Original scientific paper
https://doi.org/10.33180/InfMIDEM2019.301

Informacije MIDEM
Journal of Microelectronics,
Electronic Components and Materials
Vol. 49, No. 3(2019), 119 – 132

Memory Efficient High Speed Systolic Array Architecture Design with Multiplexed Distributed Arithmetic for 2D DTCWT Computation on FPGA

Poornima B.[1], Sumathi A.[2], Cyril Prasanna Raj P.[3]

[1]Anna University, Research Scholar, Department of Electrical and Electronics Engineering, Chennai, Tamilnadu, India, 600025
[2]Adhiyamaan College of Engineering, Department of Electronics and Communication, Hosur, Tamilnadu, India
[3]MSEC, Department of Electronics and Communication, Bangalore, India

Abstract: This paper presents customized Systolic Array Architecture (SAA) design of Dual Tree Complex Wavelet (DTCWT) sub band computation based on multiplexed Distributed Arithmetic Algorithm (DAA). The proposed architecture is memory efficient and operates at frequencies greater than 300 MHz in decomposing 256 x 256 input image. Three architectures such as reduced order structure, multiplexed DA structure and zero pad structure are designed and evaluated for its performances for DTCWT computation minimizing arithmetic operations with improved latency. The proposed design is modeled in Verilog HDL and is implemented on Spartan-6 and Virtex-5 FPGA considering Xilinx ISE FPGA design flow. The latency of proposed architectures is evaluated to be 15 clock cycles and throughput is estimated to be 4 outputs for every 5 clock cycles. The SAA architecture occupies less than 12% of FPGA resources and consumes less than 10 mW of power on FPGA platform.

Keywords: Memory efficient, high speed, FPGA, Systolic Array, Distributed Arithmetic

Spominsko učinkovita arhitektura sistoličnega polja visokih hitrosti za 2D DTCWT računanja na FPGA

Izvleček: Članek opisuje dizajn arhitekture sistoličnega polja za računanje kompleksne dvoslojne valovnice na osnovi multipleksnega distribuiranega aritmetičnega algoritma. Predlagana struktura je spominsko učinkovito in deluje s frekvenco večjo od 300 MHz pri dekompoziciji slike velikosti 256 x 256. Dizajn je narejen v Verilog HDL okolju in implementiran na Spartan-6 in Virtex-5 FPGA. Latenca arhitekture je 15 časovnih ciklov in propustnost 4 izračuni na 5 časovnih ciklov. Arhitektura zaseda manj kot 12 % FPGA spomin ain porabi 10 mW moči.

Ključne besede: učinkovitost spomina; visoke hitrosti; FPGA; sistolično polje; dinamična aritmetika

** Corresponding Author's e-mail: poornimadeep@yahoo.co.in*

1 Introduction

Wavelets have played an important role in signal and image processing applications supporting both time and frequency localization property. Hardware implementation of Discrete Wavelet Transform (DWT) is achieved by filter bank structures that are flexible and computationally less intensive. Down sampling and up sampling in the analysis and synthesis filter bank structures in DWT introduce aliasing effects leading to shift variance and directionality selectivity limitations. Complex Wavelet Transforms (CWT) and Un-decimated DWT (UDWT) have been reported in literature to overcome limitations of DWT. CWT implementation was presented by Nick Kingsbury [1] with two tree structure

for signal decomposition for image processing applications. Ioana Adam [2] in his PhD thesis presented CWT for image denoising which was performed using separable filters generating high pass sub bands providing directional information in six orientations. Selesnick et al [3] in their work have presented signal and image processing applications using Dual Tree CWT (DTCWT). Olhede and Metikas [4] have presented four different types of complex wavelets for both 1D and 2D signal decomposition. Computing DTCWT output coefficients is twice complex than DWT computation as it generates 2N DWT output coefficients, where N is the input data size. Simplified structures for computing DTCWT are presented in [5-7] that require two real DWT filter bank structures or two critically sampled DWTs that process the input data in parallel. DTCWT is well suited for image analysis such as image de-noising, texture analysis, segmentation, classification, motion estimation, watermarking and compression [6]. With additional directional features in wavelet domain arising in DTCTW sub bands image processing algorithms provide better performances for real time applications [8]. Computation complexity of DTCWT can be addressed by implementing DTCWT on FPGA platform. Several architectures for FPGA implementation of DWT [9-25] is reported in literature based on lifting scheme, distributed arithmetic algorithm, multiplierless scheme, systolic array algorithm, serial architectures, parallel architectures etc. Divakara et al [25] have reported on FPGA implementation of DTCWT for image processing applications based on reorder and symmetric structure. For improving the processing speed and reducing computation complexity of DTCWT computation redundancy among filter coefficients need be eliminated and customized arithmetic operators need to be designed. In this paper, high speed area efficient architectures for DTCWT are presented and the area, timing metrics of three different architectures are compared considering Xilinx Spartan-6 FPGA device. The proposed architectures are implemented on Spartan-6 development kit and co-simulation is performed for validation of architecture functionality. Advanced optimization options are enabled during synthesis process for further improvement in area and timing performances. Section 2 briefly introduces DTCWT algorithm and review of various architectures and methods for DWT is presented. Section 3 presents discussion on improved methods for DTCWT architectures, section 4 presents implementation details and conclusion is presented in section 5.

2 DTCWT

Kingsbury demonstrated the shift invariance property of DTCWT by using both real and imaginary tree [6] structures for signal decomposition. The aliasing effects in DWT are addressed by having two tree (tree a and tree b) structure for decomposition of input data (X) using DTCWT as shown in Figure 1. The four filters required for DTCWT decomposition are represented as La (low pass tree a coefficient), Ha (high pass tree a filter coefficient) and Lb (low pass tree b filter coefficient), Hb (high pass tree b filter coefficient). The filter coefficients defined by Daubechies 10-tap filter are presented in Table 1.

The first stage comprising of two filter pairs processes input image along the rows to generate output samples represented as {y1, y2, y3 and y4}. The second stage comprising of eight filters processes the row processed outputs along the columns to generate eight sub bands denoted as {y11, y12, y21, y22, y31, y32, y41, y42}. The first two outputs represented by y11 and y31 are the low pass sub bands and the remaining are the high pass sub bands. Sum and difference operation with scaling ($1/\sqrt{2}$) is performed on the high pass sub band outputs {y12, y21, y22, y32, y41, y42} to obtain the DTCWT high pass sub bands with six orientations. The complex sub bands of DTCWT after level-1 are represented as {LLR1, LLC1, LHR1, LHC1, HLR1, HLC1, HHR1 and HHC1}. LLR1 and LLC1 are the two low pass

Table 1: Low Pass and High Pass Filter Coefficients

Order	L_a	IL_a	H_a	IH_a	L_b	IL_b	H_b	IH_b
0	0	0	0	0	0.011226979	2	0	0
1	-0.08838834	-22	-0.011226979	-2	0.011226979	2	0	0
2	0.08838834	-22	0.011226979	2	-0.08838834	-22	-0.08838834	-22
3	0.69587998	178	0.08838834	22	0.08838834	22	-0.08838834	-22
4	0.69587998	178	0.08838834	22	0.69587998	178	0.69587998	178
5	0.08838834	22	-0.69587998	-178	0.69587998	178	-0.69587998	-178
6	-0.08838834	-22	0.69587998	178	0.08838834	22	0.08838834	22
7	0.011226979	2	-0.08838834	-22	-0.08838834	-22	0.08838834	22
8	0.011226979	2	-0.08838834	-22	0	0	0.011226979	2
9	0	0	0	0	0	0	-0.011226979	-2

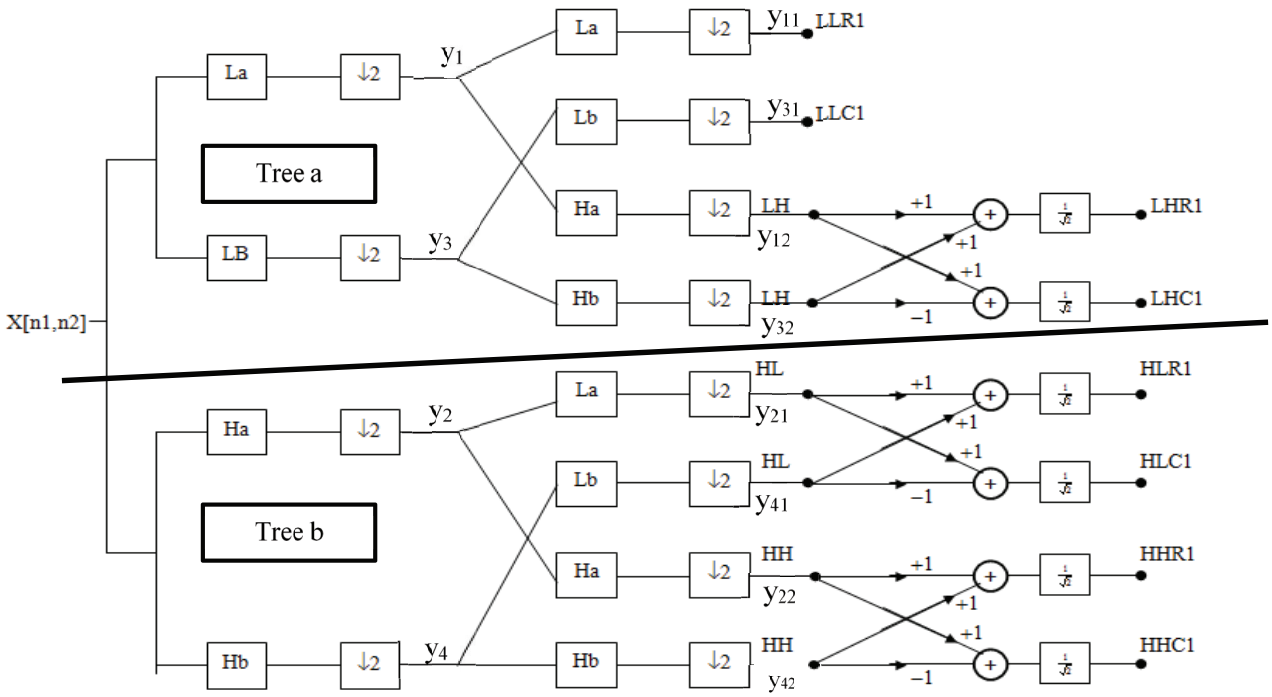

Figure 1: Level-1 2D DTCWT structure

complex sub bands and LHR1, LHC1, HLR1, HLC1, HHR1 and HHC1 are the six high pass complex sub bands. LLR refers to sub band processed by low pass filter in first stage and low pass filter in second stage.

Representing each of the filter coefficients La, Lb, Ha and Hb in binary format requires 16-bit number representation, scaling the filter coefficients by 256 and rounding off to nearest integer is carried out to represent the filter coefficients using 9-bit signed two's complement representations. La represents the low pass tree a filter coefficients and ILa refers to integer low pass tree a filter coefficient obtained after scaling. Similarly by scaling operation the filter coefficients for all other filters are obtained and are presented in Table 1. Computation complexity of DTCWT computation is expressed in terms of number of multipliers and adders required for hardware implementation. Considering single 10-tap filter computing one output coefficient requires 10 multipliers (M) and 9 adders (A). For an image of size N x N for row processing using one filter it requires 10N2 and 9N2 multipliers and adders respectively. For column processing another 10N2 and 9N2 per filter is required. Processing input data using 12 filters (both first stage and second stage), total number of multipliers and adders operations required are 120N2 and 108N2 respectively. In addition to multipliers and adders intermediate registers and memory elements are also required for DTCWT computation. Implementing DTCWT on FPGA platform requires optimizing number of arithmetic operations and memory elements. In literature, several architectures for hardware implementation of DWT optimizing arithmetic operations and memory utilization are reported improvising throughput, latency, operating speed and power dissipation. Few of the most popular methods for DWT implementation improving speed and optimizing area are reviewed in the next section that can provide an insight into the improved methods that are proposed in this work for DTCWT implementation.

2.1 Review of high speed architectures

In order to reduce arithmetic operations and area resources, multipliers are implemented using shift and add logic for computing 1D/2D DWT [10]. Barua et al [11] have presented folded multi-level DWT architecture with 100% hardware utilization requiring 12 multipliers and 16 adders with output latency of 7N. Martina and Masera [12] have presented FPGA implementation of multiplierless architecture based on Distributed Arithmetic (DA) approach considering 9/7 and 5/3 filter with folded structure. Cell based and modified lifting scheme architecture proposed is proposed by Seo and Kim [13] that requires 4 multipliers and 8 adders with Tm (multiplier delay) critical path timing. Pipelined multi-level DWT architecture is designed by Varshney et al [14] that requires 2j multipliers and j adders (j representing number of pixels) with line buffers with 60% hardware utilization and critical path delay

of 2Ta (Ta is adder delay). Distributed arithmetic based DWT architectures for 1D/2D and 3D data processing is presented by Jiang and Crookes [15] with lossy mode configuration. Lifting based 2D DWT architecture that uses non-separable scheme with complex control logic is designed with 10 multipliers and 16 adders that have a critical path Tm [16]. 2D DWT architecture with area efficient schemes and low power logic is designed by Mohanty et al [17], that require 4 multipliers and 8 adders with critical path of Tm + 2Ta. Pipelined multilevel 2D architecture based on 5/3 filter using lifting scheme is implemented on hardware platform considering 4N line buffers with 2 and 4 shifters and subtractors respectively for every stage of decomposition occupying 4N on-chip memory with 2Ta path delay requires 206 slices for every stage and consumes 1220 mW of power operating at 221.44 MHz of frequency [18]. Darji et al [19] have presented design of folded, recursive and pipelined architectures for multi-level decomposition of 2D DWT is designed and implemented on FPGA demonstrating the advantages of dual scanning method operating at speed greater than 200 MHz. Pipelined architecture that processes data in Z-scanning mode is designed by Darji et al [20] based on lifting scheme algorithm replacing multipliers with shift operators and is implemented on FPGA that requires 630 logic elements operating at maximum frequency of 353 MHz. 3D DWT based on parallel lifting algorithm 9/7 wavelet filter is designed by combining two spatial processing and four temporal processing computations [21] which are implemented on FPGA platform that requires 2852 slices and operates ate maximum frequency of 265 MHz. Systolic array based DWT architecture for 2D data with novel data scanning method is designed by Hongda Wang and Chiu-Sing Choy [22] based on lifting scheme algorithm and is implemented on VLSI platform that requires 294579 logic gates and occupies 1508243 micro meters square of area consuming less than 32.78 mW of power. Multiplierless pipelined architecture for 1D and 2D data processing is proposed by Chakraborty et al [23] based on lifting algorithm for DWT computation optimizing area requirement and latency on Spartan 3E FPGA that requires 98 adders with critical path less than T_a. High precision 3D DWT architecture for multi-level decomposition is designed and implemented on FPGA operating at 200 MHz clock consuming less than 329 mW of power for 3D image compression [24]. 2D 3-level DWT architecture is designed to operate at 365 MHz on Virtex-5 platform consuming less than 1261 slices.

Most of the architectures reported in literature focus on reducing arithmetic operations by replacing multipliers by multiplierless logic such as shift and add methods, distributed arithmetic methods and lifting method to reduce computation complexity. Pipelined and parallel processing algorithm and systolic array algorithms have also been used to improve processing speed and throughput in DWT computation. Redundancy in filter coefficients and arithmetic operations in DWT implementations have not been utilized for optimization of are and timing requirement. DTCWT is twice complex than DWT and it is required to reduce the arithmetic operation, memory requirement and processing delay by eliminating redundancy between filter bank pairs. In this work three different types of architectures are designed and implemented on FPGA optimizing area and timing requirements.

3 DTCWT architecture design

As presented in Figure 1 the first stage has four filters and second stage has eight filters. In this section design of three different architectures {reduced order, multiplexed DA and zero pad logic} for DTCWT filter is presented. The filter architecture is designed considering the scaled filter coefficients (ILa) presented in Table 1.

3.1 Reduced order architecture

Expressing the four filter outputs {y_1, y_2, y_3 and y_4} of first stage mathematically considering convolution operation is as in Eq. (1). The filter outputs of all four filters are expressed considering ILa coefficient only. As the input is common to all the four filters and the filter coefficients ILa(0) and ILa(9) are zeros the reduced order structure based on Eq. (1) is designed and is presented in Figure 2.

$$y_1(n)=\begin{pmatrix} x(n)0-[x(n-1)+x(n-2)]ILa(1)+ \\ [x(n-3)+x(n-4)]ILa(3)+[x(n-5)-x(n-6)]ILa(1)+ \\ [x(n-7)-x(n-8)]ILa(7)+x(n-9)0 \end{pmatrix} \quad (1a)$$

$$y_3(n)=\begin{pmatrix} [x(n)+x(n-1)]ILa(7)+[-x(n-2)+x(n-3)]ILa(1)+ \\ [x(n-4)-x(n-5)]ILa(3)+[x(n-6)-x(n-7)]ILa(1)- \\ x(n-8)]ILa(7)+x(n-9)0 \end{pmatrix} \quad (1b)$$

$$y_2(n)=\begin{pmatrix} x(n)0-[-x(n-1)+x(n-2)]ILa(7)+ \\ [x(n-3)+x(n-4)]ILa(1)+[-x(n-5)+x(n-6)]ILa(3)- \\ [x(n-7)+x(n-8)]ILa(1)+x(n-9)0 \end{pmatrix} \quad (1c)$$

$$y_4(n)=\begin{pmatrix} x(n)0+x(n-1)0-[x(n-2)+x(n-3)]ILa(1)+ \\ [x(n-4)-x(n-5)]ILa(3)+[x(n-6)-x(n-7)]ILa(1)+ \\ [x(n-8)-x(n-9)]ILa(7) \end{pmatrix} \quad (1d)$$

In the reduced order structure inputs are loaded into the Serial in Serial out (SISO) register that requires 10 clock cycles. Addition of input samples as per Eq. (1) considering common terms is performed by the first stage adder

structure and the results are stored in the intermediate registers. The intermediate data obtained after first stage addition is correspondingly multiplied by the ILa filter coefficients the multiplied outputs are accumulated in the second stage adder array. The first stage addition, multiplication and second stage addition requires 3 clock cycles. To compute the first output of each filter 13 clock cycles are required (10 clocks for loading data, three clocks for addition, multiplication and addition). After computing the first output at the end of 13th clock cycle every output requires four clock cycles (1 clock for loading new data into SISO, three clocks for addition, multiplication and addition). The latency is 13 clock cycles and throughput is 4 clock cycles. The reduced order structure is advantageous as it computes four filter outputs simultaneously once the data is loaded in the SISO register and hence the throughput is one clock cycle (4 outputs every four clock cycles). The reduced order architecture requires 16 multipliers, 28 adders or subtractors, 16 intermediate registers and one SISO register for implementing first stage DTCWT filter structure or row processing structure.

Implementing the reduced order design on FPGA requires use of DSP block sets for multiplication, as FPGA comprises of LUT logic in large number than the DSP block sets, distributed arithmetic based structure is designed and is presented in next section.

3.2 Multiplexed DA architecture

The second stage or column processing comprises of 8 filters that processes the four row processed outputs $\{y_1, y_2, y_3$ and $y_4\}$ to generate level-1 DTCWT sub bands. In the first stage as the input was common to all four filters, design of reduced order filter structure was an advantage. In the second stage the data samples y_1 or y_2 or y_3 or y_4 is common to two filter modules and requires 32 total numbers of multiplier. In this work, DA Algorithm is used for design of second stage filtering to demonstrate its advantages for DTCWT implementation.

$$y_{11}(n) = \begin{pmatrix} y1'(n)\,ILa(1) + y1'(n-1)\,ILa(3) + \\ y1'(n-2)\,ILa(1) + y1'(n-3)\,ILa(7) \end{pmatrix} \quad (2a)$$

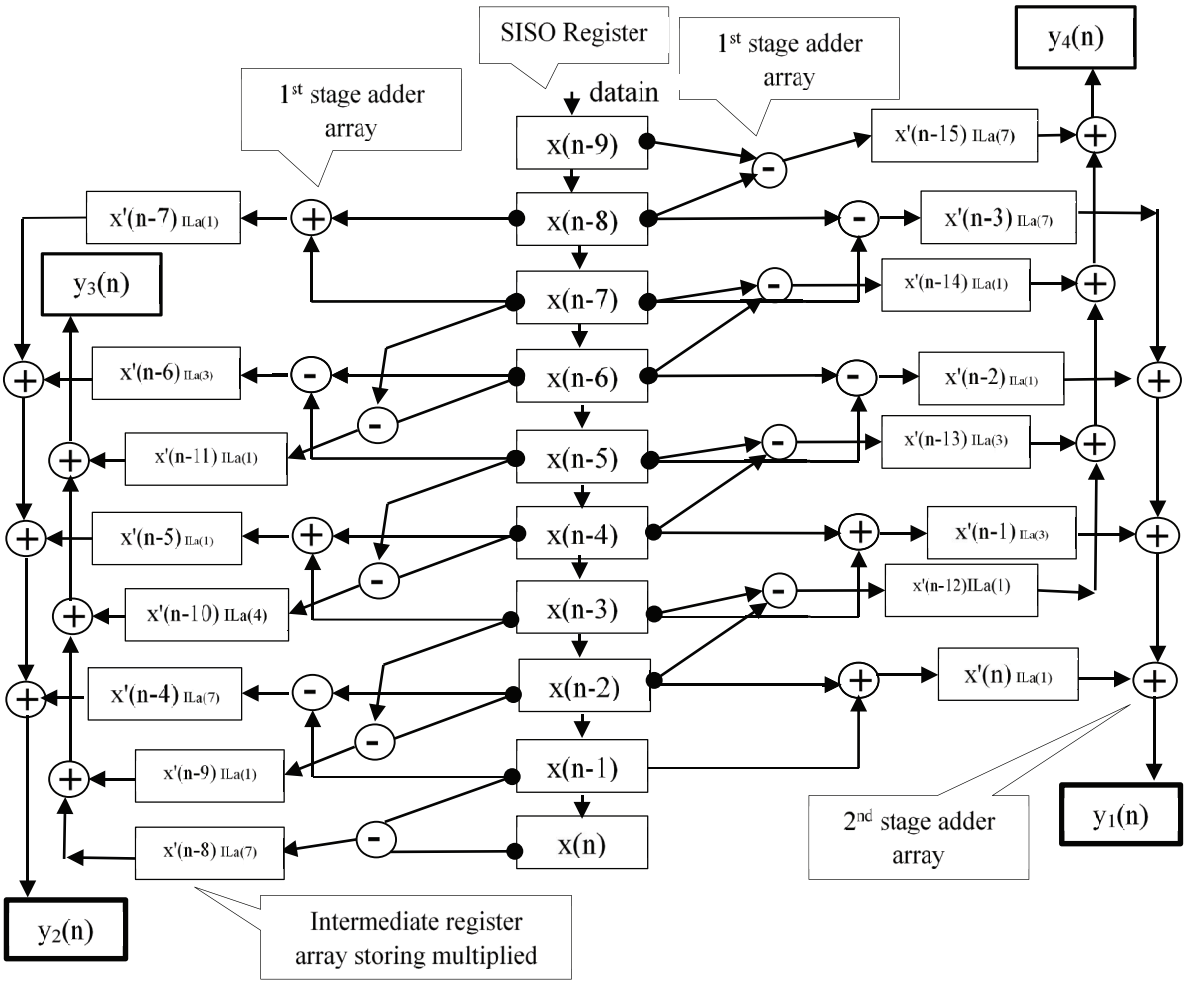

Figure 2: Reduced order DTCWT architecture

$$y_{12}(n) = \begin{pmatrix} yl'(n)\,ILa(7) + yl'(n-1)\,ILa(1) + \\ yl'(n-2)\,ILa(3) + yl'(n-3)\,ILa(1) \end{pmatrix} \quad (2b)$$

$$y_{31}(n) = \begin{pmatrix} y3'(n)ILa(7) + y3'(n-1)\,ILa(1) + \\ y3'(n-2)\,ILa(3) + y3'(n-3)ILa(1) + y3'(n-4)ILa(0) \end{pmatrix} (2c)$$

$$y_{32}(n) = \begin{pmatrix} y3'(n)ILa(0) + y3'(n-1)\,ILa(1) + \\ y3'(n-2)\,ILa(3) + y3'(n-3)ILa(1) + y3'(n-4)ILa(7) \end{pmatrix} (2d)$$

Outputs y_{11} and y_{12} are computed considering y_1 and y_{31} and y_{32} are computed considering y_3. Considering common filter coefficients the reduced expression for $y_{11,}\, y_{12,}\, y_{31}$ and y_{32} are presented in Eq. (2), the term y'(n) is obtained by adding the terms y(n) that have common filter coefficients ILa. Realizing Eq. (2) consists of two stages, the first stage computes the intermediate outputs by addition and subtraction operation and the second stage is the DA logic. Figure 3.1(a) and Figure 3.1(b) presents the two stage structure for computation of $\{y_{11,}\, y_{12}\}$ and $\{y_{31}$ and $y_{32}\}$ respectively.

In the proposed DA architecture two outputs are simultaneously computed from a single LUT that is stored with pre-computed partial products according to DA logic [12][25]. Considering Eq. 2(a) and Eq. 2(b) the order in which the ILa coefficients are multiplied with corresponding data samples y_1' are reversed. Eq. 2(b) is reorganized in accordance to Eq. 2(a) to have the ILa coefficients sequence matching expression $y_{11}(n)$.

$$y12(n) = \left(yl'(n-3)\,ILa(1) + yl'(n-2)\,ILa(3) + \right. \\ \left. + yl'(n-1)\,ILa(1) + yl'(n)\,ILa(7)\right) \quad (3)$$

Considering DA algorithm discussed in detail in [25] the expressions y_{11} (Eq. 2a) and y_{12} (Eq. 3) are expressed as in Eq. (4a) and Eq. (4b) respectively.

$$y_{12}(n) = \begin{pmatrix} \sum_{m=0}^{7} y'1,m(n)\,ILa(1)2^m + \\ \sum_{m=0}^{7} y'1,m(n-1)\,ILa(3)2^m + \\ \sum_{m=0}^{7} y'1,m(n-2)\,ILa(1)2^m + \\ \sum_{m=0}^{7} y'1,m(n-3)\,ILa(7)2^m \end{pmatrix} \quad (4a)$$

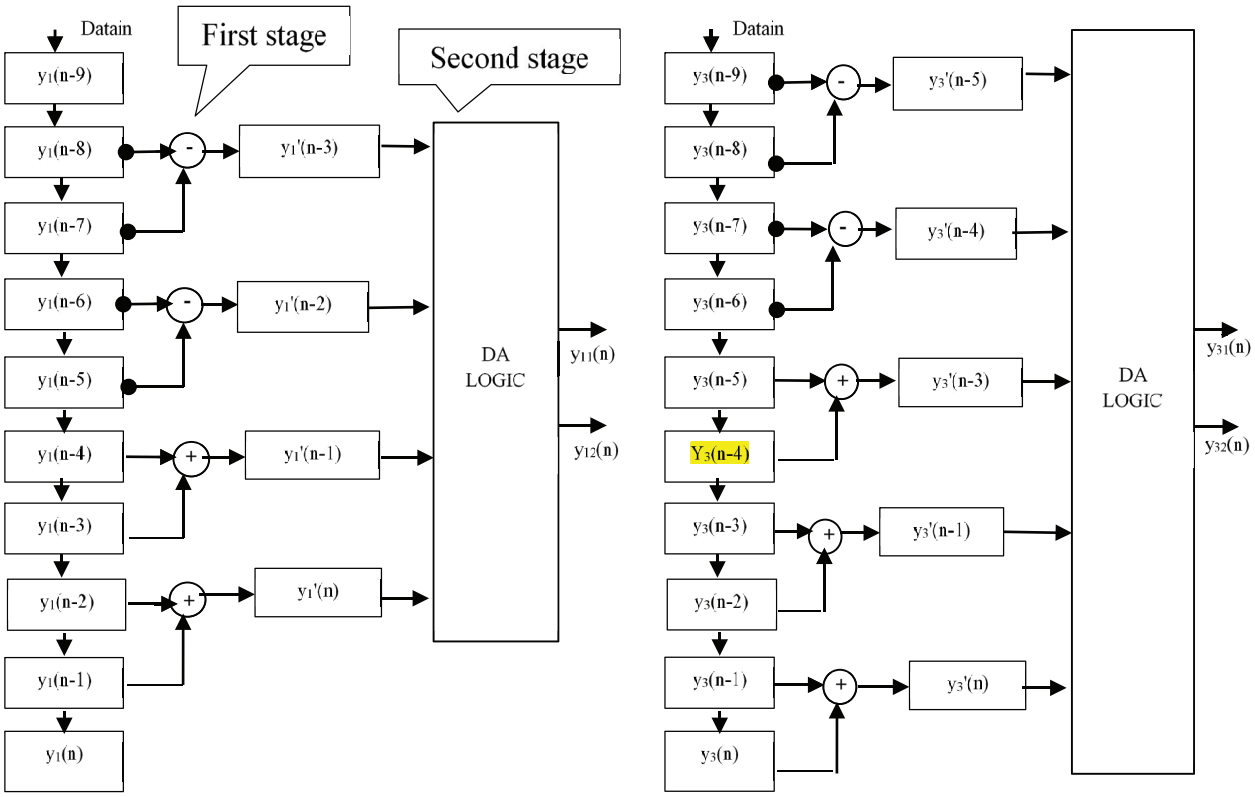

Figure 3: Second stage DTCWT processing (a) y_{11} and y_{12} structure (b) y_{13} and y_{32} structure

$$y_{12}(n) = \begin{pmatrix} \sum_{m=0}^{7} y'1,m(n-3)\,ILa(1)2^m + \\ \sum_{m=0}^{7} y'1,m(n-2)\,ILa(3)2^m + \\ \sum_{m=0}^{7} y'1,m(n-1)\,ILa(1)2^m + \\ \sum_{m=0}^{7} y'1,m(n)\,ILa(7)2^m \end{pmatrix} \quad (4b)$$

As there are four terms {ILa(1), ILa(3), ILa(7), ILa(1)} the contents of LUT for expression Eq. 4(a) is computed that comprises of sixteen terms as shown in Table 2. The address bits {A_0, A_1, A_2, A_3} are used to access the LUT contents. The inputs $y'_{n1,n2}(n)$ is the MSB and $y'_{n1,n2}(n-3)$ is the LSB for the term in Eq. (4a). For the term in Eq. 4(b) $y'_{n1,n2}(n)$ is LSB and $y'_{n1,n2}(n-3)$ is the MSB.

Table 2: LUT contents for DA logic for computing $y_{11\,and}\,y_{12}$

A_3	A_2	A_1	A_0	LUT Contents
0	0	0	0	0
0	0	0	1	ILa (7)]
0	0	1	0	ILa (1)
0	0	1	1	ILa (1) + ILa (7)
0	1	0	0	ILa (3)
0	1	0	1	ILa (3) + ILa (7)
0	1	1	0	ILa (3) + ILa (1)
0	1	1	1	ILa (3) + ILa (1) + ILa (7)
1	0	0	0	ILa (1)
1	0	0	1	ILa (1) + ILa (7)
1	0	1	0	ILa (1) + ILa (1)
1	0	1	1	ILa (1) + ILa (1) + ILa (7)
1	1	0	0	ILa (1) + ILa (3)
1	1	0	1	ILa (1) + ILa (3) + ILa (7)
1	1	1	0	ILa (1) + ILa (3) + ILa (1)
1	1	1	1	ILa (1) + ILa (3)+ ILa (1) + ILa (7)

The multiplexed DA logic that is designed in this work requires single LUT that is used to compute both y_{11} and y_{12} outputs. The contents of LUT are accessed twice in one clock during the positive edge and negative edge. During the positive state of clock the address to LUT is $\{y'_{1,7}(n-3), y'_{1,7}(n-2), y'_{1,7}(n-1), y'_{1,7}(n)\}$ and during negative state of clock the address is $\{y'_{1,7}(n), y'_{1,7}(n-1), y'_{1,7}(n-2), y'_{1,7}(n-3)\}$. Figure 4 presents the multiplexed DA architecture that comprises of input registers, multiplexer, LUT, demultiplexer, accumulator logic and output register. The input data register is of 8 bit width and there are four registers that are serially loaded with the input samples. Loading of four input registers requires 32 clock cycles. The 2:1 multiplexer has two inputs that are connected to the LSBs of four input registers. The multiplexer control signal S_1 is used to select the appropriate LSB to form the address bit for the LUT. If S_1 is '0' $\{y'_{1,7}(n-3), y'_{1,7}(n-2), y'_{1,7}(n-1), y'_{1,7}(n)\}$ form the address {A_0, A_1, A_2, A_3} to the LUT if S_1 is '0', then the address to LUT is $\{y'_{1,7}(n), y'_{1,7}(n-1), y'_{1,7}(n-2), y'_{1,7}(n-3)\}$. S_1 is connected to clock pin and s1 toggles between positive and negative levels ensuring reading of LUT contents twice in one clock cycle. The LUT contents read out are accumulated in the output section comprising of adder and Right Shift (RS) register. The demultiplexer at the output directs the LUT contents to the corresponding accumulator section. The final outputs are stored in the two output registers. Loading data samples into the input register requires 32 clock cycles, and comput-

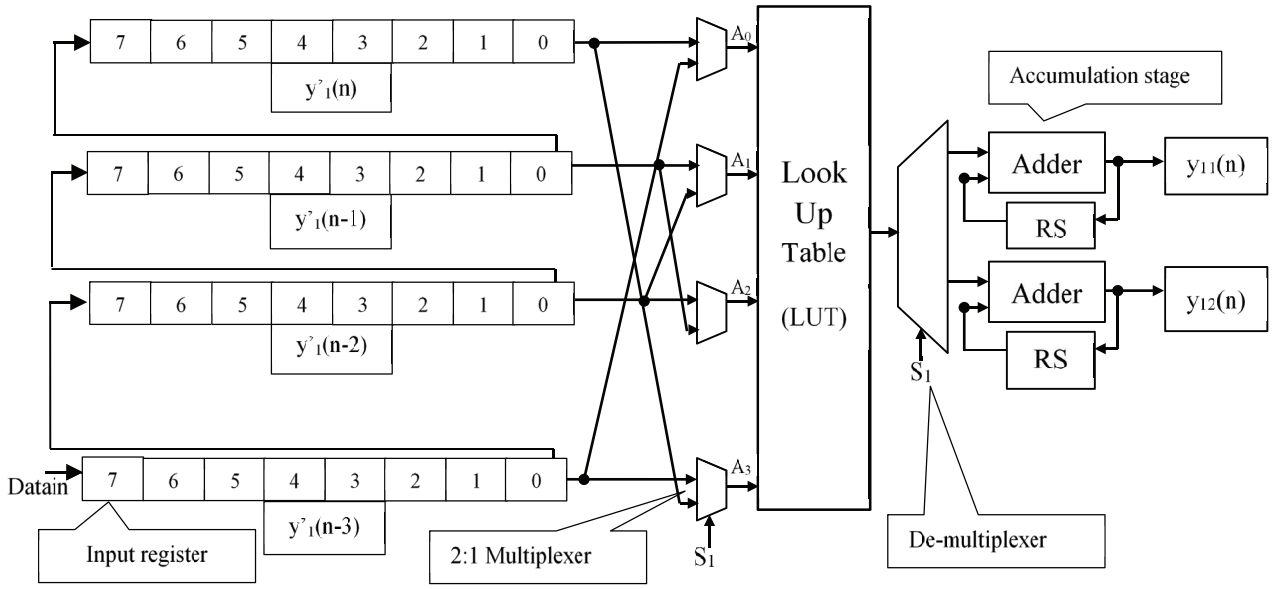

Figure 4: Multiplexed DA logic for y_{11} and y_{12} computation

ing the output samples using DA logic requires 8 clock cycles. After 40 clock cycles the first output of two filters are computed. Computing the consecutive output requires 16 clock cycles (8 clocks for new data sample loading in the input register and 8 clocks for LUT content read and accumulation). Latency for computing y_{11} and y_{12} is 40 clock cycles and throughput is 16 clock cycles for every two outputs.

In the similar way, DA structure for computation of y_{31} and y_{32} is carried out using multiplexed DA logic based on the expressions presented in Eq. (5a) and (5b) respectively.

$$y_{31}(n) = y_3'(n)IL_a(7) + y_3'(n\text{-}1)\,IL_a(1) + \\ + y_3'(n\text{-}2)\,IL_a(3) + y_3'(n\text{-}3)IL_a(1) + y_3'(n\text{-}4)IL_a(0) \quad (5a)$$

$$y_{32}(n) = y_3'(n\text{-}4)IL_a(7) + y_3'(n\text{-}3)\,IL_a(1) + \\ + y_3'(n\text{-}2)\,IL_a(3) + y_3'(n\text{-}1)IL_a(1) + y_3'(n)IL_a(0) \quad (5b)$$

The LUT size for y_{31} and y_{32} is of depth 32 as there are five input registers and five filter coefficients. The latency for computing y_{31} and y_{32} is 48 clock cycles and throughput is 16 clock cycles for every two outputs. Similar structure is designed for computing y_{21} and y_{22} samples and y_{41} and y_{42} of second stage filtering. The advantage of this designed architecture is that the for second stage structure that requires 8 filters are realized using four LUTs. The multiplexed DA logic reduces the LUT resources by 50% as compared with direct DA logic implementation. The multiplexer based DA logic is suitable for FPGA implementation as the number of DSP block sets required for DTCWT computation is reduced and a multiplierless logic is realized. In order to reduce the number of DSP blocks as well as LUT resources for DTCWT computation Zero Pad Logic (ZPL) based structure is also designed and is presented in next section.

3.3 Zero pad architecture

The scaled filter coefficients presented in Table 1 are $\{IL_a(0) = IL_a(9) = 0,\ IL_a(1) = IL_a(2) = IL_a(5) = IL_a(6) = 22,\ IL_a(3) = IL_a(4) = 178,\ IL_a(7) = IL_a(8) = 2\}$. In this design multiplierless structure is designed by using zero padding logic. The filter coefficients are represented in power of 2 as $\{ILa(1) = 22 = 16 + 4,\ IL_a(3) = 178 = 128 + 32 + 16 + 2,\ IL_a(7) = 2\}$. As the filter coefficients are represented in power of 2, multiplying the input data sample by either 2 (2^0), 4(2^2) or 2^N requires to shift left the input data by N. The first stage filter output $y_1(n)$ presented in Eq. 1(a) is rewritten as in Eq. (6), by expanding the filter coefficients in power of 2.

$$\begin{pmatrix} x'(n)[ILa1(1) + ILa2(1)] + \\ x'(n\text{-}1)[ILa1(3) + ILa2(3)] + ILa3(3) + ILa4(3)] + \\ x'(n\text{-}2)[ILa1(1) + ILa2(1)] + x'(n\text{-}3)ILa1(7) \end{pmatrix} \quad (6)$$

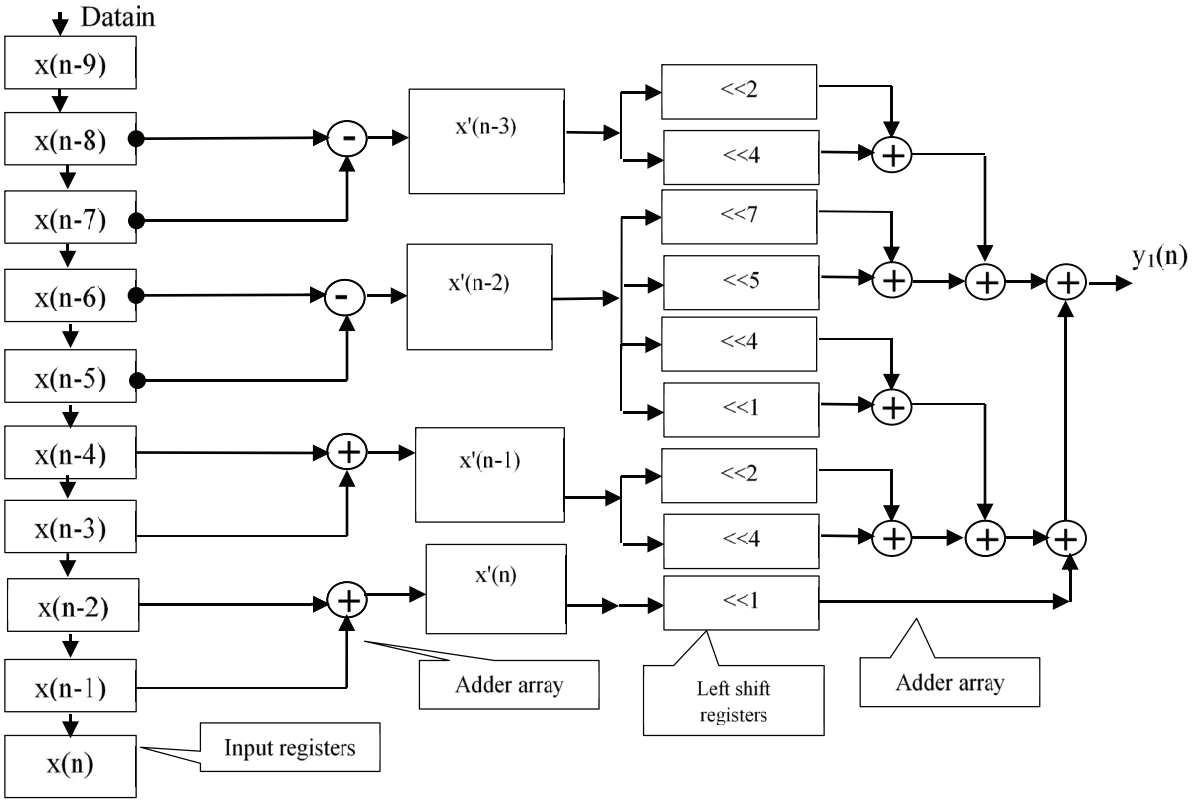

Figure 5: DTCWT filter structure based on zero pad logic

Multiplication of input data sample by 128, or 32, or 16, or 4, or 2 is equivalent of shifting x'(n) by 7, 5, 4, 2 and 1 respectively. This shift operation can also be performed by zero padding 7, 5, 4, 2 and 1 zeros at the LSB of X corresponding to multiplication by 128, or 32, or 16, or 4, or 2 respectively. The expression in Eq. (6) is represented as in Eq. (7) based on zero pad logic.

$$\begin{pmatrix} \{x'(n)ZPL(4)\} + \{x'(n)ZPL(2)\} + \{x'(n-1)ZPL(7)\} + \\ \{x'(n-1)ZPL(5)\} + \{x'(n-1)ZPL(4)\} + \{x'(n-1)ZPL(1)\} + \\ \{x'(n-2)ZPL(4)\} + \{x'(n-2)ZPL(2)\} + \{x'(n-3)ZPL(2)\} \end{pmatrix} \quad (7)$$

ZPL(4) implies shifting or zero padding 4 zeros at the LSB of data sample x'(n). Figure 5 presents the zero pad based architecture for computing y_1 term. The content of intermediate register {x'} is zero padded by 2, 4, 7, 5, 4, 1, 2, 4, 1 of zero bits at the LSBs of each of the intermediate registers. The adder array logic unit is designed to compute the final output and is designed to process data in parallel. The total time requirement from output of intermediate array register to final output is four clock cycles and throughput is 1 clock cycle for every output. Similar structure is designed for computing all 12 filters of 2D DTCWT computation. It is observed that the number of zero padding required is 9 for the architecture shown in Figure 5. Zero padding also requires memory resources on FPGA.

Figure 6 presents improved structure for zero pad logic that reduces the number of zero padding. Eq. (7) is simplified to Eq. (8) by rearranging the terms considering common factors in terms of zero pads. In the rearranged expression, addition operation of intermediate register content is carried out prior to zero padding. From the architecture design the reduced zero pad architecture requires 5 zero pads and eight adders as compared with the structure shown in Figure 5 that requires 9 zero pads and 8 adders.

$$Y_1(n) = \begin{pmatrix} \{[x'(n) + x'(n-1) + x'(n-2)]ZPL(4)\} + \\ \{[x'(n) + x'(n-2) + x'(n-3)]ZPL(2)\} + \\ \{x'(n-1)ZPL(7)\} + \\ \{x'(n-1)ZPL(5) + x'(n-1)ZPL(1)\} \end{pmatrix} \quad (8)$$

The latency In computing filter output is 3 clock cycles and throughput is one clock cycle per filter output.

In this section three different architectures for DTCWT level-1 implementation is presented. The reduced order structure optimizes the number of multipliers and adders required for row processing logic. The multiplexed DA logic reduces the number of LUTs required for computing column processing in DTCWT. The zero padding logic is a multiplierless and LUTless structure that requires adders and storage registers with logic zero contents.

3.4 Systolic array architecture

In order to improve processing speed and throughput for DTCWT computation on an N x N size image systolic array architecture that combines pipelining and parallel processing operations is designed. As every stage of DTCWT processing has four filters, the filter outputs can be generalized as in Eq. (9). Representing the inputs and filter coefficients in 2's complement number representation, subtraction operation is carried out using addition operation.

$$Y1 = Ila(0)(x0+x9) + Ila(1)(x2+x1) + \\ + Ila(3)(x3+x4) + Ila(1)(x5+x6) + Ila(7)(x7+x8) \quad (9a)$$

$$y2 = Ila(0)(x0+x9) + Ila(7)(x2+x1) + \\ +Ila(1)(x3+x4) + Ila(3)(x6+x5) + Ila(1)(x7+x8) \quad (9b)$$

$$y3 = Ila(7)(x0+x1) + Ila(1)(x3+x2) + \\ +Ila(3)(x4+x5) + Ila(1)(x7+x6) + Ila(0)(x8+x9) \quad (9c)$$

$$y4 = Ila(0)(x0+x1) + Ila(1)(x2+x3) + \\ +Ila(3)(x4+x5) + Ila(1)(x7+x6) + Ila(7)(x9+x8) \quad (9d)$$

The data path operation for systolic array structure is designed based on the data flow summarized in Table 3. The filter coefficients for computing outputs of each filter is indicated along with the data input. For computing y1 and y2 as well as y3 and y4 the data input is common, the filter coefficients corresponding to each of the data to be multiplied are indicated in the table.

Figure 7 presents the systolic array structure designed for first stage DTCWT computation based on generic expressions presented in Eq. (9). The SAA architecture consists of four processing elements represented as

Table 3: Data path operation

Clock	y_1 =	y_2 =	y_3=	y_4 =
1	$(x_0 + x_9)$ Ila(0)	$(x_2 + x_1)$ Ila(7)	$(x_0 + x_1)$ Ila(7)	$(x_0 + x_1)$ Ila(0)
2	+ $(x_2 + x_1)$ Ila(1)	+ $(x_3 + x_4)$ Ila(1)	+ $(x_3 + x_2)$ Ila(1)	+ $(x_3 + x_2)$ Ila(1)
3	+ $(x_3 + x_4)$ Ila(3)	+ $(x_6 + x_5)$ Ila(3)	+ $(x_4 + x_5)$Ila(3)	+ $(x_4 + x_5)$Ila(3)
4	+ $(x_6 + x_5)$ Ila(1)	+ $(x_7 + x_8)$ Ila(1)	+ $(x_7 + x_6)$ Ila(1)	+ $(x_7 + x_6)$ Ila(1)
5	+ $(x_7 + x_8)$ Ila(7)	+ $(x_0 + x_9)$ Ila(0)	+ $(x_8 + x_9)$ Ila(0)	+ $(x_8 + x_9)$ Ila(7)

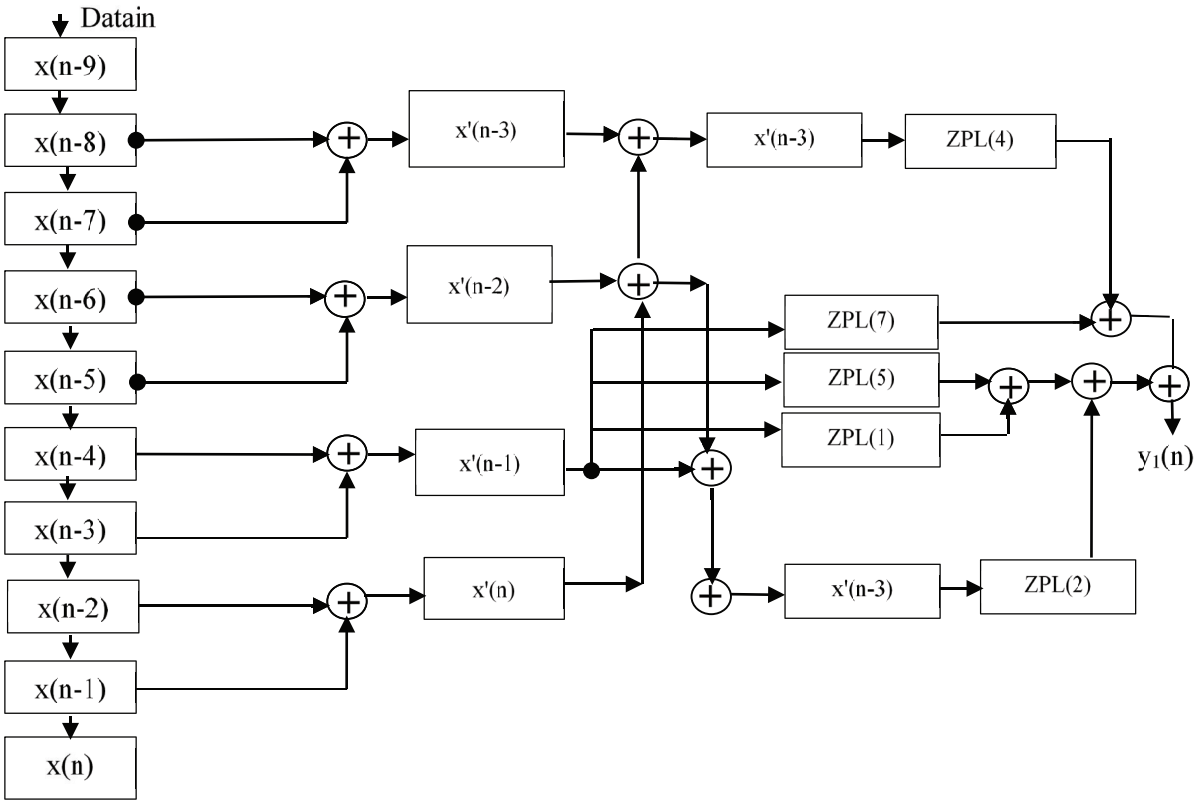

Figure 6: Reduced zero pad logic based DTCWT filter structure

{PE0, PE1, PE2 and PE3}. The data sequence into the SAA structure for filter coefficients is from left to right and for input samples is from bottom to top. From the data sequence listed in table it is found that for computation of y1 happens in PE2 with the two input vectors {c_0, c_1, c_2, c_3, c_4} and {Ila(0), Ila(1), Ila(3), Ila(1), Ila(7)} entering the PE2 from left to right and bottom to top respectively. Similarly computation is performed in PE0, PE1 elements for computing output samples y3 and y1. For computing y2 it is found that the data sequence required is { c_1, c_2, c_3, c_4, c_0} and {Ila(7), Ila(1), Ila(3), Ila(1), Ila(0)}. As the input data and filter coefficients enter PE3

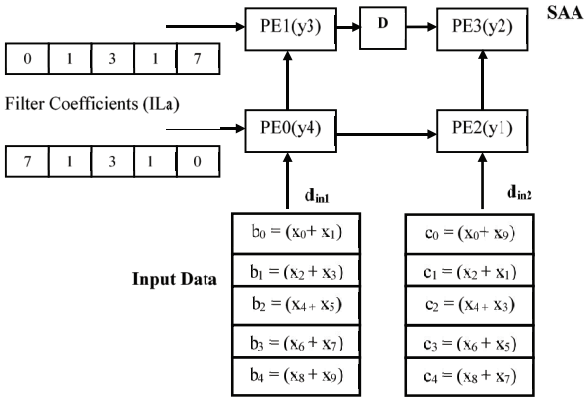

Figure 7: Systolic array structure for stage 1 DTCWT computation

there is a mismatch in data sequence, this is addressed by introducing delay element (D) in the data path between PE1 and PE3 for filter coefficients. This delay ensures that the correct data sequence enters the PE3 element to compute y_2 output.

The intermediate data computation and final output computation of systolic array structure is presented in Table 4. The PE0 generates the first output at 5th clock cycle, PE1 and PE2 generates the y_3 an y_1 outputs respectively at 6ht clock cycle and the PE3 generates y_2 output at 7th clock cycle. New Set of Data (ND2) is processed from 6th clock cycle onwards in PE0 similarly data processing is carried out in all other Pes. The latency of SAA structure is 5 clock cycles and throughput is 1 clock cycle. At every clock multiplication and addition operation is carried out which can be realized using any of the three architectures designed in previous sections.

3.5 Comparison of DTCWT architectures

Table 5 compares the performance metrics of three different architectures (A1, A2 & A3) designed in this paper in terms of arithmetic blocks, latency and throughput. From the comparisons presented, multiplexed DA and zero pad architecture are realized without multipli-

Table 4: Data computation in SAA

Clock	PE0	PE1	PE2	PE3
1	b_0.Ila(0)	0	0	0
2	b_0.Ila(0)+ b_1.Ila(1)	b_0.Ila(7)	c_0. Ila(0)	0
3	b_0.Ila(0)+b_0.Ila(1)+ b_2.Ila(3)	b_0.Ila(7)+ b_1.Ila(1)	c_0.Ila(0)+c_1. Ila(1)	0
4	b_0.Ila(0)+b_1.Ila(1)+ b_2.Ila(3)+ b_3.Ila(1)	b_0.Ila(7)+b_0.Ila(1)+ b_2.Ila(3)	c_0.Ila(0)+c_1.Ila(1)+ c_2. Ila(3)	c_1. Ila(7)
5	b_0.Ila(0)+b_1.Ila(1)+ b_2.Ila(3)+ b_3.Ila(1)+ b_4.Ila(7) = y4	b_0.Ila(7)+b_1.Ila(1)+ b_2.Ila(3)+ b_3.Ila(1)	c_0.Ila(0)+c_1.Ila(1)+ c_2.Ila(3)+ c_3.Ila(1)	c_1.Ila(7)+ c_2.Ila(1)+
6	ND2	b_0.Ila(7)+b_1.Ila(1)+ b_2.Ila(3)+b_3.Ila(1)+ b_4.Ila(0) = y3	c_0.Ila(0)+c_1.Ila(1)+ c_2.Ila(3)+ c_3.Ila(1)+ c_4.Ila(7) = y1	c_1.Ila(7)+ c_2.Ila(1)+ c_3.Ila(3)
7	ND2	ND2	ND2	c_1.Ila(7)+ c_2.Ila(1)+ c_3.Ila(3)+ c_4.Ila(1)=y2
8	ND2	ND2	ND2	0
9	ND2	ND2	ND2	ND2
10	ND2	ND2	ND2	ND2

ers and hence when implemented on FPGA will occupy less area in terms of LUTs available in Configurable Logic Blocks (CLB).

Table 5: Performance metric comparison

Metrics	A1	A2	A2
Type	Reduced order	Multiplexed DA	Zero pad
DTCWT stage	Stage 1	Stage 1	Stage 1
Adders	28	6	16
Multipliers	16	-	-
Shifters	-	-	18
Memory (SISO)	1	2	2
Intermediate Registers	16	9	16
LUTs	-	2	-
Latency	13	40	3
Throughput	4 output/ 4 Clocks	2 output/16 clocks	1 output/ 1 clocks

The number of adders in zero pad architecture is 62.5% higher than multiplexer based DA logic. The CLBs in FPGA comprises of LUTs and multiplexers hence implementing multiplexed DA on FPGA will occupy minimum resources. The latency and throughput of multiplexer based DA is 31.81% and 58% higher respectively than zero pad logic. Considering higher processing speed in terms of throughput and reduced area requirement the zero pad architecture is recommended as compared with direct implementation. For optimized area requirement and implementation of multi stage DTCWT processing multiplexed DA logic is recommended. In this work, the systolic array architecture designed in section 3.4 is implemented considering both multiplexed DA and zero pad logic for computing level-1 DTCWT.

4 FPGA Implementation

In this work, Spartan-6 FPGA development kit and Virtex-5 development kit is selected for implementation of DTCWT architecture. For validation of proposed architecture, two input images of size 256 x 256 have been chosen. Verilog code for DTCWT computation based on systolic array architecture is developed to compute level 1 DTCWT sub bands. The real and imaginary sub bands of both images are fused in the wavelet domain and the inverse transformation is performed to generate the fused image. Two images are stored in the ROM of Spartan-6 device by storing the 256 x 256 gray scale image in *.coe file and loading the file through JTAG mode while programming. The two images are read into the systolic array structure for processing and the fused image is read out through VGA controller into the PC monitor for display. Figure 8 presents the FPGA setup for architecture validations. With the proposed setup validation of DTCWT, inverse DTCWT and complexity in terms of image processing using DTCWT is evaluated. Figure 9 presents the implementation results of processing of two images based on simple fusion algorithm. The first image is a visible image and the second image is IR image that are fused in the wavelet domain by combining the DTCWT sub bands and performing inverse DTCWT operation.

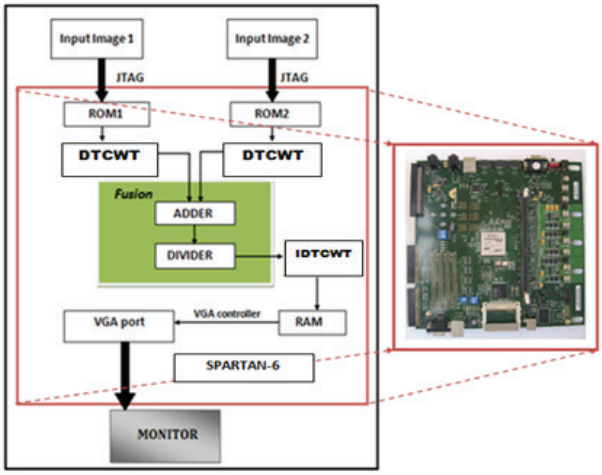

Figure 8: Evaluation of DTCWT on FPGA platform

The DTCWT architecture design is synthesized in the Xilinx tool, post place, map and route simulation also have been carried out. The design is optimized for power, area and timing. RTL synthesized block diagram of DTCWT is obtained using Xilinx ISE and the synthesis report is analyzed for estimation the performances of the proposed design in terms of area, speed and power. Table 6 compares the performances of three different architectures designed in this work. Multiplexed DA logic is found to operate at maximum frequency of 489.896 MHz and occupies LUTs less than reduced order logic. Zero pad architecture requires less than 232 LUTs and the maximum operating frequency is limited to 352.889 MHz.

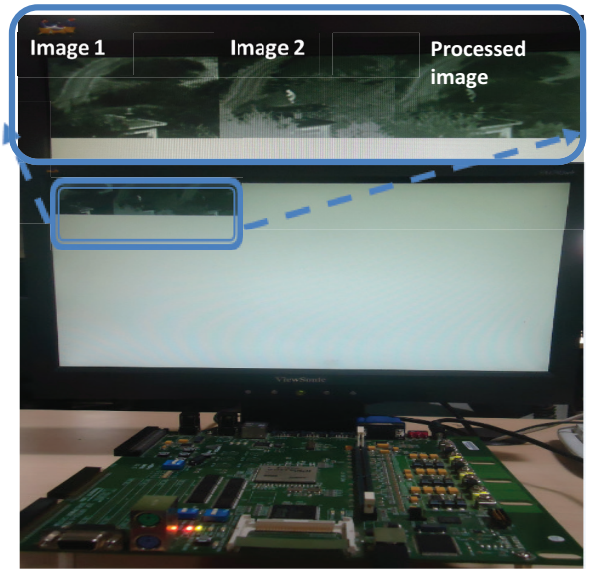

Figure 9: FPGA implementation of DTCWT based Image Fusion

Table 6: FPGA implementation of DTCWT architecture

FPGA parameters	Single filter	Two filter	First stage DTCWT
Multiplexed DA based Logic (data size 256)			
Slice Registers (Total 126800)	87	168	376
Slice LUTs (63400)	88	170	373
IOBs	24	37	68
DSP48A (240)	1	2	4
Max. Frequency (MHz)	498.262	498.262	489.896
Total power (W)	0.037	0.037	0.082
Zero pad logic (data size 256)			
Slice Registers (Total 126800)	54	106	244
Slice LUTs (63400)	52	122	232
IOBs	24	37	68
DSP48A (240)	2	4	8
Max. Frequency (MHz)	348.91	348.91	352.889
Total power (W)	0.036	0.037	0.081
Reduced order logic (data size 256)			
Slice Registers (Total 126800)	127	242	476
Slice LUTs (63400)	124	238	472
IOBs	24	37	68
DSP48A (240)	8	16	32
Max. Frequency (MHz)	332.22	332.22	325.2
Total power (W)	0.044	0.044	0.092

Power dissipation in all three architectures is limited to less than 10 mW and the results obtained are for single stage DTCWT processing. Table 7 compares the performances of SAA based architecture for computation of 256 x 256 image decomposition using DTCWT and the results are compared with direct implementation based on convolution operation.

Table 7: Comparison of hardware requirements

	SAA with Multiplexed DA	DTCWT [25]	Direct DTCWT
Number of Slice Registers	3672	4112	7482
Number of Slice LUTs	3173	4091	7224
Total power (W)	1.5702	1.71111	2.00207
Max. Frequency (MHz)	321.89	289.12	212.67

The hybrid DA architecture discussed in [25] optimizes area utilization on CLBs and DTCWT structure is implemented on Virtex-5 FPGA for four filters. The algorithm

proposed in [25] is modeled and is extended for DTCWT computation of 256 x 256 image and the results are compared with proposed SAA architecture based on multiplexed DA logic. From the results obtained, the SAA architecture consumes less than 11% power compared with existing methods for DTCWT implementation. The operating frequency of processing 256 x 256 on Virtex-5 platform is 9% faster with less than 12% of FPGA resources occupied.

5 Conclusion

In this paper, we have proposed three architectures for DTCWT computation optimizing area and timing requirement. Memory efficient architecture compatible with FPGA architecture is designed based on multiplexed DA logic that computes four filter outputs using two LUT structures. The zero pad algorithm design reduces arithmetic operations and the reduced order architecture reduces arithmetic operations. Systolic array based architecture proposed processes 256 x 256 image computing 2D DTCWT and 2D inverse DTCWT. The proposed architecture is implemented on FPGA and is demonstrated to achieve higher performance metrics in terms of speed and area on Virtex-5 and Spartan-6 FPGA. The proposed architecture for computing DTCWT sub bands operating at 321.89 MHz is suitable for high speed image processing applications such as image registration and image fusion for surveillance and remote sensing. With images acquired having high resolution, the proposed DTCWT architecture could be extended to process images of size greater than 1024 x 1024 and multiple level decomposition can be performed. Systolic array architecture design can be extended to compute multi-level DTCWT decomposition.

6 References

1. Nick Kingsbury "Image processing with complex wavelets" Philosophical Transactions of the Royal Society London A, 357 no. 1760:2543–2560, 1999 https://doi.org/10.1098/rsta.1999.0447
2. Ioana Adam "Complex Wavelet Transform: application to denoising" PhD thesis, Politehnica University of Timisoara
3. Ivan W. Selesnick, Richard G. Baraniuk, and Nick G. Kingsbury "The Dual-Tree Complex Wavelet Transform" IEEE Signal Processing Magazine, 22(6), 123–151, 2005. http://dx.doi.org/10.1109/MSP.2005.1550194
4. S. C. Olhede and Georgios Metikas "The HyperanalyticWavelet Transform" Technical report, Imperial College Statistics Section, 2006. https://doi.org/10.1109/TSP.2009.2023397
5. N.G. Kingsbury "Complex wavelets for shift invariant analysis and filtering of signals" Journal of Applied and Computational Harmonic Analysis, 10, no.3, 234–253, 2001 https://doi.org/10.1006/acha.2000.0343
6. N G Kingsbury "The dual-tree complex wavelet transform: a new efficient tool for image restoration and enhancement" In Proc. European Signal Processing Conference, EUSIPCO 98, Rhodes, 319–322, 1998 https://doi.org/10.5281/zenodo.36900
7. Ivan W. Selesnick "The Double-Density Dual-Tree DWT" IEEE Transactions on Signal Processing, 52, No. 5, 1304–1314, 2004. https://doi.org/10.1109/TSP.2004.826174
8. M. Salagean and I.Firoiu "Anomaly Detection on Network Traffic Based on Analytical Discrete Wavelet Transform" In Proceedings of International Conference Communications 2010, Bucharest, 2010 https://doi.org/10.1109/ICCOMM.2010.5509071
9. Das, A. Hazra, and S. Banerjee, "An Efficient Architecture for 3-D Discrete Wavelet Transform," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 20, No. 2, 286-296, 2010. https://doi.org/10.1109/TCSVT.2009.2031551
10. Lan, X., Zheng, N., Liu, Y. "Low-power and high-speed VLSI architecture for lifting-based forward and inverse wavelet transform" IEEE Trans. Consum. Electron., 51, (2), 379–385, 2005 https://doi.org/10.1109/TCE.2005.1467975
11. S Barua, JE Carletta, KA Kotteri, AE Bell, "An efficient architecture for lifting-based two-dimensional discrete wavelet transforms" J. Integration, VLSI J. 38(3), 341–352, 2005 https://doi.org/10.1145/988952.988967
12. Martina, M., Masera, G. "Multiplierless, Folded 9/7-5/3 Wavelet VLSI Architecture" IEEE Trans. Circuits Syst. II: Express Briefs, 54, (9), 770–774, 2007 https://doi.org/10.1109/TCSII.2007.900354
13. Seo, Y.H., Kim, D.W. "VLSI architecture of line-based lifting wavelet transform for motion JPEG 2000" IEEE J. Solid-State Circuits., 42, 431–440, 2007 https://doi.org/10.1109/JSSC.2006.889368
14. H Varshney, M Hasan, S Jain, "Energy efficient novel architectures for the lifting-based discrete wavelet transform" IET Image Process. 1(3), 305–310, 2007 https://doi.org/10.1049/iet-ipr:20060140
15. M. Jiang and D. Crookes, "Area-efficient high-speed 3D DW processor architecture" *Electron. Lett.*, vol. 43, no. 9, 502–503, 2007 https://doi.org/10.1049/el:20070201

16. W. Zhang, Z. Jiang, Z. Gao, and Y. Liu, "An efficient VLSI architecture for lifting-based discrete wavelet transform" *IEEE Trans. Circuits Syst.II,* vol. 59, no. 3, 158–162, 2012 https://doi.org/10.1109/TCSII.2012.2184369
17. K. Mohanty, A. Mahajan, and P. K. Meher, "Area- and power efficient architecture for high-throughput implementation of lifting 2-D DWT" *IEEE Trans. Circuits Syst. II*, vol. 59, no. 7, 434–438, 2012 https://doi.org/10.1109/TCSII.2012.2200169
18. SM Aziz, DM Pham, "Efficient parallel architecture for multi-level forward discrete wavelet transform processors" J. Comp. Elect. Eng. 38, 1325–1335, 2012 https://doi.org/10.1016/j.compeleceng.2012.05.009
19. Anand D Darji, Shailendra Singh Kushwah, Shabbir N Merchant and Arun N Chandorkar, "High-performance hardware architectures for multi-level lifting-based discrete wavelet transform" *EURASIP Journal on Image and Video Processing,* 2014, https://doi.org/10.1186/1687-5281-2014-47 https://doi.org/10.1186/1687-5281-2014- 47
20. Anand Darji, Arun R., Shabbir Noman Merchant, Arun Chandorkar, "Multiplier-less pipeline architecture for lifting-based two-dimensional discrete wavelet transform" IET Computers & Digital Techniques, Vol. 9, Iss. 2, 113–123, 2015 https://doi.org/10.1049/iet-cdt.2013.0167
21. B.K.N.Srinivasarao and Indrajit Chakrabarti, "High Speed VLSI Architecture for 3-D Discrete Wavelet Transform", Sep 2015 downloaded from https://www.researchgate.net/publication/281895412_High_Speed_VLSI_Architecture_for_3-D_Discrete_Wavelet_Transform https://doi.org/10.1109/VLSI-DAT.2016.7482578
22. Hongda Wang and Chiu-Sing Choy, "Systolic Array based VLSI architecture for high throughput 2-D DWT" IEEE International Conference on Electron devices and solid state circuits (EDSSC), 100-103, 2016 https://doi.org/10.1109/EDSSC.2016.7785219
23. Chakraborty, D. Chakraborty, and A. Banerjee, "A Multiplier less VLSI Architecture of Modified Lifting Based 1D/2D DWT using Speculative Adder", International Conference on Communication and Signal Processing, April 6-8, India, 2017 https://doi.org/10.1109/ICCSP.2017.828671624.
Rakesh Biswas, Siddarth Reddy Malreddy, and Swapna Banerjee, "A High-Precision Low-Area Unified Architecture for Lossy and Lossless 3D Multi-Level Discrete Wavelet Transform" IEEE transactions on circuits and systems for video technology, vol. 28, no. 9, 2386-2396, 2018 https://doi.org/10.1109/TCSVT.2017.2721113
25. S. S. Divakara, Sudarshan Patilkulkarni, Cyril Prasanna Raj, "High Speed Area Optimized Hybrid DA Architecture for 2D-DTCWT" International Journal of Image and Graphics Vol. 18, No. 1, 2018 https://doi.org/10.1142/S0219467818500043

Copyright © 2019 by the Authors. This is an open access article distributed under the Creative Commons Attribution (CC BY) License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Arrived: 06. 03. 2019
Accepted: 02. 09. 2019