On the Comparison of Different Number Systems in the Implementation of Complex FIR Filters

Gian Carlo Cardarilli, Alberto Nannarelli, Marco Re

On the Comparison of Different Number Systems in the Implementation of Complex FIR Filters

Gian Carlo Cardarilli[1], Alberto Nannarelli[2], and Marco Re[1]

[1] Department of Electronics, University of Rome Tor Vergata, Rome, Italy
{g.cardarilli, marco.re}@uniroma2.it
[2] DTU Informatics, Technical University of Denmark, Kongens Lyngby, Denmark
an@imm.dtu.dk

1 Introduction

In modern electronic systems, complex arithmetic computation plays an important role in the implementation of different Digital Signal Processing (DSP) and scientific computation algorithms [1], [2]. Most of the interest in complex signal processing is related to the implementation of wireless communication systems based on new concepts and architectures [3]. A very interesting tutorial paper on complex signal processing and its applications has been presented recently in [4]. In [4], the importance of the use of complex signal processing in wireless communications systems has been shown. Regarding communication systems, one of the most critical computation to be implemented in hardware is complex FIR filtering. In fact, FIR filters are generally characterized by a high order (number of taps) to obtain sharp transition bands that, in case of high speed real time computation, require a many resources and have high power dissipation. In particular, for complex FIR filters, the hardware complexity is mostly determined by the number of complex multipliers (i.e. each complex multiplication is actually implemented with four scalar multiplications). Different solutions have been proposed to lower the hardware complexity of the complex multiplication either at algorithmic level (Golub Rule) [5], or by using different number systems such as the Quadratic Residue Number System (QRNS) [6], [2] and the Quater-Imaginary Number System (QINS) [7].

The aim of this work is to compare in terms of performance, area and power dissipation, the implementations of complex FIR filters based on the traditional Two's Complement System (TCS), the QRNS and the QINS (or radix-2j) implemented in the Redundant Complex Number Systems (RCNS) [8].

Previous work was done on both the QRNS ([6], [9]) and on the radix-2j and the RCNS ([10], [11], [12]). In this work, we compare for a specific application, the complex FIR filter, the performance and the tradeoffs of TCS, QRNS and RCNS. The results of the implementations show that the complex filter implemented in QRNS has the lowest power dissipation and the smallest area with respect to filters implemented in TCS and RCNS.

The work is organized as follows: in Section II a background on the QRNS and the radix-2j number systems is given; the FIR filter architectures for the three number systems are described in Section III; the synthesis results and the comparisons are discussed in Section IV. Finally, the conclusions are drawn in Section V.

2 The Quadratic Residue Number System

A Residue Number System (RNS) is defined by a set of P relatively prime integers $\{m_1, m_2, \ldots, m_P\}$ which identify the RNS base. 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 $X \ mod\ m_i$ [13]. Operations on different m_i (moduli) are done in parallel

$$Z = X\ op\ Y \quad \overset{RNS}{\rightarrow} \quad \begin{cases} Z_{m_1} = \langle X_{m_1}\ op\ Y_{m_1} \rangle_{m_1} \\ Z_{m_2} = \langle X_{m_2}\ op\ Y_{m_2} \rangle_{m_2} \\ \ldots \quad \ldots \quad \ldots \\ Z_{m_P} = \langle X_{m_P}\ op\ Y_{m_P} \rangle_{m_P} \end{cases} \tag{1}$$

As a consequence, operations on large wordlengths can be split into several modular operations executed in parallel and with reduced wordlength [13].

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

$$Z = \left\langle \sum_{i=0}^{P} \overline{m_i} \cdot \langle \overline{m_i}^{-1} \rangle_{m_i} \cdot Z_{m_i} \right\rangle_M \qquad \text{with } \overline{m_i} = \frac{M}{m_i} \tag{2}$$

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

To better explain the CRT, we show an example in which we convert the RNS representation $\{3,\ 6,\ 5\}$, with RNS base $\{\ 5,\ 7,\ 8\ \}$, to integer. The dynamic range of the RNS base $\{\ 5,\ 7,\ 8\ \}$ is $M = 280$. We start by computing the values $\overline{m_i} = \frac{M}{m_i}$

$$\overline{m_1} = \frac{280}{5} = 56 \qquad \overline{m_2} = \frac{280}{7} = 40 \qquad \overline{m_3} = \frac{280}{8} = 35$$

To compute $\overline{m_i}^{-1}$, we have to find a number x such that

$$\langle \overline{m_i} \cdot x \rangle_{m_i} = 1 \tag{3}$$

For this reason, x is called the multiplicative inverse of $\overline{m_i}$ and indicated as $\overline{m_i}^{-1}$. By computer iterations, we find

$$\overline{m_1}^{-1} = 1 \qquad \overline{m_2}^{-1} = 3 \qquad \overline{m_3}^{-1} = 3$$

Finally, applying (2) to the set of residues $\{3, 6, 5\}$ we get

$$\left\langle \sum_{i=1}^{3} \overline{m_i} \cdot \langle \overline{m_i}^{-1} \rangle_{m_i} \cdot Z_i \right\rangle_{280} = \langle\ 56 \cdot 1 \cdot 3 + 40 \cdot 3 \cdot 6 + 35 \cdot 3 \cdot 5\ \rangle_{280} = \langle\ 1413\ \rangle_{280} = 13$$

We can easily verify that

$$\langle 13 \rangle_5 = 3, \quad \langle 13 \rangle_7 = 6, \quad \langle 13 \rangle_8 = 5$$

In the complex case, we can transform the imaginary term into an integer if the equation $q^2 + 1 = 0$ has two distinct roots q_1 and q_2 in the ring of integers modulo M (Z_M). A complex number $x_R + jx_I = (x_R, x_I) \in Z_M \times Z_M$, with q root of $q^2 + 1 = 0$ in Z_M, has a unique Quadratic Residue Number System representation given by

$$\begin{aligned} (x_R, x_I) \overset{QRNS}{\rightarrow} (X_i, \hat{X}_i) \quad & i = 1, 2, \ldots, P \\ X_i &= \langle x_R + q \cdot x_I \rangle_{m_i} \\ \hat{X}_i &= \langle x_R - q \cdot x_I \rangle_{m_i} \end{aligned} \tag{4}$$

The inverse QRNS transformation is given by

$$\begin{aligned} (X_i, \hat{X}_i) \overset{RNS}{\rightarrow} (X_{Ri}, X_{Ii}) \quad & i = 1, 2, \ldots, P \\ X_{Ri} &= \langle 2^{-1}(X_i + \hat{X}_i) \rangle_{m_i} \\ X_{Ii} &= \langle 2^{-1} \cdot q^{-1}(X_i - \hat{X}_i) \rangle_{m_i} \end{aligned} \tag{5}$$

where 2^{-1} and q^{-1} are the multiplicative inverses of 2 and q, respectively, modulo m_i:

$$\langle 2 \cdot 2^{-1} \rangle_{m_i} = 1 \qquad \text{and} \qquad \langle q \cdot q^{-1} \rangle_{m_i} = 1\ .$$

Then, by applying the CRT we get

$$\begin{aligned} (X_{R1}, X_{R2}, \ldots, X_{RP}) &\overset{CRT}{\rightarrow} x_R \\ (X_{I1}, X_{I2}, \ldots, X_{IP}) &\overset{CRT}{\rightarrow} x_I \end{aligned} \tag{6}$$

Moreover, it can be proved that for all the prime integers which satisfy

$$p = 4k + 1 \qquad\qquad k \in N$$

the equation $q^2 + 1 = 0$ has two distinct roots q_1 and q_2.

As a consequence, the product of two complex numbers $x_R + jx_I$ and $y_R + jy_I$ is in QRNS

$$(x_R + jx_I)(y_R + jy_I) \overset{QRNS}{\rightarrow} (\langle X_i Y_i \rangle_{m_i}, \langle \hat{X}_i \hat{Y}_i \rangle_{m_i}) \tag{7}$$

and it is realized by using two integers multiplications instead of four.

We illustrate an example of QRNS multiplication in the ring modulo 13. The complex multiplication to perform is

$$(x_R + jx_I)(y_R + jy_I) = (3 + j)(2 + j2) = 4 + j8$$

For $m = 13$ the root is $q = q_1 = 5 \leftrightarrow \langle 5 \cdot 5 \rangle_{13} = -1$. The conversion to QRNS according to (4) gives

$$\begin{array}{ll} X = \langle 3 + 5 \cdot 1 \rangle_{13} = 8 & Y = \langle 2 + 5 \cdot 2 \rangle_{13} = 12 \\ \hat{X} = \langle 3 - 5 \cdot 1 \rangle_{13} = 11 & \hat{Y} = \langle 2 - 5 \cdot 2 \rangle_{13} = 5 \end{array}$$

The two QRNS multiplications (modulus 13) are:

$$X \cdot Y = \langle 8 \cdot 12 \rangle_{13} = 5 \qquad \hat{X} \cdot \hat{Y} = \langle 11 \cdot 5 \rangle_{13} = 3$$

And finally, the conversion QRNS to integer according to (5) gives

$$\begin{array}{lll} z_R = \langle 7(5 + 3) \rangle_{13} = 4 & \text{being} & 2^{-1} = 7 \\ z_I = \langle 7 \cdot 8(5 - 3) \rangle_{13} = 8 & \text{and} & q^{-1} = 8 \end{array}$$

3 The Radix-2j Number System

It is well known that an integer x can be represented by a digit-vector

$$X = (x_{n-1}, \ldots, x_1, x_0)_r$$

such that

$$x = \sum_{i=0}^{n-1} x_i \cdot r^i$$

where r is the radix of the representation. By choosing $r = 2j$, we obtain a *Quater-Imaginary* Number System (QINS) [7]. Complex numbers can be represented in QINS by vectors with the non-redundant digit set $\{0, 1, 2, 3\}$. Therefore, a complex number $a + jb$ is represented in QINS as:

$$\begin{aligned} a + jb &= x_{n-1}(2j)^{n-1} + x_{n-2}(2j)^{n-2} + \ldots + \\ &\quad + x_3(-8j) + x_2(-4) + x_1(2j) + x_0(1) \\ &= (x_{n-1}, \ldots, x_1, x_0)_{2j} \end{aligned} \tag{8}$$

Real		Imaginary	
-8	00200.0	-8j	01000.0
-7	00201.0	-7j	01010.2
-6	00202.0	-6j	01010.0
-5	00203.0	-5j	01020.2
-4	00100.0	-4j	01020.0
-3	00101.0	-3j	01030.2
-2	00102.0	-2j	01030.0
-1	00103.0	-1j	00000.2
0	00000.0	0j	00000.0
1	00001.0	1j	00010.2
2	00002.0	2j	00010.0
3	00003.0	3j	00020.2
4	10300.0	4j	00020.0
5	10301.0	5j	00030.2
6	10302.0	6j	00030.0
7	10303.0	7j	103000.2
8	10200.0	8j	103000.0

Table 1. Representation of real and imaginary integers in QINS.

The above expression, shows that the real part is represented by the digits of even weight, while the imaginary one by the digits of odd weight. Furthermore, the sign is embedded in the representation. The imaginary number j cannot be represented by (8). To represent j, we need the power -1, which corresponds to $-\frac{1}{2}j$, that in the conventional number systems (e.g. binary) is only needed to represent fractional numbers. Table 1 shows how the real and imaginary numbers, in the range $[-8, 8]$ and $[-8j, 8j]$ respectively, are represented in QINS. Every complex number $x_R + jx_I$ can be obtained by overlapping the real and imaginary parts. For example, according to Table 1, $4 - 5j$ is represented by the digit vector 11320.2.

From Table 1 we can notice that for a given number of digits the representation is not symmetric with respect to the zero. For example, in the two's complement binary system with 8 digits we can represent the dynamic range $\{-128, 127\}$. In the QINS, for the real part, with 3 digits (equivalent to 64 different values) the dynamic range representable is $\{-12, 51\}$.

3.1 Addition

The addition of two QINS numbers can be performed by changing the *carry rule* according to (8). First, because the even weight digits represent the real part and the odd weight the imaginary one, the carry is propagated by skipping a digit. Second, because two adjacent even (or odd) weight digits have opposite sign, the carry propagated acts as a borrow. For example, if a positive weight digit generates a carry, this positive value will decrement the next digit

with negative weight, and vice-versa. In addition, the propagation of borrows can generate negative digits (e.g. -1). Therefore, because of the quaternary representation of the QINS, the negative digits are converted into positive (modulo operation) and an always positive carry propagated. Summarizing the addition algorithm is implemented as:

$$x_i, y_i, s_i \in \{0, 1, 2, 3\}$$
$$c_i \in \{\overline{1}, 0, 1\}$$

$$s_i = (x_i + y_i + c_i) \bmod 4$$

$$c_{i+2} = \begin{cases} \overline{1} \text{ if } (x_i + y_i + c_i) \geq 4 \\ 1 \text{ if } (x_i + y_i + c_i) < 0 \\ 0 \; otherwise \end{cases}$$

For example, if we wnat to add $x_R = 1$ and $y_R = 3$ in QINS we get:

$$\begin{array}{ll} X\text{: } 0\ 0\ 0\ 0\ 3.\ 0 & + \\ Y\text{: } 0\ 0\ 0\ 0\ 1.\ 0 & + \\ c\text{: } \ 1\ 0\ \overline{1}\ 0\ 0.\ 0 & = \\ S\text{: } \ 1\ 0\ 3\ 0\ 0.\ 0 & \rightarrow \ s_R = 4 \end{array}$$

3.2 The Redundant Complex Number Systems

The implementation of the basic arithmetic operators in radix-$2j$ can take advantage of the Signed-Digit (SD) representation [14], which allows carry free addition. The combination of radix-$2j$ and SD representation, resulted in the Redundant Complex Number Systems (RCNS), which is described in [8], [10], [11], [12] and [15].

We now briefly recall the characteristics of the RCNS. The RCNS is a redundant positional number system based on the radix rj where its digits can assume the $2\alpha + 1$ values: $A_\alpha = \{\overline{\alpha}, \cdots, \overline{1}, 0, 1, \cdots, \alpha\}$ where $\overline{\alpha} = -\alpha$. . In the case of the radix $2j$, two possible RCNSs [10] are:

1. RCNS 2j, 2 with digit set $A_2 = \{\overline{2}, \overline{1}, 0, 1, 2\}$
2. RCNS 2j, 3 with digit set $A_3 = \{\overline{3}, \overline{2}, \overline{1}, 0, 1, 2, 3\}$

In this work, RCNS 2j, 2 is used to recode the multiplier, and RCNS 2j, 3 is used for the signed-digit additions, as illustrated next.

4 FIR Filter Architecture

A complex FIR filter of order N is expressed by

$$\underline{y}(n) = \sum_{k=0}^{N-1} \underline{a}_k \underline{x}(n-k) \tag{9}$$

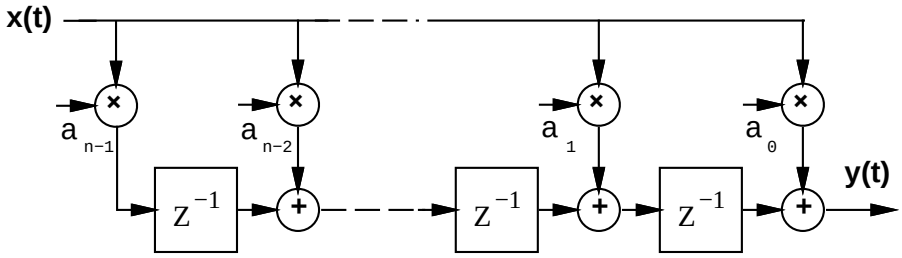

Fig. 1. Structure of FIR filter in transposed form.

where $\underline{x}$, $\underline{y}$ and $\underline{a}_k$ denote complex numbers. We consider the implementation of a FIR filter in transposed form because its structure is more regular with respect to the filter order N and it does not require a tree of adders. The filter in transposed form (Fig. 1) can be regarded as the sequence of groups, often referred as *taps*, composed of:

- a complex multiplier;
- a complex adder implemented with one adder for the real part, and one for the imaginary part;
- a register to store the real and imaginary parts.

We perform our design space exploration for programmable N-tap complex FIR filters with input and coefficients size of 10 bits for both the real part and imaginary parts. The 20 bit dynamic range of the filter guarantees error free operations[3].

4.1 TCS FIR Filter

A single tap of the The programmable N-tap TCS complex FIR filter is realized as sketched in Fig. 2. It is composed of two branches: the real branch (top part of Fig. 2) and the imaginary branch (bottom part of Fig. 2). The real and imaginary products are both realized with two Booth multipliers each, and the resulting partial products are accumulated in a Wallace's tree structure which produces a carry-save (CS) representation of the product at each side of the filter. We chose to keep the product in carry-save (CS) format to speed-up the operation, and delayed the assimilation of the CS representation to the last stage of the filter. In both branches (real and imaginary) of each tap we need to add the CS representation of the product to the value stored in the register (previous tap). Again, to avoid the propagation of the carry, we can store the CS representation. For this reason, we need to implement the addition with an array of 4:2 carry-save adders (CSA), as shown in Fig. 2.

[3] These wordlengths are derived from the specification of an actual digital filter for satellite TV broadcasting.

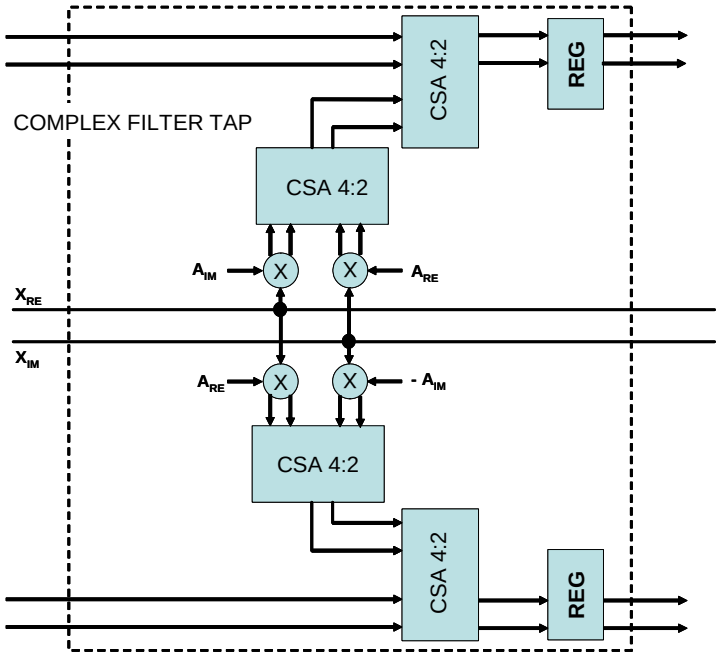

Fig. 2. Structure of tap in TCS complex FIR filter.

We convert the CS representation of y_{Re} and y_{Im} with two carry-propagate adders at the filter output.

4.2 QRNS FIR Filter

The architecture of the QRNS filter, is a direct consequence of (1), (7) and (9), and it can be realized by two RNS filters in parallel as shown in Fig. 3. Each RNS filter is then decomposed into P filters working in parallel, where P is the number of moduli used in the RNS representation. In addition, the RNS filter requires both binary to QRNS and QRNS to binary converters.

In order to have a dynamic range of 20 bits, as required by the specifications, we chose the following set of moduli:

$$m_i = \{5, 13, 17, 29, 41\}$$

such that

$$log_2(5 \cdot 13 \cdot 17 \cdot 29 \cdot 41) > 20 \ .$$

For each path mod m_i, we have to build a FIR filter with a structure similar to that of Fig. 1. Therefore, we need to implement modular multiplication and addition.

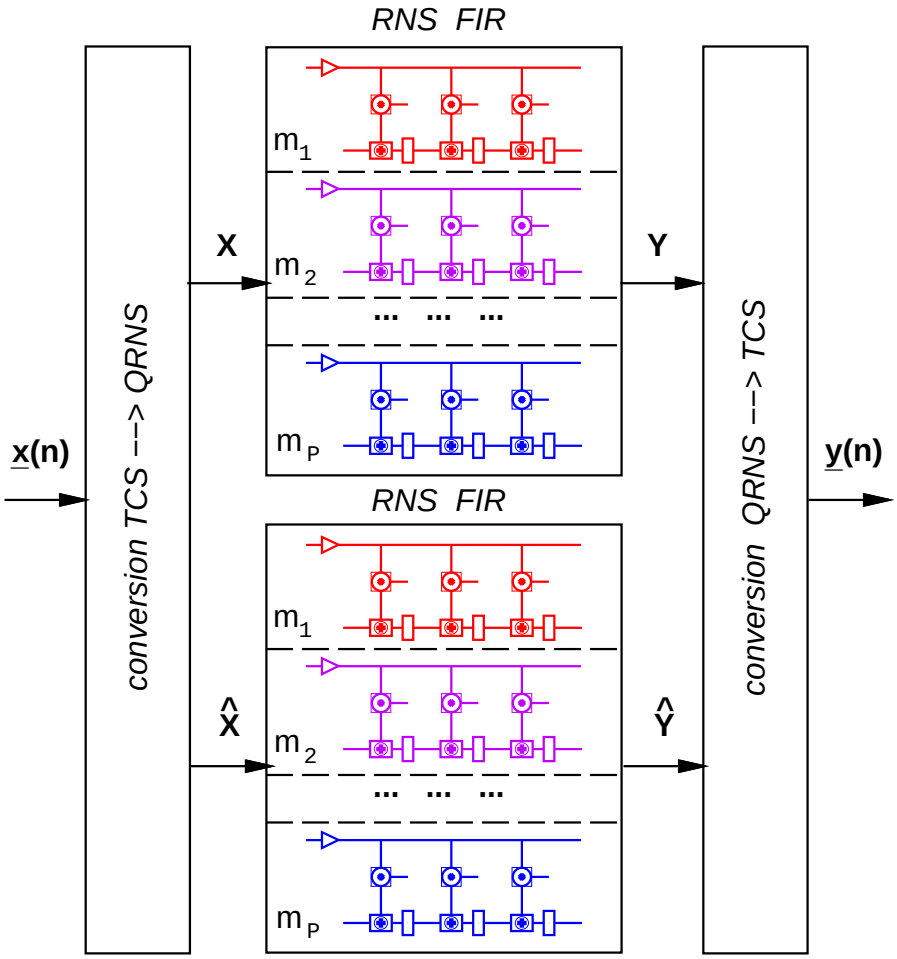

Fig. 3. QRNS FIR Filter architecture

Implementation of modular addition

The modular addition

$$\langle a_1 + a_2 \rangle_m$$

can be implemented by two additions. If the result of $a_1 + a_2$ exceeds the modulo (it is larger than $m-1$), we have to subtract the modulo m. In order to speed-up the operation we can execute in parallel the two operations:

$$(a_1 + a_2) \quad \text{and} \quad (a_1 + a_2 - m).$$

If the sign of the three-term addition is negative, it means than the sum $(a_1 + a_2) < m$ and the modular sum is $a_1 + a_2$, otherwise the modular ad-

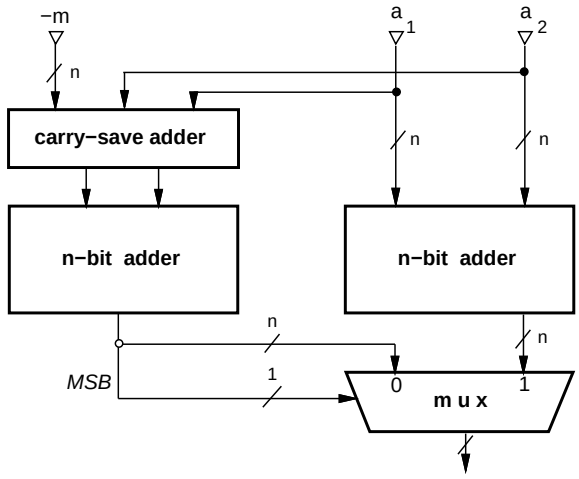

Fig. 4. Architecture of the modular adder.

n	w	$\langle q^w \rangle_m = n$
0	N/A	
1	0	$\langle 2^0 \rangle_5 = 1$
2	1	$\langle 2^1 \rangle_5 = 2$
3	3	$\langle 2^3 \rangle_5 = 3$
4	2	$\langle 2^2 \rangle_5 = 4$

Table 2. Example of isomorphic transformation for $m = 5$ ($q = 2$).

dition is the result of the three-term addition. The above algorithm can be implemented with two binary adders as shown in Fig. 4.

Implementation of modular multiplication by isomorphism

Because of the complexity of modular multiplication, it is convenient to implement the product of residues by the isomorphism technique [16] . By using isomorphisms, the product of the two residues is transformed into the sum of their indices which are obtained by an isomorphic transformation. According to [16], if m is prime there exists a primitive radix q such that its powers modulo m cover the set $[1, m-1]$:

$$n = \langle q^w \rangle_m \qquad \text{with } n \in [1, m-1] \text{ and } w \in [0, m-2].$$

An example of isomorphic transformation is shown in Table 2 for $m = 5$. In this case, the primitive radix is $q = 2$.

Both transformations $n \to w$ and $w \to n$ can be implemented with $m-1$ entries look-up tables, if the moduli are not too large (less than 8-bit wide). Therefore, the product of a_1 and a_2 modulo m can be obtained as:

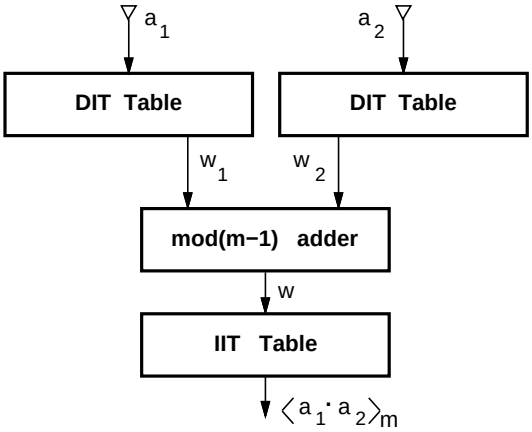

Fig. 5. Structure of isomorphic multiplication.

$$\langle a_1 \cdot a_2 \rangle_m = \langle q^w \rangle_m$$

where

$$w = \langle w_1 + w_2 \rangle_{m-1} \qquad \text{with } a_1 = \langle q^{w_1} \rangle_m \text{ and } a_2 = \langle q^{w_2} \rangle_m$$

In order to implement the modular multiplication the following operations are performed:

1) Two Direct Isomorphic Transformations (DIT) to obtain w_1 and w_2;
2) One modulo $m-1$ addition $\langle w_1 + w_2 \rangle_{m-1}$;
3) One Inverse Isomorphic Transformations (IIT) to obtain the product.

The architecture of the isomorphic multiplier is shown in Fig. 5. Special attention has to be paid when one of the two operands is zero. In this case there exists no isomorphic correspondence and the modular adder has to be bypassed.
For example, for the modular multiplication $\langle 3 \cdot 4 \rangle_5 = 2$ using the isomorphic transformation of Table 2, we have

$$\begin{aligned} &1)\ 3 = \langle 2^3 \rangle_5 \overset{DIT}{\rightarrow} w_1 = 3 \\ &\quad\ 4 = \langle 2^2 \rangle_5 \overset{DIT}{\rightarrow} w_2 = 2 \\ &2)\ \langle 2 + 3 \rangle_4 = 1 \\ &3)\ 1 \overset{IIT}{\rightarrow} \langle 2^1 \rangle_5 = 2 \end{aligned}$$

Implementation of FIR filter modulo m

By using the isomorphism technique, the product of the two residues is transformed into the sum of their indices which are obtained by an isomorphic

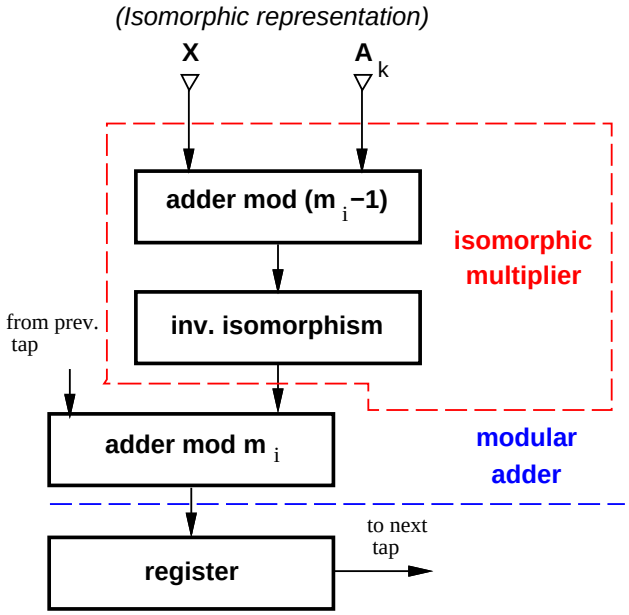

Fig. 6. Structure of RNS tap for filter in transposed form.

transformation. As a result, in each tap, the modular multiplication is reduced to a modular addition followed by an access to table (inverse isomorphism). The two input DIT tables of Fig. 5 do not need to be replicated in every tap. By observing that in computing the product $A_k X(n-k)$ the term X is common to all taps and it can be converted once in the input conversion unit, and that the term A_k can be stored directly as the index of the isomorphism. Therefore, the structure of each modular tap can be simplified as shown in Fig. 6.

4.3 Radix-2j Filter (RCNS)

Because of the radix-2j representation, the filter tap is simply implemented with a multiplier and an adder. We implement the multiplier as described in [10]. The complex $\underline{x}$ and $\underline{a}_k$ are converted in non-redundant QINS and then $\underline{a}_k$ is recoded into RCNS 2j, 2. The partial products (PPs) are then accumulated by a tree of arrays of signed-digit full-adders (SDFA) which operates in RCNS 2j, 3.
In RCNS 2j, 3, the complex number

$$\underline{X} = (X_{n-1}, \ldots, X_i, \ldots, X_1, X_0, X_{-1})$$

has digits in the set $X_i = \{\overline{3}, \overline{2}, \overline{1}, 0, 1, 2, 3\}$, which encoded in binary as

$$X_i = 2x_i^1 + x_i^0 \qquad \text{with } x_i^1, x_i^0 \in \{\overline{1}, 0, 1\} \tag{10}$$

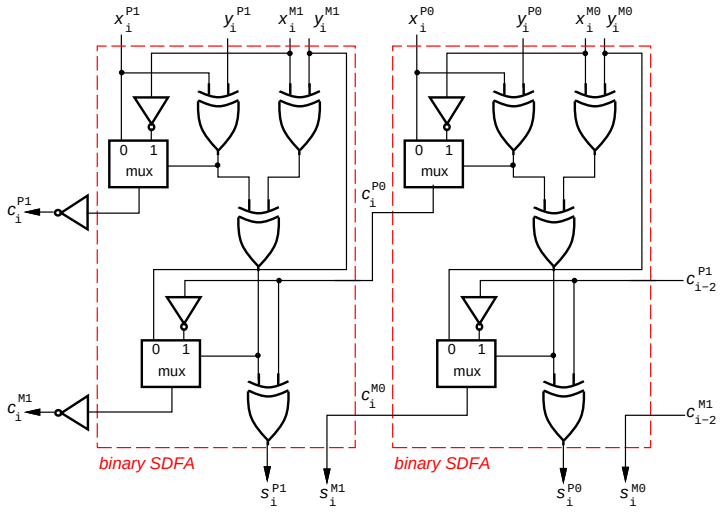

Fig. 7. Implementation of SD full-adder (SDFA).

Both x_i^1 and x_i^0 are then encoded with two bits each as shown in Table 3. Therefore, the resulting binary encoding of X_i is illustrated in Table 4. For bits are necessary to represent each RCNS 2j, 3 digit. With the encoding of Table 4 the SDFA of Fig. 7 can be derived.

x_i^1	x_i^{P1}	x_i^{M1}
1	0	1
0	0	0
0	1	1
1	1	0

x_i^0	x_i^{P0}	x_i^{M0}
1	0	1
0	0	0
0	1	1
1	1	0

Table 3. Binary encoding of x_i^1 and x_i^0.

By arranging the SDFAs in a tree the 10 PPs are reduced to 2 as shown in Fig. 8. An extra array of SDFAs adds the product $\underline{x} \cdot \underline{a}_k$ to the partial sum coming from the previous tap. As for the TCS case, we keep the carry-save representation of the digits until the last stage of the filter where we perform the conversion from RCNS 2j, 3 to radix-2 (binary) integers. Due to the CS representation of digits we need to store 8N bits in the tap's registers.

5 Filters Implementation

The filters are implemented in the 90 nm STM library of standard cells [17] and they have been synthesized by Synopsys Design Compiler. All the filters

X_i	x_i^1	x_i^0	x_i^{P1}	x_i^{M1}	x_i^{P0}	x_i^{M0}
$\overline{3}$	$\overline{1}$	$\overline{1}$	0	1	0	1
$\overline{2}$	$\overline{1}$	0	0	1	0	0
			0	1	1	1
$\overline{1}$	0	$\overline{1}$	0	0	0	1
			1	1	0	1
	$\overline{1}$	1	0	1	1	0
0	0	0	0	0	0	0
			1	1	0	0
			0	0	1	1
			1	1	1	1
1	0	1	0	0	1	0
			1	1	1	0
	1	$\overline{1}$	1	0	0	1
2	1	0	1	0	0	0
			1	0	1	1
3	1	1	1	0	1	0

Table 4. Binary encoding of X_i.

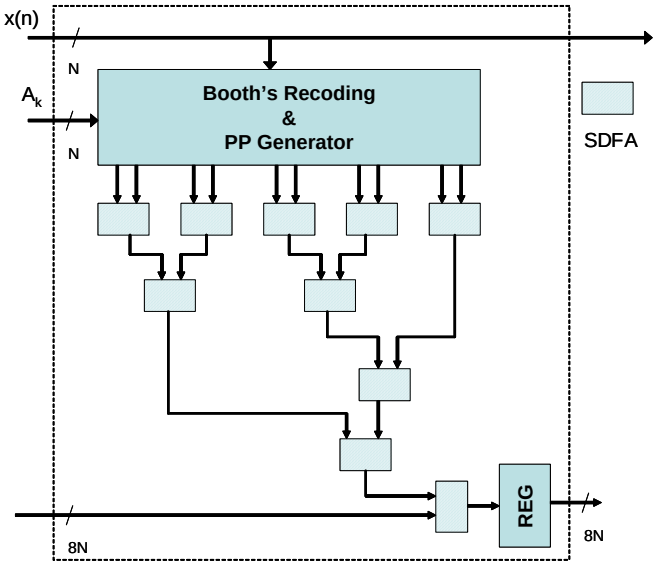

Fig. 8. Structure of RCNS tap.

can be clocked at $f_{max} = 300\ MHz$. By interpolating the results obtained by synthesis on filters of different order (number of taps), we obtain the trends shown in Fig. 9 for the area and Fig. 10 for the power. The values of area and power dissipation for the single tap (Fig. 2, Fig. 6 and Fig. 8) determine the slopes of the curves in the figures. The conversions from the TCS to the other

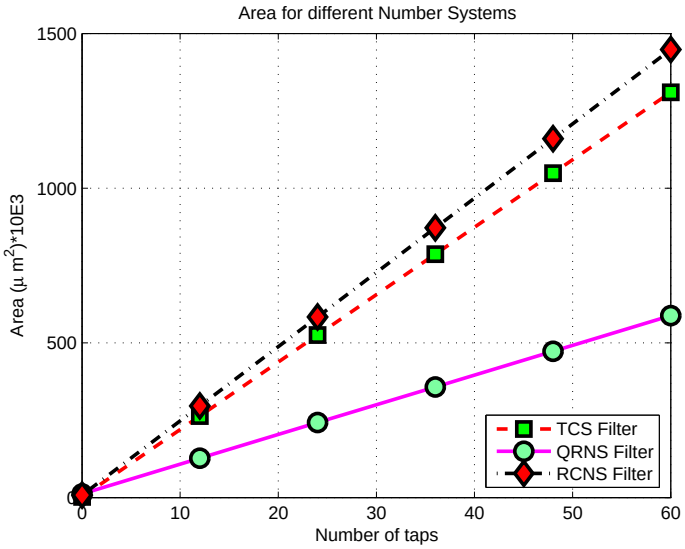

Fig. 9. Trends in area for increasing N.

	Area		P at 100 MHz	
	tap	conv.	tap	conv.
TCS	21.8K	2.0K	1.00	0.10
QRNS	9.6K	12.0K	0.25	1.20
RCNS	23.9K	8.0K	1.05	0.30
	$[\mu m^2]$		$[mW]$	

Table 5. Values of area and power dissipation.

number systems (and vice versa) are a constant contribution that does not depend on the number of taps, but only on the dynamic range of the filters. Table 5 reports the data for tap and conversion contribution for the three number systems.

The results show that complex filters implemented in QRNS consume significantly less power than the corresponding ones in TCS and RCNS. The expression for the power dissipated dynamically [18] in a system composed of n cells is

$$P_{dyn} = V_{DD}^2 f \cdot \sum_{i=1}^{n} C_{Li} a_i \tag{11}$$

where

V_{DD} is the power supply voltage;
f is the clock frequency;
C_{Li} is the load connected to the i-th cell (both active load and interconnections);

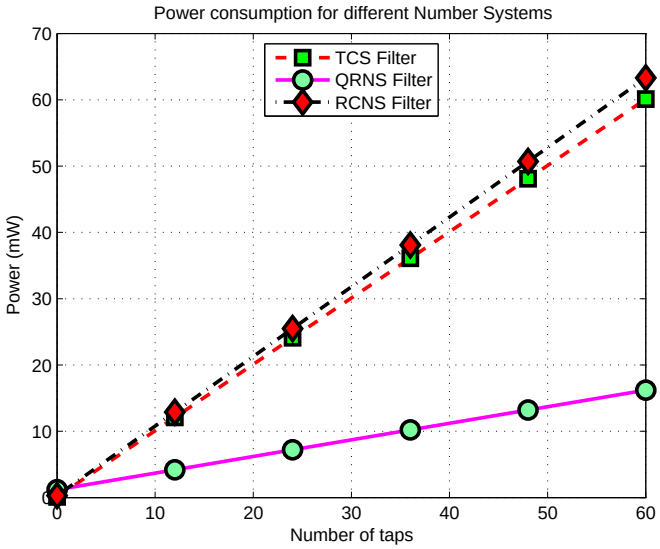

Fig. 10. Trends in power dissipation (at 100 MHz) for increasing N.

a_i is the activity factor of the i-th cell, which is the measure of how many transitions occur at its output. The activity factor is normally related to the clock $a_i \in [0, 1]$.

The lower power dissipation in the QRNS filter is due to the combination of two factors:

1. As clearly shown in Fig. 9, the smaller area results in a global reduced capacitance $\sum_{i=1}^{n} C_{Li}$ (including shorter interconnections).
2. The work in [19] showed that the number of transitions, i.e. the switching activity, for vectors of the same number of bits k, in RNS is lower than in TCS

$$\left(\sum_{i=1}^{k} a_i\right)_{RNS} < \left(\sum_{i=1}^{k} a_i\right)_{TCS}$$

Therefore, the switched capacitance $\sum_{i=1}^{n} C_{Li} a_i$, and by (11) the power consumption, in QRNS is smaller than in TCS and RCNS.

6 Conclusions

In this work, the use of different number representations for the implementation of complex FIR filters has been investigated.

Complex multipliers determine the performance, area and power dissipation of complex filters. Previously in [10], complex multipliers in TCS and

RCNS were evaluated, while in [9], complex filters in QRNS and TCS were compared. Here we extended the comparison to complex filters implemented in TCS, QRNS and RCNS.

The experimental results on complex filters with 20 bit dynamic range show that for the TCS and the RCNS the area and power dissipation are similar and confirms the findings of [10]. As for the QRNS, the results presented here, confirm those of [9], based on the implementation of TCS and QRNS complex filters in a 0.35 μm technology.

To summarize, this work shows that for complex high order FIR filters implementations based on QRNS offer significant advantages in area and power dissipation without any performance degradation.

References

1. A. V. Oppenheim and R. V. Shafer, *Digital Signal Processing.* Englewood Cliffs N.J.: Prentice Hall, 1995.
2. S. K. Mitra and K. Kaiser, *Handbook for Digital Signal Processing.* Wiley-Interscience, 1993.
3. R. W. Brodersen and M. S.-W. Chen, "Digital Complex Signal Processing Techniques for Impulse Radio," *Proc. of IEEE GLOBECOM '06 Global Telecommunications Conference*, pp. 1–5, Nov. 2006.
4. K. W. Martin, "Complex signal processing is not complex," *IEEE Transactions on Circuits and Systems I*, vol. 51, pp. 1823–1836, Sep. 2004.
5. P. S. Moharir, "Extending the scope of Golub's method beyond complex multiplication to binary converters," *IEEE Transactions on Computers*, vol. C-34, no. 5, pp. 484–487, 1985.
6. M. Sodestrand, W. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing.* New York: IEEE Press, 1986.
7. D. E. Knuth, *The Art of Computer Programming 2: Seminumerical Algorithms*, 3rd ed. Reading, MA: Addison-Wesley Publishing Company, 1998.
8. T. Aoki, H. Amada, and T. Higuchi, "Real/Complex Reconconfigurable Arithmetic using Redundant Complex Number Systems," *Proc. of 13th IEEE Symposium on Computer Arithmetic*, pp. 200–207, July 1997.
9. A. D'Amora, A. Nannarelli, M. Re, and G. C. Cardarilli, "Reducing Power Dissipation in Complex Digital Filters by using the Quadratic Residue Number System," *Proc. of 34th Asilomar Conference on Signals, Systems, and Computers*, pp. 879–883, Nov. 2000.
10. T. Aoki, K. Hosci, and T. Higuchi, "Reduntant Complex Arithmetic and its Application to Complex Multiplier Design," *Proc. of 29th IEEE International Symposium on Multiple-Valued Logic*, pp. 200–207, May 1999.
11. Y. Ohi, T. Aoki, and T. Higuchi, "Redundant Complex Number Systems," *Proc. of 25th IEEE International Symposium on Multiple-Valued Logic*, pp. 14–19, May 1995.
12. T. Aoki, Y. Ohi, and T. Higuchi, "Redundant Complex Number Arithmetic for High-Speed Signal Processing," *VLSI Signal Processing VIII (1995 IEEE Workshop on VLSI Signal Processing)*, pp. 523–532, Oct. 1995.

13. N. Szabo and R. Tanaka, *Residue Arithmetic and its Applications in Computer Technology*. New York: McGraw-Hill, 1967.
14. A. Avizienis, "Signed-Digit Number Representations for Fast Parallel Arithmetic," *IRE Trans. Electronic Computers*, vol. EC-10, pp. 389–400, Sep. 1961.
15. A. M. Nielsen and J.-M. Muller, "Borrow-Save Adders for Real and Complex Number Systems," *Proc. 2nd Conf. on Real Numbers and Computers*, Apr. 1996.
16. I. Vinogradov, *An Introduction to the Theory of Numbers*. New York: Pergamon Press, 1955.
17. STMicroelectronics, *90nm CMOS090 Design Platform*, http://www.st.com/stonline/prodpres/dedicate/soc/asic/90plat.htm.
18. N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, 2nd ed. Addison-Wesley Publishing Company, 1993.
19. T. Stouraitis and V. Paliouras, "Considering the alternatives in low-power design," *IEEE Circuits and Devices Magazine*, vol. 17, pp. 22–29, July 2001.