

## High performance DA-based DCT, DWT and DHT

K.Satya Sujith<sup>1</sup>, M.Lavanya Latha<sup>2</sup>

<sup>1,2</sup>(Electronics And Communication Engineering, Audisankara Institute of Technology/J.N.T.U.,INDIA)

---

**Abstract:** Thee transform coding is a major block in any compression technique. Transform coding helps in decorrelating the input data and for energy compaction which are required to implement compression. DCT is standard transform used in JPEG image compression standard and DWT is used in JPEG2000 image compression standard. The DHT is a transform which provides flexibility in terms of hardware as well as reduces complexity in evaluation inverse transform because it is inverse is same as forward transform hence recently DHT is also using in image compression techniques so low hardware required, high speed architecture is required for DCT, DWT and DHT. In this paper DA based optimized adder tree architecture was presented

**Keywords:** Adders, DCT- Discrete Cosine Transform, DWT- Discrete Wavelet Transform, DHT- Discrete Hartley Transform, DA- Distributed Arithmetic, OAT- optimized adder tree.

---

### I. Introduction

Today we are talking about digital networks, digital representation of images, movies, video, TV, voice, digital library-all because digital representation of the signal is more robust than the analog counterpart for processing, manipulation, storage, recovery, and transmission over long distances, even across the globe through communication networks. In recent years, there have been significant advancements in processing of still image, video, graphics, speech, and audio signals through digital computers in order to accomplish different application challenges. As a result, multimedia information comprising image, video, audio, speech, text, and other data types has the potential to become just another data type. Development of efficient image compression techniques continues to be an important challenge to us, both in academia and in industry. Digital image compression is the most important part in the multimedia applications which aims to reduce the number of bits in an image data for its efficient storage (less storage area). JPEG based still image compression follows three steps i.e. transform, quantization and coding to compress an image [1]. Reverse process comprising decoding, quantization and inverse transform is used for image de-compression [2, 3]. Discrete cosine transform (DCT) is used to transform the image from spatial domain to frequency domain. During quantization less important frequencies are discarded and is termed as loss image compression. Brace well has drawn attention to the discrete Hartley transform (DHT) as a substitute for the Discrete Fourier transform (DFT) [4, 5]. Many applications of DHT in signal processing and communications have been presented in the literatures [6-8]. DHT is used in JPEG based image compression.

The discrete wavelet transform (DWT) has been widely used in many areas of science and engineering, e.g., signal and image processing, bio-informatics, geophysics, and meteorology etc. for the applications involving compression and analysis of various forms of data. The well-known image coding standards, namely, MPEG-4 and JPEG2000 have adopted DWT as the transform coder due to its remarkable advantages over the other transforms. Multiplier-less hardware implementation approach provides a kind of solution to this problem due to its scope for lower hardware-complexity and higher throughput of computation. Several designs have been proposed for the multiplier-less implementation of DWT based on the principle of distributed arithmetic (DA).

In section-II DA principle is explained in brief, in section-III optimized adder tree is given

### II. MATHEMATICAL DERIVATION OF DISTRIBUTED ARITHMETIC

Distributed Arithmetic (DA) has been one of the popular techniques to compute the inner product equation in many DSP FPGA applications. It is applicable in cases where the filter coefficients are known a priori. The inner sum of products is rearranged so that the multiply and accumulate (MAC) operation is reduced the inner product is an important tool in digital signal processing applications. It can be written as follows:

$$Y=A^T X=\sum_{i=1}^{L-1} A_i X_i \quad \text{--- (1)}$$

Where  $A_i$ ,  $X_i$  and  $L$  are  $i$ th fixed coefficient,  $i$ th input data, and number of inputs, respectively. Assume that coefficient  $A_i$  is  $Q$ -bit two's complement binary fraction number. Equation (1) can be expressed as follows:

$$Y = [2^0 \ 2^{-1} \ 2^{-2} \ \dots \ 2^{-(Q-1)}] \begin{bmatrix} A_{10} & A_{20} & \dots & A_{L0} \\ A_{11} & A_{21} & \dots & A_{L1} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1(Q-1)} & A_{2(Q-1)} & \dots & A_{L(Q-1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{bmatrix}$$

$$Y = [2^0 \ 2^{-1} \ 2^{-2} \ \dots \ 2^{-(Q-1)}] \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{(Q-1)} \end{bmatrix}$$

$A_{i,j}$  stay between  $[1, 0]$  Note that  $y_0$  may be 0 or a negative number due to two's complement representation. In (2),  $y_0$  can be calculated by adding all  $X_i$  values when  $A_{i,j}=1$  and then the transform output  $Y$  can be obtained by shifting and adding all nonzero  $y_i$  values. Thus the inner product computation in (1) can be implemented by using shifting and adders instead of multipliers. Therefore, low hardware cost can be achieved by using DA-based architecture.

### III. OPTIMIZED ADDER TREE ARCHITECTURE

In general, the shifting and addition computation uses a shift-and-add operator in VLSI implementation in order to reduce hardware cost. However, when the number of the shifting and addition words increases, the computation time will also increase. Therefore, the shift-adder-tree (SAT) presented operates shifting and addition in parallel by unrolling all the words needed to be computed for high-speed applications. However, a large truncation error occurs in SAT, and optimized adder tree architecture is proposed in this brief to compensate for the truncation error in high-speed applications

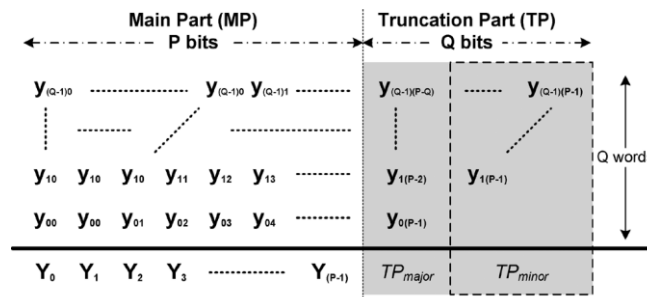


Fig:1: Q,P bit words shifting and addition operations in parallel.

In Fig. 1, the  $Q$   $P$ -bit words operate the shifting and addition in parallel by unrolling all computations. Furthermore, the operation in Fig. 1 can be divided into two parts: the main part (MP) that includes most significant bits (MSBs) and the truncation part (TP) that has least significant bits (LSBs). a large truncation error occurs due to the neglecting of carry propagation from the TP to MP.

The proposed optimized adder tree architecture is illustrated in Fig. 2 for  $(P,Q)=(12,6)$ , where block FA indicates a full-adder cell with three inputs (a, b, and c) and two outputs, a sum (s) and a carry-out (co). Also, block HA indicates half-adder cell with two inputs (a and b) and two outputs, a sum (s) and a carry-out (co).

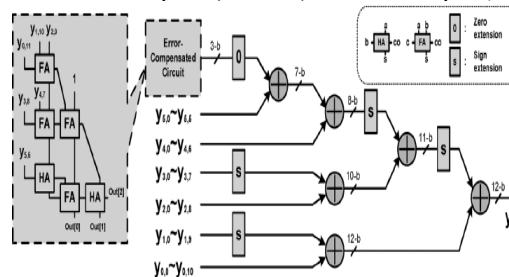


Fig: 2 proposed OPTIMIZED ADDER TREE architecture

**IV. PROPOSED 8X8 2-D DCT DESIGN**

The 1-D DCT employs the DA-based architecture and the proposed Optimized adder tree to achieve a high-speed, small area, and low-error design. The 1-D 8-point DCT can be expressed as follows:

$$Z_n = \frac{1}{2} \sum_{m=0}^{N-1} X_m \times \cos\left(\frac{(2m+1)n\pi}{16}\right)$$

Where  $X_m$  denotes the input data;  
 $Z_n$  denotes the transform output.

By neglecting the scaling factor 1/2, the 1-D 8-point DCT in above equation can be divided into even and odd parts:  $Z_e$  and  $Z_o$  as listed in below equations, respectively

$$Z_e = \begin{bmatrix} Z_0 \\ Z_2 \\ Z_4 \\ Z_6 \end{bmatrix} \begin{bmatrix} C_4 & C_4 & C_4 & C_4 \\ C_2 & C_6 & -C_6 & -C_2 \\ C_4 & -C_4 & -C_4 & C_4 \\ C_6 & -C_2 & C_2 & -C_6 \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_2 \end{bmatrix}$$

$$Z_o = \begin{bmatrix} Z_1 \\ Z_3 \\ Z_5 \\ Z_7 \end{bmatrix} \begin{bmatrix} C_1 & C_3 & C_5 & C_7 \\ C_3 & -C_7 & -C_1 & -C_5 \\ C_5 & -C_1 & C_7 & C_3 \\ C_7 & -C_5 & C_3 & -C_1 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_2 \end{bmatrix}$$

Where  $C_i = \cos(i\pi/16)$  below shows bit level formulation for  $Z_0$  and  $Z_4$ . Let see  $Z_4$  evaluation

**TABLE: 1: bit level formulation**

$Z_1$		$Z_4$	
weight	value	weight	value
$2^0$	0	$2^0$	$A_1$
$2^{-1}$	$B_0+B_1+B_2$	$2^{-1}$	$A_0$
$2^{-2}$	$B_0+B_1$	$2^{-2}$	$A_1$
$2^{-3}$	$B_0+B_3$	$2^{-3}$	$A_0$
$2^{-4}$	$B_0+B_1+B_3$	$2^{-4}$	$A_0$
$2^{-5}$	$B_0+B_2$	$2^{-5}$	$A_1$

Where  $A_0 = (X_0+X_7)+(X_4+X_3) = a_0+a_3$   
 $A_1 = (X_1+X_6)+(X_2+X_5) = a_1+a_2$   
 $B_0 = (X_0+X_7)-(X_4+X_3) = a_0.a_3$   
 $B_1 = (X_1+X_6)-(X_2+X_5) = a_1.a_2$

Input data  $A_0$  and  $A_1$ , the transform output  $Z_0$  needs only one adder to compute  $(A_0 + A_1)$  and two separated optimized adder trees to obtain the results of  $Z_0$  and  $Z_4$ . Similarly, the other transform outputs  $Z_0$  and  $Z_4$  can be implemented in DA-based forms using 10(=1 + 9) adders and corresponding optimized adder trees. Consequently, the proposed 1-D 8-point DCT architecture can be constructed as illustrated in Fig. 3 using a DA-Butterfly-Matrix, that includes two DA even processing elements (DAEs), a DA odd processing element (DAO) and 12 adders/sub tractors, and 8 optimized adder trees (one optimized adder tree for each transform output  $Z_n$ ). The eight separated optimized adder trees work simultaneously, enabling high-speed applications to be achieved. After the data output from the DA-Butterfly-Matrix is completed, the transform output  $Z$  will be completed during one clock cycle by the proposed OPTIMIZED ADDER TREES. In contrast, the traditional shift-and-add architecture requires  $Q$  clock cycles to complete the transform output  $Z$  if the DA-precision is  $Q$ -bits here is 9 bits

**V. Dwt Using Da**

The Haar wavelet is also the simplest possible wavelet. The technical disadvantage of the Haar wavelet is that it is not continuous, and therefore not differentiable. The Haar wavelet transformation is composed of a sequence of low-pass and high-pass filters, known as a filter bank. In mathematics, the Haar wavelet is a certain sequence of functions.

The haar kernel matrix is given below

$$H_8 = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix} \begin{matrix} \varphi_0(t) \\ \psi_0(t) \\ \psi_{1,0}(t) \\ \psi_{1,1}(t) \\ \psi_{2,0}(t) \\ \psi_{2,1}(t) \\ \psi_{2,2}(t) \\ \psi_{2,3}(t) \end{matrix}$$

The DA is applied to above kernel matrix the coefficient term are evaluated those are applied to OAT to get transformed value.

**Table2: DA based coefficients**

WEIGHT	Z1	Z2	Z3	Z4
-2^0	0	0	0	0
2^-1	0	0	a1-a2	a3-a4
2^-2	A0+A1	a1+a2-a3-a4	0	0
2^-3	0	0	0	b0
2^-4	A0+A1	a1+a2-a3-a4	0	0
2^-5	A0+A1	a1+a2-a3-a4	0	0
2^-6	0	0	0	0
2^-7	A0+A1	a1+a2-a3-a4	0	0
2^-8	0	0	0	0

**VI. S-DHT BASED ON DA**

DHT belongs to the family of frequency transforms that map temporal or spatial functions into frequency functions. The DHT accomplishes this in a manner similar to the better-known Fourier Transform. The significant difference between the Discrete Fourier Transform (DFT) and DHT 's alternative is that the DHT uses only real values, i.e., no complex numbers.

The kernel matrix of DHT I given by

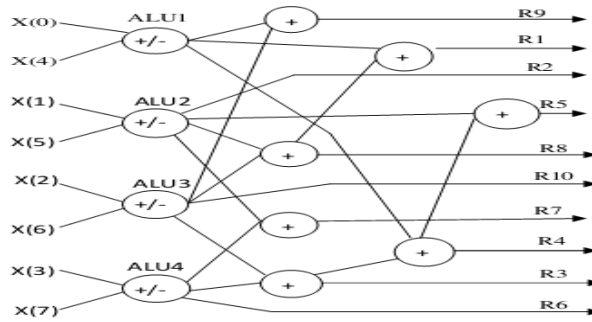
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.0000	1.4140	1.0000	0	-1.0000	-1.4140	-1.0000	0
1.0000	1.0000	-1.0000	-1.0000	1.0000	1.0000	-1.0000	-1.0000
1.0000	0	-1.0000	1.4140	-1.0000	0	1.0000	-1.4140
1.0000	-1.0000	1.0000	-1.0000	1.0000	-1.0000	1.0000	-1.0000
1.0000	-1.4140	1.0000	0	-1.0000	1.4140	-1.0000	0
1.0000	-1.0000	-1.0000	1.0000	1.0000	-1.0000	-1.0000	1.0000
1.0000	0	-1.0000	-1.4140	-1.0000	0	1.0000	1.4140

Apply DA on above kernel then we get coefficients as,

	Y(0)	Y(1)	Y(2)	Y(3)	Y(4)	Y(5)	Y(6)	Y(7)
ALU1	+	-	+	-	+	-	+	-
ALU2	+	-	+	NO	+	-	+	NO
ALU3	+	-	+	-	+	-	+	-
ALU4	+	NO	+	-	+	NO	+	-
Y <sup>-1</sup>	0	0	R3	R10	R7	R2	R8	R3
Y <sup>0</sup>	R5	R1	R5	R4	R5	R9	R5	R9
Y <sup>1</sup>	0	0	0	0	0	R2	0	R6
Y <sup>2</sup>	0	R2	0	R6	0	0	0	0
Y <sup>3</sup>	0	R2	0	R6	0	0	0	0
Y <sup>4</sup>	0	0	0	0	0	R2	0	R6
Y <sup>5</sup>	0	R2	0	R6	0	0	0	0
Y <sup>6</sup>	0	0	0	0	0	R2	0	R6
Y <sup>7</sup>	0	R2	0	R6	0	R2	0	R6

"+" means addition and "-" means subtraction and "NO" means no operation.

Here R1,R2...R10 are given by



**VII. Result And Discussions**

The proposed DCT, DHT and DWT core synthesized by using Xilinx ISE 13.4, simulated by using modelsim6.4d and the Xilinx XC2VP30 FPGA. The corresponding results are shown below

**Table 3: error due to precision taken**

Transform coding scheme	Error after reconstruction(for 256x256 image)
DCT	240
DWT	82.92
DHT	25.68

**Table 4: logic utilization**

Logic utilization	DCT	DWT	DHT
No. of slices	11%	3%	1%
No. of 4 input LUT's	10%	2%	1%
No. of bounded I/O	72%	75%	34%

Here the differentiating parameters are hardware efficiency and energy compaction as well as computation complexity. The computation complexity is less in S-DHT but provides less energy compaction compared to remaining. Throughput achieved is

DCT-380MP/S

Haar wavelet transform-389MP/S

DHT-571MP/S

**DHT modelsim result:**

**DHT matlab result:**

```

MATLAB R2012a
File Edit Debug Parallel Desktop Window Help
Current Folder: C:\Program Files\MATLAB\R2012a\bin
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
>> a=[1 1 1 1; 1 1 1 -1; 1 -1 1 -1; 1 -1 -1 1]
a =
     1     1     1     1
     1     1    -1    -1
     1    -1     1    -1
     1    -1    -1     1

>> b=[7 10 7 26]
b =
     7    10     7    26

>> dht=a*b'
dht =
     50
    -16
    -22
     16
    
```

**DCT modelsim result:**

The screenshot shows the ModelSim Objects window for a simulation named 'dadct'. It lists various simulation objects and their values. The objects are organized into a table with columns for Name and Value.

Name	Value
X0[8:0]	6
X1[8:0]	10
X2[8:0]	26
X3[8:0]	28
X4[8:0]	31
X5[8:0]	7
X6[8:0]	58
X7[8:0]	58
Z0[11:0]	77
Z1[11:0]	-46
Z2[11:0]	5
Z3[11:0]	-30
Z4[11:0]	1
Z5[11:0]	4
Z6[11:0]	-18
Z7[11:0]	11
A4[11:0]	59
B5[11:0]	0000001100
B6[11:0]	0000001100
B7[11:0]	0000001110
B8[11:0]	0000001111
B9[11:0]	0000001101
B10[11:0]	0000001100
B11[11:0]	0000010101
B12[11:0]	0000010101
B13[11:0]	0000010010
B14[11:0]	0000010011
B15[11:0]	0000010110
B16[11:0]	0000010000
B17[11:0]	0000010101
B18[11:0]	0000011100

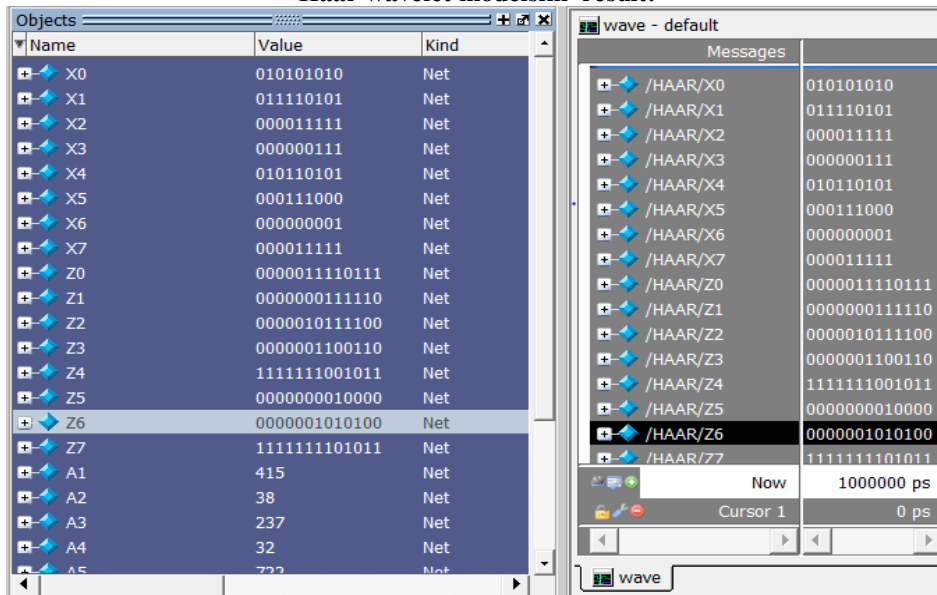
**DCT matlab result:**

```

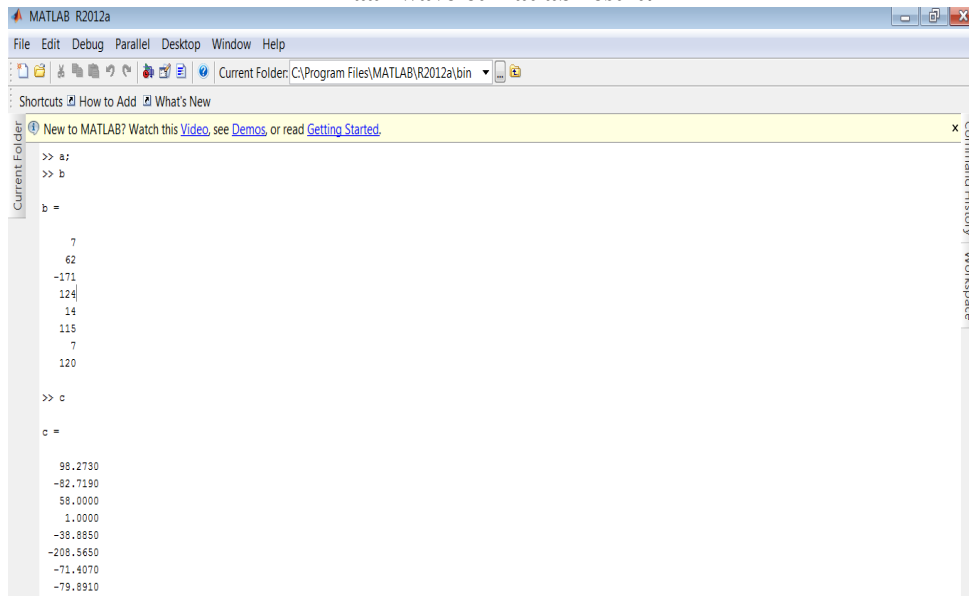
MATLAB R2012a
File Edit Debug Parallel Desktop Window Help
Current Folder: C:\Program Files\MATLAB\R2012a\bin
Shortcuts How to Add What's New
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
>> a=[6 10 26 28 31 7 58 58]
a =
     6    10    26    28    31     7    58    58

>> dct(a)
ans =
    79.1960   -40.4704    9.0067   -25.4201    7.7782    9.7002   -15.2112   17.6315
    
```

**Haar wavelet modelsim result:**



**Haar wavelet matlab result:**



**VI. CONCLUSION**

The paper contributed with specific simplifications in the multiplier stage, by using shift and adds method, which lead to hardware simplification and speed up over architecture.

**References:**

- [1] S. Uramoto, Y. Inoue, A. Takabatake, J. Takeda, Y. Yamashita, H. Yerane, and M. Yoshimoto, "A 100-MHz 2-D discrete cosine transforms core processor," *IEEE J. Solid-State Circuits*, vol. 27, no. 4, pp. 492–499, Apr. 1992.
- [2] S. Yu and E. E. S. , Jr., "DCT implementation with distributed arithmetic," *IEEE Trans. Comput.*, vol. 50, no. 9, pp. 985–991, Sep. 2001.
- [3] A. M. Shams, A. Chidanandan, W. Pan, and M. A. Bayoumi, "NEDA: A low-power high-performance DCT architecture," *IEEE Trans. Signal Process.*, vol. 54, no. 3, pp. 955–964, Mar. 2006.
- [4] C. Peng, X. Cao, D. Yu, and X. Zhang, "A 250 MHz optimized distributed architecture of 2D 8x8 DCT," in *Proc. Int. Conf. ASIC*, 2007, pp. 189–192.
- [5] P. K. Meher, T. S. Srikanthan, J. C. Patra, "Scalable and Modular Memory-Based Systolic Architectures for Discrete Hartley Transform," *IEEE Transactions on Circuits and Systems*, vol.53, no.5, pp. 1065 – 1077, May 2006.
- [6] P. Longa, A. Miri, and M. Bolic, "Modified distributed arithmetic based architecture for discrete wavelet transforms," *Electronics Letters* vol. 44, no. 4, Feb. 2008.
- [7] Y. Wang, J. Ostermann, and Y. Zhang, *Video Processing and Communications*, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.