I.J. Image, Graphics and Signal Processing, **2014, 10, 1-9**
Published Online September 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2014.10.01

Algorithmic Tricks for Reducing the Complexity of FDWT/IDWT Basic Operations Implementation

Aleksandr Cariow
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin, Poland
E-mail: atariov@wi.zut.edu.pl

Galina Cariowa
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin, Poland
E-mail: gtariova@wi.zut.edu.pl

***Abstract*—**In this paper two different approaches to the rationalization of FDWT and IDWT basic operations execution with the reduced number of multiplications are considered. With regard to the well-known approaches, the direct implementation of the above operations requires 2*L* multiplications for the execution of FDWT and IDWT basic operation plus 2(*L*-1) additions for FDWT basic operation and *L* additions for IDWT basic operation. At the same time, the first approach allows the design of the computation procedures, which take only 1,5*L* multiplications plus 3,5*L*+1 additions for FDWT basic operation and *L*+1 multiplications plus 3,5L additions for IDWT basic operation. The other approach allows the design of such computation procedures, which require 1,5*L* multiplications, plus 2*L*-1 addition for FDWT basic operation and *L*+1 addition for IDWT basic operation.

***Index Terms*—**Discrete wavelet transform, fast algorithms, matrix notation.

I. Introduction

Recently, a discrete wavelet transform (DWT) has been used in numerous computer graphics, signal and image processing applications [1-11].

In 1989, Stéphane Mallat proposed a fast wavelet decomposition and reconstruction algorithm [1]. The basic idea of the fast algorithm is, in case of both 1D DWT and 2D DWT, the decomposition of the original signal (or image) using a pair of filters (high- and low-pass) on two components and following the decomposition of the low pass component in the same hierarchical manner. This decomposition process is called analysis. The inverse process is called reconstruction (or synthesis) [3, 5, 7].

The cores of multilevel DWT decomposition and reconstruction procedures are forward DWT (FDWT) and inverse DWT (IDWT) "basic operations" – the multiplication of data vector by FDWT or IDWT base matrix, which describes the filter coefficients [12].

In the matrix-vector form FDWT basic operation can be defined in the following way:

$$\mathbf{Y}_{2\times1}^{(l)} = \mathbf{F}_{2\times L}\mathbf{X}_{L\times1}^{(l)},\ l = 0,1,\mathrm{k}\ldots, N/2^k - 1 \qquad (1)$$

While the inverse operation used to recreate signal from DWT coefficients (inverse DWT base operation) looks as demonstrated in (2):

$$\widetilde{\mathbf{X}}_{L\times1}^{(l)} = \left(\mathbf{F}_{2\times L}\right)^{\mathrm{T}}\mathbf{Y}_{2\times1}^{(l)} \qquad (2)$$

where N – is a number of the original signal samples, L – is a size of sliding window that defines the part of the signal processed by the given base operation, $k = 0,1,\ldots,K-1$ – is a decomposition step number indicating the granularity degree of data resolution, K – the total number of the decomposition steps.

Vector $\mathbf{Y}_{2\times1}^{(l)} = [y_{2l}, y_{2l+1}]^{\mathrm{T}}$ - is a two-element output data vector for each basic operation (1)-(2), where y_{2l} and y_{2l+1} are, respectively, detail and approximation components, which are computing as a result of the execution of FDWT basic operations.

Vector $\mathbf{X}_{L\times1}^{(l)} = [x_{2l}, x_{2l+1},\ldots,x_{L+2l-1}]^{\mathrm{T}}$ in (1) represents a column vector of length L whose elements represent the corresponding sequence of signal samples and vector $\widetilde{\mathbf{X}}_{L\times1}^{(l)} = [\widetilde{x}_{2l}, \widetilde{x}_{2l+1},\ldots,\widetilde{x}_{L+2l-1}]^{\mathrm{T}}$ in (2) represents the approximation of the vector $\mathbf{X}_{L\times1}^{(l)}$ obtained as a result of the reconstruction of initial data from two-element vector DWT coefficients $\mathbf{Y}_{2\times1}^{(l)}$.

Matrix

$$\mathbf{F}_{2\times L}=\begin{bmatrix} c_0 & c_1 & c_2 & \Lambda & c_{L-1} \\ c_{L-1} & -c_{L-1} & c_{L-3} & \Lambda & -c_0 \end{bmatrix}$$

in (1) is a FDWT base matrix with dimensions of $(2\times L)$, whose elements represent the coefficients of the high-pass and low-pass filter respectively. Formulas (1), (2) define FDWT and IDWT basic operation respectively [12].

From expressions (1) and (2) it is clear that the implementation of the FDWT basic operations required to perform $2L$ multiplications and $2(L-1)$ additions whereas the implementation of the IDWT basic operations required to perform $2L$ multiplications and L additions. In this paper some algorithmic tricks that reduce the number of multiplications in the implementation of the FDWT/IDWT basic operations are demonstrated.

Minimizing the number of multiplications is especially important in the design of specialized VLSI mathematical or DSP processors because reducing the number of multipliers also reduces the power dissipation and lowers the cost implementation of the entire system being implemented. Moreover, a hardware multiplier is more complicated unit than an adder and occupies much more chip area than the adder. Even if the chip already contains embedded multipliers, their number is always limited. This means that if the implemented algorithm has a large number of multiplications, the projected processor may not always fit into the chip and the problem of minimizing the number of multiplications remains relevant.

In this paper two approaches to the synthesis of rationalized algorithms for implementation of FDWT/IDWT basic operations are considered. In the first approach, the reduction of the number of multipliers in the implementation of the FDWT/IDWT basic operations is achieved by applying the Winograd's inner product formula. Another approach uses effective schemes of matrix factorization described in [13]. It should also be noted that during the construction of algorithms the extensive use of the principles of parallelization and vectorization of data processing will be made. Therefore, it is assumed that the synthesized procedures will be implemented on the hardware platforms with parallel processing.

II. Rationalized Algorithms For FDWT/IDWT Basic Operations Inplementation Using Winograd's Inner Product Formula

A. Background

Let $\mathbf{X}_{N\times 1}=[x_0,x_1,\ldots,x_{N-1}]^{\mathrm{T}}$ and $\mathbf{Y}_{M\times 1}=[y_0,y_1,\ldots,y_{M-1}]^{\mathrm{T}}$ - be N-point and M-point one dimensional data vectors respectively, and $m=\overline{0,M-1}$, $n=\overline{0,N-1}$.

The calculation of product poses a problem

$$\mathbf{Y}_{M\times 1}=\mathbf{A}_{M\times N}\mathbf{X}_{N\times 1}.$$

According to Winograd's formula for the inner product calculation each element of vector $\mathbf{Y}_{M\times 1}$ can be calculated as follows [14]:

$$y_m=\sum_{k=0}^{\frac{N}{2}-1}[(a_{m,2k}+x_{2k+1})(a_{m,2k+1}+x_{2k})]-\varsigma_m-\xi(M)$$

Where

$$\varsigma_m=\sum_{k=0}^{\frac{N}{2}-1}a_{m,2k}\cdot a_{m,2k+1}\ \text{ and }\ \xi(M)=\sum_{k=0}^{\frac{N}{2}-1}x_{2k}\cdot x_{2k+1}$$

if N is even.

It should be emphasized that if matrix $\mathbf{A}_{M\times N}$ is a constant coefficient matrix, ς_m can be calculated in advance. The calculation of $\xi(M)$ requires the $N/2$ multiplications to be performed. By exploiting some rationalization solutions based on application of the Winograd's inner product formula, the number of multipliers, necessary for fully-parallel implementation multipliers could be decreased.

B. Rationalized algorithm for FDWT basic operation implementation using Winograd's inner product formula

In the beginning, when using the elements of the matrix $\mathbf{F}_{2\times L}$, a column vector will be formed:

$$\mathbf{F}_{3L\times 1}=[\widehat{\mathbf{C}}_{L\times 1},\mathbf{0}_{L\times 1},\breve{\mathbf{C}}_{L\times 1}]^{\mathrm{T}}$$

where

$\widehat{\mathbf{C}}_{L\times 1}=[c_0,c_1,\ldots,c_{L-1}]^{\mathrm{T}}$ - is a column vector containing the impulse response coefficients of the low pass filter, $\breve{\mathbf{C}}_{L\times 1}=[c_{L-1},-c_{L-2},\ldots,\mp c_0]^{\mathrm{T}}$ - is a column vector containing the impulse response coefficients of the high pass filter, and $\mathbf{0}_{L\times 1}$ - is a column vector or matrix of size defined by low subscript and consisting of all zeros.

Next, we introduce some auxiliary matrices:

- matrix of data duplication $\mathbf{P}_{3L\times L}=\mathbf{1}_{3\times 1}\otimes\mathbf{I}_L$, where $\mathbf{1}_{3\times 1}=[1,1,1]^{\mathrm{T}}$ here and further in this paper, is a matrix of size defined by a low subscript and consisting of all 1s, $\otimes$ - is a Kronecker product sign [15];
- permutation matrix $\mathbf{P}_{3L}=\begin{pmatrix}\mathbf{I}_{\frac{3L}{2}}\otimes\mathbf{J}_2\end{pmatrix}$, where $\mathbf{I}$ - here and further in this article, is an identity matrix of size defined by low subscript, $\mathbf{J}$ - is an exchange matrix of size defined by low subscript,

where the 1 elements reside on the counter diagonal and all other elements are zero;

- partial products summation matrix

$$\mathbf{A}_{3\times\frac{3L}{2}} = \mathbf{I}_3 \otimes \mathbf{1}_{1\times\frac{L}{2}};$$

- vector $\mathbf{\Theta}_{2\times1} = [+\Xi, -\Xi,]^{\mathrm{T}}$, $\Xi = \sum_{i=0}^{\frac{L}{2}-1} c_{2i} \cdot c_{2i+1}$;

- matrices

$$\mathbf{R}_{\frac{3L}{2}\times3L} = \mathbf{I}_{\frac{3L}{2}} \otimes \mathbf{1}_{1\times2} \text{ and } \mathbf{K}_{2\times3} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$$

Taking into account the introduced vector-matrix constructions, the FDWT basic operation computational procedure with a reduced number of multiplications (or multipliers in case of hardware implementation) can be represented as follows:

$$\mathbf{Y}_{2\times1} = \mathbf{\Theta}_{2\times1} + \mathbf{K}_{2\times3}\mathbf{A}_{3\times\frac{3L}{2}}[\mathbf{R}_{\frac{3L}{2}\times3L} \circ \circ(\mathbf{P}_{3L}\mathbf{F}_{3L\times1} + \mathbf{P}_{3L\times L}\mathbf{X}_{L\times1})] \quad (3)$$

The operator " $\circ$ " named "vectorized Hadamard product" [12, 13] has been introduced for the convenience of the description of the simultaneous selected vector elements multiplication procedure – it will transform certain data column vector $\mathbf{X}_{N\times1}$ into the output column vector $\mathbf{Y}_{M\times1}$ in a following way:

$$\mathbf{Y}_{M\times1} = \mathbf{B}_{M\times N} \circ \mathbf{X}_{N\times1},$$

where $\mathbf{B}_{M\times N} = \|b_{m,n}\|$, $m = \overline{0, M-1}$, $n = \overline{0, N-1}$ - is a binary mask-matrix, and the elements y_m are determined by the following rule [12, 13]:

$$y_i = \begin{cases} \sum_{n=0}^{N-1} b_{m,n}x_n, & \text{for } \sum_{n=0}^{N-1} b_{m,n} = 1 \\ \prod_{n=0}^{N-1} b_{m,n}x_n, & \text{for } \sum_{n=0}^{N-1} b_{m,n} > 1 \end{cases} \quad \forall i = \overline{0, M-1},$$

Let us consider a concrete example. Suppose $L = 8$. Then

$$\mathbf{F}_{2\times8} = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \\ c_7 & -c_6 & c_5 & -c_4 & c_3 & -c_2 & c_1 & -c_0 \end{bmatrix},$$

and

$$\mathbf{F}_{24\times1} = [c_0, c_1, ,\ldots, c_7, 0, 0, \ldots 0, c_7, -c_6, \ldots, -c_0]^{\mathrm{T}}.$$

Whereas the computational procedure for this example takes the following form:

$$\mathbf{Y}_{2\times1} = \mathbf{\Theta}_{2\times1} + \mathbf{K}_{2\times3}\mathbf{A}_{3\times12}[\mathbf{R}_{12\times24} \circ (\mathbf{P}_{24}\mathbf{F}_{24\times1} + \mathbf{P}_{24\times8}\mathbf{X}_{8\times1}) \quad (4)$$

where $\Xi = c_0c_1 + c_2c_3 + c_4c_5 + c_6c_7$,

$$\mathbf{P}_{24\times8} = \mathbf{1}_{3\times1} \otimes \mathbf{I}_8 = \left[\begin{array}{cccccccc} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \\ \hline 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \\ \hline 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{array}\right],$$

$$\mathbf{P}_{24} = (\mathbf{I}_4 \otimes \mathbf{J}_2) \oplus \mathbf{0}_8 \oplus (\mathbf{I}_4 \otimes \mathbf{J}_2) = \left[\begin{array}{cccc|c|cccc} \mathbf{J}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_8 & \multicolumn{4}{c}{\multirow{4}{*}{$\mathbf{0}_8$}} \\ \mathbf{0}_2 & \mathbf{J}_2 & \mathbf{0}_2 & \mathbf{0}_2 & & & & & \\ \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{J}_2 & \mathbf{0}_2 & & & & & \\ \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{J}_2 & & & & & \\ \hline \multicolumn{4}{c|}{\mathbf{0}_8} & \mathbf{0}_8 & \multicolumn{4}{c}{$\mathbf{0}_8$} \\ \hline \multicolumn{4}{c|}{\multirow{4}{*}{$\mathbf{0}_8$}} & \multirow{4}{*}{$\mathbf{0}_8$} & \mathbf{J}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_2 \\ & & & & & \mathbf{0}_2 & \mathbf{J}_2 & \mathbf{0}_2 & \mathbf{0}_2 \\ & & & & & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{J}_2 & \mathbf{0}_2 \\ & & & & & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{J}_2 \end{array}\right],$$

$$\mathbf{R}_{12\times 24}=\mathbf{I}_3\otimes\left[\begin{array}{c|c|c|c} 1\ \ 1 & \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} \\ \hline \mathbf{0}_{1\times 2} & 1\ \ 1 & \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} \\ \hline \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & 1\ \ 1 & \mathbf{0}_{1\times 2} \\ \hline \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & 1\ \ 1 \end{array}\right],$$

$$\mathbf{A}_{3\times 12}=\left[\begin{array}{cccc|cccc|cccc} 1 & 1 & 1 & 1 & & & & & & & & \\ \hline & & & & 1 & 1 & 1 & 1 & & & & \\ \hline & & & & & & & & 1 & 1 & 1 & 1 \end{array}\right]$$

Fig. 1 shows a data flow diagram, which describes the rationalized algorithm for the implementation of FDWT basic operation for case $L=8$. In this paper, the data flow diagrams are oriented from left to right. The squares here and further in this paper denote additions with constant values inscribed in the field. The circles in all figures show the operation of the multiplication of two variables or multiplication by a constant inscribed inside a circle. Straight lines in the figures denote the operations of data transfer. Points where lines converge denote summation. The dashed lines indicate the subtraction operation. We deliberately use the usual lines without arrows on purpose, so as not to clutter the picture.

C. Rationalized algorithm for IDWT basic operation implementation using Winograd's inner product formula

For the synthesis of rationalized calculation procedure for IDWT basic operation implementation the following matrix constructions are introduced:

- partial products summation matrix

$$\mathbf{A}_{L\times 2L}=(\bar{\mathbf{1}}_{1\times 2}\otimes\mathbf{I}_{L/2})\oplus(\mathbf{1}_{1\times 2}\otimes\mathbf{I}_{L/2}),$$

where $\bar{\mathbf{1}}_{1\times 2}=\left[1\ \vdots\ -1\right]$ and $\oplus$ - is direct sum sign [15],

- first permutation matrix

$$\mathbf{P}_{2(L+1)\times 2}=\mathbf{1}_{(L+1)\times 1}\otimes\mathbf{I}_2,$$

- second permutation matrix

$$\mathbf{P}_{\frac{3}{2}L\times(L+1)}=\mathbf{I}_{\frac{L}{2}}\oplus\mathbf{1}_{\frac{L}{2}\times 1}\oplus\mathbf{I}_{\frac{L}{2}},$$

- third permutation matrix

$$\mathbf{P}_{2L\times\frac{3}{2}L}=\mathbf{1}_{2\times 1}\otimes\mathbf{I}_L,$$

- fourth permutation matrix

$$\mathbf{P}_{2L}=\mathbf{I}_L\oplus\mathbf{J}_{\frac{L}{2}}\oplus\mathbf{I}_{\frac{L}{2}},$$

- column vector

$$\mathbf{C}_{2(L+1)\times 1}=[(\mathbf{I}_{\frac{L}{2}}\otimes\mathbf{J}_2)\oplus\mathbf{0}_2\oplus(\mathbf{I}_{\frac{L}{2}}\otimes\mathbf{J}_2)]\mathbf{\Phi}_{2(L+1)\times 1},$$

where

$$\mathbf{\Phi}_{2(L+1)\times 1}=[c_0,c_{L-1},\ldots,c_{\frac{L}{2}-1},-c_{\frac{L}{2}},0,0,c_{\frac{L}{2}},c_{\frac{L}{2}-1},\ldots,c_{L-1},-c_0]^{\mathrm{T}},$$

- column vector

$$\mathbf{\Psi}_{\frac{3L}{2}\times 1}=\mathbf{0}_{\frac{L}{2}\times 1}\,\blacksquare\,\boldsymbol{\gamma}_{\frac{L}{2}\times 1}\,\blacksquare\,\mathbf{0}_{\frac{L}{2}\times 1},$$

Where

$$\boldsymbol{\gamma}_{\frac{L}{2}\times 1}=[(c_0c_{L-1}),(c_1c_{L-2}),\ldots,(c_{\frac{L}{2}-1}c_{\frac{L}{2}})]^{\mathrm{T}}.$$

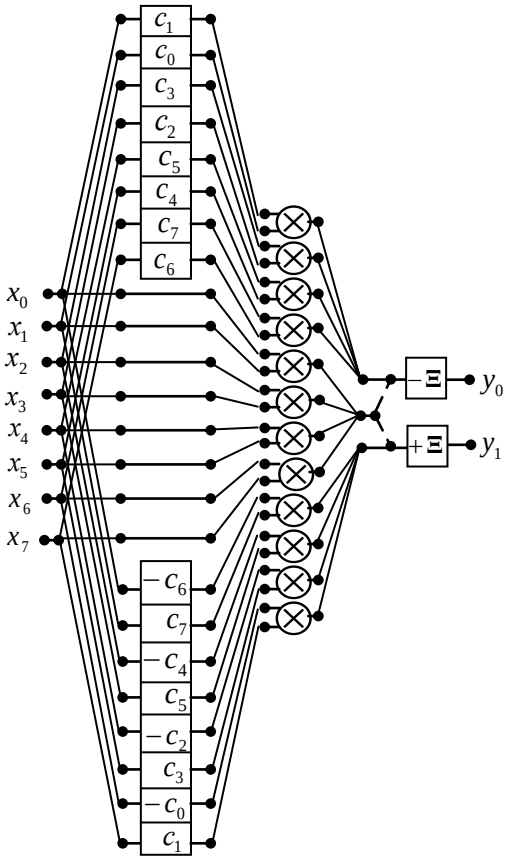

Fig 1: Data flow diagram for FDWT basic operation ralization according to the procedure (4).

When using introduced vector-matrix constructions, the IDWT basic operation computational procedure with reduced number of multiplications (or multipliers in case of hardware implementation) can be written in the following form:

$$\widetilde{\mathbf{X}}_{L\times 1}=\mathbf{A}_{L\times 2L}\mathbf{P}_{2L}\mathbf{P}_{2L\times\frac{3}{2}L}\{\mathbf{\Psi}_{\frac{3}{2}L\times 1}+\mathbf{P}_{\frac{3}{2}L\times(L+1)}[\mathbf{B}_{(L+1)\times 2(L+1)}]\circ$$
$$\circ(\mathbf{C}_{2(L+1)\times 1}+\mathbf{P}_{2(L+1)\times 2}\mathbf{Y}_{2\times 1})]\} \quad (5)$$

Let us consider the next example for same $L=8$.

The computational procedure (10) for this example takes the following form:

$$\widetilde{\mathbf{X}}_{8\times 1}=\mathbf{A}_{8\times 16}\mathbf{P}_{16}\mathbf{P}_{16\times 12}\{\mathbf{\Psi}_{12\times 1}+\mathbf{P}_{12\times 9}[\mathbf{B}_{9\times 18}]\circ$$
$$\circ(\mathbf{C}_{18\times 1}+\mathbf{P}_{18\times 2}\mathbf{Y}_{2\times 1})]\} \quad (6)$$

where

$$\mathbf{A}_{8\times 16}=(\overline{\mathbf{1}}_{1\times 2}\otimes\mathbf{I}_4)\oplus(\mathbf{1}_{1\times 2}\otimes\mathbf{I}_4)=$$
$$\mathbf{A}_{8\times 16}=\left[\begin{array}{cccccccc|cccccccc}1&&&&-1&&&&&&&&&&&\\&1&&&&-1&&&&&&&&&&\\&&1&&&&-1&&&&&&&&&\\&&&1&&&&-1&&&&&&&&\\\hline&&&&&&&&1&&1&&&&&\\&&&&&&&&&1&&1&&&&\\&&&&&&&&&&1&&1&&&\\&&&&&&&&&&&1&&1&&\end{array}\right],$$

$$\mathbf{P}_{12\times 9}=\mathbf{I}_4\oplus\mathbf{1}_{4\times 1}\oplus\mathbf{I}_4=\left[\begin{array}{ccc|c|c}1&&&&\\&1&&&\\&&1&\mathbf{0}_{4\times 1}&\mathbf{0}_4\\&&&1&\\\hline&&&1&\\&&&1&\\\mathbf{0}_4&&&1&\mathbf{0}_4\\&&&1&\\\hline&&&&1\\&&&&1\\\mathbf{0}_4&&&\mathbf{0}_{4\times 1}&1\\&&&&1\end{array}\right],$$

$$\mathbf{B}_{9\times 18}=\mathbf{I}_9\otimes\mathbf{1}_{1\times 2}=\mathbf{I}_9\otimes\left[1\ \vdots\ 1\right]=$$
$$=\left[\begin{array}{cccc|c|cccccc}1\ 1&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&&&&&&&\\\mathbf{0}_{1\times 2}&1\ 1&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\mathbf{0}_{4\times 2}&&&&\mathbf{0}_{4\times 8}&\\\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&1\ 1&\mathbf{0}_{1\times 2}&&&&&&\\\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&1\ 1&&&&&&\\\hline&&&&1\ 1&&&&&\\\hline&&&&&1\ 1&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\\&&&&&\mathbf{0}_{1\times 2}&1\ 1&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\\\mathbf{0}_{4\times 8}&&&&\mathbf{0}_{4\times 2}&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&1\ 1&\mathbf{0}_{1\times 2}&\\&&&&&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&\mathbf{0}_{1\times 2}&1\ 1&\end{array}\right],$$

$$\mathbf{P}_{18\times 2}=\mathbf{1}_{9\times 1}\otimes\mathbf{I}_2=\left[\begin{array}{c|c}1&\\\hline 1&1\\\hline 1&1\\\hline 1&1\\\hline 1&1\\\hline 1&1\\\hline 1&1\\\hline 1&1\\\hline 1&1\\\hline &1\end{array}\right],$$

$$\mathbf{C}_{18\times 1}=\mathbf{C}_{18}\mathbf{\Phi}_{2(L+1)\times 1},$$

$$\mathbf{C}_{18\times 1}=[c_7,c_0,\ldots,c_3,0,0,c_3,c_4,\ldots,c_7]^{\mathrm{T}},$$

$$\mathbf{\Phi}_{18\times 1}=[c_0,c_7,\ldots,c_3,-c_4,0,0,c_4,c_3,\ldots,c_7,-c_0]^{\mathrm{T}},$$

$$\mathbf{C}_{18}=(\mathbf{I}_4\otimes\mathbf{J}_2)\oplus\mathbf{0}_2\oplus(\mathbf{I}_4\otimes\mathbf{J}_2)=$$
$$=\left[\begin{array}{cccc|c|cccc}\mathbf{J}_2&\mathbf{0}_2&\mathbf{0}_2&\mathbf{0}_2&&&&&\\\mathbf{0}_2&\mathbf{J}_2&\mathbf{0}_2&\mathbf{0}_2&\mathbf{0}_{8\times 2}&&&\mathbf{0}_8&\\\mathbf{0}_2&\mathbf{0}_2&\mathbf{J}_2&\mathbf{0}_2&&&&&\\\mathbf{0}_2&\mathbf{0}_2&\mathbf{0}_2&\mathbf{J}_2&&&&&\\\hline\multicolumn{4}{c|}{\mathbf{0}_{2\times 8}}&\mathbf{0}_2&\multicolumn{4}{c}{\mathbf{0}_{2\times 8}}\\\hline&&&&&\mathbf{J}_2&\mathbf{0}_2&\mathbf{0}_2&\mathbf{0}_2\\&&&&&\mathbf{0}_2&\mathbf{J}_2&\mathbf{0}_2&\mathbf{0}_2\\\mathbf{0}_8&&&&\mathbf{0}_{8\times 2}&\mathbf{0}_2&\mathbf{0}_2&\mathbf{J}_2&\mathbf{0}_2\\&&&&&\mathbf{0}_2&\mathbf{0}_2&\mathbf{0}_2&\mathbf{J}_2\end{array}\right].$$

$$\mathbf{\Psi}_{12\times 1}=[\mathbf{0}_{4\times 1},\boldsymbol{\gamma}_{4\times 1},\mathbf{0}_{4\times 1}]^{\mathrm{T}},$$

$$\boldsymbol{\gamma}_{4\times 1}=\begin{bmatrix}c_0c_7\\c_1c_6\\c_2c_5\\c_3c_4\end{bmatrix},\ \mathbf{\Psi}_{12\times 1}=[\mathbf{0}_{4\times 1},\boldsymbol{\gamma}_{4\times 1},\mathbf{0}_{4\times 1}]^{\mathrm{T}}=\left[\begin{array}{c}0\\0\\0\\0\\\hline c_0c_7\\c_1c_6\\c_2c_5\\c_3c_4\\\hline 0\\0\\0\\0\end{array}\right],$$

$$\mathbf{P}_{16} = \mathbf{I}_8 \oplus \mathbf{J}_4 \oplus \mathbf{I}_4 =$$

$$\mathbf{P}_{16} = \begin{bmatrix} 1 & & & & \mathbf{0}_4 & & & & \mathbf{0}_8 & & & & \\ & 1 & & & & & & & & & & & \\ & & 1 & & & & & & & & & & \\ & & & 1 & & & & & & & & & \\ \hline & & & & 1 & & & & & & & & \\ & & & & & 1 & & & & & & & \\ & & & & & & 1 & & & & & & \\ \mathbf{0}_4 & & & & & & & 1 & & & & & \\ \hline & & & & & & & & 1 & & & & \\ & & & & & & & & & 1 & & & \\ & & & & & & & & & & 1 & & \\ \mathbf{0}_8 & & & & & & & & \mathbf{0}_4 & & & 1 & \\ & & & & & & & & & & & & 1 \end{bmatrix},$$

$$\mathbf{P}_{16\times 12} = \mathbf{1}_{2\times 1} \otimes \mathbf{I}_8 = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \\ \hline 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix}.$$

Fig. 2 shows a data flow diagram, which describes the rationalized algorithm for the implementation of IDWT basic operation for case $L = 8$.

The proposed algorithms allow the total number of multiplications to be reduced, relative to the direct method. The number of multiplications in the algorithm for the implementation of FDWT/IDWT basic operations represented by the procedure (3) is $1{,}5L$. On the other hand, the algorithm for the implementation of FDWT basic operation, represented by the procedure (3) requires only $(3{,}5L+1)$ additions, instead of $2(L-1)$ in the direct algorithm. The number of multiplications in the algorithm for the implementation of IDWT basic operations represented by the procedure (5) is $L+1$. In turn, the algorithm for the implementation of IDWT basic operation, represented by the procedure (5) requires $3{,}5L$ additions.

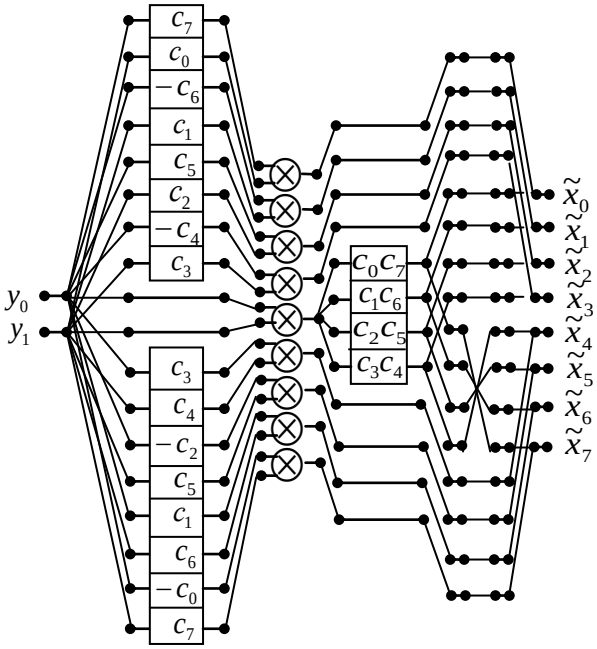

Fig 2: Data flow diagram for IDWT basic operation ralization according to the procedure (6).

III. Rationalized Algorithm For FDWT/IDWT Basic Operations Execution Using Gauss' Trick For 2×2 Matrix Factorization

A. Short background

Let us rearrange the columns of the matrix $\mathbf{F}_{2\times L}$ so that a new matrix takes the following form:

$$\tilde{\mathbf{F}}_{2\times L} = \mathbf{F}_{2\times L}\mathbf{P}_L$$

where

$\mathbf{P}_L = \|p_{ij}\|$ is a permutation matrix whose elements are defined as follows:

$$\begin{cases} p_{2i+1,2i+1} = 1, i = \overline{0, \frac{L}{2}-1}, \\ p_{2(i+1),L-2i} = 1, i = \overline{0, \frac{L}{2}-1} \end{cases}.$$

The computation process for the multiplication of matrix $\tilde{\mathbf{F}}_{2\times L}$ by vector $\mathbf{X}_{L\times 1}$ can be implemented as $L/2$ independent vector-matrix products with 2×2 matrices. The results of these calculations should be later added. We will notice that all sub blocks of the new matrix $\tilde{\mathbf{F}}_{2\times L}$ possess specific block structures. This specificity, as we show below, allows the reduction of the number of multiplications in the implementation of the partial vector-matrix products. The mentioned possibility of rationalization uses the following decomposition [13]:

$$\begin{bmatrix} a & b \\ c & a \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} c-a & 0 & 0 \\ 0 & b-a & 0 \\ 0 & 0 & a \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

As can be seen, each of such vector-matrix multiplications requires only three multiplications and five additions. The use of singularities can offer a more cost-effective way to obtain a set of partial vector-matrix products. Below the application of this trick is considered in detail.

B. Rationalized algorithm for FDWT basic operation implementation using Gauss's matrix factorization trick

At first we extract elements from $\mathbf{F}_{2\times L}$ matrix to create a new diagonal matrix:

$$\mathbf{D}_{\frac{3L}{2}} = \bigoplus_{i=0}^{\frac{L}{2}-1} \mathbf{D}_3^{(i)},$$

where submatrices

$\mathbf{D}_3^{(i)} = diag[(h_{(L-1)-2i} - h_{2i}), (h_{(L-1)-2i} + h_{2i}), h_{2i}]$ are 3×3 diagonal matrix.

Next we introduce three types of summation matrices:

$\mathbf{A}_{\frac{3L}{2}\times L} = (\mathbf{I}_{\frac{L}{2}} \otimes \mathbf{T}_{3\times 2})$ - matrix of pre-additions,

$\mathbf{A}_{L\times\frac{3L}{2}} = (\mathbf{I}_{\frac{L}{2}} \otimes \mathbf{T}_{2\times 3})$ - matrix of post-additions,

$\mathbf{A}_{3\times\frac{3L}{2}} = (\mathbf{1}_{1\times\frac{L}{2}} \otimes \mathbf{I}_3)$ - partial results summation matrix,

matrix $\mathbf{T}_{2\times 3} = \begin{bmatrix} & 1 & 1 \\ 1 & & 1 \end{bmatrix}$

and

$$\mathbf{T}_{3\times 2} = \begin{bmatrix} 1 & \\ & 1 \\ 1 & -1 \end{bmatrix}.$$

Taking into account introduced vector-matrix constructions, the FDWT basic operation computational procedure with a reduced number of multiplications can be represented as follows:

$$\mathbf{Y}_{2\times 1}^{(l)} = \mathbf{T}_{2\times 3}\mathbf{A}_{3\times\frac{3L}{2}}\mathbf{D}_{\frac{3L}{2}}\mathbf{A}_{\frac{3L}{2}\times L}\mathbf{P}_L\mathbf{X}_{L\times 1}^{(l)} \quad (7)$$

We consider against an example for $L = 8$.
Then

$$\mathbf{Y}_{2\times 1}^{(l)} = \mathbf{T}_{2\times 3}^{(3)}\mathbf{A}_{3\times 12}\mathbf{D}_{12}\mathbf{A}_{12\times 8}\mathbf{P}_8\mathbf{X}_{8\times 1}^{(l)} \quad (8)$$

Thus the previously introduced vector-matrix structures will have the following form:

$$\mathbf{P}_8 = \begin{bmatrix} 1 & & & & & & & \\ & & & & & & & 1 \\ & & 1 & & & & & \\ & & & & & 1 & & \\ & & & & 1 & & & \\ & & & 1 & & & & \\ & & & & & & 1 & \\ & 1 & & & & & & \end{bmatrix},$$

$$\mathbf{A}_{12\times 8} = \left[\begin{array}{cccccccc} 1 & & & & & & & \\ & 1 & & & & & & \\ 1 & -1 & & & & & & \\ \hdashline & & 1 & & & & & \\ & & & 1 & & & & \\ & & 1 & -1 & & & & \\ \hdashline & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & 1 & -1 & & \\ \hdashline & & & & & & 1 & \\ & & & & & & & 1 \\ & & & & & & 1 & -1 \end{array}\right],$$

$$\mathbf{D}_{12} = \bigoplus_{i=0}^{3} \mathbf{D}_3^{(i)},$$

$$\mathbf{D}_3^{(i)} = diag[(c_{7-2i} - c_{2i}), (c_{7-2i} + c_{2i}), c_{2i}],$$

$$\mathbf{D}_{12} = diag(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}),$$

$s_0 = c_7 - c_0$, $s_1 = c_7 + c_0$, $s_2 = c_0$, $s_3 = c_5 - c_2$, $s_4 = c_5 + c_2$, $s_5 = c_2$, $s_6 = c_3 - c_4$, $s_7 = c_3 + c_4$, $s_8 = c_4$, $s_9 = c_1 - c_6$, $s_{10} = c_1 + c_6$, $s_{11} = c_6$,

$$\mathbf{A}_{3\times 12} = \left[\begin{array}{cccccccccccc} 1 & & & 1 & & & 1 & & & 1 & & \\ & 1 & & & 1 & & & 1 & & & 1 & \\ & & 1 & & & 1 & & & 1 & & & 1 \end{array}\right].$$

C. Rationalized algorithm for IDWT basic operation implementation using Gauss'matrix factorization trick

The basic operation computational procedure takes the following form:

$$\tilde{\mathbf{X}}_{L\times 1}^{(l)} = \mathbf{P}_L\mathbf{A}_{L\times\frac{3L}{2}}\mathbf{D}_{\frac{3L}{2}}\mathbf{P}_{\frac{3L}{2}\times 3}\mathbf{T}_{3\times 2}\mathbf{Y}_{2\times 1}^{(l)} \quad (9)$$

Where

$$\mathbf{A}_{L\times\frac{3L}{2}} = \mathbf{I}_{\frac{L}{2}} \otimes \mathbf{T}_{2\times 3},\ \mathbf{P}_{\frac{3L}{2}\times 3} = \mathbf{A}^{\mathrm{T}}_{3\times\frac{3L}{2}}.$$

For our example the matrices $\mathbf{P}_8$, $\mathbf{D}_{12}$, $\mathbf{P}_{12\times3}=\mathbf{A}_{3\times12}^{\mathrm{T}}$, $\mathbf{T}_{3\times2}$, and $\mathbf{T}_{2\times3}$ have been defined above, just as the matrix $\mathbf{A}_{L\times\frac{3L}{2}}$ is defined as follows:

$$\mathbf{A}_{8\times12}=\left[\begin{array}{cccccccccccc}1&&1&&&&&&&&&\\&1&1&&&&&&&&&\\\hline&&&1&&1&&&&&&\\&&&&1&1&&&&&&\\\hline&&&&&&1&&1&&&\\&&&&&&&1&1&&&\\\hline&&&&&&&&&1&&1\\&&&&&&&&&&1&1\end{array}\right].$$

Then, for this example, the IDWT basic operation computational procedure takes the following form:

$$\widetilde{\mathbf{X}}_{8\times1}^{(l)}=\mathbf{P}_8\mathbf{A}_{8\times12}\mathbf{D}_{12}\mathbf{P}_{12\times3}\mathbf{T}_{3\times2}\mathbf{Y}_{2\times1}^{(l)} \qquad (10)$$

The data flow diagrams for the execution of FDWT and IDWT basic operations in accordance with the procedures (8) and (10), are shown in Figures (3) and (4), respectively.

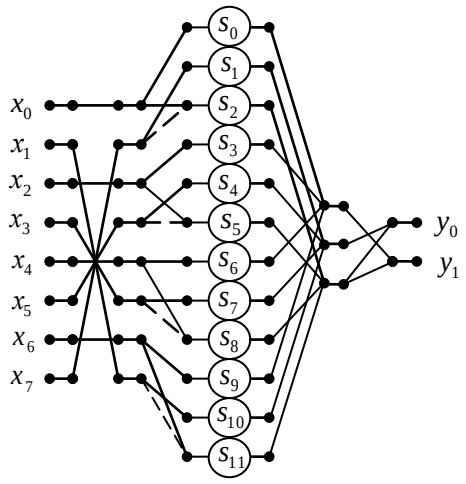

Fig 3: Data flow diagram for FDWT basic operation ralization according to the procedure (8).

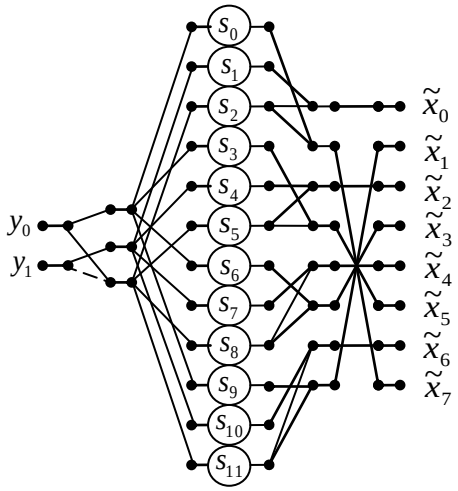

Fig 4: Data flow diagram for IDWT basic operation ralization according to the procedure (10).

The number of multiplications in the algorithm for the implementation of FDWT/IDWT basic operations, represented by the procedure (7), is $1{,}5L$. On the other hand, the algorithm for the implementation of FDWT basic operation, represented by the procedure (7), requires only $(2L-1)$ additions, instead of $2(L-1)$ in the direct algorithm. The number of multiplications in the algorithm for the implementation of IDWT basic operations represented by the procedure (9), is $1{,}5L$. In turn, the algorithm for the implementation of IDWT basic operation, represented by the procedure (9), requires $L+1$ additions.

IV. Conclusion

We see that the solutions proposed in this article allow to reduce the total number of multiplications in the implementation of FDWT/IDWT basic operations compared to the naive methods of computing. Indeed, in the general case a fully parallel implementation of FDWT/IDWT basic procedures requires $2L$ multiplications. The number of multiplications for each of proposed here algorithms is 25% less than that of the direct execution of computations.

It is noteworthy that, because all elements in $\mathbf{F}_{2\times L}$ matrix are constants, we can (but not must) use one-input units (encoders), instead of traditional multipliers. In such case, it is apparently advisable to use the second approach for the implementation of the FDWT/IDWT basic operations. This solution greatly simplifies implementation, reduces the power dissipation and lowers the price of the device. On the other hand, when we are dealing with FPGA chips that already contain a number of embedded multipliers, the construction and usage of additional encoders instead of multipliers is irrational. In this case, it would be unreasonable to refuse the possibility of using embedded multipliers. In this case, any algorithms derived from the application of both approaches can be used with approximately equal effect.

The algorithms proposed in this article allow the number of multiplications to be reduced at the cost of more additions, or more complex memory access. Such solutions make no sense in some modern high-speed architectures, where pipelined fixed-point or floating-point addition and multiplication take just one clock cycle. Therefore, the solutions presented here are intended solely for the hardware implementation of FDWT/IDWT basic operations.

REFERENCES

[1] Mallat S. G., A theory for multiresolution signal decomposition: The wavelet representation, IEEE Trans. Patt. Anal. Mach. Intell., vol. 11, pp. 674-693, July 1989.

[2] Daubechies I., Ten lectures on Wavelets, SIAM, Philadelphia, PA, 1992.

[3] Chui C. K., Montefusco L., Puccio L. Wavelets: Theory, Algorithms and Applications, Academic Press, New York, 1994.

[4] Vetterli M., Kovačević J. Wavelets and Subband Coding, Prentice Hall PTR, Englewood Cliffs, 1995.

[5] Stollnitz E. J., DeRose A. D., Salesin D. H. Wavelets for Computer Graphics, Morgan Kaufmann, 1996.

[6] Burrus C. S., Gopinath R. A. Intoduction to Wavelets and Wawelets Transforms: A Primer, Prentice Hall, New Jersey, 1998.

[7] Goswami J. C., Chan A. K. Fundamentals of Wavelet: Theory, Algorithms and Applications. Wiley-Interscience, New York, 1999.

[8] Debnath L. Wavelet Transforms and Their Applications, Birkhauser, 2001.

[9] Frazier M. W., An Introduction to Wavelets through Linear Algebra, Springer-Verlag, New York, 2001.

[10] Cohen A. Numerical Analysis of Wavelet Methods. Studies in Mathematics and Its Applications, Elsevier Science B.V. Printed in the Netherlands, 2003.

[11] Weeks M., Bayoumi M. Discrete Wavelet Transforms: Architectures, Design and Performance Issues, Journal of VLSI Signal Processing, 2003, no. 35, pp. 155-178.

[12] Ţariov A., Ţariova G., Majorkowska-Mech D. Algorithms for multilevel decomposition and reconstruction of Digital signals, Commission of Informatics, Polish Academia of Science (Gdansk branch) Press, 2012. (in Polish).

[13] Ţariov A. Algorithmic aspects of computing rationalization in digital signal processing". West Pomeranian University of Technology Press, (2011), 232 p. (in Polish).

[14] Winograd S. A new algorithm for inner product. IEEE Trans. Computers, vol. C-17, pp. 693-694, 1968.

[15] Regalia Ph. A. and Mitra S. K. Kronecker Products, Unitary Matrices and Signal Processing Applications, SIAM Review, vol. 31, no. 4, pp. 586-613, 1989.

Aleksandr Cariow, received the Candidate of Sciences and Doctor of Sciences degrees (Habilitation) in Computer Sciences from LITMO of St. Petersburg, Russia in 1984 and 2001. In September 1999, he joined the Faculty of Computer Sciences and Information Technology, at the West Pomeranian University of Technology, Szczecin, Poland, where he is currently a Professor and Chair of the Department of Computer Architectures and Telecommunications. His research interests include digital signal and image processing and transmission, fast computational algorithms, DSP VLSI architectures, and data processing parallelization.

Galina Cariowa, received the MSc degrees in mathematics from Moldavian State University, Chişinău in 1976 and PhD degree in computer science from West Pomeranian University of Technology, Szczecin, Poland in 2007. She is currently working as an assistant professor in the Department of Multimedia Systems. Her research interests include numerical mathematics, digital signal and image processing, fast computational algorithms for multimedia applications.

How to cite this paper: Aleksandr Cariow, Galina Cariowa,"Algorithmic Tricks for Reducing the Complexity of FDWT/IDWT Basic Operations Implementation", IJIGSP, vol.6, no.10, pp.1-9, 2014.DOI: 10.5815/ijigsp.2014.10.01

