Hardware Implementation of Dual-Tree Wavelet Transform Based Image Reconstruction

Hitesh Sudhakar, Lamia M Kalam, Sripathi Muralitharan, Deepu S. P., Sumam David S.
Department of Electronics and Communication Engineering
National Institute of Technology Karnataka, Surathkal, Mangalore, India
Email: 1998hitesh0902@gmail.com, lamiakalam97@gmail.com, sripathimuralitharan@gmail.com

***Abstract*—Real-time implementations of image processing algorithms on embedded platforms are gaining importance. In this paper, we propose an Application Specific Integrated Circuit (ASIC) architecture for the perfect reconstruction of images using wavelets with a view to extending this to denoising and feature extraction of images. An architecture that implements the Dual-Tree Wavelet Transform is presented. The architecture features a 128x128 single-port block memory and its addressing schemes, a simple upsampling/downsampling method and a folding and adding mechanism. It is implemented using 180nm technology. The results show perfect reconstruction of 128x128 grayscale images with up to 1-bit error in pixel values when compared to the corresponding input images.**

I. Introduction

As computer vision, robotic vision and other real-time embedded applications are gaining ground, it is essential that implementations of various image processing algorithms are optimized without compromising quality. We propose to implement the architecture for Perfect Reconstruction (PR) of images using the Dual-Tree Wavelet Transform in hardware which meets the real-time video processing requirements. This forms the base to implement algorithms for applications such as image denoising and feature extraction.

The Dual-Tree Complex Wavelet Transform (DT-CWT) proposed by Selesnick et al. [1], is a computationally efficient and low redundancy transform compared to other algorithms. DT-CWT is shift-invariant and directionally selective. Previous work in this field targeting various applications based on the Dual-Tree framework discuss various implementations and optimization. This includes an FPGA implementation of biomedical signal enhancement in [2] and [3], an algorithm for plant phenotyping in [4] and a compression algorithm in [5]. However, they do not adequately furnish details about the architecture nor do they concretely state the performance metrics. Real-time implementation and accelerators are the need of the hour and this is not adequately addressed in the available literature. In this paper, we implement the DT-CWT, a widely-used algorithm for image processing, keeping in mind the real-time requirement of the hardware implementation.

The paper is organized as follows: Section 2 gives an outline of the algorithm. Section 3 deals with the hardware architecture and Section 4 presents the experimental results.

Hitesh Sudhakar, Lamia M Kalam and Sripathi Muralitharan contributed equally to this work

II. Algorithm

The algorithm to perform the dual-tree wavelet transform for images is illustrated in Fig. 1.

A. Dual-Tree Wavelet Transform

The Discrete Wavelet Transform (DWT) has two main disadvantages, namely, it is not shift invariant and lacks directionality. The shift-variant nature of the DWT implies that the coefficients do not effectively capture the features of the input signal. The lack of directionality of the DWT prevents the edge information in an image to be captured effectively. However, wavelets with their sparsity and multi-resolution nature are still an attractive option. Complex wavelet transforms can address both these issues [1]. The dual-tree wavelet transform can implement the complex wavelet transform with two real DWTs [6], [7].

In this paper, we are implementing the real-oriented dual-tree wavelet transform. The wavelet transform is implemented over three stages of analysis and reconstructed using three stages of synthesis and there are two parallel trees as shown in Fig. 1. An image is a 2-D separable signal, therefore we separate the row and column filtering and filter the rows first and columns next. The filter coefficients were obtained using MATLAB® wavelet toolbox for the real-oriented dual-tree wavelet transform. These filters are designed using the QShift method. For level 1, 10-tap Farras filters [7] are used, which are orthogonal and nearly symmetric and for subsequent stages 10 tap Kingsbury QShift filters are used [8]. The QShift filters provide a group delay of 1/4 sample and are even length, orthonormal filters beyond level 1. They achieve the 1/2 sample delay requirement by choosing filters for tree 2 as the time-reversed versions of the corresponding filters in tree 1. The filter design methodology using the QShift method [6] is summarized below -

To design an even length low pass filter $H_L(z)$ with a group delay close to 1/4 sample, we use a linear phase low-pass FIR filter, $H_{L2}(z)$ of twice the length, twice the desired delay and half the bandwidth defined as

$$H_{L2}(z) = H_L(z^2) + z^{-1}H_L(z^{-2}) \tag{1}$$

As long as $H_{L2}(z)$ has very little gain above a quarter of its sampling frequency, there will be negligible aliasing caused by the sub-sampling and $H_L(z)$ will have a delay closely approximating 1/4 sample. $H_L(z)$ is adjusted such that the

978-1-7281-3320-1/20/$31.00 ©2020 IEEE

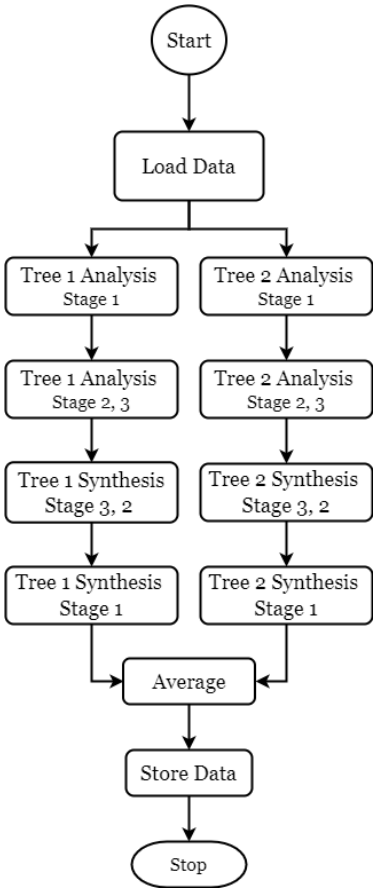


Fig. 1: Flowchart of the Dual-Tree Wavelet Transform Algorithm

squared gain of $H_{L2}(z)$ in a suitably chosen stop band is minimized subject to a perfect reconstruction of $H_L(z)$. This can be achieved by using a polyphase matrix that is factorized into a series of rotations and delays. This is illustrated below -

$$\begin{pmatrix} H_L(z) \\ z^{-1}H_L(z^{-1}) \end{pmatrix} = \mathbf{R}(\theta_n)\mathbf{Z}\mathbf{R}(\theta_{n-1})\mathbf{Z}...\mathbf{R}(\theta_1)\begin{pmatrix} 1 \\ z^{-1} \end{pmatrix} \quad (2)$$

where,

$$\mathbf{Z} = \begin{pmatrix} z & 0 \\ 0 & z^{-1} \end{pmatrix}, \mathbf{R}(\theta_i) = \begin{pmatrix} cos(\theta_i) & sin(\theta_i) \\ -sin(\theta_i) & cos(\theta_i) \end{pmatrix} \quad (3)$$

Here, $i = 1, 2, 3, ..., n$ and $2n$ is the length of the filter $H_L(z)$. If $H_L(z)$ is to have at least one zero at z = -1, then the n rotations of θ_i must sum to $\pi/4$. This leaves only $n-1$ angles to optimize, instead of $2n$ coefficients of H_L, and it automatically satisfies the PR condition as shown in [8].

Equations 1, 2 and 3 are used to design optimum filter banks with shift-invariant properties. The inverse Z-transform of $H_L(z)$ gives the filter coefficients for tree 1. The tree 2 coefficients are the time reversed versions of the coefficients for tree 1. The parallel dual-tree architecture serves to provide the effect of complex wavelets. The problem of lack of directionality is solved by the use of 2 trees which can now capture six orientations. Since we are looking at a wavelet transform that is essentially performed over different translations and scales, we downsample the outputs of filters by a factor of 2, which simultaneously doubles the frequency spectrum of the wavelet and translates it by a factor of 2 [9]. The complementary operation of upsampling is performed during reconstruction. Larger the depth of the filter bank, finer the resolution of the transform. In this paper, we implement a 3-stage wavelet transform. The operations of the transform are filtering followed by downsampling in the forward transform and upsampling followed by filtering in the backward transform which, is captured by the architecture of the filter bank used.

III. Architecture

The hardware architecture is developed to work with 128x128 images but can be extended to any desired size, NxN. The datapath as shown in Fig. 2 has been area-optimized to use a single convolution unit with changing filters based on the stage and direction of the transform. It mainly consists of two memory blocks to store the coefficients and intermediate data and the filtering units to perform the filtering. The *UPSAMPLER* and *DOWNSAMPLER* blocks are used for upsampling and downsampling the data during synthesis and analysis respectively, and are controlled using the *SELECT MEMORY* signal. It also consists of the folding and adding module (not shown in Fig. 2), which helps to maintain the signal lengths. Multiplexers, Demultiplexers, counters and the control block manage the transfer of data and use a circular addressing scheme. The following sub-sections highlight each of these aspects in detail.

A. Normalization and Scaling

Eight bit data received from the camera, ranging from 0 to 255 is normalized to a number between 0 and 1 which is represented in a fixed-point format with an 8-bit signed integer part and 16-bit fractional part. This normalization is performed to have a reasonable data representation for processing. The normalization is done by assuming the input pixels and the normalization factor, i.e., $1/255$ are in a fixed-point format with 8-bit integer part and 8-bit fractional part. The formula that is used for normalization by 255 is -

$$x\_norm = (x << 8) * (\frac{1}{255} + 1) \quad (4)$$

B. Memory

It consists of two single-port Block Memories of size NxN words with each word being 24 bits wide. One of the two memories acts as the primary memory, which stores the wavelet coefficients after every stage. The second memory is used as a secondary memory to store the wavelet coefficients after row/column processing. The memories are modeled using module instantiations of register arrays.

Counters and multiplexers are used to address the memory to read/write data. At the end of the row processing, the first half of the memory stores the results of the low pass filtering

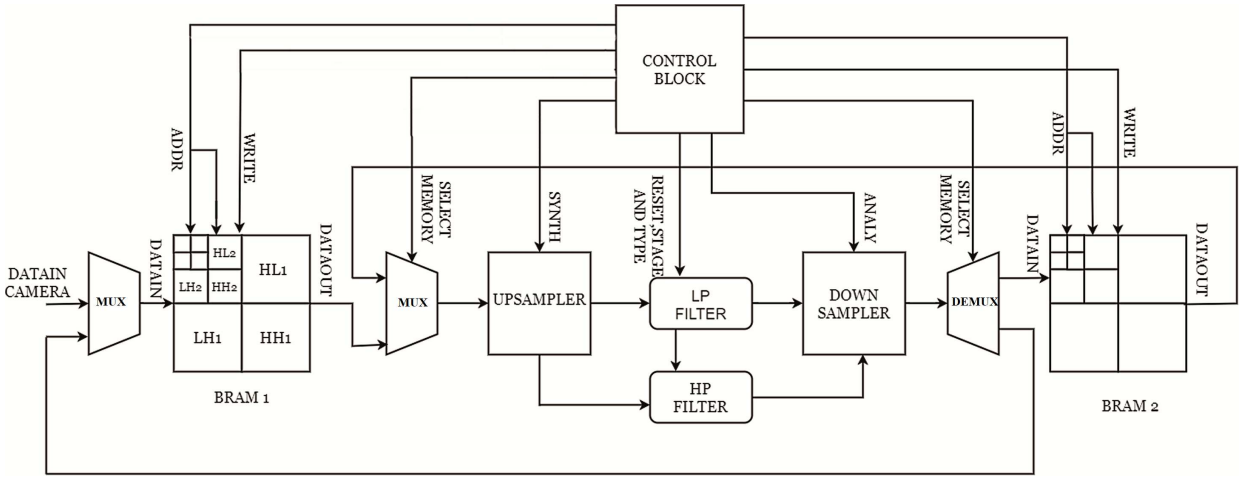


Fig. 2: Datapath of Proposed Architecture

and the other half stores the high pass filtering results. Column processing creates four sub-bands as shown in Fig. 3 and a similar memory addressing is followed as in the case of row processing but this time alternating between sub-bands column-wise. After every stage, further processing is based only on the LL band and thereby, the counter used in the address generation unit is modified accordingly by dropping the most significant bit.

C. Addressing Schemes

The memory addressing schemes are important elements that ensure the working of this algorithm. There are 2 different schemes followed for different situations which are listed below -

Reading data from memory for filtering: This is required because of the nature of the Q-shift filters chosen. The impulse response of the total system produces a shifted version of input as the output due to the delay added by filtering. To counteract this effect, the input is pre-shifted and passed to the filter.

Writing data to memory after filtering: After filtering along rows and columns the data needs to be written to the memory such that the low pass and high pass components occupy their respective positions in the memory.

D. Upsampling and Downsampling

The single-port memory architecture creates two problems: The data after analysis filtering from both high pass and low pass filters arrive at the output port simultaneously which needs to be written after downsampling. Similarly for the synthesis stage, multiple data needs to be read simultaneously from the memory, upsampled and then sent to the synthesis filters. To take care of the above issues the following methods were adopted.

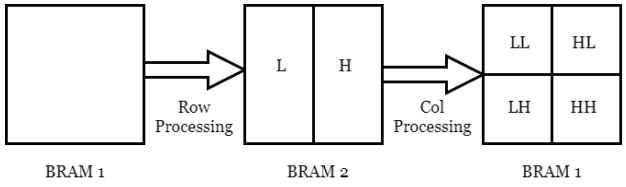


Fig. 3: Memory storage mechanism

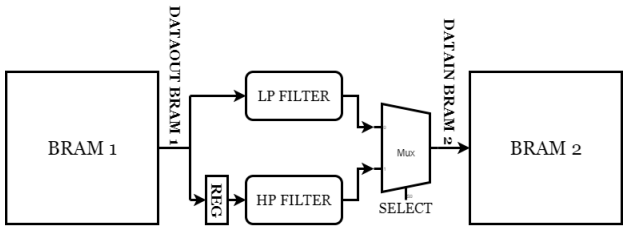


Fig. 4: Downsampler Architecture

1) Downsampling: Data before analysis filtering is read and simultaneously sent to the low-pass and high-pass filters. A multiplexer at the output of the filters chooses between the low-pass and high-pass filter outputs. This delay-multiplexing operation allows a single valid output from either filter to be available at the multiplexer at any given time which is then chosen using the *SELECT* signal as illustrated in Fig. 4

2) Upsampling: Because of the nature of the memory used, at any given time, only a single value can be read from the memory. To ensure proper synchronization at the output of the filters, we use the method as illustrated in Fig. 5. The two multiplexers at the inputs of the two filters are controlled by complementary signals. The signal chooses between the data or a zero sample. Since, the controls are complementary, when one filter chooses a data sample, the other would choose a zero. To achieve the final synchronization, a delay is introduced between the multiplexer corresponding to the low pass filter and the low pass filter itself.

E. Convolution

The filtering process, which is nothing but a convolution is carried out with a single, convolution unit with parallel multipliers which produces convolution outputs every clock.

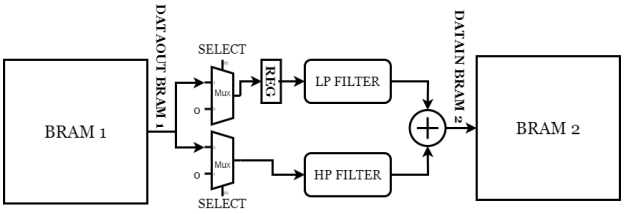
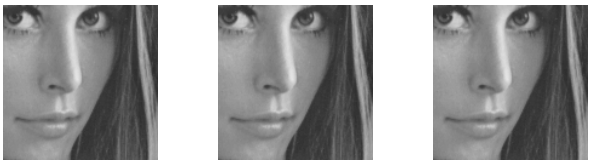


Fig. 5: Upsampler Architecture



(a) Original image (b) S/w output (c) H/w output

Fig. 6: Implementation results shown using lena image

We are using 10-tap filters and hence, 10 parallel multipliers. The same convolution unit is used for all the filters while only changing the filter coefficients used.

IV. Results

The algorithm was first implemented on Python3.6. The software implementation emulates the hardware except for the memory and data formatting used. The hardware was generated using code written in Verilog HDL and the Cadence® Design Suite was used to implement the algorithm on hardware. The Semi Conductor Laboratory (SCL) 180nm technology standard cell library is used [10]. A post-synthesis simulation was performed and the data output values are written to a file. The synthesis results are given in Table I. The proposed design runs at a maximum clock frequency of 111 MHz. The high clock speed enables this design to work with high frame rate videos as well.

TABLE I: Hardware Synthesis Results

Elements	Instances
Sequential Elements	788637
Inverters	4013
Buffers	137
Logic	1398445

Python3.6 is used to read the hardware output file and reconstruct the image. It was observed that the reconstructed image differed from the input image by a maximum of only 1 bit at LSB. The outputs are shown in Fig. 6. As observed, there is no visible difference between the software and hardware reconstructions.

A. Clock Frequency Estimation

As intended, the design can work well for real-time video implementations with high frame rates. A brief calculation highlighting this is given below:

We are assuming that the system is working in conjunction with a camera that delivers images at the rate of 30 picture frames per second. The design handles 128x128 size images. Since, the memory is a single-port memory, it requires 128x128 cycles to load the input image ($N_{Dataread}$). The first stage of analysis requires 128x128 cycles to read the data for column processing from the primary memory and 128x128 cycles to read the data from the secondary memory for column processing. Following this trend and as the image size shrinks by a factor of two along each dimension after every stage, we have -

$$\begin{aligned} N_{Analysis} &= (128 * 128 + 64 * 64 + 32 * 32) * 2 \\ &= 43008 \ cycles \end{aligned}$$

In the case of synthesis, at the first stage, it requires 32x32 reads for column processing for reading four 16x16 images followed by 32x32 reads for the row processing. This pattern pans out across stages and the total number of cycles consumed include -

$$\begin{aligned} N_{Synthesis} &= (128 * 128 + 64 * 64 + 32 * 32) * 2 \\ &= 43008 \ cycles \end{aligned}$$

In total, the number of cycles required to complete this operation (inclusive of the data read) is given by -

$$\begin{aligned} N_{Cycles} &= N_{Analysis} + N_{Synthesis} + N_{Dataread} \\ &= 43008 + 43008 + 16384 \\ &= 102400 \ cycles \end{aligned}$$

To compute the final clock frequency, we use -

$$\begin{aligned} T_{required} &= Period_{clock} * N_{Cycles} \\ \frac{1}{30} &= T_c * 102400 \\ f_c &= \frac{1}{T_c} = 3.072 \ Mhz \end{aligned}$$

Synthesising the design using this clock frequency, results in a positive slack of 297916 ps. Having a higher margin allows the design to be used for other image processing algorithms. The synthesis report gives a total power of 356.3 mW for 3.072 MHz clock frequency.

V. Conclusions and Future Work

An ASIC design was proposed to perform the perfect reconstruction of an input gray scale image using the dual-tree wavelet transform. The proposed architecture consists of a single convolution unit and a simple, yet effective upsampling and downsampling system that successfully reconstructs the image with only a 1-bit pixel error. The design allows enough compute time to include other algorithms like denoising and feature extraction whilst still maintaining real-time performance at higher frequencies. To further improve performance, a pipelined architecture of the unit under discussion can be devised along with a parallel arrangement of such units to cater to higher image resolution.

REFERENCES

[1] I. W. Selesnick, R. G. Baraniuk, and N. G. Kingsbury, "The dual-tree complex wavelet transform," *IEEE signal processing magazine*, vol. 22, no. 6, pp. 123–151, 2005.

[2] F. Canbay, V. E. Levent, G. Serbes, S. Goren, and N. Aydin, "Field programmable gate arrays implementation of dual tree complex wavelet transform," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2015, pp. 6026–6029.

[3] F. Canbay, V. E. Levent, G. Serbes, H. F. Ugurdag, S. Goren, and N. Aydin, "An area efficient real time implementation of dual tree complex wavelet transform in field programmable gate arrays," in *2015 IEEE 15th International Conference on Bioinformatics and Bioengineering (BIBE)*. IEEE, 2015, pp. 1–5.

[4] T. Y. Kumar, S. Pinjare, and P. C. P. Raj, "Image processing architecture using dtcwt modified distributed algorithm for plant phenotyping," in *International Conference on Intelligent Data Communication Technologies and Internet of Things*. Springer, 2018, pp. 385–396.

[5] S. Padmavati, V. Meshram *et al.*, "A hardware implementation of discrete wavelet transform for compression of a natural image," in *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies*. IEEE, 2017, pp. 1–5.

[6] N. Kingsbury, "Image processing with complex wavelets," *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1760, pp. 2543–2560, 1999.

[7] A. F. Abdelnour and I. W. Selesnick, "Design of 2-band orthogonal near-symmetric cqf," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, vol. 6. IEEE, 2001, pp. 3693–3696.

[8] N. Kingsbury, "Complex wavelets for shift invariant analysis and filtering of signals," *Applied and computational harmonic analysis*, vol. 10, no. 3, pp. 234–253, 2001.

[9] C. Valens, "A really friendly guide to wavelets," *ed. Clemens Valens*, 1999.

[10] "Semi Conductor Library," http://www.scl.gov.in/respond.html, 2008, [Online; accessed 19-July-2008].