Asian Journal of Information Technology 15 (2): 263-276, 2016
ISSN: 1682-3915

Multi-Core Lifting DWT Processing Engines for Image Processing

[1]M. Nagabushanam and [2]P. Kumar
[1]Department of Electronics and Communication Engineering, M.S. Ramaiah Institute of Technology, Visvesvaraya Technological University, MSRIT-Post, MSR Nagar, 560054 Bangalore, Karnataka, India
[2]Department of Electronics and Communication Engineering, K.S. Rangasamy College of Technology, Anna University, 637215 Tiruchengode, Namakkal District, Tamil Nadu, India

Abstract: In this study, micro core engines are designed for computation of predict and update outputs based on lifting scheme 9/7 Discrete Wavelet Transform (DWT) to reduce the time delay. The micro core engines are integrated into pipeline architecture for computation of 1D/2D and 3D-DWT. The 3D-DWT architecture is designed to process a 512×512 image with eight groups of frames sequentially with an improved throughput. The first stage computes 1D-DWT along the rows using four parallel processors. The memory interface with FIFO is implemented to reduce the latency between the first stage and the second stage that computes 2D-DWT along the column. The 3D-DWT is introduced to computes wavelet coefficients in the temporal direction and it is designed to operate at 512×4 clock cycles in sequence with 1D and 2D-DWT computations. The FIFO is implemented at the intermediate stages of the design tosynchronize the data movement in pipeline in all three stages. The memories attached to the input and output of every stage are designed to store the parallel processed data. The proposed architecture is suitable for high frequency and low power applications. Using the 9/7 filter for DWT computation reduces hardware complexity, memory accesses and achieves minimum error during reconstruction of images. The proposed architecture systematically combines hardware optimization techniques to develop a flexible DWT architecture that has high performance and is suitable for portable, high speed, low power applications. The 3D-DWT architecture has been implemented on Virtex-5 FPGA with utilizing of 51% of its slice registers with the frequency of operation is 373 MHz and the designed DWT-IDWT can be used as IP Core.

Key words: Lifting scheme, pipelined architecture 3D-DWT, parallel processing, FPGAs, India

INTRODUCTION

The wavelet transformation is a widely used technique for image processing applications. Unlike traditional transforms such as the Fast Fourier Transform (FFT) and Discrete Cosine Transform (DCT), the Discrete Wavelet Transform (DWT) holds both time and frequency information. It is based on a multi-resolution analysis framework. This facility improves the quality of reconstructed picture for the same compression that is possible by other transforms. The successful implementation of real time code is based on DWT requires implementation on a fast device. Field Programmable Gate Array (FPGA) implementation of DWT results in higher processing speed and lower costs when compared to other implementations such as PCs, ARM processors, DSPs, etc. The discrete wavelet transform ensures the FPGA is best suited for image coding (Sung *et al*., 2006). This is because the DWT can decompose the signals into different sub-bands by considering both time and frequency information and also provides a high compression ratio (Parhi and Nishitani,, 1993). It supports features like progressive image transmission (by quality by resolution), ease of compressed image manipulation, region of interest coding, etc. The JPEG 2000 incorporates the DWT into its standard. Recently, several VLSI architectures have been proposed to realize single chip designs for DWT. Traditionally, such algorithms were implemented using programmable DSP chips for low-rate applications or VLSI Application Specific Integrated Circuits (ASICs) for higher rates. To perform the convolution, we require a fast multiplier which is crucial in making the operations efficient. Das *et al*. (2010) have proposed the architecture of the lifting based running 3-D DWT which is a powerful image and video compression algorithm. Modified lifting

Corresponding Author: Nagabushanam, Department of Electronics and Communication Engineering, M.S. Ramaiah Institute of Technology, Visvesvaraya Technological University, MSRIT-Post, MSR Nagar, 560054 Bangalore, Karnataka, India

scheme with optimized architecture is implemented on FPGA, the architecture does not address pipelined architecture and hence reduces throughput. Chin-Fa Hsieh etc. in 2004, proposed a novel, efficient VLSI architecture for the implementation of one-dimension, lifting-based DWT. Both folded and the pipelined schemes are applied in the proposed architecture where the former scheme supports higher hardware utilization and the latter scheme speeds up the clock rate of the DWT. Chiang *et al.* (2005) have proposed a highly efficient VLSI architecture for 2-D lifting-based 5/3 filter DWT but when high speed and less area is required then using systolic array is reported in (Dastjerdi *et al*., 2009), the 3D-DWT architecture proposed is implemented using DWT (Andra *et al.*, 2002) to reduce the memory requirements, Some of the lifting-architectures proposed are reported by Taghavi and Kasaei (2003), Dai *et al.* (2004) and Xu *et al.* (2002) lifting architecture implemented with row and column processors using seven filters are reported (Kishore *et al.*, 2002) similarly in (Das and Banerjee, 2005; Mohanty and Meher, 2013) a 3D-DWT architecture proposed is of efficient regular data-flow pattern for low power, high-speed with memory efficient architectures reported by Hu and Jong (2013), the efficient ASIC implementation is reported by Hegde and Vaya (2013). The architecture is based on the pipelined and folding scheme processing to achieve near 100% hardware utilization ratio and reduce the silicon area. The proposed efficient 2-D lifting-based DWT VLSI architecture uses lossless 5/3 filter and pipelined processing. The architecture may have almost 100% hardware utilization. The advantages of the proposed DWT are higher hardware utilization, less 3D-DWT requirement and regular data flow. The architectures discussed above are suitable for FPGA implementation. In this research, lifting equations consisting of predict and update stages are designed using micro core engines that compute intermediate lifting output that are pipelined to improve throughput. The micro core engines are designed to process the data in all three dimensions leading to 3D-DWT architecture compared to the type implemented (Nagabushanam and Ramachandran, 2012).

3D-DWT architecture: Techniques involving 3D encoding of video (Parhi and Nishitani, 1993; David *et al*., 2002; Tze-Yun, 2007; Das *et al*., 2010) have demonstrated higher compression and the image quality is compared with MPEG (Chin *et al*., 2004; Chiang *et al*., 2005; Dastjerdi *et al*., 2009; Andra *et al*., 2002) based compression techniques. DWT that transforms input image into sub bands are encoded using appropriate encoding techniques leading to compression. The 3D encoding of input data reduces memory requirements with data processing in the temporal Discrete Wavelet Transform (DWT) on 3D blocks coming from a temporal splitting of the sequence. The 3D Wavelet Transforms Performs 2D-DWT on each frame and an additional 1D-DWT in the time direction (Mohanty and Meher, 2013) as shown in Fig. 1. The 3D architecture performs 2D-DWT on every frame where each frame decomposed into four sub bands. Each sub band is grouped into LL, LH, HL and HH bands and 1D-DWT is performed in the temporal domain and hence eight sub bands of LLL, LLH, LHL, LHH, HLL, HLH, HHL and HHH are obtained. The input data consisting of N×N×M (N represents number of pixels, M represents number of frames) is grouped into NxNx8 group of frames for data processing. Every GOP consists of 8 frames each of size N×N is first processed using 2D-DWT. Each 2D-DWT consists of two stages of row processing and column processing. Row processing and column processing blocks consists of low pass and high pass filters that decomposes the 1D data into low pass and high pass DWT filter coefficients. The row processing unit consists of single stage of low pass and high pass filters that decomposes the row elements of input data into L and H sub bands. The L and H outputs are processed by the column processing unit consisting of two single stage low pass and high pass filters that processes the L and H sub band of data for further decompose of data into LL, LH (low pass) and HL and HH (high pass) sub bands. Figure 2 shows the 1D-DWT processing units that consists of Low pass and high pass filters.

The 1D-DWT processes N rows of input samples each, row consist of N pixels, the processed or filtered output is down sampled by 2 to obtain N/2 outputs that are stored in a separate 3D-DWT. The processing of N rows consisting of N pixel requires $9\times N^2$ multiplication and $9\times N^2$-1 additions.

- G_0 = High pass filter
- H_0 = Low pass filter
- $\downarrow$ = Down Sample by 2

Lifting scheme reduces the number of arithmetic operations, memory elements and computation time and is widely being adopted for DWT computation (Chin *et al*., 2004). The block diagram for lifting scheme (Chin *et al*., 2004) is shown in Fig. 3.

The 9/7 filter coefficients are reduced to five coefficients represented by α, β, γ, δ and ζ are the lifting

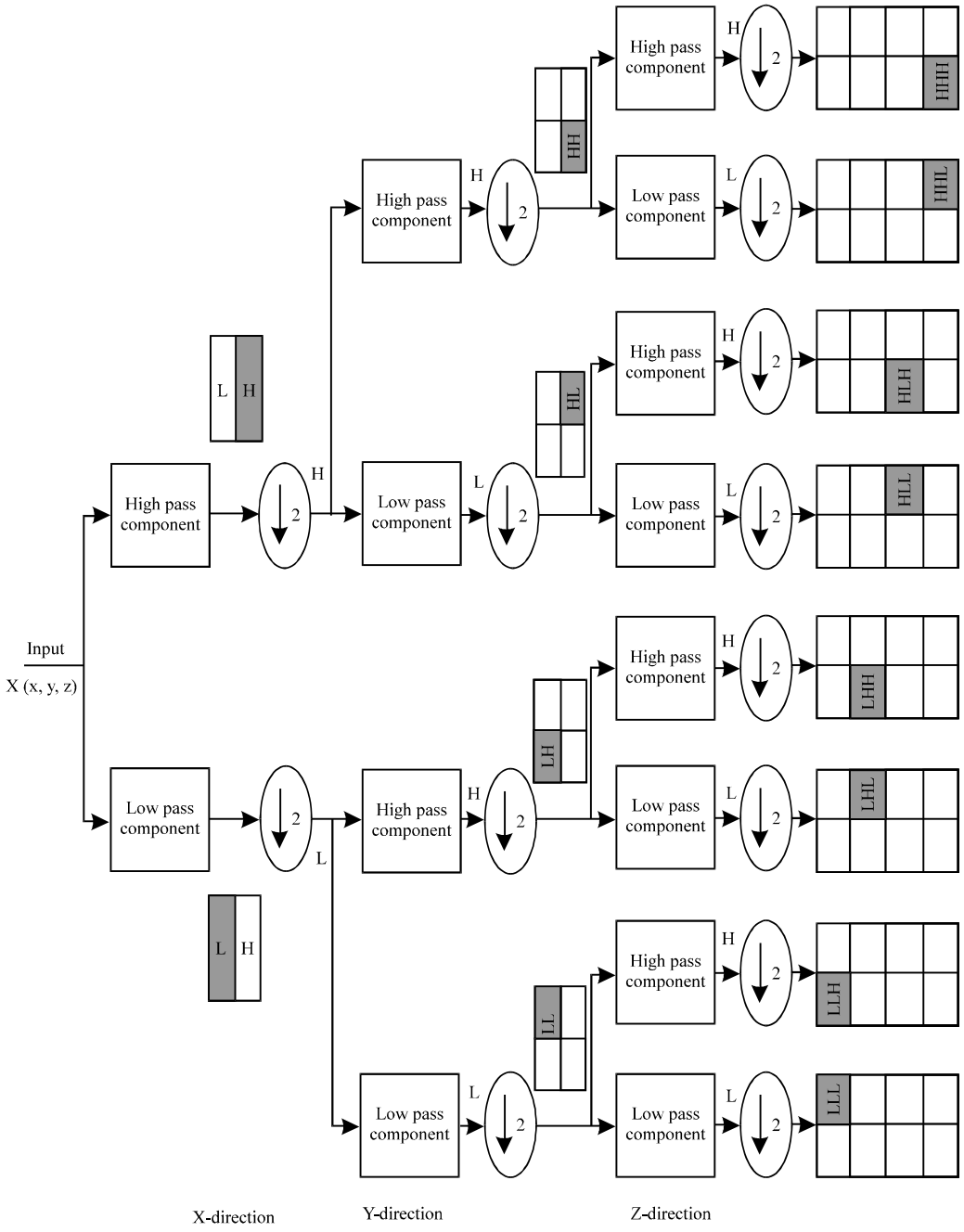

Fig. 1: 3D-DWT Processing

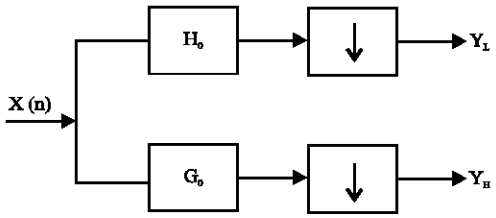

Fig. 2: The 1D-DWT decomposition

coefficients. In lifting algorithm, the input signal x(n)or (x_i) is split into even and odd parts. The even parts are represented as (x_{2i}) and an odd part is represented as (x_{2i+1}). Lifting scheme algorithm consists of 6 steps with 2 update and 2 predict stages and 2 scaling. Therefore, the first step of lifting is given by the Eq. 1 and 2.

$$d_i^1 = \alpha\left(x_{2i} + x_{2i+2}\right) + x_{2i+1} \quad (1)$$

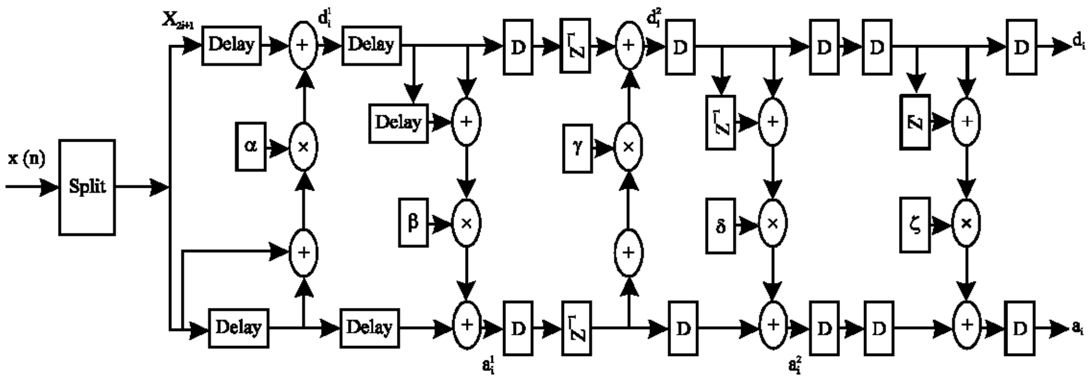

Fig. 3: Lifting scheme architecture

$$a_i^1 = \beta\left(d_i^1 + d_{i-1}^1\right) + x_{2i} \quad (2)$$

Equation 1 is Predictor P1 and Eq. 2 is Updater U1. Then the second lifting step consists of two stages predict 2 and update 2 are represented in Eq. 3 and 4:

$$d_i^2 = \gamma\left(a_i^1 + a_{i+1}^1\right) + d_i^1 \quad (3)$$

$$a_i^2 = \delta\left(d_i^2 + d_{i-1}^2\right) + a_i^1 \quad (4)$$

The last two stages are the scaling functions and it is performed in order to obtain the approximation and detail coefficients of DWT as given in Eq. 5 and 6:

$$a_i = \zeta a_i^2 = G_1 \quad (5)$$

$$d_i = d_i^2/\zeta = G_2 \quad (6)$$

It has been observed that the computation of the final coefficients requires 6 steps. Each stages of computation consist of either a_i and d_i coefficients require three samples of data, these samples are read from memory which requires three clock cycles. Computation of $a_i$output requires d_i and d_i computation requires xi, this interdependency of computation requires time and hence increases delay and reduces throughput. In order to reduce computation time and improve throughput a novel architecture is proposed in this study, the architecture is designed in a way it consists of multiple processing engines so that each engine can independently compute the a_i and d_i coefficients without time delay.

MATERIALS AND METHODS

Five stage pipelined with four fully parallel architecture: The lifting equations presented in Eq. 1-6 are expanded for time interval I = 0 to I = 3 and shown in Table 1. Scaling terms are not included in the table as they are performed by multiplication operation. The computation delays for each of the stages are computed and are shown in Table 2.

At time t = 0, d_0^1 is computed which requires samples X_{0-2}. Computation of a_0^1 requires two samples of d_0^1 and X_0, thus computation of a 01 is delayed by 6 clock cycles. Similarly, computing d_0^2 requires a_0^1, a_1^1 and d_0^1 which requires multiple clock cycles. Assuming that the input samples are pre-fetched and stored in internal memory, the latency is found to be 18 clock cycles and the throughput is found to be two clock cycles. The proposed architecture improves the through put to one clock cycl. It is observed that the computation of update 1 is delayed by three clock cycles until predict 1 data is completed, update 1 requires two samples of predict 1 data, similarly update 2 requires two samples of predict 2 data, thus causing delay and increasing throughput to more than one clock cycle. Figure 4 shows the proposed architecture for DWT computation. Predict 1 sample are computed after 12 clock cycles there will be three outputs of predict 1 samples. The memory elements at the output of predict 1 unit stores these output samples. After 12 clock cycles, update 1 unit is enable to start computation, thus update 1 unit will have all the samples required for computation and there will be no delay. As predict 1 and update 1 are operated in parallel the delay in computation of stage 1 output samples are reduced, only limitation is there will be 12 clock cycles latency in stage 1.

At the end of 12 clock cycles, there are 6 predict 1 samples that are generated and stored in 3D-DWT. Thus computation of update 1 clock is delayed by 12 clock cycles and from 13th clock cycle, update 1 is computed. Further the computation of predict 2 is delayed by 12 clock cycles as compared with update 1. Thus, the first predict 2 coefficient is obtained at the end of 25th clock cycle. Similarly, computation of update 2 is delayed by

Table 1: Lifting equation iterations

Predict 1	Predict 2	Update 1	Update 2
$d_i^1 = \alpha(X_{2i} + X_{2i+2}) + X_{2i+1}$	$d_i^2 = \gamma(a_i^1 + a_{i+1}^1) + d_i^1$	$a_i^1 = \beta(d_i^1 + d_{i-1}^1) + Z_{2i}$	$a_i^2 = \delta(d_i^2 + d_{i-1}^2) + a_i^1$
$d_0^1 = \alpha(X_0 + X_2) + X_1$	$d_0^2 = \gamma(a_0^1 + a_1^1) + d_0^1$	$a_0^1 = \beta(d_0^1 + d_{-1}^1) + X_0$	$a_0^2 = \delta(d_0^2 + d_{-1}^2) + a_0^1$
$d_1^1 = \alpha(X_2 + X_4) + X_3$	$d_1^2 = \gamma(a_1^1 + a_2^1) + d_1^1$	$a_1^1 = \beta(d_1^1 + d_0^1) + X_2$	$a_1^2 = \delta(d_1^2 + d_0^2) + a_1^1$
$d_2^1 = \alpha(X_4 + X_6) + X_5$	$d_2^2 = \gamma(a_2^1 + a_3^1) + d_2^1$	$a_2^1 = \beta(d_2^1 + d_1^1) + X_4$	$a_2^2 = \delta(d_2^2 + d_1^2) + a_2^1$
$d_3^1 = \alpha(X_6 + X_8) + X_7$	$d_3^2 = \gamma(a_3^1 + a_4^1) + d_3^1$	$a_3^1 = \beta(d_3^1 + d_2^1) + X_6$	$a_3^2 = \delta(d_3^2 + d_2^2) + a_3^1$

Table 2: Throughput and computation delay in generatic lifting architecture

Clock	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Input	x_1	x_2														
Predict 1			d_0^1	d_1^1	N_D	d_1^2	N_D	d_3^1	N_D	d_4^1	N_D	d_5^1		d_6^1		d_7^1
Update 1			a_0^1	a_1^1	N_D	a_2^1	N_D	a_3^1		a_4^1		a_5^1		a_6^1		a_7^1
Predict 2				d_0^2		d_1^2		d_2^2		d_3^2		d_4^2		d_5^2		d_6^2
Update 2				a_0^2		a_1^2		a_2^2		a_3^2		a_4^2		a_5^2		a_6^2
Approx.				a_0		a_1		a_3		a_4		a_5		a_6		a_7
Detail				d_0		d_1		d_3		d_4		d_5		d_6		d_7

N_D = No Data computed

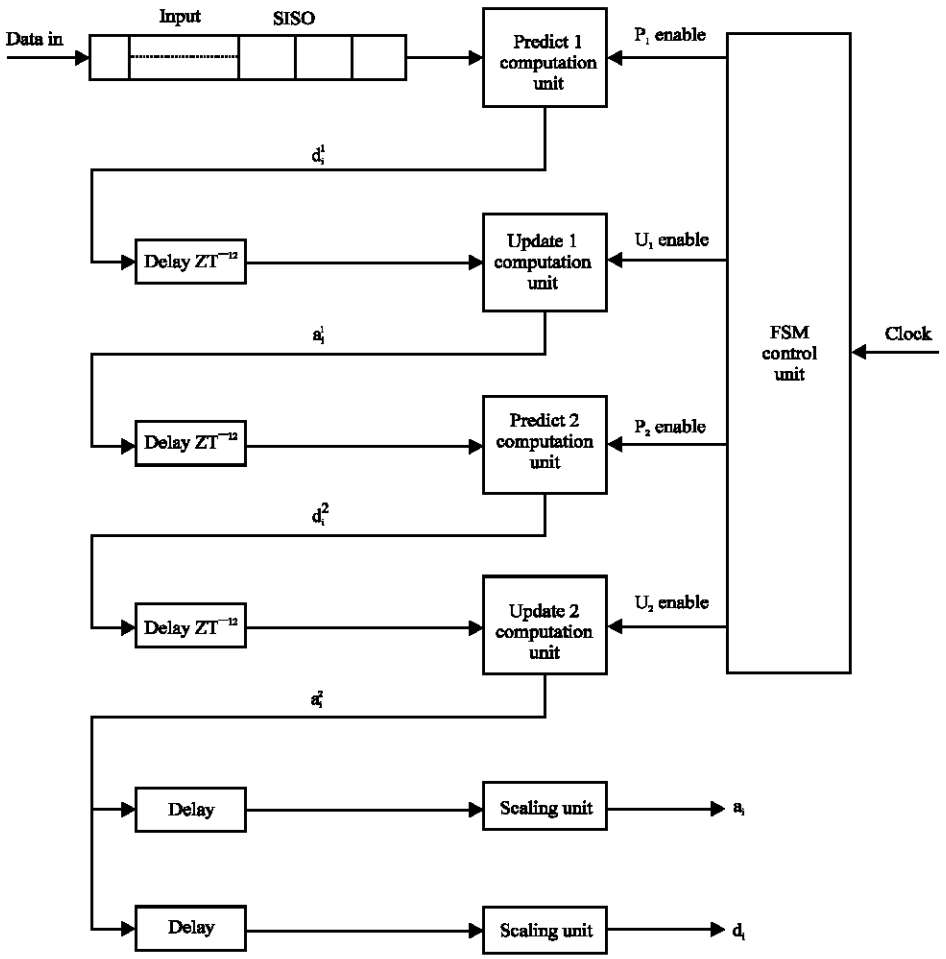

Fig. 4: Proposed 1D DWT Computation Unit

Table 3: Throughput and computation delay in proposed lifting architecture

Clock	1	2	3	4	5	6	- - -	12	13			25			36				48	49			
Input	x_1	x_2																					
P_1	-	-	-	-	-	d_0^1	d_1^1	d_5^1	d_6^1	d_7^1	d_8^1	-	-	-	-	-	-	-	-	-	-	-	d_1^1
U_1	-	-	-	-	-	-	-	-	a_0^1	a_1^1	a_2^1	a_3^1	-	-	-	-	-	-	-	-	-	-	a_0^1
P_2															d_0^2	d_1^2	d_2^2	-	-	-	-	-	a_0^2
U_2																			a_0^2	a_1^2	a_2^2	a_3^2	a_0^3
a_1																				a_0	-	-	a_n
d_1																				d_0	-	-	d_n

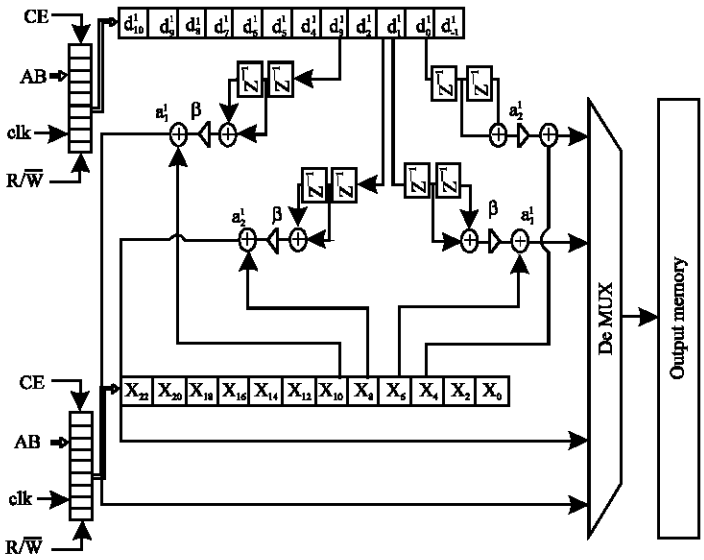

Fig. 5: Predict 1 processing engine

another 12 clock cycle. Introducing 12 clock cycle delay between intermediate steps ensures that minimum of six samples of data are available in the 3D-DWT from the previous stages. The proposed architecture reads the row elements from the main 3D-DWT into the intermediate 3D-DWT 1. The control unit is designed to read 4N elements into the intermediate 3D-DWT 1. The 4N elements are then processed by the first micro core engine to compute the d^1_i output samples that are stored in intermediate 3D-DWT 2. For the first 12 clock cycles three micro core engines are disabled. After 12 clock cycles the second micro core engine is enabled to process data from intermediate 3D-DWT 2 and the first core processes data from 3D-DWT 1. Further, the third and fourth micro core engines are enabled after 36th and 48th clock cycle. The first micro core engine completes the data processing of all rows in N×N clock cycles. The total delay in computing the 1D DWT is (N×N+48) clock cycles. As the proposed architecture introduces forced delay in computation of Predict and Update stages, the throughput is reduced to 1 clock cycle. Table 3 shows the computation delay and the through put in the proposed architecture.

The micro core engines are designed to compute predict and update phases with parallel processing approach. The intermediate 3D-DWT that stores the input data is designed to be a PIPO register that is designed to access all the contents in parallel. The predict 1 architecture shown in Fig. 5 consists of four processing engines that have three delay elements that are loaded serially with input data from intermediate 3D-DWT. The input samples are accumulated and multiplied as in lifting equations is already discussed. In every clock cycle, four predict 1 samples are generated and de-multiplexed into appropriate locations in intermediate 3D-DWT 2.

The d^1_i output samples generated are stored in intermediate 3D-DWT, at the beginning of 13th clock cycle the samples are read into another PIPO register as shown in Fig. 6. The micro core engine is enabling at the 13th clock cycle and the PIPO data is read into the parallel processing units. For computation of Update 1, it also required in having the input samples; hence there are two PIPO registers one for d_i samples and one for input samples. The computed a^1_i are stored in intermediate 3D-DWT 2. Similarly predict 2 and update 2 micro engines are designed to process data in parallel with intermediate registers and PIPO registers.

The final scaling of lifting coefficients is achieved with the scaling of update 2 and predict 2 samples.

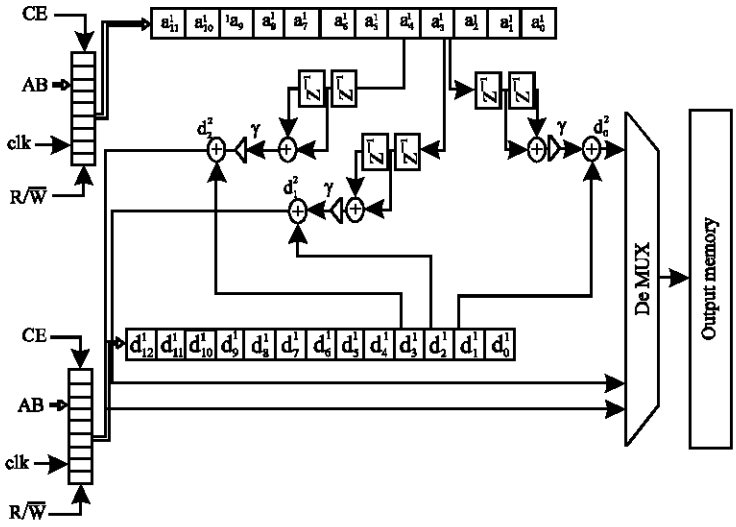

Fig. 6: Micro core engine for update 2 computation

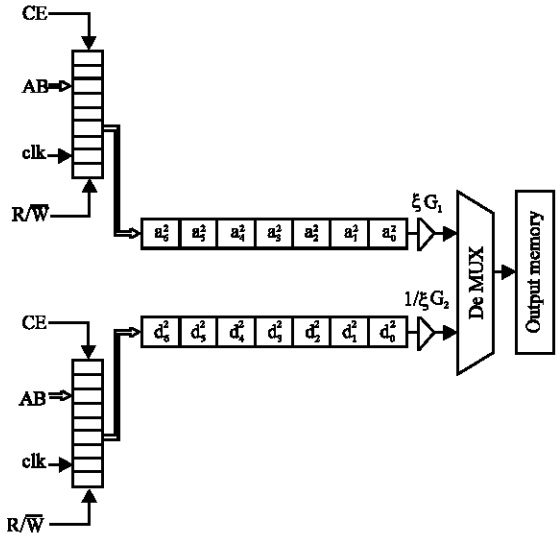

Fig. 7: Micro core engine for final stage of lifting computation

Figure 7 shows the final stage in lifting computation that consists of input 3D-DWT and PIPO registration and scaling function using multiplier operation and demultiplexer with output 3D-DWT. The five micro core engines are designed to operate in sequence so that the throughput is achieved in one clock cycle. The proposed architecture also processes four rows of input data simultaneously, each data is processed by the proposed micro core based 1D-DWT computation architecture. Figure 7 shows the top level architecture of 1D-DWT computation that computes DWT on four rows of input data simultaneously. The main clock is used to read data from input 3D-DWT to four intermediate memories. The intermediate 3D-DWT is operated at twice the clock speed as compared with intermediate 3D-DWT so as to read the data samples into the Predict 1 computation unit. As the predict 1 unit has to perform multiplication and addition operation, the multiplier is operated at clock frequency that is 16 times higher than the main clock and the adder unit is designed to operate 8 times higher than the main clock. The processed data out from the four parallel processing units needs to be reorganized in designing of de-multiplexer unit. The d_i computation unit consists of four parallel processing modules as shown in Fig. 8 for computation.

FSM based control unit is designed to control the main 3D-DWT, intermediate 3D-DWT, d_i computation unit, demultiplexer and the output 3D-DWT. The control signals from the FSM are synchronized with main clock and hence data loss is prevented. Figure 9 shows the d_i computation unit that consists of an internal FSM to enable the 3D-DWT unit and the arithmetic processing elements.

The FSM control unit controls the data synchronization between data movement from registers to the final output computation. In this work, Carry Look Ahead (CLA) Adder and Wallace Tree (WT) multipliers are used. As they have less delay and regular structure. Figure 4 shows the four stages of pipelined parallel processing architecture with micro core engines to compute the predict and update samples. The last stage of the proposed architecture is the scaling process as shown in Fig. 4.

Computation of 1D-DWT based on lifting approach in the proposed architecture is carried out using four parallel stages that operate on four input rows of image.

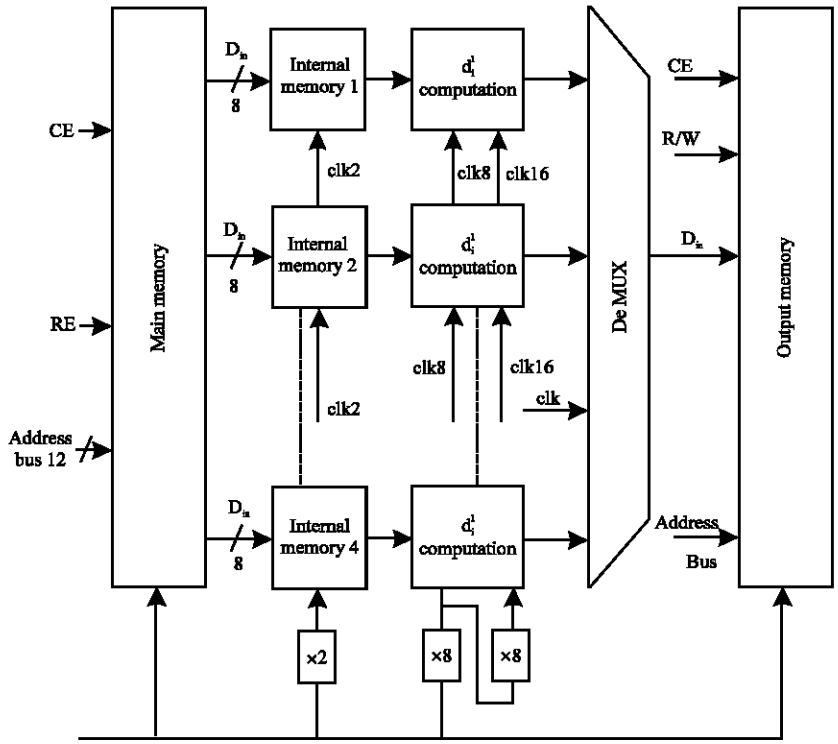

Fig. 8: First stage parallel processing engines

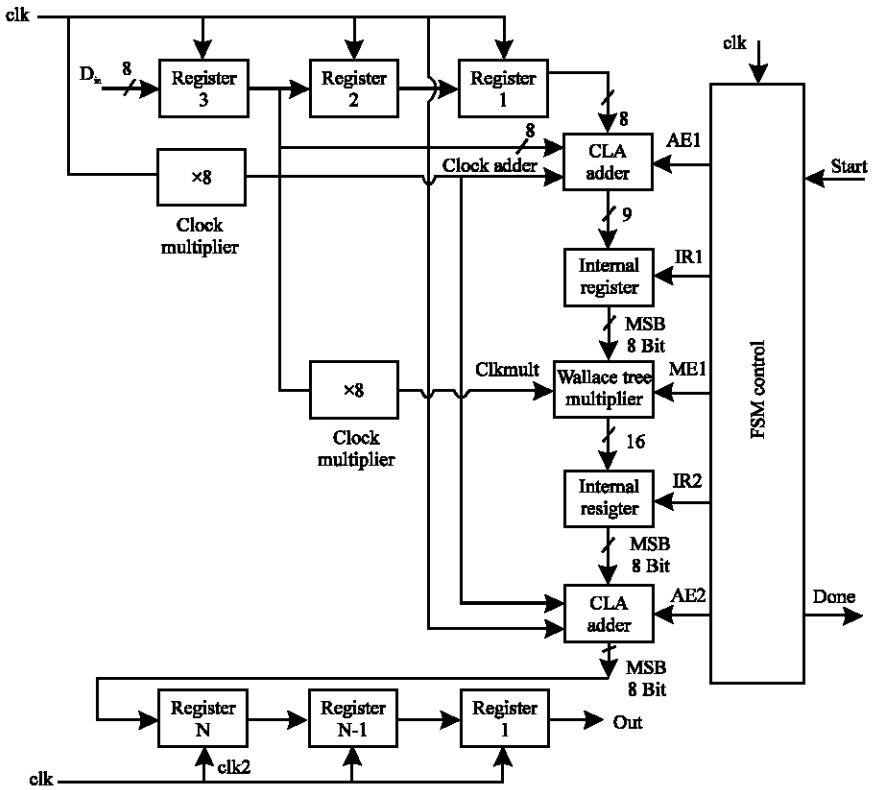

Fig. 9: Predict/Update computation architecture with FSM

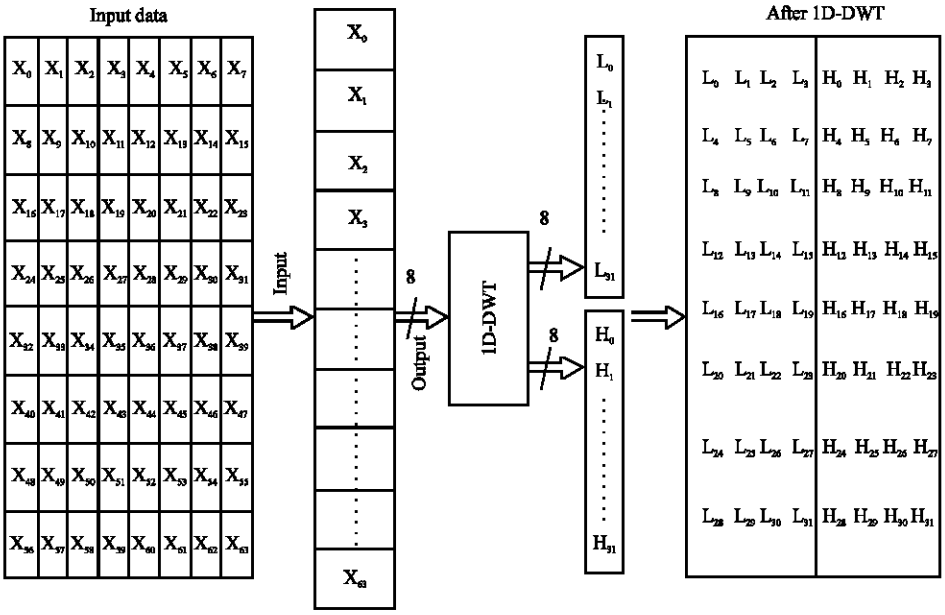

Fig. 10: The 1D-DWT decomposition

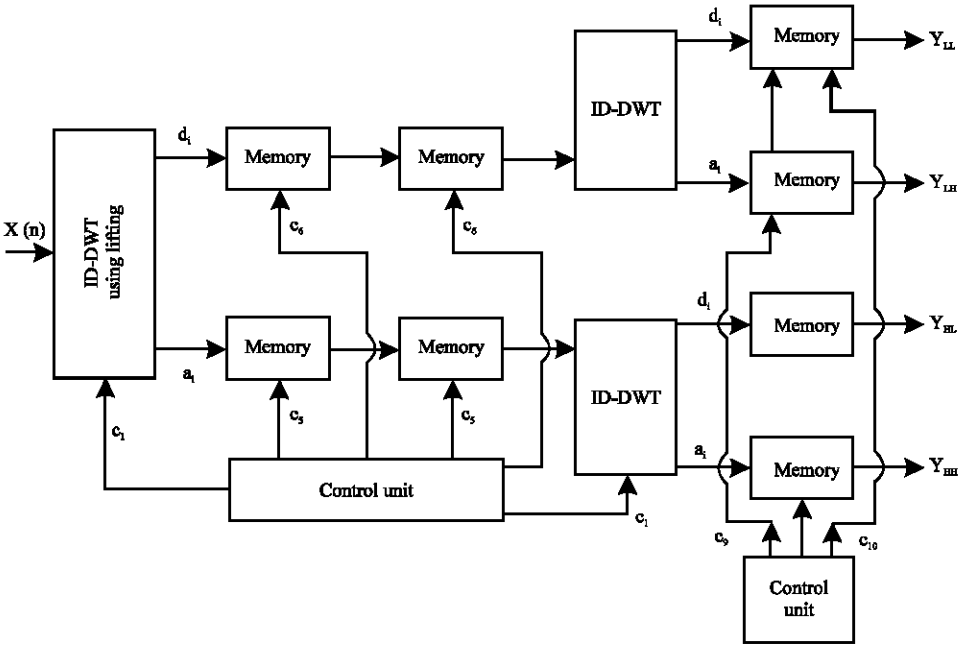

Fig. 11: The 2D-DWT architecture

The row operations are pipelined with five stages of computation units with each unit consisting of micro core engines. Figure 10 shows the 1D- DWT, with input image being decomposed into sub band outputs of L and H. After 1D-DWT the input image is reorganized into two bands of data as shown in Fig. 10.

2D-DWT architecture: Figure 11 shows the two level decomposition of 2D-DWT using proposed lifting scheme architecture. The output of 1D-DWT are stored in separate 3D-DWT for computation in second level DWT and an intermediate 3D-DWT is introduced that feeds data to the second level DWT computation logic. The two stages of 3D-DWT units between 1D and 2D-DWT

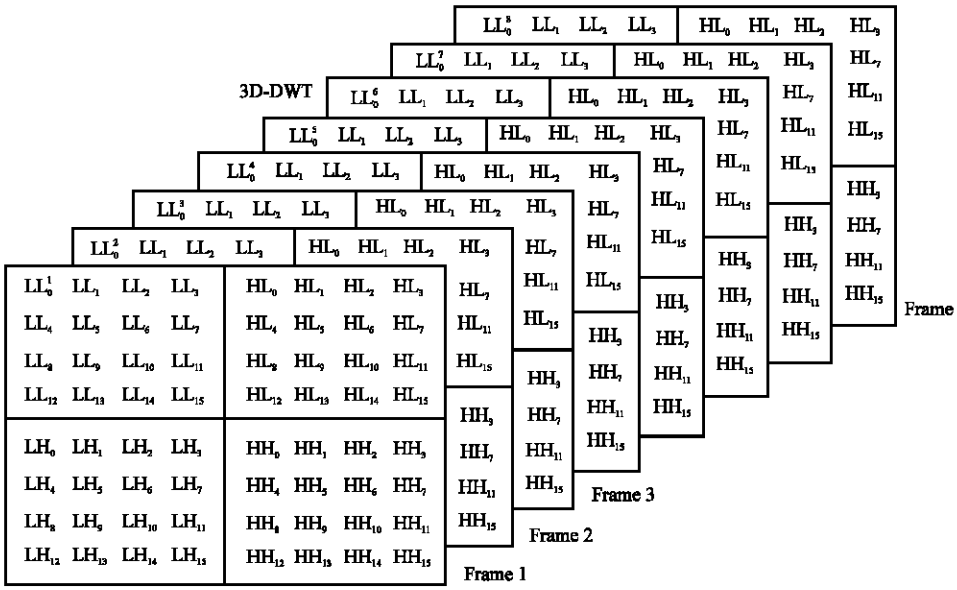

Fig. 12: The 2D-DWT of 8 input frames

improves throughput by performing parallel processing operations of 1D and 2D-DWT. The input 3D-DWT consists of L and H bands of coefficients that are read into the 2D processing module. Two 1D-DWT blocks processes the L and H sub bands simultaneously controlled by clock input. Between 1D and 2D-DWT processing, two 3D-DWT blocks are introduced to synchronize data processing, the first 3D-DWT stores the L and H sub bands and the second 3D-DWT is designed to store the rearranged data as shown in Fig. 11 prior to 2D-DWT processing. The intermediate clocks C_5 and C_6 control the read and write operations of intermediate 3D-DWT. The four memories at the last stage of 2D-DWT is controlled by C_9 and C_{10} clock cycles that reorganizes the 2D output as shown in Fig. 11. As there are two 1D-DWT processing units that operate in parallel on each of the L and H sub band unit, the 3D-DWT arrangement plays a vital role in data flow control. The 1D-DWT architecture designed (Fig. 8) is used to process L and H samples, as there are four parallel processing units in 1D-DWT and there are N/2 columns to be processed for the total number ofclock cycles required is (N/2×N/2)/4+48. Thus the computation of 2D-DWT using proposed architecture is [(N×N)/4+48+(N/2×N/2)/4+48] clock cycles and the proposed architecture requires multiple intermediate memories. For an input image of N×N×8, after 2D-DWT computation of 8 frames the input data is consists of four sub bands in each frame and there are eight frames as shown in Fig. 12.

Figure 12 shows the four level sub bands obtained using the proposed architecture by introducing intermediate memories; pipelining and parallel processing is introduced for 2D-DWT computation.

The 3D-DWT with proposed architecture: The 3D-DWT is computed by performing 1D-DWT in the temporal domain. Figure 13 shows the 3D-DWT computation process. The eight input frames each consisting of N/2xN/2 elements is processed using 1D-DWT to produce two set of outputs. The four sub bands that are reorganized and processed in the temporal domain after DWT consist of 8 sub bands.

Figure 14 shows the proposed 3D-DWT architecture with eight frames that are stored in separate memories is proposed simultaneously using eight different 2D-DWT processors. The eight input frames after 2D-DWT are decomposed into four sub bands to compute 3D-DWT on all the decomposed sub bands. The input needs to be rearranged as shown in Fig. 13. The output frames of 2D-DWT, the third level DWT is to be computed on each of the sub band of eight frames.The component LL_0^1 is the first frame and LL_0^2 is the second frame and LL_0^3 is the third frame and so onand so on till LL_0^7 of eight frame are grouped and stored in separate 3D-DWT. Similarly, HL, LH, HH components of all eight frames is regrouped into separate memories. These regrouped components are processed using four 1D-DWT process as shown in Fig. 14. The 3D output decomposes the N×N×8 input frame in to eight sub bands of N/4×N/4×4 components representing LLL, LLH, LHL, LHH, HLL, HLH, HHL and HHH.

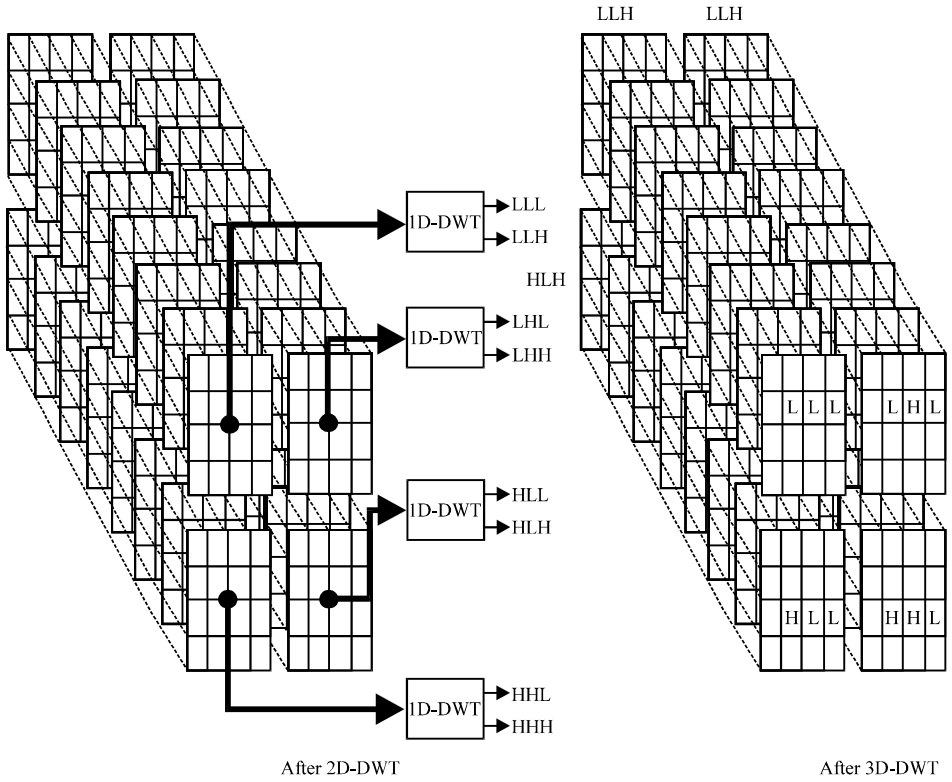

Fig. 13: The 3D-DWT computations

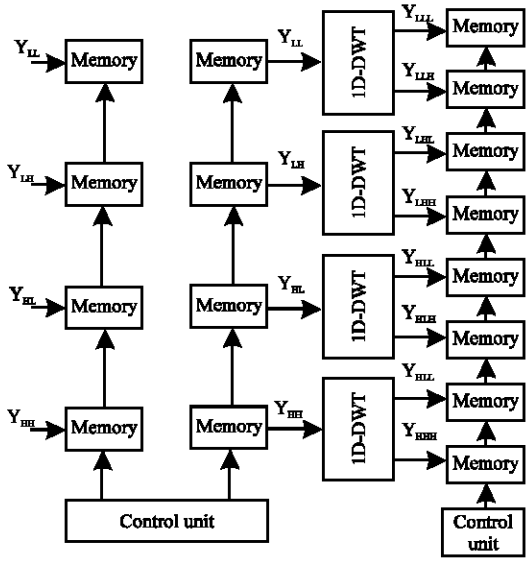

Fig. 14: 3D Lifting based DWT

RESULTS AND DISCUSSION

HDL code for the proposed model is developed in Verilog along with a test bench for functional verification. The input image consisting of group of frames are stored in an input file and the Verilog code developed with the test bench is interfaced with the text file storing the image. The test bench reads the input image files and sends the data pixel to 3D-DWT engine for DWT decomposition. The test bench reads eight image files in parallel and feeds the eight input ports simultaneously. The 3D-DWT decomposes the input images in to eight low pass and high pass filter combinations as LLL, LLH, LHL, LHH, HLL, HLH, HHL and HHH coefficients. Each combination of 3D-DWT coefficients is stored in separate output files by image file writer. Figure 15 shows the block diagram of the test bench for validation of proposed 3D-DWT architecture.

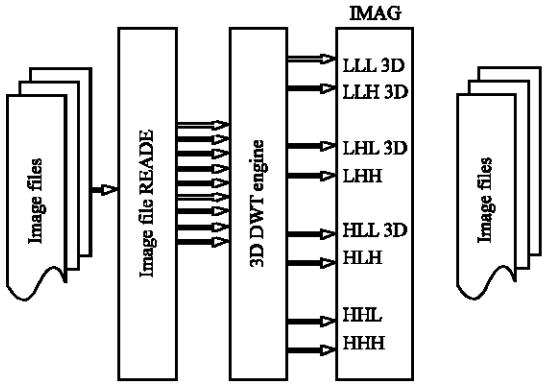

Fig. 15: Test Bench for 3D-DWT

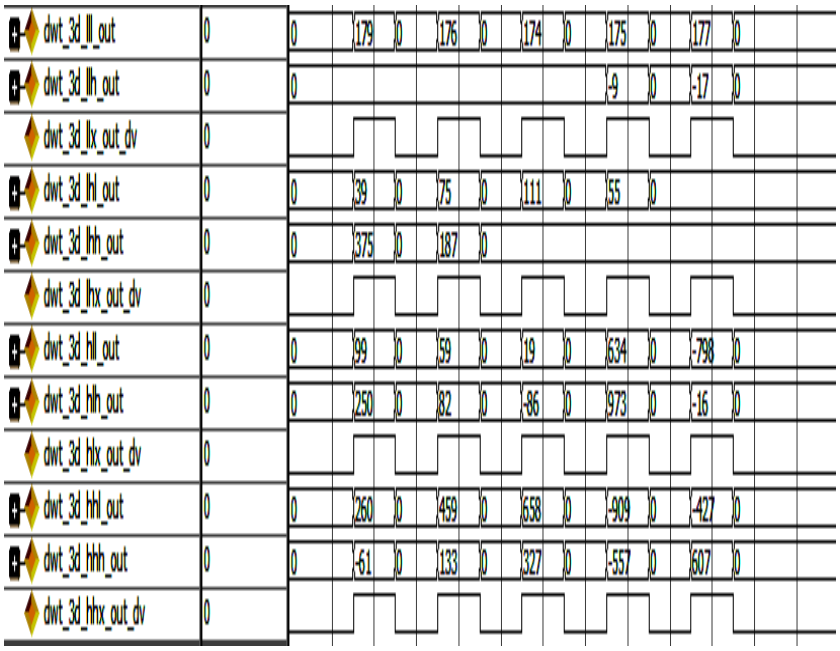

Fig. 16: ModelSim simulation results for 3D-DWT

(a)

(b)

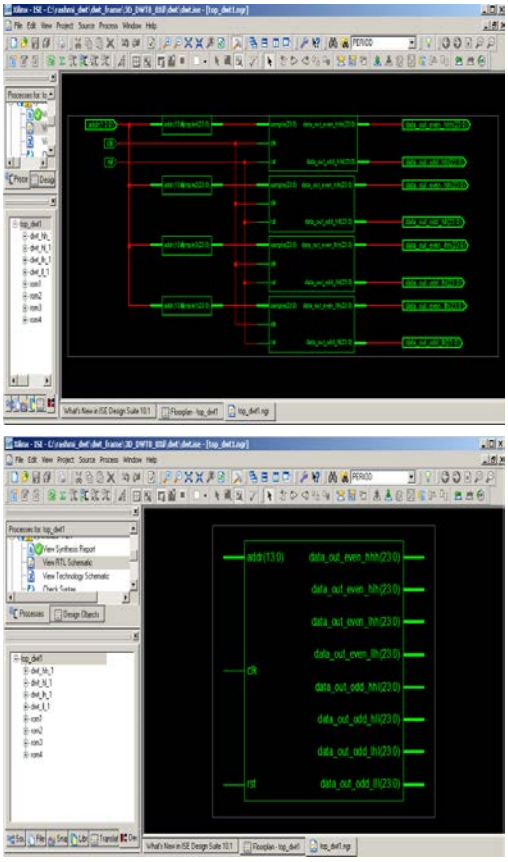

Fig. 17: The RTL Schematic of 3D-DWT

Simulation results are captured in ModelSim window and are presented in Fig. 16. The numerical values obtained are cross verified with the Matlab simulation results for logical correctness.

The functionally corrected HDL code is synthesized using Xilinx ISE targeting Virtex-5 FPGA device. The synthesis results obtained are captured and discussed. Figure 17 shows the top level RTL schematic of 3D-DWT. The internal architecture is shown in Fig. 17b, the hierarchical model consists of 1D-DWT for row processing, two 1D-DWT for column processing and four 1D-DWT for temporal processing.

The synthesis results in terms of area and speed are captured and reported in Table 4. The Virtex-5 device consists of 49645 LUTs and 3456 3D-DWT units. For 3D-DWT processing only 51 and 13% of resource share has been utilized, respectively.

Suitable constraints have been applied for speed optimization and it is found that the proposed 3D-DWT architecture operates at maximum frequency of 373 MHz (Fig. 17) and hence it is suitable for high speed applications. Table 5 shows the comparison of proposed architecture with references.

The processing speed can be further improved by the design of parallel processing architecture and multiplier less architecture that can be incorporated at the first stage of DWT and multicore engines,

Table 4: Device utilization summary

Selected device: 5v1x155tff1136-3

Slice logic utilization	Used	Total	Utilization (%)
Number of slice registers	38308	97280	39
Number of slice LUTs	49645	97280	51
Number used as logic	48189	97280	47
Number used as 3D-DWT	3456	26240	13
Number used as SRL	3456		
Slice logic distribution			
Number of LUT flip flop pairs used	52854		
Number with an unused flip flop	14546	52854	27
Number with an unused LUT	3209	52854	6
Number of fully used LUT-FF pairs	35099	52854	66
Number of unique control sets	2761		
IO utilization			
Number of IOs	237		
Number of bonded IOBs	237	640	37
Specific feature utilization			
Number of BUFG/BUFGCTRLs	2	32	6
Minimum period: 2.676 ns (Maximum frequency: 373.692 MHz)			
Minimum input arrival time before clock	1.865 ns		
Maximum output required time after clock	2.878 ns		
Maximum combinational path delay: No path found			

Table 5: Comparison report

	Dai *et al.* (2004)	Xu *et al.* (2002)	Taghavi and Kasaei (2003)	Anirban *et al.* (2010)	Proposed
Memory requirement	4N2+896N+ 968^{*2} (spatial+ temporal)	$5N^2$ (only temporal)	5N2 fast and few slow (only temporal)	$5N^2$ (temporal) +5N (spatial)	$N^2+(N/2)^2+(N/4)^2$
Memory referencing at fixed FPS (i, o/T)×3	-	5 ip/T, op/T 5	5 ip/T, 5 op/T	5 ip/(2T), 6 op/(2T)	5 ip/T, 5 op/T
Throughput	-	1 res/cycle	1 res/cycle	2 res/cycle	1 res/cycle
1 Level computational latency		$4N^2$ cycles	$4N^2$ cycles	($2N^2$) cycles	$2N^2$ cycles
Operating frequency				321 MHz (Xilinx FPGA xc4vfx140)	373 MHz (Xilinx FPGA Virtex-5)
Area	-	-	-	1825 slices	39000
Hardware utilization	-	-	-	100%	51% (only for DWT) DWT)
GOP	32 (max)	Infinite	Infinite	Infinite	Infinite
Filter bank	For D-9/7	D-9/7	For D-9/7	D-9/7	D-9/7
Design type	Complete 3-D	Temporal processor	Temporal processor	Complete 3D	Complete 3D
Number of levels				1	1

respectively. Power dissipation for the proposed architecture is found to be <1 W; the power dissipation is due to the presence of large intermediate memories (Table 1).

CONCLUSION

Processing of 3D data such as video sequences using 3D-DWT processor requires complex processing units that have higher latency. Lifting scheme reduces the computation complexity of DWT processing as there are 5 coefficients instead of 9/7 filter coefficients. Interdependency between stages causes delay in lifting scheme in order to overcome such interdependency micro engines are designed to compute the intermediate outputs of lifting algorithm. Micro core engines operate in parallel to improve the processing speed of the sub band computation. The pipelined architecture also improves the throughput of 3D-DWT computation. The parallel processing architecture that computes DWT of all the 512 rows in the first stage, the 2D-DWT in the second stage and the 3D-DWT in the third stage are pipelined by introducing intermediate memories to improve latency. The developed architecture is modeled using Verilog HDL and is validated on FPGA platform. In brief, novel architecture for 3D-DWT is designed, modeled and implemented on FPGA. The modified architecture operates at maximum frequency of 373 MHz utilizing 39% of hardware slices where the maximum power consumption is 1W out of which the quiescent power is 673 mW. The

minimum input arrival time before clock is 1.865 ns and the maximum output required time after clock is 2.878 ns hence the architecture is hardware efficient and low power.

REFERENCES

Andra, K., C. Chakrabarti and T. Acharya, 2002. A VLSI architecture for lifting-based forward and inverse wavelet transform. IEEE Trans. Signal Process., 50: 966-977.

Chiang, J.S., C.H. Hsia, H.J. Chen and T.J. Lo, 2005. VLSI architecture of low memory and high speed 2D lifting-based discrete wavelet transform for JPEG2000 applications. Proceedings of the IEEE International Symposium on Circuits and Systems ISCAS 2005, May 23-26, 2005, IEEE, USA., pp: 4554-4557.

Dai, Q., X. Chen and C. Lin, 2004. A novel VLSI architecture for multidimensional discrete wavelet transform. Circuits Syst. Video Technol. IEEE. Trans., 14: 1105-1110.

Das, A., A. Hazra and S. Banerjee, 2010. An efficient architecture for 3-D discrete wavelet transform. IEEE Trans. Circuits Syst. Video Technol., 20: 286-296.

Das, B. and S. Banerjee, 2005. Data-folded architecture for running 3D DWT using 4-tap daubechies filters. Circuits, Devices Syst. IEE. Proc., 152: 17-24.

Dastjerdi, M.M., A.A. Kusha and M. Pedram, 2009. BZ-FAD: A low-power low-area multiplier based on shift-and-add architecture. Very Large Scale Integr. Syst. IEEE. Trans., 17: 302-306.

Hegde, G. and P. Vaya, 2013. A parallel 3-D discrete wavelet transform architecture using pipelined lifting scheme approach for video coding. Int. J. Electr., 100: 1429-1440.

Hu, Y. and C.C. Jong, 2013. A memory-efficient high-throughput architecture for lifting-based multi-level 2-D DWT. Signal Proc. IEEE. Trans., 61: 4975-4987.

Mohanty, B.K. and P.K. Meher, 2013. Memory-efficient high-speed convolution-based generic structure for multilevel 2-D DWT. Circuits Syst. Video Technol. IEEE. Trans., 23: 353-363.

Nagabushanam, M. and S. Ramachandran, 2012. Fast implementation of lifting based 1D/2D/3D DWT-IDWT architecture for image compression. Int. J. Comput. Applic., 51: 35-41.

Parhi, K.K. and T. Nishitani, 1993. VLSI architectures for discrete wavelet transforms. Very Large Scale Integr. Syst. IEEE. Trans., 1: 191-202.

Sung, T.Y., H.C. Hsin, Y.S. Shieh and C.W. Yu, 2006. Low-power multiplierless 2-D DWT and IDWT architectures using 4-tap daubechies filters. Proceedings of the Seventh International Conference on Parallel and Distributed Computing, Applications and Technologies, 2006 PDCAT'06, December 4-7, 2006, IEEE, Taipei, Taiwan, pp: 185-190.

Taghavi, Z. and S. Kasaei, 2003. A memory efficient algorithm for multi-dimensional wavelet transform based on lifting. Proceedings of the 2003 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'03), April 6-10, 2003, IEEE, USA., pp: 401-404.

Xu, J., Z. Xiong, S. Li and Y.Q. Zhang, 2002. Memory-constrained 3D wavelet transform for video coding without boundary effects. Circuits Syst. Video Technol. IEEE. Trans., 12: 812-818.