MEMORY ANALYSIS AND ARCHITECTURE FOR TWO-DIMENSIONAL DISCRETE WAVELET TRANSFORM

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

DSP/IC Design Lab, Graduate Institute of Electronics Engineering and
Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan
Email:{cthuang, pctseng, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

The large amount of the frame memory access and the die area occupied by the embedded internal buffer are the most critical issues for the implementation of two-dimensional discrete wavelet transform (2-D DWT). The former would consume the most power and waste the system memory bandwidth. The latter would enlarge the chip size and also consume much power. In this paper, we categorize and analyze the 2-D DWT architectures by different external memory scan methods. Then the overlapped stripe-based scan method is proposed to provide an efficient and flexible implementation for 2-D DWT. The implementation issues of the internal buffer are also discussed, including the lifting-based and convolution-based. Some real-life experiments are given to show that the performance of area and power for the internal buffer is highly related to memory technology and working frequency, instead of the required memory bits only.

1. INTRODUCTION

DWT has been developed as an efficient DSP tool for signal analysis, image compression, and even video compression. There are many architectures proposed for the implementation of DWT. For the 1-D DWT, the architectures can be categorized into the convolution-based [1], lifting-based [2, 3], and B-spline-based [4]. When extending the 1-D DWT module to the 2-D DWT architecture, the memory issue is the most important design consideration [5] because the 2-D DWT requires a large amount data access and storage. The design trade-off mainly comes from the frame memory access bandwidth and the internal buffer size. In [5], the design alternatives are evaluated in the aspects of power and memory requirements. However, the evaluation only covers three specified 2-D architectures. The frame memory is usually off-chip so that the external frame memory access would consume the most power and waste much system memory bandwidth. As the cache is used to reduce the main memory access in the general processor architectures, so the internal buffer is used to reduce the frame memory access for 2-D DWT. However, the internal buffer would occupy much die area. Many 2-D DWT architectures using different memory structures have been proposed [6, 7, 8, 9, 10].
In this paper, we discuss two independent issues of the 2-D DWT architectures. The first one is the trade-off between the external frame memory access and the internal buffer size. And one efficient and flexible frame memory scan method is proposed. The second one is the real-life implementation methods for the internal buffer. The organization of this paper is as follows. Prerequisites for DWT architectures are given in section 2. Then we categorize previous 2-D DWT architectures in section 3 and propose an efficient scan method in section 4. In section 5, the implementation methods of the internal buffer are discussed and some real-life experiment results are also presented. A summary is given to conclude this paper in section 6.

This work was supported in part by MOE Program for Promoting Academic Excellence of Universities under the grant number 89E-FA06-2-4-8, in part by National Science Council, Republic of China, under the grant number 91-2215-E-002-035, and in part by the MediaTek Fellowship.

2. PREREQUISITE

The 1-D DWT architectures can be categorized into convolution-based, lifting-based, and B-spline-based [4]. When extending the 1-D DWT modules to the 2-D line-based architectures, the registers will become the internal line-buffer that is called temporal buffer in [7]. Besides, if the image pixels are input in raster scan, additional internal buffer, called data buffer, will be required to store the intermediate DWT coefficients because the 1-D modules are usually two-input-two-output per cycle for 100% hardware utilization.
The implementation of the temporal buffer is dependent on the adopted 1-D modules, which will be discussed in section 5. However, the data buffer is only corresponding to the raster scan and can be minimized to one line [7]. If the image pixels are input in Z-scan fashion [10], the data buffer can be eliminated. Thus, the data buffer will be excluded in the following discussion.
In [10], an optimal Z-scan is proposed for the JPEG 2000 system to minimize the total size of the internal buffer, including the temporal buffer of the DWT and the EBCOT word-to-bitplane buffer. However, if EBCOT is performed in the bitplane parallel mode, the word-to-bitplane buffer can be discarded at all [11]. As a result, the temporal buffer of DWT is a very important factor for a JPEG 2000 system.

3. PREVIOUS FRAME MEMORY SCAN METHODS

In this section, we focus on the 1-level 2-D architectures that perform 1-level 2-D DWT only [7]. There are two ways to extend them to multi-level decompositions. The first one is performing 1-level DWT recursively, which increases the frame memory access by the factor $1 + \frac{1}{4} + ... + (\frac{1}{4})^J = \frac{4}{3}(1 - (\frac{1}{4})^J)$ if J-level decompositions are required while the internal buffer size is the same. The other one is to extend the 1-level architecture to the multi-level one that performs all levels of DWT decomposition at

a time, which will be mentioned for each scan method except the direct one.

3.1. Direct scan

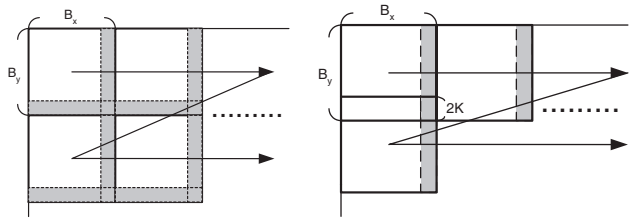
The direct scan is the straightforward implementation of 2-D DWT and uses the frame memory to store the intermediate DWT coefficients. The row-wise DWT is performed first and then the column-wise DWT. Thus, the frame memory reads and writes are both $2N^2$ words for 1-level DWT, where N is the image width and height.

3.2. Line-based scan

The line-based scan method uses some internal line buffer to store the intermediate DWT coefficients [6, 7]. The scan order is the raster scan. The size of the temporal line buffer is LN, where L is the number of registers in the adopted 1-D DWT module. For example, L is 4 and 7 for the lifting-based and convolution-based (9,7) filters without pipelining, respectively [3]. The frame memory reads and writes are both N^2 words for 1-level DWT. For the multi-level architecture, the size of line buffer is increased by the factor $\alpha = 1 + \frac{1}{2} + ... + (\frac{1}{2})^J = 2(1 - (\frac{1}{2})^J)$.

3.3. Non-overlapped and overlapped block-based scan

Block-based methods scan the frame memory block-by-block, and the DWT coefficients are also computed block-by-block. If the blocks are not overlapped with each other, the frame memory reads and writes are both N^2 words for 1-level DWT [8]. For each block, some intermediate data need to be stored for two neighboring blocks as shown in Fig. 1(a), where the grey area represents the intermediate data. We assume the blocks are scanned in the row direction first. Then the size of the internal buffer is $LN + LB_y$. For the multi-level architecture, the size of line buffer is also increased by the factor α.



(a) Non-overlapped (b) Overlapped

Fig. 1. Block-based scan methods

On the other hand, the buffer LN can be eliminated if the column-wise intermediate data are not stored. Instead, we can retransmit the required data at the block boundary from the frame memory. The block-based scheme in [9] is generalized to the overlapped block-based scan method as shown in Fig. 1(b), where $K = \lceil \frac{F-1}{2} \rceil$ and F is the DWT filter tap. That is, the blocks are overlapped $2K$ pixels in the column direction. The overlapped area is described in Fig. 2 in more detail. The DWT coefficients of the first K and last K columns are not valid. Thus, the overlapped pixels are $2K$ for deriving all DWT coefficients. The retransmission scheme increases the frame memory reads to $N^2 \frac{B_y}{B_y - 2K}$ words while the frame memory writes is still N^2 words. The internal buffer size LB_y can be reduced by shrinking the block size, but it would also increase the frame memory read bandwidth. As for the multi-level architecture, the overlapped area will become $2^J K$ that increases exponentially as J, and the frame memory reads become $N^2 \frac{B_y}{B_y - 2^J K}$ words. Thus, the overlapped block-based scan is not feasible for multi-level architectures.

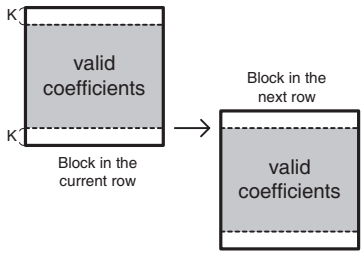


Fig. 2. Details of the overlapped blocks

3.4. Non-overlapped stripe-based scan

The optimal Z-scan method is proposed in [10], which is equivalent to perform the line-based scan in the wide block ($B_x = N$). In concept, the wide blocks can be viewed as stripes. So, this kind of method is categorized as the non-overlapped stripe-based scan as shown Fig. 3. The internal buffer size is $LN + LS$, where S is the width of the stripe. The first term is for the intermediate buffer between stripes, and the second term is for the line buffer inside stripes. For the multi-level architecture, the size of line buffer is also increased by the factor α.

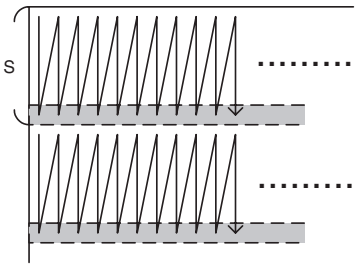


Fig. 3. Non-overlapped stripe-based scan method

4. PROPOSED OVERLAPPED STRIPE-BASED SCAN METHOD

We proposed the overlapped stripe-based scan method as shown in Fig. 4. All parameters are the same as that of the overlapped block-based scan method, except the stripe width S is used instead of B_y. This scan method can avoid the complex control circuits for block-based DWT architectures.

4.1. Comparison

The comparisons of the aforementioned scan methods are listed in Table 1. The trade-off between external memory access and internal buffer size is presented. The direct scan does not require any

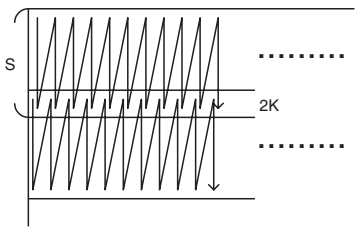


Fig. 4. Overlapped stripe-based scan method

internal buffer but suffers nearly double external memory access than other scan methods. The line-based scan method minimizes the external memory access but requires larger internal buffer size. According this table, the two non-overlapped scan methods are worse than the line-based scan method due to the larger internal buffer and higher control complexity. Between the two overlapped scan methods, the stripe-based one may be preferred for its simplicity. In fact, the overlapped stripe-based scan method can be implemented by use of a line-based architecture with the width S and an external memory address generator. Especially, the proposed scan method is degenerated to the line-based scan when $S = N$. As a result, the proposed overlapped stripe-based scan method can provide the best trade-off among external memory access, internal buffer size, and the control complexity.

5. INTERNAL BUFFER IMPLEMENTATION METHODS

In this section, we discuss the implementation methods for the internal buffer. The resulting performance is related to not only the required memory bits but also the memory structures, such as two-port or single-port memory. However, only the required memory size is discussed in literature.

5.1. Lifting-based DWT module

The registers in the lifting-based 1-D DWT module can be constructed to the internal buffer as Fig. 5. One method is to use a two-port memory to represent the registers because one-read-one-write per cycle is required. The other method is to use two single-port memories and exchange the roles of reads and writes for each different line in a ping-pong fashion. The required memory bits for the second method are the double of that for the first one. However, two-port memories are always larger and more power-consuming than single-port memories due to the memory cell design technologies.

Besides the above methods, we can slow down the DWT module by 2 so as to change the memory access to only one-read or one-write in one cycle. Thus, one single-port memory is sufficient, but the throughput of this method is halved. All registers can be merged into one corresponding address for higher density. Thus, only one two-port memory, two single-port memories, and one single-port memory are required for the first, second, and folded methods, respectively.

5.2. Convolution-based DWT module

The registers in the convolution-based DWT module can be constructed as Fig. 5 as well. The number of registers is usually larger than the lifting-based architectures, so the required memory size is also larger. However, the FIFO (first-in-first-out) register chain of

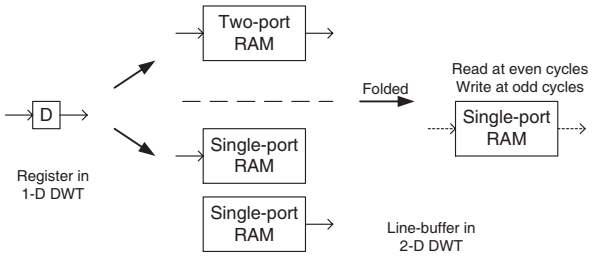


Fig. 5. 1-D Registers to 2-D line-buffer

the parallel filters can be implemented in a more efficient way. The data in the memories are not necessarily changed in every cycle. Instead, only two data need to be updated in every cycle. Thus, we can use F separate single-port memories for this rotation-like read-and-write method. In every cycle, two of them are written and others are read.

As for the folded architecture, it can be implemented as the lifting-based module to a unified single-port memory. It can also be constructed by (F-2) separate single-port memories. Two of them are written for every other cycle, and all of them are read for every other cycle.

5.3. Experiments

For examining the real-life implementation, we use the Artisan TSMC 0.25mm Process High-Density Single-Port SRAM Generator and the High-Speed Two-Port Register File Generator to generate the required single-port and two-port memories. In these experiments, we consider the (9,7) DWT filter and the image width is set as 64 or 128. The internal wordlength is 16-bit for all registers. We use the flipping structure to implement lifting-based architectures for saving the internal buffer [3]. The flipping structures can be synthesized by the Synopsys Design Compiler to about 70MHz and 130MHz by using only 4 and 7 registers with the Artisan 0.25-μm cell library while the conventional lifting-based architectures need 4 and 10 registers for 30MHz and 100MHz, respectively. The convolution-based architectures are implemented by use of parallel filters, and 93MHz can be achieved with 7 registers.

The results of all implementation methods are listed in Table 2 and 3 for working frequencies 50MHz and 100MHz, respectively. It can be found that the best performance in terms of area and power is dependent on the image width and working frequency. This illustrates the performance is highly related to the memory design technology. The smallest number of the required memory bits can not guarantee the best performance.

The throughput of folded methods at 100MHz is equivalent to that of non-folded ones at 50MHz. The results of folded methods are shown in Table 4. Compared with Table 2, the folded method in section 5.1 can provide the smallest area.

6. SUMMARY

The main contribution of this paper is to provide a detailed and feasible memory analysis for 2-D DWT architectures. We discuss two important memory issues for 2-D DWT implementation. The first one is the trade-off between external frame memory access and internal buffer size. Different frame memory scan methods are analyzed, and one efficient overlapped stripe-based scan method is also proposed. The second issue is the implementation method for

Table 1. Comparisons of scan methods for 1-level 2-D DWT

	Direct	Line-based	Non-overlapped Block-based	Overlapped Block-based	Non-overlapped Stripe-based	Proposed Overlapped Stripe-based
Frame Memory Read (words)	$2N^2$	N^2	N^2	$N^2B_y/(B_y\text{-}2K)$	N^2	$N^2S/(S\text{-}2K)$
Frame Memory Write (words)	$2N^2$	N^2	N^2	N^2	N^2	N^2
Internal Buffer Size (words)	0	LN	$L(N+B_y)$	LB_y	L(N+S)	LS
Control Complexity	Low	Medium	High	High	Medium	Medium

Table 2. Comparisons of internal line buffer at 50MHz

	N=64			N=128		
DWT module	**Flipping**	**Flipping**	**Conv.**	**Flipping**	**Flipping**	**Conv.**
Memory Type	Two-port	Single-port	Single-port	Two-port	Single-port	Single-port
Memory Configuration (bit)	64x64	2 64x64	9 64x16	128x64	2 128x64	9 128x16
Total Area (mm^2)	**0.2500**	0.2515	0.3457	0.3988	**0.3535**	0.495
Power (mW)	**39.71**	45.02	65.25	52.89	**45.89**	67.23

Table 3. Comparisons of internal line buffer at 100MHz

	N=64			N=128		
DWT module	**Flipping**	**Flipping**	**Conv.**	**Flipping**	**Flipping**	**Conv.**
Memory Type	Two-port	Single-port	Single-port	Two-port	Single-port	Single-port
Memory Configuration (bit)	64x112	2 64x112	9 64x16	128x112	2 128x112	9 128x16
Total Area (mm^2)	0.3786	0.4261	**0.3457**	0.6038	0.5969	**0.495**
Power (mW)	**62.73**	162.0	130.5	**82.90**	165.18	134.46

the internal buffer. We introduce some methods for lifting-based and convolution-based DWT modules. According to the experiments, the fewest required memory bits can not guarantee the best performance of area and power. Instead, the memory design technology may dominate the results.

7. REFERENCES

[1] 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.

Table 4. Folded methods at 100MHz for equivalent 50MHz throughput

	N=64		N=128	
DWT module	**Flipping/Conv.**	**Conv.**	**Flipping/Conv.**	**Conv.**
Memory Type	Single-port	Single-port	Single-port	Single-port
Memory Configuration (bit)	64x112	7 64x16	128x112	7 128x16
Total Area (mm^2)	**0.2130**	0.2689	**0.2985**	0.3850
Power (mW)	81.16	65.25	82.59	67.19

[2] K. Andra, C. Chakrabarti, and T. Acharya, "A VLSI architecture for lifting-based forward and inverse wavelet transform," *IEEE Transactions on Signal Processing*, vol. 50, no. 4, pp. 966–977, Apr. 2002.

[3] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "Flipping structure: An efficient VLSI architecture for lifting-based discrete wavelet transform," in *Asia-Pacific Conference on Circuits and Systems*, 2002, pp. 383–388.

[4] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "VLSI architecture for discrete wavelet transform based on B-spline factorization," in *IEEE Workshop on Signal Processing Systems*, 2003, pp. 346–350.

[5] N. D. Zervas, G. P. Anagnostopoulos, V. Spiliotopoulos, Y. Andreopoulos, and C. E. Goutis, "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.

[6] 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.

[7] 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 *Asia-Pacific Conference on Circuits and Systems*, 2002, pp. 363–366.

[8] W. Jiang and A. Ortega, "Lifting factorization-based discrete wavelet transform architecture design," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 5, pp. 651–657, May 2001.

[9] H. Yamauchi et al., "Image processor capable of block-noise-free JPEG2000 compression with 30frames/s for digital camera applications," in *IEEE International Solid-State Circuits Conference*, 2003, pp. 46–47.

[10] M.-Y. Chiu, K.-B. Lee, and C.-W. Jen, "Optimal data transfer and buffering schemes for JPEG 2000 encoder," in *IEEE Workshop on Signal Processing Systems*, 2003, pp. 177–182.

[11] H.-C. Fang, C.-T. Huang, Y.-W Chang, T.-C. Wang, P.-C. Tseng, C.-J. Lian, and L.-G. Chen, "81 M samples/s JPEG 2000 single-chip encoder with rate-distortion optimization," in *IEEE International Solid-State Circuits Conference 2004, accepted*.