BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Tomul LIX (LXIII), Fasc. 3, 2013
Secția
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ

A NOVEL SYSTOLIC ALGORITHM FOR 2-D DISCRETE SINE TRANSFORM

BY

DORU-FLORIN CHIPER[*]

“Gheorghe Asachi” Technical University of Iaşi
Faculty of Electronics, Telecommunications and Information Technology,

Received: December 7, 2012
Accepted for publication: February 25, 2013

Abstract. Using a new VLSI algorithm for 2-D discrete sine transform (DST) an efficient VLSI architecture with appealing topological features and high performances can be obtained. The new algorithm has a modular and regular computational structure and can be computed in parallel thus resulting a high throughput VLSI implementation. The proposed algorithm can be mapped into two linear systolic arrays having a high computing speed and low I/O cost with a small number of I/O channels placed at the two ends of the linear array. By combining two such linear systolic arrays we can obtain an efficient VLSI architecture for 2-D DST. The architecture that can be obtained has a highly regular and modular structure and local connections specific to the systolic array architectural paradigm.

Key words: discrete sine transform; systolic array; VLSI algorithm; VLSI architecture.

1. Introduction

The discrete cosine transform (DCT) and discrete sine transform (DST) (Ahmed *et al*., 1974; Jain, 1976, 1989) are important elements in some digital

[*] *e-mail*: chiper@etc.tuiasi.ro

signal processing applications. As is well known they are good approximations to the statistically-optimal Karhunen-Loeve transform (Jain, 1976, 1989). They can be also used in speech and image transform coding (Jain, 1989; Zhang *et al*., 2007), DCT based subband decomposition in speech and image compression (Chen, 2007), or video transcoding (Fung & Siu, 2006). Also there are some other important applications as: block filtering (Martucci & R.Mersereau, 1993), feature extraction (Jadhav & Holambe, 2008), digital signal interpolation (Wang *et al*., 1993), image resizing (Park & Park, 2006), transform-adaptive filtering (Pei & Tseng, 1996; Mayyas, 2005) and filter banks (Bergen, 2008).

It is well known that for high correlation images DCT yields better results and for low correlation ones DST yields lower bit rates. The DCT and DST represent a good approximation of the statistically optimal Karhunen-Loeve transform.

The transform length used in transform coding is 8 or 16. But we can reduce blocks artifacts using a prime transform length of 11 or 17 and an overlapping technique. In some applications a prime factor is a more suitable transform length than a power of two (Tatsaki *et al*., 1995) as it can be used in applications where the transform length is a composite number were the factors are mutually prime. Thus, there are in the literature several prime factor algorithms for 1-D DST (Chiper *et al*., 2002). Also, it is possible to combine prime-factor algorithms for an efficient computation or implementation of the 1-D DST transform for composite-lengths (Kar & Rao, 1994). In this paper we propose a new VLSI algorithm for 2-D DST that has a prime factor length.

As is well known 2-D DST is an computational intensive algorithm. Thus it is necessary to design application specific hardware that can speed up the execution of this transform or to reformulate in an appropriate manner existing algorithms for 2-D DST. In reformulation of the existing algorithm it is necessary to take into consideration the fact that data movement and transfer play a key role in obtaining an efficient VLSI implementation. In the literature there are cycle convolution and circular correlation algorithms that have remarkable advantages over other ones due to its efficient input/output operations and data transfer. These computational structures can be efficiently implemented in VLSI using distributed arithmetic (White, 1989) or systolic arrays (Kung, 1982).

The above mentioned advantages of the cycle convolution can be extended to other structures as for example skew-cycle and pseudo-cycle convolutions.

In this paper we propose a new systolic array algorithm for 2-D DST using some regular and modular computational structures. These structures can be computed in parallel resulting thus a high throughput VLSI implementation. We have used a new restructuring method of the 2-D DST into such regular structures. The proposed algorithm is appropriated for a memory-based implementation as will be discussed in Section **3**. All the advantages of a cycle

convolution based implementation as regularity, modularity, low I/O cost and a reduced data management scheme can be obtained with the proposed computational structures.

The rest of the paper is organized as follows: in Section **2** a low complexity formulation is presented for the computation of the 2-D DST transform with an example for a 2-D DST of length $N = 11$. In Section **3** we discuss some details about a VLSI implementation of the proposed algorithm using the systolic array architectural paradigm. Conclusions are presented in Section **4**.

2. Systolic Algorithm for 2-D DST

The 2-D DST for a $N \times N$ pixel block can be defined as follows:

$$Y(k,l) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} x(i,j)\sin[(2i+1)k\alpha]\sin[(2j+1)l\alpha], \tag{1}$$

where

$$\alpha = \frac{\pi}{2N} \tag{2}$$

and $x(i,j)$, $(i, j = 0,1,\ldots,N-1)$, is the pixel of an image with $Y(k,l)$, $(k,l = 1, \ldots, N)$, the transform coefficient.

To simplify our presentation, we have dropped the constant coefficient from the eq. (1) that represents the definition of 2-D DST. We will add at the end of the VLSI array a multiplier to scale the output sequence with this constant.

In the literature there are presented several 2-D VLSI architectures for DST. Most of them use the row-column decomposition method. Some of them are using a direct method to compute forward or inverse 2-D DCT or DST.

We can express (1) in a matrix form as

$$[x_N] = [S_N][X_N][S_N]^T, \tag{3}$$

where $[S_N]$ is defined as

$$[S_N]_{i,j} = \begin{cases} 1, & \text{for } i = 0, \\ \sin[(2i+1)j\alpha], & \text{otherwise.} \end{cases} \tag{4}$$

To compute (3) we have to compute first

$$[Y_N] = [X_N][S_N]^T, \tag{5}$$

along the rows of the input $[X_N]$. We can transpose the relation (5) to obtain

$$\left[Y_N\right]^T = \left[S_N\right]\left[X_N\right]^T. \tag{6}$$

To illustrate our approach we will consider a 2-D DST transform of length $N = 11$.

We can write (6) as follows:

$$\begin{bmatrix} y(1,1) & y(1,2) & \cdots & y(1,N) \\ y(2,1) & y(2,2) & \cdots & y(2,N) \\ \vdots & \vdots & \vdots & \vdots \\ y(N,1) & y(N,2) & \cdots & y(N,N) \end{bmatrix} =$$

$$= \begin{bmatrix} \sin(1\alpha) & \sin(3\alpha) & \sin(5\alpha) & \sin(7\alpha) & \sin(9\alpha) & \sin(11\alpha) & \sin(13\alpha) & \sin(15\alpha) & \sin(17\alpha) & \sin(19\alpha) & \sin(21\alpha) \\ \sin(2\alpha) & \sin(6\alpha) & \sin(10\alpha) & \sin(14\alpha) & \sin(18\alpha) & \sin(22\alpha) & \sin(26\alpha) & \sin(30\alpha) & \sin(34\alpha) & \sin(38\alpha) & \sin(42\alpha) \\ \sin(3\alpha) & \sin(9\alpha) & \sin(15\alpha) & \sin(21\alpha) & \sin(27\alpha) & \sin(33\alpha) & \sin(39\alpha) & \sin(45\alpha) & \sin(51\alpha) & \sin(57\alpha) & \sin(63\alpha) \\ \sin(4\alpha) & \sin(12\alpha) & \sin(20\alpha) & \sin(28\alpha) & \sin(36\alpha) & \sin(44\alpha) & \sin(52\alpha) & \sin(60\alpha) & \sin(68\alpha) & \sin(76\alpha) & \sin(84\alpha) \\ \sin(5\alpha) & \sin(15\alpha) & \sin(25\alpha) & \sin(35\alpha) & \sin(45\alpha) & \sin(55\alpha) & \sin(65\alpha) & \sin(75\alpha) & \sin(85\alpha) & \sin(95\alpha) & \sin(105\alpha) \\ \sin(6\alpha) & \sin(18\alpha) & \sin(30\alpha) & \sin(42\alpha) & \sin(54\alpha) & \sin(66\alpha) & \sin(78\alpha) & \sin(90\alpha) & \sin(102\alpha) & \sin(114\alpha) & \sin(126\alpha) \\ \sin(7\alpha) & \sin(21\alpha) & \sin(35\alpha) & \sin(49\alpha) & \sin(63\alpha) & \sin(77\alpha) & \sin(91\alpha) & \sin(105\alpha) & \sin(119\alpha) & \sin(133\alpha) & \sin(147\alpha) \\ \sin(8\alpha) & \sin(24\alpha) & \sin(40\alpha) & \sin(56\alpha) & \sin(72\alpha) & \sin(88\alpha) & \sin(104\alpha) & \sin(120\alpha) & \sin(136\alpha) & \sin(152\alpha) & \sin(168\alpha) \\ \sin(9\alpha) & \sin(27\alpha) & \sin(45\alpha) & \sin(63\alpha) & \sin(81\alpha) & \sin(99\alpha) & \sin(117\alpha) & \sin(135\alpha) & \sin(153\alpha) & \sin(171\alpha) & \sin(189\alpha) \\ \sin(10\alpha) & \sin(30\alpha) & \sin(50\alpha) & \sin(70\alpha) & \sin(90\alpha) & \sin(110\alpha) & \sin(130\alpha) & \sin(150\alpha) & \sin(170\alpha) & \sin(190\alpha) & \sin(210\alpha) \\ \sin(11\alpha) & \sin(33\alpha) & \sin(55\alpha) & \sin(77\alpha) & \sin(99\alpha) & \sin(121\alpha) & \sin(143\alpha) & \sin(165\alpha) & \sin(187\alpha) & \sin(209\alpha) & \sin(231\alpha) \end{bmatrix} \times$$

$$\times \begin{bmatrix} x(1,0) & x(2,0) & \cdots & x(11,0) \\ x(1,1) & x(2,1) & \cdots & x(11,1) \\ \vdots & \vdots & \ddots & \vdots \\ x(1,10) & x(2,10) & \cdots & x(11,10) \end{bmatrix}. \tag{7}$$

We can compute a row from eq. (7) as follows:

$$\begin{bmatrix} Y(1) \\ Y(2) \\ \vdots \\ Y(11) \end{bmatrix} =$$

$$= \begin{bmatrix} \sin(1\alpha) & \sin(3\alpha) & \sin(5\alpha) & \sin(7\alpha) & \sin(9\alpha) & \sin(11\alpha) & \sin(13\alpha) & \sin(15\alpha) & \sin(17\alpha) & \sin(19\alpha) & \sin(21\alpha) \\ \sin(2\alpha) & \sin(6\alpha) & \sin(10\alpha) & \sin(14\alpha) & \sin(18\alpha) & \sin(22\alpha) & \sin(26\alpha) & \sin(30\alpha) & \sin(34\alpha) & \sin(38\alpha) & \sin(42\alpha) \\ \sin(3\alpha) & \sin(9\alpha) & \sin(15\alpha) & \sin(21\alpha) & \sin(27\alpha) & \sin(33\alpha) & \sin(39\alpha) & \sin(45\alpha) & \sin(51\alpha) & \sin(57\alpha) & \sin(63\alpha) \\ \sin(4\alpha) & \sin(12\alpha) & \sin(20\alpha) & \sin(28\alpha) & \sin(36\alpha) & \sin(44\alpha) & \sin(52\alpha) & \sin(60\alpha) & \sin(68\alpha) & \sin(76\alpha) & \sin(84\alpha) \\ \sin(5\alpha) & \sin(15\alpha) & \sin(25\alpha) & \sin(35\alpha) & \sin(45\alpha) & \sin(55\alpha) & \sin(65\alpha) & \sin(75\alpha) & \sin(85\alpha) & \sin(95\alpha) & \sin(105\alpha) \\ \sin(6\alpha) & \sin(18\alpha) & \sin(30\alpha) & \sin(42\alpha) & \sin(54\alpha) & \sin(66\alpha) & \sin(78\alpha) & \sin(90\alpha) & \sin(102\alpha) & \sin(114\alpha) & \sin(126\alpha) \\ \sin(7\alpha) & \sin(21\alpha) & \sin(35\alpha) & \sin(49\alpha) & \sin(63\alpha) & \sin(77\alpha) & \sin(91\alpha) & \sin(105\alpha) & \sin(119\alpha) & \sin(133\alpha) & \sin(147\alpha) \\ \sin(8\alpha) & \sin(24\alpha) & \sin(40\alpha) & \sin(56\alpha) & \sin(72\alpha) & \sin(88\alpha) & \sin(104\alpha) & \sin(120\alpha) & \sin(136\alpha) & \sin(152\alpha) & \sin(168\alpha) \\ \sin(9\alpha) & \sin(27\alpha) & \sin(45\alpha) & \sin(63\alpha) & \sin(81\alpha) & \sin(99\alpha) & \sin(117\alpha) & \sin(135\alpha) & \sin(153\alpha) & \sin(171\alpha) & \sin(189\alpha) \\ \sin(10\alpha) & \sin(30\alpha) & \sin(50\alpha) & \sin(70\alpha) & \sin(90\alpha) & \sin(110\alpha) & \sin(130\alpha) & \sin(150\alpha) & \sin(170\alpha) & \sin(190\alpha) & \sin(210\alpha) \\ \sin(11\alpha) & \sin(33\alpha) & \sin(55\alpha) & \sin(77\alpha) & \sin(99\alpha) & \sin(121\alpha) & \sin(143\alpha) & \sin(165\alpha) & \sin(187\alpha) & \sin(209\alpha) & \sin(231\alpha) \end{bmatrix} \times$$

$$\times \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(10) \end{bmatrix}, \tag{8}$$

where $x(i)$, $(i = 0,1,...,N-1)$, is a real input sequence.

We will reformulate relation (8) as a parallel decomposition based on a skew-cycle and pseudo-cycle convolution forms using a single new input

restructuring sequence as opposed to those proposed by White, (1989), where we have used two such auxiliary input sequences. Further, we'll use the properties of DST kernel and the properties of the Galois Field of indexes to appropriately permute the auxiliary input and output sequences.

To illustrate our approach, we will consider an example with the length $N = 11$ and the primitive root $g = 2$.

Thus, we will introduce the following auxiliary input sequence: $\{x_a(i): i = 0,1,\ldots,N-1\}$. It can be recursively computed as follows:

$$x_a(N-1) = x(N-1), \tag{9}$$

$$x_a(i) = (-1)^i x(i) + x_a(i+1)\,, \tag{10}$$

for $i = N-2,\ldots,0$.

Using this restructuring input sequence we can reformulate (8) as follows

$$\begin{bmatrix} Y(1) \\ Y(2) \\ Y(3) \\ Y(4) \\ Y(5) \\ Y(6) \\ Y(7) \\ Y(8) \\ Y(9) \\ Y(10) \end{bmatrix} = x_a(0) \begin{bmatrix} \sin(\alpha) \\ \sin(2\alpha) \\ \sin(3\alpha) \\ \sin(4\alpha) \\ \sin(5\alpha) \\ \sin(6\alpha) \\ \sin(7\alpha) \\ \sin(8\alpha) \\ \sin(9\alpha) \\ \sin(10\alpha) \end{bmatrix} + 2 \begin{bmatrix} T(1)\cos(\alpha) \\ T(2)\cos(2\alpha) \\ T(3)\cos(3\alpha) \\ T(4)\cos(4\alpha) \\ T(5)\cos(5\alpha) \\ T(6)\cos(6\alpha) \\ T(7)\cos(7\alpha) \\ T(8)\cos(8\alpha) \\ T(9)\cos(9\alpha) \\ T(10)\cos(10\alpha) \end{bmatrix}, \tag{11}$$

and

$$Y(0) = x_a(0)\,. \tag{12}$$

The new auxiliary output sequence, $\{T(k): k = 1,2,\ldots,N-1\}$, can be computed in parallel as two pseudo-cycle convolutions, if the transform length, N, is a prime number, as following:

$$\begin{bmatrix} T(2) \\ T(4) \\ T(8) \\ T(6) \\ T(10) \end{bmatrix} = \begin{bmatrix} [x_a(2)+x_a(9)] & [x_a(4)+x_a(7)] & -[x_a(8)+x_a(3)] & -[x_a(5)+x_a(6)] & -[x_a(10)+x_a(1)] \\ -[x_a(10)+x_a(1)] & [x_a(2)+x_a(9)] & -[x_a(4)+x_a(7)] & [x_a(8)+x_a(3)] & [x_a(5)+x_a(6)] \\ [x_a(5)+x_a(6)] & -[x_a(10)+x_a(1)] & -[x_a(2)+x_a(9)] & [x_a(4)+x_a(7)] & -[x_a(8)+x_a(3)] \\ [x_a(8)+x_a(3)] & -[x_a(5)+x_a(6)] & -[x_a(10)+x_a(1)] & -[x_a(2)+x_a(9)] & [x_a(4)+x_a(7)] \\ -[x_a(4)+x_a(7)] & -[x_a(8)+x_a(3)] & -[x_a(5)+x_a(6)] & -[x_a(10)+x_a(1)] & -[x_a(2)+x_a(9)] \end{bmatrix} \cdot \begin{bmatrix} \sin(8\alpha) \\ \sin(6\alpha) \\ \sin(10\alpha) \\ \sin(\alpha) \\ \sin(4\alpha) \end{bmatrix}, \quad (13)$$

$$\begin{bmatrix} T(9) \\ T(7) \\ T(3) \\ T(5) \\ T(1) \end{bmatrix} = \begin{bmatrix} -[x_a(2)-x_a(9)] & -[x_a(4)-x_a(7)] & [x_a(8)-x_a(3)] & -[x_a(5)-x_a(6)] & [x_a(10)-x_a(1)] \\ [x_a(10)-x_a(1)] & -[x_a(2)-x_a(9)] & [x_a(4)-x_a(7)] & -[x_a(8)-x_a(3)] & [x_a(5)-x_a(6)] \\ [x_a(5)-x_a(6)] & [x_a(10)-x_a(1)] & [x_a(2)-x_a(9)] & -[x_a(4)-x_a(7)] & [x_a(8)-x_a(3)] \\ -[x_a(8)-x_a(3)] & -[x_a(5)-x_a(6)] & [x_a(10)-x_a(1)] & [x_a(2)-x_a(9)] & -[x_a(4)-x_a(7)] \\ [x_a(4)-x_a(7)] & [x_a(8)-x_a(3)] & -[x_a(5)-x_a(6)] & [x_a(10)-x_a(1)] & [x_a(2)-x_a(9)] \end{bmatrix} \cdot \begin{bmatrix} \sin(8\alpha) \\ \sin(6\alpha) \\ \sin(10\alpha) \\ \sin(\alpha) \\ \sin(4\alpha) \end{bmatrix}. \quad (14)$$

We have used two index mappings, $\nu(i)$ and $\mu(i)$, to realize a partition into two groups of the permutation of indexes $\{1,2,3,4,5,6,7,8,9,10\}$. They are defined as follows:

$$\{\nu(i): 1 \to 2,\ 2 \to 4,\ 3 \to 8, 4 \to 6,\ 5 \to 10\},$$
$$\{\mu(i): 1 \to 9,\ 2 \to 7,\ 3 \to 3, 4 \to 5,\ 5 \to 1\},$$

The signs of terms in eqs. (13) and (14) are given by the functions $\varepsilon(k,i)$ and $\gamma(k,i)$ defined, respectively, as follows:

$\varepsilon(k,i)$ is defined by the matrix $\begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ and

$\gamma(k,i)$ is defined by the matrix $\begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$.

Eq. (3) can be computed by N N-point DST along the rows of the input $[X_N]$, obtaining $[Y_N] = [X_N]\ [S_N]^T$, followed by N N-point DSTs along the columns of the matrix obtained from the row transformed, $[x_N] = [S_N]\ [X_N]$.

This simple decomposition method reduces the computation complexity with a factor of 4.

3. A VLSI Implementation Discussion

Using the algorithm presented in Section **2** we can obtain two linear systolic arrays using a dependence-graph based synthesis procedure. These arrays represent the main part of the architecture used for the VLSI implementation of the derived algorithm. The obtained processing elements consists of a multiplier, an adder and some multiplexers used to manage the differences in sign in eqs. (13) and (14), respectively. Note that each multiplier realizes a multiplication in eqs. (13) and (14) with a constant resulting to a further improvement, that consists in replacing of multipliers with look-up tables LUTs residing in small ROMs. We can obtain a further reduction of the hardware complexity using an appropriate hardware sharing method.

One important feature of the proposed solution is its low I/O cost, an aspect that could be very useful in designing systolic arrays where the so called *I/O bottle-neck* can seriously limit the usefulness of this concept. The tag-control mechanism can be used to place all I/O channels at the two extreme ends of each systolic array and to control the internal registers using only I/O channels placed at the two ends of the linear systolic array.

The pre-processing stage realizes the computation of the auxiliary input sequence using eqs. (9) and (10) and also a permutation of the resulting input sequence. There is also necessary to reorder the input sequence in order to compute eqs. (9) and (10). Each permutation can be obtained using a RAM with N words.

The post-processing stage has the role to reorder the auxiliary output sequence $\{T(k): k = 1,2,..., N-1\}$, in such a way that to put it in a natural order and will yield the final output sequence using eq. (11).

4. Conclusions

In this paper it is presented a new VLSI algorithm for 2-D DST that leads to an efficient VLSI architecture with appealing features for a VLSI implementation and high performances. The proposed algorithm uses some modular and regular computational structures that can be computed in parallel thus resulting a high throughput VLSI implementation. It can be mapped into two linear systolic arrays with a high throughput and low I/O cost and hardware complexity. By combining two such linear systolic arrays we can obtain an efficient VLSI architecture for 2-D DST that is highly regular and modular and having local connections, being well adapted to the VLSI technology.

REFERENCES

Ahmed N., Natarajan T., Rao K.R., *Discrete Cosine Transform.* IEEE Trans. Comput., **C-23**, *1*, 90-94 (1974).

Bergen S.W., *A Design Method for Cosine-Modulated Filter Banks Using Weighted Constrained-Least-Squares Filters.* Digital Signal Proc., **18**, *3*, 282-290 (2008).

Chen Y.-Y., *Medical Image Compression Using DCT-Based Subband Decomposition and Modified SPIHT Data Organization.* Internat. J. of Medical Informatics, **76**, 717-725 (2007).

Chiper D.F., Swamy M.N.S., Ahmad M.O., Stouraits T., *A Systolic Array Architecture for the Discrete Sine Transform.* IEEE Trans. on Signal Proc., **50**, *9*, 2347-2354 (2002).

Fung K.-T., Siu W.-C., *On Re-Composition of Motion Compensated Macroblocks for DCT-Based Video Transcoding*. Signal Proc.: Image Commun., **21**, 44-58 (2006).

Jadhav D.V., Holambe R.S., *Random and Discrete Cosine Transform Based Features Extraction and Dimensionality Reduction Approach for Face Recognition.* Signal Proc., **88**, 2604-2609 (2008).

Jain A.K., *A Fast Karhunen-Loeve Transform for a Class of Random Processes.* IEEE Trans. Comm., **COM-24**, *10*, 1023-1029 (1976).

Jain A.K., *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ, Prentice-Hall, 1989.

Kar D., Rao V.V.B., *On Prime Factor Decomposition Algorithm for the Discrete Sine Transform*. IEEE Trans. on Signal Proc., **42**, *11*, 3258-3260 (1994).

Kung H.T., *Why Systolic Architectures.* IEEE Comp., **15**, 37-46 (1982).

Martucci S.A., Mersereau R., *New Approaches to Block Filtering of Images Using Symmetric Convolution and the DST or DCT*. Proc. IEEE Internat. Symp. Circ. Syst. (ISCAS'93), May 1993, 259-262.

Mayyas K., *A Note on Performance Analysis of the DCT-LMS Adaptive Filtering Algorithm.* Signal Proc., **85**, 1465-1467 (2005).

Park Y.S., Park H.W., *Arbitrary-Ratio Image Resizing Using Fast DCT of Composite Length for DCT-Based Transcoder*. IEEE Trans. Image Proc., **15**, *2*, 494-500 (2006).

Pei S.-C., Tseng C.-C., *Transform-Domain Adaptive Filter*. IEEE Trans. Signal Proc., **44**, *12*, 3142-3146 (1996).

Tatsaki A., Dre C., Stouraitis T., Goutis C., *Prime-Factor DCT Algorithms*. IEEE Trans. on Signal Proc., **43,** *3*, 772-776 (1995).

Wang Z., Jullien G.A., Miller W.C., *Interpolation Using the Discrete Sine Transform with Increased Acuracy.* Electron. Letters, **29**, *22*, 1918-1920 (1993).

White S.A., *Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review.* IEEE ASSP Mag., **6,** *3*, 5-19 (1989).

Zhang D., Lin S., Zhang Y., Yu L., *Complexity Controllable DCT for Real-Time H.264 Encoder*. J. of Visual Commun. a. Image Repres., **18**, 59-67 (2007).

UN NOU ALGORITM SISTOLIC PENTRU TRANSFORMATA 2-D DST

(Rezumat)

Se prezintă un nou algoritm VLSI pentru transformata 2-D DST care conduce la o arhitectură VLSI eficientă având anumite caracteristici atrăgătoare pentru o implementare VLSI şi cu performanțe ridicate. Algoritmul propus utilizează nişte structuri computaționale modulare şi regulate care pot fi calculate în paralel conducând la un „throughput" ridicat. El poate fi mapat pe două arii sistolice având un „throughput" ridicat, o complexitate a operațiilor de intrare/ieşire scăzută şi o complexitate hardware redusă. Prin combinarea celor două arii sistolice se poate obține o arhitectură VLSI eficientă pentru transformata 2-D DST care are un caracter modular şi regulat, cu conexiuni locale, caracteristică specifică ariilor sistolice.