Reducing Power Dissipation in Complex Digital Filters by using the Quadratic Residue Number System *

Angelo D'Amora, Alberto Nannarelli, Marco Re and Gian Carlo Cardarilli
Department of Electrical Engineering
University of Rome "Tor Vergata" - Italy

http://dspvlsi.uniroma2.it/

Abstract

The aim of this work is to compare in terms of performance, area and power dissipation, a complex FIR filter realized in the traditional two's complement system with a Quadratic Residue Number System (QRNS) based one. The resulting implementations, designed to work at the same clock rate, show that the QRNS filter is almost half the size of the traditional one, and dissipates about one third of the energy.

1 Introduction

The new generation of telecommunication equipment often require the use of high order FIR filters for the implementation of the new modulation schemes. Moreover, low power consumption for new portable multimedia terminals is needed. In this context, computational intensive signal processing blocks can be effectively implemented by using Residue Number System (RNS) arithmetic.

The use of the RNS allows the decomposition of a given dynamic range in slices of smaller range on which the computation can be efficiently implemented in parallel [1], [2], [3]. The QRNS (Quadratic RNS) is particularly convenient when dealing with complex numbers [4] [5]. In QRNS the imaginary term of a complex number is transformed into an integer, therefore a complex multiplication which requires four integers multiplications and two sums in the conventional two's complement system, is implemented with two integer multiplications in QRNS.

The drawback presented by the RNS (and QRNS) is the overhead due to both input and output conversions binary-RNS-binary. This problem can be solved by using efficient conversion techniques [6] [7], or by converting directly the analog signal in the residue representation [8].

Recently, a number of works on low power and RNS have been presented. In [9] and [10] the power dissipation is reduced by taking advantage of the speed-up due to the parallelism of the RNS structure. The supply voltage is reduced, resulting in a quadratic reduction of power, until the speed-up $= 1$ [9], or until the desired value of delay [10]. In [11] some encoding optimization techniques for small moduli are presented.

In our work, we compare the performance, area and power of a complex FIR filter realized with the traditional binary arithmetic, with a QRNS based one.

Both filters have been designed according to the specifications of an actual filter used in a telecommunication satellite and are clocked at the same rate of 166 MHz. Although the QRNS filter has a longer latency, it can sustain the same throughput of the traditional one, while its area and power dissipation are about 57% of the total area and 34% of the total power of the traditional filter.

2 Background

A Residue Number System is defined by a set of relatively prime integers

$$\{m_1, m_2, \ldots, m_P\} \ .$$

The dynamic range of the system is given by the product of all the moduli m_i:

$$M = m_1 \cdot m_2 \cdot \ldots \cdot m_P \ .$$

Any integer $X \in \{0, 1, 2, \ldots m-1\}$ has a unique RNS representation given by:

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

* 34th Asilomar Conference on Signals, Systems, and Computers, Asilomar Hotel and Conference Grounds, Pacific Grove, CA, USA. Oct. 29 - Nov. 1, 2000.

where $\langle X \rangle_{m_i}$ denotes the operation $X \ mod \ m_i$. Operations on single moduli are done in parallel

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

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

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

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

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_i (Z_{m_i}). A complex number $x_R + jx_I = (x_R, x_I) \in Z_{m_i}$, with q root of $q^2 + 1 = 0$ in Z_{m_i} has a unique Quadratic Residue Number System representation given by

$$\begin{array}{ll} (x_R, x_I) \overset{QRNS}{\rightarrow} & (X_i, \hat{X}_i) \quad i = 0, 1, \ldots, P \\ & X_i = \langle x_R + g \cdot x_I \rangle_{m_i} \\ & \hat{X}_i = \langle x_R - g \cdot x_I \rangle_{m_i} \end{array}$$

The inverse QRNS transformation is given by

$$\begin{array}{l} x_R = \langle 2^{-1}(X_i + \hat{X}_i) \rangle_{m_i} \\ x_I = \langle 2^{-1} \cdot q^{-1}(X_i - \hat{X}_i) \rangle_{m_i} \end{array}$$

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

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

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

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

and it is realized by using two integers multiplications instead of four. Table 1 shows an example of QRNS multiplication in the ring modulo 13.

A complex N taps FIR filter (Figure 1) is expressed by

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

where $\underline{x}, \underline{y}, \underline{a}_k$ denotes complex quantities. From the QRNS theory described above, it is easy to derive for the complex filter the structure shown in Figure 2, in which both portions of the filter are realized with P RNS filters working in parallel.

example for $m = 13$:	
$q = q_1 = 5 \leftrightarrow \langle 5 \cdot 5 \rangle_{13} = -1$	
$(x_R + jx_I)(y_R + jy_I) = (3 + j)(2 + j2) = 4 + j8$	
conversion to QRNS	
$X = \langle 3 + 5 \cdot 1 \rangle_{13} = 8$	$Y = \langle 2 + 5 \cdot 2 \rangle_{13} = 12$
$\hat{X} = \langle 3 - 5 \cdot 1 \rangle_{13} = 11$	$\hat{Y} = \langle 2 - 5 \cdot 2 \rangle_{13} = 5$
multiplications	
$X \cdot Y = \langle 8 \cdot 12 \rangle_{13} = 5$	$\hat{X} \cdot \hat{Y} = \langle 11 \cdot 5 \rangle_{13} = 3$
conversion from QRNS	
$Z_R = \langle 7(5 + 3) \rangle_{13} = 4$	being $2^{-1} = 7$
$Z_I = \langle 7 \cdot 8(5 - 3) \rangle_{13} = 8$	being $q^{-1} = 8$

Table 1. Example of QRNS multiplication mod 13.

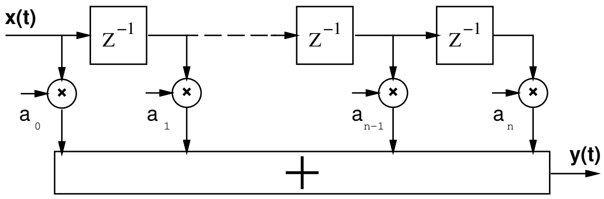

Figure 1. FIR filter in direct form.

3 Traditional FIR Filter

The starting point of our design is a programmable 64-tap FIR filter realized in direct form (Figure 1) with complex input and coefficients size of 10 bits for the real part and 10 bits for the imaginary part. These data are derived from the specification of an actual digital filter, used aboard a satellite for direct TV broadcasting. We designed a prototype filter in traditional two's complement system in order to compare its performance, area and power dissipation with a QRNS filter.

The filter can be decomposed in a real and imaginary part. A single tap is realized as sketched in Figure 3. The real and imaginary products are realized with two Booth multipliers [12] and the resulting partial products are accumulated in a Wallace's tree structure which produces a carry-save (CS) representation of the product in each side

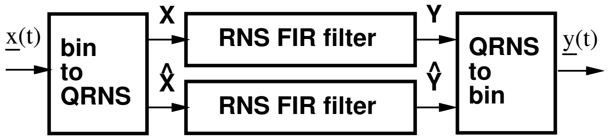

Figure 2. Structure of QRNS filter.

of the filter. The CS representation of the products, is then accumulated in two 128-addend Wallace's tree realized with 4:2 compressors [13], not depicted in Figure 3. To have an error-free filter we must keep a number of bits sufficient to hold the carry-save representation of the sum, and we need a $20 + \log_2 64$ wide tree. The carry-save representation is finally converted into two's complement representation by a carry-propagate adder (realized with a carry-look-ahead scheme) in the last stage of the filter (both real and imaginary sides).

The filter has been implemented in the AMS $0.35\mu m$ standard cells library, and it was synthesized from VHDL description using Synopsys and a constraint of 6 ns as a critical path, for this reason it resulted in a pipelined filter of 6 stages.

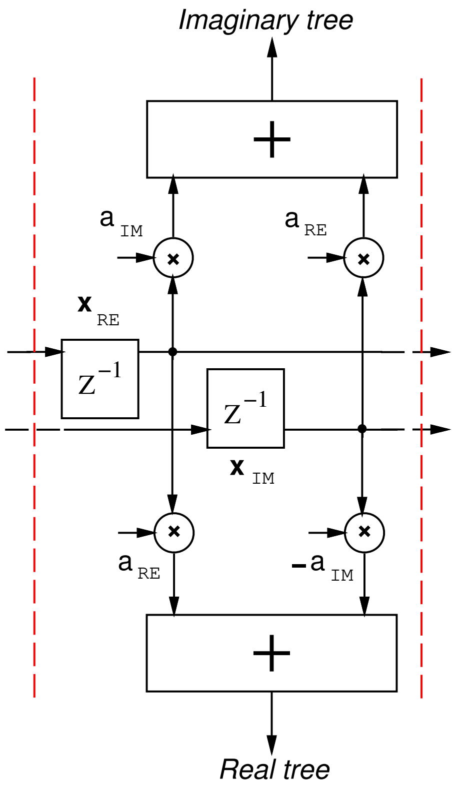

Figure 3. Structure of tap in traditional complex FIR filter.

4 QRNS FIR Filter

From Figure 2 we can see that the QRNS filter can be realized by two RNS filters in parallel. 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 in the case of the traditional implementation, we chose the following set of moduli:

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

such that

$$log_2(5 \cdot 13 \cdot 17 \cdot 29 \cdot 41) = 20.3\ .$$

4.1 Implementation of modular multiplication

In each tap, a modular multiplier is needed to compute the term $\langle a_k x(n-k)\rangle_{m_i}$. Because of the complexity of modular multiplication, we used the isomorphism technique [14] to implement the product of residues. By using isomorphism, the product of the two residues is transformed into the sum of their indices which are obtained by an isomorphic transformation. According to [14], if m is prime there exists a primitive radix r such that its powers modulo m cover the set $[1, m-1]$:

$$n_i = \langle r^{w_i}\rangle_m \qquad \text{with} \qquad \begin{array}{l} n_i \in [1, m-1] \\ w_i \in [0, m-2]. \end{array}$$

Both transformations $n \to w$ and $w \to n$ can be implemented with $m-1$ entries tables. Therefore, the product of a_1 and a_2 modulo m can be obtained as:

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

where

$$w = \langle w_1 + w_2\rangle_{m-1} \qquad \text{with} \qquad \begin{array}{l} a_1 = \langle r^{w_1}\rangle_m \\ a_2 = \langle r^{w_2}\rangle_m \end{array}$$

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

i) Two isomorphic transformations to obtain w_1 and w_2;

ii) One modulo $m-1$ addition $\langle w_1 + w_2\rangle_{m-1}$;

iii) One inverse isomorphic transformations to obtain the product.

For example, for the modular multiplication

$$\langle 3 \cdot 4\rangle_5 = 2$$

we have ($r = 2$):

$$\begin{array}{rl} i) & 3 = \langle 2^3\rangle_5 \to w_1 = 3 \\ & 4 = \langle 2^2\rangle_5 \to w_2 = 2 \\ ii) & \langle 2+3\rangle_4 = 1 \\ iii) & \langle 2^1\rangle_5 = 2 \end{array}$$

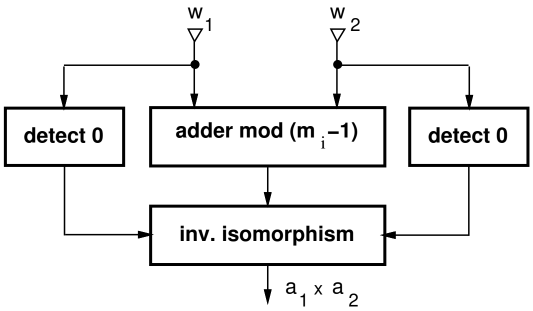

Figure 4. Multiplication implemented by isomorphism.

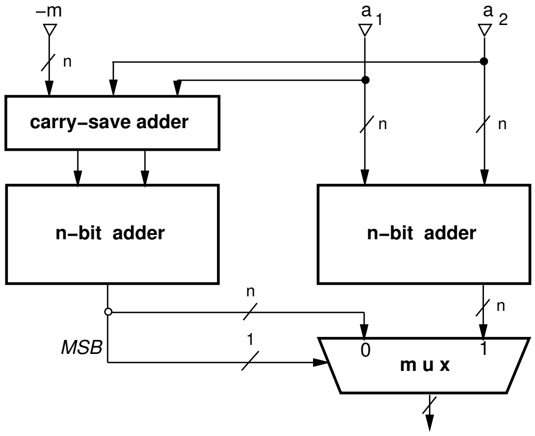

Figure 5. Adder modulo m.

The input x, although delayed, is the multiplicand of all the multiplications (see Figure 1). For this reason only one isomorphic transformation, incorporated in the binary to QRNS conversion, is necessary for all the taps. On the other hand, because the coefficients of the filter (multiplicators) are constant terms loaded once at start-up, it is convenient to load directly the isomorphic representation modulo $m_i - 1$. As a result, in each tap, we reduce the modular multiplication to a modular addition followed by an access to table (inverse isomorphism) as depicted in Figure 4. The table is implemented as synthesized logic and 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.

4.2 Implementation of modular addition

The modular addition $\langle a_1 + a_2 \rangle_m$, consists of two binary 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 addition is the result of the three-term addition. The above algorithm can be implemented with two $\lceil \log_2 m \rceil$-bit adders as shown in Figure 5.

At the output of the tree, it is necessary to reduce the sum S of all the taps to $\langle S \rangle_{m_i}$. This is done with the modulo-reduction technique described in [6].

4.3 Implementation of input/output conversions

As already mentioned, the input conversion block includes the isomorphic transformation. If x is zero, there is no exponent w such that $\langle r^w \rangle_{m_i} = 0$. As a consequence, zero is encoded with a special pattern that is then detected in the block which computes the product using the isomorphism (Figure 4).

The output conversion is implemented by using the Chinese Remainder Theorem (CRT), as described in [6].

5 Results and Comparisons

Both the traditional and the QRNS filters were implemented in the AMS $0.35\mu m$ standard cells library. Delay, area and power dissipation have been determined with Synopsys tools.

Table 2 summarizes the results. In the table, area is reported as number of NAND2 equivalent gates and power is computed at 166 MHz. However, both area and power dissipation do not take into account the contribution of interconnections.

Table 2 shows that the QRNS filter has a higher latency, due to the conversions, but it can be clocked at the same rate of the traditional filter, and consequently, it can sustain the same throughput. However, the QRNS filter is almost half the area on the traditional complex filter, and consumes one third of the energy.

6 Conclusions

We have implemented a QRNS 64-taps FIR filter and compared its delay, area and power dissipation with those of

Filter	Cycle $[ns]$	Latency (cycles)	Area (gate equiv.)	Power $[W]$
QRNS	6.0	$11+64$	$182,400$	2.5
Trad.	6.0	$6+64$	$315,700$	7.4
ratio	1.0	1.07	0.57	0.34

Table 2. Summary of results.

a corresponding complex FIR filter realized with the traditional two's complement system. The results obtained show that the QRNS filter can sustain the same clock rate, although it has a slightly longer latency. However, in terms of area and power the QRNS version is more convenient. A better improvement is expected for filters with a larger number of taps.

Acknowledgements

This work was partially supported by MURST National Project: Codesign Methods for Low Power Integrated Circuits.

References

[1] N.S. Szabo and R.I. Tanaka. *Residue Arithmetic and its Applications in Computer Technology*. New York: McGraw-Hill, 1967.

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

[3] M.A. Soderstrand and K.Al Marayati. Vlsi implementation of very high-order fir filters. *IEEE International Symposium on Circuits and Systems (ISCAS'95)*, Vol. 2:1436–1439, 1995.

[4] F. J. Taylor, G. Papadourakis, A. Skavantzos, and A. Stouraitis. A radix-4 FFT using complex RNS arithmetic. *IEEE Transactions on Computers*, Vol. C-34:573–576, June 1985.

[5] M. Abdallah and A. Skavantzos. On the binary quadratic residue system with noncoprime moduli. *IEEE Transactions on Signal Processing*, Vol. 45:2085–2091, Aug. 1997.

[6] G. Cardarilli, M. Re, and R. Lojacono. A residue to binary conversion algorithm for signed numbers. *European Conference on Circuit Theory and Design (ECCTD'97)*, Vol. 3:1456–1459, 1997.

[7] G. Cardarilli, M. Re, R. Lojacono, and G. Ferri. A new efficient architecture for binary to rns conversion. *Proc. of European Conference on Circuit Theory and Design (ECCTD'99)*, Vol. 2:1151–1154, 1999.

[8] A.P. Preethy and D. Radhakrishnan. A vlsi architecture for analog-to-residue conversion. *Third International Conference on Advanced A/D and D/A Conversion Techniques and Their Applications*, pages 83–85, 1999.

[9] M. Bhardwaj and A. Balaram. Low power signal processing architectures using residue arithmetic. *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ASSP'98)*, Vol. 5:3017–3020, 1998.

[10] W.L. Freking and K.K. Parhi. Low-power digital filters using residue arithmetic. *Thirty-First Asilomar Conference on Signals, Systems and Computers*, Vol. 1:739–743, 1998.

[11] M.N. Mahesh and M. Mehndale. Low power realization of residue number system based fir filters. *Thirteenth International Conference on VLSI Design*, pages 30–33, 2000.

[12] Israel Koren. *Computer Arithmetic Algorithms*. Prentice-Hall, Inc. , 1993.

[13] M.D. Ercegovac and T. Lang. *Division and Square Root: Digit-Recurrence Algorithms and Implementations*. Kluwer Academic Publisher, 1994.

[14] I.M. Vinogradov. *An Introduction to the Theory of Numbers*. New York: Pergamon Press, 1955.