FPGA Implementation of 2D-DWT and SPIHT Architecture for Lossless Medical Image Compression

T.Vijayakumar[1], S.Ramachandran[2]

Abstract—This paper presents an analysis of wavelet filters and SPIHT encoding techniques adopted in compression and decompression of medical images. The hardware implementation for loss less compression with tradeoff in area, timing and power, requires quantizing of filter coefficients and identification of its impact on reconstruction process. This study proposes a novel algorithm that replaces multipliers with shift operators exploiting the symmetric property of filter coefficients. The filter coefficients are scaled to integers and represented using 9-bit signed representation to ensure that the quantization error is less than 20dB. The HDL code is developed in Verilog for the modified filter architecture in 1D and 2D DWT processor is implemented on Virtex-5 FPGA. The modified algorithm occupies less than 12% slices, operates at frequency of 298 MHz consuming less than 1W of power. The results of the study demonstrate that the hardware design matches with the software model and hence does not result in any loss therefore suitable for loss less medical image compression.

Key words— Medical image compression, DWT, Multiplierless logic, quantization effects, lossless compression

1. Introduction

Image compression leads to compressing the input data to more than 100% compression, thus 25Mbits of information can be represented using few kilo bits which would requires few milliseconds to transmit over internet. Compression may lead to loss of relevant data and hence imposes restrictions to medical image compression. Hence there need to be a trade-off between compression losses, bandwidth/storage space[1]. JPEG 2000 recommends use of Discrete Wavelet Transform (DWT) for image transformation from time domain to spatial domain or frequency domain and SPIHT [2][3][4] encoding for encoding of spatial domain frequency bands. SPIHT can be easily used for either fixed bit rate or variable bit rate applications and it is also very suitable for progressive transmission [5][6][7]. Furthermore, SPIHT has about 0.6 dB peak signal-to-noise-ratios (PSNR) gain over EZW[7] and is very close to EBCOT in many circumstances[6]. SPIHT algorithm uses simpler coding procedure and needs no coding table [8][9]. The implementation of SPIHT would be much cheaper to be suitable for still image compression appliances [10][11]. In [12] DWT with SPIHT have been adopted for medical image compression. Biorthogonal and DB wavelets have found to produce higher PSNR in the range of 30db to 40db compared with Haar wavelets for most of the image considered [12]. For lossless compressions bpp of 1 to 4 achieves PSNR between 45dB to 52 dB, thus suitable for telemedicine applications. The results are based on software simulations. However, for telemedicine applications, it is required to realize the algorithm on hardware. Several attempts have been made for implementing DWT and SPIHT on FPGA. In order to achieve lossless compression, it is required to design DWT and SPIHT algorithm to processes data and reproduces results as that of software models. SPIHT is an encoding scheme that requires glue logic, DWT is a data path operator and hence introduces losses when implemented on hardware.

DWT architecture has been implemented using convolution or FIR filter bank. In architecture the filter coefficients stored in memory are multiplied with the input samples using arithmetic blocks which occupy large number of transistors. There are several architectures discussed in literature to perform lifting based DWT. General approach for 2-D DWT is to apply the 1-D DWT row-wise which produces L and H subbands and then process these sub-bands column-wise to get LL, LH, HL and HH coefficients. Several architectures like direct mapped [13], folded [14], and flipping [15] for single level and multi-level DWT have been proposed to implement 1-D and 2D lifting DWT. In this paper we propose modified systolic array architecture for computation of 1D, 2D DWT and also a modified architecture for SPIHT encoder for lossless image compression.

Section II discusses DWT and SPIHT encoding algorithm, Section III discusses quantization effects in DWT architecture and its impact on lossless compression. Section IV discusses design of modified DWT architecture and SPIHT, Section V discusses implementation results and conclusion is presented in Section VI.

2. Background Theory

The need for image compression becomes apparent when the number of bits per image resulting from typical sampling and quantization schemes is computed. Considering the amount of storage for the 'Lena' digital image, the monochrome (grayscale) version of this image with a resolution 512x512x8 bits/pixel requires a total of 2,097,152 bits or 786,432 bytes [3]. Such image should be

compressed for efficient storage or transmission. Compression is of two types: Lossless compression and Lossy compression. **Lossless compression** is one where the information can be recovered after decompressing such that no loss of data occurs. **Lossy compression** is one where the data compressed cannot be recovered. There are different forms of lossy compression of which we are interested in transform encoding based lossy compression. Transform encoding involves mathematical transformation separating image information based on gradual spatial variation of brightness from information with faster variation of brightness at edges of the image. In the next step, the information on slower changes is transmitted lossless, but information on faster local changes is transmitted with lower accuracy by quantizing the data. This results in loss of data which cannot be recovered.

2.1 Wavelet based Image Compression

Wavelet coding has become very popular due to its robustness under transmission and decoding errors at higher compression rates avoiding blocking artifacts. Wavelet based compression [16] [17] [18] [19] is based on sub-band coding. Sub band coding involves splitting the frequency band of the image into sub bands and then to code each sub band using a coder and bit rate accurately matched to the statistics of the band. Simple DWT consists of a low pass filter and high pass filter which splits the image into low frequency and high frequency sub bands. Figure 1 shows the two dimensional decomposition of image where the first one is row decomposition and the second stage is column decomposition.

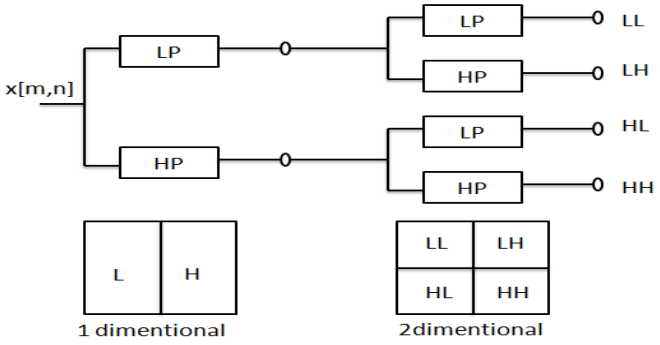


Fig. 1. 2-Dimensional DWT [16]

Figure 2 shows the decomposed image where the low frequencies are at the top left corner of the image and purely higher frequency at bottom right corner.

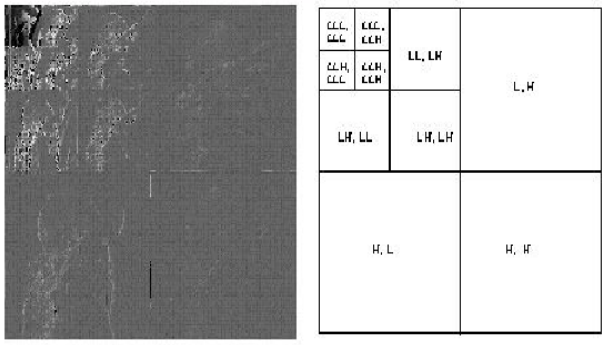


Fig.2. Representation of decomposed image [16]

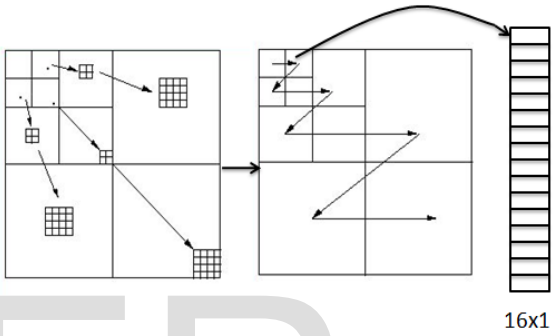


Fig.3. Framing the decomposed blocks [17]

Figure 3 shows the reorganization if the decomposed sub bands prior to SPIHT encoding. The input image which is 2D is reorganized to 1D, the reorganized data is encoded using SPIHT encoder algorithm.

2.2 SPIHT Encoder

SPIHT is an embedded coding technique. In embedded coding algorithms, encoding of the same signal at lower bit rate is embedded at the beginning of the bit stream for the target bit rate. Effectively, bits are ordered in importance. This type of coding is especially useful for progressive transmission using an embedded code, where an encoder can terminate the encoding process at any point. SPIHT algorithm is based on following concepts [5]:

1. Ordered bit plane progressive transmission.
2. Set partitioning sorting algorithm.
3. Spatial orientation trees.

SPIHT makes use of three lists – the List of Significant Pixels (LSP), List of Insignificant Pixels (LIP) and List of Insignificant Sets (LIS). These are coefficient location lists that contain their coordinates. After the initialization, the algorithm takes two stages for each level of threshold – the sorting pass (in which lists are organized) and the refinement pass (which does the actual progressive

coding transmission). The result is in the form of a bit stream.

SPIHT keeps three lists: LIP, LSP, and LIS. LIP stores insignificant pixels, LSP stores significant pixels and LIS stores insignificant sets. At the beginning, LSP is empty, LIP keeps all coefficients in the lowest sub band and LIS keeps all tree roots which are at the lowest sub band. SPIHT starts coding by running two passes. The first pass is the sorting pass. It first browses the LIP and moves all significant coefficients to LSP and outputs its sign. Then it browses LIS executing the significance information and following the partitioning sorting algorithms.

The second pass is the refining pass. It browses the coefficients in LSP and outputs a single bit alone based on the current threshold. After the two passes are finished, the threshold is divided by 2 and the encoder executes the two passes again. This procedure is recursively applied until the number of output bits reaches the desired number. The input image of size NxN is decomposed to three levels using three 2D DWT. Each 2D DWT consist of two 1D DWT that is performed along row and columns of input matrix. First level decomposition give rise to LL, LH, HL and HH sub band of size N/2xN/2. The LL sub band is decomposed to four more sub bands of N/4xN/4 size and in the third level LLL sub band is decomposed into N/8xN/8 sub band. Thus after three level decomposition there exist four sub bands of size N/8xN/8, three sub bands of size N/4xN/4 and three sub bands of N/2xN/2. SPIHT encoder algorithm encodes the ten sub bands into bit stream based on the compression ratio expressed in terms of bits per pixel (bpp). Input image is represented using 8 bit per pixel, the compressed image can be represented using bpp less than 8, thus leading to compression. The brain image has been compressed and decompressed for bpp from 0.1 to 4 using bior4.4, Haar and bd2. Medical images captured from sensors produce images of varying aspect ratio, hence complicates the compression procedure. The captured medical images have been preprocessed to uniform aspect ratio without affecting the information content by padding with zeros. The preprocessed image is compressed and decompressed using DWT and SPIHT. Biorthogonal and DB wavelets have found to produce higher PSNR in the range of 30db to 40db compared with Haar wavelets for most of the image considered. For lossless compressions bpp of 1 to 4 achieves PSNR between 45dB to 52 dB, thus suitable for telemedicine applications.

2.3 Wavelet Filters and Hardware Implementation

Wavelet transform is represented by equation (1), the mother function or analyzing wavelet $\Phi(x)$ form the basis function and is given as:

$$\phi_{(sl)}(x)=2^{(-s/2)}\phi(2^{-s}x-1) \quad (1)$$

In the Equation (1) variable 's' and 'l' are integers that scale and dilate the mother function 'Φ' to generate wavelets. The scale index 's' indicates the wavelet's width and the location index 'l' gives its position. The mother functions are rescaled or dilated by powers of two and translated by integers. The DWT is given by Equation (2) [20-22].

$$W(n,j)=\sum_{m=0}^{2n} W(\mathrm{m,j-1})\, w(\mathrm{2n-m})$$

$$W_H(n,j)=\sum_{m=0}^{2n} W(\mathrm{m,j-1})\, h(\mathrm{2n-m}) \quad (2)$$

Where W (n, j) = z(n) and w(n) and h(n) are Quadrature Mirror Filters derived from the wavelet. To span our data domain at different resolutions, the analyzing wavelet is used in a scaling shown in Equation (3).

$$W(x)=\sum_{k=-1}^{N-2}(-1)^k c_{k+1}\,\phi(2x+k) \quad (3)$$

Where, W(x) in the Equation (3) is the scaling function for the mother function Φ and c_k are the wavelet coefficients. There are several filters that satisfy the properties of wavelets. Biorthogonal wavelet supports the major properties of wavelets. In this work, bior4.4 is selected for lossless image compression. Table1 shows the filter coefficients for Biorthogonal 4.4 which has 9 filter coefficients and 7 filter coefficients for low pass and high pass respectively. The filter coefficients in Table 1 have 15 decimal digits and require 32 bits for representation. For computation of wavelet filter output coefficients 9 multiplications and 7 multiplications are required for every output sample. Thus every multiplication will increase the bit width by N+1. Complexity of the hardware increases if 32 bit representation is used. In order to reduce the hardware complexity, quantization is performed on the filter coefficients to reduce the number of bits for representation. The quantized filter coefficients are shown in Table 2.

Table 1 Biorthogonal filter coefficients for decomposition and reconstruction

No.	LO_D	LO_R	HI_D	HI_R
1	0.037828455507264	-0.0645388826286971	-0.0645388826286971	-0.037828455507264
2	-0.023849465019557	-0.0406894176091641	0.040689417609164	-0.023849465019557

3	-0.110624404418437	0.418092273221	0.418092273221617	0.110624404418437
4	0.377402855612831	0.788485616405583	-0.788485616405583	0.377402855612831
5	0.852698679008894	0.418092273221617	0.418092273221617	-0.852698679008894
6	0.377402855612831	-0.040689417609164	0.0406894176091641	0.377402855612831
7	-0.110624404418437	-0.0645388826286971	-0.0645388826286971	0.110624404418437
8	-0.023849465019557	0	0	-0.023849465019557
9	0.037828455507264	0	0	-0.037828455507264

Table 2. Quantized filter coefficients

The quantization Q of a real-world value V is represented by a weighted sum of bits. Within the context of the general slope and bias encoding scheme, the value of an unsigned fixed-point quantity is given by (4) [24]

$$\bar{V} = S.\left[\sum_{i=0}^{ws-1} b_i 2^i\right] + B, \quad (4)$$

While the value of a signed fixed-point quantity is given by (5) [24]

$$\bar{V} = S.\left[-b_{ws-1}2^{ws-1} + \sum_{i=0}^{ws-2} b_i 2^i\right] + B, \quad (5)$$

Where,

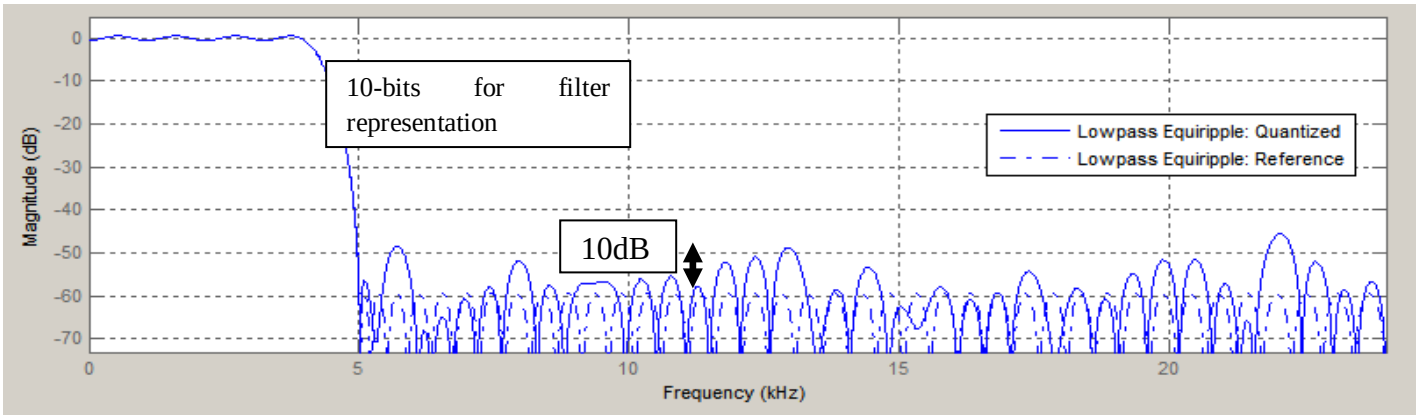
- b_i are binary digits, with $b_i = 1,0$, for $i = 0,1,\ldots,ws-1$.
- The word size in bits is given by ws, with $ws = 1, 2, 3,\ldots, 128$.
- S is given by $F2^E$, where the scaling is unrestricted because the binary point does not have to be contiguous with the word.

b_i are called *bit multipliers* and 2^i are called the *weights*.

The wavelet filters shown in Table 1 are quantized using 9 bit binary numbers based on the equations and represented in Table 2.

No.	LO_D	LO_R	HI_D	HI_R
1	10	-17	-17	-10
2	-6	-10	10	-6
3	-28	107	107	28
4	97	202	-202	97
5	218	107	107	-218
6	97	-10	10	97
7	-28	-17	-17	28
8	-6	0	0	-6
9	10	0	0	-10

Wavelet filters are represented using 24-bit binary numbers, after quantization the numbers of bits are 16 bits, thus there is no significant change in the output or there is no loss in information retrieval. Figure 4 shows the effects of quantization if the wavelet filters coefficients are represented using less than 16 bits.



12-bits for filter representation

5dB

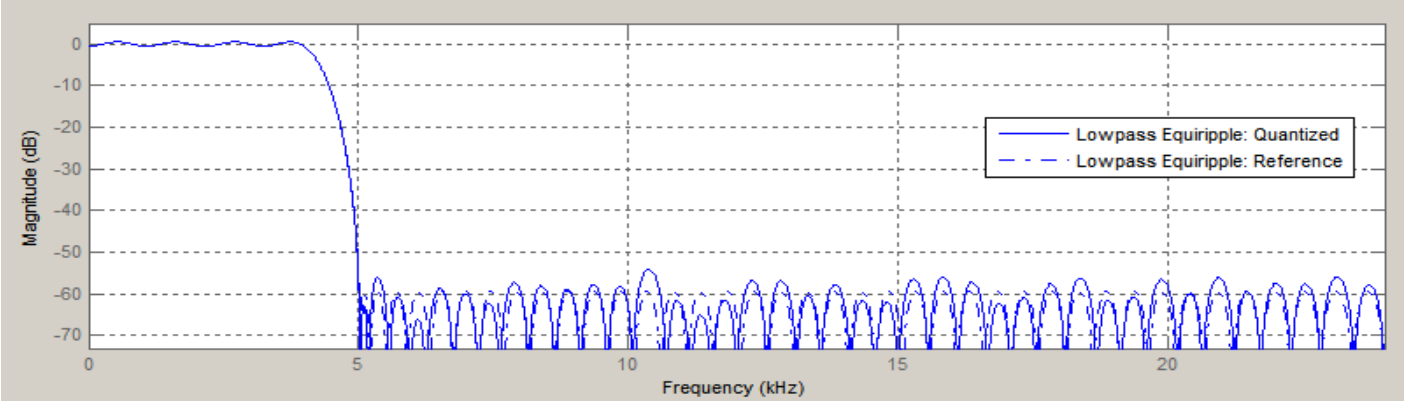


Fig.4. Effects of quantization

The filter coefficients represented with less than 12 bits, the pass band frequency gain does not change by more than 1 dB. The stop band attenuation varies from 40dB to 5dB and causes harmonics to be generated during filtering process. Thus for loss less compression, it is recommended that the filter coefficients should be represented by minimum of 9 bits to avoid loss during image reconstruction. The analysis reveals that, the quantized information can be retrieved if the coefficients are represented using minimum of 9 bits. The next section discusses the modified architecture for DWT implementation.

3. Modified DWT architecture

As shown in Figure 1, DWT computation based on convolution algorithm is implemented using multiplication and accumulation unit as shown in Figure 5.

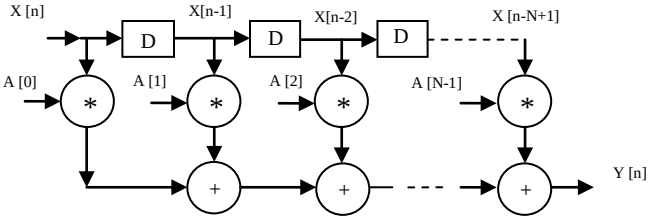


Fig. 5.Convolution architecture for DWT

In the conventional architecture for DWT based on convolution algorithm, the input samples are sequentially processed by multiplying the input samples with the filter coefficients A[N] and accumulating the samples to compute Y[n]. The DWT filters consists of 9 filter coefficients and 7 filter coefficients for low pass and high pass respectively. For every output sample computation due to the sequential approach in the MAC operation, the latency is 9 clock cycles and 7 clock cycles for low pass and high pass computation. The throughput without pipeline is 9 clock cycles and 7 clock cycles respectively. For every clock cycle multiplication and addition need to be performed and hence the clock frequency of multiplier and adder should be 10 times higher (10 bit multiplier and adder) than the clock frequency of input data. In order to reduce the computation complexity and improve throughput a modified algorithm is proposed in this work.

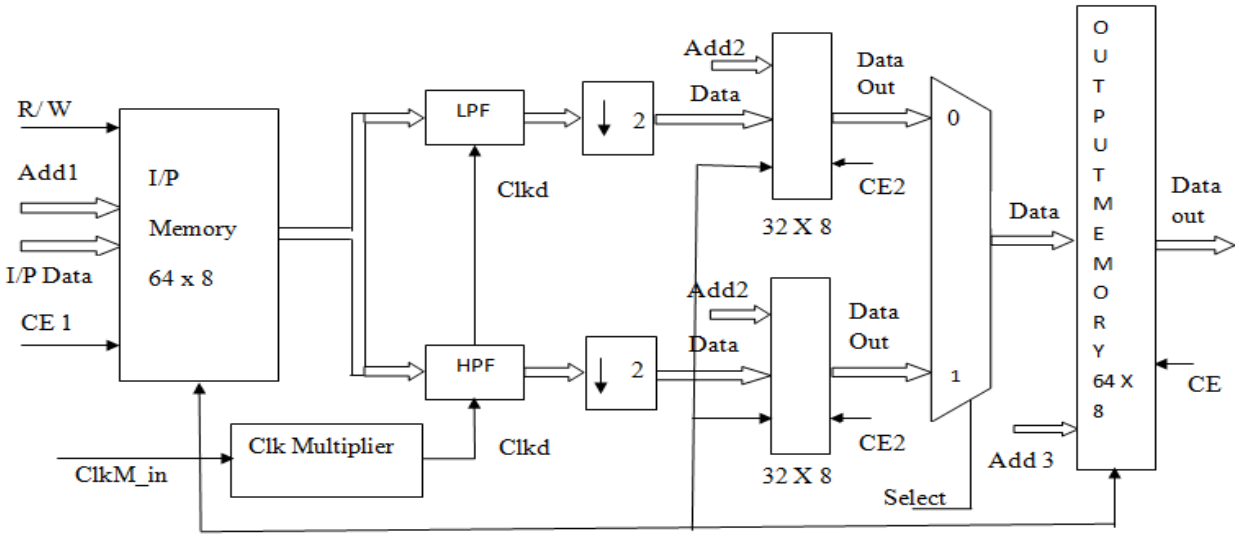


Fig.6.Top level architecture for 1D-DWT computation

Figure 6 shows the Top level architecture of proposed 1D-DWT processor. 1-D DWT processor architecture consists of an input memory, low pass and high pass filter with down sampling unit, intermediate output register and output memory. The master clock clkM_in clocks the memory circuits, the clock multiplier generates clock for arithmetic unit. The input samples consisting of 8 x 8 pixels are stored row wise in 64 x 8 memory, the intermediate memory stores the low pass and high pass filter data after processing in 32 x 8 memory size that consists of 8 rows with each 4 columns. The multiplexer unit reorders the intermediate samples into the output memory and rearranges the data for computation of 1D-DWT column wise for 2D-DWT computation.

Figure 7 shows data flow of input and computation of output sample using LPF and HPF filters using convolution algorithm. Every clock cycle a new input data enters into the processing unit and is multiplied by the low pass filter coefficients L. As the input samples do not exist for first 8 clock cycles the processing unit output is neglected until 8 clock cycles, the first out samples is Y_{L8}, hence the latency is 9 clock cycles

L_0	L_1	L_2	L_3	L_4	L_5	L_6	L_7	L_8	--	--	--	--	--
X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9				X_{14}
X_{-1}	X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8				X_{13}
X_{-2}		X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7				X_{12}
X_{-3}			X_0	X_1	X_2	X_3	X_4	X_5	X_6				X_{11}
X_{-4}				X_0	X_1	X_2	X_3	X_4	X_5				X_{10}
X_{-5}					X_0	X_1	X_2	X_3	X_4				X_9
X_{-6}						X_0	X_1	X_2	X_3				X_8
X_{-7}							X_0	X_1	X_2				X_7
X_{-8}								X_0	X_1	X_2			X_6
Y_{L0}	Y_{L1}	Y_{L2}	Y_{L3}	Y_{L4}	Y_{L5}	Y_{L6}	Y_{L7}	Y_{L8}	Y_{L9}	Y_{L10}	--	--	Y_{L15}

H_0	H_1	H_2	H_3	H_4	H_5	H_6	--	--	--	--
X_0	X_1	X_2	X_3	X_4	X_5	X_6				X_{13}
X_{-1}	X_0	X_1	X_2	X_3	X_4	X_5				X_{12}
X_{-2}		X_0	X_1	X_2	X_3	X_4				X_{11}
X_{-3}			X_0	X_1	X_2	X_3				X_{10}
X_{-4}				X_0	X_1	X_2				X_9
X_{-5}					X_0	X_1				X_8
X_{-6}						X_0	X_1	X_2		X_7
Y_{H0}	Y_{H1}	Y_{H2}	Y_{H3}	Y_{H4}	Y_{H5}	Y_{H6}	Y_{H7}	Y_{H8}		Y_{H13}

Fig. 7. LPF and HPF Computation

Row 1 Row 2 Row 8

$X = [X_0\ X_1\ X_2$ ---- $X_7,\ X_8\ X_9$ ---- $X_{15},\ X_{16}$ ---- ------$X_{56},\ldots\ldots X_{63}];$

Fig. 8. Input data sets for filtering

3.1 Modified Architecture for 1D DWT Computation

The LPF output is $Y_{L0} = X_0 L_0 + X_{-1} L_1 + X_{-2} L_2 + X_{-3} L_3 + ---- X_{-8} L_8$ and is $Y_{L1} = X_1 L_0 + X_0 L_1 + X_{-1} L_2 + ---- X_{-7} L_8$ similarly $Y_{L8} = X_0 L_0 + X_1 L_1 + X_2 L_2 + X_3 L_3 + X_4 L_4 + X_5 L_5 + X_6 L_6 + X_7 L_7 + X_8 L_8$. $L_0 = L_8$, $L_1 = L_7$, $L_2 = L_6$, $L_3 = L_5$ hence the output can be rewritten as $Y_{L8} = (X_0 + X_8)L_8 + (X_1 + X_7)L_7 + (X_2 + X_6)L_6 + X_3 + X_5)L_5 + X_4 L_4$ which can be represented as $Y_{L8} = X^1_0 L_8 + X^1_1 L_7 + X^1_2 L_6 + X^1_3 L_5 + X^1_4 L_4$ where $X^1_0=(X_0 + X_8)$. Based on the modified equation for Y_{L8} the modified architecture is derived as shown in Figure 9. The architecture consists of input memory, rearranged input memory, adder, intermediate register and processing unit for multiplication and accumulation, output memory.

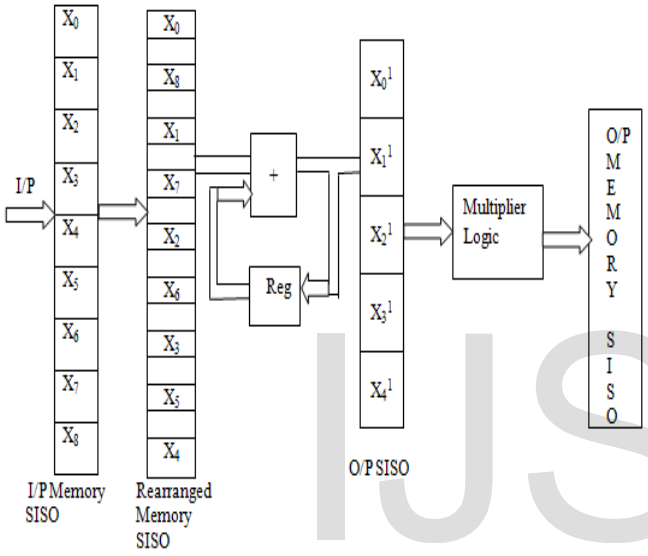


Fig.9. Modified architecture for DWT [LPF]

The input data [X_0 to X_8] is rearranged as in rearranged memory, the data is read and accumulated in the adder and stored in the O/P SISO. The samples [X^1_0 to X^1_4] are multiplied to obtain the LPF O/P. After substituting filter coefficients, the LPF equation can be written as

$$Y_{L8}=X^1_0[10]+X^1_1[-6]+X^1_2[-28]+X^1_3[97]+X^1_4[218] \quad (6)$$

Which can be rewritten as in (7),

$$Y_{L8} = X^1_0 [8 + 2] + X^1_1 [-8 + 2] + X^1_2 [-32 + 4] + X^1_3 [64 + 32] + X^1_4 [256 - 32] \quad (7)$$

The filter coefficients 10, -6, -28, 97, 218 are expressed in multiples of 2, which is expanded as in equation (8).

$$Y_{L8} = X^1_0 8 + X^1_0 2 - X^1_1 8 + X^1_1 2 - X^1_2 32 + X^1_2 4 + X^1_3 64 + X^1_3 32 + X^1_4 256 - X^1_4 32 \quad (8)$$

The input samples are to be multiplied by the modified filter coefficients, which can be realized by performing left shift operation as indicated in equation (9).

$$Y_{L8} = [X^1_0 << 3] + [X^1_0 << 1] - [X^1_1 << 3] + [X^1_1 << 1] + [X^1_3 << 5] + [X^1_3 << 6] + [X^1_4 << 8] - [X^1_4 << 5] + [X^1_2 << 2] - [X^1_2 << 5] \quad (9)$$

The filter coefficients 97 and 218 are rounded of the 96 and 224 to achieve left shift in multiples of 2. Based on the modified equation (9), the LPF architecture is realized and Figure10 shows the architecture of proposed logic for Y_{L8} computation.

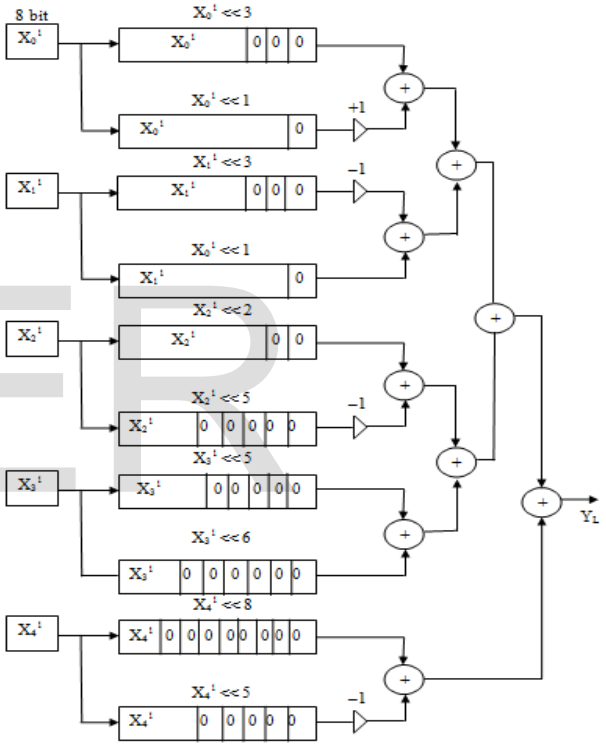


Fig.10. Multiplierless DWT LPF architecture

The top level architecture for 2D DWT processor is implemented using the modified 1D-DWT architecture discussed in Figure 10. The 2D-DWT processor consists of input memory, output memory and three 1D-DWT processors as shown in Figure 11.

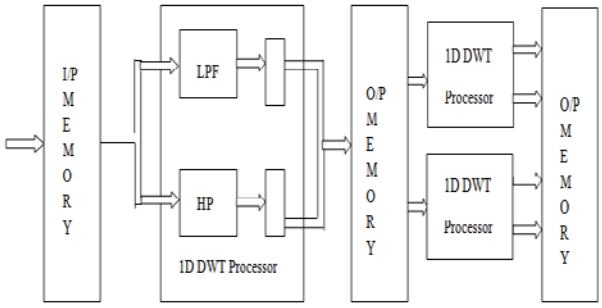


Fig.11. 2D DWT architecture

HDL code for 1D DWT processor, input memory and output memory is developed and are integrated to top module. The top module is verified using test bench written in Verilog and with know set of input vectors. The simulation results and synthesis results are obtained using Xilinx ISE.

4. Results and Discussion

The input x_i is of 8 bits, clock and reset are the other inputs. The outputs obtained were a_i and d_i which were of 16 bits signed outputs with the consideration of scaling. The negative values of outputs were obtained in 2's complement form. In one clock cycle, the corresponding values of a_i and d_i were obtained for each value of i. Verilog code for DWT was simulated using ModelSim for i from 0 to 63 and the corresponding values of a_i and d_i were obtained. Figure 14 shows the results for some values of i with input x_0=0, x_1=1. For values of x_i, for i greater than 63 no output is obtained.

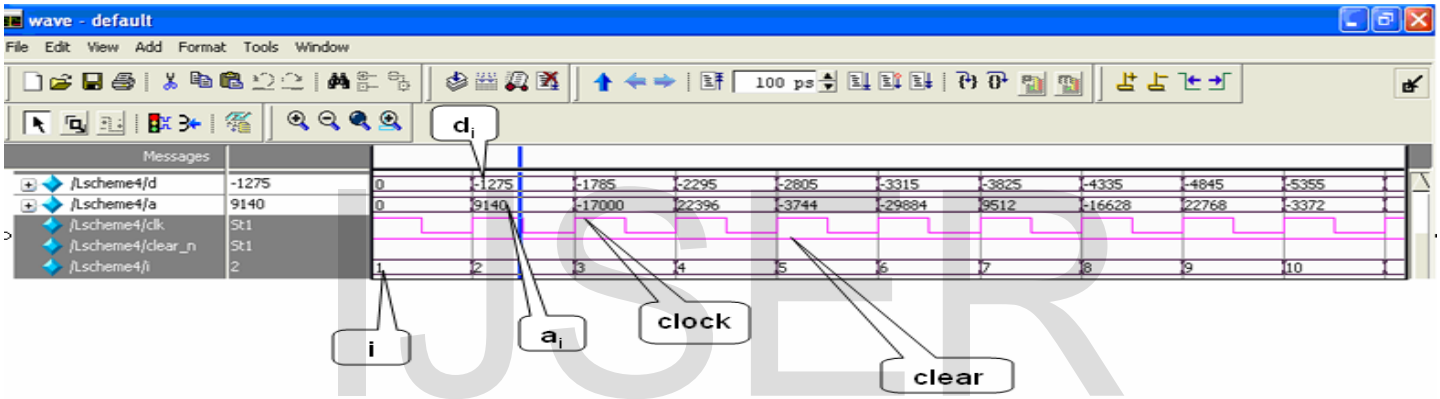


Fig.12. Simulation results of DWT

The theoretical results and practical results are compared for the proposed 1D-DWT processor. Table 4 shows the comparison results.

Table 4 Theoretical and practical values of a_i and d_i

Practical			Theoretical		
i	a_i	d_i	i	a_i	d_i
2	9140	-1275	2	9141	-1278
5	-3744	-2805	5	-3734	-2811
10	-3372	-5355	10	-3369	-5351
40	-1140	-20655	40	-1145	-20651

Since the minimum value of x is for x_{2i-4}, when i is less than 2 both a_i and d_i are 0. Since the outputs require all the input values, the inputs are first stored and then the evaluation of output begins. During each clock cycle a_i and d_i are obtained that is when i=1, a_1, d_1 are obtained. In the next cycle a_2, d_2 are obtained and so on. The calculated theoretical and practical values were found to match. The Table 4 shows the comparison of theoretical and practical values obtained. For i=2, substituting in the equations for a_i and d_i along with the values of x, it was found that a_i = 9140 and d_i = -1275, there is a maximum variation of =+/- 8. This is due to the rounding of error in the modified architecture design. The code was synthesized using Xilinx with the Virtex-5 device having gate capacity 110 million gates and the RTL schematic obtained is as shown in the Figure 13. Figure 14 shows the 2D-DWT architecture using the proposed architecture.

Fig.13. RTL schematic of DWT

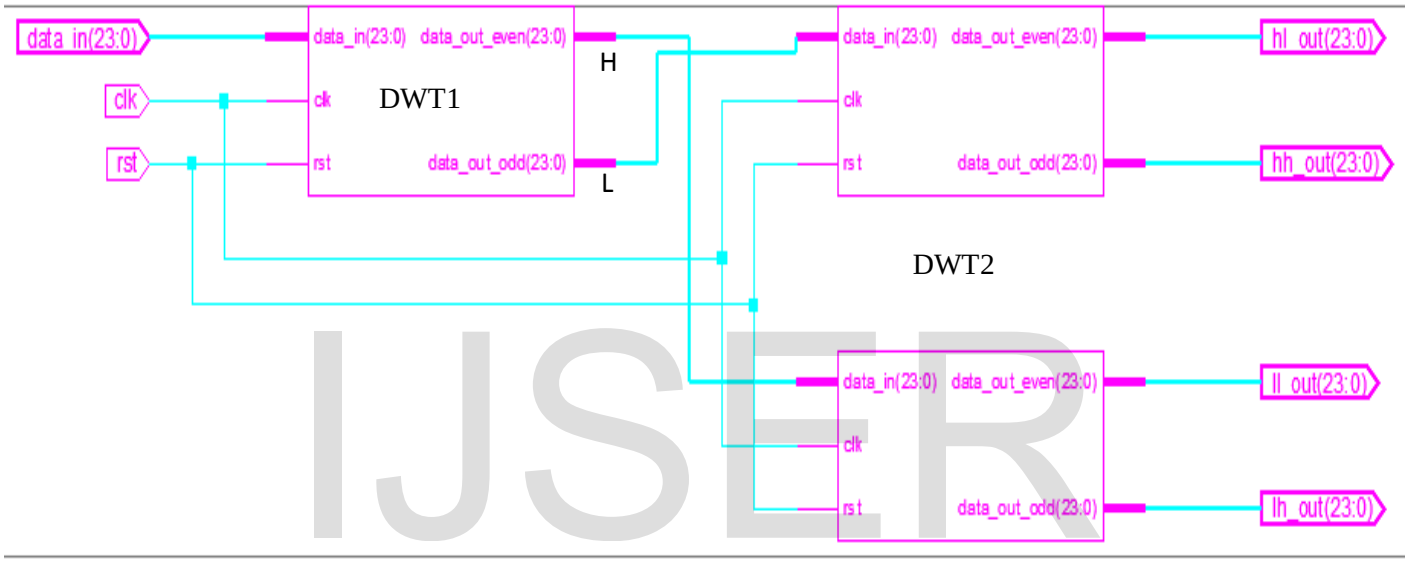


Fig.14. 2D- DWT architecture synthesis schematic

The synthesis results obtained are verified with various constraints options provided in the tool. The default options were producing best results. The area report in terms of slices, the power report and timing report have been generated and are reported in this work. Conventional DWT architecture was realized in [24] on Spartan device hence the results reported have been used for comparison. In order to compare the performance improvements in the proposed architecture, the conventional DWT architecture is modeled using HDL and implemented on Virtex-5 device. The results obtained are reported in Table 5.

Table5. Comparison of 2-D DWT architecture

Parameters	Conventional DWT [24] (on Spartan)	Conventional DWT	Proposed Design
No of Slices	566 out of 768	31105 out of 69120 45%	8295 out of 69120 12%
No of gates	37K	31105 out of 69120 45%	8295 out of 69120 45%
Clock Speed	36MHZ	237 MHz	298 MHz
Power dissipation	51mW	1.37 W	1.007 W

From the comparison results it is demonstrated that the proposed architecture consumes very less resources, as the multipliers are replaced with shift operations. The operating frequency is increased to 298 MHz and power dissipation is reduced by setting the low power constraints. One of the major challenges in the design is data

synchronization in DWT computing, as the shift operations are used for multiplication operation, it is mandatory to carefully design the control unit to keep track of the data output and read the data into register for further computation and hence there is need for a predesigned control logic to monitor the data flow logic.

5. Conclusion

DWT architecture based on 9/7 filter coefficients has been recommended for image compression. In this work, the quantization effects of filter coefficients for hardware implementation have been evaluated. From the simulation, it is proved that, minimum of 7-10 bits are required to represent the filter coefficients, so that the quantization noise does not impact the gain by more than 20 dB. Based on the results obtained, the DWT filters are quantized and represented using signed number representation for hardware realization. The DWT processor is realized using a modified algorithm that reduces the number of computation by half by exploiting the symmetric property. The filter coefficients obtained after quantization are rounded off to their nearest power of 2 and is realized using shift left operations. The modified algorithm is realized using HDL, 1D-DWT processor and 2D-DWT processor is implemented on FPGA. The results demonstrate improvement in area and speed performances without loss and hence are suitable for lossless image compression.

References:

1. Kevin Hung and Yuan-Ting Zhang, "Implementation of a WAP-Based Telemedicine System for Patient Monitoring", IEEE transactions on information technology in biomedicine, vol. 7, no. 2, pp. 101-107, June 2003.
2. G. Wallace, "The JPEG still picture compression standard," *Communications of ACM*, vol. 34, no. 4, 1991.
3. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600-612, Apr., 2004.
4. J.M. Shapiro, "Embedded image coding using Zero trees of wavelet coefficients," *IEEE Trans. on Signal Processing*, vol.41, pp. 3445-3462, Dec., 1993.
5. A.Said and W.A.Perlman, "A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees," *IEEE Trans. on Circuits and Systems for Video Technology*, vol.6, pp. 243-250, June, 1996.
6. D. Taubman, "High Performance Scalable Image Compression with EBCOT," *IEEE Trans. on Image Processing*, vol. 9, pp. 1158-1170, July, 2000.
7. B.-J.Kim and W. A. Pearlman, "An embedded video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)," *in proc. IEEE Data Compression Conf.*, pp. 251–260, Mar., 1997.
8. J. Karlenkar and U. B. Desai, "SPIHT video coder," *in Proc. IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control*, *TENCON'98*, vol. 1, pp. 45–48, 1998.
9. B.-J. Kim, Z. Xiong, and W. A. Pearlman, "Low bit-rate scalable video coding with 3-d set partitioning in hierarchical trees (3-D SPIHT)," *IEEE Trans. Circ. and Syst. for Video Technology*, vol. 10, no. 8, pp. 1374–1387, Dec. 2000.
10. J. Zho and S. Lawson, "Improvements of the SPIHT for image coding by wavelet transform," *Proc. IEEE Seminar on Time-scale and Time-Frequency Analysis and Applications (Ref. No. 2000/019)*, pp. 24/1 –24/5, 2000.
11. E. Khan and M. Ghanbari, "Very low bit rate video coding using virtual spiht," *IEEE Electronics Letters*, vol. 37, no. 1, pp. 40–42, Jan. 2001.
12. Wei Zhang, Zhe Jiang, Zhiyu Gao, and Yanyan Liu, "An Efficient VLSI Architecture for Lifting-Based Discrete Wavelet Transform" ieee transactions on circuits and systems—ii: express briefs, vol. 59, no. 3, march 2012.
13. Anirban Das, Anindya Hazra, and Swapna Banerjee (2010), "An Efficient Architecture for 3-D Discrete Wavelet Transform," IEEE transactions on circuits and systems for video technology, vol. 20, no. 2.
14. C.-Y. Xiong, J.-W. Tian, and J. Liu (2006), "A note on 'flipping structure: An efficient VLSI architecture for lifting based discrete wavelet transform','IEEE Trans. Signal Process., vol. 54, no. 5, pp. 1910–1916.
15. H. Cai and B. Zeng, "A new SPIHT algorithm based on variable sorting thresholds," *in Proc. IEEE Int. Symp. Circuits and Systems*, vol. 5, pp. 231–234, May 2001.
16. U. Bayazit, "Region adaptive spectral transformation for wavelet based color image compression," in Proc. IEEE Int. Conf. Image Process., Nov. 2009, pp. 2817–2820.
17. Kakadu JPEG2000 Codec [Online]. Available: http://www. kakadusoftware.com
18. Talukder, K.H. and Harada, K., A Scheme of Wavelet Based Compression of 2D Image, Proc. IMECS, Hong Kong, pp. 531-536, June 2006
19. Kassim and W. S. Lee, "Embedded color image coding using SPIHT With partially linked spatial orientation trees," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 2, pp. 203–206, Feb. 2003.

20. S G Mallat, 'A Theory for Multiresolution Signal Decomposition: The Wavelet Representation', IEEE Trans. Pattern Reco. and Machine Int., 11(7), PP: 674-693, 1989.
21. P P Vidyanathan, 'Multirate Systems and Filter Banks', Prentice Hall, 1993.
22. O Rioul, and M Vetterli, 'Wavelets and Signal Processing', IEEE Sig. Proc. Mag., PP: 14-38, 1991.
23. Strang, G., T. Nguyen (1996), Wavelets and filter banks, Wellesley-Cambridge Press
24. Shanthala. S, "VLSI design and analysis of low power strategies for DWT from architecture level to micro subsystem level'', VTU-PhD thesis, 2010.

Mr.T.Vijayakumar received the B.E degree in Electronics and Communication Engineering and M.E degree in Digital Communication from Bangalore University, India. 1993, 2000 respectively.

Currently he is pursuing his doctoral degree at Kuvempu University, Shimoga, India. His major research interests include VLSI architecture design and image processing.Email:vijayakumar_sjbit@yahoo.com

S. Ramachandran obtained his M.Tech. and Ph. D. degrees from the Indian Institute of Technology, Kanpur and Madras respectively. He has wide academic as well as industrial experience of over 35 years, having worked as Professor in various engineering colleges as well as design engineer in industries in India and USA, designing systems and teaching/guiding students. His research interests include developing algorithms, architectures and implementations on FPGAs/ASICs for Video Processing, DSP applications, reconfigurable computing, open loop control systems, etc. He is the recipient of the Best Design Award at VLSI Design 2000, International Conference held at Calcutta, India and the Best Paper Award of the Session at WMSCI 2006, Orlando, Florida, USA. He has completed a video course on Digital VLSI System Design at the Indian Institute of Technology Madras, India for broadcast on TV by National Programme on Technology on Enhanced Learning (NPTEL) and is being broadcast in You Tube as well. He has also written a book on Digital VLSI Systems Design, published by Springer Verlag, Netherlands (www.springer.com).