https://dx.doi.org/10.22161/ijaers/3.9 *ISSN: 2349-6495(P) | 2456-1908(O)*

Implementation of DWT Integrated Log Based FPU with SPIHT Coders on FPGA

N.RamyaRani

Assistant Professor, Department of EEE, Sri Krishna College of Eng&Tech, Coimbatore, India

Abstract— *In this work, architecture is designed for integrating lifting based discrete wavelet transform (DWT) structure with logarithmic based floating point arithmetic units. As many algorithms were proposed for coding wavelet coefficients for image compression, Set-Partitioning in hierarchical trees algorithm (SPIHT) is found to be widely used due to its low–computational complexity and better method for compressing the images. However it is suffered from the drawback of occupying high memory space and hence produced less throughput. This drawback is overcome in this work by adopting modified SPIHT algorithm termed as block-based pass-parallel SPIHT (BPS) algorithm. The designed architecture is compared with multi-precision floating point arithmetic units and the synthesis results are presented. From the experimental synthesis results it is proved that the integration of DWT structure integrated with log based FPU core and BPS coder implemented on FPGA devices provided efficient area and high speed of computations. The proposed architecture is designed using Verilog HDL and synthesized on various Xilinx FPGA devices. The architecture designed in this work is useful for compressing the images with good compression ratio, better resolution of images and to obtain high peak to signal ratio.*

Keywords— Discrete Wavelet Transform, Log based IEEE FPU, SPHIT Algorithm, Block-based pass parallel SPHIT BPS Algorithm, FPGA, Verilog HDL.

I. INTRODUCTION

Image compression is one of the important techniques in the field of image processing. It is the representation of an image with few bits in digital form by retaining the quality of the image. Generally an image is represented as a matrix of pixel values. The objective of image compression is to reduce the redundancy of the image so that the data can either be stored or transmitted in an efficient manner. Lossless and loss image compressions are the two methods of compressing the images. In lossless compression, even after the file is uncompressed all the original data in the file can be completely restored. Whereas in loss compression, only a part of the original information can be restored after the file is uncompressed. In recent years wavelet transforms have become one of the powerful tools in the field of image processing particularly in image compression techniques.[28]. In conventional Fourier transforms and its related transforms namely Discrete Cosine transform and Discrete Sine transform, sinusoids are used for the basic functions. Such transform techniques provided only the frequency information and not temporal information. As many of the image processing applications are dependent on both the type of information, wavelet transform techniques are widely acceptable. This technique distinguishes the low frequency and high frequency signals so that the required range of frequencies can be compressed without distortion. Recently, Discrete type of wavelet transform is widely used in image processing as it performs multi resolution analysis and supports features like easy manipulation of compressed image with better quality and resolution. DWT are generally implemented by convolution method but it requires larger number of arithmetic computations and hence occupied more area when implemented on Xilinx FPGAs. Hence lifting based DWT scheme came into existence. This method requires lesser number of computations as it breaks up the high pass and low pass wavelet filters into a sequence of high and low triangular matrices.

This work is focused in the design and implementation of lifting based DWT architecture on FPGA. The architecture is developed with the aim of achieving less area, high speed of computations and low power consumption. Hence the computations are based on logarithmic floating point computations rather than conventional floating point arithmetic units. [14]. several algorithms have been developed for coding wavelet coefficients. As the objective of the work is to occupy less storage space and to achieve high throughput, block based pass-parallel SPIHT algorithm is employed with the integration of DWT structure. The hardware implementation of the architecture is done using Verilog HDL and synthesized in Xilinx FPGA devices. The proposed DWT architecture is proved to be efficient in terms of area and speed by comparing with the design of multi-precision floating point arithmetic designs and logarithmic floating point units that are described in our previous works. [7]

The rest of the paper is organized as follows. Section II discusses the survey of various methods used in the design of lifting based DWT structure and logarithmic arithmetic

units. Section III provides a background on discrete wavelet transform, inverse discrete wavelet transform and Set Partitioning in Hierarchical trees (SPIHT) Algorithm.Section IV, describes the proposed architecture of DWT structure with logarithmic units and modified BPS SPIHT algorithm and the methods for image fusion. [29]. Section V provides the hardware implementation of the architecture on FPGA devices[30]. Synthesis results of multi-precision floating point arithmetic designs and logarithmic floating point designs are presented and compared with the proposed architecture. Finally, Section VI concludes the work and areas for future study.

II. LITERATURE SURVEY

G. Govindu, R. Scrofano, and V. K. Prasanna et.al proposed the design of floating point arithmetic cores.[1] Based on these cores design of arithmetic and logic units were proposed based on IEEE 754 standard.[4].G. Govindu, R. Scrofano, and V. K. Prasanna et.al proposed the area efficient architecture for the design of arithmetic expresssions using the floating point cores.[2].An island-style with embedded FPU is proposed by Beauchamp et al., while a coarse-grained FPU was suggested by Ho et al. Even et al. suggests a multiplier for performing on either single-precision or double precision floating point numbers. [3].An optimized FPU in a hybrid FPGA was suggested by Yu et al. [5] and a configurable multimode FPU for FPGAs by Chong and Parameswaran. [6].The proposed work is based on reduced complexity of arithmetic computations. Hence log based arithmetic computations are preferred rather than floating point computations. Anand et al. proposed a log lookup table (LUT)-based FPU, which utilizes a logarithmic principle to achieve good accuracy with reduced power consumption. [8].The proposed method suggests an efficient model for the development of DWT architecture integrated with logarithmic floating point computations for performing image compression. Many algorithms are proposed for coding wavelet coefficients. Some of the algorithms are embedded zero tree wavelet (EZW), embedded block coding with optimized truncation (EBCOT) floating point operations and set partitioning in hierarchical trees (SPIHT). The implementation of the proposed lifting based DWT architecture on FPGA devices are designed with the objective of achieving less area and higher speed of computations. Hence the modified SPIHT algorithm termed as BPS coder is integrated with DWT structure and log based FPU in the proposed work to achieve high throughput and less area in the implementation of Xilinx FPGA devices. [17].

III. WAVELET TRANSFORMS AND SPIHT ALGORITHM

3.1 Convolution Based Discrete Wavelet Transform

The discrete wavelet transform(DWT) is generally implemented as binary tree of filters. The input signal x[n] is splitted by filters into a low pass and high pass component. It is then down sampled by 2 separately. The low pass component is then again splitted further and decimated again according to the requirements. The output of DWT are the band pass coefficients and final low pass coefficients. As decimation is done at each step the total ouput sample rate becomes eqaul to the input sample rate and hence there is no redundancy in the transform.

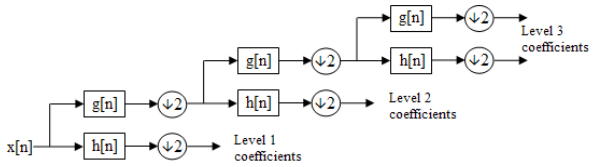

Fig.1: DWT Structure

The filtering steps are done based on multiply and accumulate operations. Depending on the charactersitics of the filter, the filtering operation extracts frequecny information from the data. The convolution based DWT computations suffers from high computational complexity and high memory utilization requirements.

3.2 Inverse Discrete Wavelet Tranform

The IDWT is exactly opposite to that of DWT. The low pass and high pass data streams are up sampled and then filtered using filters. The results are added together to produce the low pass result of the previous level. This result acn be combined with the high pass result to produce further levels and on continuing the process further the original stream of data can be reconstructed.

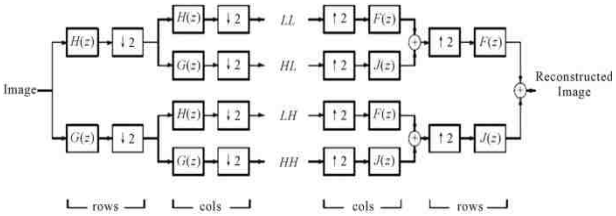

Fig.2: IDWT structure

3.3 Set partitioning in hierarchical trees (SPIHT) Algorithm

SPIHT algorithm is applied to a image in wavelet transformed domain. The image is represented as a spatial orientation tree (SOT) and the arrow represents the relationship between the parent and its off spring.Each node of the tree corresponds to a coefficient in the image. Fig 3a) represents the image of size 16×16 transfromed by 3-level DWT. The square denoted by R represents the root and 2×2 pixels numbered 0,1,2&3 corresponds to the root in fig3b).

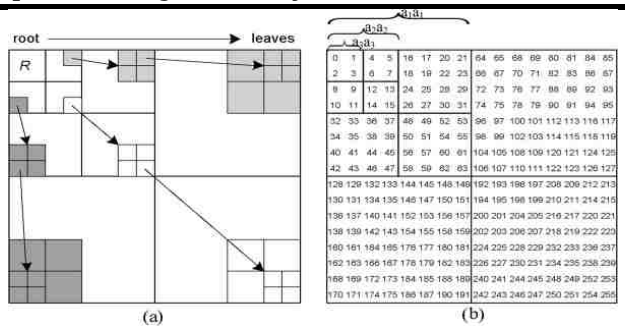

Fig.3: Spatial Orientation Tree b)Morton Scanning Order

For a given set T, SPIHT defines a function of significance that indicates whether the set *T* has pixels larger than a given threshold.

Sn(*T*), the significance of set *T* in the *n*th bit-plane is defined as

$$Sn(T) = 1, \max_{w(i) \in T}(|w(i)|) \geq 2^n \quad (1)$$

$$Sn(T) = 0, \text{else.} \quad (2)$$

When *Sn*(*T*) is "0," *T* is called an *insignificant set*. Else *T* is called a *significant set*. An insignificant set can be represented as a single-bit "0," but a significant set is partitioned into subsets, whose significances in turn are to be tested again. A SPIHT algorithm consists of three passes namely insignificant set pass (ISP), insignificant pixel pass (IPP) and significant pixel pass (SPP). [27]

According to the results of the (*n* + 1)th bit-plane, the *n*th bit of pixels are categorized and processed by one of the three passes. Insignificant pixels classified by the (*n* + 1)th bit-plane are encoded by IPP for the *n*th bit-plane whereas significant pixels are processed by SPP. The main goal of each pass is the generation of the appropriate bit stream according to wavelet coefficient information. If a set in ISP pass is classified as a significant set in the *n*th bit plane, it is decomposed into smaller sets until the smaller sets are insignificant or they correspond to single pixels. If the smaller sets are insignificant, they are handled by ISP. If the smaller sets correspond to single pixels, they are handled by either IPP or SPP depending on their significance. If the most significant bit-plane is a zero bit-plane, the bit-plane is not encoded, and consequently, the number of encoded bit-planes is decreased. The following significant bit planes are not encoded if they are also zero bit planes. SPHIT algorithm process all the three linked lists in FIFO order. This way of processing the pixels slow down the processing speed of computations. Hence modified SPIHT algorithm is proposed in this work.

IV. PROPOSED WORK

4.1 Lifting based DWT

In this scheme, the polyphase matrix of the wavelet is factorized into a sequence of alternating upper and lower triangular matrices and diagonal matrix. This scheme provides faster computations of the transforms.The polyphase matrix P(z) is splitted as follows:

$$\mathbf{P(z)}=\begin{bmatrix} h_e(z) & h_o(z) \\ g_e(z) & g_o(z) \end{bmatrix} \quad (3)$$

$h_e(z)$ *and* $h_o(z)$ deneote the even part and odd part of low pass filter analysis and $g_e(z)$ $g_o(z)$ represents the even and odd part of the high pass analysis. Then the function is factorised . Spiltting is the process of separating even and odd points. In the prediction step, the even samples are multiplied by time domain equivalent and are added to the odd samples. The odd samples are multiplied by the time domain equivalent of s(z) and are added to the even samples in the update process. Finally in scaling, the even samples are multiplied by 1/k and odd samples by k.[18]

4.2 Lifting Based DWTArchitecture

The lifting based DWT structure is integrated with logarithmic floating point units as shown in the figure. The implementation of architecture is mainly dependent

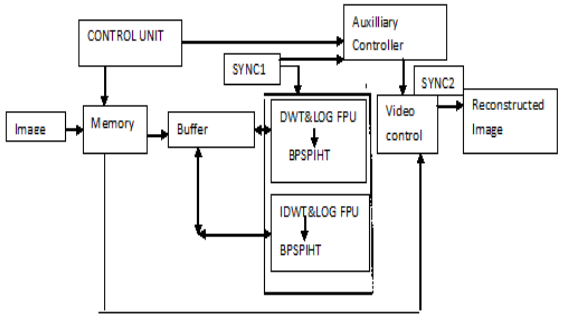

Fig.4: Lifting Based DWT with Log FPU and BPS

on the memory required for the computations. Among the various memory scan techniques available, such as line scan, block-based scan and stripe based scan, line based processing of wavelet coefficients is utilised in this work. [19-22].In this method, scan order is raster scan. An internal buffer of size LN is used, where N represents the number of pixels in a row and L represents the number of rows that are required for the computations. The presence of internal buffer avoids the accessing of external memory which in turn reduces the storage space. The inputs are fed from the memory and the synchronistion signals generated by the control units makes the memory block to be updated. These synchronisation signals are used to compensate the overall delay generated by the two critical path delays of multiplier and adder. The biorthogonal two dimensional discrete wavelet transform structure is computed in row- column manner.[23-26].The image stored in the external memory is read to the processing core in a row wise order. The horizontal filtering are done to the rows that consists of the computing modules as per the lifting scheme. Predict, update

and scaling process are performed depending on the number of modules. The resultant coefficeints are stored in the local memory or buffer.Then the column processor performs the vertical filtering to the same computing modules. The approximate coefficients considered as inputs are fed from the buffer and the sub band coefficients generated as the resultant are fed back to the external memory in row by row order. The operation is continued until all the required levels of wavelet decomposition are completed. In order to perform the computations faster and to provide area efficient design, logarithmic multipliers are designed in this work.[13]. The lifting based DWT structure integrated with logarithmic floating point units results in power and area reduction. The block based pass parallel SPIHT compression algorithm is utilised for coding the wavelet coefficients.

4.2.1 Logarithmic Number Systems

In contrast to floating point numbers, in logarithmic number systems the mantissa is always 1 and the exponent has a fractional part. Let the number A has a value of

$$A= -1^{s}_{A} \times 2^{E}_{A} \qquad (4)$$

S_A is the sign bit and E_A is a fixed point number. The sign bit represents the sign of the number and E_A represents the 2's complement fixed point number. Similarly logarithmic numbers can be represented in both very large and very small numbers.

Logarithmic number system based operations are very rare to identify the exceptions occurred during the computations. Hence these exceptions are identified with the representation of flag bits to specify the code for zero, +/- infinity, and NaN. [12]

The logarithmic number system has the similar range and precision of floating point number systems. For the sign bit, a=8 and the mantissa b=23bit of a single precision number, the range in LNS system is approximately $\pm 1.5\times10^{-39}$ to 3.4×10^{38}. For the double precision floating point number, LNS system has a range of approximately $\pm 2.8\times10^{-309}$ to $1.8 \times10^{308.}$. Thus, LNS representation covered the entire range of the corresponding IEEE 754 standard representation of floating-point numbers. [9]

4.2.2. Log based Multiplication

Multiplication operation involves a simple computation in Logarithmic Number System. [10]

The product is computed by adding the two fixed point logarithmic numbers as per the following logarithmic property:

$$\log_2 (x.y) = \log_2 (x) + \log_2 (y) \qquad (5)$$

The sign bit is computed by XORing the multiplier's and multiplicand's sign bits. The flags bits for infinities, zero, and NANs are encoded for the exceptions similar to IEEE 754 standard. Since the logarithmic numbers are the representation of a 2's complement fixed point numbers, if overflow doesn't exists, addition is an operation similar to floating point computation. And if overflow condition occurs it will result in $\pm\infty$. Overflow condition occurs, when the addition of two numbers of same bit size resulted with the increase in bit width. [11].In this work, log based 32 bit floating point multiplication was computed based on look up table method. [15].128 values of single precision floating point numbers is called as a function and accessed as 7 bit integer. Logarithmic addition was performed for logarithmic multiplication. For the corresponding multiplication result of logarithm computation the antilogarithm of the number is computed. [16]

4.2.3. Block-Based Pass-Parallel SPIHT

Let 8×8 block discrete wavelet transformed image be considered as the input to the BPS coder. The input image is sent to the bit slicing and it is decomposed into 4×4 bit blocks. BPS process one 4×4 bit block at a time. It is further decomposed into 2×2 blocks. The major block is represented as H and the sub blocks are represented as Q. According to the type of output bits generated as refining bits, sorting bits and first refinement bits, BPS consists of three passes termed as refinement pass (RP), sorting pass(SP) and first refinement pass(FRP) respectively. The RP is a combination of IPP and SPP as mentioned earlier in SPIHT algorithm and visits each sub blocks that is significant in the previous bit plane. In the next clock cycle, the insignificant bit planes are entered to sorting pass. SP transmits and generates the significance of the major block. From the significant bit stream of the insignificant sorting pass, the RP codes the significant micro blocks and generates the coded output. SP pass process the major block. RP and FRP pass process only the sub blocks. The controller stops the sorting pass when all the blocks in the 8×8 coefficients become significant. Thus unnecessary updating of the insignificant sorting passes is avoided in BPS algorithm which in turn increased the speed of computations.[27], [28].

4.3. Complex Wavelet Transform

Image fusion is the process of extracting useful information from two or more images and combined together to form a single image. DWT are widely used in the fusion of images. Real valued DWT transforms proposed in the work can also be used in image fusion but suffers from properties like shift variance,lack of directionality associated with the wavlet and not well matched with the singularities like lines ,edges. Hence complex wavelet tranform can be used to overcome the above mentioned disadvantages. In this transform, the original signal is passed through 2 real DWT filter-bank trees. The resulting coefficients are complex. The output values of the first tree are considered for real part and the output values of the second tree are considered for imaginary part. The implementation of complex wavelet

https://dx.doi.org/10.22161/ijaers/3.9 *ISSN: 2349-6495(P) | 2456-1908(O)*

transform structure with logarithmic arithmetic units will be implemented in the next part of the work.

V. EXPERIMENTAL RESULTS

The lifting based DWT architecture was designed with logarithmic multiplication computations and integrated with BPS coders.These modules were designed in Verilog. Xilinx ISE 9.2 tool was used for the synthesis of the designs and simulated using Model sim, a simulator from Model Technology (Mentor Graphics Company). Designs were synthesized in Xilinx VIRTEX 7 FPGA devices. The designed architecture was compared with multi precision floating point arithmetic designs and logarithmic arithmetic units designs and the results are presented here. The parameters are compared between the proposed architecure and the multi precision floating point designs as shown in the figure5.

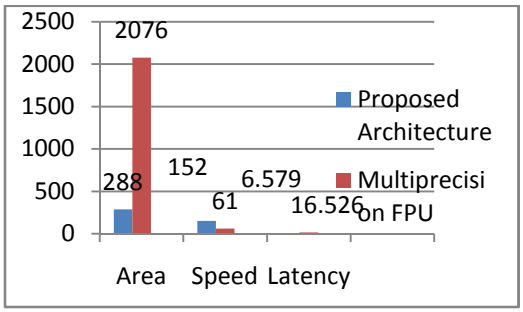

Fig.5: Parameters Comparison Chart

TABLE I: COMPARISON BETWEEN PROPOSED ARCHITECTURE AND MULTI-PRECISION FPU ARCHITECTURE

Device VIRTEX 7 FPGA	**Proposed DWT Architecture With Log FPU and BPS Coder**	**Multi-Precision FPU**
Area (slices LUTs)	288 out of 192800	2076 out of 192800
Clock Speed (MHZ)	152.001 MHz	60.509MHz
Latency(ns)	6.579ns	16.526ns

From the experimental results shown in TABLE I , it was concluded that the proposed lifting based DWT architecture integrated with logarithmic floating point units and BPS coders occupied less area and increased the speed of computations compared to the implementation of multi-precision floating point designs on FPGA deivces.

TABLE II: COMPARISON BETWEEN PROPOSED ARCHITECTURE AND LOG FPU

Device VIRTEX 7 FPGA	**Proposed DWT Architecture With Log FPU and BPS Coder**	**Logarithmic Multiplication Based-FPU**
Area (slices LUTs)	288 out of 192800	2890 out of 192800
Clock Speed (MHZ)	152.001 MHz	97.066MHz
Latency(ns)	6.579ns	10.302ns

TABLE II also proved that, the proposed lifting based DWT architecture integrated with logarithmic floating point units and BPS coders occupied less area and increased the speed of computations compared to the implementation of only logarithmic floating point multiplication units on FPGA devices. Hence the proposed work is proved to be advantageous for the implelementation on devices.

The image parameter analysis such as peak to signal noise ratio and resolution of the images will be produced in our future work. As power consumption is one of the major entitiy in VLSI designs, optimisation of power for the same architecture will be considered in further part of the work.

VI. CONCLUSION AND FUTURE WORK

In this work, the implementation of DWT and IDWT structures on FPGA devices provided efficient area and faster speed of computations. These structures were designed based on logarithmic floating point computations and hence provided efficient area compared to conventional floating point arithmetic units. From the experimental results it was concluded that, the block based pass-parallel SPIHT algorithm along with log based FPU DWT structures provided high throughput, less latency and efficient area.

In our future work, the designs will be extended with the implementation of multi mode embedded logarithmic floating point units along with BPS SPIHT coders to provide better resolution of images, improved accuracy and less power consumption. Also the complex wavelet

Transform structure will be integrated with logarithmic floating point units and BPS coders for image fusions.

REFERENCES

[1] G. Govindu, R. Scrofano, and V. K. Prasanna, “A library of parameterizable floating-point cores for FPGAs and their application to scientific computing,” in *Proc. Int. Conf. Eng. Reconfigurable Syst. Algorithms*, 2005, pp. 137–148.

[2] R. Scrofano, L. Zhuo and V.K. Prasanna, “Area-Efficient Arithmetic Expression Evaluation Using

Deeply Pipelined Floating-Point Cores," IEEE Transactions on VLSI systems,vol.16,no.2,pp.167-176,2008

[3] Zhaolin Li, Xinyue Zhang, GongqiongLi, Runde Zhou, "Design of A Fully Pipelined single-precision Floating-Point Unit", IEEE,2007,pg 60-63.

[4] IEEE Standard for Floating-Point Arithmetic, 2008.

[5] CW Yu, AM Smith, W Luk, PHW Leong, SJE Wilton, Optimizing floating point units in Hybrid FPGAs. IEEE Trans. Very Large Scale Integer (VLSI) Syst. 20(7), 45–65 (2012)

[6] YJ Chong, SParameswaran, Configurable multimode embedded floating point units for FPGAs. IEEE Trans. Very Large Scale Integer (VLSI) Syst.19 (11), 2033–2044 (2011)

[7] TH Anand, D Vaithiyanathan, R Seshasayanan, "Optimized architecture for floating point computation unit", in Int. conf. on emerging trends in VLSI, embedded sys, nano elec. and tele. sys (Thiruvannamalai, India, 2013), pp. 1–5

[8] S Paul, N Jayakumar, SP Khatri, A fast hardware approach for approximate, efficient logarithm and antilogarithm computations. IEEE Trans. Very Large Scale Integration (VLSI) Syst. 17(2), 269–277 (2009)

[9] Haohuan Fu, Member, IEEE, Oskar Mencer, Member, IEEE, and Wayne Luk, Fellow, IEEE, "FPGA Designs with Optimized Logarithmic Arithmetic", IEEE Transactions On Computers, VOL. 59, NO. 7, JULY 2010.

[10] *Nikolaos Alachiotis, Alexandros Stamatakis*, "Efficient Floating-Point Logarithm Unit for Fpgas" in IEEE, 2010.

[11] Ramin Tajallipout,Mf.Asraful Islam and Khan A.Wahid, "Fast Algorithm of a 64 bit Decimal Logarithmic Converter", Journal of Computers,vol.5,No.12,December 2010.

[12] Mark G.Arnold and Sylvain Collange, "A Real/Complex Logarithmic Number System ALU", IEEE Transactions on Computers, Vol.60, No.2, February 2011.

[13] M. Haselman, M. Beauchamp, K. Underwood, and K. Hemmert, "A Comparison of Floating Point and Logarithmic Number Systems for FPGAs," Proc. IEEE Int'l Symp. Field-Programmable Custom Computing Machines (FCCM), pp. 181-190, 2005.

[14] [14]J. Coleman, E. Chester, C. Softley, and J. Kadlec, "Arithmetic on the European Logarithmic Microprocessor," IEEE Trans. Computers, vol. 49,no. 7, pp. 702-715, July 2000.

[15] B. Lee and N. Burgess, "A Parallel Look-Up Logarithmic Number System Addition/Subtraction Scheme for FPGA," Proc. Int'l Conf. Field-Programmable Technology (FPT), pp. 76-83, 2003.

[16] Mark G. Arnold, Member, IEEE, and Sylvain Collange, "A Real/Complex Logarithmic Number System ALU", IEEE Transactions On Computers, Vol. 60, No. 2, February 2011

[17] M.G. Arnold and S. Collange, "A Dual-Purpose Real/Complex Logarithmic Number System ALU," Proc. 19th IEEE Symp. Computer Arithmetic, pp. 15-24, June 2009.

[18] Xuguang Lan, Nanning Zheng, *Senior Member, IEEE*, Yuehu Liu, "Low-Power and High-Speed VLSI Architecture for Lifting-Based Forward and Inverse Wavelet Transform", IEEE, 2005.

[19] T Acharya, C Chakrabarti, A survey on lifting-based discrete wavelet transform architectures. J. VLSI Signal Process. 42, 321–339 (2006)

[20] S Barua, JE Carletta, KA Kotteri, AE Bell, An efficient architecture for lifting-based two-dimensional discrete wavelet transforms. Integr. VLSI J.38(3), 341–352 (2005)

[21] K Andra, C Chakrabarti, T Acharya, A VLSI architecture for lifting-based forward and inverse wavelet transform. IEEE Trans. Signal Process 50(4), 966–977 (2002)

[22] G Shi, W Liu, L Zhang, F Li, An efficient folded architecture for lifting-based discrete wavelet transform. IEEE Trans. Circuits Syst.-II 56(4), 290–294 (2009)

[23] CT Huang, PC Tseng, LG Chen, Flipping structure: an efficient VLSI architecture of lifting based discrete wavelet transform. IEEE Trans. Signal Process. (2004)

[24] W Zhang, Z Jiang, Z Gao, Y Liu, An efficient VLSI architecture for lifting-based discrete wavelet transform. IEEE Trans. Circuits Syst.–II 59(3), 158–162 (2012)

[25] C Cheng, KK Parhi, High-speed VLSI implement of 2-D discrete wavelet transforms. IEEE Trans. Signal Process. 56(1), 393–403 (2008)

[26] X Tian, L Wu, YH Tan, JW Tian, Efficient multi-input/multi-output VLSI architecture for two dimensional lifting-based discrete wavelet transform. IEEE Trans. Comput. 60(8), 1207–1211 (2011)

[27] T Fry, S Hauck, SPIHT image compression on FPGAs. IEEE Trans. Circuits Syst. Video Technol. 15(9), 1138–1147 (2005)

[28] Y Jin, HJ Lee, A Block-Based, Pass-parallel SPIHT algorithm. IEEE Trans. Circuits Syst. Video Technol. 22(7), 1064–1075 (2012)

[29] J. Muller, Elementary Functions: Algorithms and Implementation. Springer, 2006.

[30] Virtex-7 Family Overview, Xilinx, Inc., http://www.xilinx.com, 2007.