Journal of VLSI Signal Processing 33, 171–190, 2003

Implementation of RNS-Based Distributed Arithmetic Discrete Wavelet Transform Architectures Using Field-Programmable Logic

JAVIER RAMÍREZ AND ANTONIO GARCÍA
Departamento de Electrónica y Tecnología de Computadores, Campus Universitario Fuentenueva, University of Granada, 18071 Granada, Spain

UWE MEYER-BÄSE
Department of Electrical and Computer Engineering, FAMU-FSU College of Engineering, Tallahassee, FL 32310-6046, USA

FRED TAYLOR
High-Speed Digital Architecture Laboratory, Electrical and Computer Engineering, Computer and Information Science Engineering, University of Florida, Gainesville FL, 32611-6130, USA

ANTONIO LLORIS
Departamento de Electrónica y Tecnología de Computadores, Campus Universitario Fuentenueva, University of Granada, 18071 Granada, Spain

Received October 27, 2000; Revised July 12, 2001

Abstract. Currently there are design barriers inhibiting the implementation of high-precision digital signal processing (DSP) objects with field programmable logic (FPL) devices. This paper explores overcoming these barriers by fusing together the popular distributed arithmetic (DA) method with the residue number system (RNS) for use in FPL-centric designs. The new design paradigm is studied in the context of a high-performance filter bank and a discrete wavelet transform (DWT). The proposed design paradigm is facilitated by a new RNS accumulator structure based on a carry save adder (CSA). The reported methodology also introduces a polyphase filter structure that results in a reduced look-up table (LUT) budget. The 2C-DA and RNS-DA are compared, in the context of a FPL implementation strategy, using a discrete wavelet transform (DWT) filter bank as a common design theme. The results show that the RNS-DA, compared to a traditional 2C-DA design, enjoys a performance advantage that increases with precision (wordlength).

Keywords: field-programmable logic, residue number system, distributed arithmetic, discrete wavelet transform, digital signal processing

1. Introduction

Digital signal processing (DSP) is arithmetic-intensive. DSP-facilitating technologies include general-purpose microprocessors, application-specific integrated circuits (ASIC), application specific standard products (ASSP), and field programmable logic (FPL) devices that include field programmable gate arrays (FPGA). Within this mix, ASIC are becoming the dominant technology with the Y-2000 DSP CBIC (cell-based integrated circuit) ASIC market valued in excess of \$13B, compared to \$8B for DSPs. The FPL ASIC market is expected to expand at a rate of 20% *per annum* rate, with DSP applications leading the way. While FPL houses champion their technology as a provider of system-on-a-chip (SOC) DSP solutions, engineers have historically viewed FPLs as a prototyping technology. It should be noted that 40% of the current FPL design starts are rated at 1,500 gates. This figure falls well below the reported 50,000+ gates that account for 50% of standard cell ASIC designs [1]. When one considers that an FPGA typically requires 10× more gates than a CBIC to implement a common logic function, a typical 50 k gate standard cell ASIC design would require a large 500 k gate FPGA. In order for FPL to begin to compete in areas currently controlled by low-end standard cell, a means must be found to more efficiently implement DSP objects.

A review of FPL vendor supplied application notes, establishes that FPGAs have intrinsically weak arithmetic capabilities. A general-purpose $n \times n$-bit multiplier or multiply-accumulate (MAC) unit, for example, is inferior to a well designed ASIC ALU in both speed and area [2]. In addition, FPL deficiencies increase geometrically with precision (wordlength). It is the FPGA's arithmetic limitations that have caused solution developers to consider alternative structures. The most popular technique, found in common practice, is called distributed arithmetic [3–6], or DA. The DA method is routinely used to implement linear DSP algorithms with fixed known a priori coefficients. The DA technique reduces an algorithm to a set of sequential lookup table (LUT) calls, and two's-complement (2C) shift-adds. A survey of the contemporary FPL DA art indicates that most solutions are of low-order and low-precision. The reported FPL limitations are the result of architectural limitation. FPL device families, such as Altera FLEX10K [7] or XILINX Virtex [8], are organized in channels (typically 8-bits wide). Within these channels are found short delay propagation paths, and dedicated memory blocks, with programmable address and data spaces that are commonly used to synthesize small RAM and ROM functions. Performance rapidly suffers when carry bits and/or data have to propagate across a channel boundary. We call this the *channel barrier* problem. Existing 2C-DA designs encounter the channel barrier problem whenever wordwidths exceed the channel width in bits. An alternative design paradigm is advocated in this paper that is based on fusion of DA and the residue number system or RNS [9–11]. The RNS advantage is gained by reducing arithmetic to a set of concurrent operations that reside in small wordlength non-communicating channels. This attribute makes the RNS potentially attractive for implementing DSP objects with a FPL. Specifically, the paper develops a mechanism of achieving synergy within an FPL-defined environment for implement arithmetic intensive DSP solutions. The quantifiable benefits of this approach are studied in the context of a design example, namely a discrete wavelet transform (DWT) filterbank using an Altera FLEX10K FPL reference device. The work will build upon previous works [12–15] and RNS-FPL design studies [16–19].

2. Background

There is emerging evidence that an arithmetic technology, called the RNS, can overcome the channel barrier and become a FPL enabling technology [20–22]. Computer arithmeticians have long held that the RNS offers the best MAC speed-area advantage [9]. In the RNS, numbers are represented in terms of a relatively prime basis set (moduli set) $P = \{m_1, \dots m_L\}$. Any number $X \in Z_M = \{0, \dots, M-1\}$, where $M = \Pi m_i$, has a unique RNS representation $X \leftrightarrow [X_{m_1}, \dots, X_{m_L}]$, with $X_{m_i} = X \mathrm{mod}(m_i)$. Like the 2C system, the RNS arithmetic is exact as long as the final result is bounded within the system's dynamic range M. Mapping from the RNS back to the integer domain is defined by the Chinese Remainder Theorem (CRT) [9, 10]. RNS arithmetic is defined by pair-wise modular operations:

$$
\begin{aligned}
Z = X \pm Y &\leftrightarrow \left[\left|X_{m_1} \pm Y_{m_1}\right|_{m_1}, \dots, \left|X_{m_L} \pm Y_{m_L}\right|_{m_L}\right] \\
Z = X \times Y &\leftrightarrow \left[\left|X_{m_1} \times Y_{m_1}\right|_{m_1}, \dots, \left|X_{m_L} \times Y_{m_L}\right|_{m_L}\right]
\end{aligned} \tag{1}
$$

where $|Q|_{m_j}$ denotes $Q \bmod(m_j)$. The individual modular arithmetic operations are typically performed as LUT calls to small memories. The RNS differs from traditional weighted numbering systems in that the RNS arithmetic is a *carry-free* and can operate at a constant speed over a wide range of wordwidths (precision). It is the ability of the RNS to do arithmetic within independent small wordlength channels that makes it particularly attractive for FPL insertion. The

RNS has been studied in an FPL context by Hamann and Sprachmann [12] who explored implementing a simple DSP solution using Xilinx parts. Jullien [13] combined number theory with FPLs to design a FIR. Fermat primes were used as the basis elements and a Xilinx FPGA used to implement index arithmetic ALU. Performance is strongly influenced by the choice of basis functions (moduli) that were limited by the size of the FPL LUT address space, and channel width. A demonstration of the RNS as an enabling FPL technology is the communication filter reported in [20, 21] using an Altera FLEX10K part. The digital filters were designed to accept 8-bit inputs and 10-bit coefficients, with an internal 24-bit dynamic range, and ran at 49.3 MHz in 2C and 76.6 MHz in the RNS. Complexity was comparable to a 2C design. These preliminary works provide the motivation to advance this work in a rigorous and comprehensive manner.

3. Discrete Wavelet Transform

Interest in the wavelet transform [23, 24] has grown dramatically during the last decade [25–30]. Wavelet transforms are routinely used in speech, image and video signal processing, and other applications. Discrete wavelet transforms (DWT) are defined over a sequence of embedded closed subspaces, $V_J \subset V_{J-1} \subset \cdots \subset V_1 \subset V_0$, where $V_0 = l_2(Z)$ is the space of square-summable sequences. These subspaces satisfy the upward completeness property, $\cup V_j = l_2(Z)$, $j \in [0, J]$. Assume that any element in V_j can be uniquely expressed as the sum of two elements from V_{j+1} and W_{j+1}, where $V_j = V_{j+1} \oplus W_{j+1}$. For orthogonal wavelets, W_{j+1} is defined as the orthogonal complement of V_{j+1} in V_j. Assuming a sequence $\bar{g}_n \in V_0$ exists such that $\{\bar{g}_{n-2k}\}_{k\in Z}$ is a basis for V_1, a sequence $\bar{h}_n \in V_0$ can then be found such that $\{\bar{h}_{n-2k}\}_{k\in Z}$ is a basis for W_1. Thus, V_0 can be decomposed as: $V_0 = W_1 \oplus W_2 \oplus \cdots \oplus W_J \oplus V_J$ by simply iterating the decomposition rule J times. An attractive feature of the wavelet series expansion is that the underlying multiresolution structure leads to an efficient discrete-time algorithm based on a filter bank implementation. The octave-band analysis filter bank computes the inner products with the basis functions for $W_1, W_2, \ldots, W_J$, and V_J. The orthogonal projection of the input signal onto $W_1, W_2, \ldots, W_J$, and V_J is computed after convolution with the synthesis filters. Then, the sequence is decomposed into a coarse resolution version in V_J with added details in W_i ($i = 1, 2, \ldots, J$). Thus a 1-D Nth-order DWT decomposition of a sequence x_n is defined by the recurrent equations:

$$\begin{aligned} a_n^{(i)} &= \sum_{k=0}^{N-1} g_k a_{2n-k}^{(i-1)} \quad i = 1, 2, \ldots, J \\ d_n^{(i)} &= \sum_{k=0}^{N-1} h_k a_{2n-k}^{(i-1)} \quad a_n^{(0)} \equiv x_n \end{aligned} \tag{2}$$

where $a_n^{(i)}$ and $d_n^{(i)}$ are level-i approximation and detail sequences, respectively, and g_k and h_k ($k = 0, 1, \ldots, N-1$) correspond to the low-pass and high-pass analysis filter coefficients. On the other hand, the signal x_n can be perfectly recovered through its multiresolution decomposition $\{a_n^{(J)}, d_n^{(J)}, d_n^{(J-1)}, \ldots, d_n^{(1)}\}$ by iteration on:

$$\hat{a}_m^{(i-1)} = \begin{cases} \displaystyle\sum_{k=0}^{N/2-1} \bar{g}_{2k} \hat{a}_{\frac{m}{2}-k}^{(i)} + \sum_{k=0}^{N/2-1} \bar{h}_{2k} \hat{d}_{\frac{m}{2}-k}^{(i)} & m \text{ even} \\ \displaystyle\sum_{k=0}^{N/2-1} \bar{g}_{2k+1} \hat{a}_{\frac{m-1}{2}-k}^{(i)} + \sum_{k=0}^{N/2-1} \bar{h}_{2k+1} \hat{d}_{\frac{m-1}{2}-k}^{(i)} & m \text{ odd} \end{cases} \tag{3}$$

where $\bar{g}_k$ and $\bar{h}_k$ represent low-pass and high-pass synthesis filter coefficients. In order to ensure perfect recovery of the input signal, the coefficients of the analysis and synthesis filter banks are conveniently related to each other according to the perfect reconstruction condition [23, 24].

4. 2C-DA Architectures for the DWT

Conventional 2C-DA has been successfully applied to the implementation of FIR filters and other linear discrete transforms. DA designs replace general multiplication with scaling operations implemented using LUT calls. DA architectures are bit-serial and are often reported to be faster than MAC-centric designs. In addition, compared to programmable MAC solutions, DA designs often have a lower roundoff error budget [31]. The DA advantage is amplified in FPL applications, a technology possessing an intrinsically weak MAC capability. A DA FPL design must, however, be mindful of the channel barrier problem since FPLs restrict DA LUTs to reside in an FPL's logic element (LE). For Altera FLEX10K devices, each LE consists of a $2^4 \times 1$ LUT, an output register and dedicated logic for fast carry and cascade chains. The larger embedded array blocks (EABs) found in a FLEX10K device consist of 2K-bit memory blocks configurable as $2^8 \times 8$, $2^9 \times 4$, $2^{10} \times 2$ or $2^{11} \times 1$. The FLEX10KE family now provides 4K-bit EABs organized as $2^8 \times 16$, $2^9 \times 8$, $2^{10} \times 4$ or $2^{11} \times 2$.

A 2C-DA design assumes that the input to a DSP-object (e.g., FIR filter) is the B_i-bit word:

$$a_n^{(i-1)} = -2^{B_i-1} a_{n,B_i-1}^{(i-1)} + \sum_{l=0}^{B_i-2} 2^l a_{n,l}^{(i-1)} \tag{4}$$

where $a_{n,l}^{(i-1)}$ is the l-th bit of the input sample $a_n^{(i-1)}$. For the case where the filters define a 1-D DWT, filter pairs would be defined by (after Eq. (2)):

$$\begin{aligned} a_n^{(i)} &= -2^{B_i-1} \sum_{k=0}^{N-1} g_k a_{2n-k,B_i-1}^{(i-1)} + \sum_{k=0}^{N-1} g_k \sum_{l=0}^{B_i-2} 2^l a_{2n-k,l}^{(i-1)} \\ d_n^{(i)} &= -2^{B_i-1} \sum_{k=0}^{N-1} h_k a_{2n-k,B_i-1}^{(i-1)} + \sum_{k=0}^{N-1} h_k \sum_{l=0}^{B_i-2} 2^l a_{2n-k,l}^{(i-1)} \end{aligned} \tag{5}$$

Define the DA LUT functions, $\Phi_g(l)$ and $\Phi_h(l)$ to be:

$$\Phi_g(l) = \sum_{k=0}^{N-1} g_k a_{2n-k,l}^{(i-1)} \quad \Phi_h(l) = \sum_{k=0}^{N-1} h_k a_{2n-k,l}^{(i-1)} \tag{6}$$

which results in the DA equations:

$$\begin{aligned} a_n^{(i)} &= -2^{B_i-1} \Phi_g(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_g(l) \\ d_n^{(i)} &= -2^{B_i-1} \Phi_h(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_h(l) \end{aligned} \tag{7}$$

The computation of the ith-octave-approximations, $a_n^{(i)}$, and details, $d_n^{(i)}$, $(i = 1, 2, \ldots, J)$ is carried using two $2^N \times W$ LUTs representing the functions $\Phi_g(l)$ and $\Phi_h(l)$, which are addressed by the N-bit vector $\{a_{2n,l}^{(i-1)}, a_{2n-1,l}^{(i-1)}, \ldots, a_{2n-N+1,l}^{(i-1)}\}$, $W \le b + \lceil \log_2(N) \rceil$, and b represents the filter coefficient precision. Observe that computing Eq. (7) requires repeated calls to the tables $\Phi_g(l)$ and $\Phi_h(l)$, followed by a shift-add (scaled accumulation). Figure 1 shows the 2C-DA architecture for the computation of the ith octave filter bank output. The first table look-up is shift subtracted while the following are added to the accumulator. Note that decimation by 2 is carried out efficiently by considering two consecutive input samples, $a_{2n-1}^{(i-1)}$ and $a_{2n}^{(i-1)}$. The sampling clock (sCLK) is generated by dividing the

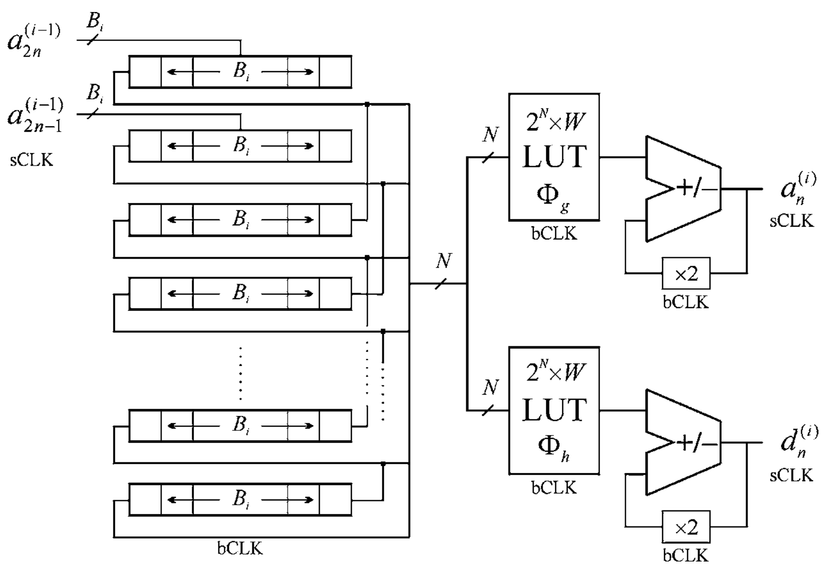

Figure 1. 2C-DA 1-D DWT architecture.

accumulation bit clock (bCLK) by B_i. Two input values are sampled in each sampling clock (sCLK) cycle. Finally, registers are left-shifted, since the MSB (most significant bit) is the first bit processed.

A polyphase filter bank representation can, however, lead to a reduction in the LUT address space. Defining the even- and odd-indexed filters to be $g_0(k) = g_{2k}$, $h_0(k) = h_{2k}$, $g_1(k) = g_{2k+1}$, and $h_1(k) = h_{2k+1}$ ($k = 0, 1, \ldots, N/2 - 1$), a polyphase DA architecture, for the ith-octave analysis and synthesis filter bank consists of four independent DA filters operating on even- and odd-indexed input samples:

$$\begin{aligned} a_n^{(i)} &= \sum_{k=0}^{N/2-1} g_0(k) a_{2n-2k}^{(i-1)} + \sum_{k=0}^{N/2-1} g_1(k) a_{2n-2k-1}^{(i-1)} \\ d_n^{(i)} &= \sum_{k=0}^{N/2-1} h_0(k) a_{2n-2k}^{(i-1)} + \sum_{k=0}^{N/2-1} h_1(k) a_{2n-2k-1}^{(i-1)} \end{aligned} \tag{8}$$

Thus, polyphase filters convolve two distinct sample subsets. The even indexed values of $a^{(i-1)}$ are convolved with $g_0(k)$ and $h_0(k)$ and the odd indexed values are filtered using $g_1(k)$ and $h_1(k)$. In this way, the defining DA relationships are:

$$\begin{aligned} \Phi_{g_0}(l) &= \sum_{k=0}^{N/2-1} g_0(k) a_{2n-2k,l}^{(i-1)} \\ \Phi_{g_1}(l) &= \sum_{k=0}^{N/2-1} g_1(k) a_{2n-2k-1,l}^{(i-1)} \\ \Phi_{h_0}(l) &= \sum_{k=0}^{N/2-1} h_0(k) a_{2n-2k,l}^{(i-1)} \\ \Phi_{h_1}(l) &= \sum_{k=0}^{N/2-1} h_1(k) a_{2n-2k-1,l}^{(i-1)} \end{aligned} \tag{9}$$

and the filter bank outputs become:

$$\begin{aligned} a_n^{(i)} &= -2^{B_i-1}\Phi_{g_0}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{g_0}(l) - 2^{B_i-1}\Phi_{g_1}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{g_1}(l) \\ d_n^{(i)} &= -2^{B_i-1}\Phi_{h_0}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{h_0}(l) - 2^{B_i-1}\Phi_{h_1}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{h_1}(l) \end{aligned} \tag{10}$$

The architecture is shown in Fig. 2 and consists of four $2^{N/2} \times W'$ LUTs, where $W' \leq b' + \lceil \log_2(N/2) \rceil$ and b' represents the filter bank coefficient precision. Notice that it is necessary to introduce a delay in the odd-indexed sequence, and that the two outputs $a_n^{(i)}$ and $d_n^{(i)}$ are computed by adding the low-pass and high-pass polyphase filter outputs respectively.

The 1-D IDWT (inverse DWT) can be also computed through the DA scheme in a manner motivated in the 2C-DA design narrative. By representing the inputs to the ith-octave reconstruction filter bank, for $\bar{B}_i$-bit words, as:

$$\begin{aligned} \hat{a}_n^{(i)} &= -2^{\bar{B}_i-1}\hat{a}_{n,\bar{B}_i-1}^{(i)} + \sum_{l=0}^{\bar{B}_i-2} 2^l \hat{a}_{n,l}^{(i)} \\ \hat{d}_n^{(i)} &= -2^{\bar{B}_i-1}\hat{d}_{n,\bar{B}_i-1}^{(i)} + \sum_{l=0}^{\bar{B}_i-2} 2^l \hat{d}_{n,l}^{(i)} \end{aligned} \tag{11}$$

The IDWT LUTs are defined by:

$$\begin{aligned} \Phi_{\bar{h}}^{\mathrm{e}}(l) &= \sum_{k=0}^{N/2-1} \bar{h}_{2k} \hat{d}_{\frac{m}{2}-k,l}^{(i)} & \Phi_{\bar{h}}^{\mathrm{o}}(l) &= \sum_{k=0}^{N/2-1} \bar{h}_{2k+1} \hat{d}_{\frac{m-1}{2}-k,l}^{(i)} \\ \Phi_{\bar{g}}^{\mathrm{e}}(l) &= \sum_{k=0}^{N/2-1} \bar{g}_{2k} \hat{a}_{\frac{m}{2}-k,l}^{(i)} & \Phi_{\bar{g}}^{\mathrm{o}}(l) &= \sum_{k=0}^{N/2-1} \bar{g}_{2k+1} \hat{a}_{\frac{m-1}{2}-k,l}^{(i)} \end{aligned} \tag{12}$$

and the computation of the DA IDWT is defined to be:

$$\hat{a}_m^{(i-1)} = \begin{cases} -2^{B_i-1}\Phi_{\bar{h}}^{\mathrm{e}}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{\bar{h}}^{\mathrm{e}}(l) - 2^{B_i-1}\Phi_{\bar{g}}^{\mathrm{e}}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{\bar{g}}^{\mathrm{e}}(l) & m \text{ even} \\ -2^{B_i-1}\Phi_{\bar{h}}^{\mathrm{o}}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{\bar{h}}^{\mathrm{o}}(l) - 2^{B_i-1}\Phi_{\bar{g}}^{\mathrm{o}}(B_i - 1) + \sum_{l=0}^{B_i-2} 2^l \Phi_{\bar{g}}^{\mathrm{o}}(l) & m \text{ odd} \end{cases} \tag{13}$$

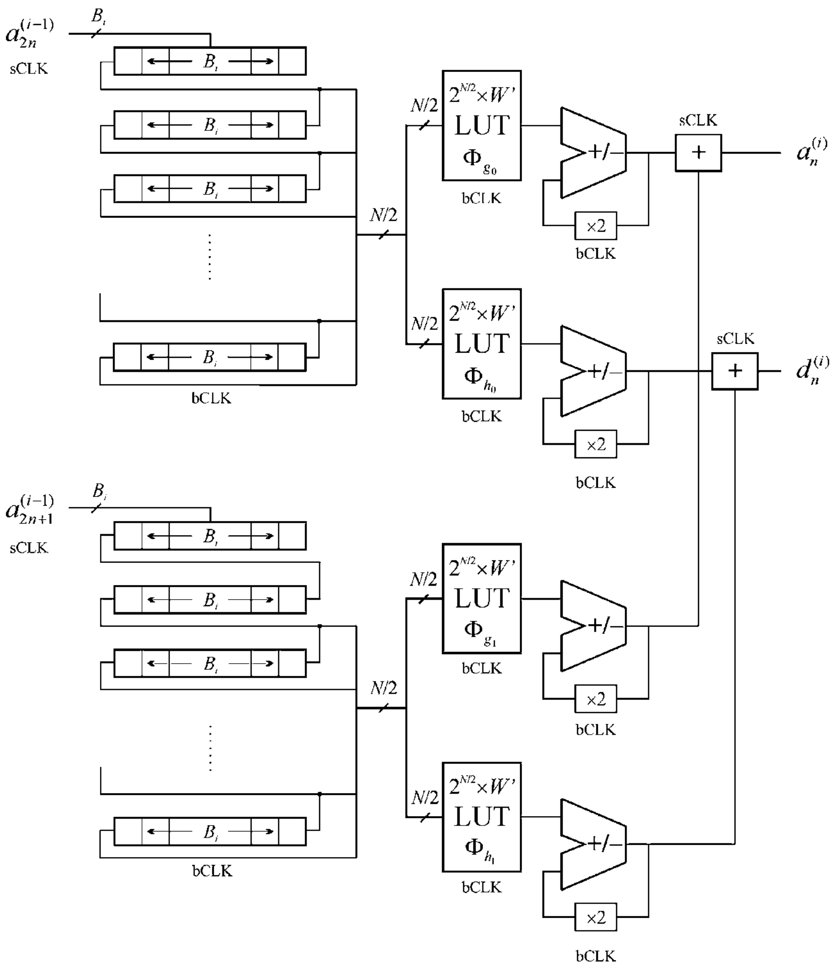

Figure 2. Polyphase 2C-DA 1-D DWT architecture.

The 2C-DA architecture is shown in Fig. 3 in the context of the ith-octave synthesis filter. The low- and high-pass filter outputs are computed according to the DA paradigm that computes the even and odd outputs simultaneously. Two consecutive output samples, $\hat{a}_m^{(i-1)}$ and $\hat{a}_{m+1}^{(i-1)}$, (m even) are computed concurrently over the set of samples $\{\hat{a}_{m/2}^{(i)}, \hat{a}_{m/2-1}^{(i)}, \ldots, \hat{a}_{m/2-N/2+1}^{(i)}, \hat{d}_{m/2}^{(i)}, \hat{d}_{m/2-1}^{(i)}, \ldots, \hat{d}_{m/2-N/2+1}^{(i)}\}$ by accumulating the outputs of four $2^{N/2} \times \bar{W}$ LUTs, where $\bar{W} \leq \bar{b} + \lceil \log_2(N/2) \rceil$, where $\bar{b}$ represents the filter bank coefficient precision. Implementation data for these 2C-DA architectures will be given in Section 6.

5. RNS-DA Architectures for the DWT

In concept, the RNS represents a potentially efficient means of implementing a DA-based FPL DSP solution [11]. Input sequences are encoded and manipulated concurrently within small wordlength channels. An RNS system was defined in terms of a moduli set $P = \{m_1, m_2, \ldots, m_L\}$ as developed in Section 2. Since typically $m_i \leq 2^8$, the solution can be defined to reside within 8-bit channels. A comparable procedure to that shown in Eq. (7) for the 2C-DA mechanization will be developed for the RNS-DA. Thus, for every channel, the current result must be multiplied by 2 prior to accumulating with the output of a LUT. This would require a specific LUT for 2 mod m_j multiplication, which would add unwanted complexity to the recursive path. However, using a scaled modular accumulator computing $|y(n)|_{m_j} = |2y(n-1) + x(n)|_{m_j}$, an RNS-DA solution can be realized, leading to an efficient implementation of inner product based DSP algorithms. This accumulator is designed according to the following selection

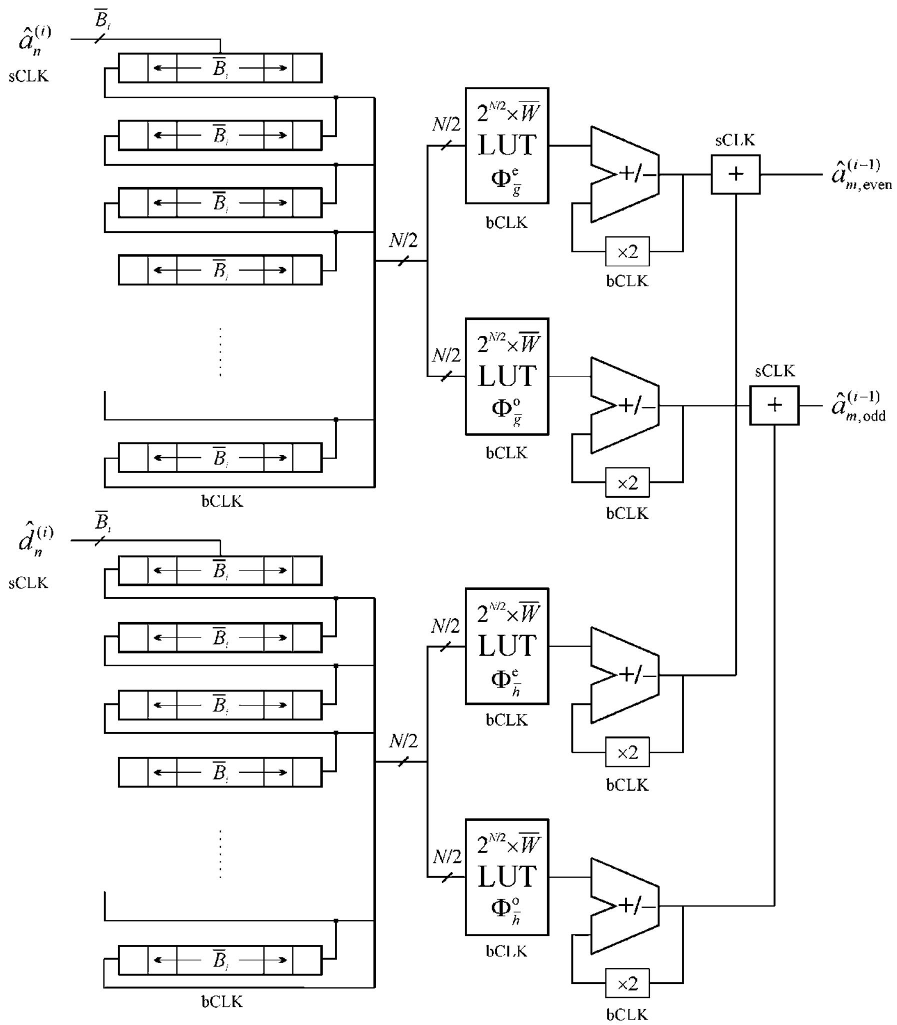

Figure 3. 2C-DA 1-D IDWT architecture.

rules:

$$|y(n)|_{m_j} = \begin{cases} 2|y(n-1)|_{m_j} + |x(n)|_{m_j} & \text{if } 2|y(n-1)|_{m_j} + |x(n)|_{m_j} < m_j \\ 2|y(n-1)|_{m_j} + |x(n)|_{m_j} - m_j & \text{if } m_j \le 2|y(n-1)|_{m_j} + |x(n)|_{m_j} < 2m_j \\ 2|y(n-1)|_{m_j} + |x(n)|_{m_j} - 2m_j & \text{if } 2m_j \le 2|y(n-1)|_{m_j} + |x(n)|_{m_j} < 3m_j \end{cases} \tag{14}$$

The design of a modulo m_j scaling accumulator, for RNS-DA applications, was presented in [11]. An improved architecture can be innovated by using CSAs (carry save adders) to realize the 2nd and 3rd terms in Eq. (14), as shown in Fig. 4. The improved design shortens the carry propagation chain, thus improving the performance of an RNS-DA based system. The new accumulator also uses one CPA (carry propagate adder) to realize the term $2|y(n-1)|_{m_j} + |x(n)|_{m_j}$, with two CSAs used to compute $2|y(n-1)|_{m_j} + |x(n)|_{m_j} - m_j$ and $2|y(n-1)|_{m_j} + |x(n)|_{m_j} - 2m_j$ respectively. These terms are computed concurrently and, as a result, have only one carry propagation stage in the critical path. The final result is selected on the basis of the carries generated in the summation stages as reported in Table 1.

In Table 2, a comparison is made between the improved accumulator and that was originally reported in

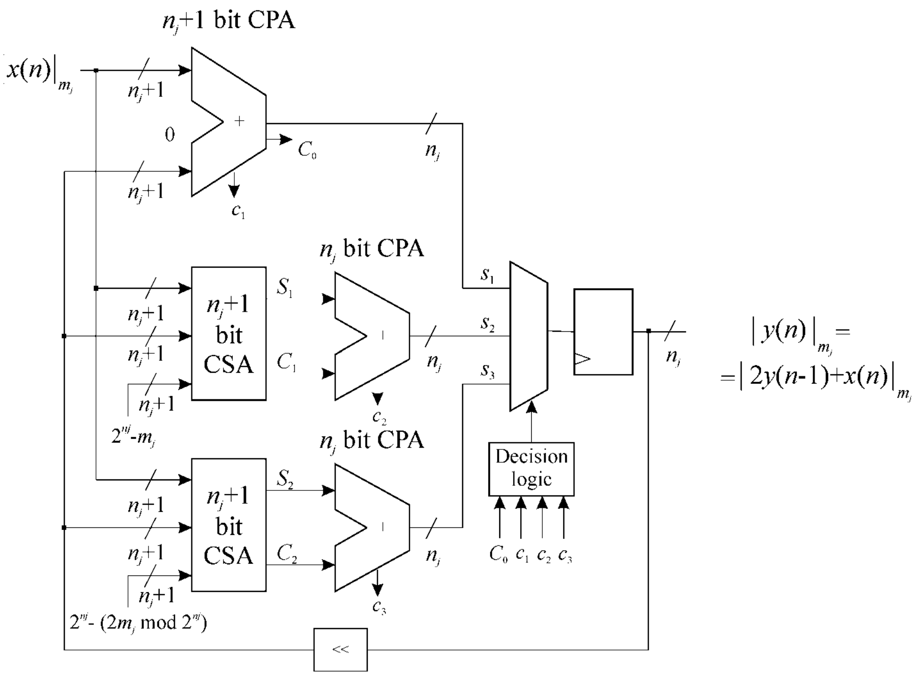

Figure 4. Improved design of the RNS-DA accumulator.

[11], in terms of LE requirements, and speed, for 6-, 7- and 8-bit modulus accumulators. CPAs are synthesized using fast carry chains while each 3-input logic function (required by a CSA) is mapped to a single LE. Since only one CPA stage appears in the signal path, the advantage in performance increases with the modulus wordwidth. Thus, compared to the previous modulo accumulator, the throughput improvement for the new CSA-based accumulator, for 6-, 7- and 8-bit moduli, is 3.13%, 4.02% and 5.74% for—3 speed grade devices, and 1.68%, 2.75% and 4.10% for—4 grade devices. Additional benefits in area and speed are expected when the proposed accumulator is used for a standard cell ASIC design. The reason for that is that FPL's carry chains used in [11] are almost as fast as the CSA implementation using $2^4 \times 1$ SRAM logic elements.

Table 1. Scaled accumulator decision logic table.

$C0$	$c1$	$c2$	$c3$	Result
0	0	0	0	$s1$
0	0	0	1	$s1$
0	0	1	1	$s2$
1	0	1	1	$s2$
1	0	1	0	$s3$
1	0	0	0	$s3$
0	1	0	0	$s3$
Any other combination				–

Table 2. Comparison between the proposed modulo m_j accumulator and the accumulator in [11].

	Modified modulo accumulator in [11]			CSA-based modulo accumulator		
		Throughput (MHz)			Throughput (MHz) improve (%)	
	LEs	-3^a	-4^a	LEs	-3^a	-4^a
6-bit modulus	37	50.50	40.48	55	52.08 (3.13%)	41.15 (1.68%)
7-bit modulus	43	48.70	39.24	61	50.66 (4.02%)	40.32 (2.75%)
8-bit modulus	49	47.18	38.05	74	49.89 (5.74%)	39.61 (4.10%)

[a]Device speed grade.

The RNS-DA mechanization will now be derived for wavelet filter banks. A modulo m_j path of the direct RNS-DA implementation of the ith-octave filter bank is defined in terms of an n_j-bit unsigned number:

$$\left|a_n^{(i-1)}\right|_{m_j} = \sum_{l=0}^{n_j-1} 2^l \left|a_{n,l}^{(i-1)}\right|_{m_j} \tag{15}$$

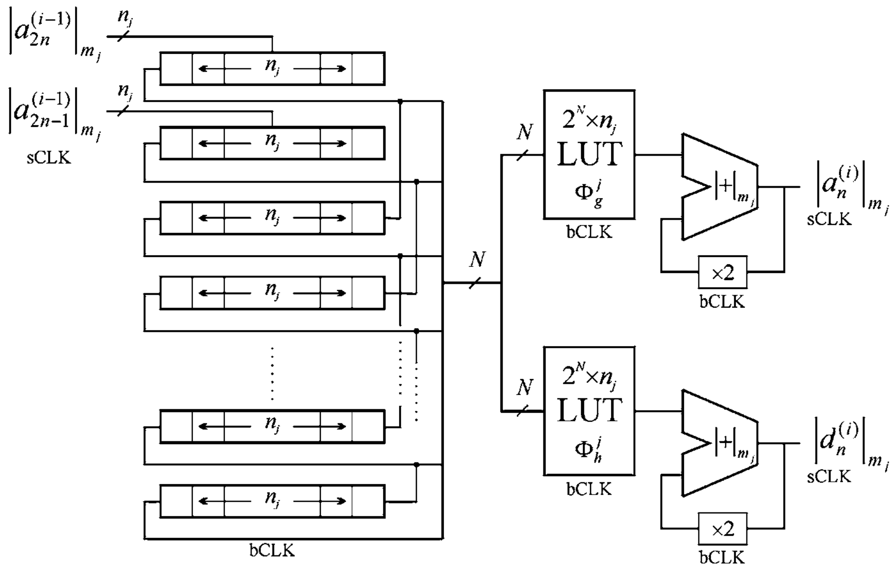

Figure 5. RNS-DA 1-D DWT architecture.

where $n_j = \lceil \log_2(m_j) \rceil$, and $|a_{n,l}^{(i-1)}|_{m_j}$ is the lth bit of residue $|a_n^{(i-1)}|_{m_j}$. Substituting Eq. (15) into Eq. (2), interpreted in a modulo m_j sense, the ith-octave-approximation and detail sequences can be written as:

$$\begin{aligned} \left|a_n^{(i)}\right|_{m_j} &= \left| \sum_{k=0}^{N-1} g_k \sum_{l=0}^{n_j-1} 2^l \left|a_{2n-k,l}^{(i-1)}\right|_{m_j} \right|_{m_j} \\ i &= 1, 2, \ldots, J \\ \left|d_n^{(i)}\right|_{m_j} &= \left| \sum_{k=0}^{N-1} h_k \sum_{l=0}^{n_j-1} 2^l \left|a_{2n-k,l}^{(i-1)}\right|_{m_j} \right|_{m_j} \\ \left|a_n^{(0)}\right|_{m_j} &\equiv \left|x_n\right|_{m_j} \end{aligned} \tag{16}$$

Finally, by defining the DA functions:

$$\Phi_g^j(l) = \left| \sum_{k=0}^{N-1} g_k a_{2n-k,l}^{(i-1)} \right|_{m_j} \quad \Phi_h^j(l) = \left| \sum_{k=0}^{N-1} h_k a_{2n-k,l}^{(i-1)} \right|_{m_j} \tag{17}$$

and interchanging the order of summations in Eq. (16), the RNS encoded ith-octave filter bank outputs are computed to be:

$$\left|a_n^{(i)}\right|_{m_j} = \left| \sum_{l=0}^{n_j-1} 2^l \Phi_g^j(l) \right|_{m_j} \quad \left|d_n^{(i)}\right|_{m_j} = \left| \sum_{l=0}^{n_j-1} 2^l \Phi_h^j(l) \right|_{m_j} \tag{18}$$

Figure 5 summarizes the modulo m_j RNS-DA architecture for the ith-octave analysis filter bank. N registers are used to left shift the input samples. The two inputs are sampled at a rate sCLK, the two LUTs storing Φ_g^j and Φ_h^j are accessed in each bit clock (bCLK) cycle to generate the terms of the relationship given in Eq. (18). The clock sCLK is easily generated dividing bCLK by the modulus width, n_j. Finally, two modified modulo m_j accumulators compute recursively and in parallel the ith-octave-approximation, $|a_n^{(i)}|_{m_j}$, and detail, $|d_n^{(i)}|_{m_j}$ sequences.

As in Eqs. (8) and (9) for a 2C-DA design, the polyphase filter bank [24] implementation can be considered in an attempt to reducing the DA LUT size. A polyphase RNS-DA architecture, suitable for high-order filter banks, is shown in Fig. 6. The production of $|a_n^{(i)}|_{m_j}$ and $|d_n^{(i)}|_{m_j}$ is computed as a polyphase filter bank, represented by:

$$\begin{aligned} \left|a_n^{(i)}\right|_{m_j} &= \left| \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{g_0}^j(l) \right|_{m_j} + \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{g_1}^j(l) \right|_{m_j} \right|_{m_j} \\ \left|d_n^{(i)}\right|_{m_j} &= \left| \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{h_0}^j(l) \right|_{m_j} + \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{h_1}^j(l) \right|_{m_j} \right|_{m_j} \end{aligned} \tag{19}$$

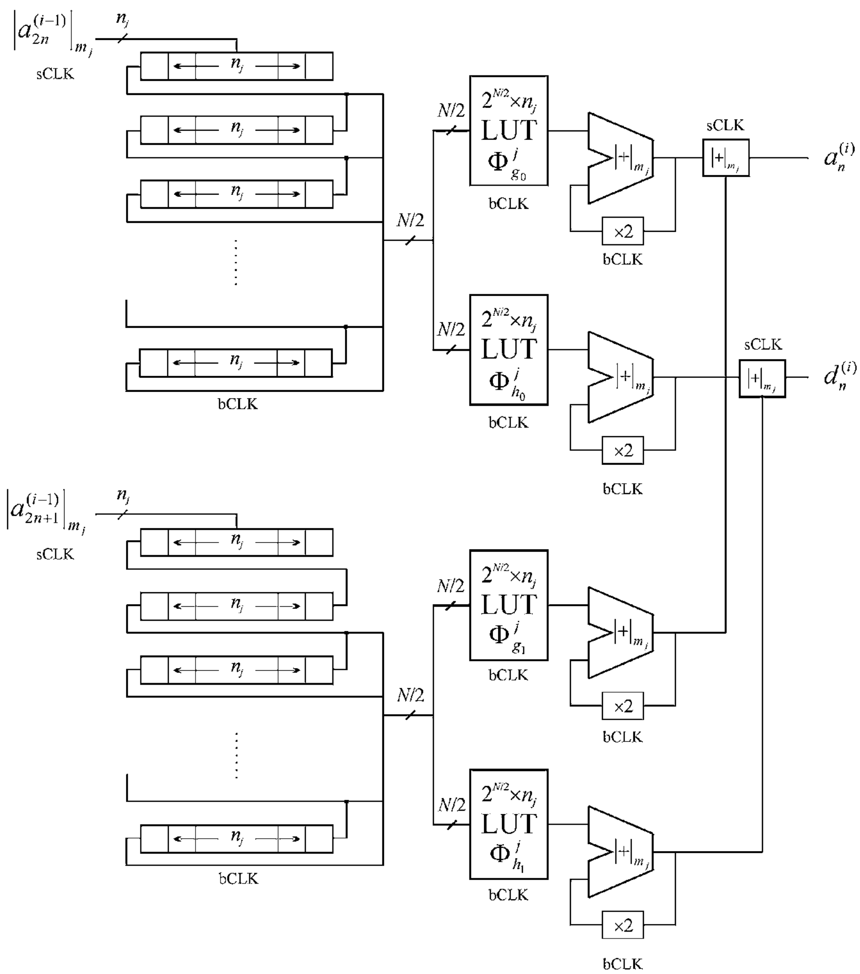

Figure 6. Polyphase RNS-DA 1-D DWT architecture.

where the contents of the four $2^{N/2} \times n_j$ LUTs are given by:

$$\begin{aligned}
\Phi_{g_0}^j(l) &= \left| \sum_{k=0}^{N/2-1} g_0(k) a_{2n-2k,l}^{(i-1)} \right|_{m_j} \\
\Phi_{g_1}^j(l) &= \left| \sum_{k=0}^{N/2-1} g_1(k) a_{2n-2k-1,l}^{(i-1)} \right|_{m_j} \\
\Phi_{h_0}^j(l) &= \left| \sum_{k=0}^{N/2-1} h_0(k) a_{2n-2k,l}^{(i-1)} \right|_{m_j} \\
\Phi_{h_1}^j(l) &= \left| \sum_{k=0}^{N/2-1} h_1(k) a_{2n-2k-1,l}^{(i-1)} \right|_{m_j}
\end{aligned} \tag{20}$$

In this way, four $2^{N/2} \times n_j$ are necessary to compute Eq. (19) instead of two $2^N \times n_j$ LUTs as in the previous approach. The four LUT outcomes read in each bCLK cycle are processed by means of four aforementioned modified modulo m_j accumulator, and finally pair-wise modulo added [32] to obtain the final outputs. Detailed information about the implementation of modulo adders using FPL technology can be found in [21]. This architecture enables a reduction in LUT address space requirements and is suitable for high-order wavelet filters.

An RNS-DA architecture for the ith-octave synthesis filter bank is derived by representing the inputs as n_j-bit unsigned words:

$$\left|\hat{a}_n^{(i)}\right|_{m_j} = \sum_{l=0}^{n_j-1} 2^l \left|\hat{a}_{n,l}^{(i)}\right|_{m_j} \quad \left|\hat{d}_n^{(i)}\right|_{m_j} = \sum_{l=0}^{n_j-1} 2^l \left|\hat{d}_{n,l}^{(i)}\right|_{m_j} \tag{21}$$

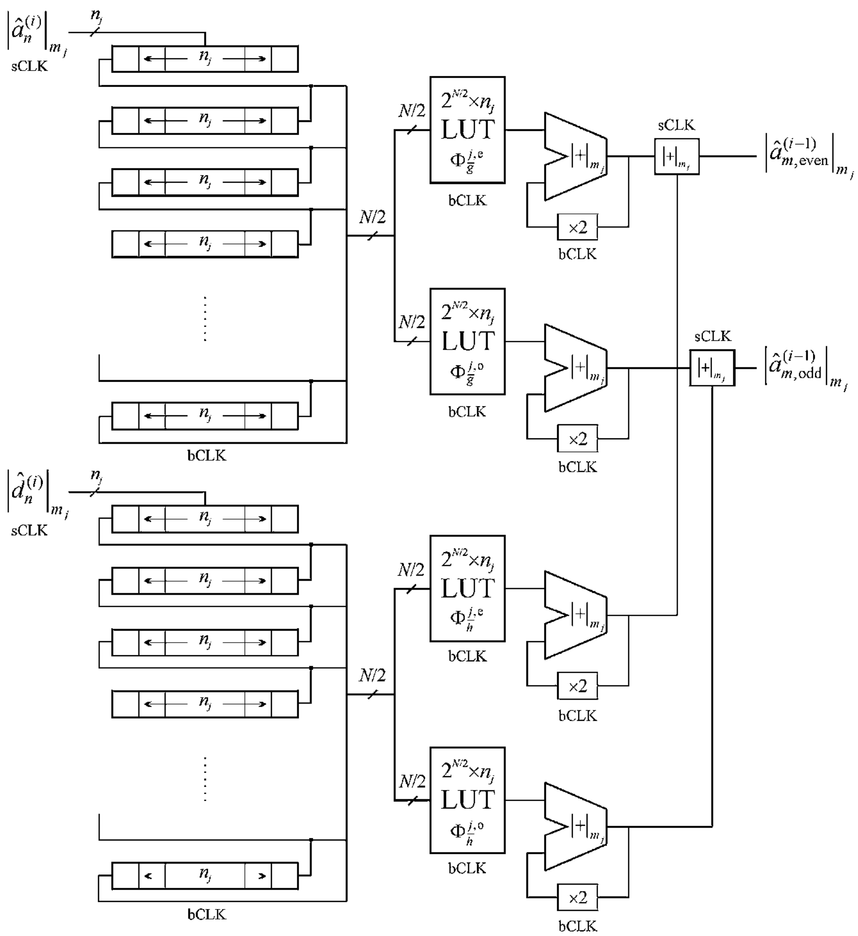

Figure 7. RNS-DA 1-D IDWT architecture.

By interchanging the order of summations, the octave-i reconstruction filter bank is computed as:

$$|\hat{a}_m^{(i-1)}|_{m_j} = \begin{cases} \left| \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{\bar{h}}^{j,e}(l) \right|_{m_j} + \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{\bar{g}}^{j,e}(l) \right|_{m_j} \right|_{m_j} & m \text{ even} \\ \left| \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{\bar{h}}^{j,o}(l) \right|_{m_j} + \left| \sum_{l=0}^{n_j-1} 2^l \Phi_{\bar{g}}^{j,o}(l) \right|_{m_j} \right|_{m_j} & m \text{ odd} \end{cases} \tag{22}$$

where:

$$\Phi_{\bar{h}}^{j,e}(l) = \left| \sum_{k=0}^{N/2-1} \bar{h}_{2k} \hat{d}_{\frac{m}{2}-k,l}^{(i)} \right|_{m_j}$$

$$\begin{aligned} \Phi_{\bar{h}}^{j,o}(l) &= \left| \sum_{k=0}^{N/2-1} \bar{h}_{2k+1} \hat{d}_{\frac{m-1}{2}-k,l}^{(i)} \right|_{m_j} \\ \Phi_{\bar{g}}^{j,e}(l) &= \left| \sum_{k=0}^{N/2-1} \bar{g}_{2k} \hat{a}_{\frac{m}{2}-k,l}^{(i)} \right|_{m_j} \\ \Phi_{\bar{g}}^{j,o}(l) &= \left| \sum_{k=0}^{N/2-1} \bar{g}_{2k+1} \hat{a}_{\frac{m-1}{2}-k,l}^{(i)} \right|_{m_j} \end{aligned} \tag{23}$$

The resulting modulo m_j RNS-DA architecture for the ith-octave synthesis filter bank is shown in Fig. 7. The inputs $|\hat{a}_n^{(i)}|_{m_j}$ and $|\hat{d}_n^{(i)}|_{m_j}$ are sampled at a rate sCLK. Two buffers, consisting of $N/2$ registers, are used to shift the input sequence and four LUTs store the functions $\Phi_{\bar{g}}^{j,e}$, $\Phi_{\bar{g}}^{j,o}$, $\Phi_{\bar{h}}^{j,e}$ and $\Phi_{\bar{h}}^{j,o}$. The output is computed concurrently as the summation of the low-pass and high-pass filters over even and odd cycles.

In this manner, the values $|\hat{a}_{m,\text{even}}^{(i-1)}|_{m_j}$ and $|\hat{a}_{m,\text{odd}}^{(i-1)}|_{m_j}$ are computed using two modulo m_j modified accumulators clocked by the bit clock rate bCLK, and two modulo m_j adders, clocked at sCLK. Note that sCLK is generated by dividing bCLK by n_j.

6. Comparison of 2C-DA and RNS-DA DWT: FPL Implementation

The RNS-DA scheme takes advantage of short wordlength computations to overcome the channel barrier problem. Processing within an RNS channel is accomplished in n_j accumulation cycles, where n_j is small. RNS-DA designs can exceed the performance of a 2C-DA system that requires more accumulation cycles to implement a typical filter. The sampling frequency of the RNS-DA scheme, $f_{\text{RNS-DA}}^{\text{sCLK}}$, is related to that of the 2C-DA rate $f_{\text{2C-DA}}^{\text{sCLK}}$, by:

$$f_{\text{RNS-DA}}^{\text{sCLK}} = \frac{B}{n_j}\frac{f_{\text{RNS-DA}}^{\text{bCLK}}}{f_{\text{2C-DA}}^{\text{bCLK}}} f_{\text{2C-DA}}^{\text{sCLK}} \tag{24}$$

where $f_{\text{RNS-DA}}^{\text{bCLK}}$ and $f_{\text{2C-DA}}^{\text{bCLK}}$ are the accumulation clock frequency of RNS-DA and 2C-DA schemes, respectively. In this way, the RNS-DA improvements are proportional to the wordlength ratio (i.e., $B{:}n_j$), and the bit-clock ratio (i.e., $f_{\text{RNS-DA}}^{\text{bCLK}} : f_{\text{2C-DA}}^{\text{bCLK}}$).

Implementation of a DWT was considered using Altera FLEX10K devices. Different input, coefficient, and output precisions were considered in order to assess the advantage of the schemes proposed on the overall sampling frequency. The use of 5-bit wide RNS channels was found to be an attractive choice since the sampling frequency is divided by the modulus width. The dynamic range is covered with 6-bit moduli, say {32, 31, 29, 27, 25} and {32, 31, 29, 27, 25, 23}, cover a range from 23 to 29-bits respectively. Table 3 audits the number of LEs and EABs required for the analysis and synthesis filter bank, as well as the throughput of a 2C-DA and 5- and 6-bit moduli RNS-DA schemes. These tables include results for a modulo 32 and 64 DA channel, as well as for generic 5- and 6-bit RNS channels. The eight-tap filters, considered in Table 3, were compared for RNS-DA and 2C-DA DWT. Although non-polyphase RNS-DA architectures require $2^8 \times 6$ LUTs, instead of $2^4 \times 6$, these structures are preferred over polyphase architectures since less EABs are needed. However, polyphase architectures are ideal for higher order filters to enable fitting DA LUTs on embedded FPL device resources. On the other hand, for the special case of the 8-tap theme polyphase filter bank, no EABs are required if DA LUTs are mapped to $2^4 \times 1$ LEs as shown in Table 3.

The maximum bandwidth provided by DWT filterbanks based on 2C-DA and RNS-DA was compared for different precisions. Figure 8 yields the maximum sampling rate of a DWT filterbank using 2C-DA and RNS-DA (for 5- and 6-bit modulus sets) as a function of the input precision. Notice that, for a 5-bit RNS-DA solution, the sampling frequency is always higher than for a 2C-DA DWT filterbank. The overall throughput of 2C-DA filter banks decreases as the input precision increases. However, this decrease in the overall sampling rate does not occur in RNS-DA scheme if fixed bit-width modulus are used to handle the increasing dynamic range. Thus, RNS-DA provided an increase in the overall sampling rate, up to 136.63% and 156.27%, for –3 and –4 grade devices, respectively, for a 14-bit input design. As a result, RNS-DA is seen to represent an efficient tool for providing a sustained throughput when the precision is increased.

Table 3 provides results obtained for the synthesis filter bank too. Different implementations of the system in Fig. 7 were considered. In this way, mapping the DA LUTs on LEs was found to be more efficient than using EABs in terms of hardware requirements. When compared to an analysis filter bank, these architectures require less memory bits since 4-bit LUTs, instead of 8-bit address memories, are required. In order to take advantage of the built-in memory blocks and reduce the number of accumulators from 4 to 2, a new RNS-DA architecture that uses two $2^N \times n_j$ LUTs, was derived. The enhancement can be carried out by considering the computation of an even filter involving the even coefficients of the low-pass and high-pass, and an odd filter involving the odd coefficients of the low-pass and high-pass filters. Such filters are two-input filters and are defined as follows:

$$\left|\hat{a}_m^{(i-1)}\right|_{m_j} = \begin{cases} \left|\displaystyle\sum_{l=0}^{n_j-1} 2^l \Phi_{\bar{g},\bar{h}}^{j,\text{e}}(l)\right|_{m_j} & m \text{ even} \\ \left|\displaystyle\sum_{l=0}^{n_j-1} 2^l \Phi_{\bar{g},\bar{h}}^{j,\text{o}}(l)\right|_{m_j} & m \text{ odd} \end{cases} \tag{25}$$

Table 3. Hardware requirements and performance for 5- and 6-bit RNS-DA DWT channels and for different precision 2C-DA architectures.

8-tap DWT filter bank implementation	Number of LEs	Number of EABs (memory bits)	f_{bCLK} (MHz) Speed grade −3 and −4	f_{sCLK}(MHz) Speed grade −3 and −4	Throughput improvement of RNS-DA
6-bit modulus RNS-DA channel	189[a]	2 (3072)[a]	34.72 (−4)	5.59 (−4)	
	373[b]	4 (384)[b]	43.47 (−3)		
	397[c]	0[c]			
Modulo 64 RNS-DA channel	100[a]	2 (3072)[a]	65.35 (−4)	7.25 (−3)	
	168[b]	4 (384)[b]	81.96 (−3)		
	192[c]	0[c]			
5-bit modulus RNS-DA channel	159[a]	2 (2560)[a]	39.84 (−4)	7.97 (−4)	
	312[b]	4 (320)[b]	49.12 (−3)		
	332[c]	0[c]			
Modulo 32 RNS-DA channel	85[a]	2 (2560)[a]	74.73 (−4)	9.82 (−3)	
	140[b]	4 (320)[b]	91.12 (−3)		
	160[c]	0[c]			
2C-DA					
8-bit input	220[d]	4 (6656)[d]	50.00 (−4)	6.25 (−4)	27.52% (−4)
10-bit coeffs.	434[e]	6 (768)[e]	66.66 (−3)	8.33 (−3)	17.89% (−3)
21-bit output	482[f]	0[f]			
10-bit input	248[d]	4 (6656)[d]	45.87 (−4)	4.59 (−4)	73.64% (−4)
10-bit coeffs.	482[e]	6 (768)[e]	61.72 (−3)	6.17 (−3)	59.16% (−3)
23-bit output	530[f]	0[f]			
12-bit input	292[d]	4 (7680)[d]	46.51 (−4)	3.88 (−4)	105.41% (−4)
12-bit coeffs.	574[e]	8 (896)[e]	63.69 (−3)	5.31 (−3)	84.93% (−3)
27-bit output	630[f]	0[f]			
14-bit input	320[d]	4 (7680)[d]	43.47 (−4)	3.11 (−4)	156.27% (−4)
12-bit coeffs.	630[e]	8 (1024)[e]	58.13 (−3)	4.15 (−3)	136.63% (−3)
29-bit output	694[f]	0[f]			

[a] Architectures RNS-DA shown in Figs. 5 and 9.
[b] Architectures RNS-DA shown in Figs. 6 and 7.
[c] Architectures RNS-DA shown in Figs. 6 and 7 synthesized without using EABs.
[d] Architectures 2C-DA shown in Fig. 1 or the 2C design of Fig. 9.
[e] Architectures 2C-DA shown in Figs. 2 and 3.
[f] Architectures 2C-DA shown in Figs. 2 and 3 synthesized without using EABs.

where $\Phi^{j,\mathrm{e}}_{\bar{g},\bar{h}}(l)$ and $\Phi^{j,\mathrm{o}}_{\bar{g},\bar{h}}(l)$ are defined as:

$$\begin{aligned}\Phi^{j,\mathrm{e}}_{\bar{g},\bar{h}}(l) &= \left|\sum_{k=0}^{N/2-1}\left[\bar{g}_{2k}\hat{a}^{(i-1)}_{\frac{m-1}{2}-k,l} + \bar{h}_{2k}\hat{d}^{(i-1)}_{\frac{m-1}{2}-k,l}\right]\right|_{m_j} \\ \Phi^{j,\mathrm{o}}_{\bar{g},\bar{h}}(l) &= \left|\sum_{k=0}^{N/2-1}\left[\bar{h}_{2k+1}\hat{d}^{(i-1)}_{\frac{m}{2}-k,l} + \bar{g}_{2k+1}\hat{a}^{(i-1)}_{\frac{m}{2}-k,l}\right]\right|_{m_j}\end{aligned} \tag{26}$$

which can be stored in two $2^N \times n_j$ LUTs. Both are addressed by the N-bit vector involving $N/2$ input samples of each input sequence $\{\hat{a}^{(i)}_{m/2}, \hat{a}^{(i)}_{m/2-1}, \ldots, \hat{a}^{(i)}_{m/2-N/2+1}, \hat{d}^{(i)}_{m/2}, \hat{d}^{(i)}_{m/2-1}, \ldots, \hat{d}^{(i)}_{m/2-N/2+1}\}$ (m even). In addition, two consecutive samples of the output sequence are computed concurrently. Figure 9 shows this new RNS-DA architecture for the ith-octave reconstruction filter bank. Note that this architecture requires only two scaled modulo m_j accumulators and half the EAB resources of the previous approach, as shown in Table 3. Since only two CSA-based modulo accumulators are required for the new architecture, the total number of LEs needed is reduced by almost 50%.

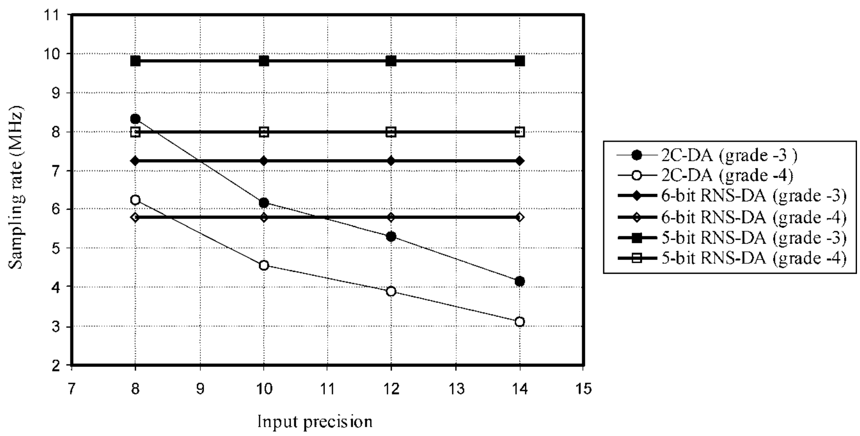

Figure 8. Overall 1-D DWT throughput of different input precision configurations.

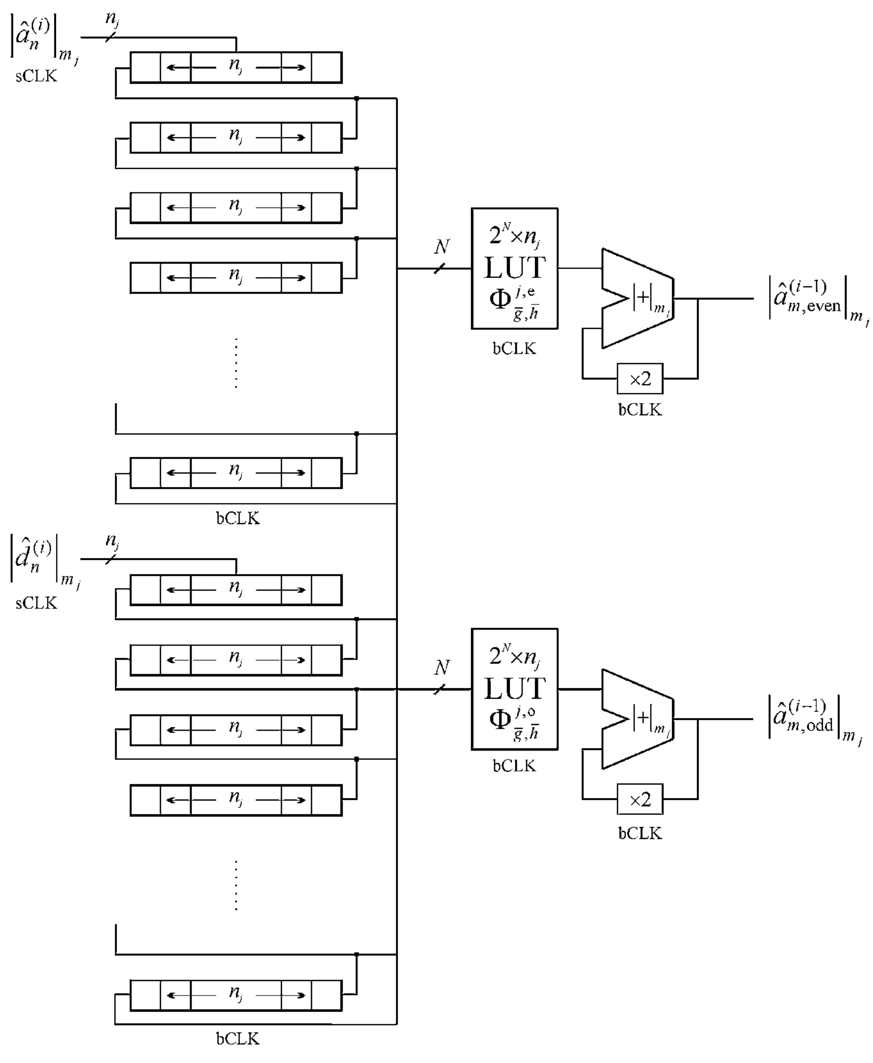

Figure 9. Architecture with only two scaled accumulator for the octave-i synthesis filter bank.

7. Parallel RNS-DA DWT Architectures

The original DA paradigm, disclosed by Peled and Liu [5], considered a number of versions of the basic architecture. The DA architecture presented in Eq. (10), requires a minimum number of LUTs. At the other extreme, Peled and Lui proposed a multi-LUT architecture designed for speed. Maximum throughput for a 2C-DA would occur if, in Eq. (10), every one of the B_i-bit locations had a dedicated LUT. Such a design contains B_i copies of the four $2^{N/2} \times W'$ LUT groups shown in Fig. 2. The RNS-DA, however, provides a different opportunity. The computation of Eq. (18) can be implemented with a multi-LUT architecture, specifically in terms of $2n_j\ 2^N \times n_j$ LUTs implementing the functions $|2^l \Phi_g^j|_{m_j}$ and $|2^l \Phi_h^j|_{m_j}$ for $l = 0, 1, \ldots, n_j - 1$. The address of the lth LUT consists of the l-th bit of N buffered signal samples. The output sequences, $|a_n^{(i)}|_{m_j}$ and $|d_n^{(i)}|_{m_j}$, are computed by adding n_j LUT outcomes per cycle by means of the pipelined modulo m_j adder [21] tree shown in Fig. 10. The computation of the ith-octave reconstruction (synthesis) filter bank can also be carried out by using the parallel polyphase RNS-DA design shown in Fig. 11. This is achieved by using $4n_j\ 2^{N/2} \times n_j$ LUTs to implements the function $|2^l \Phi_{\tilde{g}}^{j,\mathrm{e}}|_{m_j}$, $|2^l \Phi_{\tilde{g}}^{j,\mathrm{o}}|_{m_j}$, $|2^l \Phi_{\tilde{h}}^{j,\mathrm{e}}|_{m_j}$ and $|2^l \Phi_{\tilde{h}}^{j,\mathrm{e}}|_{m_j}$, for $l = 0, 1, \ldots, n_j - 1$. The lth LUT address consists of the lth bits of $N/2$ buffered samples of $|\hat{a}_n^{(i)}|_{m_j}$ ($|\hat{d}_n^{(i)}|_{m_j}$). Finally, the sequence $|\hat{a}_m^{(i-1)}|_{m_j}$ can be computed by adding the LUT words by means of a modulo m_j adder tree as shown in Fig. 11. Note that in these instances, only a clock rate is required. Both the computation of the 1-D DWT and 1-D IDWT, using the fast parallel RNS-DA design methodology, leads to a significant increase in the number of LUTs and adders. Table 4 summarizes the parallel architecture in the context of an 8-tap analysis filter bank. Table 5 shows hardware requirements of recursive RNS-DA and parallel RNS-DA architectures for the 1-D DWT and its inverse when N-tap filters are computed by means of K parallel $\lceil N/K \rceil$-tap sub-filters. For instance, a 16-tap filter bank for the 1-D DWT would require two $2^{16} \times n_j$ LUTs using the normal RNS-DA design methodology or, in polyphase form, using small $2^8 \times n_j$ LUTs by means of 8-tap sub-filters. Thus, the proposed strategy efficiently reduces the LUT address space of these systems by considering multiple RNS-DA units working in parallel.

Table 4. Hardware requirements and throughput for 6-bit modulus 8-tap parallel RNS-DA architectures over Altera FPL devices.

Parallel RNS-DA architecture	FLEX10K Speed grade −3 and −4	FLEX10KE Speed grade −1 and −3
Number of LEs	240	240
Number of EABs (memory bits)	2 × 6 ($2 \times 6 \times 2^8 \times 6$)	6 ($6 \times 2^8 \times 12$)
Throughput for several speed grade devices (MHz)	84.74 (−3) 68.96 (−4)	135.13 (−1) 86.20 (−3)

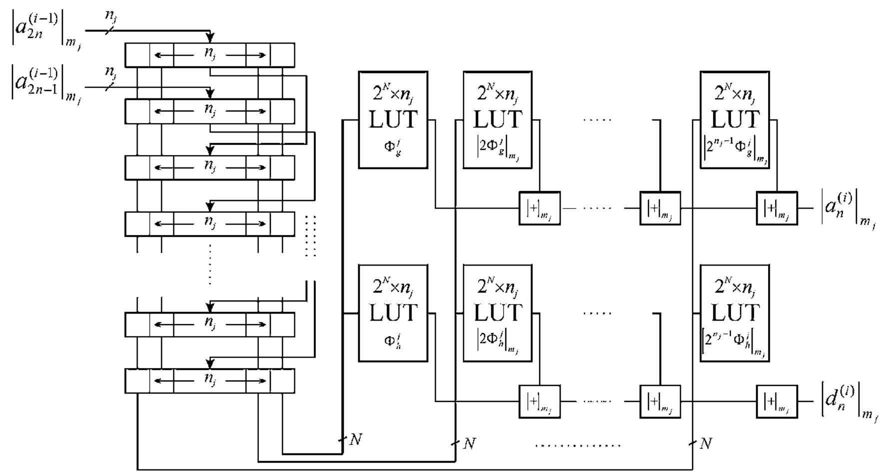

Figure 10. Parallel RNS-DA 1-D DWT architecture.

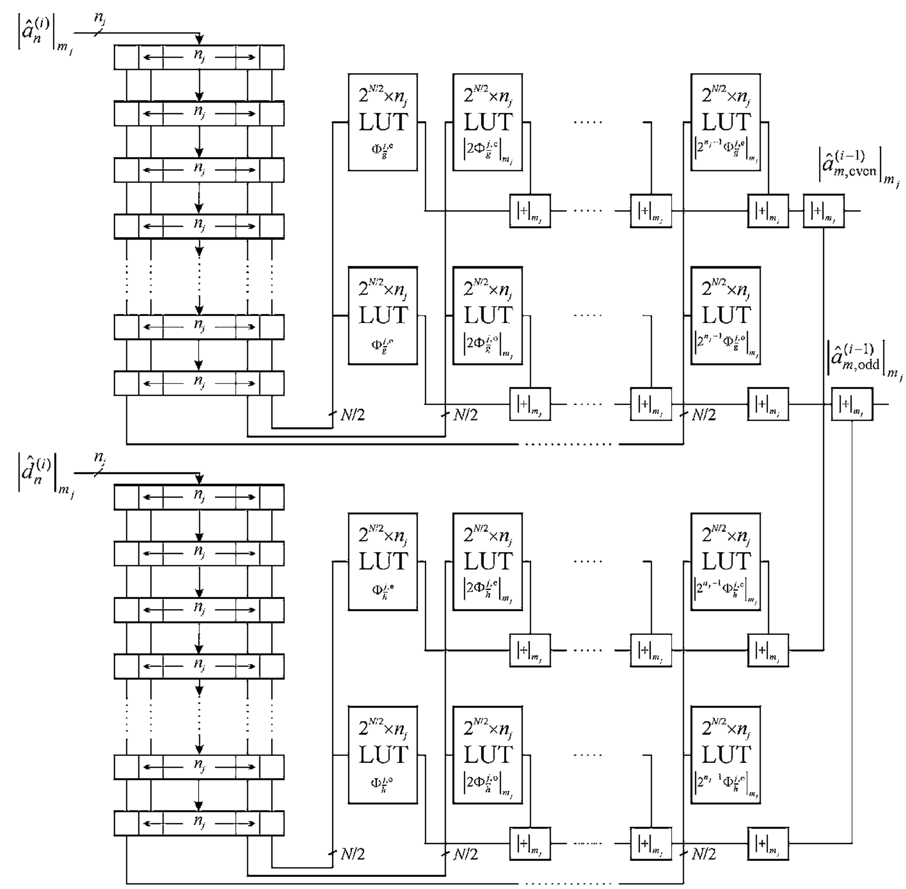

Figure 11. Parallel RNS-DA 1-D IDWT architecture.

8. Binary-to-RNS and RNS-to-Binary Conversion

A historical barrier to the use of the RNS at the system-level has been the overhead penalty associated with binary-to-RNS and RNS-to-binary conversion. Binary-to-RNS conversion can be carried out efficiently on FPL devices by decomposing the 2C B-bit word, say x, into a weighted sum of smaller words $\bar{x}_i$ (e.g., 4-bit words). Equation (27) exemplifies the case of a 4-bit decomposition, namely:

$$\begin{aligned} |x|_{m_j} &= \left| -2^{B-1}x_{B-1} + \sum_{l=0}^{B-2} 2^l x_l \right|_{m_j} \\ &= \left| -2^{B-1}x_{B-1} + \sum_{i=0}^{p-1} \bar{x}_i 2^{4i} \right|_{m_j} \end{aligned} \tag{27}$$

Table 5. Hardware requirements of recursive RNS-DA and parallel RNS-DA architectures for the 1-D DWT and its inverse when N-tap filters are computed by means of K DA subfilters.

	ith -octave analysis filter bank				ith-octave synthesis filter bank			
	LUTs	LUT size	mod m_j adders	mod m_j accs.	LUTs	LUT size	mod m_j adders	mod m_j accs.
RNS-DA based	$2K$	$2^{\lceil N/K\rceil} \times n_j$	$2(K-1)$	$2K$[a]	$4K$	$2^{\lceil N/2K\rceil} \times n_j$	$2+4(K-1)$	$4K$[a]
Parallel RNS-DA architecture	$2Kn_j$	$2^{\lceil N/K\rceil} \times n_j$	$2K(n_j-1)$ $+2(K-1)$	–	$4Kn_j$	$2^{\lceil N/2K\rceil} \times n_j$	$4K(n_j-1)$ $+4(K-1)+2$	–

[a]Modified modulo m_j accumulators.

Table 6. FPL resource requirements for the binary-to-RNS and RNS-to-binary converters.

		2C-to-RNS		RNS-to-2C	
	Modulus set	Number of LEs	Number of EABs (memory bits)	Number of LEs	Number of EABs (memory bits)
8-bit input	{64, 63, 51, 59}	81	0	48	8 (8192)
10-bit coeffs.	{32, 31, 29, 27, 25}	92	0	720	0
21-bit output					
10-bit input	{64, 63, 51, 59}	153	0	48	8 (8192)
10-bit coeffs.	{32, 31, 29, 27, 25, 23}	184	0	812	0
23-bit output					
12-bit input	{64, 63, 51, 59, 55}	208	0	96	10 (10240)
12-bit coeffs.	{32, 31, 29, 27, 25, 23}	200	0	812	0
27-bit output					
14-bit input	{64, 63, 51, 59, 55}	284	0	96	10 (10240)
12-bit coeffs.	{32, 31, 29, 27, 25, 23}	244	0	812	0
29-bit output					

requires only $2^4 \times n_j$ LUTs. Each can be efficiently mapped to n_j LEs, and modulo m_j adders as required. RNS-to-binary conversion implies the use of a CRT (Chinese Remainder Theorem)-based converter. The use of such a CRT-based converter is adequate for recursive RNS-DA DWT applications, since they do not demand high output conversion data rates. However, CRT conversion can often be a barrier in certain applications. The auto-scaling RNS-to-binary converter (ε-CRT) proposed by Griffin et al. [33] can overcome these drawbacks by using a few LUTs and binary (modulo 2^n) adder. For a scaled n-bit binary output, and a n_j-bit modulus set, this converter needs one $2^{n_j} \times n$ LUT for each modulus of the RNS and a n-bit adder tree. This solution is more appropriate for most applications demanding high data rates [22], including the presented parallel RNS-DA DWT architecture. Implementation data, using Altera FLEX10K devices, of the 2C-to-RNS and RNS-to-2C converters are provided in Table 6 for 5- and 6-bit modulus sets. The design for the 2C-to-RNS converter was derived from Eq. (27) while the ε-CRT algorithm with a 16-bit output was used for the RNS-to-2C converter. The results showed that using a 5-bit modulus set is not only optimum in the sense that it requires less clock cycles to compute the recursive RNS-DA equations, but it yields excellent performance improvements and enables the implementation of the converters with no need of using FPL embedded memory blocks. On the other hand, the operating frequency of both converters was found to be even higher than the sCLK frequency, so the high throughput of the presented RNS-DA architectures was not degraded when converters were inserted in the system.

9. Conclusions

This paper considers the design and implementation of digital filters using an RNS-DA paradigm and FPL technology. To achieve a high level of performance a new CSA-based scaled modulo accumulator was developed. With this, and other innovations, the RNS-DA architecture was shown to be well suited for integrating DSP objects with FPL devices. To test the voracity of the proposed methodology, a DWT filter bank was used as a standard. The exhaustive comparison of a 2C-DA and RNS-DA was carried out using commercially available FPL technology with the RNS-DA shown to be advantageous, especially for high-precision applications. A DWT filterbank having a 14-bit input, designed by means of the reported RNS-DA methodology, achieved a performance improvement over the equivalent 2C system of up to 156.27%, and with the conversion stage not degrading the throughput of the overall system.

Acknowledgments

J. Ramírez, A. García, and A. Lloris were supported by the Comisión Interministerial de Ciencia y

Tecnología (Spain) under project PB98-1354. CAD tools and supporting material were provided by Altera Corp., San Jose CA, under the Altera University Program. We would like to thank the anonymous reviewers for their valuable comments and suggestions that contributed to enhance the material presented in this paper.

References

1. D. Lautzenheiser, "Rapid Time to Production and ASICs Aren't Mutually Exclusive," *ISD Magazine*, Nov. 1998.
2. C. Dick, "FPGAs: The High-End Alternative for DSP Applications," *DSP Engineering*, Spring 2000.
3. F. Taylor and J. Mellot, *Hands-On Digital Signal Processing*, New York: McGraw Hill, 1998.
4. F. Taylor, "An Analysis of the Distributed Arithmetic Digital Filter," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, no. 5, 1986, pp. 1165–1170.
5. S.A. White, "Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review," *IEEE Acoustics, Speech and Signal Processing Magazine*, 1989, pp. 4–19.
6. A. Peled and B. Liu, "A New Hardware Realization of Digital Filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-22, no. 6, 1974, pp. 456–462.
7. Altera Corp., *Altera Digital Library*, June 2000.
8. Xilinx Inc., *The Programmable Logic Data Book*, 1999.
9. M. Soderstrand, W. Jenkins, G.A. Jullien, and F.J. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*, IEEE Press, 1986.
10. N.S. Szabo and R.I. Tanaka, *Residue Arithmetic and its Applications to Computer Technology*, New York: McGraw-Hill, 1967.
11. A. García, U. Meyer-Bäse, A. Lloris, and F. Taylor, "RNS Implementation of FIR Filters Based on Distributed Arithmetic Using Field-Programmable Logic," in *Proc. of the 1999 IEEE International Symposium on Circuits and Systems*, 1999, vol. 1, pp. 486–489.
12. V. Hamann and M. Sprachmann, "Fast Residual Arithmetic with FPGAs," in *Proc. of the Workshop on Design Methodologies for Microelectronics*, Slovakia, Sept. 1995.
13. H. Safiri, H. Ahamadi, G. Jullien, and V. Dimitrov, "Design and FPGA Implementation of Systolic FIR Filters Using the Fermat ALU," in *Proc. of the Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, 1996.
14. E. Di Claudio, F. Piazza, and G. Orlandi, "Fast Combinational RNS Processors for DSP Applications," *IEEE Transactions on Computers*, 1995, pp. 624–633.
15. L. Maltar, F.M.G. Franca, V.C. Alves, and C.L. Amorim, "Implementation of RNS Addition and RNS Multiplication into FPGAs," in *Proc. of the 6th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, April 1998, pp. 331–332.
16. U. Meyer-Bäse, J. Buros, W. Trautmann, and F. Taylor, "Fast Implementation of Orthogonal Wavelet Filterbanks Using Field-Programmable Logic," in *Proc. of the 1999 IEEE International Conference on Acoustics, Speech and Signal Processing*, March 1999, vol. 4, pp. 2119–2122.
17. J. Ramírez, A. García, P.G. Fernández, L. Parrilla, and A. Lloris, "RNS-FPL Merged Architectures for the Orthogonal DWT," *Electronics Letters*, vol. 36, no. 14, 2000, pp. 1198–1199.
18. J. Ramírez, A. García, P.G. Fernández, L. Parrilla, and A. Lloris, "Analysis of RNS-FPL Synergy for High Throughput DSP Applications: Discrete Wavelet Transform," in *Field Programmable Logic: The Roadmap to Reconfigurable Computing*, R.W. Hartenstein and H. Gruenbacher (Eds), LNCS Series, Berlin: Springer Verlag, 2000, pp. 342–351.
19. J. Ramírez, A. García, P.G. Fernández, and A. Lloris, "An Efficient RNS Architecture for the Computation of Discrete Wavelet Transforms on Programmable Devices," in *Proc. of the X European Signal Processing Conference*, Sept. 2000, pp. 255–258.
20. U. Meyer-Bäse, A. García, and F. Taylor, "Implementation of a Communications Channelizer Using FPGAs and RNS Arithmetic," *Journal of VLSI Signal Processing*, vol. 28, 2001, pp. 115–118.
21. A. García, U. Meyer-Bäse, and F.J. Taylor, "Pipelined Hogenauer CIC Filters Using Field-Programmable Logic and Residue Number System," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, WA, May 1998, vol. 5, pp. 3085–3088.
22. 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 *Proc. of the 2000 International Symposium on Circuits and Systems*, May 2000, vol. 5, pp. 321–324.
23. G. Strang and T. Nguyen, *Wavelets and Filter Banks*. Wellesly-Cambridge Press, 1997.
24. M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*. Englewood Cliffs, NJ: Prentice Hall, 1995.
25. C. Chakrabarti and M. Vishwanath, "Efficient Realizations of the Discrete and Continuous Wavelet Transform: From Single Chip Implementations to Mappings on SIMD Array Computers," *IEEE Transactions on Signal Processing*, vol. 43, 1995, pp. 759–771.
26. F. Marino, "A "Double-Face" Bit-Serial Architecture for the 1-D Discrete Wavelet Transform," *IEEE Transactions on Circuits and Systems II*, vol. 47, no. 1, 2000, pp. 65–71.
27. J. Fridman and E.S. Manolakos, "Distributed Memory and Control VLSI Architectures for the 1-D Discrete Wavelet Transform," *VLSI Signal Processing*, vol. VII, 1994, pp. 388–397.
28. T.C. Denk and K.K. Parhi, "VLSI Architectures for Lattice Structure Based Orthogonal Discrete Wavelet Transforms," *IEEE Transactions on Circuits and Systems II*, vol. 44, no. 2, 1997, pp. 129–132.
29. K.K. Parhi and T. Nishitani, "VLSI Architectures for Discrete Wavelet Transforms," *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, vol. 1, 1993, pp. 191–202.
30. M. Vishwanath, R.M. Owens, and M.J. Irwin, "VLSI Architectures for the Discrete Wavelet Transform," *IEEE Transactions on Circuits and Systems II*, vol. 42, no. 5, 1995, pp. 305–316.
31. F. Taylor, *Digital Filter Design Handbook*, Marcel Dekker, 1983.

32. M. Dugdale, "VLSI Implementation of Residue Adders Based on Binary adders", *IEEE Trans. on Circuits and Systems II*, vol. 39, no. 5, 1992, pp. 325–329.
33. M. Griffin, F.J. Taylor, and M. Sousa, "New Scaling Algorithms for the Chinese Remainder Theorem," in *Proc. of the 22nd Asilomar Conf. on Signals, Syst. and Comp.*, CA, 1988.

Javier Ramírez received the M.A.Sc. degree in Electronic Engineering in 1998, and the Ph.D degree in Electronic Enginnering in 2001, all from the University of Granada. Since 2001, he is an Assistant professor at the Dept. of Electronics and Computer Technology of the University of Granada (Spain). His research interest includes residue number system arithmetic, high performance digital signal processing and FPGA and VLSI signal processing systems. He has authored more than 50 technical journal and conference papers in these areas. He has served as reviewer for several international journals and conferences. He is a member of IEEE, and a SP and C&S Society member.
javierrp@ugr.es

Antonio García received the M.A.Sc. degree in Electronic Engineering (being awarded the Nation Best Academic Record) in 1995, the M.Sc. degree in Physics (majoring in Electronics) in 1997 and the Ph.D. degree in Electronic Engineering in 1999, all from the University of Granada (Spain). He was an Associate Professor at the Department of Computer Engineering of the Universidad Aut¢noma de Madrid before joining the Deparment of Electronics and Computer Technology at the University of Granada as an Associate Professor. His research interests include Residue Number System arithmetic, the application of RNS to high-performance digital signal processing, VLSI and FPL implementation of RNS-based systems and the use of RNS for low-power VLSI systems. He has authored over 50 technical papers in international journals and conferences and has served as reviewer for several international journals and conferences. He is a member of IEEE and a C, C&S and SP Society member.
agarcia@ditec.ugr.es

Uwe Meyer-Bäse received his BSEE, MSEE, and Ph.D. "Summa cum Laude" from the Darmstadt University of Technology in 1987, 1989, and 1995, respectively. In 1994 and 95 he hold a post-doc position in the "Inst. of Brain Research" in Magdeburg. In 1996 and 1997 he was a Visiting Professor at the University of Florida. From 1998 to 2000 Dr. Meyer-Baese worked in the ASIC industry. He is now a Professor in the Electrical and Computer Engineering Department at Florida State University. During his graduate studies he worked part time for TEMIC, Siemens, Bosch, and Blaupunkt. He holds 3 patents, has supervised more than 60 master thesis projects in the DSP/FPGA area, and gave four lectures at the University of Darmstadt in the DSP/FPGA area. He is author of three books including "Digital Signal Processing with Field Programmable Gate Arrays" and "Fast Digital Signal Processing" published by Springer-Verlag. He received in 1997 the Max-Kade Award in Neuroengineering. Dr. Meyer-Baese is a IEEE, BME, SP and C&S society member.
Uwe.Meyer-Baese@ieee.org

Fred J. Taylor received his BSEE degree from the Milwaukee School of Engineering in 1965; MS and Ph.D. degrees form the University of Colorado in 1965 and 1969 respectfully. He was a Member of the Technical Staff of Texas Instruments Corp. (1969/70), held academic positions at the University of Texas at El Paso (1970/75), University of Cincinnati (1975/83), and the University of Florida since 1983, where he is currently a professor of Electrical and Computer Engineering (ECE) and Computer and Information Science Engineering (CISE). He has held other visiting academic appointments at Southern Methodist University, CNAM (Paris), and THD (Darmstadt). He serves as a consultant to a number of Fortune 500 companies as well as government agencies. He has authored ten textbooks, twelve-chapter contributions for handbooks in the field of signal processing and encyclopedias, and three U.S. patents. Dr. Taylor has also authored over 100 archived journal articles. He is the co-founder of the Athena Group inc. in 1986, a DSP semiconductor silicon intellectual property company, where he now serves as the Chairman of Board.
fjt@hsdal.ufl.edu

Antonio Lloris received the M.Sc. Degree and the Ph.D. degree from the Universidad Complutense (Madrid). He was at the Centro de Investigaciones Tcnicas de Guip£zcoa (Spain) as a researcher and, as a lecturer, at the Escuela Tcnica Superior de Ingenieros Industriales de San Sebastin. He was at the Universities of Malaga and Murcia (Spain). Now he is a Full Professor at the University of Granada (Spain). His research interest include multiple-value logic, testing of digital circuits and signal processing using the residue number system.
lloris@ditec.ugr.es