Programmable Power-of-two RNS Scaler and its Application to a QRNS Polyphase Filter

G.C. Cardarilli, A. Del Re, A. Nannarelli[†] and M. Re
Dept. of Electronic Engineering University of Rome "Tor Vergata", Italy
[†] Dept. of Informatics and Mathematical Modelling, Technical University, Denmark

Abstract— **The scaling operation, i.e. the division by a constant factor followed by rounding, is a commonly used technique for reducing the dynamic range in Digital Signal Processing (DSP) systems. Usually, the constant is a power of two, and the implementation of the scaling is reduced to a right shift. This basic operation is not easily implementable in the Residue Number System (RNS) due to its non positional nature. A number of different algorithms have been presented in the literature for the RNS scaling. In this paper, several RNS dynamic reduction techniques have been analyzed and the selected one is applied to a polyphase filter bank. A comparison of the filter bank scaled with RNS to binary and binary to RNS conversions, and the RNS scaled implementation is presented. A reduction of area and power consumption of about 30% for the scaling block is obtained.**

I. Introduction

The use of alternative number systems in the implementation of application specific Digital Signal Processing (DSP) systems has gained a remarkable importance in recent years because of the lower power consumption over their two's complement counterparts [1]. In particular, a number of papers have been presented on specific applications using Logarithmic Number System (LNS) or Residue Number Systems (RNS). The renewed success of these techniques is mainly related to the low power requirements, but also to the availability of hardware platforms, such as FPGAs, particularly suitable for the implementation of LNS and RNS arithmetic blocks. Techniques to implement RNS operations are often based on look-up tables, which are the basic blocks to instantiate combinational logic in FPGAs.

On the other hand, several basic operations such as sign detection and truncation, which are trivial in two's complement, are not easy to implement in RNS. In DSP, the most common operation affected by this problem is the scaling operation (i.e. the division by a constant factor followed by rounding) used for reducing the dynamic range in DSP units. In large systems, dynamic range reduction might be needed in different parts of the datapath requiring several scaling blocks. For these reasons, the investigation of optimized RNS scaling techniques is an important aspect of the RNS implementation of DSP systems.

In this paper, starting from the analysis of different RNS scaling algorithms proposed in the literature, an optimized method for power of two scaling is proposed and the results of the comparison with other techniques are given. Moreover, the application of the proposed scaling algorithm to a polyphase filter bank is illustrated. Results shows a 30% reduction of area and power consumption in the scaling block.

II. Background on RNS

A Residue Number System (RNS) is defined by a set of relatively prime integers $\{m_1, m_2, \ldots, m_P\}$. Its dynamic range is given by the product $M = m_1 \cdot m_2 \cdot \ldots \cdot m_P$.
Any integer $X \in \{0, 1, 2, \ldots M-1\}$ has a unique RNS representation given by:

$$X \overset{RNS}{\rightarrow} (\ \langle X \rangle_{m_1}, \langle X \rangle_{m_2}, \ldots, \langle X \rangle_{m_P}\ )$$

where $\langle X \rangle_{m_i}$ denotes the operation $X\ mod\ m_i$ [2].
Operations on different m_i (moduli) are done in parallel

$$Z = X\ op\ Y \quad \overset{RNS}{\rightarrow} \quad \begin{cases} Z_{m_1} = \langle X_{m_1}\ op\ Y_{m_1} \rangle_{m_1} \\ Z_{m_2} = \langle X_{m_2}\ op\ Y_{m_2} \rangle_{m_2} \\ \ldots \quad \ldots \quad \ldots \\ Z_{m_P} = \langle X_{m_P}\ op\ Y_{m_P} \rangle_{m_P} \end{cases}$$

As a consequence, operations on large wordlengths can be split into several modular operations executed in parallel and with reduced wordlength.

The conversion of the RNS representation of Z can be accomplished by the Chinese Remainder Theorem (CRT):

$$Z = \left\langle \sum_{i=1}^{P} \overline{m_i} \cdot \langle \overline{m_i}^{-1} \rangle_{m_i} \cdot Z_{m_i} \right\rangle_M \tag{1}$$

with $\overline{m_i} = \frac{M}{m_i}$ and $\overline{m_i}^{-1}$ obtained by $\langle \overline{m_i} \cdot \overline{m_i}^{-1} \rangle_{m_i} = 1$.

III. RNS Scaling Techniques

RNS scaling techniques presented in the literature can be classified in three main groups:

1) Scaling by several moduli of the RNS base, [3], [4], [5], [6], [7], [8].
2) Scaling by an integer belonging to the RNS dynamic range [9].
3) Scaling by a power of two [10].

In the first group, different algorithms are used to reduce the dynamic by a scaling factor obtained by the product of some moduli composing the RNS base. In the second group, a factorization of the CRT (1) is exploited to scale by an integer number S in the dynamic range of the RNS representation. Differently from the above recalled algorithms, the last technique does not require a binary or a Mixed Radix System (MRS) conversion, since the scaling operation is completed in the RNS domain [10]. Moreover, scaling by a power of two is the classical way for dynamic reduction in binary systems and it is possible to obtain a programmable scaler with a slightly increasing in the hardware complexity. For these reasons, we chose to use this scaling technique in our system. It is based on the Division Remainder Zero Theorem: given $[x_1, x_2, ..., x_n]$ the RNS representation of X, and s, a divider of X, co-prime with the moduli m_i, then:

$$\begin{aligned} \langle X/s \rangle_M &= \langle X \cdot s^{-1} \rangle_M = \\ &= \langle x_1 \cdot s^{-1} \rangle_{m_1}, \langle x_2 \cdot s^{-1} \rangle_{m_2}, ..., \langle x_n \cdot s^{-1} \rangle_{m_n} \end{aligned} \tag{2}$$

Applying equation (2), the scaling by 2 operation is straightforward when X is an even number. If X is odd, we apply the above procedure to $X + 1$, obtaining:

$$\begin{aligned} &\langle (X+1)/2 \rangle_M = \langle (X+1) \cdot 2^{-1} \rangle_M = \\ &= \langle 2^{-1}(x_1+1) \rangle_{m_1}, \langle 2^{-1}(x_2+1) \rangle_{m_2}, ..., \langle 2^{-1}(x_n+1) \rangle_{m_n} \end{aligned} \tag{3}$$

Therefore, the scaling by 2 operation requires a block for the parity detection of X. For this purpose, a base extension to the modulus $m_{n+1} = 2$ is needed. A second modification to equation (2) must be considered when signed numbers have to be represented. In RNS, positive numbers are mapped in the range $[0, (M-1)/2]$, while negative numbers are in the range $[(M+1)/2, M-1]$, in a manner similar to the two's complement representation. Given a signed integer $\widetilde{X}$, whose absolute value is X, we have:

$$\begin{aligned} \widetilde{X} &= X & \text{if} \quad \widetilde{X} \geq 0 \\ \widetilde{X} &= M - X & \text{if} \quad \widetilde{X} < 0 \end{aligned} \tag{4}$$

The scaling operation result must be:

$$\begin{aligned} \widetilde{X'} &= X/s & \text{if} \quad \widetilde{X} \geq 0 \\ \widetilde{X'} &= M - \frac{X}{s} & \text{if} \quad \widetilde{X} < 0 \end{aligned} \tag{5}$$

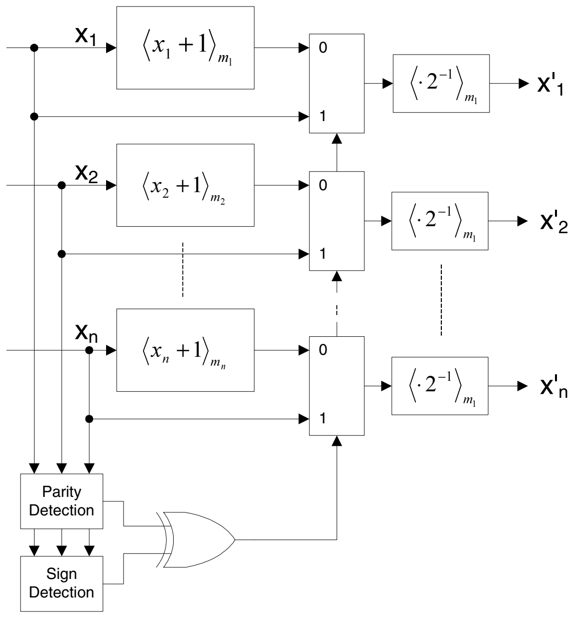
As a consequence, there is the need for a sign detection block. Given a negative integer $\widetilde{X}$ whose absolute value is X, its RNS representation is $M - X$. From equation (5), the scaled by 2 value $\widetilde{X'}$ must be

$$\widetilde{X'} = M - \frac{X}{2} = M - \frac{M - \widetilde{X}}{2} = \frac{M + \widetilde{X}}{2} \tag{6}$$

From equation (6), when $\widetilde{X} < 0$ scaling must be applied to the quantity $Y = \widetilde{X} + M$, which belongs to the enlarged dynamic range obtained by the base extension. Since $M = \prod_i m_i$ is odd, we will have:

$$\begin{aligned} <Y>_2 &= 0 & \text{if} \quad <X>_2 = 1 \\ <Y>_2 &= 1 & \text{if} \quad <X>_2 = 0 \end{aligned} \tag{7}$$

while the remaining digits $[y_1, y_2, ..., y_n]$ is equal to the representation $[\widetilde{x_1}, \widetilde{x_2}, ..., \widetilde{x_n}]$ of $\widetilde{X}$. Therefore, when $\widetilde{X}$ is positive, the output of the sign detection block is 0 and the output of the parity block is correct, while, if $\widetilde{X}$ is negative, the output of the sign detection block is 1 and the output of the parity block must be negated. Therefore, a XOR gate is used to obtain the correct parity check both for positive and negative input, avoiding the addition of M. The block diagram of the basic scaling-by-2 operation is sketched in Fig. 1. It is composed of the blocks previously indicated: the sign and parity detectors, the modular adders and multipliers and the 2-to-1 multiplexers.



X1
$\langle x_1+1 \rangle_{m_1}$
$\langle \cdot 2^{-1} \rangle_{m_1}$
X'1
X2
$\langle x_2+1 \rangle_{m_2}$
$\langle \cdot 2^{-1} \rangle_{m_1}$
X'2
Xn
$\langle x_n+1 \rangle_{m_n}$
$\langle \cdot 2^{-1} \rangle_{m_1}$
X'n
0
1
Parity Detection
Sign Detection

Fig. 1. Block diagram for the basic scaling by 2 operation

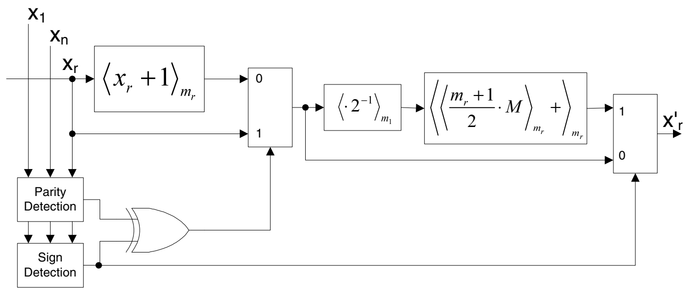
In the following, we describe the algorithms and the design choices made in the implementation of the parity and sign detection blocks of Fig. 1.

A. Parity Check Algorithm

Several algorithms have been analyzed for the parity check of the input signed integer $\widetilde{X}$, requiring the base extension to the modulo $m_{n+1} = 2$. We analyzed the technique proposed in [2] in the original architecture and a modified structure obtained by inverting the order of moduli in the base extension block. Also, we analyzed the Barsi-Pinotti method of [6], and the fractional CRT [11]:

$$\begin{aligned} \langle X_s \rangle_2 &= \left\langle \sum_{i=1}^{N} \frac{2}{m_i} \left\langle \widehat{m}_i^{-1} \cdot x_i \right\rangle_{m_i} - 2\alpha \right\rangle_2 \\ &= \left\langle \sum_{i=1}^{N} \frac{2}{m_i} \left\langle \widehat{m}_i^{-1} \cdot x_i \right\rangle_{m_i} \right\rangle_2 \end{aligned} \tag{8}$$

The Shenoy-Kumaresan method [12] requires the use of a redundant modulus in order to complete the base extension. If a redundant modulus is not available, the complexity of the related base extension must be taken into account. Moreover, from equation (6), it can be noted that the scaled by 2 value of a negative integer is a function of the dynamic range M of the RNS representation. Since the parity check block requires an additional modulus m_r, the new dynamic range is $M' = M \cdot m_r$. This does not affect the scaling of residues related to the base moduli (m_r is not part of the RNS base by definition), but attention must be paid to the scaling of the



x_1
x_n
x_r
$\langle x_r + 1 \rangle_{m_r}$
0
1
$\langle \cdot 2^{-1} \rangle_{m_1}$
$\left\langle \left\langle \frac{m_r+1}{2} \cdot M \right\rangle_{m_r} + \right\rangle_{m_r}$
1
0
x'_r
Parity Detection
Sign Detection

Fig. 2. Scaling block for the redundant modulus

Method	Area	Comparison	Latency
Szabo-Tanaka incr. (ST-in) [2]	2786	+11.2%	2
Szabo-Tanaka decr. (ST-dec) [2]	2493	+0%	2
Barsi-Pinotti (BP)[6]	3460	+38.7%	4
Fractional CRT (CRT)[11]	2306	-9%	2
Shenoy-Kumaresan (SK)[12]	395	-84.1%	1

TABLE I
COMPARISON OF THE IMPLEMENTATION RESULTS

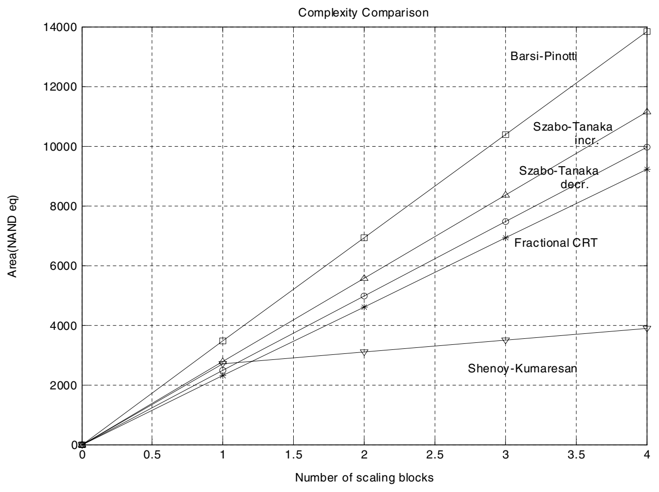


Fig. 3. Silicon area vs number of scaling blocks

residue x_r. In particular, if $\widetilde{X}$ is negative, we obtain for Y:

$$Y = M' + \widetilde{X} = M \cdot m_r + \widetilde{X} = M(m_r + 1) - X \qquad (9)$$

and for the scaled value X':

$$X' = \frac{Y}{2} = \frac{M' + \widetilde{X}}{2} = \frac{M \cdot m_r + \widetilde{X}}{2} = \frac{M(m_r + 1) - X}{2} \qquad (10)$$

The implementation of equation (10) requires modifications to the scaling-by-2 block which are illustrated in Fig. 2. We implemented each scheme using the same RNS base (described in the next section) and compared latency and area obtained after synthesis on standard cells[1]. The results are summarized in Table I, where area is reported as number of NAND2 equivalent gates and the latency refers to a clock period of 5 ns.

From Table I, the Shenoy-Kumaresan (SK) method seems to be very convenient with respect to the area, but it requires the implementation of a fractional CRT unit as well (for the redundant modulus). Taking into account the above considerations, a more precise comparison is given in Fig. 3, where the silicon area is drawn as a function of the number of cascaded scaling blocks, showing that, if a scaling by 2^n with $n \geq 2$ is required, the SK method is convenient.

In conclusion, for the implementation of the parity check block, we chose the Shenoy-Kumaresan algorithm, using the fractional CRT method for the related base extension to the redundant modulus ($m_r = 5$).

B. Sign Detection Algorithm

The sign detection can be reduced to a parity detection multiplying by 2 the integer X. If X is positive

$$0 \leq X \leq (M-1)/2 \quad \Rightarrow 0 \leq 2X \leq M - 1$$

[1]We used Synopsys tools and the AMS 0.35 μm library of standard cells.

the integer $2X$ is still in the dynamic range defined by the RNS base, giving $\langle 2 \cdot X \rangle_2 = 0$. If $X < 0$

$$(M+1)/2 \leq X \leq M - 1 \quad \Rightarrow \quad M + 1 \leq 2X \leq 2M - 2$$

the integer $2X$ is out of the dynamic range, giving $\langle 2 \cdot X \rangle_2 = 1$. In that way, it is possible to determine the sign of X applying one of the above algorithm for the parity detection to $2X$. From Fig. 3, the method proposed in [11] results as the more convenient algorithm for a single scaling by 2 operation. Adding the hardware required for the multiplication by 2, we obtain a total complexity of 2485 NAND eq.

A slightly different algorithm was proposed in [13]. The sign of X is given by:

$$sign(X) = \begin{cases} 0 & \text{if} \quad 0 \leq X \leq \frac{M-1}{2} \\ 1 & \text{if} \quad \frac{M+1}{2} \leq X < M \end{cases} \qquad (11)$$

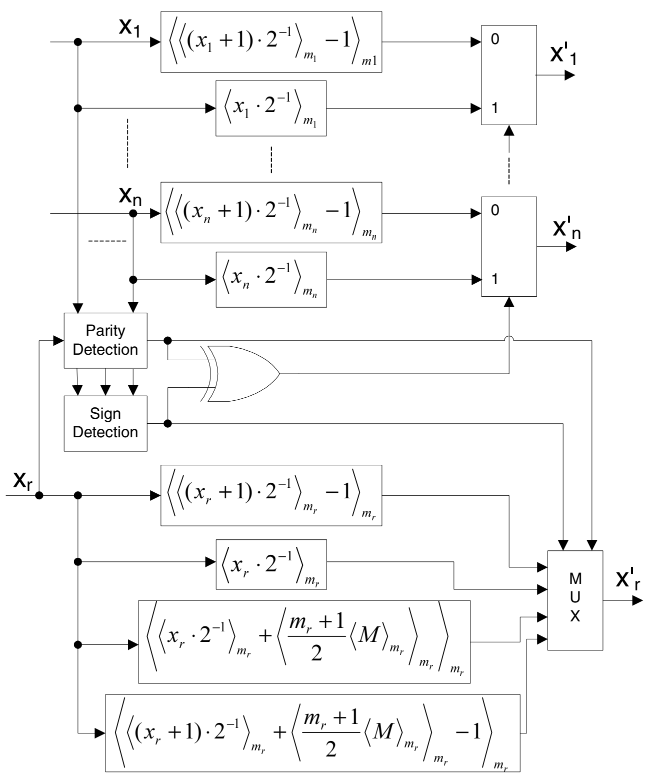
Dividing X by $M/2$ we obtain

$$sign(X_s) = \begin{cases} 0 & \text{if} \quad 0 \leq X_s < 1 \\ 1 & \text{if} \quad 1 \leq X_s < 2 \end{cases} \qquad (12)$$

So the sign of X_s can be determined by its parity check by applying the fractional CRT (8). Synthesis results of the VHDL description of the last method gives a complexity of 2160 NAND eq, resulting more convenient with respect to the previous algorithm.

C. Error Correction Block

When an odd number has to be scaled by a power of two an error is unavoidable. This error can occur in every block of the scaler, but the major contribution is provided by the last block. In fact, the possible error generated in the first blocks is scaled by the following one, differently from what happens for the last. In order to reduce the error to be less than 0.5, it is sufficient to introduce the correction in the last scaling block. The related block diagram is sketched in Fig. 4.



x_1
$\langle\langle(x_1+1)\cdot 2^{-1}\rangle_{m_1}-1\rangle_{m1}$
$\langle x_1\cdot 2^{-1}\rangle_{m_1}$
0
1
x'_1
x_n
$\langle\langle(x_n+1)\cdot 2^{-1}\rangle_{m_n}-1\rangle_{m_n}$
$\langle x_n\cdot 2^{-1}\rangle_{m_n}$
0
1
x'_n
Parity Detection
Sign Detection
x_r
$\langle\langle(x_r+1)\cdot 2^{-1}\rangle_{m_r}-1\rangle_{m_r}$
$\langle x_r\cdot 2^{-1}\rangle_{m_r}$
$\langle\langle x_r\cdot 2^{-1}\rangle_{m_r}+\langle\frac{m_r+1}{2}\langle M\rangle_{m_r}\rangle_{m_r}\rangle_{m_r}$
$\langle\langle(x_r+1)\cdot 2^{-1}\rangle_{m_r}+\langle\frac{m_r+1}{2}\langle M\rangle_{m_r}\rangle_{m_r}-1\rangle_{m_r}$
MUX
x'_r

Fig. 4. The last scaling block with error correction

IV. Comparisons

We implemented in standard cells a programmable power of two scaler to reduce the dynamic range in a QRNS polyphase filter bank. The QRNS polyphase filter bank is the same described in [14], based on the following set of moduli: $m_i = \{13, 17, 29, 37, 41, 53\}$. In [14], the scaling was performed by converting (by CRT) the QRNS representation to binary, by truncating the resulting two's complement value and by re-converting in QRNS. The scaling by a power of two operation is implemented by cascading as many scaling-by-2 blocks as needed. In our case, we use seven scaling blocks to obtain the division by 2^7 needed in the filter. A multiplexer is used between a scaler-by-2 block and a second one for input routing, resulting in a slightly increased hardware complexity. Table II shows the comparison between the programmable power of two scaler and the scaling (performed by QRNS/2's compl/QRNS) of [14], both implemented in the QRNS filter bank described above. Results show a reduction of about 28% in area and 32% in power dissipation obtained for the new scaling scheme with respect to the scaling of [14]. For the whole filter, both the scaled versions reduce area and power of about 37%, confirming that the specific scaling algorithm has a small impact on large systems. Power figures were obtained by Synopsys based on actual switching information.

V. Conclusion

In this paper, a programmable RNS scaler by a power of two has been presented. Several RNS dynamic reduction techniques have been analyzed and the selected one is applied to a QRNS polyphase filter bank. A reduction of area and power consumption of about 30% for the scaling block has been obtained, both for area and power consuption. When the entire QRNS filter is considered, both the scaled versions reduce area and power of about 37%, proving that the optimization of scaling blocks and converters is not a primary concern in large systems. However, this suggests that using truncation in several parts of the datapath can be beneficial for those systems.

Scaling Method	Area	Ratio	Power	Ratio	Lat.
CRT [14]	19432	1.00	22.36 mW	1.00	6
New scaler	13903	0.72	15.09 mW	0.67	n+2

Complete Filter	Area	Ratio	Power	Ratio	Lat.
Error Free [14]	1670K	1.00	1510 mW	1.00	11
CRT [14]	1050K	0.63	950 mW	0.63	17
New scaler	1041K	0.62	943 mW	0.62	21

TABLE II
Comparison of scaler and filter implementations

Acknowledgment

Thanks to Cristina Fidati for her contribution to this work.

References

[1] T. Stouraitis and V. Paliouras, "Considering the alternatives in low-power design," *IEEE Circuits Devices Mag.*, vol. 17, pp. 22–29, July 2001.
[2] R. I. T. N. S. Szabo, *Residue Arithmetic and its Application to Technology*. New York, NY (USA): McGraw-Hill, 1967.
[3] Z. D. Ulman and M. Czyzak, "Highly parallel, fast scaling of numbers in non-redundant residue arithmetic," *IEEE Trans. Signal Processing*, vol. 46, pp. 487–496, Feb. 1998.
[4] A. Garcia and A. Lloris, "A look-up scheme for scaling in the RNS," *IEEE Trans. Comput.*, vol. 48, pp. 748–751, July 1999.
[5] M. A. P. Shenoy and R. Kumaresan, "A fast and accurate RNS scaling technique for high speed signal processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 929–937, June 1989.
[6] F. Barsi and M. C. Pinotti, "Fast base extension and precise scaling in RNS for look-up table implementations," *IEEE Trans. Signal Processing*, vol. 43, pp. 2427–2430, Oct. 1995.
[7] G. C. Cardarilli, M. Re, R. Lojacono, and G. Ferri, "A systolic architecture for high performance scaled residue to binary conversion," *IEEE Trans. Circuits Syst. I*, vol. 47, pp. 1523–1526, Oct. 2000.
[8] N. Burgess, "Scaled and unscaled residue number system to binary conversion techniques using the core function," in *Proc. of 13th Symposium on Computer Arithmetic*, June 1997, pp. 250–257.
[9] M. Griffin, M. Sousa, and F. Taylor, "Efficient Scaling in the Residue Number System," in *Proc. of Intl. Conf. on Acoustics, Speech, and Signal Processing*, May 1989, pp. 1075–1078.
[10] U. Mayer-Base and T. Stouraitis, "New power-of-two scaling scheme for cell-based IC design," *IEEE Trans. VLSI Syst.*, vol. 11, pp. 446–450, Apr. 2003.
[11] M. Lu and J. S. Chang, "A novel division algorithm for the residue number system," *IEEE Trans. Comput.*, vol. 41, pp. 1026–1032, Aug. 1992.
[12] M. A. P. Shenoy and R. Kumaresan, "Fast base extension usign a redundant modulus in RNS," *IEEE Trans. Comput.*, vol. 38, pp. 292–297, Feb. 1989.
[13] T. V. Vu, "Efficient implementations of the chinese remainder theorem for sign detection and residue decoding," *IEEE Trans. Comput.*, vol. 34, pp. 667–669, July 1985.
[14] G. C. Cardarilli, A. Del Re, A. Nannarelli, and M. Re, "Low-power Implementation of Polyphase Filters in Quadratic Residue Number System," in *Proc. of IEEE Intl. Symposium on Circuits and Systems*, vol. 2, May 2004, pp. 725–728.