Analysis of RNS-FPL Synergy for High Throughput DSP Applications: Discrete Wavelet Transform

Javier Ramírez [1], Antonio García [1], Pedro G. Fernández [2],
Luis Parrilla [1] and Antonio Lloris [1]
[1] Dept. of Electronics and Computer Technology,
Campus Universitario Fuentenueva, 18071 Granada, Spain
{jramirez, agarcia, lloris, lparrilla}@ditec.ugr.es
[2] Dept. of Electrical Engineering,
Escuela Politécnica Superior, 23071 Jaén, Spain
pfernan@ujaen.es
http://ditec.ugr.es/

Abstract. This paper focuses on the implementation over FPL devices of high throughput DSP applications taking advantage of RNS arithmetic. The synergy between the RNS and modern FPGA device families, providing built-in tables and fast carry and cascade chains, makes it possible to accelerate MAC intensive real-time and DSP systems. In this way, a slow high dynamic range binary 2's complement system can be partitioned into various parallel and high throughput small word-length RNS channels without inter-channel carry dependencies. To illustrate the design methodology, novel RNS-based architectures for multi-octave orthogonal DWT and its inverse are implemented using structural level VHDL synthesis. Area analysis and performance simulation are conducted. A relevant throughput improvement for the proposed RNS-based solution is obtained, compared to the equivalent 2's complement implementation.

1 Introduction

FPL devices have recently generated interest for use in Digital Signal Processing systems due to their ability to implement custom solutions while maintaining flexibility through device reprogramming. These devices represent an alternative to DSPs and ASICs since a significant performance improvement over DSPs, with moderately low development cost, can be obtained while maintaining system programmability.

Furthermore, the number of applications requiring high-precision and high-bandwidth signal processing is constantly increasing and the development of structures for these systems is of growing interest. Traditional binary 2's complement arithmetic systems suffer from long carry propagation delays. Thus, the RNS [1, 2] (Residue Number System) is gaining relevance to solve the disadvantages of the carry propagation delay in binary adders and multipliers, so a number of tasks performed in the main DSP computation procedures can be accelerated through the use of efficient RNS arithmetic units improved for the MAC (Multiply and Accumulate) operation [3, 4]. In the RNS, in contrast to binary 2's complement weighted arithmetic, there is no

performance degradation with increasing word-width. Arbitrary word-width RNS adders and multipliers maintain a constant throughput and do not suffer from carry propagation delay drawbacks. The RNS is a non-weighted and limited carry arithmetic numeric system whose parallelism between high throughput small word-length channels enables the design of high performance DSP systems.

An RNS implementation of an arithmetic-intensive application leads to several copies of the original system performing modular arithmetic and working over small dynamic ranges but globally handling the total one. Thus, it is possible to consider the mapping of RNS systems into FPL devices. However, the LUT (Look-Up Table) requirements for some RNS operations have been a serious obstacle for the development of such structures. The new programmable device families, such as Altera FLEX10K or APEX20K [5], Virtex [6] and Actel ProASIC [7], have overcome this problem by means of small built-in embedded memories and logic blocks with fast carry and cascade chains, thus enabling these devices for RNS arithmetic.

In this paper, the advantages of the use of RNS rather than FPL devices to implement arithmetic-intensive DSP applications is analyzed. To illustrate this, novel RNS-based architectures for multi-octave orthogonal DWT (Discrete Wavelet Transform) are implemented using structural VHDL synthesis. Area analysis and performance simulations are conducted leading to relevant throughput improvements compared to the equivalent binary 2's complement architectures.

2 Residue Number System Arithmetic

An RNS is defined by a set of L positive and pairwise relatively prime integers $\{m_1, m_2, \ldots, m_L\}$, called moduli. Let $M = \prod_{i=1}^{L} m_i$; thus, the ring of integers modulo M, $Z(M)$, represents the dynamic range of the RNS. Thus, any positive integer $X \in Z(M)$ is uniquely represented by the L-tuple $[x_1, x_2, \ldots, x_L]$ of its residues, where $x_i = X \bmod m_i$. Given two integers X, $Y \in Z(M)$, and their corresponding residue representations $[x_1, x_2, \ldots, x_L]$ and $[y_1, y_2, \ldots, y_L]$, arithmetic in the RNS is defined by:

$$X \circ Y \leftrightarrow \left\lfloor \left|x_1 \circ y_1\right|_{m_1}, \left|x_2 \circ y_2\right|_{m_2}, \ldots, \left|x_L \circ y_L\right|_{m_L} \right\rfloor \qquad (1)$$

where $\circ$ represents either addition, subtraction or multiplication. In this way, RNS arithmetic is defined over the direct sum of integer rings modulo m_i, $Z(m_i)$, and it is performed in parallel without dependencies between the residue digits.

Conversion from the residue representation $[x_1, x_2, \ldots, x_L]$ to its integer value X is based on the Chinese Remainder Theorem (CRT). The theorem states that there exists a unique $X \in Z(M)$ for a given residue representation and that it can be computed as:

$$\left|X\right|_M = \left| \sum_{j=1}^{L} \hat{m}_j \left| \frac{x_j}{\hat{m}_j} \right|_{m_j} \right|_M \qquad \hat{m}_j \equiv \frac{M}{m_j} \ . \qquad (2)$$

Thus, the RNS provides a fundamental methodology for the partitioning of large dynamic range systems into a number of smaller but independent channels over which computations are performed in parallel.

3 FPL Implementation of RNS-Based DSP Algorithms

To date, RNS design methodologies have been most successfully applied for Finite Impulse Response (FIR) filters, which can take full advantage of fast RNS arithmetic while avoiding the problems associated with scaling. Thus, those DSP algorithms that rely upon or can be formulated as convolution are the main target of this numeric system.

Given an input sequence, $x(n)$, a FIR filter is characterized by the convolution operation:

$$y(n) = \sum_{k=0}^{N-1} h_k x(n-k) \quad , \tag{3}$$

where $y(n)$ is the filter output, the h_k terms are the filter coefficients and N denotes the length of the filter. When (3) is encoded in RNS arithmetic, the equations to be computed are described by:

$$\left|y(n)\right|_{m_j} = \left|\sum_{k=0}^{N-1} h_k^j x_j(n-k)\right|_{m_j} \qquad j = 1, ..., L \tag{4}$$

where h_k^j and $x_j(n\text{-}k)$ denote the residues of h_k and $x(n\text{-}k)$ modulo m_j, respectively and all operations and multiplications in (4) are defined as modulo m_j operations. In a digit-parallel design, each modulo m_j processor in (4) for j= 1, 2, …, L is implemented independently and the ensemble is operated in parallel for maximum performance. Thus, each modular FIR filter requires N multipliers and N-1 adders. Moreover, applications requiring a high number of taps can benefit from the use of DA (Distributed Arithmetic) in the RNS domain [3] to reduce memory use in parallel filter implementations.

3.1 Implementation of RNS Arithmetic Modules on FPL Devices

FPL devices providing embedded LUTs and dedicated logic blocks are potential solutions for this kind of RNS-based MAC-intensive algorithms. These devices consist of LEs (Logic Elements) and LUTs. Depending on the family, each LE includes one or more variable input size LUTs (typical are $2^6\times1$, $2^5\times1$, $2^4\times1$ or $2^3\times1$), fast carry propagation logic and one or more flip-flops. Likewise, LUTs allow us to build specialized memory functions such as ROM or RAM.

Specifically, each LE included in the Altera FLEX10K [5] device consists of a $2^4\times1$ LUT, an output register and dedicated logic for fast carry and cascade chains in arithmetic mode. A number of Embedded Array Blocks (EABs), providing a 2K-bit

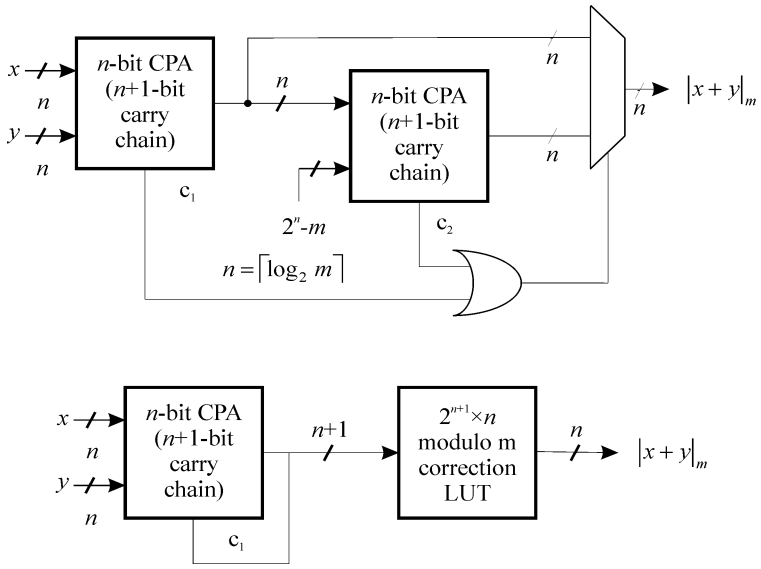

Fig. 1. FPL implementation of residue adders.

RAM or ROM and configurable as $2^8\times8$, $2^9\times4$, $2^{10}\times2$ or $2^{11}\times1$, are the cores for the implementation of RNS LUT-based multipliers. Other FPL device families supporting RNS arithmetic are Altera APEX20K, Actel ProASIC and Virtex. These devices allow the implementation of high throughput modular arithmetic DSP processors, so RNS-FPL merged solutions can achieve the demanding high-precision and high-bandwidth requirements of modern DSP technology.

Several alternatives exist for the implementation of modular adders [8]. Specially oriented for FPLs and only using LEs is the adder that generates the two possible results and selects the correct one according to the carries. Another alternative is to add the input operands and then implement modulo reduction with a LUT. Fig. 1 shows FPL designs of both alternatives. The first of them does not use memory resources and takes full advantage of short carry chains. At the same time, registers provided by LEs make that pipelining does not require additional resources. Table 1 shows area usage and performance for the two adder designs using grade –4 FLEX10K devices.

Table 1. Performance given in MSPS (MegaSamples Per Second) and LEs and EABs required by 6-, 7- and 8-bit residue adders using grade –4 speed FLEX10K devices.

	6-bit moduli			**7-bit moduli**			**8-bit moduli**		
	LEs	**EABs**	**MSPS**	**LEs**	**EABs**	**MSPS**	**LEs**	**EABs**	**MSPS**
Two stage CPA	15	0	40.00	18	0	42.73	19	0	34.36
Two stage CPA (pipelined)	16	0	71.42	18	0	58.47	20	0	57.47
LUT-based	7	1	30.39	8	1	30.03	9	2	28.65
LUT-based (pipelined)	7	1	76.33	8	1	76.33	9	2	76.33

On the other hand, the throughput increase of LUT-based modulo multiplication over binary multiplication is even better than that of addition since a small synchronous EAB can be clocked at frequencies up to 70 MHz for a grade –4 speed FLEX10K device and is not prejudiced by the increasing precision of modern DSP technology.

4 FPL Implementation of RNS-Based DWT Filter Banks

In the last few years, many new transform techniques have arisen that are specifically oriented towards image coding. These techniques frequently appear as multiresolution analysis, time-frequency analysis, pyramid algorithms and wavelet transforms [9, 10]. They offer better compression ratios than DCT-based coding techniques and do not suffer from blocking artifact. Thus, different VLSI architectures have been reported [11, 12] for the DWT, ranging from parallel filter implementations to bit-serial, systolic arrays, distributed memory, etc. A novel FPL implementation of one-octave and two-octave RNS decomposition and reconstruction filter banks for the 1-D DWT and its inverse is shown to illustrate the FPL-RNS synergy described above.

4. 1 Discrete Wavelet Transform

A signal can be expressed as a series expansion using a set of basis functions generated from a basic wavelet function by dilations and translations. The wavelet series expansion represents signals with a sequence of wavelet approximation and detail coefficients corresponding to the sampling in the time-scale space. The successive discrete approximation sequences are lower and lower resolution versions of the original, each sampled twice as sparsely as its predecessor. The DWT can be implemented by a tree-structured algorithm in the multiresolution analysis framework. Thus, the level J 1-D DWT decomposition of a sequence $x(n)$ is defined by the recurrent equations:

$$\begin{aligned} a_n^{(i)} &= \sum_{k=0}^{M_L-1} g_k a_{2n-k}^{(i-1)} & i &= 1, 2, ..., J \\ d_n^{(i)} &= \sum_{k=0}^{M_H-1} h_k a_{2n-k}^{(i-1)} & a_n^{(0)} &\equiv x(n) \end{aligned} \tag{5}$$

where the sequences $a_n^{(i)}$ and $d_n^{(i)}$ are the approximation and detail sequences at level i, and the g_k and h_k coefficients identify the wavelet family used. These equations correspond to an identical structure which consists of a pair of convolvers and decimators that is repeated at each decomposition level. The signal $x(n)$ can be perfectly recovered through its multiresolution decomposition $\{ a_n^{(J)}, d_n^{(J)}, d_n^{(J-1)}, \ldots, d_n^{(1)} \}$.

While there are different kinds of wavelet families, most applications make use almost exclusively of orthogonal wavelets. The perfect reconstruction property and the orthogonality conditions imposed on the wavelets mean that the high-pass decomposition and reconstruction filter coefficients h_k and $\overline{h}_k$ can be obtained from the low-pass decomposition and reconstruction filters g_k and $\overline{g}_k$, respectively:

$$\begin{aligned} h_k &= (-1)^{k+1} g_{N-k-1} \\ \overline{h}_k &= (-1)^{k+1} \overline{g}_{N-k-1} \end{aligned} \qquad k = 0, 1, ..., N-1 \tag{6}$$

where N is the number of taps of the analysis and synthesis FIR filters. Thus, orthogonality condition given in (4) can be exploited to halve the number of multipliers and adders in each octave of the analysis and synthesis multiresolution wavelet filters.

4. 2 RNS Architectures for the Orthogonal Discrete Wavelet Transform

By substituting the orthogonality condition (4) in (3), the octave-i detail sequence $d_n^{(i)}$ can be expressed as:

$$d_n^{(i)} = \sum_{k=0}^{N-1} (-1)^{k+1} g_{N-k-1} a_{2n-k}^{(i-1)} = \sum_{k=0}^{N/2-1} g_{N-2k-2} a_{2n-2k-1}^{(i-1)} - \sum_{k=0}^{N/2-1} g_{N-2k-1} a_{2n-2k}^{(i-1)} \ . \tag{7}$$

Thus, the octave-i approximation $a_n^{(i)}$ and detail $d_n^{(i)}$ sequences can be computed sharing the N multipliers of the low-pass filter coefficients $\{g_0, g_1, ..., g_{N-1}\}$ in alternate cycles in accordance with:

$$\begin{aligned} a_n^{(i)} &= \sum_{k=0}^{N-1} g_k a_{2n-k}^{(i-1)} && i = 1, 2, ..., J \\ d_n^{(i)} &= \sum_{k=0}^{N/2-1} g_{N-2k-2} a_{2n-2k-1}^{(i-1)} - \sum_{k=0}^{N/2-1} g_{N-2k-1} a_{2n-2k}^{(i-1)} && a_n^{(0)} \equiv x(n) \end{aligned} \tag{8}$$

The same idea can be applied for the computation of the 1-D DWT in the RNS. By taking modulo m_j, (6) can be expressed as:

$$\begin{aligned} \left| a_n^{(i)} \right|_{m_j} &= \left| \sum_{k=0}^{N-1} \left| g_k a_{2n-k}^{(i-1)} \right|_{m_j} \right|_{m_j} && i = 1, 2, ..., J \\ \left| d_n^{(i)} \right|_{m_j} &= \left| \left| \sum_{k=0}^{N/2-1} \left| g_{N-2k-2} a_{2n-2k-1}^{(i-1)} \right|_{m_j} \right|_{m_j} - \left| \sum_{k=0}^{N/2-1} \left| g_{N-2k-1} a_{2n-2k}^{(i-1)} \right|_{m_j} \right|_{m_j} \right|_{m_j} \end{aligned} \tag{9}$$

which allows us to synthesize an architecture that halves the number of LUT-based (Look-Up Table) modular multipliers of each octave channel. In this way, an octave-i modular decomposition level consists of N shared LUTs and two modular adder trees

that compute the approximation and detail sequences. The LUTs are clocked at the input sample frequency and are used in alternate clock cycles to compute the products of the low-pass and high-pass filters. These products are added by two modular adder trees controlled by half of the input frequency out-of-phase clocks *clock1* and *clock2*. The resulting 1-D DWT RNS architecture for 8-tap decomposition filters is shown in Fig. 2.

In a similar way, the reconstruction algorithm involved in the IDWT (Inverse Discrete Wavelet Transform) can benefit from the relations between the synthesis filter coefficients $\overline{h}_k$ and $\overline{g}_k$ given in (4) to share the LUTs of only one filter. Each of the reconstruction stages consists of a pair of interpolators and convolvers that compute:

$$\hat{a}_m^{(i-1)} = \begin{cases} \sum_{k=0}^{N/2-1} \overline{h}_{2k+1} \hat{d}_{\frac{m}{2}-k}^{(i)} + \sum_{k=0}^{N/2-1} \overline{g}_{2k+1} \hat{a}_{\frac{m}{2}-k}^{(i)} & m \text{ even} \\ \sum_{k=0}^{N/2-1} \overline{h}_{2k} \hat{d}_{\frac{m-1}{2}-k}^{(i)} + \sum_{k=0}^{N/2-1} \overline{g}_{2k} \hat{a}_{\frac{m-1}{2}-k}^{(i)} & m \text{ odd} \end{cases} . \quad (10)$$

Introducing relation (4) and taking modulo m_j, (8) can be expressed as:

$$\left| \hat{a}_m^{(i-1)} \right|_{m_j} = \begin{cases} \left| \left| \sum_{k=0}^{N/2-1} \left| \overline{g}_{N-2k-2} \hat{d}_{\frac{m}{2}-k}^{(i)} \right|_{m_j} \right|_{m_j} + \left| \sum_{k=0}^{N/2-1} \left| \overline{g}_{2k+1} \hat{a}_{\frac{m}{2}-k}^{(i)} \right|_{m_j} \right|_{m_j} \right|_{m_j} & m \text{ even} \\ \left| \left| \sum_{k=0}^{N/2-1} \left| \overline{g}_{2k} \hat{a}_{\frac{m-1}{2}-k}^{(i)} \right|_{m_j} \right|_{m_j} - \left| \sum_{k=0}^{N/2-1} \left| \overline{g}_{N-2k-1} \hat{d}_{\frac{m-1}{2}-k}^{(i)} \right|_{m_j} \right|_{m_j} \right|_{m_j} & m \text{ odd} \end{cases} \quad (11)$$

that leads to an architecture for the orthogonal 1-D IDWT in which the LUTs of the low-pass synthesis filter are shared to generate the low-pass and high-pass filter outputs as shown in Fig. 3. In this way, the N LUTs operate at the input rate producing $N/2$ partial products of each filter which are added by two modular adder trees clocked by out-of-phase clocks *clock1* and *clock2* at half of the sampling rate.

The proposed RNS VLSI architectures for the 1-D DWT and IDWT implement the analysis and synthesis filters respectively, halving the number of LUTs due to the orthogonality relations between the filter coefficients. With this scheme, the decimation and interpolation processes involved in the DWT and IDWT allow us to compute the two half of the input frequency outputs $\left| a_n^{(i)} \right|_{m_j}$ and $\left| d_n^{(i)} \right|_{m_j}$ and the twice the input frequency output $\left| a_n^{(i-1)} \right|_{m_j}$ with only half of the memory resources. An additional resource requirement reduction can be obtained if only one modular adder tree clocked at the input frequency is used. However, this alternative reduces the performance of the system since the modular adder tree must operate at the input frequency.

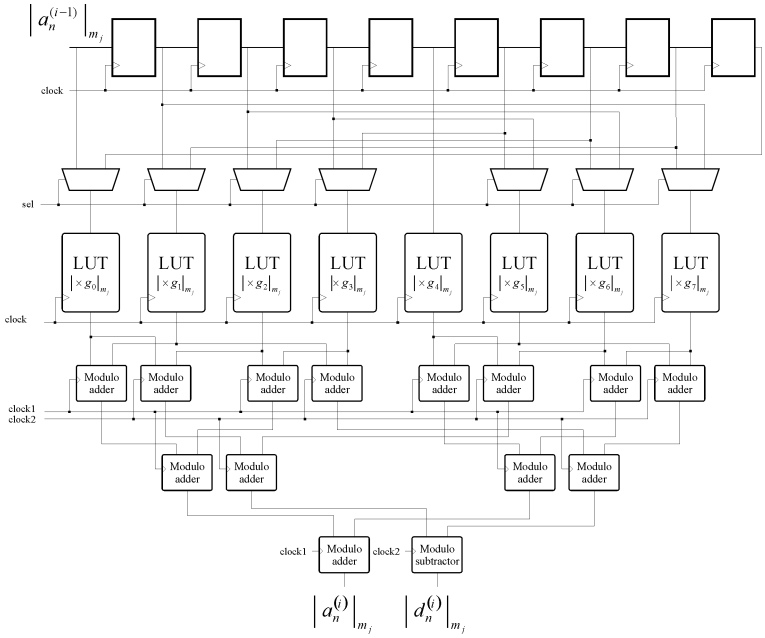

Fig. 2. Modulo m_j RNS architecture for the octave-*i* decomposition filter bank.

4. 3 Implementation over FPL Devices. Area Analysis and Performance Comparison

Two's complement arithmetic and RNS versions of the proposed 1-D DWT and IDWT architectures were implemented over FPL devices to compare hardware complexity and performance. Altera FLEX10K [5] devices were used.

One-level and two-level decomposition and reconstruction architectures were synthesized using VHDL to quantify parameters such as area and performance. 8-bit input samples and 10-bit filter coefficients were assumed, so that one octave and two octaves required 21- and 34-bit dynamic ranges, respectively. Table II shows the results obtained for each 6-, 7- and 8-bit RNS channel and for traditional arithmetic 1-D DWT and 1-D IDWT implementations over grade –4 speed FLEX10K devices. Hardware requirements were assessed in terms of the number of LEs and EABs while performance was evaluated in terms of the register-to-register path maximum delay. The performance advantage of the 6-, 7- and 8-bit RNS-based architectures proposed increased with the dynamic range considered. In this way, the performance is up to 23.45% and 96.58% better for one octave and two octaves, respectively, when compared to equally pipelined traditional arithmetic implementations.

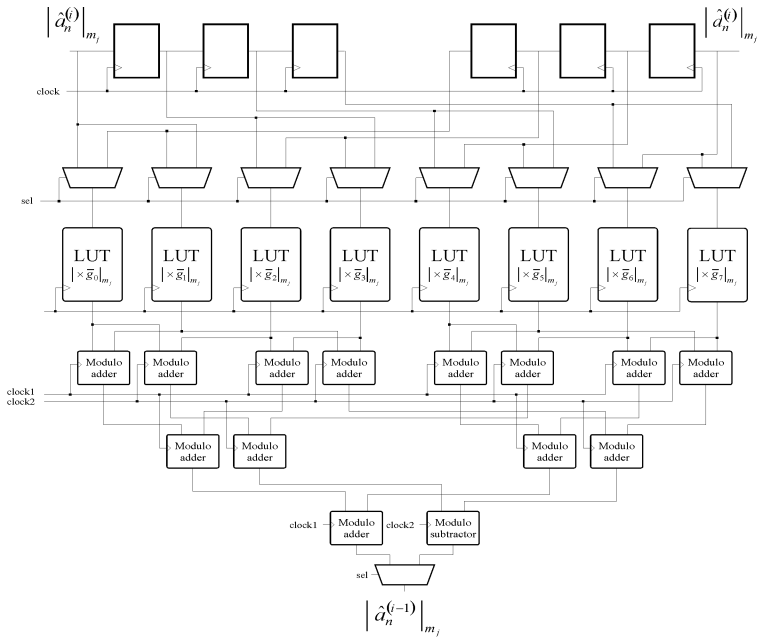

Fig. 3. Modulo m_j RNS architecture for the octave-i reconstruction filter bank.

5 Conclusion

The methodology to design RNS-enabled DSP architectures over FPL devices was assessed in this paper. The synergy between the RNS and modern FPL device families providing embedded LUTs can overcome the demanding high-precision and high-throughput obstacle of the traditional binary 2's complement arithmetic for modern real-time DSP applications. FIR filtering is the main target of this study since it can take full advantage of the RNS while avoiding the problems associated with scaling. When FPL devices are used for such systems, modular adders can benefit from the short carry chains and pipelining supported by LEs while modulo multipliers yield an even more throughput increase over binary multipliers since they are based in small synchronous LUTs. The design methodologies are illustrated by synthesizing RNS architectures for the multi-octave orthogonal DWT and its inverse. These structures are highly flexible concerning the length of the filters and their performance is independent of the precision of the input samples and filter coefficients. The relation between the low-pass and high-pass filter coefficients allows us to halve the number of LUTs in each octave. VHDL and Altera FLEX10K devices were used to model the systems at structural level and to assess hardware requirements and throughput. The performance advantage achieved by the RNS-FPL merged solutions proposed is 23.45% and 96.58% for one octave and two octave transforms, respectively.

Table 2. FPL implementation of RNS and 2's complement 1-D DWT and 1-D IDWT.

	1-D DWT		1-D IDWT		Throughput (MSPS)
	LEs	EABs (Memory bits)	LEs	EABs (Memory bits)	
6-bit channel	316	8 (8×6×64)	321	8 (8×6×64)	68.96
7-bit channel	362	8 (8×7×128)	367	8 (8×7×128)	67.11
8-bit channel	403	8 (8×8×256)	409	8 (8×8×256)	65.78
One-octave 21-bit (Binary)	1164	-	1173	-	55.86
Two-octave 34-bit (Binary)	3689	-	3705	-	35.08

Acknowledgements

The authors were supported by the Dirección General de Enseñanza Superior (Spain) under project PB98-1354. CAD tools and supporting material were provided by Altera Corp. under the Altera University Program.

References

1. M. Soderstrand, W. Jenkins, G. A. Jullien, and F. J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. IEEE Press Reprint Series. IEEE Press, 1986.
2. N. Szabo and R. Tanaka, *Residue Arithmetic and its Applications to Computer Technology*. McGraw-Hill, 1967.
3. A. García, U. Meyer-Bäse, A. Lloris and F. J. Taylor, "RNS Implementation of FIR Filters Based on Distributed Arithmetic using Field-Programmable Logic", *Proc. IEEE Int. Symp. on Circuits and Systems*, Orlando, FL, vol. 1, pp. 486-489, Jun. 1999.
4. J. Ramírez, A. García, P. G. Fernández, L. Parrilla and A. Lloris, "A New Architecture to Compute the Discrete Cosine Transform Using the Quadratic Residue Number System," in *2000 IEEE Int. Symp. on Circuits and Systems*, Geneva, Switzerland, May 2000.
5. Altera Corporation, *1998 Data Book*, Jan. 1998.
6. Xilinx Inc., *The Programmable Logic Data Book*, 1999.
7. Actel Corporation, *ProASIC 500K Family Data Sheet,* 2000.
8. M. A. Bayoumi, G. A. Jullien and W. C. Miller, "A VLSI Implementation of Residue Adders." *IEEE Trans. on Circuits and systems*, vol. 34, no. 3, pp. 284-288, Mar. 87.
9. G. Strang, T. Nguyen, *Wavelets and Filter Banks*, Wellesly-Cambridge Press, 1997.
10. M. Vetterli, J. Kovacevik, *Wavelets and Subband Coding*, Prentice Hall, 1995
11. C. Chakrabarti, C. Mumford, "Efficient Realizations of Encoders and Decoders Based on the 2-D Discrete Wavelet Transform," *IEEE Trans. VLSI Syst*., vol. 2, no. 3, Sep. 1999.
12. M. Vishwanath, M. Owens, M. J. Irwin, "VLSI Architectures for the Discrete Wavelet Transform," *IEEE Trans. Circuits Syst. II*, vol. 42, no. 5, May 1995.