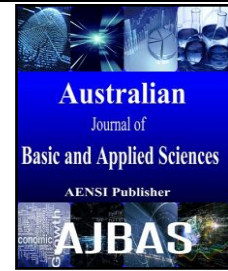




ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com



High Throughput Reconfigurable Resource Sharing Parallel Architecture for a 2D Lifting Based DWT

¹S.Senthil Kumar and ²R. Radhakrishnan¹Anna University, SVS College of Engineering, ECE Dept, Associate Professor, Coimbatore, India.²Anna University, Vidhya Mandhir Institute of Technology, Principal, Erode, India.

ARTICLE INFO

Article history:

Received 16 April 2015

Accepted 12 June 2015

Available online 1 July 2015

Keywords:

Data broadcast, parallel architecture, embedded decimation, 2D DWT, lifting scheme, FPGA implementation.

ABSTRACT

Two dimensional discrete wavelet transform has been widely used for image compression and it is also the major source of computational complexity. To reduce the complexity of computation and to increase speed of 2-D DWT, two different architectures have been developed by employing data broadcast and resource sharing techniques. This paper presents these techniques composing of two filter modules namely the row and column filters working in parallel and pipeline fashion with 100% hardware utilization. An embedded decimation technique has been employed, which uses resource architecture capable of performing J levels of decomposition for N X N image in approximately $2N^2(1-4^{-J})/3$ internal clock cycles. The two proposed architectures exploits the parallelism among four sub-band transforms in lifting-based 2-D DWT. The proposed work has been discussed in terms of hardware complexity, critical path and registers, and the comparison results reveal that the data broadcast architecture exhibits around 60% reduction in adders and 38% reduction in usage of registers. The resource sharing architecture shows around 50% reduction in adders and 13% reduction in usage of registers. In addition, the proposed design exhibits a reasonable reduction in the power and speed efficiency for ASIC implementation.

© 2015 AENSI Publisher All rights reserved.

To Cite This Article: S.Senthil Kumar and R. Radhakrishnan., High Throughput Reconfigurable Resource Sharing Parallel Architecture for a 2D Lifting Based DWT. *Aust. J. Basic & Appl. Sci.*, 9(20): 345-357, 2015

INTRODUCTION

The two-dimensional discrete wavelet transform (2-D DWT) has been widely used in many image compression techniques, and it has been adopted to be an ingredient in many image compression standards, such as JPEG2000 (Christopoulos, 2000). Moreover, the DWT can decompose the signals into different sub-bands with both time and frequency information and facilitate to achieve a high compression ratio. Image compression technique based on 2-D discrete wavelet transform (DWT) has already gained superiority over traditional JPEG based discrete cosine transform and is standardized in forms like JPEG2000 (Lewis and Knowles, 1992). The discrete wavelet transform (DWT) has been widely employed in many signal analysis, image processing, image compression applications (Shapiro, 1993; Taubman, 2000). DWT is recommended to be used in video coding standards such as H.261-3, MPEG1-2,4 by rendering the quality features like better peak signal-to-noise ratio (PSNR) and the absence of blocky artifacts in low bit rates. The lifting scheme for wavelet decomposition has a plenty of useful properties like symmetric

forward and inverse transform, in-place computation and integer-to-integer wavelet transform. The interest in discrete wavelet transform (DWT) is growing tremendously in recent years due to its adoptions in JPEG2000 and MPEG4. The multi-resolution representation derived from DWT time-frequency decomposition demonstrates extraordinary advantages in signal analysis and compression, widely employed in image processing, communication and robotics. The full-frame nature of DWT de-correlates the image over a larger scale and eliminates blocking artifacts at high compression ratios, which is a major shortcoming of block-based transformation. While extending the 1-D DWT module to the 2-D DWT architecture, the memory issue is the most important design consideration, as 2-D DWT requires large amount of data access and storage. The design trade-off mainly originates from the frame memory access bandwidth and the internal buffer size. The frame memory is usually off-chip, because the external frame memory access would consume enormous power and results in wastage of more system memory bandwidth. As the cache is used to reduce the main memory access in the general processor architectures, the internal buffer is

Corresponding Author: S.Senthil Kumar, Anna University, SVS College of Engineering, ECE Dept, Associate Professor, Coimbatore, India.
E-mail: sentheyan@yahoo.co.in

employed to reduce the frame memory access of 2-D DWT architecture. However, the internal buffer would occupy much die area. Various 2-D DWT architectures using different memory structures have been proposed in this vicinity.

MATERIALS AND METHODS

Lifting DWT:

The lifting scheme has been developed as a flexible tool, which is suitable for constructing the second generation wavelet. It consists of three basic operation stages: splitting, predicting and updating. The 1D lifting scheme of the wavelet filter computation for a signal has been discussed in (Lewis and Knowles, 1991; Parhi and Nishitani, 1993):

Split step:

The signal is split into even and odd points, because the maximum correlation between the adjacent pixels can be utilized for the next predict step.

Predict step:

In this step, even samples are multiplied by the predict factor and then the results are added to the odd samples to generate the detailed coefficients.

$$\text{Predict P1: } D(i) = X_{odd}(i) + \alpha(X_{even}(i) + X_{even}(i+1)) \quad (3)$$

$$\text{Update U1: } S(i) = X_{even}(i) + \beta(D(i-1) + D(i)) \quad (4)$$

$$\text{Predict P2: } Y_H(i) = D(i) + \gamma[S(i) + S(i+1)] \quad (5)$$

$$\text{Update U2: } Y_L(i) = S(i) + \lambda[Y_H(i-1) + Y_H(i)] \quad (6)$$

Scaling step:

This operation is performed to obtain the approximate and the detailed value coefficients of the DWT. The value of α , β , γ and λ are obtained which will be a real number.

$$Y_H(i) = KY_H(i) \quad (7)$$

$$Y_L(i) = (1/K) Y_L(i) \quad (8)$$

The lifting-based computation involves lesser number of multipliers, adders and storage elements compared to the convolution-based algorithm (C. Chakrabarti and M. Vishwanath, 1995). The attributes mentioned above make the lifting scheme more appropriate for low-complexity hardware implementation of DWT. The overall hardware complexity of the 2-D DWT structures could be broadly divided into two components: the arithmetic component and memory component (Yu and Chen, 1997). The arithmetic component comprises of multipliers and adders, and the memory component comprises of transposition-buffer, temporal-buffer and the frame-buffer. Transposition and temporal buffers are usually on-chip, while frame-buffer is located off-chip because of its larger size. The size of

Update step:

In this step, the detailed coefficients computed by the predict step are multiplied by the update factors and then the results are added to the even samples to get the coarse coefficients.

Split Step:

The input samples $X(n)$ is delayed by a unit delay so as to provide the odd and even points, and the same is passed to a separate FIFO with multi read and write option. The splitted samples are shown in equation (1) and (2) respectively.

$$X_{even} \leftarrow X(2i) \quad (1)$$

$$X_{odd} \leftarrow X(2i+1) \quad (2)$$

Lifting step:

The Lifting step consists of two major computations: a) Predict and b) Update. The Predict operation makes the average of even samples and adds up with the odd samples in order to predict the next samples. The Update operation finds the difference between predicted and the actual sample value. The operations supported by these two steps are mathematically represented from equations 3 to 6.

the on-chip and off-chip memory greatly affects the speed and power performance of the 2-D DWT structure. The lifting scheme (Parhi and Nishitani, 1993) has been widely used to reduce the required multiplications and additions by exploring the relation of low pass and high pass filters spatially. According to C. Chakrabarti and M. Vishwanath, any DWT filter bank of perfect reconstruction can be decomposed into a finite sequence of lifting steps. And, this decomposition corresponds to a factorization for the polyphase matrix of the target wavelet filter into a sequence of alternating upper and lower triangular matrices and a constant diagonal matrix. Most of the lifting-based architectures in the literature are implemented with the above lifting factorization directly (D. Taubman, 2000; Liao *et al.*, 2004).

Although the lifting scheme could offer many advantages, such as fewer arithmetic operations and in-place implementation, the potentially longer critical path is a drawback for hardware implementation. Here, the timing crisis has been discussed in detail and it is addressed by using pipelining technique.

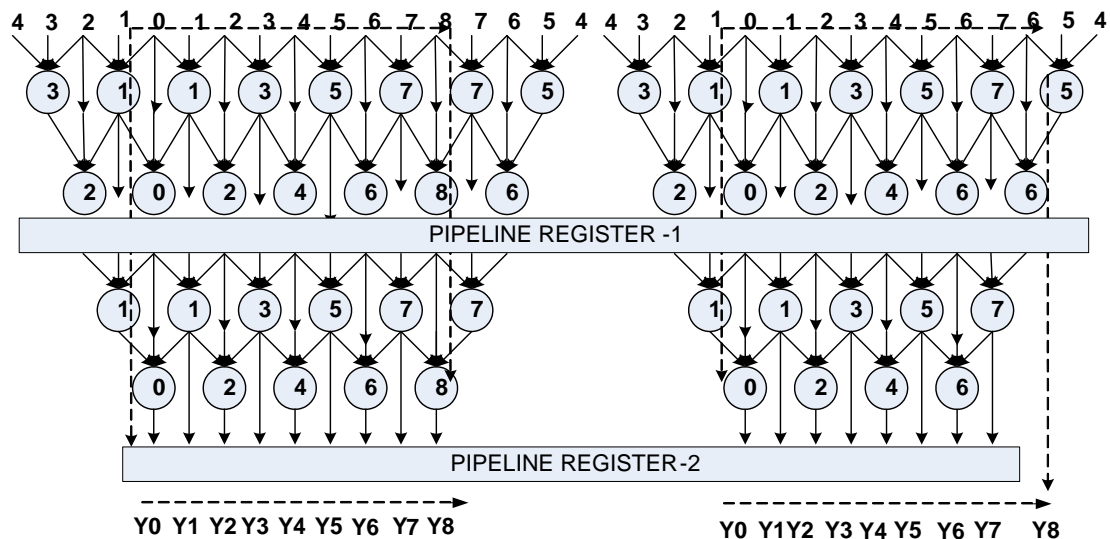


Fig. 1: Data Flow diagram for the Pipelined architecture of 2D DWT

This work tries to explore the possibility of computation reduction by applying pipelining and parallel processing technique to increase the speed and to reduce the hardware complexity. The idea of two stage pipelining is applied to the 9/7 filter as shown in figure 1, which clearly shows the data flow of the architecture. The registers included will reduce the critical path and thereby increase the speed of the DWT architecture. In trade-off to this, there will be a slight increase in register usage. But, the area is not a design constraint compared to the speed of the architecture.

Previous Works:

Many works have been reported in this area of research, and few of the efficient architectures have been discussed in the following section. Andra *et al.*, 2002 proposed four-processor architecture for 2-D DWT by classifying the wavelet filters into two cases: two-matrix (2M) lifting factorization and four-matrix (4M) lifting factorization. These architectures are block-based implementation of 2-D DWT, which require large embedded memory and have long output latency.

Dillen *et al.*, 2003 proposed a combined line-based architecture for the 5-3 and 9-7 wavelet transform by using lifting scheme, but with inefficient hardware utilization, and Liao *et al.*, 2004 proposed a 2-D dual scan architecture, but their methods required two lines of data samples simultaneously for forward 2-D DWT. Liao *et al.*, 2004 also proposed another lifting-based 2-D recursive architecture, which performed all stages of decomposition interleaved that results in inefficient hardware utilization and complicated control circuitry. Barua *et al.*, 2005 proposed a hybrid of level-by-level and line-based architecture for 2-D DWT, in which the image is scanned into the row processor in a raster format. In this paper, a novel

architecture for lifting-based 2-D DWT has been proposed, in which the image is scanned into the row processor in a raster format as in Barua *et al.*, 2005.

Pingping Yu *et al.*, 2012 discussed about the efficient pipeline and parallel architecture design, which is used to speed up the process and achieve better hardware utilization. They have adopted time division multiplexed (TDM) design to realize the prediction step and update step using the same architecture. It also exploited the embedded mirror symmetric boundary extension technique to optimize the architecture for 1-D DWT. After a vast literature survey being made, this research finds a scope to work in the direction of usage of data broadcast structure for reducing the latency and increasing the frequency, and also explores the possibility of sharing the resources by using the idea of re-configurability so as to optimize the resource utilization and reducing the power consumption. In addition, this also explores the incorporation of pipelining and parallel processing in the proposed architecture.

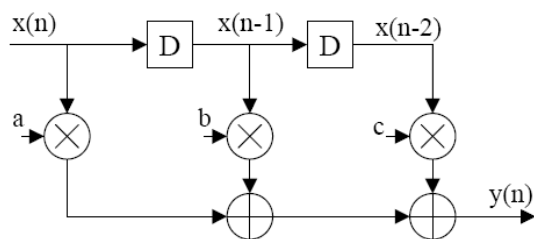
Proposed Architectures:

This work tries to develop two different architectures for 2D DWT 9/7 filter. The first architecture uses data broadcast structure and the second uses the resource sharing technique. The idea of embedded decimation technique is being used for the resource sharing architecture.

The control circuitry for the data flow uses time division multiplexing, which is being used for prediction and updating step and thereby reducing the size of the architecture. Parallel and pipelining techniques have been used for increasing the speed. Row and column DWT modules work independently. The data and the four coefficients of 9/7 filter is passed to the data broadcast mode so that there will not be any internal delay in the architecture. The proposed architecture provides a solution for a 3

level 2D architecture. The update step is done to have the coarse coefficient. Prediction and updating is being carried out in different clock cycles so as to reduce the power and latency of the system. It is executed in row-wise order followed by column. Multiplexers can ensure the re-use of the hardware resource and the samples join the associated computation according to the timing plan. The salient features of the architectures are,

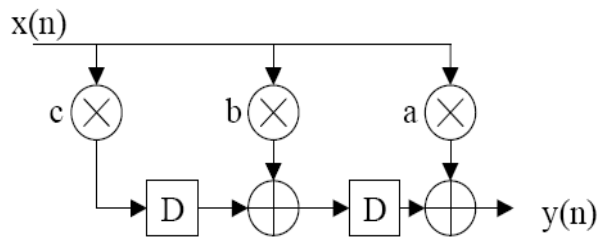
i) Dual edge triggered clocking



2(a) Regular FIR filter

- ii) Resource sharing (static and dynamic part)
- iii) Pipelining and parallel processing
- iv) Proposed VLSI structure for row and column DWT
- v) Folded Memory usage.

The clocking mechanism used for the design of the proposed model is the dual edge trigger clocking, which can initiate two parallel operations of the architecture, one in the positive edge and the other in the negative edge of the clocking.



2(b) Data broadcast structure

Fig. 2: Illustration of Data broadcast architecture (Keshab Parhi, 1993)

This naturally reduces the clock frequency by half and thereby it also reduces the power consumption of the architecture, since power dissipation is directly proportional to frequency. In other way, this also reduces the clock generation complexity to greater extent.

Proposed Data Broadcast Architecture (DBA):

Row DWT architecture:

The idea of data broadcast structure is depicted in figure 2. In figure 2(a) the input samples are passed through the delay element and then fed to the computation units like adder and multiplier, thereby the input itself is delayed to the computation which will further reduce the speed and increase the latency. To solve this issue, the technique of data broadcast structure introduced by Keshab Parhi, 1993 is shown in figure 2(b), where the data samples are fed to the computational block without any delay and thereby it reduces the critical path timing by the reduction of one adder in the critical path and increases the speed. The proposed architecture is optimized in terms of processing speed, as illustrated in figure 3. The multiplication is optimized by using shifting and adding operation. In this way, the row processor consists of six registers, five multiplexers, one adder and one shifting adder. All the hardware resources of the row processor can be time-multiplexed. One single line is calculated at a time.

When a lifting step is performed, initially two consecutive even-numbered samples are added and multiplied with the corresponding lifting coefficient before adding to the middle odd-numbered sample. That means one pixel data is encoded in one clock. This reduces the storage cells. In row DWT module, the input lines are partitioned into even and odd samples (which need two parallel row DWT units). Embedded mirror symmetric boundary data extension algorithm is implemented by using two multiplexers controlled by signals sel0, sel1, which results in significant reduction in the amount of internal storage and the access times of the external memory. The control signals (sel0, sel1) of multiplexers and the corresponding model of the lifting-scheme are coordinated by an FSM. Time-multiplexing row processor is implemented by conducting the predict step in even clocks and the update step in odd clocks. The control signals sel1 and sel0 of the multiplexers are generated by a counter. The row processor is optimized in pipelined way, and the samples are encoded continuously as the samples are the inputs. Hardware utilization reaches approximately 100%, and the control logic is simple. The input data flows through the proposed module of 9/7DWT and the results for rows with 8 samples are pushed out, where H_i (L_i) represents the i^{th} high-pass (low-pass) output.

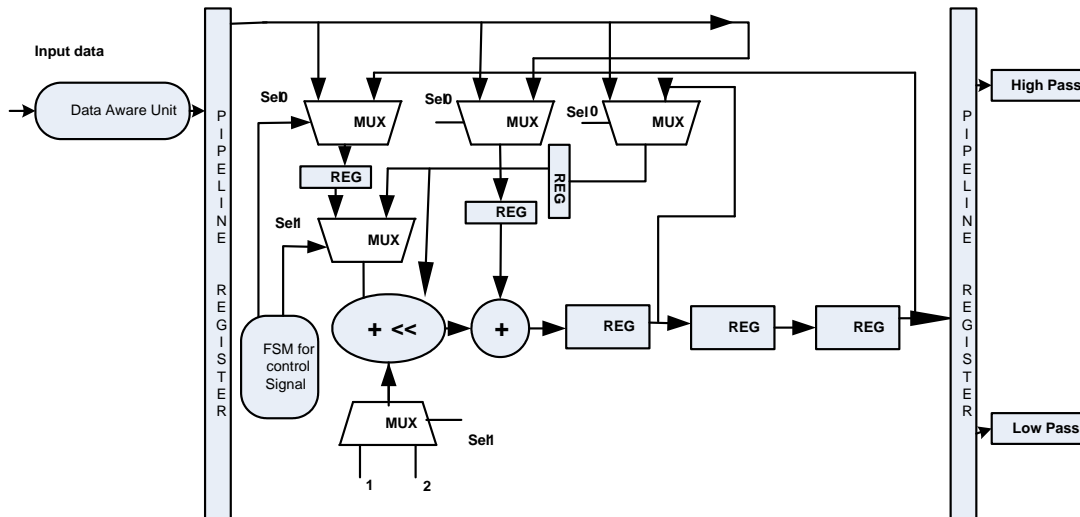


Fig. 3: Proposed Transposed architecture for ROW DWT architecture.

The input samples are passed through the data aware module (senthilkumar, 2014) which will look into the data correlation on the input. Inter and intra data correlations are taken into consideration for the

reduction of computation. The control signals required for the selective skipping of operation is being taken care by the correlation computation unit and the timing control unit in FSM.

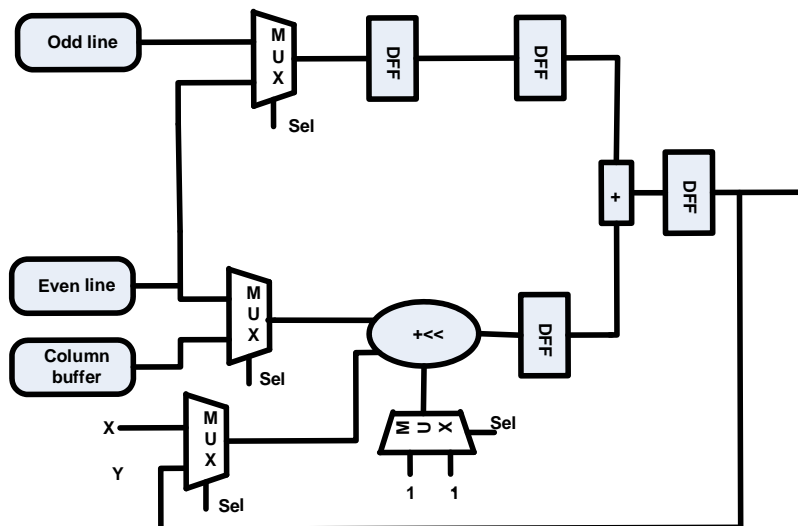


Fig. 4: Data broadcast architecture for column DWT architecture

Proposed Column DWT Module:

The proposed Column DWT architecture is shown in figure 4. In order to reduce the system latency, the column DWT has to be executed in the row-wise order. The input data are stored in even line buffer, and odd line buffer is naturally separated into even samples and odd samples along the column. Embedded mirror symmetric boundary data extension algorithm has also been implemented in the Column DWT module. There are four multiplexers employed to control the steps (1 represents prediction step and 0 represents update step). Multiplexers ensure the re-use of hardware resource and those samples joining the associated computation according to the timing plan. The

column DWT module begins to calculate the samples after computing the first two lines in row DWT. Initially, multiplexers are set to 1 and the column processor conducts prediction step. The results of the prediction step are taken as output and the same is stored in the column buffer simultaneously. Next, the multiplexers are set to 0, and column processor conducts the update step, then the results are exported directly. The column processor is also optimized in pipelined manner to increase the speed of the wavelet transform. The data flow of the column DWT is similar to that of the row DWT.

Resource sharing architecture (RSA) for the (9, 7) Filter:

This architecture is developed with an idea of reconfigurability. It consists of two blocks: the static and the dynamic blocks as shown in figure 5. The static block is the dedicated block for row/column computation, while the dynamic block is the reconfigurable block which can route for either of the two computations. Few of the computational resources are made dynamic, namely adders and registers. The data aware property (senthilkumar, 2014) is also included in the architecture so as to reduce the computation. A separate FSM block has been added to route the control signal required. This 2-D architecture is called the fast architecture, which can perform J levels of decomposition for $N \times N$ image. Moreover, another efficient generic line-based 2-D architecture has been proposed by exploiting the parallelism among four sub-band transforms in lifting-based 2-D DWT. The throughput rate of this architecture is increased by two times when compared to the former 2-D architecture with lesser computational resources.

In this architecture, two efficient line-based architectures for 2-D DWT have been proposed, where the image is scanned into the row processor in a raster format. An embedded decimation technique is exploited to optimize the 1-D DWT architecture, which is designed to receive an input and generate an output with low and high-frequency components of original data being available alternately. Based on this 1-D DWT architecture, a novel line-based architecture for 2-D DWT has been further proposed by employing pipeline and parallel techniques, which is mainly composed of two horizontal filter modules and one vertical filter module, working in

pipeline and parallel fashion with 100% hardware utilization.

It has been found that the critical path latency of the proposed architecture in figure 6 is reduced from that specified in Pingping Yu, 2012, by optimizing the order of addition operations. T_m , T_s and T_a denote the latencies of multiplication, shifting and addition operations respectively. Pipelining can be employed as shown in figure 1, to further reduce the critical path, while the total number of registers required is increased from 7 to 23. Figure 6 and figure 8 demonstrates the two architectures for forward and inverse DWT operations. The resourcing sharing based FDWT and IDWT have exactly the same architecture as the order of primary lifting, dual lifting and scale normalization, which is also different from that based on the conventional lifting scheme. It indicates that the proposed reconfigurable architecture forms an efficient alternative for conventional FDWT and IDWT.

The Euclidean algorithm factorizes the poly-phase matrix of a DWT filter as the multiplication of alternative upper and lower triangular matrices as well as a diagonal matrix. The poly-phase matrix of wavelet transform can be represented as in equation 9, 10, 11 and 12. Every finite impulse response filter wavelet can be factored into lifting steps, and the lifting strategy is highly a non-unique process. Let $L[n]$ and $H[n]$ denote the low-pass and high-pass analysis filters respectively, then the corresponding decomposition and reconstruction poly-phase matrices are denoted as $P(z)$ and $\bar{P}(z)$.

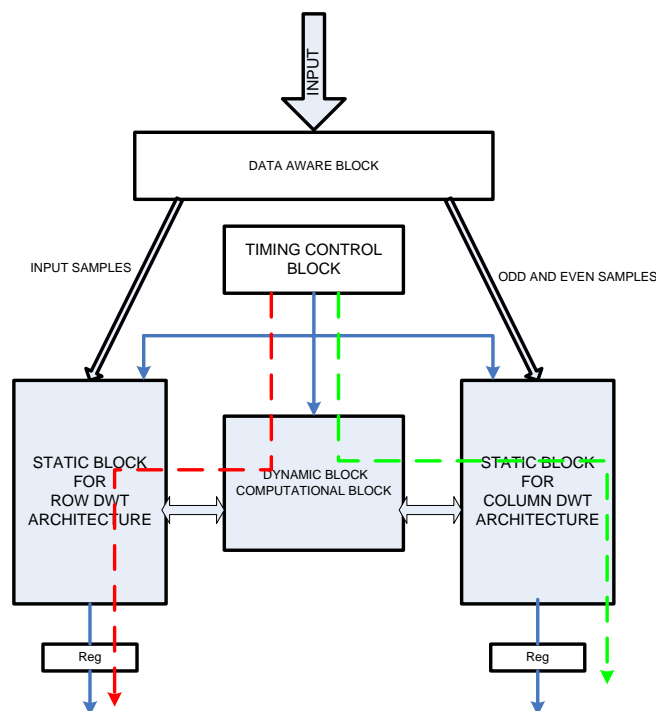


Fig. 5: Proposed Resource sharing reconfigurable architecture

Performance comparison of these proposed architectures are illustrated in tables II & III. If the pipeline technique is employed in the implementation of multipliers, then the critical path of architecture will be further reduced. The

comparison results demonstrate that the proposed architectures are found to be more efficient than the flipping structure presented in XuguangLan *et al.*, 2005, in reducing critical path and/or the number of registers.

$$\begin{aligned}
 \mathbf{P}(z) &= \begin{bmatrix} 1 & a(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ b(1+z) & 1 \end{bmatrix} \begin{bmatrix} 1 & c(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ d(1+z) & 1 \end{bmatrix} \begin{bmatrix} k & 0 \\ 0 & \frac{1}{k} \end{bmatrix} \\
 &= \begin{bmatrix} \frac{1}{a} & 1+z^{-1} \\ 0 & \frac{1}{a} \end{bmatrix} \begin{bmatrix} \frac{1}{b} & 0 \\ (1+z) & \frac{1}{b} \end{bmatrix} \begin{bmatrix} \frac{1}{c} & 1+z^{-1} \\ 0 & \frac{1}{c} \end{bmatrix} \times \begin{bmatrix} \frac{1}{d} & 0 \\ 1+z & \frac{1}{d} \end{bmatrix} \begin{bmatrix} abcdK & 0 \\ 0 & \frac{abcd}{k} \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 1+z^{-1} \\ 0 & \frac{1}{a} \end{bmatrix} \begin{bmatrix} \frac{1}{ab} & 0 \\ 1+z & 1 \end{bmatrix} \begin{bmatrix} 1 & 1+z^{-1} \\ 0 & \frac{1}{bc} \end{bmatrix} \times \begin{bmatrix} \frac{1}{cd} & 0 \\ 1+z & 1 \end{bmatrix} \begin{bmatrix} abcdK & 0 \\ 0 & \frac{abc}{k} \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 1+z^{-1} \\ 0 & a' \end{bmatrix} \begin{bmatrix} b' & 0 \\ 1+z & 1 \end{bmatrix} \begin{bmatrix} 1 & 1+z^{-1} \\ 0 & c' \end{bmatrix} \times \begin{bmatrix} d' & 0 \\ 1+z & 1 \end{bmatrix} \begin{bmatrix} k_1 & 0 \\ 0 & k_0 \end{bmatrix} \tag{9}
 \end{aligned}$$

The term $x[n]$ has been used to represent the original input sequence, and $x[2n]$, $(x[2n+1])$ to represent even (odd) indexed samples. The intermediate values computed during lifting are denoted as $H^{(m)}[n]$ and $L^{(m)}[n]$ ($m=1,2$), and the low and high-frequency coefficients are expressed as the square $L[n]$ and $H[n]$, respectively. Hence, following the implementation for forward (9, 7)

DWT, the mathematical notations can be written as follows:

$$L^{(0)}[n] = x[2n], H^{(0)}[n] = x[2n+1] \tag{10a}$$

$$H^{(1)}[n] = a' * H^{(0)}[n] + L^{(0)}[n] + L^{(0)}[n+1] \tag{10b}$$

$$L^{(1)}[n] = b' * L^{(0)}[n] + H^{(1)}[n] + H^{(1)}[n-1] \tag{10c}$$

$$H^{(2)}[n] = c' * H^{(1)}[n] + L^{(1)}[n] + L^{(1)}[n+1] \tag{10d}$$

$$L^{(2)}[n] = d' * L^{(1)}[n] + H^{(2)}[n] + H^{(2)}[n-1] \tag{10e}$$

$$H[n] = k_0 * H^{(2)}[n], L[n] = k_1 * L^{(2)}[n]. \tag{10f}$$

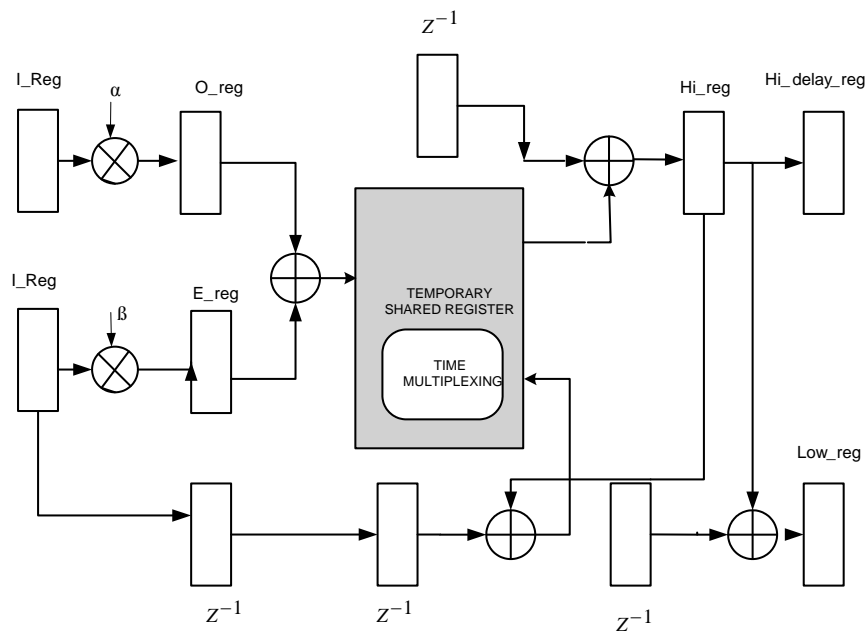


Fig. 6: Forward resource sharing architecture

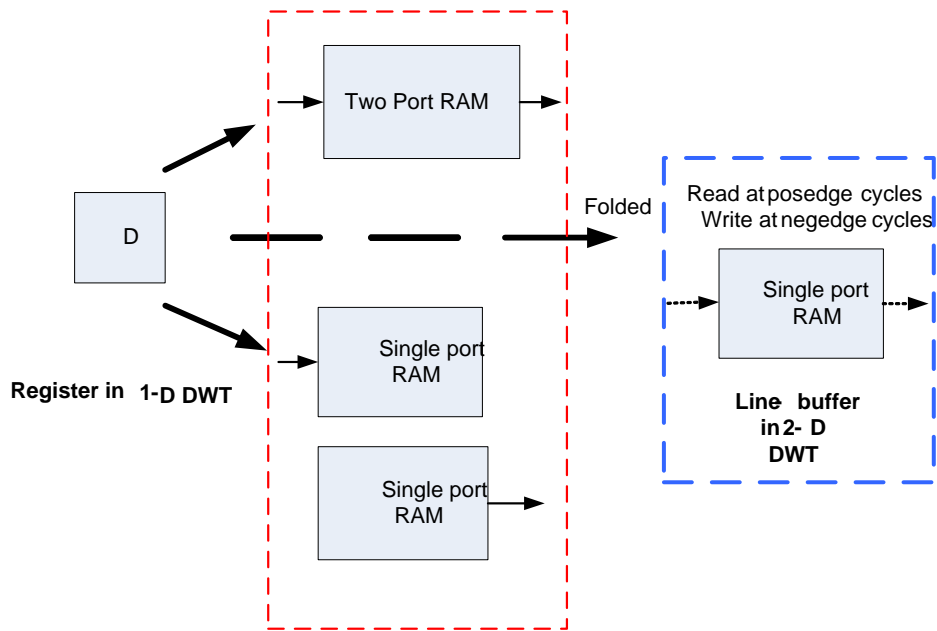


Fig. 7: Folded RAM which is used in shared memory

Temporary shared memory is the module which is mostly shared for the memory usage, its read and write operations play a major role in determining the speed and power consumption of the architecture. There are several memories like single and two ports, but this architecture uses single port RAM with read and write being performed in positive edge and negative edge of the clock in the same clock cycle. This dual edge triggering reduces the power to a great extent. This RAM is called as Folded RAM as outlined in figure 7.

Inverse Parallel Lifting 9/7:

Figure 8 shows the architecture for the inverse operation of DWT. This architecture also uses a shared memory concept with a shared adder, whose computation cycle is decided by the timing diagram and FSM of the circuit. The final Low and High pass filter output is taken from the registers. The expressions guiding the inverse operation are shown in equations 11 and 12.

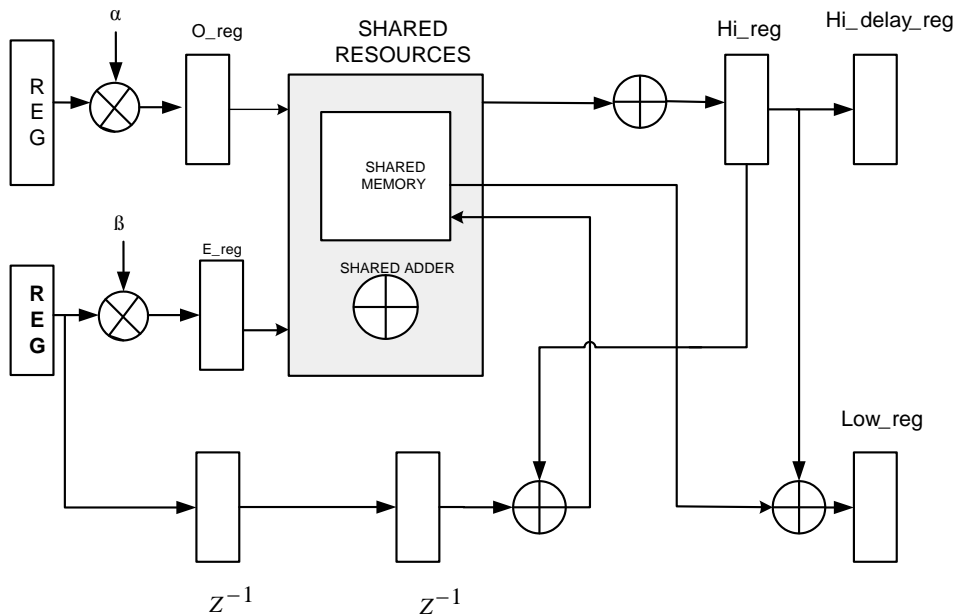


Fig. 8: Inverse Resource sharing architecture

$$\begin{aligned}
 \bar{P}(z) &= \begin{bmatrix} 1 & 0 \\ -a(1+z) & 1 \end{bmatrix} \begin{bmatrix} 1 & -b(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ -c(1+z) & 1 \end{bmatrix} \begin{bmatrix} 1 & -d(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} \frac{1}{k} & 0 \\ 0 & k \end{bmatrix} \\
 &= \begin{bmatrix} Kabcd & 0 \\ 0 & Kabcd \end{bmatrix} \begin{bmatrix} \frac{1}{a} & 0 \\ -(1+z) & \frac{1}{a} \end{bmatrix} \times \begin{bmatrix} \frac{1}{b} & -(1+z^{-1}) \\ 0 & \frac{1}{b} \end{bmatrix} \begin{bmatrix} \frac{1}{c} & 0 \\ -(1+z) & \frac{1}{c} \end{bmatrix} \\
 &\times \begin{bmatrix} \frac{1}{d} & -(1+z^{-1}) \\ 0 & \frac{1}{d} \end{bmatrix} \begin{bmatrix} \frac{1}{k^2} & 0 \\ 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} Kbcd & 0 \\ 0 & Kabcd \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -(1+z) & \frac{1}{ab} \end{bmatrix} \times \begin{bmatrix} \frac{1}{bc} & -(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -(1+z) & \frac{1}{cd} \end{bmatrix} \\
 &\times \begin{bmatrix} \frac{1}{ak^2} & -(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} k'_1 & 0 \\ 0 & k'_0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -(1+z) & a'' \end{bmatrix} \times \begin{bmatrix} b'' & -(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -(1+z) & c'' \end{bmatrix} \times \begin{bmatrix} d'' & -(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \tag{11}
 \end{aligned}$$

$$\begin{aligned}
 L^{(2)}[n] &= d'' * L[n] - (H[n] + H[n-1]) \\
 H^{(2)}[n] &= c'' * H[n] - (L^{(2)}[n] + L^{(2)}[n+1]) \tag{12a}
 \end{aligned}$$

$$\begin{aligned}
 L^{(1)}[n] &= b'' * L^{(2)}[n] - (H^{(2)}[n] + H^{(2)}[n-1]) \\
 H^{(1)}[n] &= a'' * H^{(2)}[n] - (L^{(1)}[n] + L^{(1)}[n+1]) \tag{12b}
 \end{aligned}$$

$$H^{(0)}[n] = K'_0 * H^{(1)}[n], L^{(0)}[n] = k'_1 * L^{(1)}[n] \tag{12c}$$

$$y[2n] = L^{(0)}[n], y[2n+1] = H^{(0)}[n]. \tag{12d}$$

implementation is done using cyclone II family with EP2C20F484C7 device. Figure 9 shows the NCSIM simulation results of the proposed data broadcast architecture followed by the figure 10 which shows the RTL synthesized diagram for 45nm technology from TSMC. Figures 11 and 12 shows the functional simulation results and RTL synthesized results of resource sharing architecture respectively. Figure 15 shows the complete ASIC chip layout after the completion of backend using SOC Encounter. Partition, placement and routing are done using cadence encounter tool. The FPGA implementation is done using ALTERA tool targeting for DE-II education board. Figures 13 and 14 show the synthesized and simulation results of resource sharing architecture.

RESULT AND DISCUSSION

The circuit is designed and modeled with VERILOG, simulated using MODEL SIM/ NC SIM of cadence. The synthesis of the design is done using Encounter ® RTL compiler with 45 nm technology with the operating condition of typical balanced tree and wire load model to be segmented. The synthesis shows a positive slack of 966ps. Resource sharing design has the slack of 230ps. The FPGA

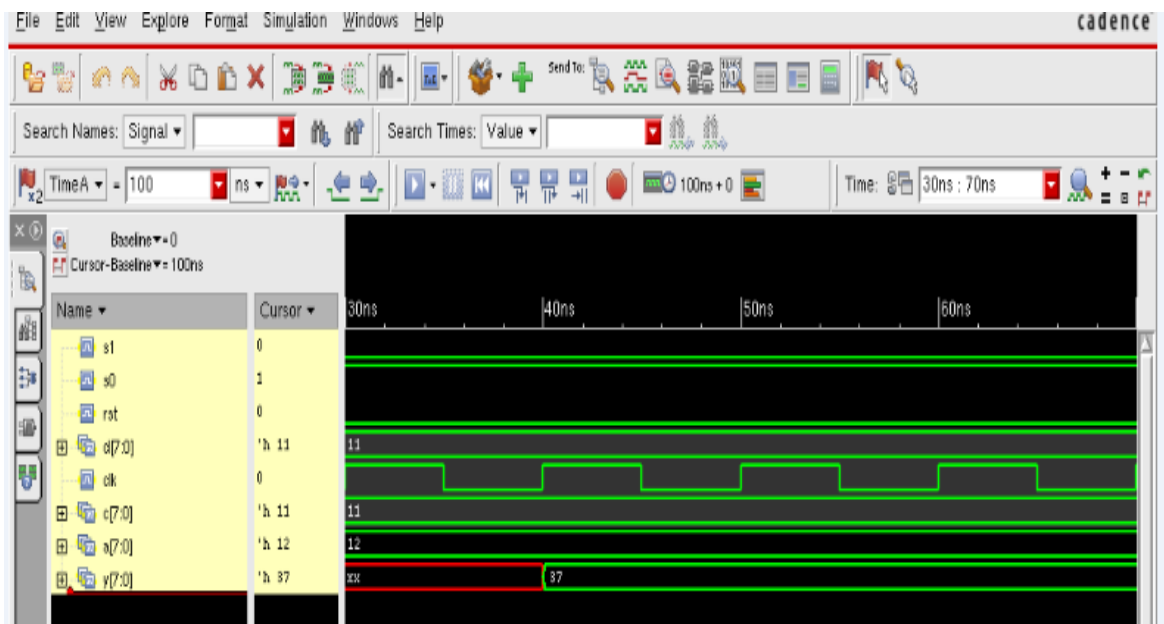


Fig. 9: Simulation results of data broadcast structure.

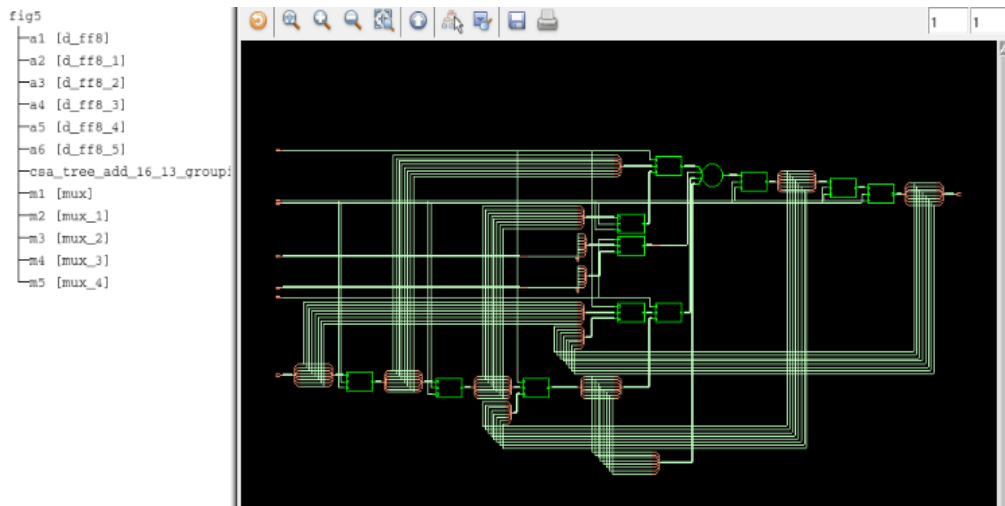


Fig. 10: Synthesized RTL results of Data broadcast structure.

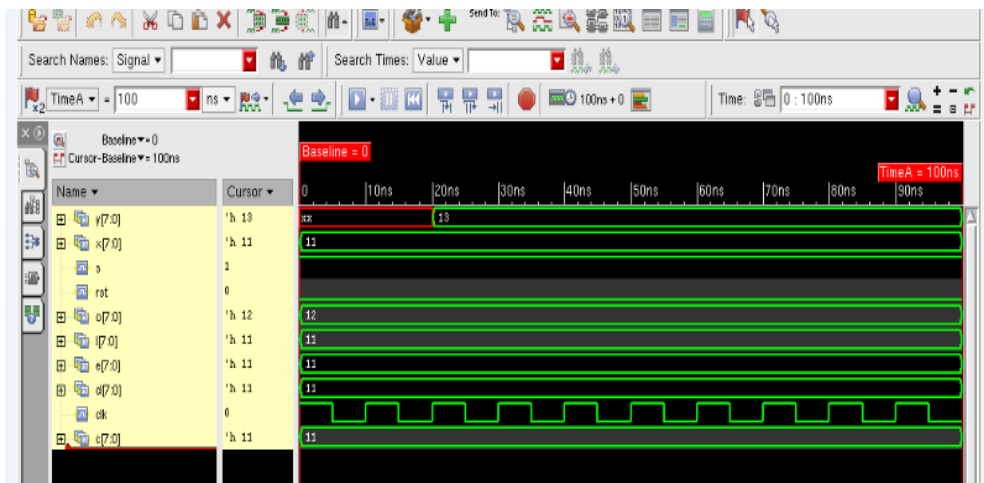


Fig. 11: Simulation results of Resource sharing architecture.

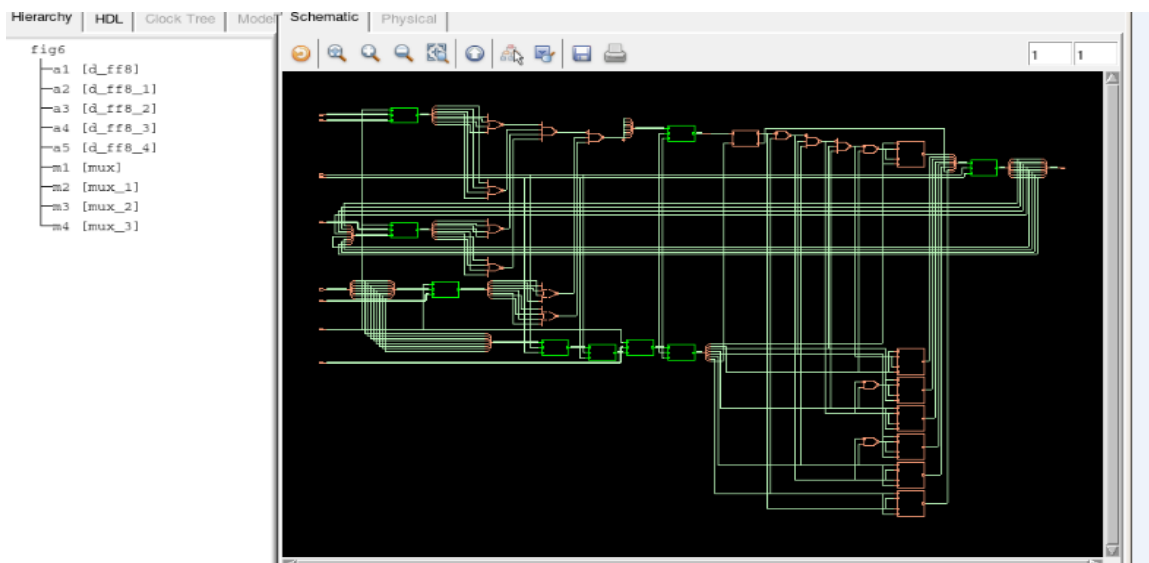


Fig. 12: Synthesized RTL results of Resource sharing architecture.

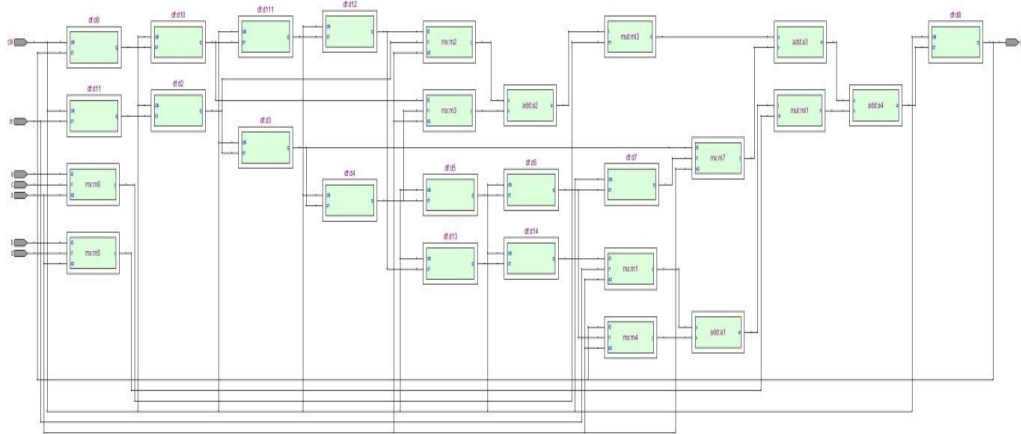


Fig. 13: FPGA Synthesized RTL results of Resource sharing architecture.

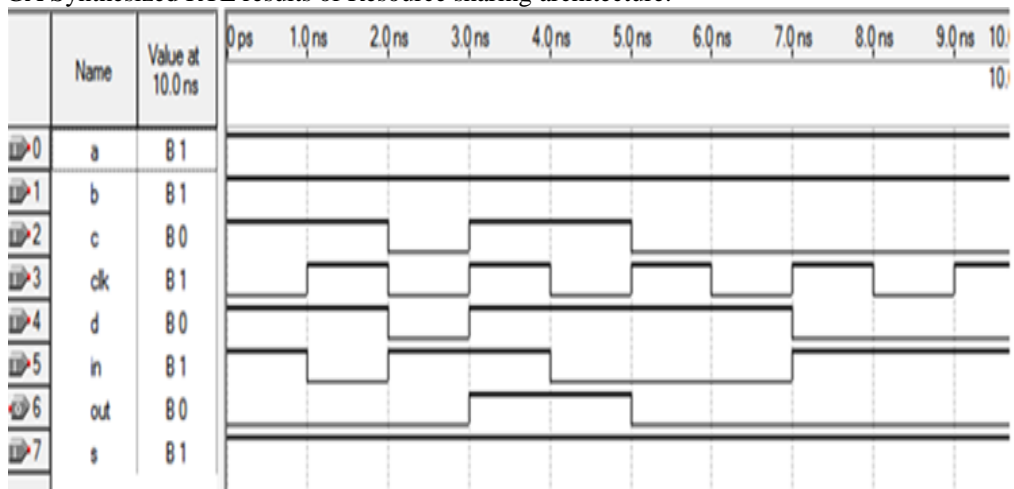


Fig. 14: FPGA Simulation results of Resource sharing architecture.

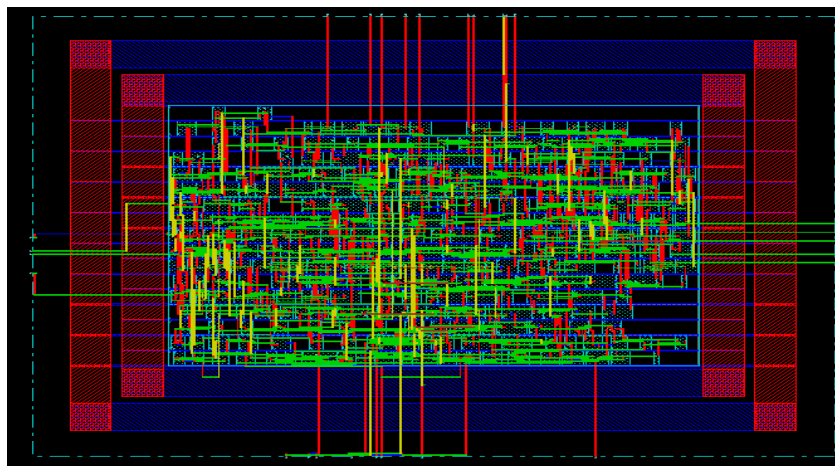


Fig. 15: Complete ASIC chip after backend

Comparison of Results:

The comparison results in table I clearly reveals that the data broadcast architecture shows nearly 75% reduction in adder usage compared to Chin-Fa Hsieh *et al.*, 2007; Chen, 2004 and 50% less compared to XuguangLan *et al.*, 2005. The register

usage is reduced by 38% compared to Chin-Fa Hsieh *et al.*, 2007 and Chen, 2004 without pipeline and if 2 stage pipelines are employed, the register consumption increases by 120% but the speed gets almost double. The critical path in table II shows that ($T_m + 5 T_a$) is reduced to one shifter and one adder.

The Resource sharing architecture results in table I shows around 50% reduction in adder usage compared to Chin-Fa Hsieh *et al.*, 2007; Chen, 2004 and 13% reduction in registers usage compared to XuguangLan *et al.*, 2005. Since this architecture uses the shared folded memory, the speed has been increased. Table II shows only one multiplier in the critical path. Since there is a reduction in the computation elements in the critical path it is expected to increase the speed and throughput of the system.

Table III gives the synthesized results of both ASIC and FPGA platforms. The proposed design is

synthesized using 45nm technology of TSMC, whereas the compared results are synthesized using 90nm technology, though it is not the right way to make comparison between different technologies, this attempt has been carried out to have an idea of the area, power and frequency. The results show that ASIC implementation of data broadcast structure has nearly 80% reduction in area, 100% reduction in power with almost the frequency gets doubled. The resource sharing architecture shows 85% reduction in area, 90% reduction in power with speed increasing nearly to 5 times.

Table I: Usage of components in a processing element

Architecture	Multiplexer	Adder	Multiplier	Register
XuguangLan <i>et al.</i> , 2005	4	4	3	27
Chin-Fa Hsieh <i>et al.</i> , 2007	4	8	3	8
Chen, 2004	4	8	2	8
Peng Cao <i>et al.</i> , 2007	2	4	3	12
Proposed DBSWP [#]	4	2	1 shifter	5
Proposed DBSP2 [@]	5	2	1 shifter	21
Proposed RSAWP [*]	--	4	2	7
Proposed RSAP2 ^{&}	--	4	2	23

Table II: Usage of gates in critical path

Architecture	Multipliers	Adders	Critical Path	Registers
Pingping Yu <i>et al.</i> , 2012 + no pipeline	6	8	$T_m + 5T_a$	4
Pingping Yu <i>et al.</i> , 2012 + 5 stages pipeline	6	8	T_m	12
Proposed DBSWP [#]	1 shifter	2	$T_s + 1T_a$	5
Proposed DBSP2 [@]	1 shifter	2	T_s	21
Proposed RSAWP [*]	2	4	T_m	7
Proposed RSAP2 ^{&}	2	4	T_m	23

DBAWP[#] - Data Broad Cast Architecture without pipelining; DBAP2[@] - Data Broad Cast Architecture with 2 stage pipelining
 RSAWP^{*} - Resource sharing architecture without pipelining; RSAP2[&] - Resource sharing architecture with 2 stage pipelining
 T_s - Timing of a shift register; T_m - Timing of a multiplier

Table III: Design Parameter comparison of the proposed architecture with the existing method

Design	ASIC Implementation			FPGA Implementation		
	Area (mm^2)	Power (mW)	Frequency (MHz)	Area (Logic Elements)	Power (mW)	Timing (ns)
Proposed DBSWP [#]	0.45	1.3	323	2184	71.75	11.44
Proposed RSAWP [*]	0.31	2.5	501	1982	122.08	8.50
Yeong-Kang Lai <i>et al.</i> , 2009 [#]	2.16	102	100	NA	NA	NA
Basant Kumar <i>et al.</i> , 2013 [#]	2.13	15	40	NA	NA	NA

DBAWP[#] - Data Broad Cast Architecture without pipelining; DBAP2[@] - Data Broad Cast Architecture with 2 stage pipelining
 RSAWP^{*} - Resource sharing architecture without pipelining; RSAP2[&] - Resource sharing architecture with 2 stage pipelining
 # -- Synthesized with 90nm Technology; * -- Synthesized with 45nm Technology

Conclusion:

In this research work, two different VLSI architectures for 2D DWT for 9/7 have been proposed and the design is being simulated and synthesized. The data broadcast architecture shows huge resource reduction and the power consumption also has been reduced when compared with the existing architectures. The Resource sharing architecture shows that the speed has been increased by many folds. Apart from this, the proposed architecture uses shared folded memory concepts, and also the techniques of pipelining and time multiplexed parallel processing. It is clearly proved that the two proposed architectures could be employed for image compression DWT applications.

REFERENCES

- Lewis, A.S. and G. Knowles, 1992. "Image compression using the 2-Dwavelet transform," IEEE Trans. Image Process., 1(3): 244-250.
- Shapiro, J.M., 1993. "An embedded hierarchical image coder using zero trees of wavelet coefficients," in Proc. IEEE Data Compression Conf., Snowbird, UT, pp: 214-223.
- Taubman, D., 2000. "High performance scalable image compression with EBCOT," IEEE Trans. Image Process., 9(7): 1158-1170.
- Christopoulos, C., A. Skodras, and T. Ebrahimi, 2000. "The JPEG2000 still image coding system: an overview," IEEE Trans. Consum. Electron., 46(4): 1103-1127.

- Lewis, A.S. and G. Knowles, 1991. "VLSI architecture for 2-D Daubechies wavelet transform without multipliers," *Electron. Lett.*, 27: 171-173.
- Parhi, K.K. and T. Nishitani, 1993. "VLSI architectures for discrete wavelet transforms," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 1(3): 191-202.
- Chakrabarti, C. and M. Vishwanath, 1995. "Efficient realizations of discrete and continuous wavelet transforms: from single chip implementations to mappings on SIMD array computers," *IEEE Trans. Signal Process.*, 43(3): 759-77.
- Yu, C. and S.J. Chen, 1997. "VLSI implementation of 2-D discrete wavelet transform for real-time video signal processing," *IEEE Trans. Consum. Electron.*, 43(4): 1270-1279.
- Andra, K., C. Chakrabarti and T. Acharya, 2002. "A VLSI architecture for lifting-based forward and inverse wavelet transform," *IEEE Trans. Signal Process.*, 50(4): 966-977.
- Dillen, G., B. Georis, J.D. Legat and O. Cantineau, 2003. "Combined line based architecture for the 5-3 and 9-7 wavelet transform of jpeg2000," *IEEE Trans. Circuits Syst. Video Technol.*, 13(9): 944-950.
- Liao, H., M.K. Mandal and B.F. Cockburn, 2004. "Efficient architectures for 1-D and 2-D lifting-based wavelet transforms," *IEEE Trans. Signal Process.*, 52(5): 1315-1326.
- Barua, S., J.E. Carletta, K.A. Kotteriand A.E. Bell, 2005. "An efficient architecture for lifting-based two-dimensional discrete wavelet transform," *Integr. VLSI J.*, 38(3): 341-352.
- Pingping Yu, Suying Yao, Jiangtao Xu, 2012. "An Efficient Architecture for 2-D Lifting-based Discrete Wavelet Transform", *IEEE Transactions on circuits and systems-II*, 59: 3.
- Xuguang Lan, Nanning Zheng and Yuehu Liu, 2005. "Low-power and high speed VLSI architecture for lifting-based forward and inverse wavelet transform," *IEEE Trans. on Consumer Electronics*, 51: 379-385.
- Chin-Fa Hsieh, Tsung-Han Tsai and Chih-Huang Lai, 2007. "Implementation of an efficient DWT using a FPGA on a Real-time Platform," *IEEE, ICICIC, Second International Conference*, pp: 235-235.
- Chen, P.Y., 2004. "VLSI implementation for one-dimensional multilevel lifting based wavelet transform," *IEEE Trans. on Computers*, 53: 386-398.
- Peng Cao, XinGuo and Chao Wang, 2007. "Efficient architecture for two dimensional discrete wavelet transform based on lifting scheme," *IEEE 7th International Conference*, pp: 225-228.
- Yeong-Kang Lai, Lien-Fei Chen and Yui-Chih Shih, 2009. "A High-Performance and Memory-Efficient VLSI Architecture with Parallel Scanning Method for 2-D Lifting-Based Discrete Wavelet Transform" *IEEE Transactions on Consumer Electronics*, 55(2): 400-407.
- Basant Kumar Mohanty and Pramod Kumar Meher, 2013. "Memory-Efficient High-Speed Convolution-based Generic Structure for Multilevel 2-D DWT" *IEEE Transactions on circuits and systems for video technology*, 23(2): 353-363.
- Senthilkumar, S., R. Radhakrishnan, 2014. "Data aware high speed low power architecture using modified lifting scheme for 2 D DWT", *International journal of applied engineering research*, 9: 16725-16736.