A Scalable Formulation For Multidimensional Walsh-Hadamard Transform

AYMAN ELNAGGAR

Department of Information Engineering,
Sultan Qaboos University,
P.O. Box 33, Muscat 123,
OMAN

MOKHTAR ABOELAZE

Department of Computer Science,
York University,
Toronto, Ontario, M3J 1P3,
CANADA

Abstract: - This paper presents a new recursive formulation for Walsh-Hadamard Transform (WHT) that allows the generation of higher order (longer size) multidimensional (*m*-d) WHT architectures from 2^m lower order (shorter sizes) WHT architectures. Our methodology is based on manipulating tensor product forms so that they can be mapped directly into modular parallel architectures. The resulting WHT circuits have very simple modular structure and regular topology.

Key-Words: - Tensor Product – WHT – Multidimensional Transforms – Modular Structures – Recursive Formulations – Permutation Matrices

1 Introduction

The Walsh-Hadamard Transform (WHT) has been used in many DSP, image, and video processing applications such as filter generating systems [9], block orthogonal transforms (BOTs) [5], and block wavelet transforms [2]. Other applications in communications are in CDMA [1] and spread spectrum [6].

This paper proposes an efficient and cost-effective methodology for mapping WHT onto VLSI structures. The main objective of this paper is to derive a design methodology and recursive formulation for the multidimensional (*m*-d) WHT which is useful for the true modularization and parallelization of the resulting computation.

The main result reported in this paper shows that a large two-dimensional **2-d** WHT computation on an $n \times n$ input image can be decomposed recursively into three stages as shown in Fig. 1 for the case $n = 4$.

The second stage is constructed recursively from $\mathbf{2^2}$ parallel (data-independent) blocks each realizing a smaller-size WHT. The pre-additions and the post-permutations stages serve as "glue" circuits that combine the $\mathbf{2^2}$ lower order WHT blocks to construct the higher order WHT architecture. We also show that the proposed algorithm can be extended such that an ***m*-d** WHT can be constructed from $\mathbf{2^m}$ smaller size *m*-d WHTs. Observe that, we have drawn our networks such that data flows from right to left. We chose this convention to show the direct correspondence between the derived algorithms and the proposed VLSI networks.

Although, as far as we know from the literature, the recursive 1-d WHT algorithm is widely presented in the literature [8], [11], neither the proposed *m*-d WHT algorithm nor the modular forms were previously derived.

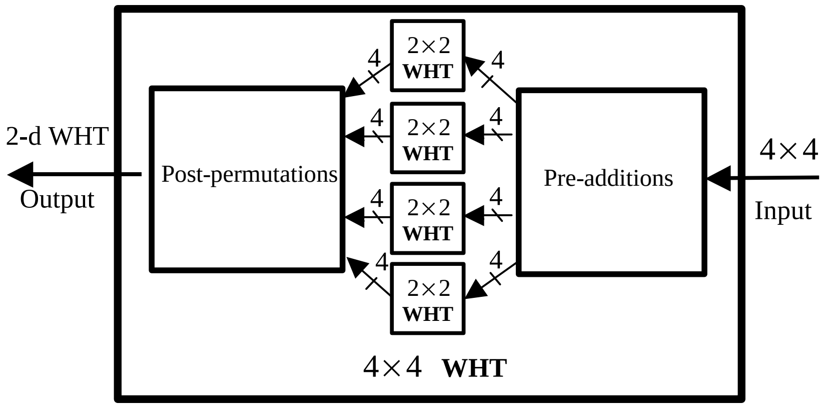

Fig. 1 The proposed 2-d WHT recursive realization for a 4×4 input image

Our work is based on a non-trivial generalization of the 1-d WHT using tensor product. When coupled with permutation matrices, tensor products provide a unifying framework for describing a wide range of fast recursive algorithms for various transform [3], [4], [10].

Some of the tensor product properties that will be used throughout this paper are [3], [10]:

$$AB \otimes CD = (A \otimes B)(C \otimes D) \quad (1)$$

$$(A \otimes B) \otimes C = A \otimes (B \otimes C) \quad (2)$$

If $n = n_1 n_2$, then

$$A_{n_1} \otimes B_{n_2} = P_{n,n_1}(I_{n_2} \otimes A_{n_1})\, P_{n,n_2}(I_{n_1} \otimes B_{n_2}) \quad (3)$$

If $n = n_1 n_2 n_3$, then

$$I_{n_1} \otimes A_{n_2} \otimes I_{n_3} = P_{n,n_1 n_2}(I_{n_1 n_3} \otimes A_{n_2}) P_{n,n_3} \quad (4)$$

If $2n = n_1 n_2$, then

$$P_{n,2} = P_{n,n_1} P_{n,n_2} \quad (5)$$

Where $\otimes$ denotes the tensor product, I_n is the identity matrix of size *n*, and $P_{n,s}$ is an $n \times n$ binary matrix specifying an *n*/*s* shuffle (or *s*-stride) permutation.

2 The Modified Formulations of the 1-d WHT

In this section, we modify the original 1-d WHT to the iterative form that allows a hardware saving without affecting the processing speed.
The original 1-d WHT transform matrix is defined as [8], [11]

$$W_n = \begin{pmatrix} W_{n/2} & W_{n/2} \\ W_{n/2} & -W_{n/2} \end{pmatrix} \quad , \quad W_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (6)$$

where W_2 is the 2-point WHT.

2.1 The 1-d WHT Iterative Formulation

Let $k = \log_2 n$, we can write equation (6) in the iterative tensor-product form

$$\begin{aligned} W_n &= W_2 \otimes W_{n/2} = W_2 \otimes W_2 \otimes \cdots \otimes \cdots W_2 \\ &= \prod_{i=0}^{k-1} (I_{2^i} \otimes W_2 \otimes I_{2^{k-i-1}}) \end{aligned} \quad (7)$$

which using property (4), can be modified to

$$W_n = \prod_{i=0}^{k-1} P_{n,2^{i+1}}\,(I_{2^{k-1}} \otimes W_2)\, P_{n,2^{k-i-1}} \quad (8)$$

As an example, we can express W_8 as

$$\begin{aligned} W_8 = &\left[P_{8,2}\,(I_4 \otimes W_2)\, P_{8,4}\right]. \\ &\left[P_{8,4}\,(I_4 \otimes W_2)\, P_{8,2}\right]\left[P_{8,8}\,(I_4 \otimes W_2)\, P_{8,1}\right] \end{aligned} \quad (9)$$

The realization of W_8 is shown in Fig. 2 (a).
Applying property (5) to equation (9), now the adjacent permutations $P_{8,2}\ P_{8,8}$ (from the first and the second stages) will be replaced by the single permutation $P_{8,2}$ and the adjacent permutations $P_{8,4}\ P_{8,4}$ (from the second and the third stages) will be replaced by the single permutation $P_{8,2}$ as shown in Fig. 2 (b).
Similarly, equation (8) can be simplified to

$$W_n = \prod_{i=0}^{k-1} P_{n,2}\,(I_{2^{k-1}} \otimes W_2) \quad (10)$$

Thus, W_n can be computed by the cascaded product of *k* similar stages (independent of *i*) of double matrix products instead of the triple matrix products in equation (8). Alternatively, we can realize (10) by a single block of $P_{n,2}\,(I_{2^{k-1}} \otimes W_2)$ and take the output after k iterations as shown in Fig. 3 for the case $n = 8$.

2.2 The 1-d WHT Recursive Formulation

Applying property (1), equation (7) can be modified to

$$\begin{aligned} W_n &= W_2 \otimes W_{n/2} = I_2\, W_2 \otimes W_{n/2}\, I_{n/2} \\ &= (I_2 \otimes W_{n/2})\,(W_2 \otimes I_{n/2}) \\ &= (I_2 \otimes W_{n/2})\, Q_n \end{aligned} \quad (11)$$

where

$$Q_n = (W_2 \otimes I_{n/2}) \quad (12)$$

Equation (11) represents the two-stage recursive tensor product formulation of the 1-d WHT in which the first stage is the pre-additions (Q_n), followed by the second stage of the core computation $(I_2 \otimes W_{n/2})$ that consists of a parallel stage of two identical smaller WHT computations each of size $n/2$ as shown in Fig.4.

3 The Proposed Formulation of the 2-d WHT

This section will develop two recursive methodologies for realizing the 2-d WHT computations from smaller WHT computations. First, we present a direct method for realizing the 2-d WHT computations using the conventional row-column decomposition of the 1-d WHT in a tensor product form. The second methodology provides a truly 2-d recursive structures that employ one stage of smaller 2-d WHTs to realize the large 2-d WHT.

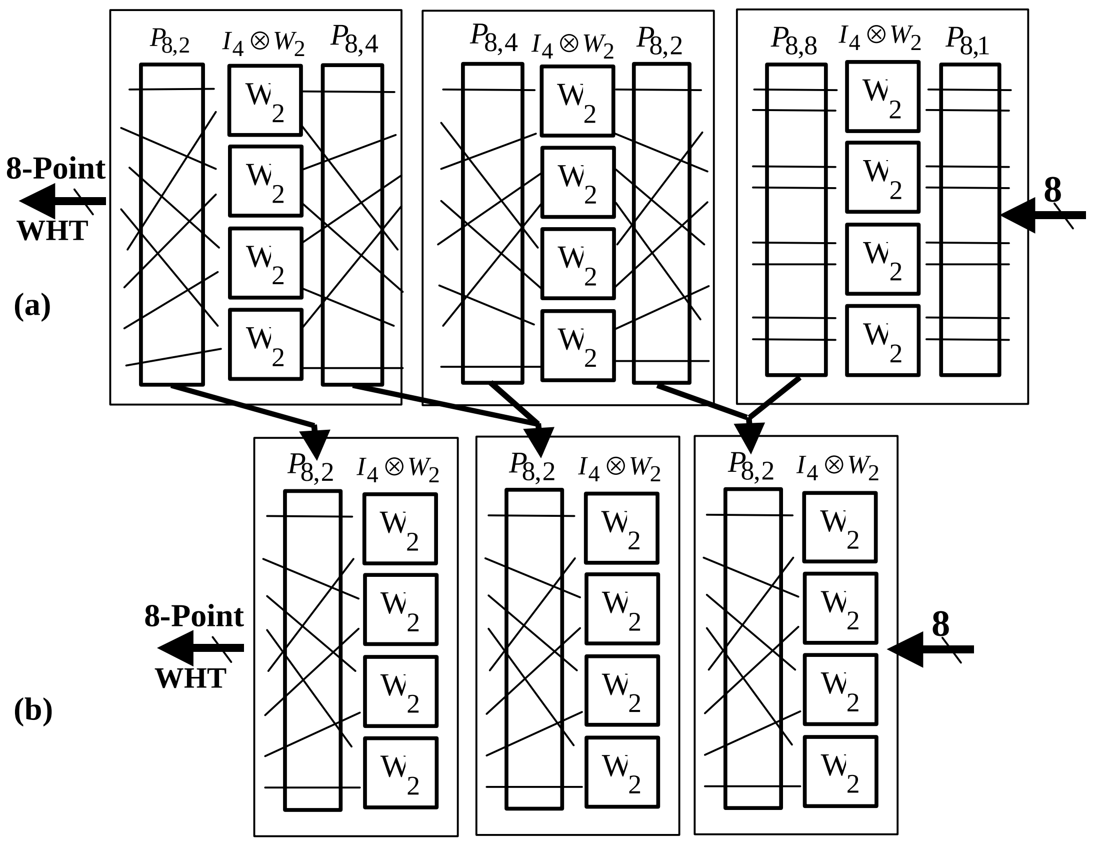

Fig. 2 The realization of W_8 : **(a)** The original 1-d WHT iterative algorithm. **(b)** The modified 1-d WHT algorithm.

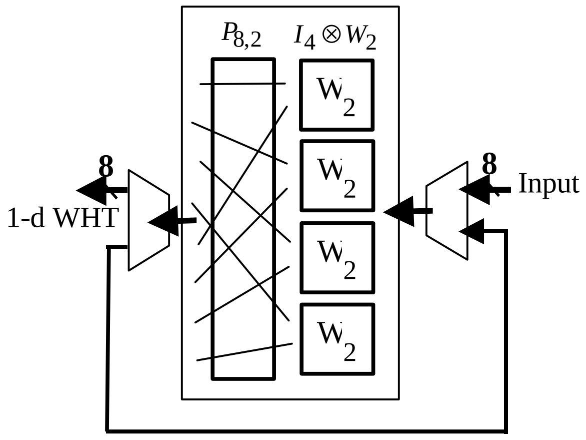

Fig. 3 The reduced hardware realization of the modified 1-d WHT algorithm

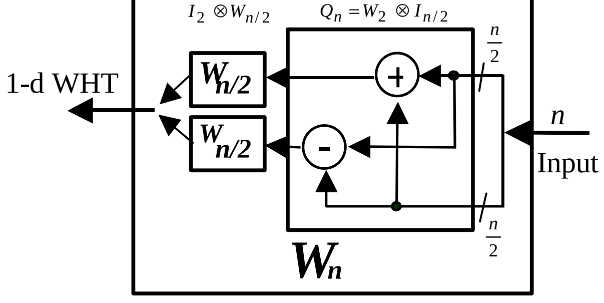

Fig. 4 The realization of the recursive 1-d WHT

3.1 Conventional Row-Column Decomposition

Since the WHT matrix is separable [7], the 2-d WHT for an input image of dimension $n_1 \times n_2$ can be computed by a stage of n_2 parallel 1-d WHT computations on n_1 points each, followed by another stage of n_1 parallel 1-d WHT computations on n_2 points each. This can be represented by the matrix-vector form

$$X = W_{n_1,n_2} x , \tag{13}$$

where W_{n_1,n_2} is the 2-d WHT transform matrix for an $n_1 \times n_2$ image, X and x are the output and input column-scanned vectors, respectively.

For separable transforms, the matrix W_{n_1,n_2} can be represented by the tensor product form [7]

$$W_{n_1,n_2} = W_{n_1} \otimes W_{n_2} \tag{14}$$

where W_{n_1} and W_{n_2} are the row and column 1-d WHT operators, as defined by equation (7), on *x*, respectively.

By substituting (14) in (13), we have

$$X = (W_{n_1} \otimes W_{n_2})\, x . \tag{15}$$

Which using equation (7) can be expressed as

$$X = W_{n_1 \times n_2}\, x \tag{16}$$

Therefore, the 2-d WHT on an $n_1 \times n_2$ input is equivalent to a 1-d WHT on a 1-d input vector of size $n_1 \times n_2$ that can be implemented using either the modified 1-d iterative algorithm given by (10) or the modified 1-d recursive algorithm given by (11).

3.2 The Truly Recursive Formulation of the 2-d WHT

Now we will derive a truly 2-d recursive formulation of the WHT by further manipulation of equation (15). Substituting (11) in (15), the 2-d WHT transform matrix can be written as

$$\begin{aligned} W_{n_1,n_2} &= [(I_2 \otimes W_{n_1/2})\, Q_{n_1}] \otimes \\ &\quad [(I_2 \otimes W_{n_2/2})\, Q_{n_2}] \end{aligned} \tag{17}$$

Applying property (1), we can write W_{n_1,n_2} as

$$\begin{aligned} W_{n_1,n_2} &= [(I_2 \otimes W_{n_1/2}) \otimes (I_2 \otimes W_{n_2/2})] \\ &\quad [Q_{n_1} \otimes Q_{n_2}] \\ &= [C_{n_1,n_2}\; Q_{n_1,n_2}] \end{aligned} \tag{18}$$

where

$$\begin{aligned} C_{n_1,n_2} &= [(I_2 \otimes W_{n_1/2}) \otimes (I_2 \otimes W_{n_2/2})], \\ Q_{n_1,n_2} &= [Q_{n_1} \otimes Q_{n_2}] \end{aligned} \tag{19}$$

Now, from property (2), we can write C_{n_1,n_2} in the form

$$C_{n_1,n_2} = (I_2 \otimes W_{n_1/2} \otimes I_2) \otimes W_{n_2/2} \tag{20}$$

Applying property (4), we can write (20) in the form

$$\begin{aligned} C_{n_1,n_2} &= [P_{2n_1,n_1}(I_4 \otimes W_{n_1/2})\; P_{2n_1,2}] \otimes [W_{n_2/2}] \\ &= [P_{2n_1,n_1}(I_4 \otimes W_{n_1/2})\; P_{2n_1,2}] \otimes \\ &\quad [I_{n_2/2}\; W_{n_2/2}\; I_{n_2/2}] \\ &= [P_{2n_1,n_1} \otimes I_{n_2/2}][(I_4 \otimes W_{n_1/2}) \otimes W_{n_2/2}] \\ &\quad [P_{2n_1,2} \otimes I_{n_2/2}] \\ &= [P_{2n_1,n_1} \otimes I_{n_2/2}][(I_4 \otimes W_{n_1/2,\, n_2/2}] \\ &\quad [P_{2n_1,2} \otimes I_{n_2/2}] \end{aligned} \tag{21}$$

where

$$W_{n_1/2\,,\, n_2/2} = W_{n_1/2} \otimes W_{n_2/2} \tag{22}$$

Finally, substituting (21) in (18), we have

$$W_{n_1,n_2} = [\tilde{R}_{n_1,n_2}\,(I_4 \otimes W_{n_1/2\,,\,n_2/2})\,\tilde{Q}_{n_1,n_2}] \tag{23}$$

where

$$\begin{aligned} \tilde{Q}_{n_1,n_2} &= (P_{2n_1,2} \otimes I_{n_2/2})\; Q_{n_1,n_2} , \\ \tilde{R}_{n_1,n_2} &= (P_{2n_1,n_1} \otimes I_{n_2/2}) . \end{aligned} \tag{24}$$

Equation (23) represents the truly recursive 2-d WHT in which $\tilde{Q}_{n_1,n_2}$ and $\tilde{R}_{n_1,n_2}$ are the pre- and post-processing glue structures, respectively, that combine 2^2 identical lower-order 2-d WHT modules each of size $n_1/2 \times n_2/2$ in parallel, to construct the higher order 2-d WHT of size $n_1 \times n_2$ as shown previously in Fig.1.

4 The Multi-Dimensional WHT

We can extend the 2-d WHT derivation proposed in Section 2 to the *m*-d case. From (14), the *m*-d WHT can be written in the tensor product form

$$\begin{aligned} X &= (W_{n_1} \otimes W_{n_2} \otimes \cdots \otimes W_{n_m})\, x \\ &= (\bigotimes_{i=1}^{m} W_{n_i})\, x \end{aligned} \tag{25}$$

Where, $(W_{n_1} \otimes W_{n_2} \otimes \cdots \otimes W_{n_m})$ is the *m*-d WHT transform matrix for an *m*-d input, W_{n_i} is the 1-d WHT coefficient matrix for an input vector of length n_i as defined in (7), *X* and *x* are the output and input column-scanned vectors, respectively.

Following the steps in Section 2, we can write (25) in the form

$$X = [\hat{R}\,(I_{2^m} \bigotimes_{i=1}^{m} W_{n_i/2})\,\hat{Q}]\; x \quad , \qquad (26)$$

where $\bigotimes_{i=1}^{m} W_{n_i/2}$ is the lower order *m*-d WHT and

$$\hat{Q} = \left(\prod_{i=1}^{m-1} \left(P_{u_1,u_2} \bigotimes_{k=1}^{i} I_{u_3} \right) \right) \left(\bigotimes_{i=1}^{m} Q_i \right) \quad ,$$

$$\hat{R} = \prod_{i=1}^{m-1} \left(P_{w_1,w_2} \bigotimes_{l=i}^{m-1} I_{w_3} \right) \quad ,$$

$$u_1 = 2 \prod_{j=1}^{m-i} n_j \quad , \quad u_2 = 2 ,$$

$$u_3 = \frac{1}{2} \prod_{j=1}^{i} (n_{m-j+1}),$$

$$w_1 = \frac{1}{2} \prod_{k=1}^{i} (n_k),$$

$$w_2 = \prod_{k=1}^{i} (n_k),$$

$$w_3 = \frac{1}{2} \prod_{j=i+1}^{i} (n_j) ,$$

Q_i is the 1-d pre-processing as defined by (12) Equation (26) extends our results by showing that a large *m*-d WHT can be computed from a single stage of smaller *m*-d WHTs.

5 Conclusion

In this paper, we proposed a new recursive formulation for Walsh-Hadamard Transform (WHT) that allows the generation of higher order (longer size) multidimensional (*m*-d) WHT architectures from 2^m lower order (shorter sizes) WHT architectures in a parallel realization with the addition of two glue stages of pre-additions and post-permutations. The resulting networks have a very simple modular structure and highly regular topology.

References:

[1] F. Adachi, K. Ohno, A. Higashi, T. Dohi, and Y. Okumura, "Coherent Multicode DS-CDMA Mobile Radio Access", *IEICE Trans. on Communications*, Vol. E79-B, 1996, pp. 1316-1325.

[2] A. E. Cetin, O. N. Gerek, and S. Ulukus, "Block Wavelet Transforms for Image Coding", *IEEE Trans. on Circuits and systems for Video Technology*, Vol. 3, 1993, pp. 433-435.

[3] A. Elnaggar and M. Aboelaze, "An Efficient Architecture for Multi-Dimensional Convolution", *IEEE Trans. on Circuits and Systems II*, Vol. 47, No. 12, 2000, pp. 1520-1523.

[4] A. Elnaggar and M. Aboelaze, “A Modified Shuffle Free Architecture for Linear Convolution, *IEEE Trans. on Circuits and Systems II*, Vol. 48, No. 9, 2001, pp. 862-866.

[5] A. Makur, "BOT’s Based on Non-uniform Filter Banks", *IEEE Trans. on Signal Processing*, Vol. 44, 1996, pp. 1971-1981.

[6] R. L. Peterson, R. E. Ziemer, and D. E. Borth, *Introduction to Spread Spectrum Communications*, Prentice Hall, 1995.

[7] W. K. Pratt, *Digital Image Processing*, John Wiley & Sons, Inc., 1991.

[8] S. Rahardja and B.J. Falkowski, “Family of unified complex Hadamard transforms”, *IEEE Trans. On Circuits and Systems II*, Vol. 46, 1999, pp. 1094-1100.

[9] S. Samadi, A. Nishihara and H. Iwakura, "Filter Generating Systems", *IEEE Trans. on Circuits and Systems II*, Vol. 47, No. 8, 2000, pp. 214-221.

[10] R. Tolimieri, M. An, C. Lu, *Algorithms for Discrete Fourier Transform and Convolution*, Springer-Verlag, New York 1989.

[11] Yarlagadda, RKE and Hershey, JE, *Hadamard Matrix Analysis and Synthesis With Applications to Communications and Signal/Image Processing*, Kluwer Academic Publishers, Boston 1997.