A High Speed Architecture for Lifting-based 2-D Cohen-Daubechies-Feauveau (5,3) Discrete Wavelet Transform used in JPEG2000

Mohammad Rafi and Najeed-ud-Din

Abstract—**For real-time applications, efficient VLSI implementation of DWT is desired. In this paper, DWT architecture based on retiming for pipelining and unfolding is presented. The architecture is based on lifting one-dimensional Cohen-Daubechies-Feauveau (CDF) (5,3) wavelet filter, which is easily extended to 2-D implementation. It consists of low complexity and easily repeatable components. This paper is focused on the critical path minimization and throughput optimization at the same time. The architecture has been implemented on Virtex 6 Xilinx FPGA platform. The implementation results show that the critical path is minimized four to five times, while throughput is doubled, making the overall architecture approximately ten times faster when compared with the conventional lifting-based DWT architecture. Further with parallel implementation, the throughput has doubled without any increase in number of row buffers, implying that the architecture is memory efficient as well. The even and odd rows of the image are scanned in parallel fashion. To perform the 2-D DWT transform of an image of size 15 Megapixels, it takes 16.86 ms, which implies 59 images of that size can be processed in one second. This can be utilized for real-time video processing applications even for high resolution videos.**

I. Introduction

The discrete wavelet transform (DWT) has completely replaced the discrete cosine transformation (DCT) in image coding because it supports progressive image transformation, multi-resolution, ease of compressed image manipulation, region of interest coding, etc. Traditionally, DWT was implemented using convolution. Such an implementation requires both, a large number of computation and a large storage which are undesirable for any high speed or low power application. A new mathematical formulation that replaces the convolution-based wavelet transformation [1]–[4] has been proposed by Sweldens [5], [6], namely lifting-based wavelet transformation. The main feature of the lifting-based discrete wavelet transform scheme is to break up the high-pass and low-pass wavelet filters into a sequence of smaller filters that in turn can be converted into a sequence of upper and lower triangular matrices. The idea behind the lifting scheme is to use data correlation to remove the redundancy. Some of the advantages of this reformulation of the DWT includes ”in-place” computation of the DWT, integer-to-integer wavelet transform (IWT), symmetric forward and inverse transform, suitability of parallelism and many more [7]. The lifting scheme decomposes every DWT operation into a sequence of lifting steps. The basic steps involved are splitting, predicting and updating [8]. Further the filters may be classified into 2M consisting of one predict and one update step and 4M consisting of two predict and two update steps.

The state-of-the-art compression technique, JPEG2000, is striving for the development of efficient architecture of wavelet transform. Presently JPEG2000 uses Cohen-Daubechies-Feauveau (5,3) and (9,7) wavelet filters for lossless and lossy compression schemes respectively [9]–[11]. CDF (5,3) wavelet filter is 2M based, while CDF (9,7) is 4M based; 2M consists of one predict and one update stage, while 4M consists of two predict and two update stages. (5,3) indicate that the number of highpass and lowpass filter taps are 5 and 3 respectively. Since (5,3) and (9,7) are used in JPEG2000, lot of work has been done for their efficient implementation. The parameters under consideration are: speed, throughput, computational complexity and memory reduction. A few papers have worked on reduction of critical path as well.

Advantages of lifting-scheme over convolution-based has been presented in [12] for wavelet (9,7). [13] has presented a survey on different VLSI architectures on lifting based DWT. Architectures for reduction of memory accesses and hardware complexity have been proposed in [14]–[17]. Throughput optimization has always had a trade-off with architectural area. Few papers that have proposed architectures for throughput optimization are [16], [17]. Speed can also be increased by reducing the critical path delay of a design. But critical path reduction is always obtained at the cost of increase in latency and number of registers. The papers that have worked on critical path minimization are [18], [19]. The implementation of most of these architectures is based on FPGA using VHDL synthesis, however MATLAB/Simulink/Xilinx System Generator can also be utilized for the same [20], which helps portability and rapid time-to-market of the architecture. Many of the papers have aimed at either multiplier less architecture or shift based multipliers [14]. The other wavelet filters which are frequently considered for optimization are Daubechies-4 (D4), Daubechies-6 (D6), Daubechies-8 (D8), CDF (2,2), (6,10) etc. Universal embedded hardware implementation of a variety of wavelet kernels have been implemented in [8], [21]. The implementation methods are either based on parallel architecture of each kernel or processing element (PE). The parallel

Manuscript received October 7, 2016, revised January 31, 2017

M. Rafi is with the Department of Electronics and Communication Engineering, National Institute of Technology, Srinagar 190 006 India (phone: +91-9622-515357; e-mail: mohdrafi_11phd13@nitsri.net).

Najeeb-ud-Din is with the Department of Electronics and Communication Engineering, National Institute of Technology, Srinagar 190 006 India(e-mail: najeeb@nitsri.net).

method implements multiple wavelet kernels in parallel, which helps in increasing speed at the cost of some extra hardware. While in processing element method, resources are shared between different wavelet kernels, hence lesser resources are needed as compared to parallel implementation.

The proposed work exploits the critical path of the design by using algorithms for retiming for pipelining and unfolding. The throughput of the proposed architecture has increased while as the computation time has reduced when compared with the conventional lifting architecture. The rest of the paper is organized as follows. Section II provides a brief introduction of the CDF (5,3) filter and its implementation. Section III describes the proposed algorithm and architecture. The implementation results are provided in Section IV. Finally, concluding remarks are given in Section V.

II. 2-D CDF (5,3) Discrete Wavelet Transform

A. Mathematical Formulation

The standard lifting scheme has been divided into three stages: Split, predict and update as discussed in [5] and [7].

Split: The original input sequence and the filter coefficients are split into two branches of even and odd components. To make sure that each branch gets only its desired components and the output samples remain the same as obtained by convolution method, down-sampling by 2 is required in each branch.

Predict: Generate the prediction residual d[n] as the error in predicting odd samples from even input samples using predictor P.

Update: The coarse approximation c[n] is accomplished by applying an update operator U to d[n] and adding to even input samples.

The CDF (5,3) wavelet filter has the following coefficients:

Lowpass: (-1/8, 2/8 ,6/8 ,2/8, -1/8);

Highpass: (-1/2, 1, -1/2).

The polyphase matrix of the filter is:

$$P_1(z) = \begin{bmatrix} -\frac{1}{8}z + \frac{6}{8} - \frac{1}{8}z^{-1} & \frac{2}{8} + \frac{2}{8}z \\ -\frac{1}{2} - \frac{1}{2}z^{-1} & 1 \end{bmatrix} \tag{1}$$

The Split stage is given by factorization of the polyphase matrix

$$P_1(z) = \begin{bmatrix} 1 & 0.25(1+z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -0.5(1+z^{-1}) & 1 \end{bmatrix} \tag{2}$$

The predict step can be interpreted by following equation

$$y_{2i+1} = x_{2i+1} - 0.5(x_{2i} + x_{2i+2}), \tag{3}$$

The update step can be interpreted by following equation

$$y_{2i} = x_{2i} + 0.25(y_{2i+1} + y_{2i-1}) \tag{4}$$

Where x_{2k} and x_{2k+1} are the even and odd input samples and y_{2k+1} and y_{2k} represent the low and the high output coefficients respectively [22].

B. VLSI Architecture

The basic architecture of (5,3) is implemented in [14], [21] and [16]. [14] provides the conventional lifting based design of 2D DWT, as shown in Fig. 1. It consists of two stages of 1-D DWT, each stage having different length of delay element R(n). In the first stage (the Row processor), R(n) represents one delay element while in the second stage (the column processor), R(n) is N delays where N is the number of pixels in each row of the original image, as shown in Fig. 2. The input image $(N \times N)$ is fed to the architecture pixel by pixel using row by row scanning. In each clock cycle, a single pixel is fed. In the row processor, 1-D DWT of each row is computed to yield the low and high frequency components of each row. Then the column processor computes full set of 2-D DWT components; low-low (LL), low-high (LH), high-low (HL) and high-high (HH).

In the row processor, the input stream is split into odd and even streams, then predict and update stages follow. The pixels that take part in the computation of present output y[n] are x[n], x[n-1], x[n-2], x[n-3], and x[n-4]. Hence four delay elements are needed here. Similarly, in the column processor, five pixels from a single column are to be accessed for computation, so four row buffers (R(n) = N) are to be used.

III. Proposed Architecture

Timing refers to the logic delays between sequential elements. When a design does not meet timing, we mean that the delay of the critical path, that is, the largest delay between flip-flops (composed of combinatorial delay, clk-to-out delay, routing delay, setup timing, clock skew, and so on) is greater than the target clock period. In other words, the critical path delay sets upper limit on the clock frequency of a design. The standard metrics for timing are clock period and frequency [23].

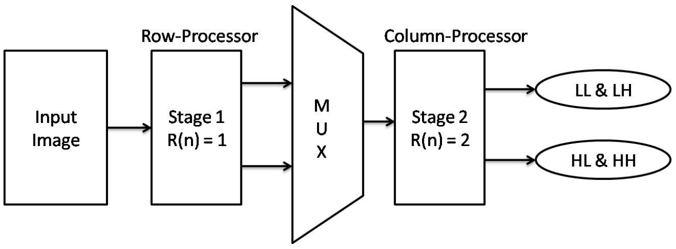

Fig. 1. Single level 2D CDF (5,3) DWT

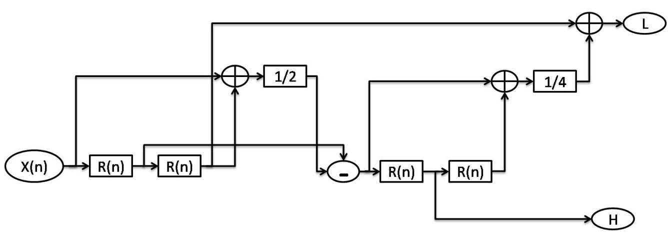

Fig. 2. 1-D DWT processor (Row or Column)

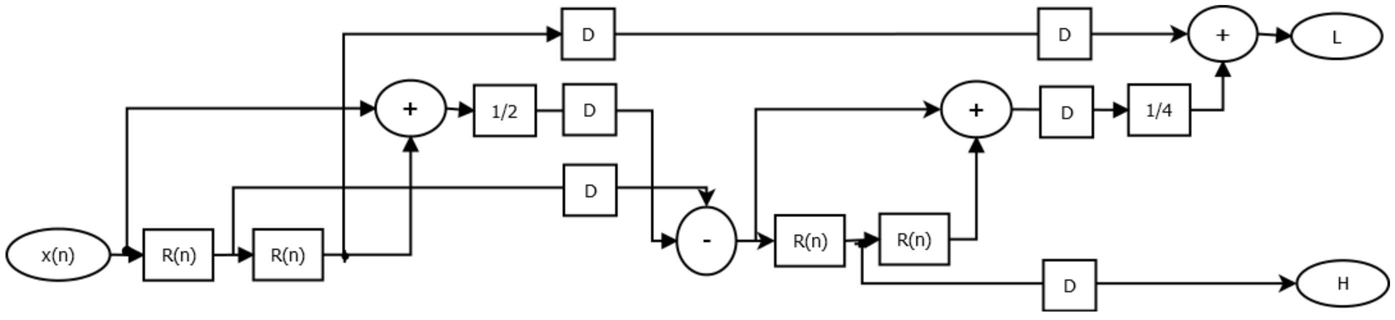

Fig. 3. Proposed 1-D DWT processor (row or column)

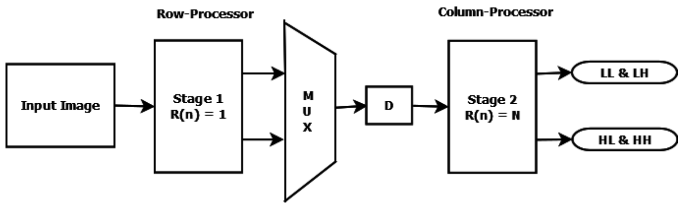

Fig. 4. Proposed single level 2-D CDF (5,3) DWT

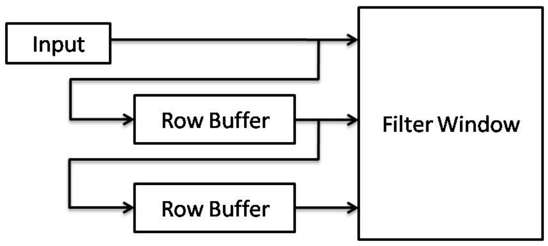

Fig. 5. Stream process filtering

Two types of optimizations are proposed in this paper:

1) *Pipeline retimed architecture:*
2) *Unfolding retimed architecture:*

A. Pipeline retimed architecture

The latency of the conventional structure is $2N + 2$ which means 1026 clock cycles for 512×512 image or 2050 clock cycles for an image of size 1024×1024. A small increase of 5 more clock cycles will have no effect on overall latency. But this will definitely help in optimization of timing. In order to increase the clock frequency, the critical path delay has to be minimized [23], [24]. The conventional implementation had a critical path delay of 8 adders and 4 multipliers, which will definitely needs to be minimized. [25] has provided two algorithms to check the feasibility of pipelining for obtaining a reduced clock period. Using Floyd-Warshall algorithm, we find that there is a feasibility of reducing the clock period. Now the pipelined architecture is obtained by cutset technique, in which a set of edges is removed from the signal flow graph in such a way that it creates two disconnected subgraphs. Then, a delay element is added in each of these edges. The resulting 2-D DWT for reduced clock period is shown in Fig. 4, with its 1-D DWT processor shown in Fig. 3.

As is evident from the Fig. 3, we have inserted delays in the cut-set tree so that the combinational critical path be evenly or near evenly divided. Now the critical path is 2 adders or 1 adder and 1 shifter. We can say the critical path has been reduced from $8T_A + 4T_M$ to $2T_A$ or $T_A + T_S$, where T_A, T_M and T_S are the computation times of adder, multiplier and shifter. The implementation results show that the minimum critical path delay has reduced from 13.566 ns to 3.01 ns, which is fourfold decrease in critical path delay. Conversely we can say that the proposed architecture is four times faster.

B. Unfolding retimed architecture

The filter function and the clock speed of the architecture are fixed to their optimal levels so far. We need to improve its throughput and latency, which can be accomplished by simultaneously processing multiple adjacent rows [26]. This requires multiple pixels to be input per clock cycle and exploits the fact that the windows for vertically adjacent outputs overlap significantly, as can be seen in Fig. 5 and Fig. 6. This is partially unrolling the vertical scan loop through the image. Note that the number of row buffers is unchanged. For an unroll factor of k (processing k rows of pixels in parallel) the combined window size is $W \times (W + k - 1)$. Of this, k rows of data are streamed in from input in parallel, so the remaining W-1 rows must come from row buffers. These buffers are arranged with a pitch of k rather than simply being chained. The parallel implementation will require k copies of

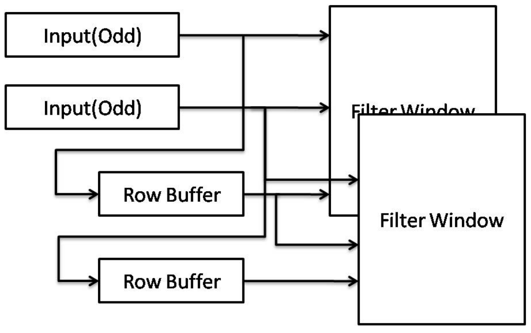

Fig. 6. Unfolded structure

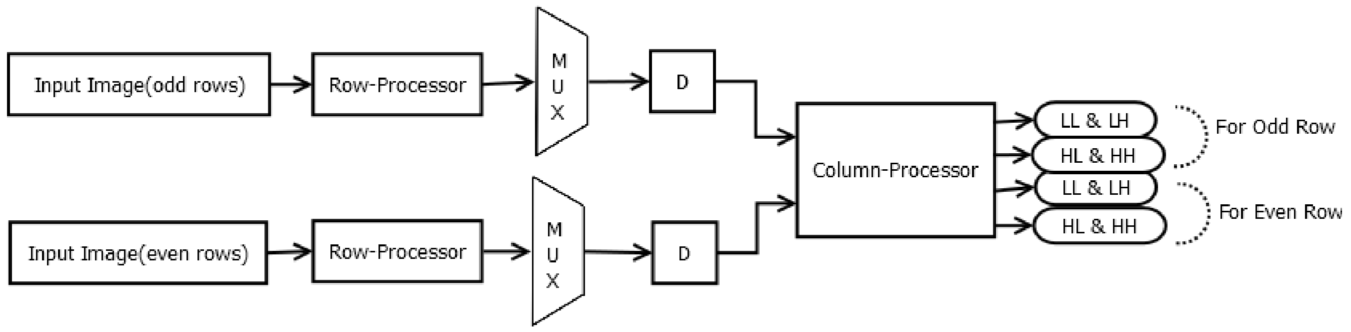

Fig. 7. Unfolded structure of 2-D DWT

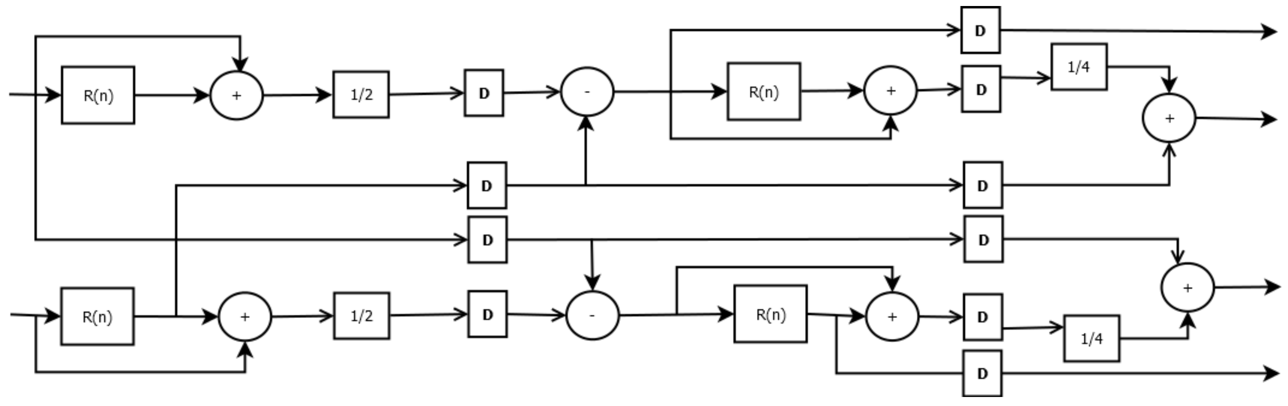

Fig. 8. Unfolded column-processor architecture

the filter function. However, with some filters, the overlap in windows can even enable some of the filter function logic to be shared [27], reducing the resource requirements further. Our case corresponds to k = 2, so two parallel inputs will scan the image in alternate manner, one input scanning the odd rows of the image in pixel by pixel manner while other scanning the even rows of the input image.

Here two rows (even and odd) are processed in parallel with resources shared between the two parallel branches. The actual implementation is shown in Fig. 7, with column processor architecture as shown in Fig. 8. It is evident from Fig. 8 that two transformed coefficients are output in each clock cycle, so throughput is improved and reaches twice its initial value. Also now the output latency has been reduced to $N + 7$.

1) Multiplier Design: Multiplication in binary can be represented as a series of addition and shift operations. Usually multiplication requires more logic and computation time than addition. So, it is always a good practice to minimize the use of multipliers in hardware designs. Here, we have multiplication with the coefficients having values 0.5 and 0.25, which can be easily obtained using shift operations. Multiplication of the input with $0.5(= 2^{-1})$ and $0.25(= 2^{-2})$ requires the input to be shifted right by one and two bits respectively.

IV. Implementation Results

We have proposed two 2-D DWT architectures of CDF (5,3) in this paper. The structures are synthesized on 6VLX760FF1760-2 Xilinx FPGA platform. The word length is set to 16 bit using signed arithmetic operations, while evaluating the design. The architecture is designed for image size of 512×512 and can be easily extended for any even-sized image without modifying much. Also the hardware can be easily replicated for J number of levels. Zero padding is used for the computation of the whole image, in order to take care of the filter effects on image boundary. The architecture of [14] is also implemented alongside our proposed architectures to reflect the optimizations made therein. The comparison of resource utilization and timing is made in table I and table II. It is evident from table I that the hardware requirements of the proposed architecture has increased but the cost to be paid for the extra hardware is much less compared to the high performance achieved from the proposed design, as shown in table II.

It should be noted that the maximum clock frequency of the pipeline retimed architecture has drastically improved by minimizing the critical path delay. This has cost as many as 13 additional delay elements in total, maximum of five delays in feed-forward path i.e. five more additional latency cycles. The register balancing option available in Xilinx ISE has reduced four delay registers without affecting any other parameter. The unfolded retimed architecture has paralleled the pipeline retimed architecture to yield twice the throughput and half the output latency, by doubling the filter function, without any addition to the row buffers. So unfolded retimed architecture provides optimized throughput, reduced latency as well as optimized clock frequency design. In table III, the performance of the proposed architecture is compared with other existing efficient architectures. It is evident that our

TABLE I
SYNTHESIS RESULTS FOR HARDWARE UTILIZATION OF 2D CDF 5/3 DWT ARCHITECTURE AND COMPARISON WITH EXISTING ARCHITECTURE (J: NUMBER OF DWT LEVELS)

Architecture	**Adders/Subtracters**	**Multipliers/Shifters**	**Registers**	**Slice Registers**	**Slice LUTs**	**IOBs**	**Control Complexity**
Al_Azawi [14]	8J	4J	4N + 4	359	368	26	Simple
Proposed pipelined architecture	8J	4J	4N + 13	411	361	26	Simple
Proposed unfolded architecture	16J	8J	4N + 25	511	433	50	Medium

TABLE II
TIMING PERFORMANCE COMPARISON OF 2D CDF 5/3 DWT ARCHITECTURE WITH EXISTING ARCHITECTURE

Architecture	**Output Latency (clock cycles)**	**Computation Time (clock cycles)**	**Critical Path (combinational logic delay)**	**Critical Path Delay (ns)**	**Minimum Clock Period (ns)**	**Throughput (samples per clock cycle)**
Al_Azawi [14]	$2N+2$	N^2+2N+2	$8T_A+4T_M$	13.566	5.930	1
Proposed pipe_arch	$2N+7$	N^2+2N+7	$2T_A$ OR T_A+T_S	3.010	2.228	1
Proposed unfold_arch	$N+7$	$N^2/2+N+7$	$2T_A$ OR T_A+T_S	3.040	2.228	2

TABLE III
HARDWARE AND TIMING PERFORMANCE COMPARISON AMONG DIFFERENT EXISTING 2D CDF 5/3 DWT ARCHITECTURES

Architecture	**Multipliers**	**Adders**	**Storage Size**	**Computation Time (clock cycles)**	**Output Latency (clock cycles)**	**Throughput (samples per clock cycle)**	**Critical Path (combinational delay)**	**Control Complexity**
Wu [29]	16	16	$N^2/4$	$N^2/2$	2N	1	T_M+2T_A	Medium
Andra [7]	4	8	$N^2/4$	$N^2/2$	2N	1	T_M	Medium
Laio [30]	4	8	$6N$	N^2	2N	1	T_M+2T_A	Complex
Barua [17]	4	8	$N^2/4$	$N^2/2$	5N	1	–	Simple
Chengy (FA) [28]	4	8	$N^2/4$	$N^2/2$	N	1	T_M+2T_A	Simple
Chengy (HA) [28]	8	16	$N^2/4$	$N^2/4$	$N/2$	1	T_M+2T_A	Simple
Tian [31]	8	16	$N^2/2$	$N^2/2$	2N	2	T_M+2T_A	Simple
Al_Azawi [14]	4	8	$4N+4$	N^2+2N+2	$2N+2$	1	$4T_M+8T_A$	Simple
Proposed architecture	8	16	$4N+25$	$N^2/2+N+7$	$N+7$	2	$2T_A$ OR T_A+T_S	Medium

architecture is the only one that provides optimized throughput along with minimized clock period to such extent.

We have tested the implementation of the proposed architecture on several images of sizes 0.25 Megapixels (512×512), 5 Megapixels (2560×1920), 15 Megapixels (5184×2920) and 20 Megapixels (5184×3888). The computation time for the four image sizes are 0.397 ms, 5.84 ms, 16.86 ms and 22.50 ms respectively. The corresponding number of images that can be transformed per second are 2500, 171, 59 and 44 respectively.

V. Conclusion

In this paper, we have presented a high speed memory-efficient architecture that can convert images into their corresponding 2D-DWT coefficients at very high speed. The simulation results depict that approximately 4 rows of input image need to be stored in buffers for encoding an image. The proposed architecture can encode as many as 59 images, each of size 15 megapixels or 45 images, each of size 20 megapixels in approximately one second, making our architecture a potential candidate for video processing applications. Even higher resolution images can be transformed without affecting the video quality. Furthermore, the proposed architecture is quite simple and can be applied to images having even size.

References

[1] M. Vishwanath, R. Owens, and M. J. Irwin, "VLSI architectures for the discrete wavelet transform," *IEEE Trans. on Circuits Syst. II*, vol. 42, pp. 305-316, May 1995.

[2] K. K. Parhi and T. Nishitani,"VLSI architectures for discrete wavelet transforms," *IEEE Trans. on VLSI Systems*, vol. 1, pp. 191-202, June 1993.

[3] A. Grzeszczak, M. K. Mandal, S. Panchanathan, and T. Yeap,"VLSI implementation of discrete wavelet transform," *IEEE Trans. on VLSI Systems*, vol. 4, pp. 421-433, June 1996.

[4] C. Chakrabarti and M. Vishwanath, "Efficient realizations of the discrete and continuous wavelet transforms: From single chip implementations to mappings on SIMD array computers," *IEEE Trans. on Signal Processing*, vol. 43, pp. 759-771, March 1995.

[5] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting schemes," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 247-269, May 1998.

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

[7] A. R. Calderbank, I. Daubechies,W. Sweldens, and B. L. Yeo, "Wavelet transforms that map integers to integers," *Applied Computational Harmonic Analysis*, vol. 5, pp. 332-369, July 1998.

[8] K. Andra, C. Chakrabarti and T. Acharya, "A VLSI Architecture for Lifting-Based Forward and Inverse Wavelet Transform," *IEEE Trans. on Signal Processing*, vol. 50, no. 4, April 2002.

[9] JPEG2000 verification model 8.5 (Technical Description), Sept 13, 2000.

[10] ITU-T Recommend. T.800-ISO FCD15444-1: JPEG2000 Image Coding System. International Organization for Standardization, ISO/IEC JTC1 SC29/WG1, 2000.

[11] M. Unser and T. Blu, "Mathematical Properties of the JPEG2000 Wavelet Filters," *IEEE Trans. on Image Processing*, vol. 12, no. 9, September 2003.

[12] W Wang, Z. Du and Y. Zong, "High-Speed FPGA Implementation for DWT of Lifting Scheme," *5th International Conference on Wireless Communications, Networking and Mobile Computing*, September 2009.

[13] U. Bhanu N and A. Chilambuchelvan, "A Detailed Survey on VLSI Architectures for Lifting based DWT for efficient hardware implementation," *International Journal of VLSI design & Communication Systems (VLSICS)*, vol.3, no.2, April 2012.

[14] S. Al-Azawi, Y. A. Abbas and R. Jidin, "Low Complexity Multidimensional CDF 5/3 DWT Architecture," *9th International Symposium on Communication Systems, Networks & Digital Sign (CSNDSP)*,2014.

[15] X. Fan, Z. Pang, D. Chen and H. Z. Tan, "A Pipeline Architecture for 2-D Lifting-based Discrete Wavelet Transform of JPEG2000," *International conference on Multimedia Technology*, October 2010.

[16] A. D. Darji, R. Bansal, S. N. Merchant and A. N. Chandorkar, "High Speed VLSI Architecture for 2-D Lifting Discrete Wavelet Transform," *Conference on Design and Architectures for Signal and Image Processing*, November 2011.

[17] S. Barua, J. E. Carletta, K. A. Kotteri and A. E. Bell, "An efficient architecture for lifting-based two-dimensional discrete wavelet transforms," *Integration, the VLSI journal*, vol. 38, pp. 341-352, 2005.

[18] W. Zhang, Z. Jiang, Z. Gao, and Y. Liu, "An Efficient VLSI Architecture for Lifting-Based Discrete Wavelet Transform," *IEEE Trans. on Circuits and SystemsII: Express Briefs*, vol. 59, no. 3, March 2012.

[19] V. Gupta and K. Raj, "An Efficient Modified Lifting Based 2-D Discrete Wavelet Transform Architecture," *1st International Conference on Recent Advances in Information Technology*, 2012.

[20] T. S. M. Atri, Y. Said and R. Tourki, "Real Time FPGA acceleration for Discrete Wavelet Transform of the 5/3 Filter for JPEG 2000," *6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, 2012.

[21] C. Desmouliers, E. Oruklu and J. Saniie, "Discrete wavelet transform realisation using run-time reconfiguration of field programmable gate array (FPGA)s," *IET Circuits Devices Syst.*, vol. 5, no. 4, pp. 321-328, 2011.

[22] N. Sriram and R. Shyamsunder, "3-D medical image compression using 3-D wavelet coders," emphDigital Signal Processing, vol. 21, pp. 100-109, 2011.

[23] S. Kilts, "Advanced FPGA Design Architecture, Implementation, and Optimization," *John Wiley & Sons, Inc., Hoboken, New Jersey*, 2007.

[24] D. G. Bailey, "Design for Embedded Image Processing on FPGAs," *John Wiley & Sons (Asia) Pvt. Ltd.*, 2011.

[25] K. K. Parhi, "VLSI Digital Signal Processing Systems: Design and Implementation," *John Wiley & Sons (New York) Pvt. Ltd.*, 1999.

[26] B. A. Draper, J. R. Beveridge, A. P. W. Bohm, C. Ross, M. Chawatche and J. Hammes, "Accelerated image processing on FPGAs," *IEEE Trans. on Image Processing*, 2003.

[27] L. E. Lucke and K. K. Parhi "Parallel structures for rank order and stack filters," *IEEE International Conference on Acoustics, Speech and Signal Processing*, San Francisco, California, USA, vol. 5, March, 1992.

[28] 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, March 2007.

[29] P. C. Wu and L. G. Chen, "An Efficient Architecture for Two-Dimensional Discrete Wavelet Transform," *IEEE Trans. on Circuits and Systems for Video Technology*,vol. 11, no. 4, April 2001.

[30] H. Liao, M. Kr. Mandal and B. F. Cockburn, "Efficient Architectures for 1-D and 2-D Lifting-Based Wavelet Transforms," *IEEE Trans. on Signal Processing*, vol. 52, no. 5, May 2004.

[31] X. Tian, L. Wu, Y.-H. Tan, and J.-W. Tian, "Efficient Multi-Input/Multi-Output VLSI Architecture for Two-Dimensional Lifting-Based Discrete Wavelet Transform," *IEEE Trans. on Computers*, vol. 60, no. 8, August 2011.