# Systolic Algorithms and a Memory-Based Design Approach for a Unified Architecture for the Computation of DCT/DST/IDCT/IDST

Doru Florin Chiper, M. N. S. Swamy, *Fellow, IEEE*, M. Omair Ahmad, *Fellow, IEEE*, and Thanos Stouraitis, *Senior Member, IEEE*

***Abstract*—In this paper, an efficient design approach for a unified very large-scale integration (VLSI) implementation of the discrete cosine transform /discrete sine transform /inverse discrete cosine transform/inverse discrete sine transform based on an appropriate formulation of the four transforms into cyclic convolution structures is presented. This formulation allows an efficient memory-based systolic array implementation of the unified architecture using dual-port ROMs and appropriate hardware sharing methods. The performance of the unified design is compared to that of some of the existing ones. It is found that the proposed design provides a superior performance in terms of the hardware complexity, speed, I/O costs, in addition to such features as regularity, modularity, pipelining capability, and local connectivity, which make the unified structure well suited for VLSI implementation.**

***Index Terms*—Forward and inverse cosine and sine transforms, memory-based implementation techniques, systolic arrays, very large-scale integration (VLSI) algorithms.**

## I. Introduction

THE discrete cosine transform (DCT) and discrete sine transform (DST) [1]–[3] are orthogonal transforms, which are represented by basic functions used in many signal processing applications, especially in speech and image transform coding [4]. These transforms are good approximations to the statistically-optimal Karhunen–Loeve transform (KLT) [3], [4]. The choice of the DCT or DST depends on the statistical properties of the input signal, which in the case of image processing is subject to relatively fast changes. The DCT provides better results for a wide class of signals. However, there are other statistical processes, such as the first-order Markov sequences with specific boundary conditions, for which the DST is a better solution. Also, for low correlated input signals, DST provides a lower bit rate [4]. There are some applications as in [5] and [6], where both the DCT and DST are involved. Thus, a very large-scale integration (VLSI) structure that allows the use of both the DCT and DST is desired.

Since both the DCT and DST are computationally intensive, many efficient algorithms have been proposed to improve the performance of their implementation, but most of these are only good software solutions. For hardware implementation, appropriate restructuring of the classical algorithms or the derivation of new ones that can efficiently exploit the embedded parallelism is highly desirable. In order to obtain an optimal hardware implementation, it is necessary to treat the development of the algorithm, its architecture, and implementation in a synergetic manner.

Fast DCT and DST algorithms, based on a recursive decomposition, result in butterfly structures with a reduced number of multiplications, but lead to irregular architectures with complicated data routing and large design time, due to the structure of their signal flow graphs, even though efforts have been made to improve their regularity and modularity as in [7] and [8]. Also, the successive truncations involved in a recursive decomposition structure lead to a degradation in accuracy for a fixed point implementation. The VLSI structures based on time-recursive algorithms [8]–[11] are not suitable for pipelining due to their recursive nature, and suffer from numerical problems, which can severely compromise their low hardware complexity.

The data movement and transfer play a key role in determining the efficiency of a VLSI implementation of the hardware algorithms [13]–[16]. This is the reason why regular computational structures such as the circular correlation and cyclic convolution lead to efficient VLSI implementations [13]–[15] using modular and regular architectural paradigms such as the distributed arithmetic [17] and systolic arrays [18]. These structures also avoid complex data routing and management, thus leading to VLSI implementations with reduced complexity, especially when the transform length is sufficiently large.

Systolic arrays [18] represent an appropriate architectural paradigm that leads to an efficient VLSI implementation due to its regularity and modularity, with simple and local interconnections between the processing elements (PEs); at the same time, they yield a high-performance by exploiting concurrency through pipelining or parallel processing. However, a large portion of the chip is consumed by the multipliers, putting a severe limitation on the allowable number of PEs that could be included.

Manuscript received July 22, 2003; revised September 16, 2004. This work was supported by the Micronet National Network of Centers of Excellence, Natural Sciences and Engineering Research Council (NSERC) of Canada, and Fonds pour la Formation des Chercheurs et l'Aide a la Recherche of Quebec. This paper was recommended by Associate Editor N. Ramamurthy.

D. F. Chiper, M. N. S. Swamy, and M. O. Ahmad are with the Center for Signal Processing and Communications, Department of Electrical and Computer Engineering, Concordia University, Montreal, QC H3G 1M8, Canada (e-mail: swamy@ece.concordia.ca).

T. Stouraitis is with the Department of Electrical and Computer Engineering, University of Patras, Rio 26500, Greece.

Digital Object Identifier 10.1109/TCSI.2005.849109

The memory-based techniques [14], [15], [17], [19] are known to provide improved efficiency in the VLSI implementation of DSP algorithms through increased regularity, low hardware complexity and higher processing speed by efficiently replacing multipliers with small ROMs as in the distributed arithmetic (DA) or in the look-up table approach. The DA is popular in various digital signal processing (DSP) applications dominated by inner-product computations, where one of the operands can be fixed. It uses ROM tables to store the pre-computed partial sums of the inner product. Such a scheme has been adopted to implement several commercial products due to its efficiency in VLSI implementation [20], [21]. However, the main problem is that the ROM size increases exponentially with the transform size, thus rendering the technique impractical for large transform sizes. Moreover, due to the feedback connection in the accumulator stage, the structure obtained is difficult to pipeline.

In [14], a new memory-based implementation technique that combines some of the characteristics of the DA and systolic array approaches has been proposed. When one of the operands is fixed, one can efficiently replace the multipliers by small ROMs that contain the pre-computed results of the multiplication operations. If the size of the ROM is small, a significant increase in the processing speed can be obtained since the ROM access time is considerably smaller than the time required for a multiplication. Further, this technique has the following features [14].

1) It does not involve combined bit-serial and bit-parallel operations.
2) It allows the reduction of the overall ROM size from $2^N$ required for the standard DA approach to $N \times 2^L$ words, where $L$ is the word length of the operands, and $N$ is the transform length.
3) The resulting VLSI structures are easy to pipeline allowing an efficient combination of the memory-based implementation techniques with the systolic array concept.

Using the partial sums technique [20], it has been shown that the size of the ROM necessary to replace a multiplier can be further reduced to $2^{(L/2+1)}$ at the cost of an extra adder [14].

Most of the reported unified systolic array-based VLSI designs [11], [12], [23], [24] obtain the flexibility of computing DCT/DST and/or inverse DCT/inverse DST (IDCT/IDST) by feeding the different transform coefficients into the hardware structure. They cannot use efficiently the memory-based implementation techniques since they are not able to use the constant property of the coefficients, namely that for both the DCT and DST, the coefficients are the same and are fixed for each processor. Moreover, they use an additional control module to manage the feeding of the transform coefficients into the VLSI structure, and have a high I/O cost.

The unified DA-based implementations of the DCT/DST and IDCT/IDST algorithms based on a general formulation, presented in [25] and [26], also do not exploit the constant property of the transform coefficients in each processor, nor do they benefit from the advantages of the cyclic convolution structures. Thus, the unification is achieved with a lower computational throughput and a higher hardware complexity. In addition, they have the overheads of the bit-serial implementations with parallel-to-serial and serial-to-parallel conversions, and lower processing speeds compared to the bit-parallel ones as they need more than one clock cycle per operation. Moreover, they are difficult to pipeline and are appropriate only for small values of the transform length $N$.

We present an efficient design strategy to obtain a unified VLSI implementation of DCT, DST, IDCT, and IDST using a dual-port ROM-based DA-like realization technique. This unified implementation is achieved by an appropriate reformulation of the DCT, DST, IDCT and IDST algorithms, whose transform length is a prime number, so that they retain all the advantages of the cyclic convolution-based implementations. Thus, an efficient unified VLSI structure, wherein a large percentage of the chip area is shared by all the transforms, and which results in a high computing speed with a low hardware complexity, low I/O cost, and a high degree of regularity, modularity and local connectivity, is presented.

The paper is organized as follows. In Section II, the unified memory-based hardware algorithms encapsulated into the systolic arrays are presented. The proposed computational structures are illustrated using examples for the forward and inverse DCT/DST algorithms. In Section III, the relevant design aspects of the unified VLSI structure of implementing the DCT, DST, IDCT, IDST using a dual-port ROM-based DA-like realization technique is presented. In Section IV, a performance analysis and comparisons of the proposed scheme is carried out and compared with that of the relevant unified structures. Section V contains a brief conclusion.

## II. Unified Systolic Algorithms for DCT/ DST/IDCT/IDST

In general, we can obtain a significant improvement in any computational structure of a VLSI implementation by appropriately restructuring the algorithmic computational structure of a DSP algorithm. This is sometimes called *algorithmic engineering* [27]. In order to efficiently make use of this restructuring approach, it is necessary to have a clear architectural target, which in this case is the systolic array paradigm. In the following, we show how such an approach can be used to allow the application of the memory-based implementation techniques in such a manner that a unified structure for the forward and inverse DCT/DST can be obtained.

In any computational structure, a multiplier could be replaced by a ROM of size $2^{2L}$, where $L$ is the word length of the two operands. In [14], it has been shown, in the case of DFT and DCT, that if one of the two operands in each multiplier is fixed, then one can successfully apply the memory-based implementation techniques to obtain significant reduction in the ROM size from $2^{2L}$ to $2^L$ in replacing each multiplier. However, restructuring the forward and inverse DCT/DST algorithms in such a way that we can obtain an efficient unified structure that allows the use of the memory-based implementation techniques is a challenging design problem. This is due to the fact that one operand should be fixed and the same in each of the corresponding multipliers in the structures realizing the four transforms. We now reformulate the four algorithms so that the multipliers in each PE have one of their operands fixed and be the same for each of the four transforms.

In order to obtain the desired computational structures for the VLSI algorithms of all the four transforms, we use appropriate index mappings defined by

$$\psi(k) = \begin{cases} \varphi'(k), & \text{if } \varphi'(k) \le \frac{(N-1)}{2} \\ \varphi'(N-1+k), & \text{otherwise} \end{cases} \quad (1)$$

$$\phi(k) = \begin{cases} \varphi'(k), & \text{if } \varphi'(k) > \frac{(N-1)}{2} \\ \varphi'(N-1+k), & \text{otherwise} \end{cases} \quad (2)$$

and

$$\xi(k) = (\zeta \circ \psi)(k) = \zeta(\psi(k)) \quad (3)$$

$$\eta(k) = (\zeta \circ \phi)(k) = \zeta(\phi(k)) \quad (4)$$

with

$$\varphi'(k) = \begin{cases} \varphi(k), & \text{if } k > 0 \\ \varphi(N-1+k), & \text{otherwise} \end{cases} \quad (5)$$

$$\varphi(k) = \langle g^k \rangle_N \quad (6)$$

$$\zeta(k) = \langle 2k \rangle_N \quad (7)$$

where $g$ is the primitive root of the Galois field of indexes and $\langle x \rangle_N$ represents $x$ modulo $N$.

### A. Unified VLSI Algorithm for the Forward DST and DCT

The DST and DCT of the input sequence $\{x(i) : i = 0, \ldots, N-1\}$ are respectively defined as [2], [1]

$$Y(k) = \sum_{i=0}^{N-1} x(i) \cdot \sin[(2i+1)k\alpha], \qquad k = 1, \ldots, N \quad (8)$$

$$Y(k) = \sum_{i=0}^{N-1} x(i) \cdot \cos[(2i+1)k\alpha], \qquad k = 0, \ldots, N-1 \quad (9)$$

where $\alpha = \pi/2N$.

If the length $N$ of the transforms is a prime number greater than 2, we can reformulate the computation of the two transforms in a unified manner. Introducing two new input sequences, defined as

$$x_a(N-1) = x(N-1) \quad (10)$$

$$x_a(i) = \begin{cases} x(i) + x_a(i+1), & \text{for DST} \\ (-1)^i x(i) + x_a(i+1), & \text{for DCT,} \end{cases} \quad \text{for } i = (N-2), \ldots, 1, 0 \quad (11)$$

$$x_b(N-1) = \begin{cases} -x(N-1), & \text{for DST} \\ x(N-1), & \text{for DCT} \end{cases} \quad (12)$$

$$x_b(i) = \begin{cases} (-1)^{i+1} x(i) - x_b(i+1), & \text{for DST} \\ x(i) - x_b(i+1) & \text{for DCT,} \end{cases} \quad \text{for } i = (N-2), \ldots, 1, 0 \quad (13)$$

and using appropriate permutations of the new sequences, we can decompose the computation of the DST and the DCT into the following two half-length cyclic convolutions having the same structure:

$$T_a(\psi(k)) = \sum_{i=1}^{(N-1)/2} [x_a(\psi(i-k)) + x_a(\phi(i-k))] \cdot \cos[\psi(i) \cdot 4\alpha] \quad (14)$$

$$T_b(\phi(k)) = \sum_{i=1}^{(N-1)/2} [x_b(\psi(i-k)) + x_b(\phi(i-k))] \cdot \cos[\psi(i) \cdot 4\alpha] \quad \text{for } k = 1, 2, \ldots, \frac{(N-1)}{2}. \quad (15)$$

The two convolutions given by (14) and (15) can be concurrently computed. Further, they have the same length and computational structures. However, as only the input sequences are different, one unified structure can be used to implement both (14) and (15), whether it be for the computation of DCT or DST. Moreover, as will be illustrated later through *Example* 1, (14) and (15) have a form that allows an efficient use of memory-based implementation techniques.

The output sequence can then be obtained using the following computational equations:

$$[x_a(0) + 2T_a(\psi(k))] \cdot P(k) = \begin{cases} Y(\xi(k)), & \text{for DST} \\ Y(\eta(k)), & \text{for DCT} \end{cases} \quad (16)$$

$$[x_b(0) + 2T_b(\phi(k))] \cdot Q(k) = \begin{cases} Y(\eta(k)), & \text{for DST} \\ Y(\xi(k)), & \text{for DCT,} \end{cases} \quad \text{for } k = 1, \ldots, \frac{(N-1)}{2} \quad (17)$$

$$x_b(0) + 2 \sum_{i=1}^{(N-1)/2} [x_b(\psi(i)) + x_b(\phi(i))] = \begin{cases} -Y(N), & \text{for DST} \\ Y(0), & \text{for DCT} \end{cases} \quad (18)$$

where

$$P(k) = \begin{cases} \sin[\psi(k) \cdot 2\alpha], & \text{for DST} \\ \sin[\phi(k) \cdot 2\alpha], & \text{for DCT} \end{cases} \quad (19)$$

$$Q(k) = \begin{cases} \cos[\phi(k) \cdot 2\alpha], & \text{for DST} \\ \cos[\psi(k) \cdot 2\alpha], & \text{for DCT.} \end{cases} \quad (20)$$

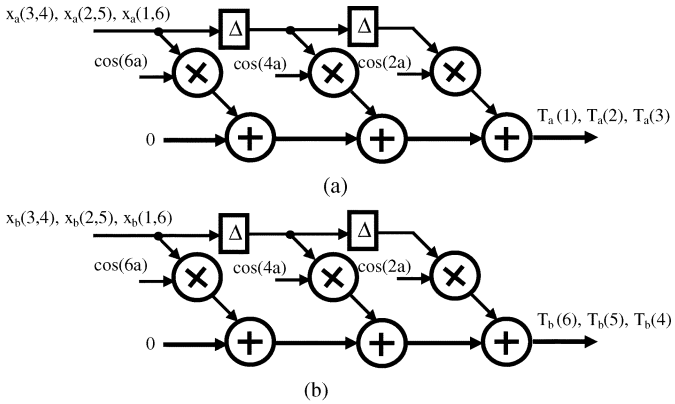
It is noted that in the computation of the DCT/DST using the unified structure, the core of the computation is in the implementation of (14) and (15), and it is of $O(N^2)$. The rest of the computation is relatively small, with a complexity of $O(N)$.

*Example 1:* We consider an example of DCT/DST with length $N = 7$ and primitive root $g = 3$. The conclusions drawn from this particular case can be easily extended for any length, which is a prime number. The DCT and DST are now reformulated using (14) and (15) in the form

$$\begin{bmatrix} T_a(3) \\ T_a(2) \\ T_a(1) \end{bmatrix} = \begin{bmatrix} x_a(1)+x_a(6) & x_a(3)+x_a(4) & x_a(2)+x_a(5) \\ x_a(2)+x_a(5) & x_a(1)+x_a(6) & x_a(3)+x_a(4) \\ x_a(3)+x_a(4) & x_a(2)+x_a(5) & x_a(1)+x_a(6) \end{bmatrix} \cdot \begin{bmatrix} \cos(12\alpha) \\ \cos(8\alpha) \\ \cos(4\alpha) \end{bmatrix} \quad (21)$$

$$\begin{bmatrix} T_b(3) \\ T_b(2) \\ T_b(1) \end{bmatrix} = \begin{bmatrix} x_b(1)+x_b(6) & x_b(3)+x_b(4) & x_b(2)+x_b(5) \\ x_b(2)+x_b(5) & x_b(1)+x_b(6) & x_b(3)+x_b(4) \\ x_b(3)+x_b(4) & x_b(2)+x_b(5) & x_b(1)+x_b(6) \end{bmatrix} \cdot \begin{bmatrix} \cos(12\alpha) \\ \cos(8\alpha) \\ \cos(4\alpha) \end{bmatrix}. \quad (22)$$

Equations (21) and (22) have specific structural properties that can be exploited to significantly increase the efficiency of the VLSI implementation. It is noted from (21) that the $\cos(k \cdot 2a)$, $k = 1, \ldots, (N-1)/2$ operands are time invariant and are respectively allocated to the $(N-1)/2$ PEs, such that one




**Note:** 1. $x_a(i,j)=x_a(i)+x_a(j)$ ; $x_b(i,j)=x_b(i)+x_b(j)$ ;
2. The input sequences have been extended circularly
3. a = $2\alpha$

Fig. 1. Signal flow graphs (SFG) of the computational core of the forward DCT/DST algorithm. (a) SFG representing (21). (b) SFG representing (22).

operand in each multiplier is fixed, as shown in Fig. 1(a). A comparison of Fig. 1(a) and (b), shows that, for a given $k$, the fixed operands $\cos(k \cdot 2a)$ in (21) and (22) are the same in any given multiplier.

In the matrix of (21), all the diagonal elements are identical. Also, the elements along any line parallel to the diagonal are also identical. Similar statements hold for the matrix of (22). This feature renders the input data elements to be efficiently used in all the PEs, thus providing a significant reduction in the I/O cost.

The output sequence for the DST is computed using

$$\begin{bmatrix} Y(6) \\ Y(4) \\ Y(2) \end{bmatrix} = \begin{bmatrix} [2T_a(3)+x_a(0)]\cdot\sin(6\alpha) \\ [2T_a(2)+x_a(0)]\cdot\sin(4\alpha) \\ [2T_a(1)+x_a(0)]\cdot\sin(2\alpha) \end{bmatrix} \tag{23}$$

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \end{bmatrix} = \begin{bmatrix} [2T_b(4)+x_b(0)]\cdot\cos(8\alpha) \\ [2T_b(5)+x_b(0)]\cdot\cos(10\alpha) \\ [2T_b(6)+x_b(0)]\cdot\cos(12\alpha) \end{bmatrix} \tag{24}$$

and

$$Y(7) = -\left[x_b(0)+2\sum_{i=1}^{3}[x_b(\psi(i))+x_b(\phi(i))]\right] \tag{25}$$

whereas the output sequence for the DCT is obtained using

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \end{bmatrix} = \begin{bmatrix} [2T_a(3)+x_a(0)]\cdot\sin(8\alpha) \\ [2T_a(2)+x_a(0)]\cdot\sin(10\alpha) \\ [2T_a(1)+x_a(0)]\cdot\sin(12\alpha) \end{bmatrix} \tag{26}$$

$$\begin{bmatrix} Y(6) \\ Y(4) \\ Y(2) \end{bmatrix} = \begin{bmatrix} [2T_b(4)+x_b(0)]\cdot\cos(6\alpha) \\ [2T_b(5)+x_b(0)]\cdot\cos(4\alpha) \\ [2T_b(6)+x_b(0)]\cdot\cos(2\alpha) \end{bmatrix} \tag{27}$$

and

$$Y(0) = x_b(0)+2\sum_{i=1}^{3}[x_b(\psi(i))+x_b(\phi(i))]. \tag{28}$$

### B. Unified Systolic Algorithm for the IDST/IDCT

The input sequences $\{Y(i) : i = 1,\ldots,N\}$ for the IDST and $\{Y(i) : i = 0,\ldots,N-1\}$ for the IDCT are respectively defined as:

$$x(k)=\sum_{i=1}^{N} Y(i)\cdot\sin[(2k+1)i\alpha], \qquad \text{for } k=0,1,\ldots,N-1 \tag{29}$$

$$x(k)=\sum_{i=0}^{N-1} Y(i)\cdot\cos[(2k+1)i\alpha], \qquad \text{for } k=0,1,\ldots,N-1 \tag{30}$$

where $\alpha = \pi/2N$.

In order to efficiently unify the VLSI implementations for the IDST and IDCT, we will reformulate these in such a manner that similar computational structures could be obtained for the two transforms.

If the transform length is a prime number greater than 2, the two inverse transforms can be concurrently computed using the following computational equations [22]:

$$\begin{aligned} &x(k) \\ &=\begin{cases} A(k)+(-1)^k\cdot B(k)+Y(0), & \text{for IDCT} \\ A(k)-(-1)^k\cdot B(k)+(-1)^k\cdot Y(N), & \text{for IDST} \end{cases} \\ &x(N-k) \\ &=\begin{cases} A(k-1)+(-1)^k\cdot B(k-1)+Y(0), & \text{for IDCT} \\ -A(k-1)+(-1)^k\cdot B(k-1)-(-1)^k\cdot Y(N), & \text{for IDST,} \end{cases} \\ &\text{for } k=1,\ldots,\frac{(N-1)}{2}. \end{aligned} \tag{31, 32}$$

The two auxiliary sequences $\{A(i) : i = 0,\ldots,(N-1)/2\}$ and $\{B(i) : i = 0,\ldots,(N-1)/2\}$ used in (31) and (32) can be computed recursively as follows:

$$A(0) = \begin{cases} \sum_{k=1}^{(N-1)/2} Y_b(\psi(k)), & \text{for IDCT} \\ \sum_{k=1}^{(N-1)/2} Y_a(\psi(k)), & \text{for IDST} \end{cases} \tag{33}$$

$$\begin{aligned} A(k) &= \begin{cases} 2\cdot T_b(k)-A(k-1), & \text{for IDCT} \\ 2\cdot T_a(k)+A(k-1), & \text{for IDST,} \end{cases} \\ &\text{for } k=1,\ldots,\frac{(N-1)}{2} \end{aligned} \tag{34}$$

and

$$B(0) = \begin{cases} \sum_{k=1}^{(N-1)/2} Y_a(\phi(k)) & \text{for IDCT} \\ \sum_{k=1}^{(N-1)/2} Y_b(\phi(k)), & \text{for IDST} \end{cases} \tag{35}$$

$$\begin{aligned} B(k) &= \begin{cases} 2\cdot T_a(k)+B(k-1), & \text{for IDCT} \\ 2\cdot T_b(k)-B(k-1), & \text{for IDST,} \end{cases} \\ &\text{for } k=1,\ldots,\frac{(N-1)}{2} \end{aligned} \tag{36}$$

where we have used the following auxiliary input sequences:

$$Y_a(k) = Y(\zeta(k))\cdot\sin(2k\alpha) \tag{37}$$

$$Y_b(k) = Y(\zeta(k))\cdot\cos(2k\alpha). \tag{38}$$

In (34) and (36), we have used two new auxiliary sequences $\{T_a(k): k=1,\ldots,(N-1)/2\}$ and $\{T_b(k): k=1,\ldots,(N-1)/2\}$ that can be computed in parallel as follows:

$$T_b(\psi(k)) = \begin{cases} \sum_{i=1}^{(N-1)/2} Y_b(\psi(i-k))\cdot\cos[\psi(i)\cdot 4\alpha], & \text{for IDCT} \\ \sum_{i=1}^{(N-1)/2} Y_b(\phi(i-k))\cdot\cos[\psi(i)\cdot 4\alpha], & \text{for IDST} \end{cases} \tag{39}$$

$$T_a(\psi(k)) = \begin{cases} \sum_{i=1}^{(N-1)/2} Y_a(\phi(i-k))\cdot\cos[\psi(i)\cdot 4\alpha], & \text{for IDCT} \\ \sum_{i=1}^{(N-1)/2} Y_a(\psi(i-k))\cdot\cos[\psi(i)\cdot 4\alpha], & \text{for IDST,} \end{cases}$$
$$\text{for } k=1,\ldots,\frac{(N-1)}{2}. \tag{40}$$

As can be seen from (39), the relations used to compute the auxiliary sequence $\{T_b(\psi(k)): k=1,\ldots,(N-1)/2\}$ for the IDCT and IDST represent cyclic convolutions having similar forms and the same length. A similar statement holds for $\{T_a(\psi(k)) : k=1,\ldots,(N-1)/2\}$. Also, the relations used in (39) and (40) have similar structures and the same length as those used in (14) and (15) for the forward DCT/DST, only the input sequences are different. Moreover, using the same considerations as those used for the forward DCT/DST, it can be shown from (39) and (40) that one operand in each multiplier is fixed and is the same, for any given $k$.

*Example 2:* In order to illustrate the special features of the proposed unified algorithm for IDCT/IDST, we now consider the case, where the length $N=7$ and the primitive root is $g=3$.

For IDCT, the two cyclic convolutions given by (39) and (40) take the form

$$\begin{bmatrix} T_a(3) \\ T_a(2) \\ T_a(1) \end{bmatrix} = \begin{bmatrix} Y_a(6) & Y_a(4) & Y_a(5) \\ Y_a(5) & Y_a(6) & Y_a(4) \\ Y_a(4) & Y_a(5) & Y_a(6) \end{bmatrix} \cdot \begin{bmatrix} \cos(12\alpha) \\ \cos(8\alpha) \\ \cos(4\alpha) \end{bmatrix} \tag{41}$$

$$\begin{bmatrix} T_b(3) \\ T_b(2) \\ T_b(1) \end{bmatrix} = \begin{bmatrix} Y_b(1) & Y_b(3) & Y_b(2) \\ Y_b(2) & Y_b(1) & Y_b(3) \\ Y_b(3) & Y_b(2) & Y_b(1) \end{bmatrix} \cdot \begin{bmatrix} \cos(12\alpha) \\ \cos(8\alpha) \\ \cos(4\alpha) \end{bmatrix}. \tag{42}$$

For IDST, the cyclic convolution structures given by (39) and (40) can be computed as:

$$\begin{bmatrix} T_a(3) \\ T_a(2) \\ T_a(1) \end{bmatrix} = \begin{bmatrix} Y_a(1) & Y_a(3) & Y_a(2) \\ Y_a(2) & Y_a(1) & Y_a(3) \\ Y_a(3) & Y_a(2) & Y_a(1) \end{bmatrix} \cdot \begin{bmatrix} \cos(12\alpha) \\ \cos(8\alpha) \\ \cos(4\alpha) \end{bmatrix} \tag{43}$$

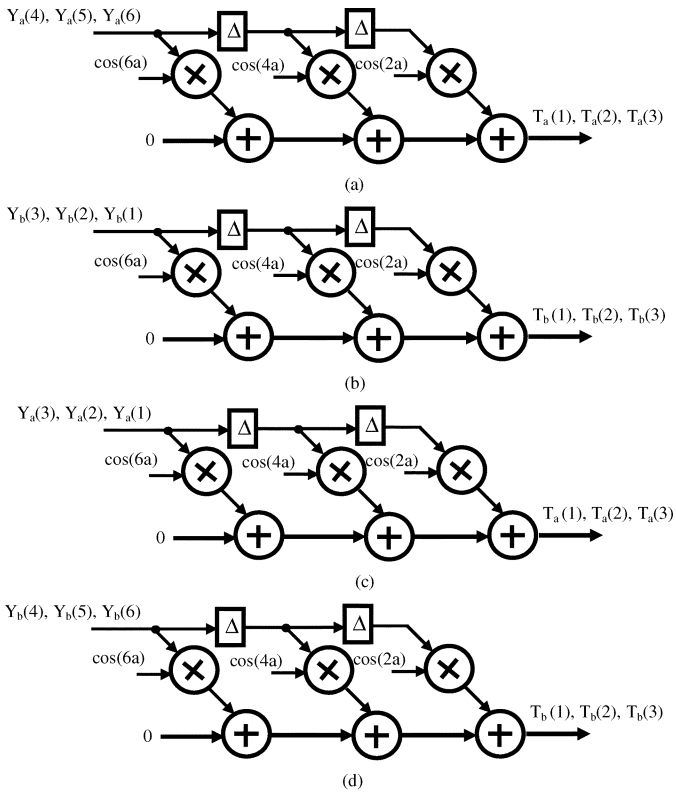
$$\begin{bmatrix} T_b(3) \\ T_b(2) \\ T_b(1) \end{bmatrix} = \begin{bmatrix} Y_b(6) & Y_b(4) & Y_b(5) \\ Y_b(5) & Y_b(6) & Y_b(4) \\ Y_b(4) & Y_b(5) & Y_b(6) \end{bmatrix} \cdot \begin{bmatrix} \cos(12\alpha) \\ \cos(8\alpha) \\ \cos(4\alpha) \end{bmatrix} \tag{44}$$

where

$$Y_a(k) = Y(\zeta(k))\cdot\sin(2k\alpha), \tag{45}$$

$$Y_b(k) = Y(\zeta(k))\cdot\cos(2k\alpha), \qquad k=1,2,\ldots,6. \tag{46}$$

Comparing (41)–(44), it is seen that they all have the same structure and length. As in the case of the forward DCT/DST unified algorithm, for the IDCT/IDST case also the operands $\cos(k\cdot 2a)$ are fixed and are the same, for any given $k$, as is seen from Fig. 2. This feature allows an efficient use of the memory-based implementation techniques. Moreover, due to the fact that the fixed operands $\cos(k\cdot 2a)$ used in the computational relations to compute the sequences $\{T_a(\psi(k)): k=1,\ldots,(N-1)/2\}$ and $\{T_b(\psi(k)): k=1,\ldots,(N-1)/2\}$ for IDCT or IDST are respectively the same in any given multiplier, the contents of the ROMs used to implement the multipliers of the computational structures are respectively the same, for any given multiplier.




**Note.** The input sequences have been extended circularly, and a = $2\alpha$

Fig. 2. SFGs of the computational core of the IDCT/IDST algorithm for Example 2. (a) SFG of (41) representing the IDCT part of (39). (b) SFG of (42) representing the IDCT part of (40), (c) SFG of (43) representing the IDST part of (40). (d) SFG of (44) representing the IDST part of (39).

Finally, the output sequence for the IDCT and IDST can be computed in parallel using (31) and (32) as

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \begin{bmatrix} A(0)+Y(0) \\ A(1)+Y(0) \\ A(2)+Y(0) \\ A(3)+Y(0) \end{bmatrix} + \begin{bmatrix} B(0) \\ -B(1) \\ B(2) \\ -B(3) \end{bmatrix} \tag{47}$$

$$\begin{bmatrix} x(6) \\ x(5) \\ x(4) \end{bmatrix} = \begin{bmatrix} A(0)+Y(0) \\ A(1)+Y(0) \\ A(2)+Y(0) \end{bmatrix} + \begin{bmatrix} -B(0) \\ B(1) \\ -B(2) \end{bmatrix} \tag{48}$$

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \begin{bmatrix} A(0)+Y(7) \\ A(1)-Y(7) \\ A(2)+Y(7) \\ A(3)-Y(7) \end{bmatrix} - \begin{bmatrix} B(0) \\ -B(1) \\ B(2) \\ -B(3) \end{bmatrix} \tag{49}$$

$$\begin{bmatrix} x(6) \\ x(5) \\ x(4) \end{bmatrix} = - \begin{bmatrix} A(0)-Y(7) \\ A(1)+Y(7) \\ A(2)-Y(7) \end{bmatrix} + \begin{bmatrix} -B(0) \\ B(1) \\ -B(2) \end{bmatrix} \tag{50}$$

respectively.

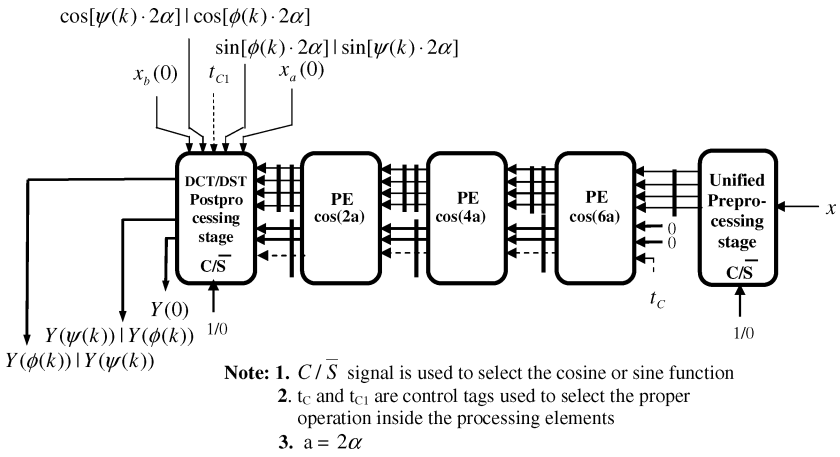



Fig. 3. Unified VLSI architecture for DCT/DST of length $N = 7$.

Comparing (41)–(44) that represent the computational core of the IDCT/IDST with (21) and (22) that were used to implement the forward DCT/DST, we see that they all have similar structures and the same length, with the fixed operand $\cos(k \cdot 2a)$ being the same in the corresponding multipliers in the four algorithms. Only the input and output sequences are different and nothing else. This special feature of the proposed algorithms leads to an efficient unified VLSI array that implements all the four transforms, as shown in the next section.

## III. Memory-Based Unified Architecture for DCT/DST/IDCT/IDST

In this section, we present a memory-based unified VLSI architecture that implements the systolic algorithms for DCT, DST, IDCT, and IDST, presented in the previous section. For the sake of simplicity of the discussion, we confine ourselves to the particular case of $N = 7$ and $g = 3$. However, the same ideas can be extended to the general case, where $N$ is any prime number. We first present a unified hardware architecture for the forward DCT/DST, followed by that for the IDCT/IDST, and finally present the design considerations that have been used to efficiently unify the architectures for all these four transforms.

### A. Unified Hardware Architecture for DCT/DST

Based on the unified algorithm for the DCT/DST presented in Section II, and employing the data-dependence graph-based design procedure [28] and the tag control scheme [29], two linear systolic arrays can be obtained for (21) and (22). From (21) and (22), it can be seen that the same kernel sequence $\cos(k \cdot 2a)$ is used to implement the two cyclic convolution structures, and each coefficient is fixed and allocated to one of the $(N-1)/2$ PEs. A unified VLSI architecture for implementing both the DCT and DST is shown in Fig. 3, when $N = 7$. It is seen that the central core consisting of the 3 PEs is the same for both DCT and DST, and only small differences exist in the architectures for the pre-processing (post-processing) stages depending on whether the unified architecture is used for the DCT or DST. This will be discussed later on in this section. The functionality of a typical

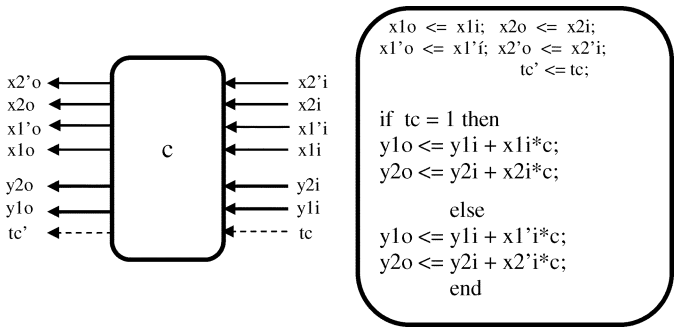



Fig. 4. Functionality of a PE PE in Fig. 3.

PE is shown in Fig. 4, where it is seen that there are 2 multipliers in each PE. One of the operands in both the multipliers is fixed and is the same. This feature allows us to introduce a dual-port ROM-based implementation technique to replace the two multipliers in each PE, as shown in Fig. 5. We will briefly discuss the characteristics of the dual-port ROM-based realization. As seen from Fig. 5(a), since the ROM tables necessary to implement the two multipliers in each PE are the same, we can use only one dual-port ROM to implement both the multipliers in each PE with a memory of only $2^L$ words, instead of two such ROMs. This results in a substantial reduction in the hardware cost. Using the partial sums technique [20] for such a realization, we can further reduce the ROM size to $2^{(L/2+1)}$ words at a cost of two extra adders, as shown in Fig. 5(b). Thus, we can reduce the total ROM size to $((N-1)L/2)2^{(L/2+1)}$ bits to compute an $N$-point transform in $(N-1)/2$ cycles. Due to the fact that the hardware structure used to implement the transforms is a synchronous one, the control structure that is used to avoid conflicts in accessing the content of the shared memory can be significantly simplified, as shown in Fig. 6. Fig. 7(a) and (b), respectively, shows the unified systolic array functioning in the DCT and DST modes.

As mentioned earlier, only the pre-processing and post-processing stages used to generate the auxiliary input sequences $\{x_a(i) : i = 0, \ldots, (N-1)\}$ and $\{x_b(i) : i = 0, \ldots, (N-1)\}$, from the original input sequence and to convert the auxiliary

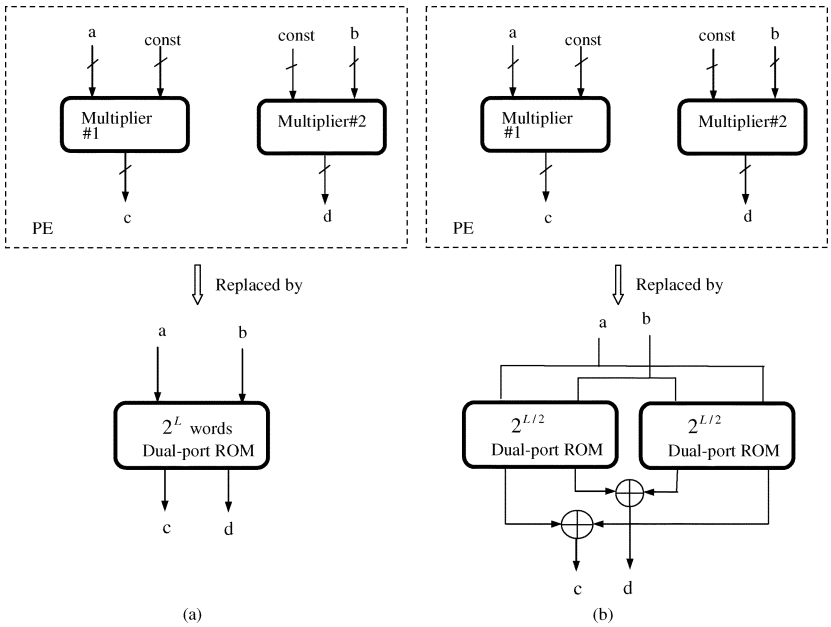



Fig. 5. Illustration of the principle of implementation of a dual-port ROM replacing a multiplier. (a) Direct replacement of multipliers with ROMs. (b) Using partial sums technique.

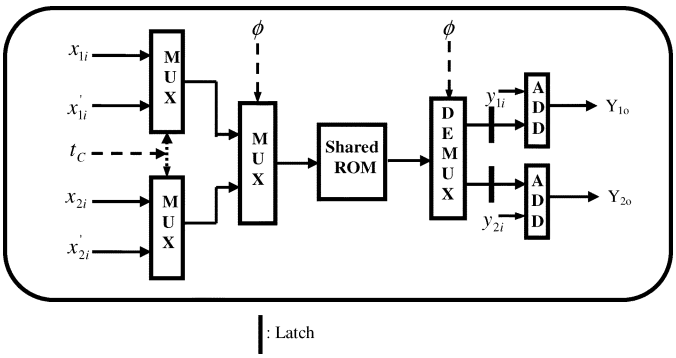



Fig. 6. Structure of the PE.

output sequences $\{T_a(k) : k = 1, \ldots, (N-1)/2\}$, and $\{T_b(k) : k = 1, \ldots, (N-1)/2\}$ into the desired output sequence, are different. There are only small differences in each of these stages when we switch from one transform to the other.

The structure of the unified pre-processing stage for the DCT/DST for the case of $N = 7$ is shown in Fig. 8(a). It is observed that we need two small circuits to change the sign of the elements of the input sequence. The structure of the permutation unit of the unified pre-processing stage for the DCT/DST is shown in Fig. 8(b). It allows the successive data blocks to be loaded into the array without any time delay in such a manner that the whole array can be fully pipelined with an average computation time of $(N-1)T/2$. As seen from Fig. 8(b), the elements of two successive sequences are shifted synchronously in the two shift registers of the permutation unit and then loaded in parallel into the latches, by turn. Then, they are appropriately selected by MUXs and sent to the output of the permutation unit in two groups, as shown in Fig. 8(b).

The structure of the unified post-processing stage for the DCT/DST architecture is shown in Fig. 9. It is seen from Fig. 9(b) that a switching circuit is introduced to switch the sequences $Y(\psi(k))$ and $Y(\phi(k))$, when we shift the computation from the DCT to DST, or vice-versa. A small circuit to change the sign of the output signal $y'_0$ and a MUX to choose the right sign, are also introduced, in order to change the sign of the output sample $\mathrm{Y}(0)$ for the DCT and $\mathrm{Y}(7)$ for the DST appropriately.

It is well known that the systolic arrays used to implement the DCT and DST have to minimize the number of I/O channels and their bandwidth in order to be used in practical real-time signal processing applications. They also have to avoid pre-loading of the data at the beginning of the computational cycle, and reduce to a minimum the number of boundary cells [30]. These problems are solved in our design in a straightforward way, due to the specific structure of the cyclic convolution structure, where the elements located on a given diagonal line in the matrices of (21) and (22) are the same. This reduction in the I/O cost is achieved, since each data element introduced into the systolic array is used in all the PEs. Also, all the input and output channels are placed at one of the two extreme ends of the array and their number is independent of the transform length $N$. These are appealing features for a VLSI implementation. In contrast to this simple way of solving the I/O problems, in [24] the transform kernels are generated recursively into the array leading to a significant increase of the hardware complexity.

There is no computational overhead to compute both the transforms using the same VLSI structure, but some minor hardware modifications are necessary when we switch the computation from one transform to the other.

### *B. Unified Hardware Architecture for IDCT/IDST*

Based on (39) and (40) it can be seen, as in the case of the systolic array for the forward DCT/DST already presented in

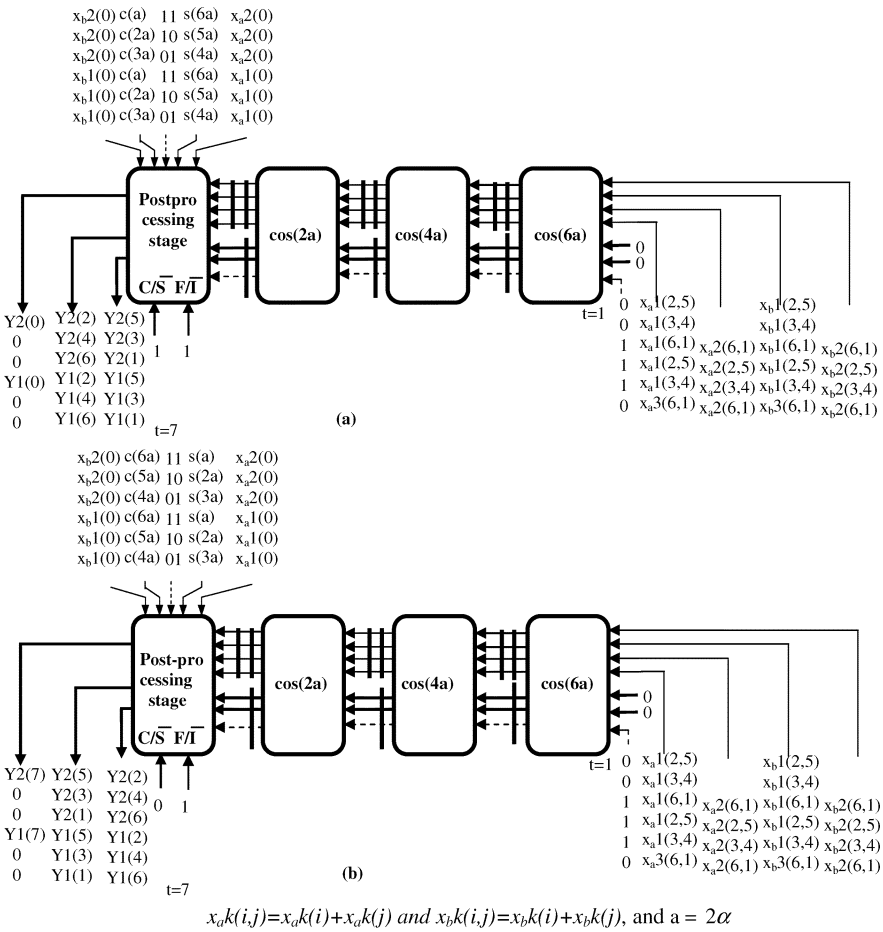



Fig. 7. Systolic array functioning in (a) DCT mode and (b) DST mode.

Fig. 3, that the main core is the same for the both inverse transforms. However, in the pre- and post-processing stages, there are some minor changes in the hardware structure when we switch from IDCT to IDST. For the case of $N = 7$, the unified systolic array for the IDCT and IDST is shown in Fig. 10. The PEs in the systolic array that forms the hardware core used to implement the two cycle convolution defined by (41) and (42) for IDCT, and (43) and (44) for IDST, are the same as those used for the DCT/DST. The function and structure of the PEs have already been shown in Figs. 4 and 6, respectively.

The structure of the pre-processing stage of the unified IDCT/IDST VLSI array is shown in Fig. 11. It is seen that there are no modifications in the hardware structure, when we switch from IDCT to IDST. We have only to replace the input sequences $\sin[\psi(k) \cdot 2\alpha]$ and $\sin[\phi(k) \cdot 2\alpha]$ by $\cos[\psi(k) \cdot 2\alpha]$ and $\cos[\phi(k) \cdot 2\alpha]$, respectively.

The structure of the post-processing stage of the unified IDCT/IDST is shown in Fig. 12. It is seen that some small circuits are introduced to change the sign of the operands in the adders to change from IDCT to IDST. They are as follows.

1 ) Two circuits, as shown in Fig. 12(a), for the computation of $x(0)$ and $x(6)$.
2 ) Three circuits, as shown in Fig. 12(b), for the computation of $x(i)$ and $x(N-1-i)$.

### C. Unification of the DCT/DST/IDCT/IDST

Due to the fact that all four algorithms have the same computational core given by (14) and (15) for the forward DCT/DST, and (41) and (42) for the inverse DCT/DST, we can efficiently unify the two architectures to obtain a single unified linear VLSI array. If we compare the VLSI architectures for the DCT/DST presented in Fig. 7 with those of the IDCT/IDST shown in Fig. 10, we see that only the pre-processing and post-processing stages are different, but they represent a small percentage of the overall hardware complexity and their complexity does not depend on the transform length, except for the number of the shift registers used in the pre-processing and post-processing stages to appropriately permute the data sequences. The unified VLSI architecture for the DCT/DST/IDCT/IDST is shown in Fig. 13. It is seen that a common permutation unit is introduced at the end of the unified systolic array in order to obtain the samples of the output sequence in the natural order, since the outputs of the post-processing stages for the DCT/DST (Fig. 7) and IDCT/IDST (Fig. 10) are in a permuted order.

As the two pipelined multipliers used in the pre-processing stage for IDCT/IDST and in post-processing stage for DCT/DST are not used at the same time, we can further reduce the hardware complexity by unifying the pre- and post-processing stages in one processing stage. Thus, we can share

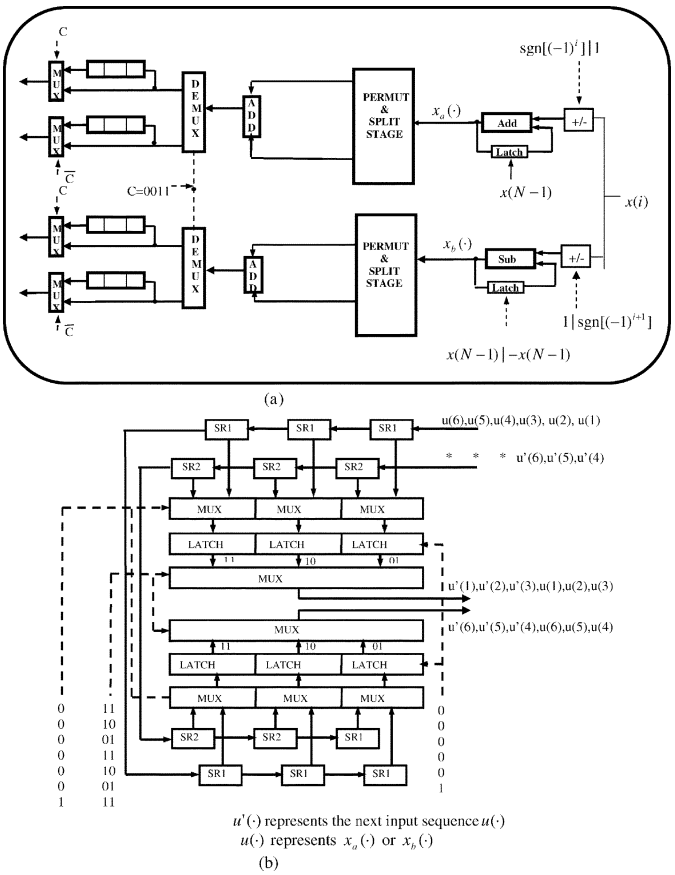



Fig. 8. (a) Structure of the unified pre-processing stage for DCT/DST of length $N = 7$. (b) Permutation block from the pre-processing stage of the DCT/DST VLSI array.

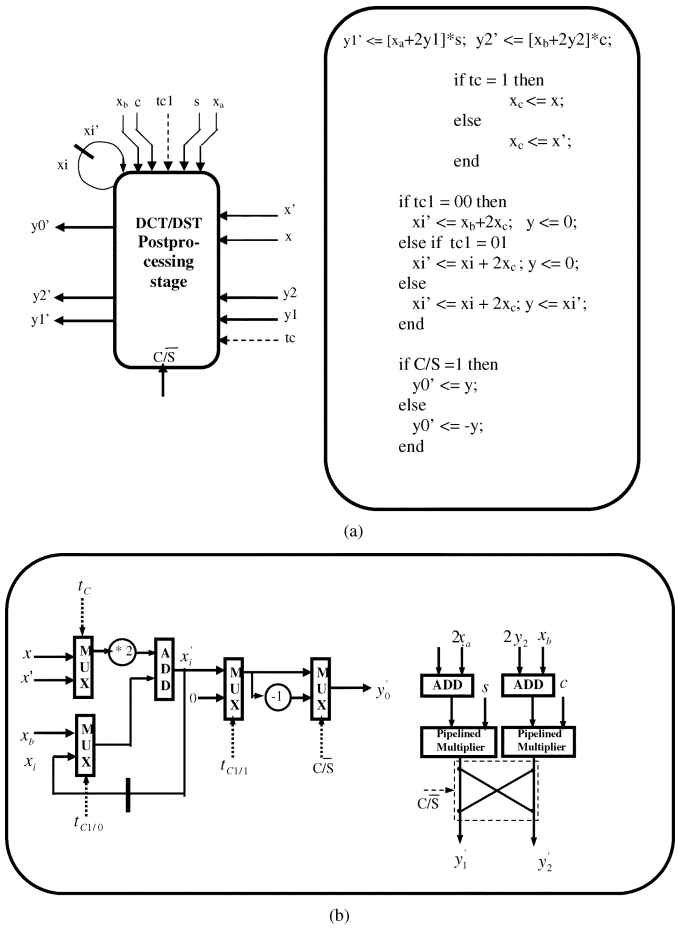



Fig. 9. Functionality and structure of the unified post-processing stage for the DCT/DST of length $N = 7$. (a) Functionality (b) Structure.

the two multipliers at a cost of some extra MUXs in a manner similar to the one described in [25]. The array design has been verified through computer simulations using VHDL hardware description language in a Synopsys environment.

In summary, the proposed unified architecture for DCT/DST/IDCT/IDST has several distinctive features. First, the input and output data are loaded and drained out from the I/O channels placed at the two extreme ends of the array and the input data volume to the PEs has been significantly reduced. Second, the proposed unified algorithm allows an efficient application of the memory-based implementation techniques in a unified way such that all the multipliers can be efficiently implemented using small ROMs and adders to attain a low hardware cost and a high processing speed. Third, due to its regularity, modularity, simplicity, and local connections, the proposed unified systolic array is well suited for VLSI implementation.

## IV. Performance Comparison

The hardware complexity of our design is given by two pipelined multipliers, all in the pre- and post-processing stages, $(2N + 3)$ adders and $((N-1)L/2)2^{L/2+1}$ ROM bits. The average computation time is $(N-1)T/2$, where $T$ is the cycle time, and the throughput is $2/(N-1)$. By using pipelined multipliers, the cycle time $T$ is reduced to $T_{\text{Mem}} + T_{\text{add}}$, where $T_{\text{Mem}}$ is the accessing time to the dual-port memory and $T_{\text{add}}$ is the propagation time through one adder. The number of I/O channels is $(7L+1)$ and is independent of the transform length. The advantages of the proposed design are more evident when the transform length $N$ is large.

The hardware cost and the speed performance of our design together with those of some recently reported unified VLSI designs are summarized in Tables I and II. Compared to the systolic arrays based on the Clenshaw's recurrence formula, given in [11] and [12], in our design the throughput is doubled with a shorter cycle time, and it has a comparable hardware complexity, but with a significantly lower I/O cost ($(N+1)L$ as compared to $7L+1$). Compared to the hardware complexity of [23] given by $2(N-1)$ adders and $N$ multipliers, we see that the hardware complexity of our design is lower for the usual values of $L$. For example, if we choose $N = 17$ and $L = 8$, only 2048 ROM bits are necessary to replace the 17 multipliers in the different PEs of the unified array (excepting the two pipelined multipliers). Also, the throughput is slightly better and the cycle time significantly shorter. The I/O cost in [23] is significantly greater, especially for larger values of $N$ as seen from Table I. Moreover, in order to provide the computing flexibility, the systolic arrays in [11], [12] and [23]

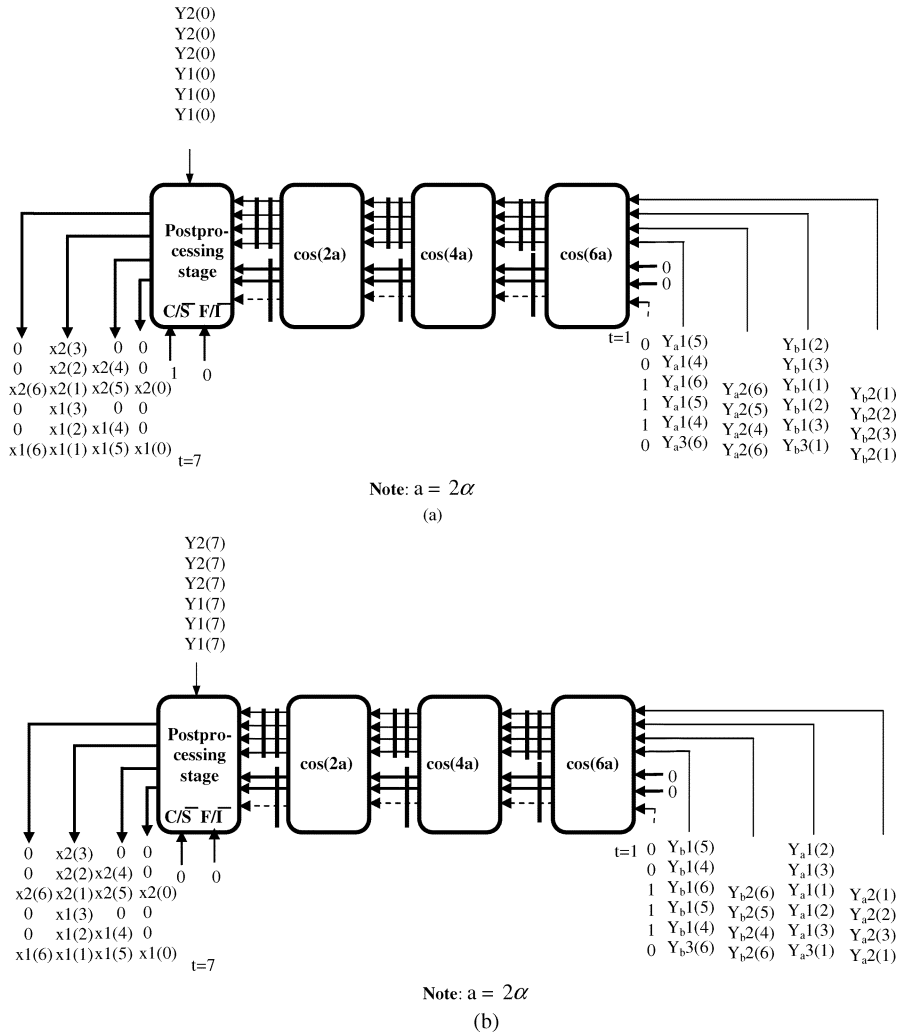



Fig. 10. Architecture and activity of the unified systolic array of the IDCT/IDST of length 7. (a) Unified systolic array in the IDCT mode. (b) Unified systolic array in the IDST mode.

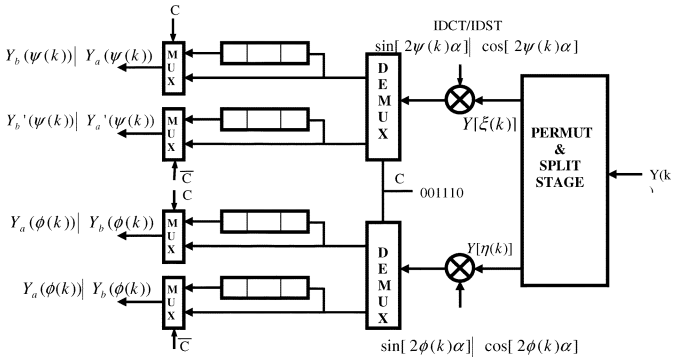



Fig. 11. Structure of the pre-processing stage of the unified IDCT/IDST VLSI array.

feed different transform coefficients into the array using an additional control module and with a high I/O cost.

The hardware complexity in [25], given by one multiplier and $NL(2^{N/2}+2)$ ROM bits is larger than that of 2 multipliers and $((N-1)L/2)(2^{L/2+1})$ ROM bits needed in our design, if the transform length $N$ is large. Also, in the present design the throughput is doubled with a shorter cycle time.

Compared to a similar memory-based design given in [14], even though it is only for the DCT and does not allow for an efficient unified design, in our design the throughput is doubled with about the same hardware complexity and I/O cost, while maintaining all the benefits of the implementations based on cyclic convolutions such as modularity, regularity, regular and local connections, and a simple control structure.

## V. Conclusion

In this paper, we have presented a unified architecture for DCT/DST/IDCT/IDST by appropriately formulating the four transforms into cyclic convolution structures in a unified manner. By using such computational structures, the high computing speed achieved in the systolic array that forms the main core of the unified structure is no longer limited by its I/O bandwidth. This new formulation has allowed an efficient memory-based systolic array implementation of the

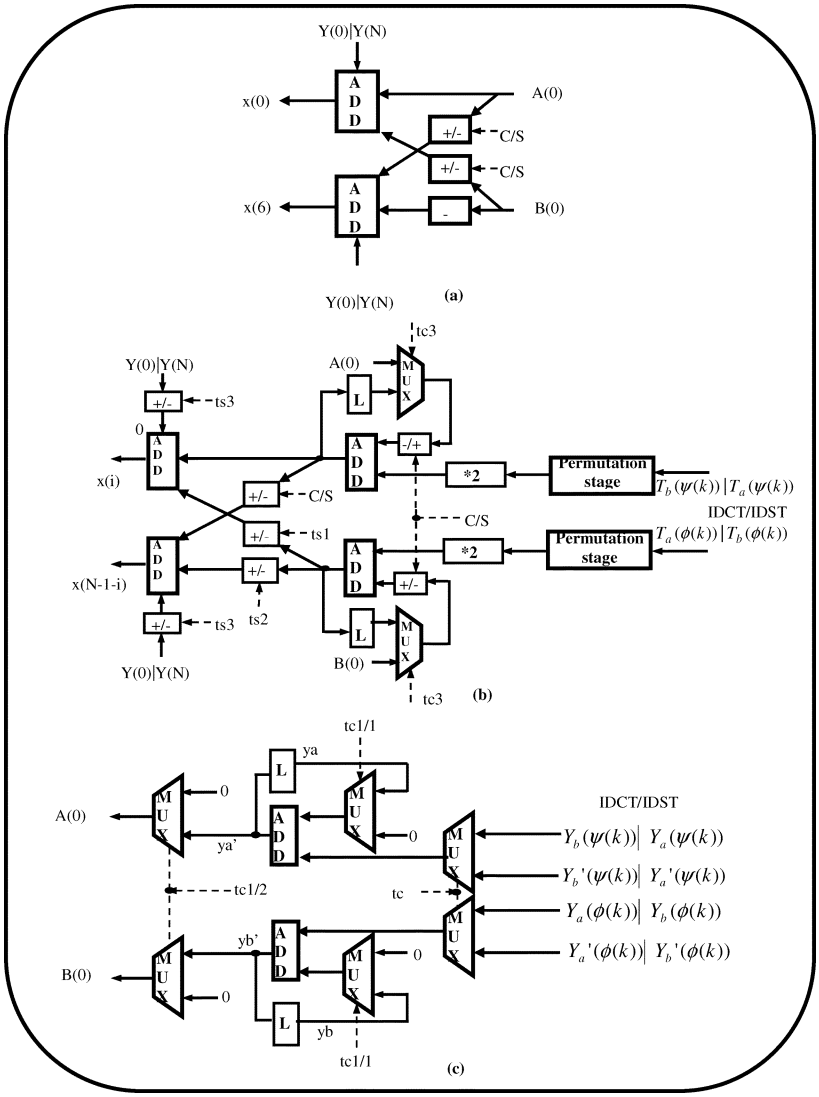



Fig. 12. Structure of the post-processing stage of the unified IDCT/IDST VLSI array.

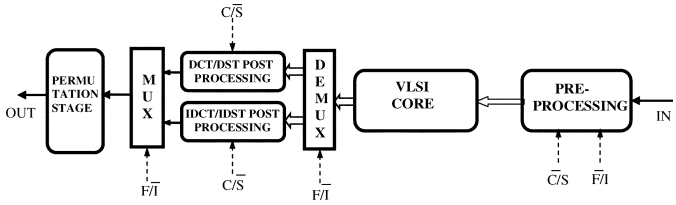



Fig. 13. Overall unified VLSI architecture for DCT/DST/IDCT/IDST.

unified architecture using dual-port ROMs and appropriate hardware-sharing techniques.

The unified architecture comprises a core part, common to all the transforms, in which all the cyclic convolution structures are implemented, and pre- and post-processing stages. Since most of the computations are performed in the core part, the core has been optimized from the point of view of hardware complexity, speed, and I/O cost. The pre- and post-processing stages have also been unified for all the four transforms to a large extent. However, the changes needed to switch from one transform to another in these pre- and post-processing stages contribute very little to the overall hardware complexity of the unified architecture, when the transform length is large.

The performance of the unified architecture proposed in this paper has been compared to some of the existing ones in terms of the hardware complexity, speed, I/O costs and other features. It has been shown that the present design provides an improved performance over that of the existing memory-based unified structures for large values of the transform length, as well as over the systolic array-based solutions for the usual values of the transform length.

Due to its regular and modular topology with local connections, specific to cyclic convolution-based systolic arrays, and its highly modular and regular implementation style specific to memory-based techniques, this architecture is well suited for VLSI implementation. It substantially improves the performances of the DA-based unified structures for larger values of the transform length and of the systolic array- based solutions for usual values of the word-length $L$.

TABLE I
COMPARISON OF HARDWARE COMPLEXITY OF VARIOUS DCT AND IDCT DESIGNS

| Design | Multipliers | Adders | ROMs Bits | No. MUXs \| No. Inputs | No. of I/O Channels | Control Complexity | Scalability |
|---|---|---|---|---|---|---|---|
| Fang and Wu[11,12] | N/2+4 | N+3 | | | (N+1)L | Small | Excellent |
| Pan and Park [23] | N | 2N-2 | | | 2(N+1)L | Small | Low |
| Guo and Li [25] | 1 | 3N | $NL\cdot(2^{N/2}+2)$ | | 2L | Small | Excellent |
| Guo et al.[26] | 1 | 2N | $NL\cdot 2^{N}$ | | 2L | Small | Excellent |
| Proposed | 2 | 2N+3 | $\frac{(N-1)L}{2}\cdot 2^{(L/2+1)}$ | 28+(N-1) \| 26+5(N-1)/2 | 7L+1 | Small | Excellent |

TABLE II
COMPARISON OF SPEED AND OTHER FEATURES OF VARIOUS DCT AND IDCT DESIGNS

| Design | Throughput | Cycle Time | $N_{in}\cdot B$ | Pipelining | Two-level pipelining |
|---|---|---|---|---|---|
| Fang and Wu [11,12] | 1/N | $T_{Mul}+2T_{Add}$ | O(N) | Yes | Difficult |
| Pan and Park [23] | 2/N | $T_{Mul}+T_{Add}$ | $O(N^2)$ | Yes | Easy |
| Guo and Li [25] | 1/N | $T_{Mul}+T_{Add}$ | O(N) | Difficult | No |
| Guo et al. [26] | 1/N | $T_{Mul}+T_{Add}$ | O(N) | Difficult | No |
| Proposed | 2/(N-1) | $T_{Mem}+T_{Add}$ | O(N) | Yes | Easy |

## REFERENCES


[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–94, Jan. 1974.

[2] H. B. Kekre and J. K. Solanki, "Comparative performance of various trigonometric unitary transforms for transform image coding," *Int. J. Electron.*, vol. 44, no. 3, pp. 305–315, 1978.

[3] A. K. Jain, "A fast Karhunen–Loeve transform for a class of random processes," *IEEE Trans. Comm.*, vol. COM-24, no. 10, pp. 1023–1029, Oct. 1976.

[4] ——, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[5] K. Rose, A. Heiman, and I. Dinstein, "DCT/DST alternate-transform of image coding," *IEEE Trans. Comm.*, vol. 38, no. 1, pp. 94–101, Jan. 1990.

[6] U. T. Koc and K. J. R. Liu, "Discrete cosine/sine transform based motion estimation," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, 1994, pp. 771–775.

[7] D. Sundararajan, M. O. Ahmad, and M. N. S. Swamy, "Vector computation of the discrete Fourier transform," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 45, no. 4, pp. 449–461, Apr. 1998.

[8] V. Britanak, "DCT/DST universal computational structure and its impact on VLSI design," in *Proc. IEEE DSP Workshop*, Hunt, TX, Oct. 15–18, 2000.

[9] L-P. Chau and W-C. Siu, "Direct formulation for the realization of discrete cosine transform using recursive filter structure," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 42, no. 1, pp. 50–52, Jan. 1995.

[10] J. F. Yang and C-P. Fang, "Compact recursive structures for discrete cosine transform," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 4, pp. 314–321, Apr. 2000.

[11] W. H. Fang and M. L. Wu, "An efficient unified systolic architecture for the computation of discrete trigonometric transforms," in *Proc. IEEE Symp. Circuits and Systems*, vol. 3, 1997, pp. 2092–2095.

[12] W. H. Fang and M-L. Wu, "Unified fully-pipelined implementations of one- and two-dimensional real discrete trigonometric transforms," *IEICE Trans. Fund. Electron. Commun. Comput. Sci.*, vol. E82-A, no. 10, pp. 2219–2230, Oct. 1999.

[13] C. M. Rader, "Discrete Fourier transform when the number of data samples is prime," *Proc. IEEE*, vol. 56, no. 6, pp. 1107–1108, Jun. 1968.

[14] J. Guo, C. M. Liu, and C. W. Jen, "The efficient memory-based VLSI array design for DFT and DCT," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 39, no. 10, pp. 436–442, Oct. 1992.

[15] Y. H. Chan and W. C. Siu, "On the realization of discrete cosine transform using the distributed arithmetic," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 39, no. 9, pp. 705–712, Sep. 1992.

[16] D. F. Chiper, M. N. S. Swamy, M. O. Ahmad, and T. Stouraitis, "A systolic array architecture for the discrete sine transform," *IEEE Trans. Signal Process.*, vol. 50, no. 9, pp. 2347–2353, Sep. 2002.

[17] S. A. White, "Applications of the distributed arithmetic to digital signal processing: A tutorial review," *IEEE ASSP Mag.*, vol. 6, no. 3, pp. 5–19, Jul. 1989.

[18] H. T. Kung, "Why systolic architectures," *IEEE Comp.*, vol. 15, pp. 37–46, Jan. 1982.

[19] S. Yu and E. E. Swartzlander, "DCT implementation with distributed arithmetic," *IEEE Trans. Comp.*, vol. 50, no. 9, pp. 985–991, Sep. 2001.

[20] M. T. Sun *et al.*, "VLSI implementation of a 16 × 16 discrete cosine transform," *IEEE Trans. Circuits Syst.*, vol. 36, no. 4, pp. 610–617, Apr. 1989.

[21] S. Uramoto *et al.*, "A 100 MHz 2-D discrete cosine transform core processor," *IEEE J. Solid-State Circuits*, vol. 27, no. 4, pp. 482–498, Apr. 1992.

[22] D. F. Chiper, "A systolic array algorithm for an efficient unified memory-based implementation of the inverse discrete cosine transform," in *Proc. IEEE Conf. Image Processing*, Kobe, Oct. 1999, pp. 764–768.

[23] S. B. Pan and R-H. Park, "Unified systolic array for computation of DCT/DST/DHT," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 2, pp. 413–419, Apr. 1997.

[24] L. W. Chang and W. C. Wu, "A unified systolic array for discrete cosine and sine transforms," *IEEE Trans. Signal Process.*, vol. 39, no. 1, pp. 192–194, Jan. 1991.

[25] J-I. Guo and C-C. Li, "A generalized architecture for the one-dimensional discrete cosine and sine transforms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 7, pp. 874–881, Jul. 2001.

[26] J. Guo, C. Chen, and C-W. Jen, "Unified array architecture for DCT/DST and their inverses," *Electron. Lett.*, vol. 31, no. 21, pp. 1811–1812, 1995.

[27] J. G. McWhirter, "Algorithmic engineering in adaptive signal processing," *Proc. Inst. Elect. Eng. F*, vol. 139, no. 3, pp. 226–232, Jun. 1992.

[28] S. Y. Kung, *VLSI Array Processors*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[29] C. W. Jen and H. Y. Hsu, "The design of systolic arrays with tag input," in *Proc. IEEE Int. Symp. Circuits and Systems ISCAS'88*, Finland, 1988, pp. 2263–2266.

[30] N. R. Murthy and M. N. S. Swamy, "On the real-time computation of DFT and DCT through systolic arrays," *IEEE Trans. Signal Process*, vol. 42, no. 4, pp. 988–991, Apr. 1994.

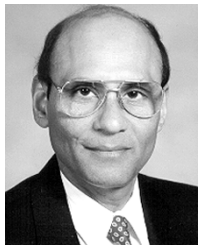
**Doru Florin Chiper** was born in Iasi, Romania, in 1956. He received the diploma and the Ph.D. degrees in electronics and telecommunications from the Technical University "Gh. Asachi," Iasi, Romania, in 1981 and 1996, respectively.

In September 1999, he was with Institute National pour Telecommunication, Evry, France, as a Visiting Researcher, and from October 1999 until April 2001, he was with Concordia University, Montreal, QC, Canada, as a Postdoctoral Fellow. He is currently an Associate Professor with the Technical University "Gh. Asachi." His research interests include digital signal processing (DSP), adaptive signal processing, blind equalization/identification, fast computational algorithms, parallel and very large-scale integration algorithms, and architectures for communications and digital signal processing and high-level DSP design

**M. N. S. Swamy** (S'59–M'62–SM'74–F'80–LF'01) received the B.Sc. (Hons.) degree in mathematics from Mysore University, Mysore, India, in 1954, the Diploma in electrical communication engineering from the Indian Institute of Science, Bangalore, India, in 1957, and the M.Sc. and Ph. D. degrees in electrical engineering from the University of Saskatchewan, Saskatoon, SK, Canada, in 1960 and 1963, respectively. In August 2001 he was awarded a Doctor of Science in Engineering (Honoris Causa) by Ansted University "In recognition of his exemplary contributions to the research in Electrical and Computer Engineering and to Engineering Education, as well as his dedication to the promotion of Signal Processing and Communications Applications."

He is presently a Research Professor and the Director of the Center for Signal Processing and Communications in the Department of Electrical and Computer Engineering at Concordia University, Montreal, QC, Canada, where he served as the Chair of the Department of Electrical Engineering from 1970 to 1977, and Dean of Engineering and Computer Science from 1977 to 1993. Since July 2001, he holds the Concordia Chair (Tier I) in Signal Processing. He has also taught in the Electrical Engineering Department of the Technical University of Nova Scotia, Halifax, NS, Canada, and the University of Calgary, Calgary, AB, Canada, as well as in the Department of Mathematics at the University of Saskatchewan. He has published extensively in the areas of number theory, circuits, systems and signal processing, and holds four patents. He is the co-author of two book chapters and three books: *Graphs, Networks and Algorithms* (New York: Wiley, 1981), *Graphs: Theory and Algorithms* (New York: Wiley, 1992), and *Switched Capacitor Filters: Theory, Analysis and Design* (London, U.K.: Prentice-Hall International UK Ltd., 1995). A Russian translation of the first book was published by Mir Publishers, Moscow, U.S.S.R., in 1984, while a Chinese version was published by the Education Press, Beijing, in 1987. He is a member of Micronet, a National Network of Centers of Excellence in Canada, and also its coordinator for Concordia University.

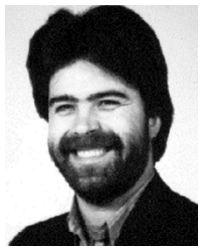
Dr. Swamy is the Past President of the IEEE Circuits and Systems (CAS) Society. He has served the IEEE CAS Society in various capacities such as Vice President (Publications) during 2001–2002 and as Vice-President in 1976, Editor-in-Chief of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: FUNDAMENTAL THEORY AND SYSTEMS during 1999–2001, Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS during 1985–1987, Program Chair for the 1973 IEEE CAS Symposium, General Chair for the 1984 IEEE CAS Symposium, Vice-Chair for the 1999 IEEE CAS Symposium and a member of the Board of Governors of the CAS Society. He is the recipient of many IEEE-CAS Society awards including the Education Award in 2000, Golden Jubilee Medal in 2000, and the 1986 Guillemin–Cauer Best Paper Award. He is a Fellow of the Institute of Electrical Engineers (U.K.), the Engineering Institute of Canada, the Institution of Engineers (India), and the Institution of Electronic and Telecommunication Engineers (India).



**M. Omair Ahmad** (S'59–M'78–SM'83–F'01) received the B. Eng. degree from Sir George Williams University, Montreal, QC, Canada, and the Ph.D. degree from Concordia university, Montreal, QC, Canada, both in electrical engineering.

From 1978 to 1979, he was a member of the Faculty of the New York University College, Buffalo. In September 1979, he joined the Faculty of Concordia University, where he was an Assistant Professor of Computer Science. Subsequently, he joined the Department of Electrical and Computer Engineering of the same university, where presently he is Professor and Chair of the Department. He is a Researcher in the Micronet National network of Centers of Excellence and was previously an Examiner of the Order of Engineers of Quebec. He has published extensively in the area of signal processing and holds three patents. His current research interest includes the area of multidimensional filter design, image and video signal processing, nonlinear signal processing, communication digital signal processing, artificial neural networks, and very large-scale integration circuits for signal processing.

Dr. Ahmad was an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: FUNDAMENTAL THEORY AND SYSTEMS from June 1999 to December 2001. He was the Local Arrangements Chairman of the 1984 IEEE International Symposium on Circuits and Systems (CAS). During 1988, he was a member of the Admission and Advancement Committee of the IEEE. Presently, he is the Chairman of the IEEE CAS Chapter (Montreal Section). He is a recipient of the Wighton Fellowship from the Sanford Fleming Foundation.



**Thanos Stouraitis** (SM'98) received the B.S. degree in physics and the M.S. degree in electronic automation, from the University of Athens, Athens, Greece, in 1979, and 1981, respectively, the M.Sc. degree in electrical and computer engineering from the University of Cincinnati, Cincinnati, OH, in 1983, and the Ph.D. degree in electrical engineering from the University of Florida, Gainesville, in 1986 (for which he received the Outstanding Ph.D. Dissertation Award).

He is a currently a Professor of Electrical and Computer engineering at the University of Patras, Patras, Greece, where he directs an international graduate studies program on digital signal processing systems. He has served on the faculty of The Ohio State University, Columbus, and has visited the University of Florida and Polytechnic University, Brooklyn, NY. His current research interests include signal and image processing systems, application-specific processor technology and design, computer arithmetic, and design and architecture of optimal digital systems. He has led several digital signal processing (DSP) processor design projects funded by the European Union, American organizations, and the Greek government and industry. He has authored or co-authored over 140 technical papers. He holds one patent on DSP processor design. He has authored several book chapters, the book *Digital Signal Processing* (Patras, Greece: Univ. of Patras Press, 1997) and co-authored the book *Digital Filter Design Software for the IBM PC* (New York: Marcel Dekker, 1987).

Dr. Stouraitis serves as a Regional Editor for Europe for the *Journal of Circuits, Systems and Computers*, an Editor-at-Large for Marcel Dekker Inc., has served as an Associate Editor for the *Journal of Circuits, Systems and Computers*, and as a consultant for various industries. He regularly reviews for the *IEE Proceedings E, F, G* and for conferences like IEEE ISCAS, ICASSP, VLSI, SiPS, Computer Arithmetic, Euro DAC, etc. He also reviews proposals for the National Science Foundation, the European Commission, and other agencies. Dr. Stouraitis has served as Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: ANALOG AND DIGITAL SIGNAL PROCESSING, for the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, and as Editor for the *IEEE Interactive Magazine*. He regularly reviews for the IEEE TRANSACTIONS ON SIGNAL PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, IEEE TRANSACTIONS ON COMPUTERS, and IEEE TRANSACTIONS ON EDUCATION. He has served as the Chair of the VLSI Systems and Applications (VSA) Technical Committee and as a member of the DSP and the Multimedia Technical Committees of the IEEE Circuits and Systems Society. He is a founder and the Chair of the IEEE Signal Processing chapter in Greece. He was the General Chair of the 1996 IEEE International Conference on Electronics, Circuits, and Systems (ICECS) and the Technical Program Chair of Eusipco '98 and ICECS '99. He has served as Chair or as a member of the Technical Program Committees of a multitude of IEEE Conferences, including ISCAS (Program Committee Track Chair). He is the General Chair of ISCAS 2006. He received the 2000 IEEE Circuits and Systems Society Guillemin–Cauer Award (best paper) for the paper "Multi-Function Architectures for RNS Processors."