High Speed Memory Efficient 3D DTCWT Architecture for Video Salient Object Detection

Suresh Babu D[1, *], Dr. K B Raja[2], Dr. Venugopal K R[3]

[1] *Research Scholar, Dept. of ECE, UVCE,*
Bangalore, Karnataka, India
[2] *Professor & Chairman, Dept. of ECE, UVCE,*
Bangalore, Karnataka, India
[3] *Former Vice Chancellor, Bangalore University*
Bangalore, Karnataka, India
[*] [1] *sureshbd1976@gmail.com* [2] *raja_kb@yahoo.com* [3] *venugopalkr@gmail.com*

Abstract

Computer vision application deals with salient objects in image and video sequences and correlates them with objects in its neighborhood which is similar to the cognitive property of human visual system. The salient objects are detected using the directional selectivity property of the complex wavelets. Memory efficient Distributive architecture is considered for 3D DTCWT for the detection of salient objects in the video sequence. Proposed are the reduced modular DA architecture I and II and also optimized modular DA architecture I and II. Again, for the detection of salient objects in the video sequence using 3D DTCWT, High throughput Systolic Array architecture is considered. Reduced modular SA architecture and optimized modular SA architecture are proposed. The proposed architectures are implemented on SPARTAN 6 FPGA and tested in the XILINX.

Keywords: *DTCWT, Video Saliency, Distributive Arithmetic, Systolic Array FPGA*

1. Introduction

Salient object detection is the process of detecting salient objects in a given scene and segmenting the object considering the accurate boundary of the object of interest. In salient object detection process based on deep learning algorithms, edge features of the object of interest is detected first or the CNN is trained to learn edge features by the first fundamental layers. Wavelets are used with CNNs as pre-processing layer for classification of images, detection of textures and for improving face resolution [16]. Wavelet sub bands are combined or fused prior to classification [17] or wavelet sub bands are used to compute feature vectors for classification [18] or significant features are extracted from wavelet feature for classification [19]. Waking et al. [20] have demonstrated the advantages of processing 3D DWT over 2D DWT for video coding to achieve higher compression as the sub bands capture both spatial and temporal information in the decomposed sub bands helpful in removing redundancy. Encoding of video sequences has been carried out by computing motion estimation and compensation algorithms that are computationally intensive. 3D Dual Tree Complex Wavelet Transforms (DTCWT) [21] have been used as alternative to encode video sequences to capture sub bands that are oriented in different spatiotemporal directions isolating image features and providing information on inherent motions in all directions. 3D DTCWT generates four times as many sub bands compared with 3D DWT (2^m:1 redundancy for every m-dimension decomposition) and hence the challenge in identifying redundant information is four times complex as compared to that of 3D DWT sub bands [22]. Selesnick and Li have reduced the computation complexity by considering only the real sub bands for video coding and have demonstrated perfect reconstruction of video sequences. 3D DWT sub bands are generated considering filter pairs that satisfy Hilbert property, from these DWT sub bands 3D DTCWT real sub bands are derived by linear operations. The directional features in different directions are captured in 28 different high frequency sub bands and the low frequency information is captured in 4 sub bands. Four separate groups of sub bands with each group comprising of 7 sub bands are organized similar to that of 3D DWT sub bands with the advantage that the organized 3D DTCWT sub bands capture more directional features in different orientations. Correlation between 3D DTCWT real valued sub bands has been used to identify to eliminate redundancy and encode the coefficients. Even with only real valued sub bands there is redundancy in information and very few critical coefficients are sufficient to represent the information content in the given input data. Reeves and Kingsbury have proposed noise shaping method [23] to increase sparsity between transform coefficients and insignificant coefficients are removed. It is also observed that the insignificant coefficients have less intensity levels but these insignificant coefficients form coherent regions in every sub band and when removed using noise shaping method constitutes loss of motion vectors. The noise shaping method de-correlates the wavelet coefficients and the number of non-zero coefficients is less than the coefficients of 3D DWT and reconstructing the video data from these coefficients have generated same level of video data as compared with reconstructed data with 3D DWT sub bands[24]. With motion information in 3D DTCWT sub bands Wang et al.[25] have encoded the noise shaped sub bands using bit plane coding exploiting inter redundancy in high frequency sub bands for video coding. The four low pass sub bands coefficients are encoded using 16-bit vector considering 2 x 2 sub blocks in each of the four sub bands and 28 the 28 high pass sub bands are encoded using 28-bit vector with adaptive arithmetic coding. Julia Neumann and Gabriele Steidl [26] in their work have used DTCWT for feature extraction from complex sub bands computing 2-norm of each of the DTCWT sub bands generated from m-level decomposition. Support Vector Machine (SVM) is used for classification of these features. The 3D DTCWT sub bands are organized with the same number of sub bands as that of 3D DWT sub bands, by capturing the six directional orientations from both the real and imaginary sub bands by gradient operations and averaging methods. Deep learning algorithm that processes the directional features captured with SPIHTL algorithm from the reorganized 3D DTCWT sub bands. The computation complexity of 3D DTCWT is four times complex than 3D DWT, for real time detection of salient objects in surveillance applications; there is a need for fast processing algorithms for computing 3D DTCWT sub bands. In this paper, high speed architectures with low power schemes are proposed, implemented on FPGA platform for computing 3D DTCWT sub bands for feature detection in video salient object detection. Section II discusses DTCWT algorithm and 3D DTCWT structure, section III discusses proposed architecture for 3D DTCWT, section IV presents FPGA implementation and results, section V presents conclusion.

2. DTCWT Architecture

The DTCWT sub band computation is similar to DWT with DTCWT employing Hilbert Pair of DWT filters to produce complex wavelet sub bands comprising of real and imaginary parts. The shift invariance property of DTCWT has the advantage of overcoming aliasing loss as reported by Kingsbury [31] in DWT and it is reported by Simoncelli et al. [32] on the interoperability of sub band coefficients. The wavelet filters bandwidth for DTCWT are approximately one octave wide and hence the features such as edges or surfaces are localized in space within the 28 sub bands and are also found to be uniformly spaced at any scale. The optimal characteristics of DTCWT of directional selective property are hence suitable for feature selection and improve image registration process. Fig. 1 shows the 3D DTCWT level-1 structure. The 3D DTCWT structure is shown in Figure 1(a). The first stage is the 2D DTCWT processor comprising of two stages and the third stage is the processor in temporal domain.3D input data comprising of 52 frames with each frame of size 512 x 512 is processed simultaneously by 52 numbers of 2D DTCWT processing unit. Every 2D frame (represented by $S^1(Z_1Z_2)$) is processed row wise first and then column wise in the 2D processor generating four sub bands from each frame. The third stage DTCWT processor processes the sub bands in the z-direction to generate eight sub bands thus representing the first octave bands (shown in Figure 1(c)). Similarly, the 3D DTCWT structure generates eight octaves with each octave comprising of eight sub bands of one low pass band and seven high pass bands. The 2D DTCWT processor comprises of two stage processing unit, the row processing and column processing. The row processing stage consists of four filter banks represented by {Hoa, Hob, H1a & H1b}, each of these filter outputs are further processed by four column filters. 3D DTCWT sub-bands of low pass ($C^1_{1a/b}$) and high pass ($D^m_{1a/b}$) are mathematically represented as in Equations (1) – (4), where A and B are real and imaginary filter coefficients respectively.

$$C^{1}_{1a/b}(Z_1Z_2Z_3) = (2^{j}\downarrow)\left[\left(A^{j}_{a}(Z_1) \pm iA^{j}_{b}(Z_1)\right)\left(A^{j}_{a}(Z_2) + iA^{j}_{b}(Z_2)\right)\left(A^{j}_{a}(Z_3) + iA^{j}_{b}(Z_3)\right)\ S(Z_1Z_2Z_3)\right] \quad (1)$$

$$D^{1}_{1a/b}(Z_1Z_2Z_3) = (2^{j}\downarrow)\left[\left(A^{j}_{a}(Z_1) \pm iA^{j}_{b}(Z_1)\right)\left(B^{j}_{a}(Z_2) + iB^{j}_{b}(Z_2)\right)\left(A^{j}_{a}(Z_3) + iA^{j}_{b}(Z_3)\right)\ S(Z_1Z_2Z_3)\right] \quad (2)$$

$$D^{1}_{2a/b}(Z_1Z_2Z_3) = (2^{j}\downarrow)\left[\left(B^{j}_{a}(Z_1) \pm iB^{j}_{b}(Z_1)\right)\left(A^{j}_{a}(Z_2) + iA^{j}_{b}(Z_2)\right)\left(B^{j}_{a}(Z_3) + iB^{j}_{b}(Z_3)\right)\ S(Z_1Z_2Z_3)\right] \quad (3)$$

$$D^{1}_{3a/b}(Z_1Z_2Z_3) = (2^{j}\downarrow)\left[\left(B^{j}_{a}(Z_1) \pm iB^{j}_{b}(Z_1)\right)\left(B^{j}_{a}(Z_2) + iB^{j}_{b}(Z_2)\right)\left(B^{j}_{a}(Z_3) + iB^{j}_{b}(Z_3)\right)\ S(Z_1Z_2Z_3)\right] \quad (4)$$

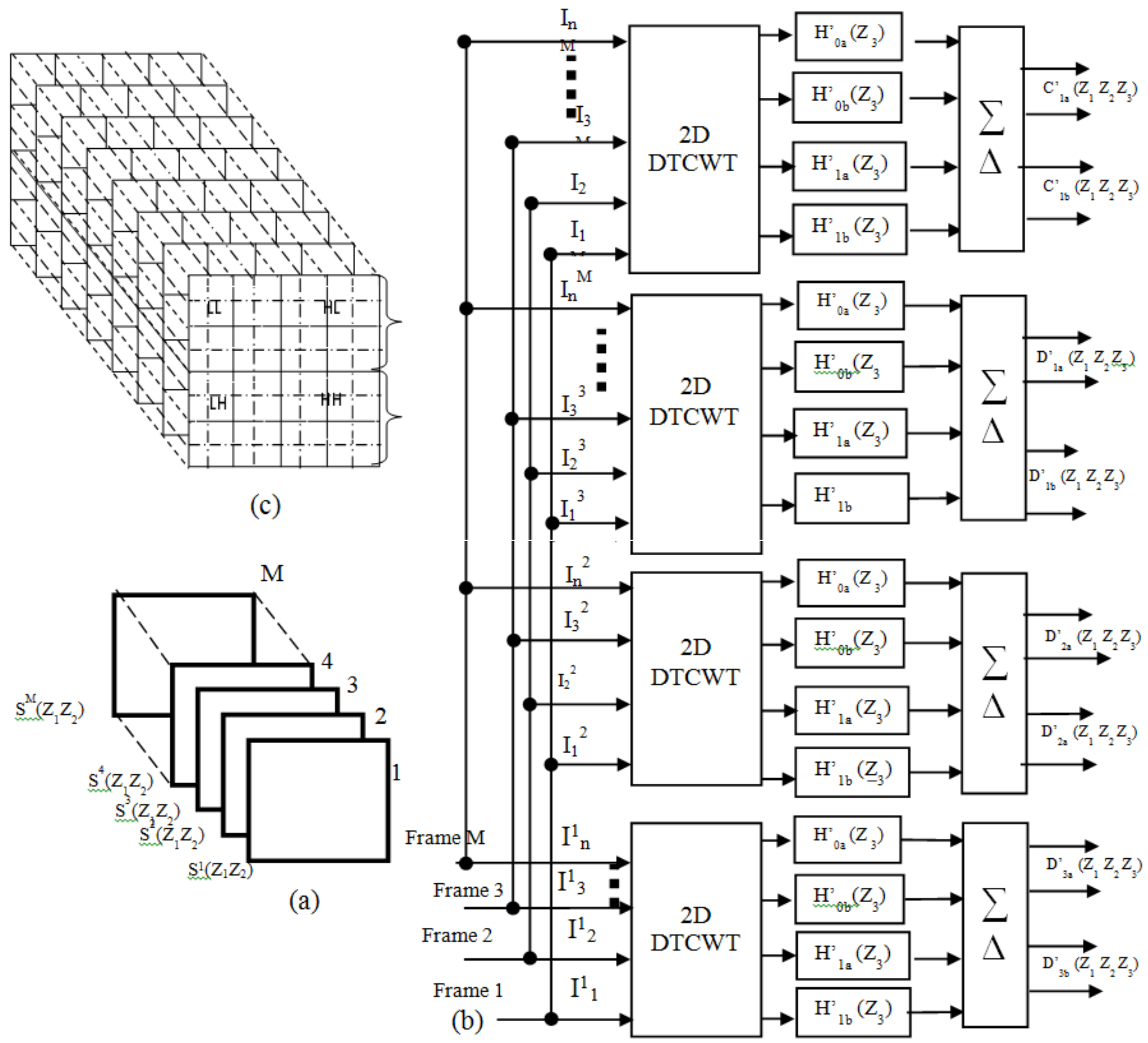

Figure. 1 DTCWT structure (a) 3D input image (b) 3D DTCWT (c) First octave of DTCWT

The 3D DTCWT module in Figure 1 is realized using dual tree filter structure shown in Figure 2 that generates six complex high-pass bands and two complex low pass sub bands at every level. The parameter r represents the real part, i_1 represent the row processed imaginary sub band and i_2 represents the column processed imaginary sub band. The output of each sub band in the filter bank is computed from the 4-tuple (represented by the expression $r + si_1 + ti_2 + u\, i_1 i_2$). The real term sub bands are obtained by setting $i_1 = i_2 = i$ for the Ca, D1a, D2a, D3a components and $-i_1 = i_2 = i$ for the Cb and D1b, D2b, D3b components.

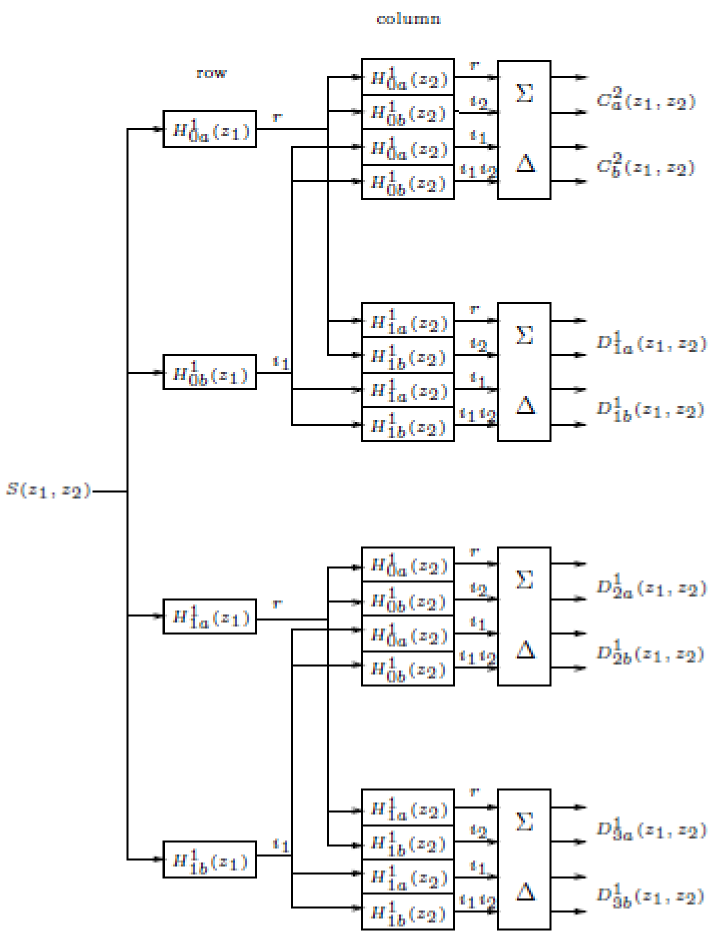

Figure. 2 2D DTCWT structure

VLSI implementation of DTCWT algorithm requires large number of arithmetic operations and memory operations to be performed. Ferhat Canbay et al. have implemented DTCWT on Spartan 6 FPGA for biomedical signal processing applications [11]. The architecture implemented on FPGA is designed with one multiplier and one adder. The hardware results of DTCWT output are compared with MATLAB simulation results. The input data to the FPGA is from the computer interfaced through Ethernet. Ferhat Canbay et al. have also implemented DTCWT on FPGA using code generator tool considering one adder, one multiplier scheme, one adder one multiplier with N channel that have been demonstrated to operate at maximum frequency of 1840 kHz and 920/N kHz respectively [12]. The architectures designed by Ferhat et al. are very basic modules for biomedical signal processing. DTCWT algorithm is similar to that of DWT it is required to develop suitable architecture considering 3D DWT architectures.

Jiang and Crookes [13] have designed area-efficient high-throughput 3D DWT architecture based on Distributed Arithmetic (DA) algorithm on Virtex-5 FPGA. Input data of size 128 x 128 x128 is regrouped into 68 x 68 x 68 of sub groups with four pixel overlap and each of these sub blocks are processed by 3D DWT based on DA algorithm designed using 9/7 wavelet filter [14 - 15]. The device utilization is 1271 slices operating at maximum frequency of 85 MHz. In 3D DWT algorithm the input data is processed along the rows in the first stage, along the columns in the second stage to compute 2D sub bands for each of the frame and in the third stage the 2D sub bands are processed along the temporal direction to generate 3D DWT sub bands. Wavelet filters in each of the stage is either 9/7 or 5/3 filters that requires arithmetic units such as adders and multipliers for low pass or high pass filter implementation. Convolution based 3D DWT schemes are faster with high usage of arithmetic resources [1], lifting scheme based 3D DWT [2-3] reduces the number of arithmetic operations, DA based algorithms is multiplierless structure [4], transpose based schemes with parallel processing schemes provide faster computation [5 – 6] and in systolic array based methods throughput and latency is improved. The design challenges in 3D DWT are optimizing the number of arithmetic operations, processing speed, power requirement and memory utilization. Reconfiguration of 3D DWT algorithm and architecture for processing multilevel decomposition is one of the key requirements. One of the methods that is based on parallel architecture and memory optimization is proposed by Y. Hu et al. [7] for 2D DWT. Darji et al. [34] have designed 3D DWT structure with memory efficient schemes and high throughput. Srinivasa rao and Chakrabarti [8] have designed 3D DWT with parallel processing schemes and lifting schemes that utilize less memory resources based on 9/7 filter. ASIC implementation of 3D DWT architecture and Xilinx platform implementation for the design is carried out demonstrating the architecture operating at maximum frequency of 200 MHz and 265 MHz respectively. Divakar et al. [9] have designed nine stage 2D parallel processing module with modified DA algorithm to compute 3D DWT and 3D IDWT processing based on 9/7 Daubechies filter. The 3D DWT engine is implemented on Xilinx FPGA Virtex-5 XC5VLX155T operating at maximum frequency of 381 MHz clock. Divakar et al. [9] have designed 2D-DTCWT based on hybrid architecture (combining DA algorithm and multiplexer logic) and is implemented on FPGA that occupies 112 slices and operates at maximum frequency of 496MHz consuming power less than 0.2W. Poornima[10] 3D DTCWT architectures reported in literature are designed considering DA algorithm, systolic array schemes and parallel processing techniques. Computing the 3D DTCWT sub bands minimizing number of arithmetic operations by exploiting redundancy in filter coefficients will be advantageous for multi-level decomposition of input data. A detailed discussion on improved methods for 3D DTCWT implementation on FPGA platform is presented.

3. 3D DTCWT Architecture

The first stage of 3D DTCWT algorithm comprises of four filters that process the input data along the rows, the second stage processes the data along the columns as shown in Figure 2 requires 16 filters. The third stage requires 16 filters that process data in the temporal directions. The 3D DTCWT structure is modular in nature and requires 9 filter banks with each filter bank comprising of four filters. The four filter coefficients considered in this work are 10-tap filters (Kingsbury 10-tap filter) and are presented in Table 1. The filter outputs are mathematically represented as in Eq. (4) based on convolution expression. The filter transfer function is represented by {H0a, H0b, H1a, H1b}, the input is represented by S(n1,n2) and the filter outputs are represented as YLR, YHR, YLI, YHI. The number of multiplications and additions for generating one output sample per filter will be 10 and 9 respectively. The total number of arithmetic operations for processing N rows of data having N pixels will be $40N^2$ and $36N^2$ multipliers and adders. In this work two different approaches for implementing the filter bank is presented, first is based on DA logic and the second is based on systolic array logic.

Table 1- Filter Coefficients

Dtf1_df				Dtf1_rf			
0	0	0	0	0	0	0	0
-0.0884	-23	-0.0112	-3	0.0112	3	-0.0884	-23
0.0884	23	0.0112	3	0.0112	3	-0.0884	-23
0.6959	178	0.0884	23	-0.0884	-23	0.6959	178
0.6959	178	0.0884	23	0.0884	23	-0.6959	-178
0.0884	23	-0.6959	-178	0.6959	178	0.0884	23
-0.0884	-23	0.6959	178	0.6959	178	0.0884	23
0.0112	3	-0.0884	-23	0.0884	23	0.0112	3
0.0112	3	-0.0884	-23	-0.0884	-23	-0.0112	-3
0	0	0	0	0	0	0	0

3.1 Memory efficient DA DTCWT architecture

The proposed modular Distributed Array (DA) architecture for the filter coefficients consists of 10 input registers and each of it is of 10 bits in size all operating in parallel. The DTCWT coefficient enters the first input register and goes to the memory and to the second input register just above the first. When the second coefficient enters the first input register the first coefficient is available in the second input register. When the coefficient number 10 is arrives at the first input register, the coefficient 1 would be at input register 10. Thus after 10 coefficients, the coefficient 11 will be operated in parallel with the previous 9 coefficients.
In the reduced modular DA architecture 1, the parallelism is done for the first group of 5 input coefficients and the second group of 5 input coefficients which is the architecture depicted in the figure 3.This increases the speed as compared to the previous architecture.

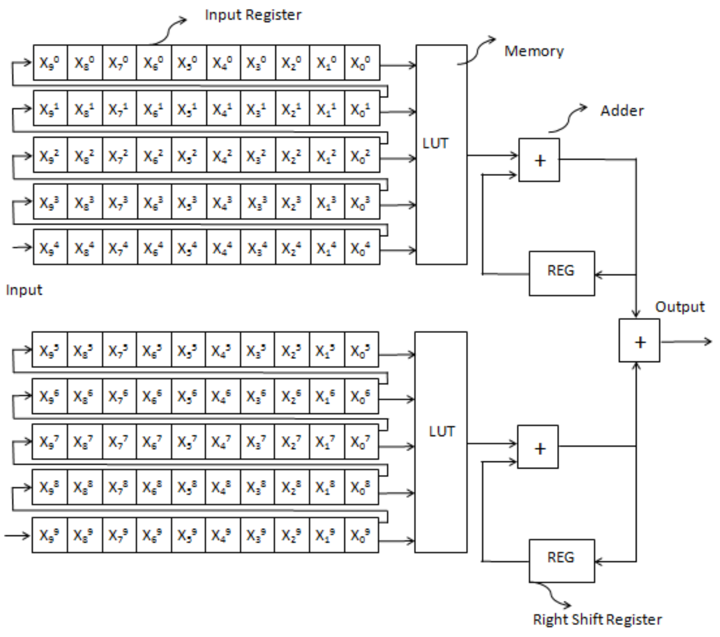

Figure 3 Reduced Modular DA Architecture–1

In the next architecture, called the reduced modular DA architecture 2, the parallelism is carried out for the first group of lower order 5 input coefficients and a second group of higher order 5 input coefficients of the first 5 inputs, a third group of lower order 5 input coefficients of the second 5 input coefficients and fourth group of higher order 5 input coefficients of the second 5 input coefficients. This architecture is as depicted in the figure 4.This increases the speed as compared to the previous 2 architecture.

Equation 5 shown below is used for the computation of the coefficients.

$$\left\{\begin{array}{c} 0[2^9X_0^9+2^8X_0^8+2^7X_0^7+2^6X_0^6+2^5X_0^5+2^4X_0^4+2^3X_0^3+2^2X_0^2+2^1X_0^1+2^0X_0^0] \\ -23[2^9X_1^9+2^8X_1^8+2^7X_1^7+2^6X_1^6+2^5X_1^5+2^4X_1^4+2^3X_1^3+2^2X_1^2+2^1X_1^1+2^0X_1^0] \\ 23[2^9X_2^9+2^8X_2^8+2^7X_2^7+2^6X_2^6+2^5X_2^5+2^4X_2^4+2^3X_2^3+2^2X_2^2+2^1X_2^1+2^0X_2^0] \\ 178[2^9X_3^9+2^8X_3^8+2^7X_3^7+2^6X_3^6+2^5X_3^5+2^4X_3^4+2^3X_3^3+2^2X_3^2+2^1X_3^1+2^0X_3^0] \\ 178[2^9X_4^9+2^8X_4^8+2^7X_4^7+2^6X_4^6+2^5X_4^5+2^4X_4^4+2^3X_4^3+2^2X_4^2+2^1X_4^1+2^0X_4^0] \end{array}\right\}$$

$$+$$

$$\left\{\begin{array}{c} +23[2^9X_5^9+2^8X_5^8+2^7X_5^7+2^6X_5^6+2^5X_5^5+2^4X_5^4+2^3X_5^3+2^2X_5^2+2^1X_5^1+2^0X_5^0] \\ -23[2^9X_6^9+2^8X_6^8+2^7X_6^7+2^6X_6^6+2^5X_6^5+2^4X_6^4+2^3X_6^3+2^2X_6^2+2^1X_6^1+2^0X_6^0] \\ 03[2^9X_7^9+2^8X_7^8+2^7X_7^7+2^6X_7^6+2^5X_7^5+2^4X_7^4+2^3X_7^3+2^2X_7^2+2^1X_7^1+2^0X_7^0] \\ 03[2^9X_8^9+2^8X_8^8+2^7X_8^7+2^6X_8^6+2^5X_8^5+2^4X_8^4+2^3X_8^3+2^2X_8^2+2^1X_8^1+2^0X_8^0] \\ 0[2^9X_9^9+2^8X_9^8+2^7X_9^7+2^6X_9^6+2^5X_9^5+2^4X_9^4+2^3X_9^3+2^2X_9^2+2^1X_9^1+2^0X_9^0] \end{array}\right\}$$

Equation (5)

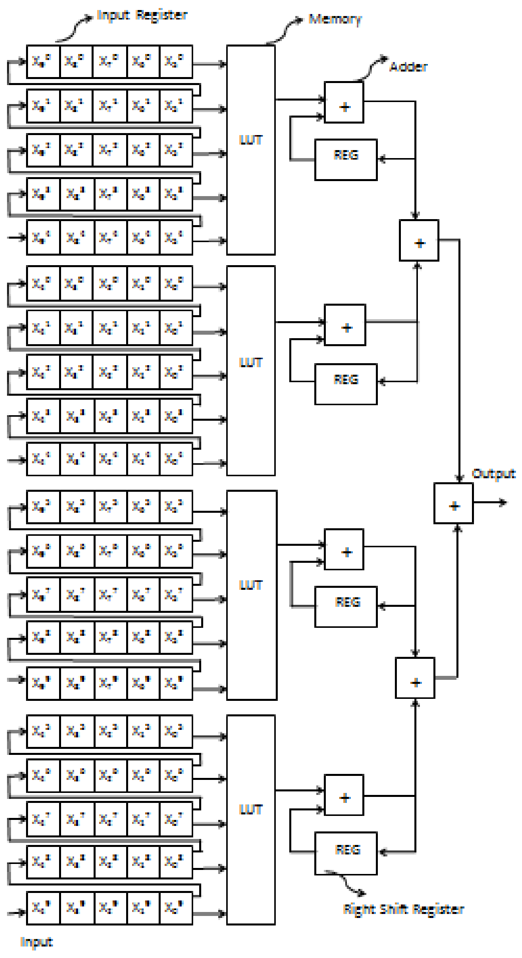

Figure 4 Reduced Modular DA Architecture–2

The equation 6 is used for the coefficient calculations for the reduced modular DA architecture 2 depicted in figure 4.

$$\left\{\begin{array}{l} 0\ 2^5(X_0^9 2^4 + X_0^8 2^3 + X_0^7 2^2 + X_0^6 2^1 + X_0^5 2^0) \\ -23\ 2^5(X_1^9 2^4 + X_1^8 2^3 + X_1^7 2^2 + X_1^6 2^1 + X_1^5 2^0) \\ 23\ 2^5(X_2^9 2^4 + X_2^8 2^3 + X_2^7 2^2 + X_2^6 2^1 + X_2^5 2^0) \\ 178\ 2^5(X_3^9 2^4 + X_3^8 2^3 + X_3^7 2^2 + X_3^6 2^1 + X_3^5 2^0) \\ 178\ 2^5(X_4^9 2^4 + X_4^8 2^3 + X_4^7 2^2 + X_4^6 2^1 + X_4^5 2^0) \end{array}\right. + \left|\begin{array}{l} 0\ (X_0^4 2^4 + X_0^3 2^3 + X_0^2 2^2 + X_0^1 2^1 + X_0^0 2^0) \\ -23\ (X_1^4 2^4 + X_1^3 2^3 + X_1^2 2^2 + X_1^1 2^1 + X_1^0 2^0) \\ 23\ (X_2^4 2^4 + X_2^3 2^3 + X_2^2 2^2 + X_2^1 2^1 + X_2^0 2^0) \\ 178\ (X_3^4 2^4 + X_3^3 2^3 + X_3^2 2^2 + X_3^1 2^1 + X_3^0 2^0) \\ 178\ (X_4^4 2^4 + X_4^3 2^3 + X_4^2 2^2 + X_4^1 2^1 + X_4^0 2^0) \end{array}\right)\Bigg\}$$

$$+\left\{\begin{array}{l} 23\ 2^5(X_5^9 2^4 + X_5^8 2^3 + X_5^7 2^2 + X_5^6 2^1 + X_5^5 2^0) \\ -23\ 2^5(X_6^9 2^4 + X_6^8 2^3 + X_6^7 2^2 + X_6^6 2^1 + X_6^5 2^0) \\ 03\ 2^5(X_7^9 2^4 + X_7^8 2^3 + X_7^7 2^2 + X_7^6 2^1 + X_7^5 2^0) \\ 03\ 2^5(X_8^9 2^4 + X_8^8 2^3 + X_8^7 2^2 + X_8^6 2^1 + X_8^5 2^0) \\ 0\ 2^5(X_9^9 2^4 + X_9^8 2^3 + X_9^7 2^2 + X_9^6 2^1 + X_9^5 2^0) \end{array}\right. + \left|\begin{array}{l} 23\ (X_5^4 2^4 + X_5^3 2^3 + X_5^2 2^2 + X_5^1 2^1 + X_5^0 2^0) \\ -23\ (X_6^4 2^4 + X_6^3 2^3 + X_6^2 2^2 + X_6^1 2^1 + X_6^0 2^0) \\ 03\ (X_7^4 2^4 + X_7^3 2^3 + X_7^2 2^2 + X_7^1 2^1 + X_7^0 2^0) \\ 03\ (X_8^4 2^4 + X_8^3 2^3 + X_8^2 2^2 + X_8^1 2^1 + X_8^0 2^0) \\ 0\ (X_9^4 2^4 + X_9^3 2^3 + X_9^2 2^2 + X_9^1 2^1 + X_9^0 2^0) \end{array}\right)\Bigg\}$$

Equation (6)

Another memory efficient Modular DA architecture is proposed, which is as shown in the figure 5. This is called the optimized structure 1. In this architecture, the optimum filter coefficients are taken for the calculation of coefficients.

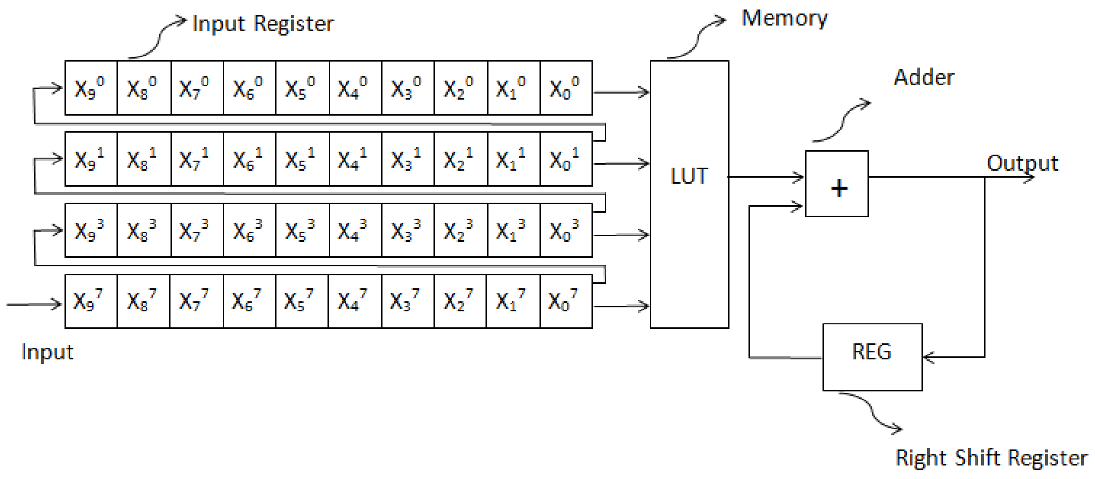
Input Register
Memory
Adder
X_9^0 X_8^0 X_7^0 X_6^0 X_5^0 X_4^0 X_3^0 X_2^0 X_1^0 X_0^0
X_9^1 X_8^1 X_7^1 X_6^1 X_5^1 X_4^1 X_3^1 X_2^1 X_1^1 X_0^1
X_9^3 X_8^3 X_7^3 X_6^3 X_5^3 X_4^3 X_3^3 X_2^3 X_1^3 X_0^3
X_9^7 X_8^7 X_7^7 X_6^7 X_5^7 X_4^7 X_3^7 X_2^7 X_1^7 X_0^7
LUT
+
Output
REG
Input
Right Shift Register

Figure. 5 Optimized Modular DA architecture 1

A variant of the above structure is the other memory efficient modular DA architecture depicted in the figure 6. This is the optimized structure 2.

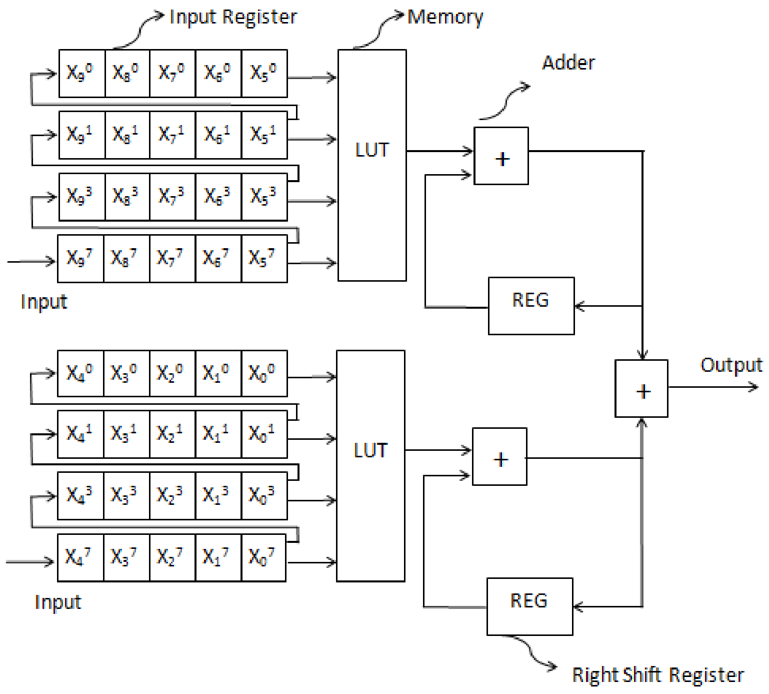

Figure. 6 Optimized Modular DA architecture 2

The optimised way of the coefficients calculation is done with the 2 equations 7 and 8 which yield a comparable results

$$\left\{\begin{array}{c}
0[2^9X_0^9+2^8X_0^8+2^7X_0^7+2^6X_0^6+2^5X_0^5+2^4X_0^4+2^3X_0^3+2^2X_0^2+2^1X_0^1+2^0X_0^0] \\
3[2^9X_7^9+2^8X_7^8+2^7X_7^7+2^6X_7^6+2^5X_7^5+2^4X_7^4+2^3X_7^3+2^2X_7^2+2^1X_7^1+2^0X_7^0] \\
20[2^9X_1^9+2^8X_1^8+2^7X_1^7+2^6X_1^6+2^5X_1^5+2^4X_1^4+2^3X_1^3+2^2X_1^2+2^1X_1^1+2^0X_1^0] \\
155\ [2^9X_3^9+2^8X_3^8+2^7X_3^7+2^6X_3^6+2^5X_3^5+2^4X_3^4+2^3X_3^3+2^2X_3^2+2^1X_3^1+2^0X_3^0]
\end{array}\right\}$$

Equation – (7)

$$\left\{\begin{array}{c}
0\ 2^5(X_0^9 2^4+X_0^8 2^3+X_0^7 2^2+X_0^6 2^1+X_0^5 2^0) \\
3\ 2^5(X_7^9 2^4+X_7^8 2^3+X_7^7 2^2+X_7^6 2^1+X_7^5 2^0) \\
20\ 2^5(X_1^9 2^4+X_1^8 2^3+X_1^7 2^2+X_1^6 2^1+X_1^5 2^0) \\
155\ 2^5(X_3^9 2^4+X_3^8 2^3+X_3^7 2^2+X_3^6 2^1+X_3^5 2^0)
\end{array} + \left|\begin{array}{c}
0\ (X_0^4 2^4+X_0^3 2^3+X_0^2 2^2+X_0^1 2^1+X_0^0 2^0) \\
03\ (X_7^4 2^4+X_7^3 2^3+X_7^2 2^2+X_7^1 2^1+X_7^0 2^0) \\
20(X_1^4 2^4+X_1^3 2^3+X_1^2 2^2+X_1^1 2^1+X_1^0 2^0) \\
155(X_3^4 2^4+X_3^3 2^3+X_3^2 2^2+X_3^1 2^1+X_3^0 2^0)
\end{array}\right.\right\}$$

Equation – (8)

3.2 High throughput DTCWT architecture

Systolic Array architecture has a higher throughput. Shown below, in figure 7 is the scheme of the SA architecture, consisting a number of Processing Elements (PE). The M inputs enters the processing element sequentially with the input sequence and ‘a’ coefficients this output enters the next processing element where the input id processed with the ‘b’ coefficients.

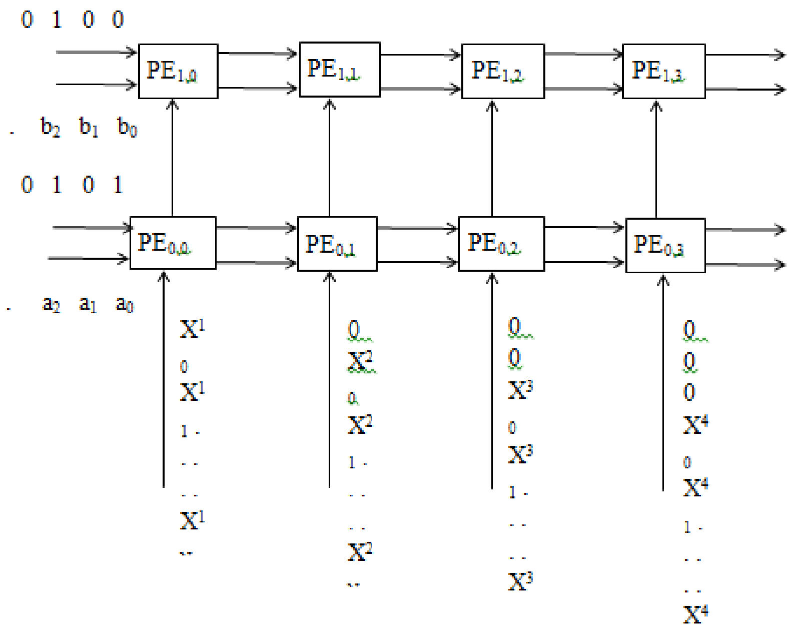

Figure.7 SA Architecture

Figure 8 depicts the PE used as building block in the figure 7. It consists of a multiplier which multiplies the a coefficients with the X inputs. The Even sequence sign change vector changes the sign of the even coefficients.

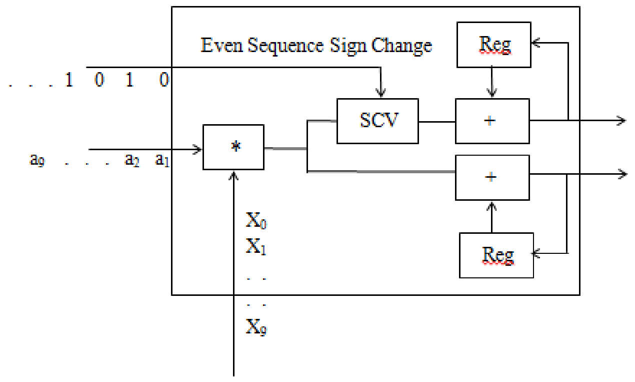

Figure. 8 PE for Even sequence

Figure 9 depicts the PE used as building block in the figure 7. It consists of a multiplier which multiplies the a coefficients with the X inputs. The Odd sequence sign change vector changes the sign of the Odd coefficients.

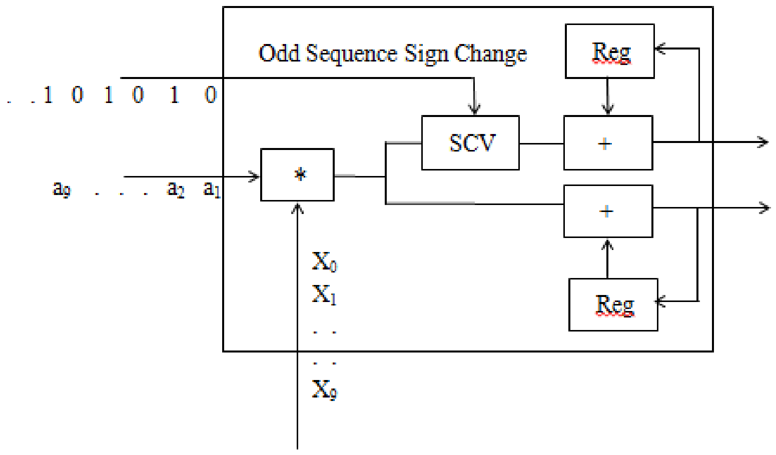

Figure. 9 PE for Odd sequence

Figure 10 depicts the reduced modular architecture using the PE described above. The even and odd bits are split and processed with LVT for producing the filter bank coefficients.

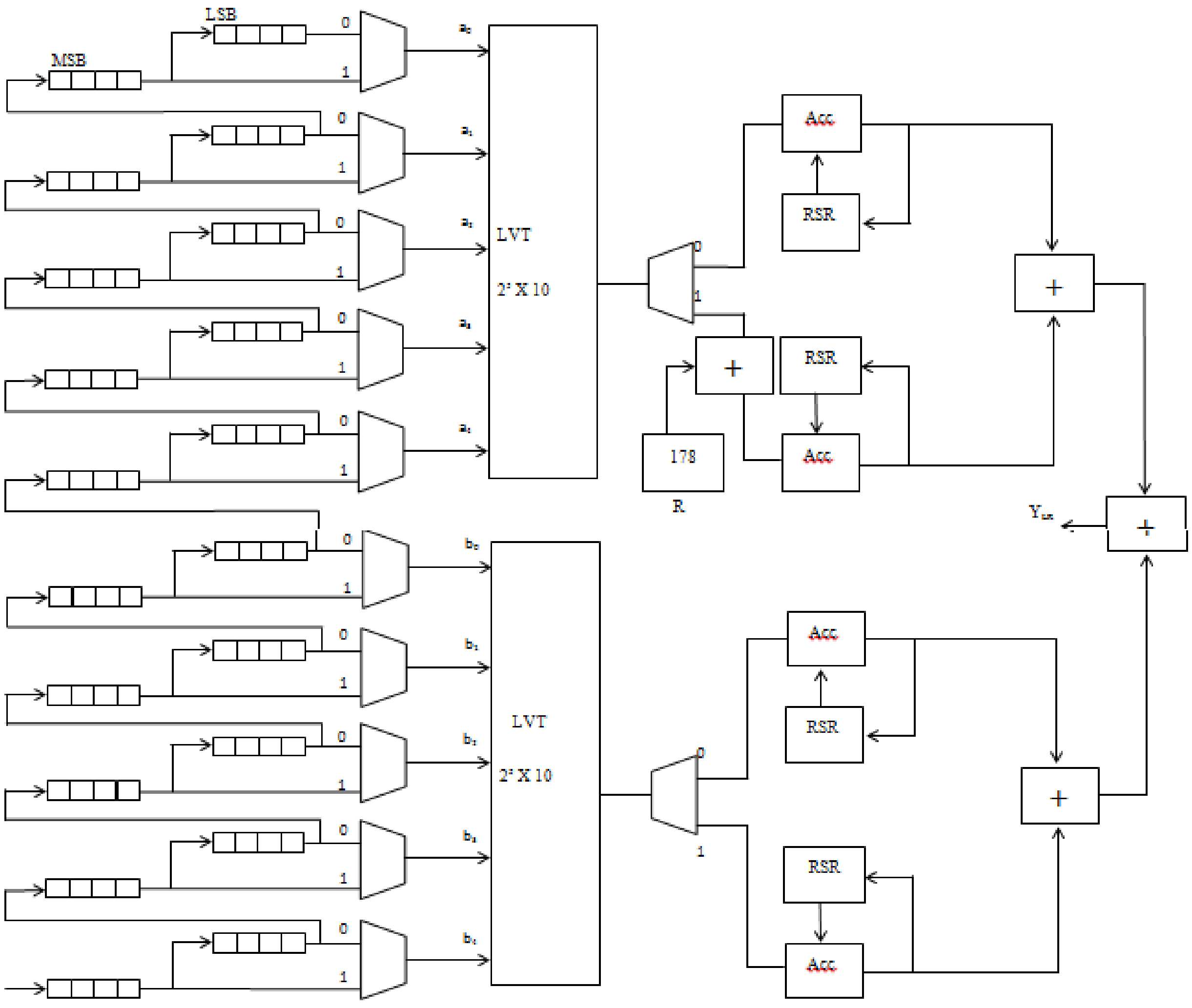

Figure. 10 Reduced modular SA architecture

Figure 11 depicts the optimized modular SA architecture using the PE described above. The even and odd bits are split and processed with LVT for producing the filter bank coefficients. The optimum filter bank coefficients are used in the LVT

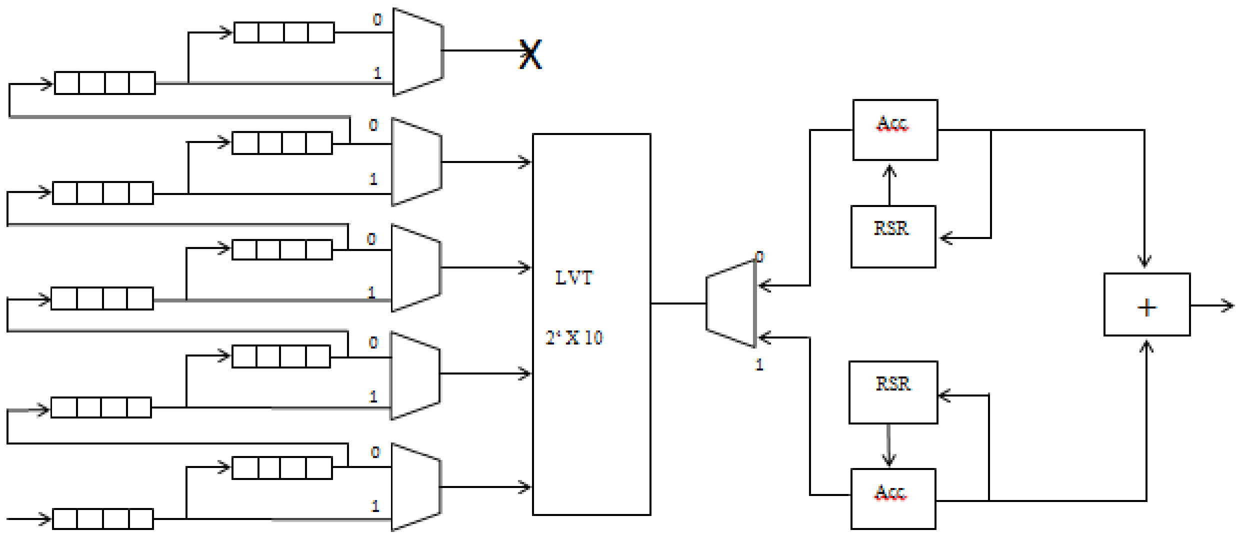

Figure. 11 Optimized modular SA Architecture

The Concept of the SA architecture is as depicted below. With the input X's multiplied with the h_{0a} gives the outputs Y_{0a}'s and multiplying it with h_{1a} gives Y_{1a}. Similarly, X's multiplied with h_{0b} yields the Y0b and its multiplication with the h_{1b} yields the Y_{1b}.

$$\begin{bmatrix} X4 & X3 & X2 & X1 & X0 & X1 & X2 & X3 & X4 & X5 \\ X2 & X3 & X4 & X5 & X6 & X7 & X8 & X9 & X10 & X11 \\ X8 & X9 & X10 & X11 & X12 & X13 & X14 & X15 & X16 & X17 \\ X14 & X15 & X16 & X17 & X18 & X19 & X20 & X21 & X22 & X23 \\ X20 & X21 & X22 & X23 & X24 & X25 & X26 & X27 & X28 & X29 \\ X25 & X26 & X27 & X28 & X29 & X30 & X31 & X32 & X33 & X34 \\ X31 & X32 & X33 & X34 & X35 & X36 & X37 & X38 & X39 & X40 \\ X38 & X39 & X40 & X41 & X42 & X43 & X44 & X45 & X46 & X47 \\ X44 & X45 & X46 & X47 & X48 & X49 & X50 & X51 & X52 & X53 \\ X50 & X51 & X52 & X53 & X54 & X55 & X56 & X57 & X58 & X59 \end{bmatrix} \begin{bmatrix} h_{0a}(0) & h_{1a}(0) & h_{0b}(0) & h_{1b}(0) \\ h_{0a}(1) & h_{1a}(0) & h_{0b}(1) & h_{1b}(1) \\ h_{0a}(2) & h_{1a}(0) & h_{0b}(2) & h_{1b}(2) \\ h_{0a}(3) & h_{1a}(0) & h_{0b}(3) & h_{1b}(3) \\ h_{0a}(4) & h_{1a}(0) & h_{0b}(4) & h_{1b}(4) \\ h_{0a}(5) & h_{1a}(0) & h_{0b}(5) & h_{1b}(5) \\ h_{0a}(6) & h_{1a}(0) & h_{0b}(6) & h_{1b}(6) \\ h_{0a}(7) & h_{1a}(0) & h_{0b}(7) & h_{1b}(7) \\ h_{0a}(8) & h_{1a}(0) & h_{0b}(8) & h_{1b}(8) \\ h_{0a}(9) & h_{1a}(0) & h_{0b}(9) & h_{1b}(9) \end{bmatrix} = \begin{bmatrix} Y^0_{0a} & Y^0_{1a} & Y^0_{0b} & Y^0_{1b} \\ Y^1_{0a} & Y^1_{1a} & Y^1_{0b} & Y^1_{1b} \\ Y^2_{0a} & Y^2_{1a} & Y^2_{0b} & Y^2_{1b} \\ Y^3_{0a} & Y^3_{1a} & Y^3_{0b} & Y^3_{1b} \\ Y^4_{0a} & Y^4_{1a} & Y^4_{0b} & Y^4_{1b} \\ Y^5_{0a} & Y^5_{1a} & Y^5_{0b} & Y^5_{1b} \\ Y^6_{0a} & Y^6_{1a} & Y^6_{0b} & Y^6_{1b} \\ Y^7_{0a} & Y^7_{1a} & Y^7_{0b} & Y^7_{1b} \\ Y^8_{0a} & Y^8_{1a} & Y^8_{0b} & Y^8_{1b} \\ Y^9_{0a} & Y^9_{1a} & Y^9_{0b} & Y^9_{1b} \end{bmatrix}$$

4. FPGA Implementation & Results

The FPGA implementation with results obtained of the proposed architectures are discussed in this section. The black box usage is discussed in Table 2. the RMSA architecture uses maximum BELS in all and does not use the MULT_AND sections. The RMDA architectures uses the most number of Flipflops/Latches and IO buffers. The DSP blocks are not used in other architectures but in RMSA Architectures.

Table: 2 Blackbox Usage

	RMSA	ORMSA	OMDA-1	OMDA-2	RMDA1	RMDA-2
BELS	2246	576	1208	1208	2160	2160
MULT_AND	–	–	64	64	224	224
Flipflops/Latches	1524	1524	860	860	1728	1728
I/O Buffers	322	322	111	111	322	379
DSPs	16	16	–	–	–	–

The comparison of SLUs is presented in Table 3. the RMDA architecture uses maximum number of Slice registers and LUTs and also the number of Logic used is high in RMDA architecture but the highest is in RMSA. The Memory used is highest in RMDA architecture. The number used as SLR is most in the RMSA architecture. The percentage of usage is highest in RMSA and least in RMDA1

Table : 3 Comparison of SLUs

	RMSA	ORMSA	OMDA-1	OMDA-2	RMDA1	RMDA-2
Number of Slice Registers:	860	419	1524	1224	1728	1528
Number of Slice LUT	708	403	1534	1231	1620	1502
Number used as Logi	584	327	1056	1056	1156	1156
Number used as Memory:	368	368	584	584	1056	1056
Number used as SRL	368	368	124	124	144	144
Number of LUT Flip Flop pairs used:	802	388	872	572	1872	1637
Number with an unus Flip Flop:	113	113	12	12	48	48
Number with an unus LUT:	113	113	164	164	576	576
Number of fully used LUT-FF pairs:	1411	1411	696	696	1152	1152
Number of unique control sets:	80	80	32	32	80	80
% of Usage	86.19	86.19	72.07	79.81	64.86	68.55
Number of IOs:	325	325	114	114	382	325
Number of bonded IOBs:	323	323	112	112	380	323
Number of BUFG/BUFGCTRL/ FHCEs:	1	1	1	1	1	1
Number of DSP48A1	16	4	–	–	–	–

The comparison of timing report is shown in Table 4. the RMDA architecture uses maximum number of Slice registers and LUTs and also the number of Logic used is high in RMDA architecture but the highest is in RMSA.

Table: 4 Timing Report

	RMSA	ORMSA	OMDA-1	OMDA-2	RMDA1	RMDA-2
Minimum period:	3.508ns	3.561ns	1.988ns	1.988ns	4.017ns	4.101ns
Maximum Frequen (MHz)	285.083	280.796	503.069	248.967	248.967	243.873
Minimum input arr time before clock:	3.893ns	3.215ns	1.561ns	1.212ns	4.450ns	3.960ns
Maximum output required time after clock:	3.634ns	3.634ns	1.591ns	0.511ns	3.597ns	3.597ns
Total number of pa / destination ports:	13812/299	3843/827	17184/2368	17184/236	17184/2368	9540/121
Total REAL time to Xst completion:	154.00 sec	104.00 se	48.00 secs	47.00 secs	87.00 secs	87.00 sec
Total CPU time to completion:	153.68 sec	103.76 se	47.87 secs	46.91 secs	86.76 secs	6.76 ec

5. Conclusion

The tables 2 – 4 gives the comparison of the results obtained using the proposed architectures - RMSA, ORMSA, RMDA and OMDA for the evaluating the coefficients of DTCWT. The RMSA architectures proposed, uses DSP blocks and provides high throughput as compared to the RMDA and OMDA architecture. The RMDA and the OMDA architectures are the memory efficient architectures as compared to the RMSA.

The Memory used is highest in RMDA architecture. The number used as SRL is most in the RMSA architecture. The percentage of usage is highest in RMSA and least in RMDA1. The percentage of usage is maximum in the SA architecture and is least in the RMDA architecture.

Acknowledgement:

I am deeply indebted to Dr. Cyril Prasanna Raj, Professor, Dept of ECE, Cambridge Institute of Technology, Bangalore, Karnataka, and thank him for his valuable inputs untiring effort in helping me to get this paper published.

References

1. *Qionghai Dai; Xinjian Chen; Chuang Lin , "A novel VLSI architecture for multidimensional discrete wavelet transform" IEEE Transactions on Circuits and Systems for Video Technology (Volume: 14, Issue: 8, Aug. 2004)*

2. Zhang, G., Badaway, W., Talley, M., Weeks, M., and Bayoumi, M.: 'A low-power prototype for a 3-D discrete wavelet transform'. Proc. of 1999 IEEE Int. Symp. on Circuits and Systems, Florida, USA, 1999, Vol. 1, p. 145

3. Das, B., and Banerjee, S.: 'A memory efficient 3-D DWT architecture'. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Florida, USA, May 2002, Vol. 3, p. 3224

4. Ja'Afar, N.H.; Ahmad, A.; Amira, A. Distributed arithmetic architecture of Discrete Wavelet Transform (DWT) with hybrid method. In Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems, Abu Dhabi, United Arab Emirates, 8–11 December 2013; pp. 501–507.

5. Ja'Afar, N.H.; Ahmad, A.; Amira, A. Rapid prototyping of three-dimensional transform for medical image compression. In Proceedings of the 2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA), Montreal, QC, Canada, 2–5 July 2012; pp. 842–847,

6. Ahmad, A.; Ja'afar, N.H.; Amira, A. FPGA-based implementation of 3-D Daubechies for medical image compression. In Proceedings of the 2012 IEEE-EMBS Conference on Biomedical Engineering and Sciences, Langkawi, Malaysia, 17–19 December 2012; pp. 683–688.

7 . Y. Hu and C. C. Jong, "A Memory-Efficient Scalable Architecture for Lifting-Based Discrete Wavelet Transform,"IEEE Transactions on Circuits and Systems-II: Express Briefs, VOL. 60, NO. 8, pp. 502-506, Aug. 2013

8. B.K.N.Srinivasarao and Indrajit Chakrabarti," High performance VLSI architecture for 3-D discrete wavelet transform", 2016 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), DOI: 10.1109/VLSI-DAT.2016.7482578

9. S. S. Divakara, Sudarshan Patilkulkarni, Cyril Prasanna Raj, "High Speed Area Optimized Hybrid DA Architecture for 2D-DTCWT" International Journal of Image and GraphicsVol. 18, No. 1, 2018 https://doi.org/10.1142/S0219467818500043

10. Poornima B., Sumathi A., Cyril Prasanna Raj P., " Memory Efficient High Speed Systolic Array Architecture Design with Multiplexed Distributed Arithmetic for 2D DTCWT Computation on FPGA ", Journal of Microelectronics, Electronic Components and Materials Vol. 49, No. 3(2019), 119 – 132

11. Ferhat Canbay, Vecdi Emre Levent , Gorkem Serbes, H. Fatih Ugurdag, Sezer Goren, and Nizamettin Aydin, "A Multi-channel Real Time Implementation of Dual Tree Complex Wavelet Transform in Field Programmable Gate Arrays", XIV Mediterranean Conference on Medical and Biological Engineering and Computing 2016, IFMBE Proceedings 57, DOI: 10.1007/978-3-319-32703-7_24

12. Ferhat Canbay, Vecdi Emre Levent, Gorkem Serbes, H. Fatih Ugurdag, Sezer Goren, Nizamettin Aydin, "Code generator for implementing dual tree complex wavelet *transform on reconfigurable architectures for mobile applications", Healthcare Technology Letters, 2016, Vol. 3, Iss. 3, pp. 184–188 doi: 10.1049/htl.2016.0034*

13. M. Jiang, D. Crookes, "FPGA Implementation of 3D Discrete Wavelet Transform for Real- Time Medical Imaging", Electronics Letters, 43(9), 502- 503, https://doi.org/10.1049/el:20070201

14. M. Nagabushanam, Cyril Prasanna Raj P, S. Ramachandran, "Design and Implementation of Parallel and Pipelined Distributive Arithmetic Based Discrete Wavelet Transform IP Core", European Journal of Scientific Research ISSN 1450-216X Vol.35 No.3 (2009), pp.378-392

15. Yu and Chen, "Design_of_an_efficient_VLSI_architecture_for_2-D_DWT", IEEE transactions on consumer electronics, Vol. 45, No.1, Feb 1999, pages 135 – 140

16. Travis Williams, Robert Li, "Advanced Image Classification Using Wavelets and Convolutional Neural Networks", 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)Year: 2016, Volume: 1, Pages: 233-239

17. Travis Williams, Robert Li, " An Ensemble of Convolutional Neural Networks Using Wavelets for Image Classification", Journal of Software Engineering and Applications, 2018, 11, 69-88

18. Qin Qin, Jianqing Li, Li Zhang, Yinggao Yue & Chengyu Liu," Combining Low-dimensional Wavelet Features and Support Vector Machine for Arrhythmia Beat Classification", Scientific Report S | 7: 6067 | DOI:10.1038/s41598-017-06596-z

19. Kamarul Hawari Ghazali, Mohd Fais Mansor, Mohd. Marzuki Mustafa and Aini Hussain, "Feature Extraction Technique using Discrete Wavelet Transform for Image Classification", The 5th Student Conference on Research and Development -SCOReD 2007 11-12, December 2007, Malaysia

20 . M. B. Wakin, J.N. Laska, M. F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. F. Kelly and R.G. Baraniuk, "Compressive imaging for video representation and coding", In Proceedings of Picture Coding Symposium (PCS-06), pp. 1-6, 2006.

21. N. G. Kingsbury, "Complex wavelets for shift invariant analysis and filtering of signals," Journal of Applied Computational Harmonic Analysis, vol. 10, pp. 234–253, May 2001

22. I.W. Selesnick and K. Y. Li, "Video denoising using 2D and 3D dual-tree complex wavelet transforms," in Wavelets: Applications in Signal and Image Processing X, M. A. Unser, A. Aldroubi, and A. F. Laine, Eds. San Diego, CA: Proc. SPIE 5207, August 2003, pp. 607– 618

23. T. H. Reeves and N. G. Kingsbury, "Overcomplete image coding using iterative projection- based noise shaping," in Proceedings of the International Conference on Image Processing, vol. 3, Rochester, NY, September 2002, pp. 597–600

24. Beibei Wang, Yao Wang, Ivan Selesnick and Anthony Vetro, "Video coding using 3-D dual-tree wavelet transform," EURASIP Journal on Image and Video Processing, vol. 2007, 2007, article ID 42761, 15 pages.

25. B. Wang, Y. Wang, I. Selesnick, and A. Vetro, "An investigation of 3D dual tree wavelet transform for video coding," in Proceedings of the International Conference on Image Processing, vol. 2, Singapore, October 2004, pp. 1317–1320

26. *Julia Neumann and Gabriele Steidl, "Dual Tree Complex Wavelet Transform in the Frequency Domain and an Application to Signal Classification" International Journal of Wavelets, Multi-resolution and Information Processing Vol. 03, No. 01, pp. 43-65 (2005)*
27. *Suresh Babu D, Dr. K R Raja, Dr Cyril Prasanna Raj, "Complex Wavelets and SPIHTL for Edge Feature Enhancement and Optimum Feature Selection with Deep Learning Algorithm for Video Salient Object Detection", YMER, Volume 21, Issue 5,pp. 1134 - 1155 May 2022*
28. *M. Nagabushanam, S. Ramachandran, " Fast Implementation of Lifting based 1D/2D/3D DWT-IDWT Architecture for Image Compression", International Journal of Computer Applications (0975 – 8887) Volume 51– No.6, August 2012, pages 35 - 41*
29. *Fahad Qureshi, Jarmo Takala, Anastasia Volkova, Thibault Hilaire, "Multiplierless Unified Architecture for Mixed Radix 2/3/4 FFTs" 2017, 25th European Signal Processing Conference (EUSIPCO), pages – 1374 - 1378*
30. *H.T. Kung. Why systolic architectures? Computer, 15(1):37–46, Jan 1982.*
31. *N. G. Kingsbury," The dual-tree complex wavelet transform: a new technique for shift invariance and directional filters", In the Proceedings of the IEEE Digital Signal Processing Workshop, 1998.*
32. *Eero P Simoncelli, William T Freeman, Edward H Adelson, and David J Heeger" Shiftable multi-scale transforms". IEEE Trans. Information Theory, 38(2):587–607, March 1992. Special Issue on Wavelets*
33. *N.G. Kingsbury, "A dual-tree complex wavelet transform with improved orthogonality and symmetry properties," in Proc. IEEE Int. Conf. Image Process., Sep. 2000, pp. 375–378.*
34. *Anand Darji, Arun R., Shabbir Noman Merchant, Arun Chandorkar, "Multiplier-less pipeline architecture for lifting-based two-dimensional discrete wavelet transform" IET Comput. Digit. Tech., 2015, Vol. 9, Iss. 2, pp. 113–123.*