# *Memory-Efficient Architecture of 2-D Dual-Mode Discrete Wavelet Transform Using Lifting Scheme for Motion-JPEG2000*

Wei-Ming Li
Department of Electrical Engineering
Tamkang University
Tamsui, Taipei, Taiwan
E-mail: wmli@ee.tku.edu.tw

Chih-Hsien Hsia
Department of Electrical Engineering
Tamkang University
Tamsui, Taipei, Taiwan
E-mail: chhsia@ee.tku.edu.tw

Jen-Shiun Chiang
Department of Electrical Engineering
Tamkang University
Tamsui, Taipei, Taiwan
E-mail: chiang@ee.tku.edu.tw

*Abstract*—**In this work, we propose a memory-efficient architecture of lifting based two-dimensional discrete wavelet transform (2-D DWT) for motion-JPEG2000. The proposed 2-D DWT architecture consists of a 1-D row processor, internal memory, and a 1-D column processor. The main advantage of this 2-D DWT is to reduce the internal memory requirement significantly. For an *N*×*N* image, only 2*N* and 4*N* sizes of internal memory are required for the 5/3 and 9/7 filters, respectively, to perform the one-level 2-D DWT decomposition. Moreover, it supports both lossless and lossy operation for 5/3 and 9/7 filters with high operation speed. The proposed 2-D DWT surpasses the existed lifting-based designs in the aspects of low internal memory requirement. It is suitable for VLSI implementation and can support various real-time image/video applications such as JPEG2000, motion-JPEG2000, MPEG-4 still texture object decoding, and wavelet-based scalable video coding.**

## I. INTRODUCTION

Discrete wavelet transform (DWT) has been used for a wide range of applications, such as speech analysis, numerical analysis, signal analysis, image coding, pattern recognition, computer vision, and biometrics [1]. The DWT can be viewed as a multiresolution decomposition of a signal in which it decomposes a signal into several components in different frequency bands, requiring a large number of computation and large internal memory. Moreover, the DWT is a modern powerful tool for signal processing applications, such as JPEG2000 still image compression, denoising, region of interest, and watermarking. To achieve real-time processing, it is necessary to reduce the memory requirement and computational complexity to increase the hardware utilization efficiency. Generally, the realization of the 2-D DWT can be classified into two categories: convolution-based operation [1] and lifting-based operation (LDWT) [2]. Because the convolution-based implementation of DWT has high computational complexity and large memory requirements, the lifting-based implementation of DWT was proposed to overcome drawbacks of the convolution-based DWT [2]. The lifting-based scheme can provide a low-complexity solution for image/video compression applications, such as JPEG2000 [10], motion-JPEG2000 [9], MPEG-4 still image coding [10], and MC-EZBC [3].

Low transpose memory requirement is the major concerns in space-frequency domain implementation. Several VLSI architectures of 2-D LDWT have been proposed to reduce the transpose memory requirements and communication between the processors, such as the architectures presented in [11] and [12]. However, these hardware architectures still need a quantitative transpose memory. For 9/7 filter with an *N*×*N* image, [11] needs 22*N* and [12] needs 14*N* sizes of transpose memory. In order to reduce the transpose memory further, this paper proposes a new architecture to efficiently reduce the sizes of the internal memory to 2*N* (5/3 filter) and 4*N* (9/7 filter).

The rest of this paper is organized as follows. Section II reviews the discrete wavelet transform and lifting scheme for 9/7 filter. Section III describes the proposed one-level 2-D dual-mode DWT. In Section IV, the performance analysis for the proposed 2-D dual-mode architecture and comparison results with other architectures are illustrated. Section V gives a brief summary.

## II. DISCRETE WAVELET TRANSFORM AND LIFTING-BASED METHOD

The lifting-based scheme proposed by Daubechies and Sweldens requires fewer computations than the traditional convolution-based approach [2]. It is in integer operation but can avoid the problems caused by the finite precision or rounding. The Euclidean algorithm can factorize the poly-phase matrix of a DWT filter into a sequence of alternating upper and lower triangular matrices and a diagonal matrix. The variables *h*(*z*) and *g*(*z*) in (1) denote the low-pass and high-pass analysis filters, which can be divided into even and odd parts to generate a poly-phase matrix *P*(*z*) as in (2).

$$g(z)=g_e(z^2)+z^{-1}g_o(z^2),$$
$$h(z)=h_e(z^2)+z^{-1}h_o(z^2) \quad (1)$$

978-1-4244-3828-0/09/$25.00 ©2009 IEEE

$$P(z)=\begin{bmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{bmatrix} \tag{2}$$

The Euclidean algorithm recursively finds the greatest common divisors of the even and odd parts of the original filters. Since h(z) and g(z) form a complementary filter pair, P(z) can be factorized into (3):

$$P(z)=\prod_{i=1}^{m}\begin{pmatrix} 1 & si(z) \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ ti(z) & 1 \end{pmatrix}\begin{pmatrix} k & 0 \\ 0 & 1/k \end{pmatrix} \tag{3}$$

where *si*(*z*) and *ti*(*z*) are Laurent polynomials corresponding to the prediction and update steps, respectively, and $k$ is a nonzero constant. Therefore, the filter bank can be factorized into three lifting steps.

The lifting steps of the 9/7 filter are specified in JPEG2000 [5] and described from Eq. (4) to Eq. (11).

1. Splitting step:

$$d_i^0 = x_{2i+1}, \tag{4}$$

$$s_i^0 = x_{2i} \tag{5}$$

2. Lifting step:

(first lifting step)

$$d_i^1 = d_i^0 + \alpha\times(s_i^0 + s_{i+1}^0)\text{, (prediction step)} \tag{6}$$

$$s_i^1 = s_i^0 + \beta\times(d_{i-1}^1 + d_i^1)\text{ (update step)} \tag{7}$$

(Second lifting step)

$$d_i^2 = d_i^1 + \alpha\times(s_i^1 + s_{i+1}^1)\text{, (prediction step)} \tag{8}$$

$$s_i^2 = s_i^1 + \beta\times(d_{i-1}^2 + d_i^2)\text{ (update step)} \tag{9}$$

3. Scaling step:

$$d_i = K_2\times d_i^2 \tag{10}$$

$$s_i = K_1\times s_i^2 \tag{11}$$

Although the lifting-based scheme involves low complexity, the long and irregular signal paths are the major limitations for efficient hardware implementation. Additionally, the increasing number of the pipelined registers increases the internal memory size of the 2-D DWT architecture [5]. Generally, the 2-D DWT uses a vertical 1-D DWT subband decomposition and a horizontal 1-D DWT subband decomposition to obtain the 2-D DWT coefficients. Therefore, the memory requirement dominates the hardware cost and complexity of the architectures for 2-D DWT.

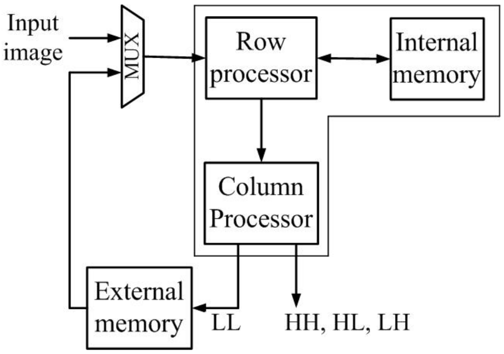

Figure 1. The one-level 2-D DWT architecture.

## III. Proposed One-Level 2-D DWT Architecture

Fig. 1 shows the block diagram of the proposed one-level 2-D DWT architecture. It consists of one 1-D row processor, internal memory, and one 1-D column processor. After the row processor performs the 1-D row-wise DWT operation, the row-processing coefficients are stored in the internal memory. Once enough row-processing coefficients are collected, the column processor performs the 1-D column-wise DWT operation. Both the row and column processors include 5/3 and 9/7 filters. Without loss of generality let us take the 9/7 filter as our example. The external memory is used to store the LL band output coefficients for the next decomposition operation. The details of the main components and the overall 2-D DWT architecture are discussed in the following subsections.

### A. Row Processor

Fig. 2 shows the row processor element. The row processor reads two input signals and writes two output signals, and it consists of two identical processing elements. Each processing element contains multiplexers, multipliers, adders, and registers. Bp, Ba, Bb, and Bc are the internal memory blocks used to store the original pixels and coefficients a, b, c, respectively. The details of the internal memory are discussed in next subsection.

Fig. 3 indicates the input sequence order of the row processor. The $x_{(i,j)}$ represents the position of the input signal. $i$ and $j$ represent the direction of the row and column, respectively. The input sequence order is from left to right, top to bottom. At every two continuous rows, two signals are input per clock cycle respectively except the first and the last rows that one signal is input per clock cycle. For example, at beginning the input sequence order is from $x_{(0,0)}$ to $x_{(0,7)}$. Next it starts from $x_{(1,0)}$ and $x_{(2,0)}$ to $x_{(1,7)}$ and $x_{(2,7)}$. When the input sequence order of the last row is finished, it starts from the next column and then repeats the identical manner as mentioned until the last element.

### B. Internal Memory

Because of the input sequence order of the row processor, the internal memory requirement of the *N*×*N* 2-D 5/3 mode DWT is 2*N*, and that of the 2-D 9/7 mode DWT is 4*N*. Fig. 4

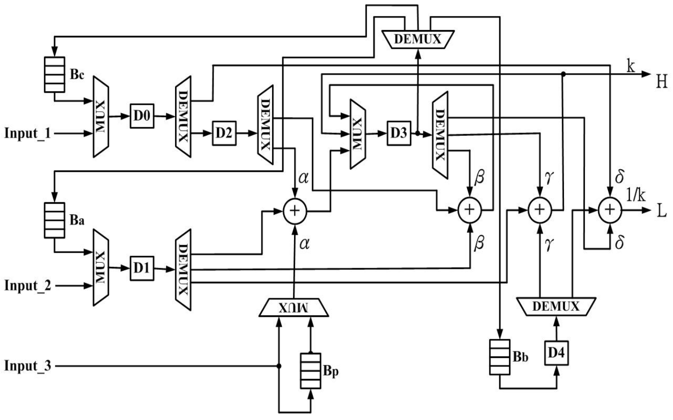

Figure 2. The row processor element.

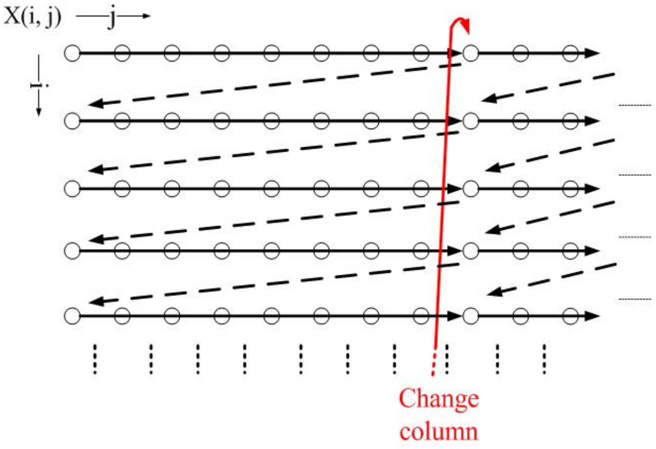

Figure 3. The input sequence order of the row processor.

shows the internal memory that is used to store the three coefficients *a*, *b*, *c*, and the original pixel *p*. *a*, *b*, and *c* are output after calculating equations (6), (7), and (8). When the input sequence order changes to next column, all of the new output coefficients are calculated by the overlapped coefficients and two original coefficients. For example, $a_{(0,2)}$ is calculated by the original pixels $p_{(0,4)}$, $p_{(0,5)}$, and $p_{(0,6)}$. $b_{(0,2)}$ is calculated by the overlapped coefficient $a_{(0,1)}$, the original pixel $p_{(0,4)}$, and the overlapped coefficient $a_{(0,2)}$. $c_{(0,1)}$ is calculated by the overlapped coefficients $b_{(0,1)}$, $a_{(0,1)}$, and $b_{(0,2)}$. $d_{(0,1)}$ is calculated by the overlapped coefficients $c_{(0,0)}$, $b_{(0,1)}$, and $c_{(0,1)}$. Since the overlapped coefficients are needed for calculating, the internal memory is required to store them for the row processing.

### C. Column Processor

Fig. 5 shows the column processor element. The column processor also reads two signals and writes two signals, and it consists of two identical processing elements. For the column processor, it also has overlapped problems, but only a few registers are used to store these overlapped coefficients instead of internal memory.

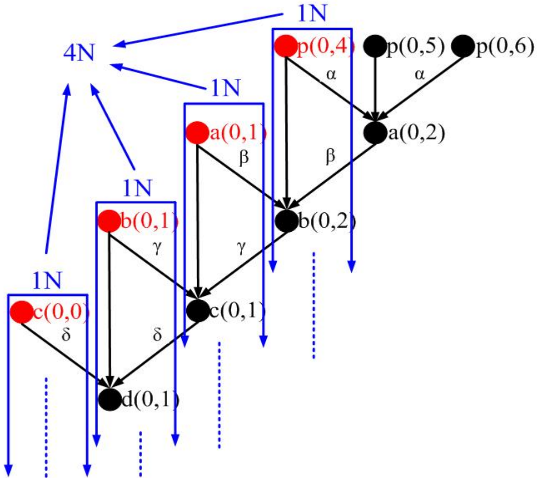

Figure 4. The diagram of the internal memory.

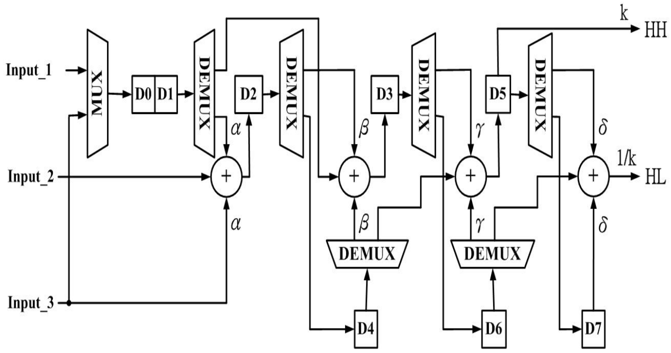

Figure 5. The column processor element.

Fig. 6 shows the input sequence order of the column processor. The column processor takes high-pass coefficients and low-pass coefficients as its inputs. Here let us take the high-pass coefficients as an example. $H_{(i,j)}$ represents the position of the high-pass coefficient; *i* and *j* represent the direction of the row and column, respectively. The input sequence order is from top to bottom, left to right. At beginning the input sequence order is from $H_{(0,0)}$ to the last one of the first column. Next it starts from $H_{(0,1)}$ and repeats the identical manner until the last one of the last column.

## IV. Experimental Results and Comparisons

The proposed 2-D DWT architecture has considered the trade-offs between low internal memory and low complexity in the VLSI implementation. The hardware cost and performance comparisons of our 2-D DWT 9/7 filter architecture and other similar architectures for JPEG2000 are listed in Table I. According to Table I, the proposed 2-D DWT architecture outperforms previous works in the aspects of internal memory size and critical path. This 2-D DWT architecture is frame-based that the reduction of the internal memory is significant. It adopts parallel and pipelining schemes to reduce the internal memory requirement and increase the operation speed. Shifters and adders are used to replace multipliers in the computation to reduce the hardware cost. A dual mode (5/3 and 9/7) 256×256 2-D DWT was designed and simulated with VerilogHDL and further synthesized by the Synopsys design compiler with TSMC 0.18μm 1P6M CMOS standard process technology to verify the performance of the proposed hardware architecture; the performance specifications are listed in Table II.

The proposed architecture is capable of processing 1080p with processing rate of 124M samples/sec, frame rate of 30 fps, and sampling rate of (4:2:2). Because the processing throughput of our proposed architecture is 2, the processing rate can be reduced to 62M samples/sec. The operation speed of this architecture is high enough to support the processing rate.

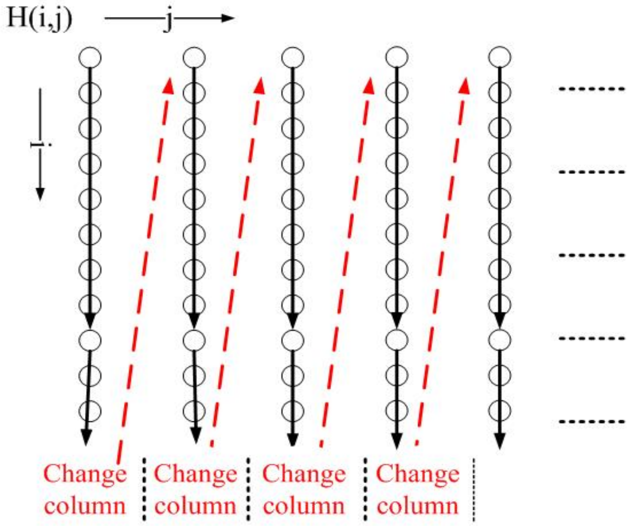

Figure 6. The input sequence of the column processor

TABLE I. HARDWARE COST AND PERFORMANCE COMPARISONS OF 2-D DWT ARCHITECTURE FOR 9/7 FILTER

| **Architecture** | **Multipliers** | **Adders** | **Temporal memory** | **Critical path** | **Throughput** |
|---|---|---|---|---|---|
| Huang *et al.*[4] | 10 | 16 | 5.5N | Tm + 5Ta | 2 input/output |
| Tseng *et al* [6] | 10 | 16 | 5.5N | 4Tm + 8Ta | 2 input/output |
| Xiong *et al* [7] | 10 | 16 | 5.5N | N/A | 2 input/output |
| Liao *et al* [8] | 12 | 16 | 4N | 4Tm+8Ta | 2 input/output |
| Proposed | 12 | 16 | 4N | 1Tm + 2Ta | 2 input/output |

TABLE II. DESIGN SPECIFICATION OF THE PROPOSED 2-D DWT

| | |
|---|---|
| **Chip specification** | *N* = 256, Tile size = 256×256 |
| **Gate count** | 29,196 gates |
| **Power supply** | 1.8V |
| **Technology** | TSMC 0.18mm 1P6M (CMOS) |
| **Internal memory size** | 2-D 5/3 DWT: 512 bytes<br>2-D 9/7 DWT: 1,024 bytes |
| **Latency** | (3/2)*N*+3 = 387 |
| **Computing time** | (3/4)*N*2+(3/2)*N*+7 = 49,543 |
| **Maximum clock rate** | 83 MHz |

## V. CONCLUSION

This paper proposes a memory-efficient VLSI architecture of lifting-based 2-D DWT for motion-JPEG2000. Because of the modified lifting-based algorithm, for an *N*×*N* image the internal memory sizes of the proposed 2-D DWT for 5/3 and 9/7 filters are 2*N* and 4*N*, respectively. Compared with other previous architectures, the internal memory size of the proposed architecture is very small. Based on the proposed architecture, a dual mode (5/3 and 9/7) 256×256 2-D DWT was designed and simulated with VerilogHDL and further synthesized by the Synopsys design compiler with TSMC 0.18μm 1P6M CMOS standard process technology. The prototyping chip takes 29,196 gate counts and can operate at 83 MHz operating frequency. Due to the characteristics of low memory size and high operation speed, it is suitable for VLSI implementation and can support various real-time image/video applications such as JPEG2000, motion-JPEG2000, MPEG-4 still texture object decoding, and wavelet-based scalable video coding.

## REFERENCES

[1] S. G. Mallat, “A theory for multi-resolution signal decomposition: The wavelet representation,” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 11, no. 7, pp. 674-693, July 1989.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] W. Sweldens, “The lifting scheme: A custom-design construction of biorthogonal wavelets,” Applied and Computation Harmonic Analysis, vol. 0015, no. 3, pp.186-200, March 1997.K. Elissa, “Title of paper if known,” unpublished.

[3] J.-R. Ohm, “Advances in scalable video coding,” Proc. of The IEEE, vol. 95, no.1, pp. 42-56, Jan. 2005.

[4] C. T. Huang, P. C. Tseng and L. G. Chen, “Flipping structure: an efficient VLSI architecture for lifting-based discrete wavelet transform,” IEEE Trans. Signal Processing, vol. 52, no. 4, pp. 1080-1089, Apr. 2004.

[5] JPEG2000 Part 1 Final Committee Draft Version 1.0, “ISO/IEC 15444-1 JTC1/SC29 WG1, Information Technology,” 2000.

[6] P. C. Tseng, C. T. Huang, and L. G. Chen, “Generic RAM-based architecture for two-dimensional discrete wavelet transform with line-based method,” in Proc. Asia-Pacific Conference on Circuits and Systems, 2002, pp. 363-366.

[7] C. Xiong, J. Tian, and J. Liu, “Efficient architectures for two-dimensional discrete wavelet transform using lifting scheme,” IEEE Trans. on Image processing, vol. 16, no. 3, Mar. 2007.

[8] H. Liao, M. Kr. Mandal, and B. F. Cockburn, “Efficient architectures for 1-D and 2-D lifting-based wavelet transforms,” IEEE Trans. Signal Processing, vol. 52, no. 5, pp. 1315-1326, May 2004.

[9] Motion JPEG2000, “ISO/IEC ISO/IEC 15444-3, Information Technology,” 2002.

[10] Coding of Moving Picture and Audio, “ISO/IEC JTC1/SC29 WG11, Information Technology,” 2001.

[11] M. Vishwanath, R. M. Owens, and M. J. Irwin, “VLSI architecture for the discrete wavelet transform,” IEEE Transactions on Circuits and Systems II, vol. 42, no. 5, pp. 305-316, May 1995.

[12] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, “Efficient VLSI architecture of lifting-based discrete wavelet transform by systematic design method,” IEEE International Symposium on Circuits and Systems, vol. 5, pp. 26-29, May 2002.