Impact of RNS Coding Overhead on FIR Filters Performance

Gian Carlo Cardarilli, Andrea Del Re†, Alberto Nannarelli* and Marco Re

Department of Electronics, University of Rome Tor Vergata, Rome, Italy
†Skytechnology s.r.l., Rome, Italy
*Dept. of Informatics & Math. Modelling, Technical University of Denmark, Kongens Lyngby, Denmark

***Abstract*— In this paper a design space exploration for FIR filter implementations in Residue Number System (RNS) is presented. The exploration regards different aspects of the RNS FIR filter design such as the dynamic range, the overhead due to the coding of the RNS base with respect to the application dynamic range, and delay-area tradeoffs. The design space exploration and its results, are helpful in evaluating the effects of the RNS coding overhead and to choose an efficient filter architecture trading-off filter order, dynamic range, clock frequency and area.**

I. Introduction

The use of non traditional number systems in the implementation of application specific Digital Signal Processing (DSP) systems is gaining importance due to the needs of strong optimizations in terms of power consumption and speed. This is mainly related to the extremely fast growth in complexity of the algorithms to be implemented on silicon often targeting portable platforms. The renewed interest in these alternative number systems is also related to the availability of hardware platforms (FPGAs) and technologies suitable for the implementation of arithmetic blocks that are based on the use of look-up tables.

A number of papers and books illustrating DSP applications based on the Logarithmic Number System (LNS) [1] and the Residue Number System (RNS) [2], [3] have been presented in the literature. Recently, studies about power consumption in LNS/RNS and the implementation of FIR filters have been presented in [4], [5], [6], [7] and [8]. In [9] the authors show that filters implemented in RNS not only are convenient in terms of dynamic power dissipation, but also in terms of static power reduction when state-of-the-art dual threshold transistor libraries are used in the synthesis process.

In [8] and [9] the power dissipation of RNS FIR filters is characterized as a function of the filter order for a fixed dynamic range. In this paper, an exploration of the design space for FIR filters considering delay-area tradeoffs as a function of the filter dynamic range is presented.

The Design Space Exploration (DSE) is carried out taking advantage of the results of a characterization of the composing blocks of the filters. Moreover, the DSE is validated by actual implementations of a selected set of complete filters.

The results show that for high-order and large dynamic range, that are typical of many today's applications, filters implemented in RNS are significantly smaller than filters implemented in the traditional two's complement number system (TCS).

II. Background and Metrics

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 $X \ mod \ m_i$ [10]. 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 [10].

By defining D the dynamic range of an application, which corresponds to d bits ($D = 2^d$) in the two's complement number system (TCS), the RNS base is chosen such that

$$\prod_{i=1}^{P} m_i = M \geq D = 2^d \ .$$

Because each modulus of the RNS base is encoded in binary for a total number of bits

$$b = \sum_{i=1}^{P} \lceil \log_2 m_i \rceil \ ,$$

we define $\mathrm{OH} = b - d$ as the *RNS coding overhead*.

The effect of this overhead is investigated as a function of the dynamic range in the implementation of FIR filter architectures. Specifically, we evaluate all combinations of P prime moduli covering the required dynamic range, select the suitable moduli set with different criteria (see Section IV), and compare in terms of delay and area the RNS implementation of the filter with the TCS implementation. We do not consider power-of-two moduli ($m_i = 2^k$) because they do not add coding overhead with respect to the TCS.

978-1-4244-2110-7/08/$25.00 ©2007 IEEE

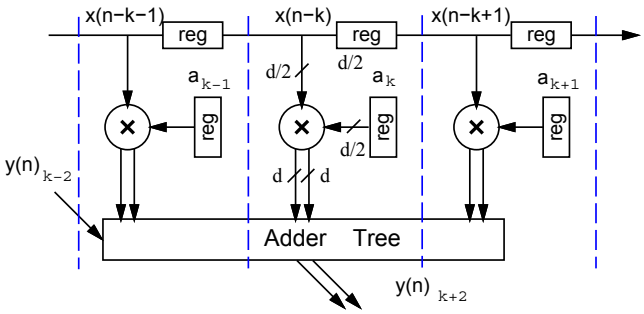

Fig. 1. TCS FIR filter in direct form.

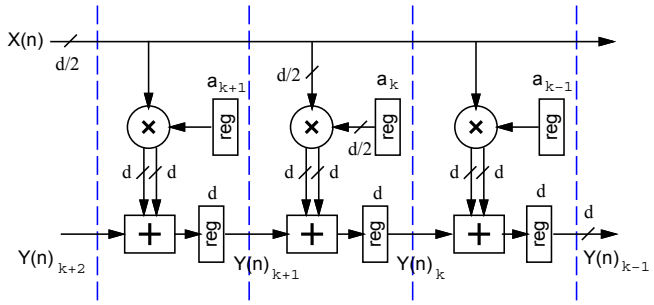

Fig. 2. TCS FIR filter in transposed form.

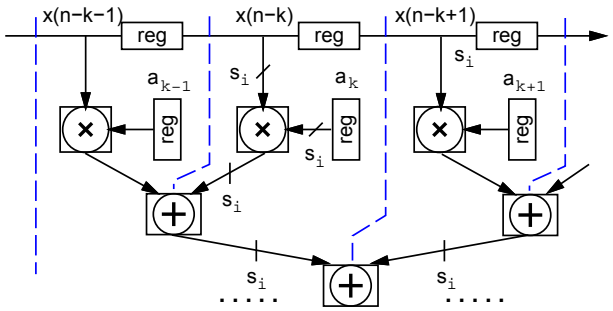

Fig. 4. RNS FIR filter in direct form.

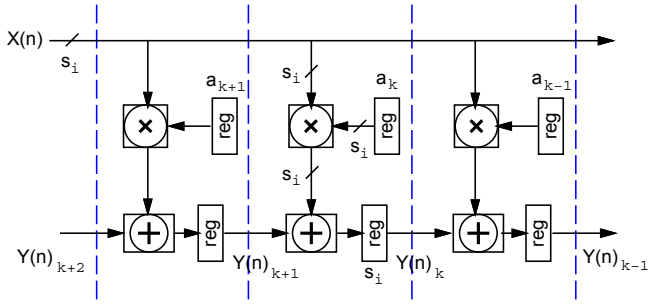

Fig. 5. RNS FIR filter in transposed form.

III. FIR FILTER ARCHITECTURES

A Finite Impulse Response (FIR) filter of order N is described by the expression

$$y(n) = \sum_{k=0}^{N-1} a_k x(n-k) \qquad (2)$$

We consider error-free programmable FIR filters in both direct and transposed form implemented in RNS and TCS.

A. FIR filter in TCS

The straightforward implementation of (2) is the implementation of the filter in direct form as shown (for a portion of the filter) in Fig. 1.

We assume to have a dynamic range of d bits, generated at the output of $\frac{d}{2} \times \frac{d}{2}$ square multipliers, which guarantees error-free operations for the given N. In order to keep the delay small, the multiplication is performed carry-save and, therefore, the input to the adder tree consists of $2N$ d-bit operands. The carry-save representation of $y(n)$ is converted in two's complement by a carry-propagate adder (CPA) placed at the output of the tree.

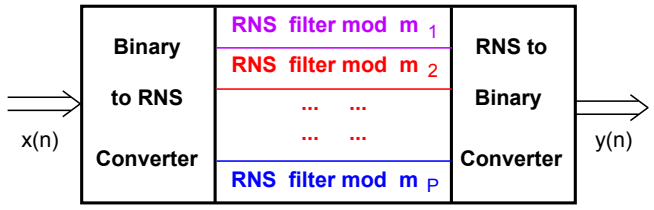

Fig. 3. FIR filter implemented in RNS

Alternatively, expression (2) can be implemented in transposed form as shown in Fig. 2. We make the same assumptions for the dynamic range as for the direct form filter case, and perform again carry-save multiplication, but accumulate (by using a CPA) the partial value of $y(n)$ in each register. Although this solution is not the fastest, the area is reduced to a minimum.

The storage, in number of bits, required for each tap is $2 \times \frac{d}{2} = d$ bits for filters in direct form, and $\frac{d}{2} + d = \frac{3}{2}d$ bits for filters in transposed form.

B. FIR filter in RNS

As a direct consequence of (1) and (2), a FIR filter is implemented in RNS by decomposing it into P filters working in parallel, as sketched in Fig. 3.

Each of the P RNS filters can be implemented in either direct or transposed form with schemes similar to those of Figs. 1 and 2, and by replacing multipliers and adders with their modular counterparts. By choosing prime moduli of limited wordlength ($m_i < 2^6$), the modular multiplication can be efficiently implemented by isomorphism [11] (see [12] for the implementation). The resulting RNS filter architectures are shown in Fig. 4 (direct form) and Fig. 5 (transposed form). In the figures, the bitwidth $s_i = \lceil \log_2 m_i \rceil$ in each filter mod m_i is such that $b = \Sigma_{i=1}^{P} s_i$.

Differently from the TCS case, the storage bits required per tap is the same for the direct and transposed form: $2 \times b$ bits.

IV. DESIGN SPACE EXPLORATION

The design space exploration is carried out to identify the tradeoffs between filters realized in TCS and RNS with respect

to dynamic range, clock frequency (throughput) and area.

We use the architectures presented in the previous section for TCS and RNS filters in direct and transposed form. For filters in direct form, the size and depth of the adder tree depend on the filter order N. For this reason, we have characterized TCS and RNS direct form filters of the same order (number of taps).

In the DSE, we run the following *experiments*:

EXP-1: In this experiment, we consider the RNS moduli selected to obtain the *"worst case"* coding overhead (OH) conditions for the RNS filters.

EXP-2: In this experiment, we consider the RNS moduli selected to obtain the best tradeoff delay-area for the given dynamic range.

EXP-3: The moduli are chosen as in EXP-2, but the evaluation is carried on a complete N-tap filter.

In EXP-1 and EXP-2, we do not include the TCS/RNS/TCS conversion and consider the area value per tap, while in EXP-3 we take into account the impact of the converters on FIR filters of the same order N.

The first two experiments are based on the characterization of the filter composing blocks (multipliers, adders, registers, etc.) performed by Synopsys Design Compiler and the STM 90 nm library of standard cells [13]. The results of EXP-3 are obtained by synthesis of the actual filters.

EXP-1: worst case OH

In this experiment, the choice of moduli (RNS base) is done by selecting the set that for the given dynamic range has the largest RNS coding overhead $OH = b - d$. This is done in two steps:

1) The group of RNS bases for which $M \geq 2^d$ are first selected;
2) The base which has the largest b is selected. In case of tie, the set with the largest number of moduli P is chosen.

The results of this selection are shown in Table I.

In EXP-1, we consider a very large clock period so that the evaluation is based on the results of the characterization of circuits optimized for minimum area.

Because for the RNS bases of Table I there is not a significant difference in area per tap between the filters in direct and transposed form, in Fig. 6 we plot the curves[1] of direct and transposed TCS (D-TCS and T-TCS) and just one curve for the RNS filters. The area unit is 1 μm^2.

Fig. 6 show that for dynamic ranges d from 12 to 24, the area is roughly the same for the four architectures (D-TCS is slightly better), and that for $d > 24$ bits RNS filters are significantly smaller than TCS.

EXP-2: best delay-area tradeoff

The results of EXP-1 are obtained for the worst case choice of the RNS base and considering a very long clock period. In this experiment, we make more realistic assumptions by introducing a timing constraint on the clock frequency (500 MHz) and by selecting the moduli according not to the worst case OH, but to the results of the characterization of delay/power/area done with a RNS filter design tool [14]. Given a target clock period, the tool selects the RNS base which guarantees the smallest area (or lower power dissipation). The selected RNS bases are reported in Table II.

For filters in direct form (see Fig. 1 and Fig. 4) the timing constraint might cause modifications in the filter architecture when for deep adder trees (large N) it might be necessary to break the critical path and introduce pipeline registers. For this reason, we limit EXP-2 to the case of filters in transposed form, where the critical path is independent[2] of the number of taps (see Fig. 2 and Fig. 5).

The results of the experiment are plotted in Fig. 7. The dotted curves refer to the corresponding results of EXP-1 (Fig. 6). The figure clearly shows that, when a timing constraint ($T_C = 2.0$ ns) is introduced, the taps in the TCS filter grow larger (tradeoff speed-area), while for the RNS this growth is compensated by a choice of the moduli set which minimizes the area. The difference is larger as the dynamic range increases. For $d = 48$ the area of the TCS is double than the RNS filter.

d	**Moduli Set**	b	$b-d$
12	{ 5, 7, 11, 17 }	15	3
16	{ 5, 7, 11, 17, 19 }	20	4
20	{ 3, 5, 11, 17, 19, 37 }	25	5
24	{ 3, 5, 11, 13, 17, 19, 37 }	29	5
28	{ 3, 5, 11, 13, 17, 19, 23, 29 }	33	5
32	{ 3, 5, 7, 11, 13, 17, 19, 37, 41 }	38	6
36	{ 3, 5, 7, 11, 13, 17, 19, 23, 29, 37 }	42	6
40	{ 3, 5, 11, 13, 17, 19, 37, 41, 43, 47 }	47	7
44	{ 3, 5, 11, 13, 17, 19, 23, 29, 37, 41, 43 }	51	7
48	{ 3, 5, 7, 11, 17, 19, 23, 29, 31, 37, 41, 43 }	55	7

TABLE I
MODULI SET AND DYNAMIC RANGE FOR EXP-1.

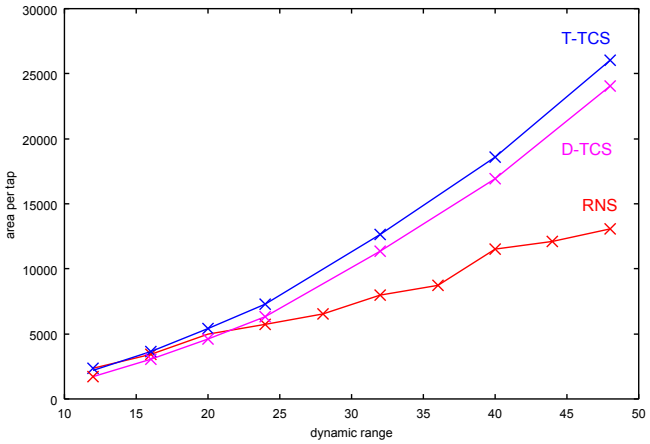

Fig. 6. Results of EXP-1: direct and transposed TCS vs. RNS

[1]Actually the curve is a visual aid to better display the distribution of the experiment's discrete points.

[2]We do not consider the increased buffering on x for large N in this experiment.

d	**Moduli Set**	b	$b-d$
12	{ 3, 7, 13, 17 }	14	2
16	{ 5, 7, 11, 13, 17 }	19	3
20	{ 3, 7, 11, 13, 17, 23 }	23	3
24	{ 3, 7, 11, 13, 17, 19, 23 }	28	4
28	{ 3, 5, 7, 13, 17, 19, 23, 31 }	32	4
32	{ 3, 5, 7, 11, 13, 17, 19, 29, 31 }	36	4
36	{ 5, 7, 11, 13, 17, 23, 29, 31, 41 }	40	4
40	{ 11, 13, 17, 19, 23, 29, 31, 37, 41 }	45	5
44	{ 17, 19, 23, 29, 31, 37, 41, 43, 47 }	49	5
48	{ 23, 29, 31, 37, 43, 47, 53, 59, 61 }	51	3

TABLE II

MODULI SET AND DYNAMIC RANGE FOR EXP-2.

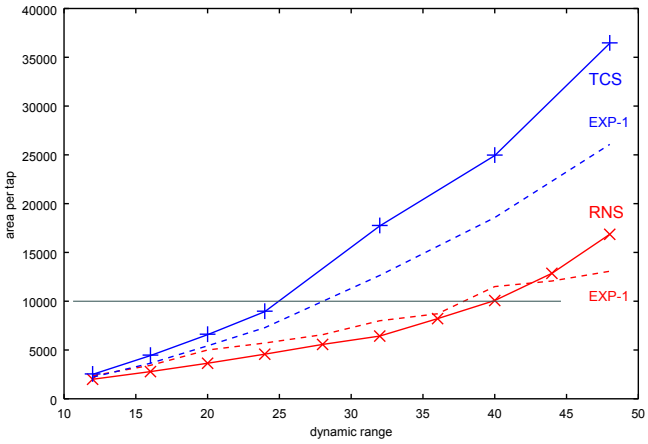

Fig. 7. Results of EXP-2: transposed TCS vs. RNS

Moreover, by tracing horizontal lines in Fig. 7, we can roughly determine the truncated TCS tap equivalent to the error-free RNS tap. For example, for $d = 40$ the RNS area per tap is $10,000\ \mu m^2$. This area is equivalent to that of a truncated TCS with dynamic range $d = 24$, that is a TCS filter with error $\geq 2^{16}$ unit-in-last-position with respect to the RNS.

EXP-3: complete N-order filter

In this experiment we validate the results of the DSE with an actual implementation to see the impact of the TCS/RNS and RNS/TCS converters.

From Fig. 7 we select $d = 16$ and determine the filter order N for which the RNS has a smaller area than the TCS filter. We implement 10, 50 and 100-tap complete filters in both TCS and RNS and plot the data-points, together with their trends in Fig. 8. From the figure we can see that for $N^\star > 12$ taps the RNS filter is smaller than the TCS filter.

V. CONCLUSIONS

In this work we investigated the tradeoffs between TCS and RNS filters in terms of filter order, dynamic range, clock frequency and area. With a few exceptions for very small dynamic ranges and worst case selection of the RNS base, filters implemented in RNS are significantly smaller than the corresponding filters implemented in the two's complement system (TCS).

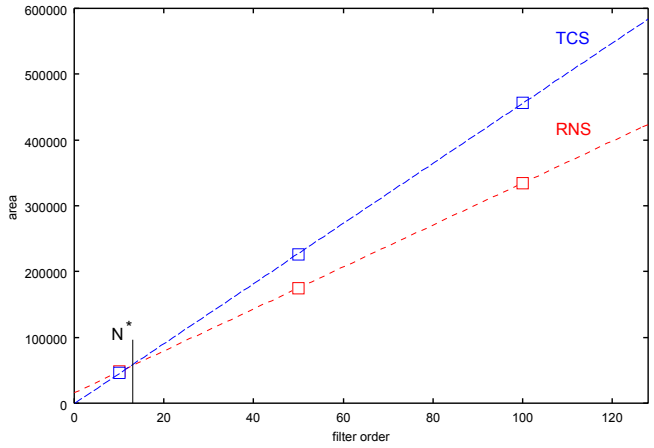

Fig. 8. Results of EXP-3: complete filter TCS vs. RNS

ACKNOWLEDGMENTS

This work was partially supported by University of Rome Tor Vergata, Department of Electronics and Denmark Technical University, Department of Informatics and Mathematical Modelling through visiting professor grants.

REFERENCES

[1] E. Swartzlander, "The Sign/Logarithm Number System," *IEEE Transactions on Computers*, vol. 24, pp. 1238–1242, Dec. 1975.

[2] 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.

[3] S. Mitra and J. F. Kaiser, *Handobook for Digital Signal Processing*. John Wiley & Sons, 1993.

[4] T. Stouraitis and V. Paliouras, "Considering the alternatives in low-power design," *IEEE Circuits and Devices Magazine*, vol. 17, pp. 22–29, July 2001.

[5] 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, pp. 3017–3020, 1998.

[6] W. Freking and K. Parhi, "Low-power digital filters using residue arithmetic," *Thirty-First Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 739–743, 1998.

[7] M. Mahesh and M. Mehndale, "Low power realization of residue number system based FIR filters," *Thirteenth International Conference on VLSI Design*, pp. 30–33, 2000.

[8] G. C. Cardarilli, A. Del Re, R. Lojacono, A. Nannarelli, and M. Re, "RNS Implementation of High Performance Filters for Satellite Demultiplexing," *Proc. of IEEE Aerospace Conference*, vol. 3, pp. 1365–1379, Mar. 2003.

[9] G. Cardarilli, A. D. Re, A. Nannarelli, and M. Re, "Low power and low leakage implementation of RNS FIR filters," *Proc. of 39th Asilomar Conference on Signals, Systems, and Computers*, pp. 1620–1624, Oct. 2005.

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

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

[12] A. Nannarelli, G. C. Cardarilli, and M. Re, "Power-delay tradeoffs in residue number system," *IEEE International Symposium on Circuits and Systems (ISCAS 2003)*, vol. 5, pp. 413–416, May 2003.

[13] STMicroelectronics. 90nm CMOS090 Design Platform. [Online]. Available: http://www.st.com/stonline/prodpres/dedicate/soc/asic/90plat.htm

[14] A. Del Re, A. Nannarelli, and M. Re, "A tool for automatic generation of RTL-level VHDL description of RNS FIR filters," *Proc. of 2004 Design, Automation and Test in Europe Conference (DATE)*, vol. 48, pp. 686–687, Feb. 2004.