An Efficient VLSI Architecture for Lifting-Based Discrete Wavelet Transform

Wang Chao, Wu Zhilin, Cao Peng, Li Jie

Nation ASIC system Engineering Research Center, Southeast University, Nanjing, China

ABSTRACT

In this paper, we propose an efficient VLSI architecture which performs the two-dimensional (2-D) discrete wavelet transform (DWT) of 9/7 filter for JPEG2000. Based on the modified lifting-based DWT algorithm, an efficient VLSI architecture for one-dimensional (1-D) DWT is derived to reduce the hardware cost and shorten the critical path. The proposed 2-D DWT architecture is composed of two 1-D processors (row and column processors). Based on the line-based architecture, the column processor can start column-wise transform while only two rows have been processed. For an MxN image, only 5.5N internal memory is required for the 9/7 filter to perform the 2-D DWT with the critical path of one multiplier. Finally, Verilog simulation results are presented to show that the proposed architecture in comparison with other existing architectures is fast and efficient for the 2-D DWT computation.

1. INTRODUCTION

Discrete wavelet transform has been widely used in image compression. The well-know image coding standards, MPEG-4 still texture coding and JPEG2000 still image coding have adopted DWT as their transform coder.

Processing speed and internal memory requirement are the main issues for hardware implementation of DWT. For 1-D DWT, the architectures can be mainly categorized into the convolution-based and lifting-based [2] [3]. Although the lifting scheme needs less computation and lower memory, but the longer critical path limits the efficiency of hardware implementation. Adding more pipeline registers would shorten the critical path but would increase the internal memory size of 2-D DWT. Huang et al. proposed the flipping structure to shorten the critical path without hardware overhead [6]. The temporal buffer size of 2-D architecture for an MxN image is 11N with one multiplier delay. To shorten the critical path and lower the memory requirement of 2-D implementation, a modified lifting-based algorithm is proposed.

The rest of this paper is organized as follows. Section 2 presents the conventional and modified lifting-based DWT algorithm. Section 3 shows the proposed one-level 2-D DWT architecture based on the modified algorithm. Experimental results and performance comparison are described in Section 4. Finally, Section 5 gives a brief summary.

2. LIFTING-BASED DWT ALGORITHM

2.1. Conventional Lifting-Based DWT

The lifting scheme is a method for constructing wavelets by spatial approach [7]. According to [8], any DWT filterbank of perfect reconstruction can be decomposed into a finite sequence of lifting steps. This decomposition can factorize the ploy-phase matrix of the target wavelet filter into a sequence of alternating upper and lower triangular matrices and a constant diagonal matrix, which can be expressed as follows:

$$\begin{aligned} h(z) &= h_e(z^2) + z^{-1}h_o(z^2) \\ g(z) &= g_e(z^2) + z^{-1}g_o(z^2) \\ P(z) &= \begin{bmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{bmatrix} \end{aligned} \qquad (1)$$

$$= \prod_{i=1}^{m} \begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & \frac{1}{K} \end{bmatrix} \qquad (2)$$

where h(z) and g(z) are the low-pass and high-pass analysis filters, respectively, the (2) is the poly-phase decomposition, and P(z) is the poly-phase matrix.

For the 9/7 wavelet, four lifting steps and one scaling can be used. The detailed lifting steps are described from Eq. (3) - Eq. (10).

1) Splitting Step:

$$d_i^0 = x_{2i+1} \qquad (3)$$

$$s_i^0 = x_{2i} \qquad (4)$$

2) Lifting Step:

(First Lifting Step)

$$d_i^1 = d_i^0 + \alpha(s_i^0 + s_{i+1}^0) \qquad \text{(Predictor Step)} \quad (5)$$

$$s_i^1 = s_i^0 + \beta \times (d_{i-1}^1 + d_i^1) \qquad \text{(Updater Step)} \quad (6)$$

(Second Lifting Step)

$$d_i^2 = d_i^1 + \gamma(s_i^1 + s_{i+1}^1) \qquad \text{(Predictor Step)} \quad (7)$$

$$s_i^2 = s_i^1 + \delta \times (d_{i-1}^2 + d_i^2) \qquad \text{(Updater Step)} \quad (8)$$

1-4244-1017-7/07/$25.00 ©2007 IEEE

3) Scaling step:

$$d_i = \frac{1}{K} \times d_i^2 \qquad (9)$$

$$s_i = K \times s_i^2 \qquad (10)$$

The direct mapping hardware architecture for the 1-D lifting-based DWT uses four pipeline stages to shorten the critical path [5]. But the critical path is still restricted by the computation of predictor or update (i.e., two adders and one multiplier delay). Moreover, it needs 32 pipeline registers to minimize the critical path to one multiplier delay. This constraint becomes a bottleneck for increasing the processing speed of the conventional lifting structure.

2.2 Modified Lifting-Based DWT

According to the direct mapping architecture, the computation of predictor or updater mainly restricts the critical path. To shorten the critical path and reduce the number of arithmetic units and registers, the above equations are changed as follows.

Splitting Step:

$$d'^0_i = \frac{1}{\alpha} x_{2i+1} \qquad (11)$$

$$s'^0_i = x_{2i} \qquad (12)$$

Lifting Step:

(First Lifting Step)

$$d'^1_i = (d'^0_i + s'^0_i) + s'^0_{i+1} \qquad (13)$$

$$s'^1_i = (\frac{1}{\alpha\beta} s'^0_i + d'^1_{i-1}) + d'^1_i \qquad (14)$$

(Second Lifting Step)

$$d'^2_i = (\frac{1}{\beta\gamma} d'^1_i + s'^1_i) + s'^1_{i+1} \qquad (15)$$

$$s'^2_i = (\frac{1}{\gamma\delta} s'^1_i + d'^2_{i-1}) + d'^2_i \qquad (16)$$

Scaling step:

$$d'_i = \frac{\alpha\beta\gamma}{K} \times d'^2_i \qquad (17)$$

$$s'_i = \alpha\beta\gamma\delta K \times s'^2_i \qquad (18)$$

From the conventional lifting algorithm, it can be observed that the multiplication and addition operations in all lifting steps are completed in sequence, which increases the critical path latency of the VLSI architecture. Utilizing the similarity of the Eq.13 and Eq.14, Eq.15 and Eq.16, a new folded architecture is proposed which reuse one module to perform both the predictor and updater step in each lifting step. This leads to the reduction of the number of arithmetic units (multipliers and adders) and registers of the 1-D DWT, and the internal memory of the 2-D DWT. Furthermore, the critical path of the modified algorithm is decreased to one multiplier without adding pipeline registers. The 2-D architecture only requires the temporal memory with 4N to perform one level decomposition for 9/7 filter. In the following discussion, we will present the one-level 2-D DWT architecture based on the modified algorithm.

Fig.2 shows the proposed one-level RAM-based 2-D DWT architecture with line-based method [4]. It consists of three key modules — the row processor, the data buffer and the column processor [6]. The row processor performs 1-D row-wise DWT and the row-processed data is stored in the data buffer. While enough row-processed data are obtained, the column processor starts to perform the column-wise transform as soon as possible. The MN/4 external RAM is used to store the LL band output coefficients for the next decomposition. In the following, the details of these three key modules and the whole 2-D DWT architecture are discussed.

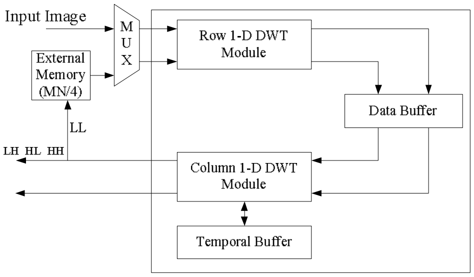


Fig.2 Proposed one-level 2-D DWT architecture

3. PROPOSED 2-D DWT ARCHITECTURE

3.1 Row Processor

The row processor is a 1-D DWT processor performing on the row-wise image data. Fig.4 depicts the detailed architecture for Eq. (13) and Eq. (14), which represents the calculation of the first lifting step in the modified algorithm. The processor utilizes the similarity of Eq. (13) and Eq. (14). It reads two input sample and writes two output samples in every two cycles. Fig.3 shows the data-flow graph (DFG) of the first lifting step for the 9/7 filter. The architecture of the second lifting step for 9/7 filter is the same as the first one, but it does not need the last two data registers because the data can be directly output to the data buffer. Every lifting step module is defined as a processing element (PE).

The 9/7 filter is composed of two lifting steps and one scaling step. Since both lifting steps have the same computation flow, the whole 1-D architecture for the 9/7 filter

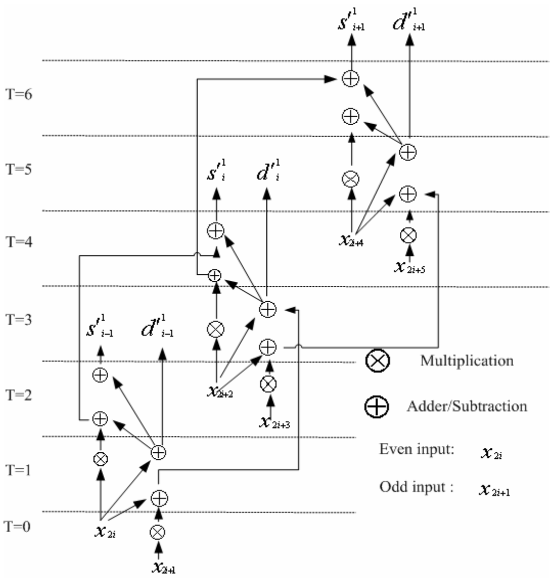


Fig.3 DFG of the first lifting step of the modified 9/7 filter can be realized by cascading two PEs and two scaling multipliers. In order to save the number of the multipliers, the scaling multipliers of the row processor are eliminated and combined with the scaling multiplier of the column processor.

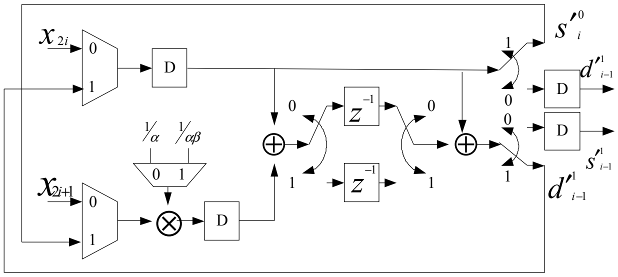
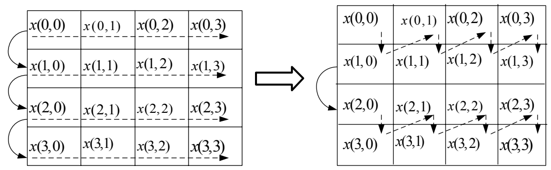


Fig.4 The architecture of first lifting step

3.2 Data Buffer

The proposed line-based 2-D DWT architecture needs 1.5N data buffer to store the row-processed data which are consumed by the column processor [4]. Since the proposed 1-D DWT module is designed as a means of two-input/two-output, the column processor module requires two lines of data simultaneously. But the row processor can only output the data in line-by-line way. The data buffer stores one complete even-row data firstly. Once the data of the odd row is inputted, the data buffer starts to output the row-processed data in the raster order of two rows [6]. We use a 4x4 sample to present the input and output orders of the data buffer as shown in Fig.5.

3.3 Column Processor



x(0,0)	x(0,1)	x(0,2)	x(0,3)
x(1,0)	x(1,1)	x(1,2)	x(1,3)
x(2,0)	x(2,1)	x(2,2)	x(2,3)
x(3,0)	x(3,1)	x(3,2)	x(3,3)

x(0,0)	x(0,1)	x(0,2)	x(0,3)
x(1,0)	x(1,1)	x(1,2)	x(1,3)
x(2,0)	x(2,1)	x(2,2)	x(2,3)
x(3,0)	x(3,1)	x(3,2)	x(3,3)

Fig.5 The input and output order of the data buffer

The column-wise transform can be considered as the transpose of the row process. The internal memory size of column processor highly depends on registers used in the 1-D architecture. Once the row-processed data are collected, the column-wise transform is then partially performed and the results are temporarily stored in the memory. Similar to the 1-D case, the column-wise transform for the 9/7 filter can be implemented by cascading two column processing element. Fig.6 shows the data-flow graph (DFG) of the column transform for the 9/7 filter. Fig.7 presents the architecture of the column processor. The temporal buffer (MEM) instead of the register of the row processor is used to store the temporal data. The input order is in a raster format of one pair row data. The two dual port ram read the previous data to execute the column transform and update the temporal results.

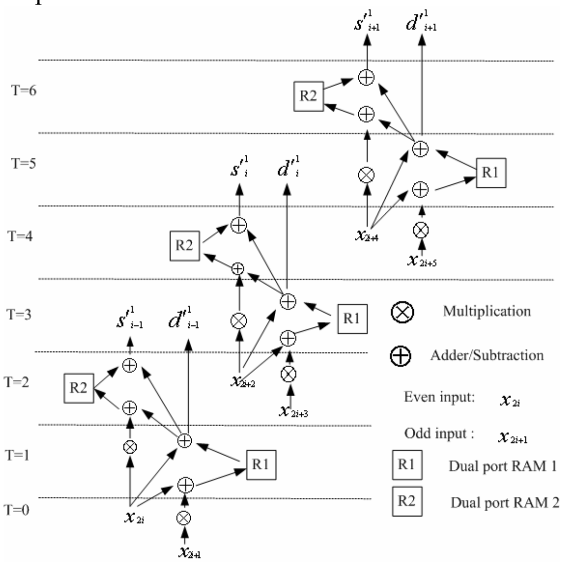


Fig.6 DFG of the column-wise transform

The 2-D DWT architecture can be realized by cascading the row processor, data buffer, column processor and scaling multipliers as shown in Fig. 8. First, the 1-D row processor processes the image data and the output data are then rearranged by the data buffer. After one row delay, the column processor starts to perform the column-wise transform. Finally, two multipliers are used for the scaling step to produce the coefficients of the LL, LH subband and the coefficients of the HL, HH subband. The LL subband data are stored in the external memory and can be used for the next level decomposition.

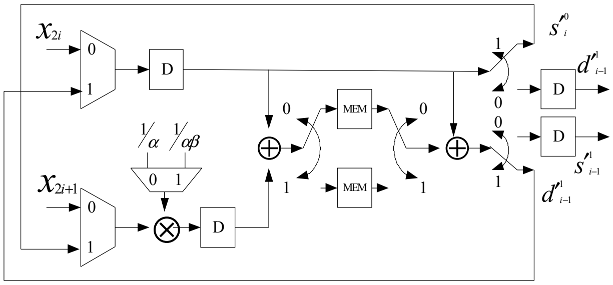


Fig.7 The architecture of column processing element

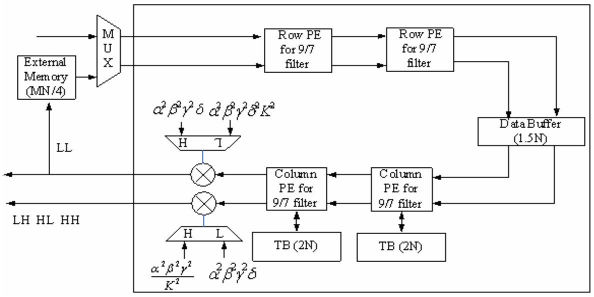


Fig. 8 The one-level 2-D DWT architecture for the 9/7 filter

4. EXPERIMENTAL RESULTS AND COMPARISONS

4.1 Experimental results

The hardware specification of the three key modules is presented in Table 1. For the proposed 2-D DWT architecture, the critical path is a single multiplier delay and the internal memory is 5.5N. The proposed architecture has been verified by use of Verilog and the experimental result shows that the architecture can be performed at 153.3MHz.

Table 1 Hardware specification of three key modules(Tm: The delay time of a multiplier)

Module	Multipliers	Adders	Registers	Critical Path	Internal Memory
Row Processor	2	4	10	Tm	-
Column Processor	2	4	10	Tm	4N
Data Buffer	-	-	-	-	1.5N

4.2 Comparisons

Table 2 compares several 1-D DWT architectures with the proposed architecture. Based on the modified algorithm, the proposed architecture achieves the one multiplier delay with 10 registers. Since the modified predictor and updater step are similar, the proposed folded architecture merges the predictor and the updater step into one module which only requires the half of arithmetic resources.

Table 2 Comparisons of various 1-D DWT architectures of 9/7 filters (Ta: the delay time of an adder, Tm: the delay time of a multiplier)

Architectures	Multipliers	Adders	Registers	Critical Path
Direct[8]	4	8	6	4Tm+8Ta
Direct+fully pipeline	4	8	32	Tm
Flipping[6]	4	8	4	Tm+5Ta
Flippling+5 pipeline	4	8	11	Tm
Proposed	2	4	10	Tm

Table 3 compares several one-level 2-D DWT architectures. The long critical path and temporal buffer size are two critical issues of 2-D DWT implementation. Compared with other listed architectures, the proposed 2-D DWT architecture only needs about half of the arithmetic resources. Only 4N temporal buffer memory is required for the column processor to perform one multiplier delay. The tradeoff between the temporal buffer size and critical path can be eased.

Table 3 Comparisons of various one-level lifting-based 2-D DWT architectures of 9/7 filter

Architectures	Multiplier	Adder	Data buffer	Temporal buffer	Critical Path
Generic RAM-based[4]	10	16	1.5N	4N	4Tm+8Ta
Flipping[6]	10	16	1.5N	4N	Tm+5Ta
Flipping + 5 stages pipeline	10	16	1.5N	11N	Tm
Proposed	6	8	1.5N	4N	Tm

5. CONCLUSION

This paper proposes a modified lifting-based DWT algorithm and also presents the 2D-DWT VLSI architecture for the 9/7 filter of JPEG2000. Based on the modified lifting-based DWT, the predictor and updater are merged into one single module to shorten the critical path to one multiplier delay and reduce the number of arithmetic units and registers. Due to the reduction of the registers in 1-D DWT, the temporal buffer size of column processor in the 2-D DWT is minimized to 4N. Thus, the critical issues for high processing speed and low memory size for the 2-D DWT can be eased.

6. REFERENCES

[1] Information Technology—JPEG 2000 Image Coding System, *ISO/IEC*. ISO/IEC 15 444-1, 2000.

[2] I. Daubechies and W. Sweldens, "Factoring wavelet transform into lifting steps," *J. Fourier Anal. Applicat*, vol. 4, pp. 247–269, 1998.

[3] W. Sweldens, "The new philosophy in biorthogonal wavelet constructions," *Proc. SPIE*, vol. 2569, pp. 68–79, 1995.

[4] 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 Conf. Circuits and Systems*, 2002, pp. 363–366.

[5] C. T. Huang, P. C. Tseng, and L. G. Chen, "Flipping structure: An efficient VLSI architecture for lifting-based discrete wavelet transform," in *Proc. IEEE ISCAS*, 2002, pp. 383–388.

[6] Bing-Fei Wu, Chung-Fu Lin, A high-performance and memory-efficient pipeline architecture for the 5/3 and 9/7 discrete wavelet transform of JPEG2000 codec. *IEEE Trans. Circuits Syst. Video Techn.* , vol. 15, issue 12, pp.1615-1628, Dec 2005.

[7] J. M. Jou, Y. H. Shiau, and C. C. Liu, "Efficient VLSI architectures for the biorthogonal wavelet transform by filter bank and lifting scheme," in *Proc. IEEE ISCAS*, vol. 2, 2001, pp. 529–529