GENERIC RAM-BASED ARCHITECTURE FOR TWO-DIMENSIONAL DISCRETE WAVELET TRANSFORM WITH LINE-BASED METHOD

Po-Chih Tseng, Chao-Tsung Huang, and Liang-Gee Chen

DSP/IC Design Lab, Department of Electrical Engineering, and
Graduate Institute of Electronics Engineering, National Taiwan University
Phone:+886-2-2363-5251, Fax:+886-2-2363-8247, Email:{pctseng, cthuang, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

In this paper, a generic RAM-based architecture by using line-based method is proposed to construct the corresponding two-dimensional architectures efficiently for any given hardware architecture of one-dimensional wavelet filters, including conventional convolution-based and advanced lifting-based architectures. The categories of line buffer and the strategy to optimize the size of internal memory are also described. For multi-level two-dimensional discrete wavelet transform, the recursive pyramid algorithm is adopted to turn our proposed architecture into another efficient architecture. According to the comparison results, the proposed architecture outperforms previous arts in the aspects of memory size, control complexity, and flexibility.

1. INTRODUCTION

Discrete Wavelet Transform (DWT) has been developed as an effective multiresolution analysis tool since it was presented by Mallat [1]. Due to its well time-frequency characteristics, DWT has been widely used for signal analysis and compression. For example, emerging image coding standards, such as MPEG-4 still texture coding and JPEG2000 still image coding, have adopted DWT as their transform coder. However, DWT requires heavy arithmetic computation because it is a 2-channel filter bank essentially. Thus, lifting scheme is proposed to reduce the arithmetic complexity and provide in-place implementation [2].

For 2-D DWT, there have been many VLSI architectures proposed, such as [3, 4, 5]. Nevertheless, only RAM-based architectures are likely to be implemented in real-life designs because of the highest regularity/density of storage, compared to systolic or semi-systolic routing [6]. Furthermore, it is the memory issue that dominates hardware cost and complexity of the architectures for 2-D DWT, but not multipliers. The simplest method to implement separable 2-D DWT is directly performing 1-D DWT in the row direction and then in the column direction. But this direct method needs a frame buffer that is usually in the external. According to the evaluation of [6], the external memory access consumes the most power in the 2-D DWT hardware implementation. Therefore, the line-based method [7] may be preferred if low power and easy integration are required because of the minimal external memory access. But the line-based method needs some internal line buffer that is proportional to the image width. As a result, how to minimize the line buffer becomes a critical problem.

This paper is organized as follows. Three kinds of RAM-based architectures for 2-D DWT are described and analyzed in section 2. Then we propose a generic line-based architecture for 1-level 2-D DWT, extend it to the multi-level case with recursive pyramid algorithm (RPA), and deduce a method to minimize the line buffer in section 3. In section 4, the comparison of our proposed architecture with others is shown to prove the efficiency and feasibility. Finally, conclusions about this paper are given in section 5.

2. 2-D DWT ARCHITECTURE

In this section, we will present three main architectures for separable 2-D DWT of the dyadic decomposition type, whose mathematical formulas are as follows:

$$\begin{aligned}
x^J_{LL}(n_1,n_2) &= \sum_{i_1,i_2} h(i_1)h(i_2)x^{J-1}_{LL}(2n_1-i_1,2n_2-i_2)\\
x^J_{LH}(n_1,n_2) &= \sum_{i_1,i_2} h(i_1)g(i_2)x^{J-1}_{LL}(2n_1-i_1,2n_2-i_2)\\
x^J_{HL}(n_1,n_2) &= \sum_{i_1,i_2} g(i_1)h(i_2)x^{J-1}_{LL}(2n_1-i_1,2n_2-i_2)\\
x^J_{HH}(n_1,n_2) &= \sum_{i_1,i_2} g(i_1)g(i_2)x^{J-1}_{LL}(2n_1-i_1,2n_2-i_2) \quad (1)
\end{aligned}$$

where J is the DWT decomposition level, x^0_{LL} is the input image, as well as $h(n)$ and $g(n)$ are the lowpass and highpass filters, respectively.

First, the most straightforward implementation of (1) is to perform 1-D DWT in one direction and save the intermediate coefficients in a frame buffer. And then perform 1-D DWT with these intermediate coefficients in the other direction to complete 1-level 2-D DWT. For the other decomposition levels, the LL subband of the current level can be treated as the input of the next level. Then perform 1-level 2-D DWT recursively. This direct architecture can be described as Fig. 1(a). It requires few hardware cost for its simplicity but spends much external memory access that can be expressed as:

$$4 \times (1+\frac{1}{4}+\frac{1}{16}+...+(\frac{1}{4})^{J-1})N^2(words)$$

0-7803-7690-0/02/$17.00 ©2002 IEEE

Architecture	External Memory Access	Line Buffer Size	Frame Buffer	System Integration
Direct	$5.33N^2$	no	N^2	Difficult
1-level	$2.67N^2$	KN	$N^2/4$	Medium
Multi-level	$2N^2$	2KN	no	Easy

Table 1: Summary of 2-D architectures ($J \rightarrow \infty$)

where N is the width and height of the image. The above expression includes both the external memory writes and reads.

Second, every level of the DWT decomposition can be performed at a time, such as Fig. 1(b). The 1-level 2-D architecture is based on the line-based method, which needs some internal buffer to perform 2-D DWT inside. This internal buffer is proportional to the image width and K, the number of line buffers used. Unlike that the direct architecture requires a frame buffer of size N^2, the 1-level 2-D architecture only needs the quarter of a frame [5]. In utilization of the internal buffer, the external memory access is merely the half of the case in the direct architecture.

Third, we can perform all of the decomposition levels simultaneously as shown as the multi-level 2-D architecture in Fig. 1(c). However, using J 1-level 2-D architectures to implement directly will result in very low hardware utilization. Generally, the tasks of all decomposition levels are allocated to a few modules for 1-D DWT and the size of required internal buffer is as follows:

$$(1+\frac{1}{2}+\frac{1}{4}+...+(\frac{1}{2})^{J-1})KN(words)$$

The multi-level 2-D architecture requires more internal buffer and suitable assignments for 1-D DWT modules so as to reduce the external memory access to the minimum, $2N^2$.

The above three architectures of 2-D DWT are summarized in Table 1. The issue of system integration is mainly influenced by the bandwidth requirement of external memory. It should be noted that the access of external memory consumes the most power in the hardware implementation of 2-D DWT. And, if the minimal external memory access is demanded, some internal buffer and an architecture of higher complexity than the direct method will be required.

3. PROPOSED LINE-BASED ARCHITECTURES

A generic line-based architecture of 2-D DWT will be proposed in this section. In our proposed architecture, throughput of the 1-D DWT architecture is considered as 2-input/2-output per cycle for feasibility and generality. Furthermore, the lifting-based 1-D DWT architectures can be easily integrated into our 2-D architecture to construct an optimal hardware implementation.

3.1. Proposed 1-level architecture

As shown in Fig. 2, the proposed architecture for 1-level 2-D DWT is basically composed of two 1-D DWT modules

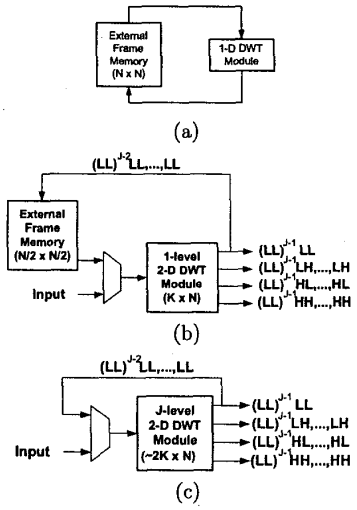

Figure 1: (a) direct 2-D architecture; (b) 1-level 2-D architecture; (c) multi-level 2-D architecture

and two kinds of line buffers. The data buffer is used to save the intermediate data after 1-D DWT in the row direction for providing the input signals of column 1-D DWT module, so it is independent of 1-D DWT architectures and only related to the throughput. On the other hand, the temporal buffer is highly dependent on the adopted 1-D DWT architecture. If the original circuits in the 1-D DWT architecture is as described as Fig. 3(a), the column 1-D DWT module and the temporal buffer need to be modified as Fig. 3(b).That is, using the temporal buffer to save the register data for the column 1-D DWT module. Because the memory access of temporal buffer is very regular and simple, we use K_0 two-port RAMs of size N to implement it, where K_0 is the number of registers in the column 1-D DWT module.

As for the data buffer, the lower bound of its size is 1N because the data flow always keep 2-input/2-output per cycle after the 1-D DWT coefficients of the first row are obtained. However, it is clear that extremely high complexity is followed with this minimal data buffer. In the following, we will describe a feasible data flow for data buffer of size 1.5N using two-port RAM , contrary to [8] where FIFO is used. As shown in Fig. 4, one RAM, called RAM_A, of size N is used for storing the coefficients of even rows, and the other one, RAM_B, of size 0.5N is for odd rows. After the first row coefficients fill out RAM_A as Fig. 4(a), the column 1-D DWT can start off. From (a) to (d), the second row coefficients get into RAM_B while the column 1-D DWT module consumes data of both RAM_A and RAM_B. Then the third row coefficients feed RAM_A from (d) to (e), and the data flow continues from (a) recursively. This data flow is without conflict because the speed of writing data is the double of reading data for each row. In fact, RAM_A and

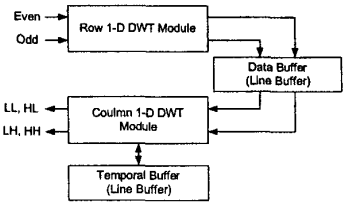

Figure 2: Proposed generic architecture for 1-level 2-D DWT

RAM_B are both composed of two two-port RAMs, one is for lowpass signals and the other one is for the highpass signals.

Here, we want to briefly mention how to be close to the lower bound. At first, $2^k + 1$ RAMs of size $N/2^k$ are required. After filling 2^k RAMs with the first row coefficients, we can treat the left one as RAM_B and follow the same steps from Fig. 4(a) to (d). The following is to perform the data flow in Fig. 5 recursively. In the beginning, take the two RAMs, in which the stored data is required by the column 1-D DWT module immediately, as RAM_C and RAM_D while the empty RAM is called RAM_E. Then we can transfer data from Fig. 5(a) to (d). Thus, an another empty RAM is obtained and the coming data is stored in RAM_C and RAM_E. Keeping this data transfer can complete the total data flow of 2-D DWT with only memory size of $(1 + (1/2)^k)N$.

Although the lower bound can be reached very closely, the additional control circuits and complex address generators may become impractical if k is too large. Besides the data flows mentioned above, no control circuits are demanded if a data buffer of size 3N is allowable. In [9], six two-port RAMs, three for lowpass signals and three for highpass signals, and a rotation-like address generator are adopted to perform the data transfer between the row and column 1-D DWT modules.

On the other hand, if lifting-based DWT is adopted as the 1-D DWT module, the number of multipliers can be further reduced by 2 with the observation of Fig. 6. Originally, two multipliers are required for both the row and column 1-D DWT module. But we can take the normalization multipliers out of the two 1-D DWT modules and use two multipliers, S^2 and $1/S^2$, to implement the normalization at the output of the column 1-D DWT module, where S is the normalization coefficient.

3.2. Proposed multi-level architecture

Our proposed 1-level architecture can be easily extended to multi-level architectures by scheduling the multi-level tasks for the two 1-D DWT modules with RPA [10] as described in Fig. 7. In Fig. 8, the RPA schedule for 3-level 2-D DWT is shown as an example. Because the data amount of the next level is the quarter of the current level,

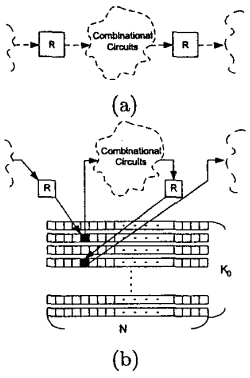

Figure 3: (a) original circuits; (b) modified line-based circuits (R: register)

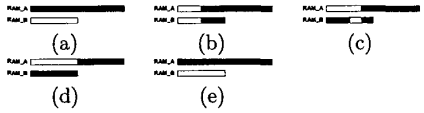

Figure 4: Data flow for data buffer of 1.5N (black: data stored; white: no data)

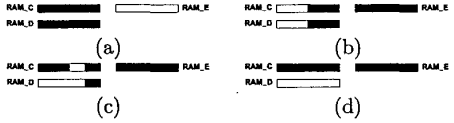

Figure 5: Data transfer for data buffer of $(1+(1/2)^k)N$

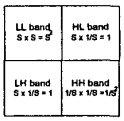

Figure 6: The normalization step of 2-D lifting-based DWT

the hardware utilization is:

$$\frac{1}{2} \times (1 + \frac{1}{4} + \frac{1}{16} + ... + (\frac{1}{4})^{J-1})$$

And the limit is 2/3 as J is indefinitely large. Moreover, the throughput is decreased to 1-input/1-output per cycle here because the hardware is shared for different decomposition levels.

The registers in the row 1-D DWT module needs to be increased to J times for each level of 2-D DWT decomposition while the total size of data buffer and temporal buffer becomes

$$(1 + \frac{1}{2} + \frac{1}{4} + ... + (\frac{1}{2})^{J-1})(K_0 + K_1)N(words)$$

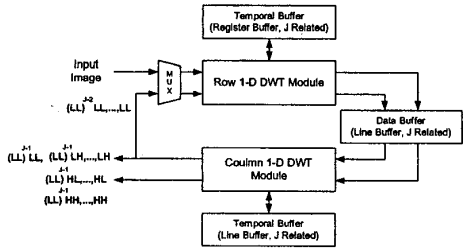

Figure 7: Proposed generic architecture for multi-level 2-D DWT

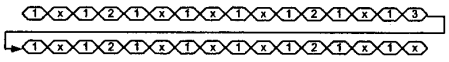

Figure 8: RPA schedule for 3-level 2-D DWT

where K_0 and K_1 are the numbers of line buffers in the temporal buffer and data buffer, respectively.

4. COMPARISON

In this section, we will show the efficiency of our architecture by the presentation of comparison results of different DWT architectures. The previous arts of 2-D DWT architectures are based on the convolution-based DWT architecture mostly. However, only the simplest lifting scheme, JPEG2000 defaulted lossless (5,3) filter, is considered in [8]. Our proposed architecture can adopt any kind of lifting-based DWT architecture to obtain the optimal hardware implementation. Thus, we use the popular JPEG2000 defaulted lossy (9,7) filter as the target 1-D DWT module for comparison. Here, we set K_1 of our proposed architecture as 1.5 for the reasonable balance of memory size and control complexity. The comparison between the proposed architecture, systolic-parallel architecture [4], and parallel-parallel architecture [3] for multi-level 2-D DWT is given in Table 2, where the lifting-based DWT module is from [11]. Moreover, throughputs of the architectures in Table 2 are all the same.

5. CONCLUSION

We have proposed a generic RAM-based architecture of high efficiency and feasibility with line-based method for both 1-level and multi-level 2-D DWT. Both convolution-based and lifting-based DWT architectures can be integrated into the proposed architecture flexibly while most previous arts can only support the former. According to the comparison results, the outstanding performance of our proposed architecture is proven.

Architecture (2-D)	Multiplier	Adder	Line Buffer	DWT module
Systolic-Parallel	36	36	22N	Convolution
Parallel-Parallel	36	36	19N	Convolution
Proposed	18	28	17N	Convolution
Proposed	10	16	11N	Lifting Scheme

Table 2: Comparison of 2-D architectures for (9,7) filter ($J \rightarrow \infty$)

REFERENCES

[1] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, July 1989.

[2] W. Sweldens, "The lifting scheme: a custom-design construction of biorthogonal wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, no. 15, pp. 186–200, 1996.

[3] C. Chakrabarti, M. Vishwanath, and R. M. Owens, "Architectures for wavelet transforms: A survey," *Journal of VLSI Signal Processing*, vol. 14, pp. 171–192, 1996.

[4] M. Vishwanath, R. M. Owens, and M. J. Irwin, "VLSI architectures for the discrete wavelet transform," *IEEE Transactions on Circuis and Systems-II: Analog and digital signal processing*, vol. 42, no. 5, pp. 305–316, May 1995.

[5] P.-C. Wu and L.-G. Chen, "An efficient architecture for two-dimensional discrete wavelet transform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 4, pp. 536–545, Apr. 2001.

[6] N. D. Zervas and et al., "Evaluation of design alternatives for the 2-D-discrete wavelet transform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 12, pp. 1246–1262, Dec. 2001.

[7] C. Chrysafis and A. Ortega, "Line-based, reduced memory, wavelet image compression," *IEEE Transactions on Image processing*, vol. 9, no. 3, pp. 378–389, Mar. 2000.

[8] C. Diou, L. Torres, and M. Robert, "A wavelet core for video processing," in *Proc. of International Conference on Image Processing*, 2000, pp. 395–398.

[9] P.-C. Tseng, C.-T. Huang, and L.-G. Chen, "VLSI implementation of shape-adaptive discrete wavelet transform," in *Proc. of SPIE International Conference on Visual Communications and Image Processing*, 2002, pp. 655–666.

[10] M. Vishwanath, "The recursive pyramid algorithm for the discrete wavelet transform," *IEEE Transactions on Signal Processing*, vol. 42, no. 3, pp. 673–677, Mar. 1994.

[11] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "Efficient VLSI architectures of lifting-based discrete wavelet transform by systematic design method," in *IEEE International Symposium on Circuits and Systems*, 2002, vol. 5, pp. 565–568.