memoryless reverse converter for the 4-moduli superset $\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$," *J. Circuits, Syst., Comput.*, vol. 10, nos. 1–2, pp. 85–99, 2000.

[10] M. Bhardwaj, T. Srikanthan, and C. T. Clarke, "A reverse converter for the 4-moduli superset $\{2^n-1, 2^n, 2^n+1, 2^{n+1}+1\}$," in *Proc. IEEE Symp. Comput. Arith.*, Apr. 1999, pp. 168–175.

[11] P. V. A. Mohan and A. B. Premkumar, "RNS-to-binary converters for two four-moduli $\{2^n-1, 2^n, 2^n+1, 2^{n+1}-1\}$ 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, Jun. 2007.

[12] B. Cao, C.-H. Chang, and T. Srikanthan, "A residue-to-binary converter for a new five-moduli set," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 5, pp. 1041–1049, May 2007.

[13] A. Skavantzos, M. Abdallah, and T. Stouraitis, "Large dynamic range RNS systems and their residue to binary converters," *J. Circuits, Syst., Comput.*, vol. 16, no. 2, pp. 267–286, 2007.

[14] R. Conway and J. Nelson, "New CRT-based RNS converter using restricted moduli set," *IEEE Trans. Comput.*, vol. 52, no. 5, pp. 572–578, May 2003.

[15] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*, vol. 2, 2nd ed. New York: Oxford Univ. Press, 2000.

[16] N. B. Chakraborti, J. S. Soundararajan, and A. L. N. Reddy, "An implementation of mixed-radix conversion for residue number applications," *IEEE Trans. Comput.*, vol. 35, no. 8, pp. 762–764, Aug. 1986.

[17] S. S. Erdem and Ç. K. Koç, "A less recursive variant of Karatsuba–Ofman algorithm for multiplying operands of size a power of two," in *Proc. IEEE Symp. Comput. Arith.*, Jun. 2003, pp. 28–35.

[18] P. V. A. Mohan, *Residue Number Systems: Algorithms and Architectures*. Norwell, MA: Kluwer, 2002.

[19] L. Kalampoukas, D. Nikolos, C. Efstathiou, and J. Kalamatianos, "High-speed parallel-prefix modulo 2^n-1 adders," *IEEE Trans. Comput.*, vol. 49, no. 7, pp. 673–680, Jul. 2000.

[20] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron. Comput.*, vol. 13, no. 1, pp. 14–17, Feb. 1964.

[21] R. Zimmermann, "VHDL library of arithmetic units," in *Proc. Int. Forum Design Lang.*, Sep. 1998, pp. 1–6.