International Journal of
Science and Engineering Investigations vol. 2, issue 22, November 2013
ISSN: 2251-8843

VLSI Implementation of the Discrete Wavelet Transform (DWT) for Image Compression

Aarti S. Shinde[1], A. K. Pathrikar[2]
[1]PG Student Department of E&TC, Savitribai Phule Women's Engineering College Aurangabad
[2]Assistant Professor, Department of Electronics, Savitribai Phule Women's Engineering College Aurangabad
([1]shinde.aarti99@gmail.com, [2]anand_pathrikar@rediffmail.com)

Abstract- This paper presents new VLSI architectures for finite impulse response (FIR) filters and discrete wavelet transform, intended for very high-speed signal and image processing. The design follows the JPEG2000 standard and can be used for both lossy and lossless compression. In order to reduce complexities of the design, linear algebra view of DWT has been used in this paper. This design can be used for image compression in a robotic system. In this project, Discrete Wavelet Transform filter bank is presented. The filter bank implemented here is Daubechies 9/7-tap bi orthogonal filter bank. The proposed structure can increase the work frequency (85%) at a low cost of additional hardware elements (55%). The systems are verified, using JPEG2000 coefficients filters, on Xilinx Field Programmable Gate Array (FPGA) devices.

Keywords- *Discrete Wavelet Transform (DWT), FPGA, Fast Convolution, Finite Impulse Response (FIR), Filter, VLSI*

I. INTRODUCTION

Image compression plays a critical role in telemetric applications. It is desired that either single mages or sequences of images be transmitted over computer networks at large distances so as that they could be used for a multitude of purposes. For instance, it is necessary that medical images be transmitted so as that reliable, improved and fast medical diagnosis performed by many centers could be facilitated. To this end, image compression is an important research issue. The difficulty, however, in several applications lies on the fact that, while high compression rates are desired, the applicability of the reconstructed images depends on whether some significant characteristics of the original images are preserved after the compression process has been finished. For instance, in medical image compression applications, diagnosis is effective only when compression techniques preserve all the relevant and important image information needed. This is the case with lossless compression techniques. Lossy compression techniques, on the other hand, are more efficient in terms of storage and transmission needs but there is no warranty that they can preserve the characteristics needed in medical image processing and diagnosis. One important such characteristic of such images is texture. More specifically, texture analysis of these images can lead to very significant results concerning real tissue motion and thus, can result in improved diagnosis. More generally, however, it is well known that significant visual effects for any kind of image rely on their textural characteristics. The main motivation for the research effort presented in this paper is, precisely, how these important visual effects could be kept without being severely distorted during image compression process. To this end, their efficient coding is necessary. It is well known that such textural characteristics cannot be captured from images using first order gray scale statistics.

II. BASIC ARCHITECTURES

An FIR filter is defined by its impulse response coefficients, h(n), with $h(n) = 0$ for $n < 0$ and $n > M$, where M is the filter order. The filter output y(n) is given by the convolution of h(n) with the input sequence x(n): $y(n) = h(n) * x(n)$ (3) Expanding the convolution gives the difference equation of the so-called direct form [5]:

Where h(k) is the impulse response of the filter, and the filter has M +1 coefficients. The direct form is shown in Figure 2. If TA, TM and Tat are the delays ofthe adder, multiplier and adder tree, respectively, then the critical path of the direct form filter is M+Tat. The graph reversal theorem can be exploited to obtain the so-called transposed form [5], which is shown in Figure 3. The transposed form has the significant advantage that its critical path is greatly reduced to TM+TA, since its adders have only two operands and are followed by a register. In designing frequency selective digital filters, it is usually desirable to have approximately constant frequency-response magnitude and minimum phase distortion [6].

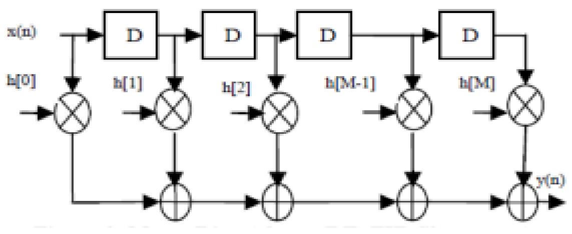


Figure 1. M-tap direct form DF filter structure

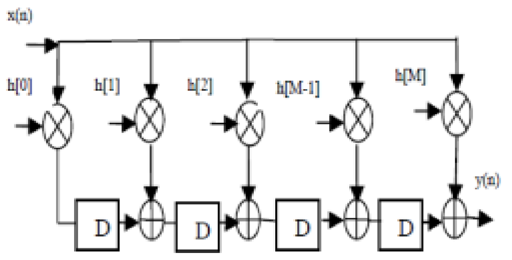


Figure 2. M-tap Transpose form(TF) FIR Filter structure

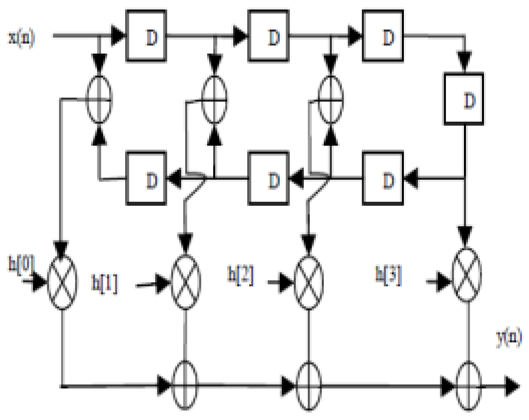


Figure 3. 8-tap Linear Phase (LP) FIR Filter structure

A linear phase with integer slope is replaced by a simple delay in the time domain and it reduces the phase distortion to a minimum in the frequency domain. Therefore, it is required to design digital filters with exactly or approximately linear phase. Figure 4 shows the linear phase FIR filter. A linear phase FIR filter of order N is characterized by symmetric impulse response [6] h[n] = h[N-n], (3) For 8-tap FIR filter, direct and transpose form required 8 multipliers and 7 adders, whereas linear phase requires 4 multipliers and 7 adders.

Pipelined Architectures

Pipelining reduces the effective critical path by introducing pipelining latches along the data path. Pipelining transformation leads to a reduction in the critical path, which can increase the clock speed (or sample speed). K.Parhi [5] presented pipelined FIR filter by using feed-forward cutset method. The critical path is now reduced from TM + 2TA to TM+TA, where TM is the time taken for multiplication and TA is the time needed for an addition operation. The pipelining latches can only be placed across any feed forward cut set of the graph. Ramsey S. Hourani [7] used the FIR architecture in low power equalizer, in which the feed forward cutset is used in the multiplier data path, by placing a register at the output of each multiplier. This decouples the multiplication and addition delays, and it allows a more equal comparison with the direct form. The timing of each multiplier may differ therefore, for giving the data to the adder at the same time, D latches are used, as a result, the speed of the circuit is improved. Direct form (DF) architectures are suitable for area sensitive small filter orders while transposed structures are suitable for large filter orders.

However, according to [9] replacing the multi operand adder in the direct form with the pipelined binary adder tree will achieve a very high speed FIR filter. This arrangement is most efficient when the number of filter coefficients is an integer power of two. Otherwise, some terms must be delayed prior to addition. M. Maamoun [10] presented very high speed FIR filter architecture, which is based on combining pipelining and parallel arithmetic methods. In this architecture, efficiently distributed D-latches and multipliers are used, thus this critical path is reduced to Tm/2 (if TA < TM/2).

We propose a new high speed linear phase FIR filter in which pipelining and parallel arithmetic methods are used [10].Thus, using equal amount of multipliers high speed architecture can be achieved compared to DF. This fast FIR filter is a combination of alternate multiplication and binary adder tree as shown in Figure 5. In the alternate multiplication for a given clock, the input signal is divided into even and odd signal, Dm1 is triggered for odd part and Dm2 is triggered for even part. To increase the processing frequency, Dm1 and Dm2 D-latches work at the same frequency (Fm1 = Fm2) and the input and output D latches work with the same phase and frequency (Fd).Where,

Fd = 2 * Fm1

TABLE I. COMPARISON OF 8-TAP FIR STRUCTURES

Architectures	Mul.(s)	Add.(s)	Reg.(s)	Delay (ns)
DF FIR	8	7	9	30.904
Linear Phase FIR	4	7	9	21.120
FIR [9]	8	7	23	11.034
Fast FIR [10]	16	7	59	5.045
Proposed FIR	8	7	23	5.045

Table I depict the requirement of multipliers, adders, registers and the critical path delay for the individual filter structure. The proposed design needs equal amount of multipliers and adders in comparison to direct form but requires an approx twice number of more registers.

We observe that the DF FIR filter has highest delay. In FIR filter [9], the multiple products are rearranged with binary tree of adders. This decouples the multiplication and addition delays, by this delay is reduced by 64% Fast FIR filter [10] and proposed architecture uses pipelining and parallelism thus the delay is reduced by 83% as compared to DF.

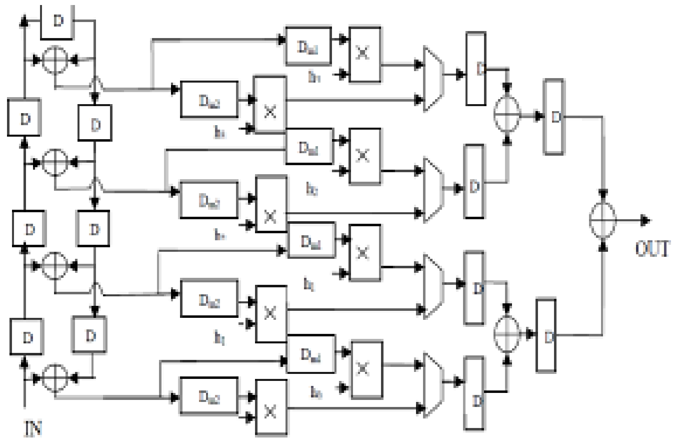


Figure 4. The proposed architecture

Principles behind Compression

Number of bits required to represent the information in an image can be minimized by removing the redundancy present in it. There are three types of redundancies:

i. Spatial redundancy, which is due to the correlation or dependence between neighboring pixel values;

ii. Spectral redundancy, which is due to the correlation between different color planes or spectral bands;

iii. Temporal redundancy, which is present because of correlation between different frames in images.

A. *Types of compression*

Lossless versus Lossy compression:- Lossless versus Lossy compression: In lossless compression schemes, the reconstructed image, after compression, is numerically identical to the original image. However lossless compression can only a achieve a modest amount of compression. Lossless compression is preferred for archival purposes and often medical imaging, technical drawings, clip art or comics.

Predictive versus Transform coding- In predictive coding, information already sent or available is used to predict future values, and the difference is coded. Since this is done in the image or spatial domain, it is relatively simple to implement and is readily adapted to local image characteristics.

B. *Image Compression Model*

Basic model for image compression is shown in figure 1.1 and 1.2. The different component & their function as follows.

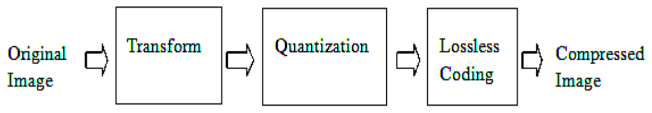


Figure 5. Image Compression Model

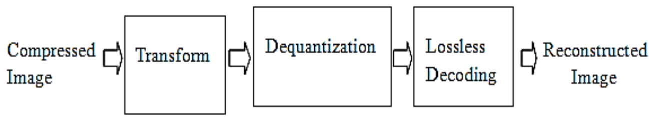


Figure 6. Image Decompression Model

Transformer- It transforms the input data into a format to reduce inter pixel redundancies in the input image. Transform coding techniques use a reversible, linear mathematical transform to map the pixel values onto a set of coefficients, which are then quantized and encoded. The key factor behind the success of transform-based coding schemes is that many of the resulting coefficients for most natural images have small magnitudes and can be quantized without causing significant distortion in the decoded image.

Transform coding algorithms usually start by partitioning the original image into sub images (blocks) of small size (usually 8×8).

Quantizer- It reduces the accuracy of the transformer's output in accordance with some pre-established fidelity criterion reduces the psycho visual redundancies of the input image. This operation is not reversible and must be omitted if lossless compression is desired. The quantization stage is at the core of any lossy image encoding algorithm. Quantization at the encoder side, means partitioning of the input data range into a smaller set of values. There are two main types of quantizers: scalar quantizers and vector quantizers.

Vector quantization (VQ) techniques extend the basic principles of scalar quantization to multiple dimensions.

Symbol (entropy) encoder- Most important types of entropy encoders used in lossy image compression techniques are arithmetic encoder, Huffman encoder and run-length encoder.

It creates a fixed or variable-length code to represent the quantizer's output and maps the output in accordance with the code.

III. SYSTEM IMPLEMENTATION

A. *Data Compression*

Compression is the process of reducing the size of the data sent, thereby, reducing the bandwidth required for the digital representation of a signal. Many inexpensive video and audio applications are made possible by the compression of signals. Compression technology can result in reduced transmission time due to less data being transmitted. It also decreases the storage requirements because there is less data. However, signal quality, implementation complexity, and the introduction of communication delay are potential negative factors that should be considered when choosing compression technology.

Signals can be compressed because of the spatial, spectral, and temporal correlation inherent in these signals. Spatial correlation is the correlation between neighboring samples in an image frame. Temporal refers to correlation between samples in different frames but in the same pixel position. Spectral correlation is the correlation between samples of the same source from multiple sensors.

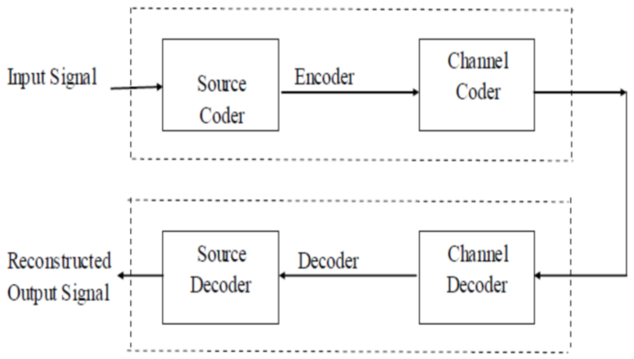


Figure 7. Block Diagram of a Compression/Decompression System

The Discrete Wavelet Transform (DWT) has become one of the most used techniques for image compression and is applied in a large category of applications. DWT can provide significant compression ratios without great loss of visual quality than the previous techniques such as the Discrete Cosine Transform (DCT) and the Discrete Fourier Transform (DFT). The DWT present the main part of the JPEG2000 standard, which permits both lossy and lossless compression of digital images. It allows an encoded image to be reconstructed progressively. The compression phase is mainly divided into three sequential steps: (1) Discrete Wavelet Transform, (2) Quantization, and (3) Entropy Encoding. After preprocessing, each component is independently analyzed by an appropriate discrete wavelet transform. The computational block diagram of the functionalities of the compression system is shown in fig.8.

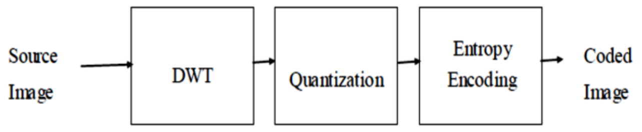


Figure 8. DWT Based Compression System

Conventionally, programmable DSP chips are used to implement such algorithms for low-rate applications and the VLSI application specific integrated circuits (ASICs) for higher rates. The FPGAs are programmable logic devices that provide sufficient quantities of logic resources that can be adapted to support a large parallel distributed architecture. Lifting and convolution present the two computing approaches to achieve the discrete wavelet transform. While conventional lifting-based architectures require fewer arithmetic operations compared to the convolution-based approach for DWT, they sometimes have long critical paths. If Ta and Tm are the delays of the adder and multiplier, respectively, then the critical path of the lifting-based architecture for the (9,7) filter is (4×Tm + 8×Ta), while that of the convolution implementation is (Tm + 2×Ta). In addition to this and for the reason to preserve proper precision, intermediate variables widths are larger in lifting based computing. As a result, the lifting multiplier and adder delays are longer than the convolution ones. Many VLSI lifting-based DWT architectures have been developed and implemented to reduce the memory requirements and the critical path.

B. Compression Steps

1. Digitize the source image into a signal s, which is a string of numbers.
2. Decompose the signal into a sequence of wavelet coefficients w.
3. Use threshold to modify the wavelet coefficients from w to w'.
4. Use quantization to convert w' to a sequence q.
5. Entropy encoding is applied to convert q into a sequence e.

C. VLSI Architectures for the DWT

So far the theory and an efficient algorithm to perform a DWT has been described. Several different VLSI implementations are proposed in the literature. The main concern of a designer about a VLSI chip realization is the required total area A. This parameter is mainly effected by the number of multiply/accumulate (MAC) cells, memory cells (registers) and the space needed for signal routing. A design is also aimed to use the available hardware resources efficiently (grade of hardware utilization). A further criterion is input/output (IO) rate of the implementation. The highest possible clock rate is usually affected by the critical path (CP). The CP could be shortened using pipeline techniques, but these may increase the latency of the implementation, which might not be tolerable for some applications. Some additional requirements for the DWT implementations should be considered. The architectures should be able to perform the forward and the backward transform for the selected decomposition level on the same hardware. The architecture should also be easily expandable to compute a two dimensional WT. It might also be important to reconfigure the realization for the use of different filter coefficients and filter lengths.

D. A systolic 1D-DWT implementation

The first published architectures have been designed and optimized for a specific wavelet and decomposition level. Therefore, they are neither easily scalable nor applicable to other filters. The architecture uses a two dimensional, regular routing network, which can be easily extended to a certain decomposition level or filter length. First, the pyramid algorithm is slightly modified in order to map it on a systolic array. This is also needed to perform the operation online with a minimum amount of memory. The so called recursive pyramid algorithm (RPA) schedules each output at the earliest possible instance, where the ith level decomposition precedes the (i+1)th level one. This also ensures that all data needed for the next level decomposition is available at the right time. The following table illustrates this output scheduling of the wavelet coefficients for an input length of N = 16 and J = 4 decomposition levels:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	16
1	x		x		x		x		x		x		x		x
2		x				x				x				x	
3				x								x			
4								x							

The first octave for the online input data is computed every other cycle and the following octaves are scheduled between this. The RPA could further be modified to suit filter realizations with latency. The memory needed for the RPA realization is now L .J, where L is the length of the FIR filter. All computations are performed using the same filter unit. The standard pyramid algorithm would need either J computation units or N cells of storage and one computation unit. The systolic array architecture consists of a two filtering units, one to computes the high pass and the other one the low pass coefficients. Both are linear systolic arrays with L MAC units. The low pass coefficients are fed into the memory cells that hold the last results for the filtering units. The memory cells could be implemented as systolic or semi systolic routing network or a conventional RAM. The structure is illustrated in figure. Now, the RPA is executed on this structure as described next. In the odd cycles, the filter units receive their input directly and compute the first octave outputs. The high pass output is transmitted to the output ports while the low pass output is fed back to the routing network. The subsequent octaves are computed during the even cycles. Therefore the filter units take their input from the routing network, where the last low pass coefficients are stored. The routing networks consist of three different cell types. The first column on the left side (Rl) receives the low pass filter outputs and stores them in the appropriate memory cells S. These cells hold the values and pass them to their neighbors (up and to the right) when they receive a clock event. These clock events are produced by the rightmost column (Rr). As the coefficients are transmitted through the routing network, they have to reach the filter at the right time according the given schedule. There, it is recommended to replace the vertical local interconnects by vertical busses. This simplifies the routing cells S and the inter cell signaling without an increase in the design complexity. The circuit takes one input every second cycle and produces one output for every input. This leads to a hardware utilization of the MAC units of 50%. This utilization can be improved to 100% if only one filtering unit is used and modified to compute both, the high- and the low pass coefficients. The described architecture has many advantages. It is nearly optimal in both area and time. 100% utilization of the filtering unit can be achieved. The implementation uses only L & J memory cells, independent from the number of inputs N. Further it scales easily with the filter length and the number of computed octaves, simply by adding columns or rows to the routing network and the linear array. The straight forward implementation of a separable 2D-DWT would perform a filtering operation for every row, followed by a column wise filtering. The intermediate results would need to be stored, using N & N registers. This is only practical for offline algorithms, where the data is already stored in memory. Further, this implementation would have a high latency. The systolic parallel architecture proposed is capable of performing the 2D-DWT with a latency of only one cycle. It also uses a minimum amount of memory to store the intermediate results between the row and column filtering. Basically, the architecture consists of two 1D-DWT blocks (S1, S2) as described above, a memory array and two parallel filters (P1, P2), see figure 6. The systolic filtering blocks receive the input data directly and the low pass coefficients from the parallel filters are fed to the routing network of S1. The output from this row filtering operation is routed to the holding cells. This memory is used to store the intermediate results and to prepare the data stream for the parallel filters, which need the data input in column wise order. Now, the wavelet coefficients for one octave.

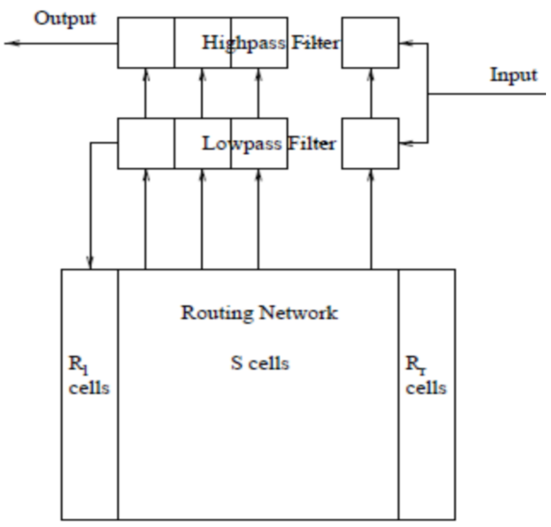


Figure 9. 1D DWT Architecture Using Systolic Arrays

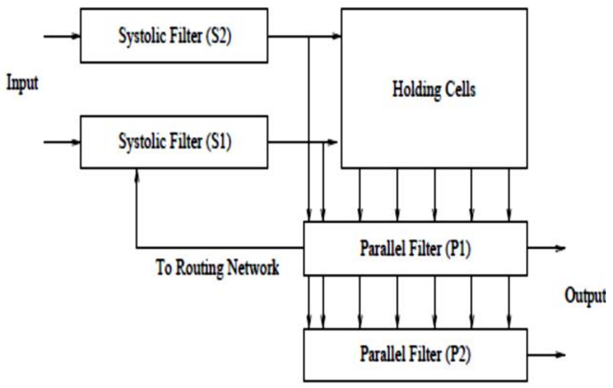
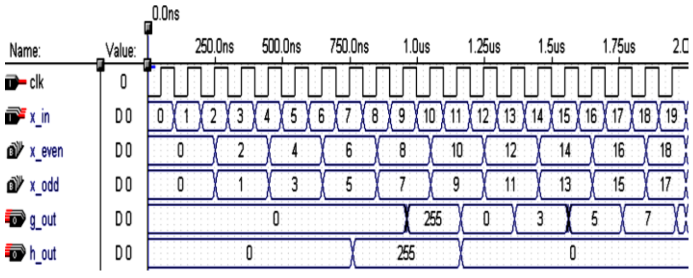


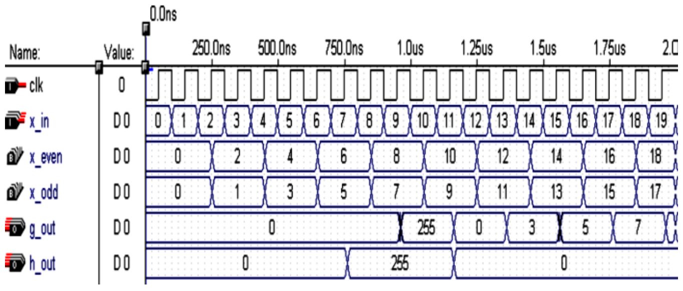
Figure 10. 2D DWT Systolic Parallel Architecture

E. *Performance Analysis*

The simulation waveforms generated by the MAX+plus II simulator verify the correct functioning of the design. Figure 4.5(a) shows the simulation results for polyphase implementation, figure 4.5(b) shows the simulation results for polyphase with modified DA implementation. In all cases, 8-bit signed input samples were used. Also, the output signals were scaled to have the same number of bits as the input.



(a)



(b)

Figure 11. (a) and (b) Simulation results of the db9/7 Filter Bank

Statistical Analysis- The hardware requirements, in terms of the number of LCs utilized by the architectures, is provided in fig.12, where B1 corresponds to polyphase implementation and B2 corresponds to polyphase implementation.

TABLE II. STATISTICAL COMPARISON BETWEEN DCT

Type of encoders using ASIC	No. of Logic gates of the ASIC used	Amount of on chip RAM used by the encoders	Data processing rates of the encoders using ASIC
DCT	34000	128 byte	210 MSa/sec
DWT	55000	55 kbyte	150 MSa/sec

The results show that B1 consumes 809 logic cells, which corresponds to 46% of the total available logic cells in FLEX 10K30A device. B2 consumes 626 logic cells which is 36% of the total LCs. Thus, for an area efficient design, B2, which is the polyphase implementation with modified DA, is most suitable.

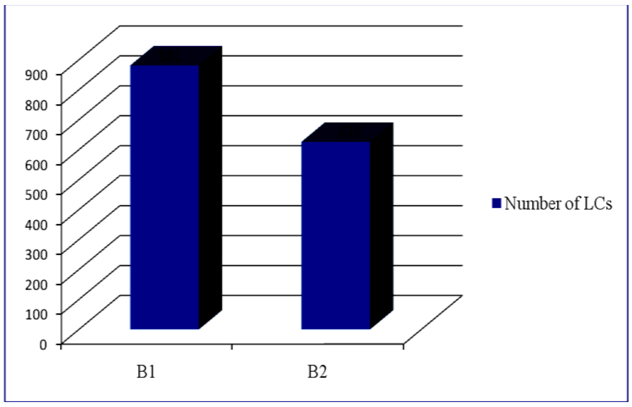


Figure 12. Comparison of Performance for the db9/7 Filter Banks

The performance of the architectures in terms of the maximum frequency using timing analyzer is shown in fig.13 as observed, B2 has slightly lower performance than B1.

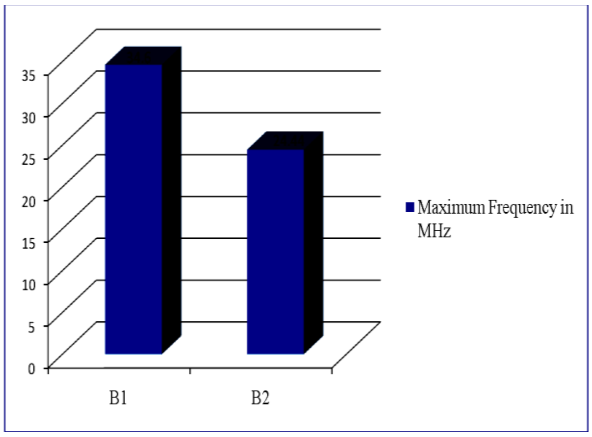


Figure 13. Comparison of Power Consumption for the db9/7 Filter Banks

It is observed that B2 consumes 23.91mW while B1 consumes 33.7mW. Therefore, B2 is more power efficient than B1. By overall comparison, it is seen that B2, i.e., the polyphase with modified DA architecture, is the better of the two, as it consumes much less area and power with a slightly lower performance than B1.

IV. CONCLUSION

In this project an improved and efficient Discrete Wavelet Transform algorithm has been developed and tested in MATLAB. Then the algorithm has been modified for implementation in VHDL. From the satisfactory simulation results we can conclude that algorithm works properly. This VHDL code was also synthesized to achieve the gate level architecture of the design which is now ready to be implemented in hardware. I can conclude that wavelet analysis is very powerful and extremely useful for compressing data such as image. Although other transforms have been used, for example the DCT was used for the JPEG format to compress images, wavelet analysis can be seen to be far superior, in that it doesn't create 'blocking artifacts'. This is because the wavelet analysis is done on the entire image rather than sections at a time. A well-known application of wavelet analysis is the compression of fingerprint images by the FBI. Changing the decomposition level changes the amount of detail in the decomposition. Thus, at higher decomposition levels, higher compression rates can be gained. However, more energy of the signal is vulnerable to loss. The wavelet divides the energy of an image into an approximation sub signal, and detail sub signals. Wavelets that can compact the majority of energy into the approximation sub signal provide the best compression. This is because a large number of coefficients contained within detailed sub signals can be safely set to zero, thus compressing the image. However, little energy should be lost. Wavelets attempt to approximate how an image is changing, thus the best wavelet to use for an image would be one that approximates the image well. However, although this report discusses some relevant image properties, there was not time to research or investigate how to find the best wavelet to use for a particular image.

REFERENCES

[1] Mallat, S.: A Theory for Multi resolution Signal Decomposition: The Wavelet Representation. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 11, No. 7. (1989) 674-693.

[2] Gnavi, S., Penna, B., Grangetto, M., Magli, E., Olmo, G.: Wavelet kernels on a DSP: A comparison between lifting and filter banks for image coding. Applied Signal Processing: Special Issue on Implementation of DSP and Communication Systems. Vol. 2002. No. 9. (2002) 981-989.

[3] Daubechies, I., Sweldens, W.: Factoring wavelet transforms into lifting schemes. The Journal of Fourier Analysis and Applications. vol. 4. (1998) 247-269.

[4] Acharya, T., Tsai, P. S.: JPEG2000 Standard for Image Compression. A John Wiley & Sons, Inc. USA (2005)

[5] Andra, K., Chakrabarti, C, Acharya, T.: A VLSI Architecture for Lifting-Based Forward and Inverse Wavelet Transform. IEEE Transactions on Signal Processing, vol. 50. No. 4. (2002) 966-977 (Pubitemid 34459791)

[6] Andra, K., Acharya, T., Chakrabarti, C.: A High-Performance JPEG2000 Architecture. IEEE Transactions on Circuits and Systems for Video Technology, vol. 13. No. 3. (2003) 209-218

[7] B.-F. Wu and C.-F. Lin, "An efficient architecture for JPEG2000 coprocessor", IEEE Trans. Consum. Electron., vol. 50, no. 4, pp. 1183-1189, Nov. 2004 (Pubitemid 40072109)

[8] K. K. Parhi and T. Nishitani, "VLSI architectures for discrete wavelet transforms," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 1, no. 6, pp. 191–202, Jun. 1993.

[9] C. Chakrabarti, M. Vishwanath, and R. M. Owens, "Architectures for wavelet transforms: A survey," J. VLSI Signal Process., vol. 14, pp. 171–192, 1996.

[10] H. Olkkonen, J. T. Olkkonen, and P. Pesola, "Effcient lifting wavelet transform for micorprocessor and VLSI applications," IEEE Signal Process. Lett., vol. 12, pp. 120–122, 2005.

[11] P. K. Campbell, K. E. Jones, R. J. Huber, K. W. Horch, and R. A. Normann, "A silicon-based, three-dimensional neural interface: Manufacturing processes for an intracortical electrode array," IEEE Trans. Biomed. Eng., vol. 38, no. 8, pp. 758–768, Aug. 1991.

[12] K. D. Wise, D. J. Anderson, J. F. Hetke, D. R. Kipke, and K. Najafi, "Wireless implantable micro systems: High-density electronic interfaces to the nervous system," Proc. IEEE, vol. 92, no. 1, pp. 76–97, Jan. 2004.

[13] D. M. Taylor, S. I. Tillery, and A. B. Schwartz, "Direct control of 3-D neuroprosthetic devices," Science, vol. 296, pp. 1829–1832, 2002.

[14] J.Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan, and M. A. L. Nicolelis, "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates," Nature, vol. 408, pp. 361–365, 2000.

[15] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue, "Instant neural control of a movement signal," Nature, vol. 416, pp. 141–142, 2002.

[16] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "Flipping structure: An efficient VLSI architecture for lifitng based discrete wavelet transform, IEEE Trans. Signal Process., vol. 52, no. 4, pp. 1080–1089, Apr. 2004.

[17] M. Unser and T. Blu, "Wavelet theory demystified," IEEE Trans. Signal Process., vol. 51, no. 2, pp. 470–483, Feb. 2003.

[18] C.-T. Huang, P.-C. Tseng, and L.-G. Chen, "VLSI architecture for forward discrete wavelet transform based on B-spline factorization," J. VLSI Signal Process., vol. 40, pp. 343–353, 2005.