ISSN XXXX XXXX © 2018 IJESC

IJESC

Research Article Volume 8 Issue No.5

Design and Analysis of Level-1 DWT Architecture using Vedic Multiplier

KritikaNijhawan[1], JayatiShukla[2], PrasheelThakre[3]
Student[1, 2], Assistant Professor[3]
Department of Electronics and Communication
RCOEM, Nagpur, India

Abstract
Discrete Wavelet Transform (DWT) is an effective analysis tool for many real time applications like biomedical signal processing and ultra-wideband wireless communication. The Low Pass filters (LPF) and High Pass filters (HPF) used in DWT are implemented using Vedic Multiplier. Filtering operation can also be seen as convolution of two signals. Convolution can be done using Vedic multiplier which uses one of the sixteen sutras of Vedic mathematics, called UrdhvaTriyagbhyam Sutra. This Sutra is used to increase the speed of multiplication, reduce the area and delay. The DWT architecture has been designed in Xilinx ISE Design suite 14.7 software using VHDL. The stimulation and synthesis are performed on this software and the area and delay for the DWT architecture is computed.

Keywords: Discrete Wavelet Transform, Daubechies db1 (Haar) wavelet coefficients, Vedic Multiplier, UrdhvaTiryagbhyam, FIR filters, Down sampling

I. INTRODUCTION

Fourier transform is the amplitude-frequency representation of a signal. It only gives us the frequency components that exist in the signal. Time-frequency representation of the signal cannot be determined using Fourier transform. It doesn't show the time and frequency information at the same time. It is suited for stationary data. The drawbacks of Fourier transform were overcome by Short Time Fourier transform (STFT). In STFT, the signal is divided into small enough segments, where these segments (portions) of the signal can be assumed to be stationary. For this purpose, a window function *"w"* is chosen. The width of this window must be equal to the segment of the signal where its stationarity is valid. But STFT also has some drawbacks. In STFT, the width of the window remains unchanged. There is also dilemma of resolution which states that for narrow window we get poor frequency resolution but good time resolution and for wide window we get poor time resolution but good frequency resolution. The STFT roots go back to Heisenberg Uncertainty Principle which is originally applied to the momentum and location of moving particles can be applied to time-frequency information of a signal. This principle states that one cannot know the exact time-frequency representation of a signal. These drawbacks were overcome by using wavelet transform which is the time-frequency representation of signal where the multi-resolution analysis is taken into account. In wavelet transform, the width of the window is changed as the transform iscomputed for every single spectral component.

Wavelet transform acts as an analysis tool for many real time applications like data compression, compression of electrocardiograph (ECG) signals, signal processing of self-mixing interferometry (SMI) and speech recognition.

II. LEVEL-1 DISCRETE WAVELET TRANSFORM ARCHITECTURE

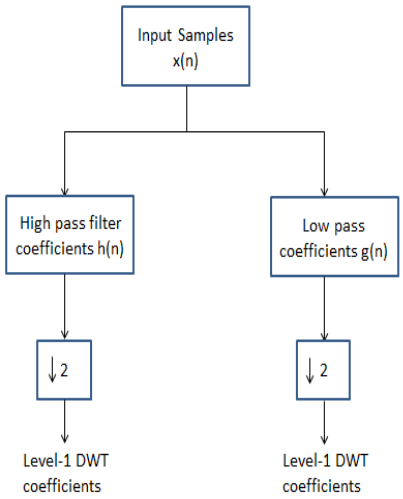


Figure 1: Level-1 DWT Architecture

A system for computation of level-1 discrete wavelet transforms coefficients particularly the Haar (dB1) coefficients has been discussed in this paper.
The procedure starts with passing an input sequence $x(n)$ through a digital low pass filter with impulse response $h(n)$.Filtering a signal corresponds to the mathematical operation of convolution of the signal with the impulse response of the filter. After passing the signal through a low pass filter, half of the samples can be eliminated. This elimination process is done by down sampling. Down sampling by a factor *'n'* reduces the number of samples in the signal *'n'* times.Down sampling by two refers to dropping

every alternate sample of the signal. The scale of the signal gets doubled. The low pass filtering removes the high frequency information, but leaves the scale unchanged. Only the down sampling process changes the scale. Resolution, on the other hand, is related to the amount of information in the signal and hence it is affected by the filtering operation. Low pass filtering removes half of the frequencies, which means losing half of the information. Thus the resolution is halved after the filtering operations. Down sampling after filtering does not affect the resolution as half the number of samples is redundant anyway. Half of the samples can be discarded without any loss of information. Similar procedure is followed when input sequence *x(n)* is passed through a digital high pass filter with impulse response *h(n)*.The FIR filters of level-1 DWT architecture have been designed using Vedic multiplier.

III. ALGORITHM FOR FINDING THE HAAR BASIS FOR FILTER OF ORDER N

1. Determine order *N*of haar basis.
2. Determine $n=log_2 N$
3. Find $p$and $q$where $0 \le p \le n-1$;
   If $p = 0 \rightarrow q = 0 \; or \; 1$
   If $p \neq 0 \rightarrow 1 \le q \le 2^p$
4. Determine k where $k = 2^p + q - 1$
5. Determine $z$where

$z[0,1] \rightarrow [0/N, 1/N \ldots\ldots (N-1)/N]$

6. If $k = 0$ then

$H(z) = \frac{1}{\sqrt{2}}$

7.If $k \neq 0$ then

$$H(z) = \frac{1}{\sqrt{2}} \begin{cases} -2^p \frac{q-1}{2^p} \le z < \frac{q-\frac{1}{2}}{2^p} \\ -2^p \frac{q-\frac{1}{2}}{2^p} \le z < \frac{q-1}{2^p} \\ 0 \qquad \text{Otherwise} \end{cases}$$

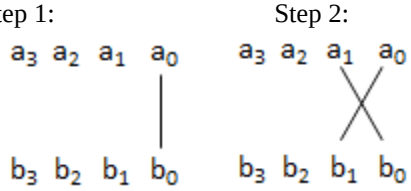
On solving through the above method we get,

$$Hr_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Thus the haar filter coefficients are
Low pass filter coefficients – [0.707, 0.707]
High pass filter coefficients – [0.707, -0.707]

IV. URDHVA TRIYAGBHYAM SUTRA

It is one of the sixteen sutras of Vedic mathematics. It means "vertically and crosswise". This multiplication method is applicable for binary and real numbers.



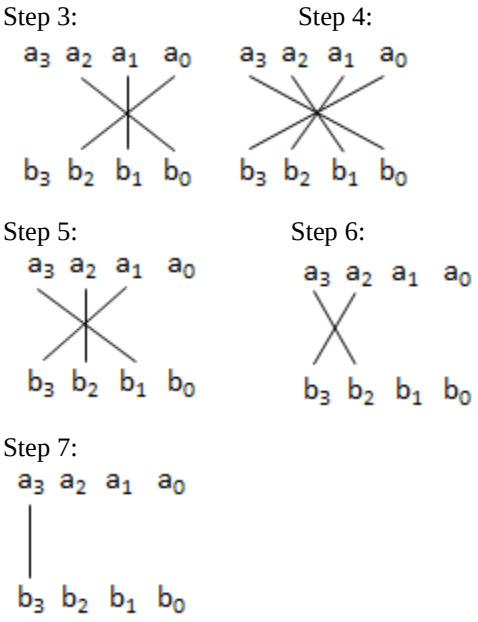


Figure 2: 4 bit Vedic multiplication method

The above steps show the stepwise computation of partial products $[p_0 p_1 p_2 p_3 p_4 p_5 p_6 p_7]$ by multiplication of 4 bit binary numbers $[a_3 a_2 a_1 a_0]$ and $[b_3 b_2 b_1 b_0]$. In the first step, a_0 and b_0 are multiplied and the result is stored in p_0. Similarlyin the second step a_0 and b_1 are multiplied and $b_0$and a_1 are multiplied and a full adder is used for their addition. The sum is stored in $p_1$and carry is given to the next step. Similarly the process continues and we obtain the result.

V. 4 BIT VEDIC MULTIPLIER

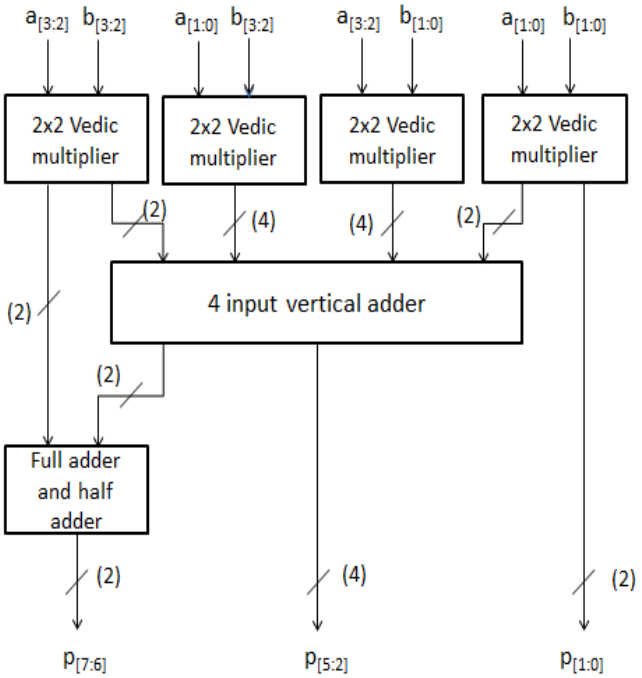


Figure 3: 4x4 bit Vedic multiplier

In this multiplier, we have used 2x2 bit Vedic multipliers, 4 input vertical adder, full adder and half adder. We have considered two 4 bit inputs *a*[3:0]and *b*[3:0] respectively.As per the above block diagram, the partial products *p*[1:0] are the 2 LSBs obtained from Vedic multiplication of *a*[1:0]and

b[1:0]. The 4 input vertical adder adds 4 bits at a time and provides one sum bit and two carry bit each time.The inputs given to 4 input vertical adder are as shown in figure 3. The outputs from vertical adder are partial products *p*[5:2] and the outputs from full adder and half adder circuitry are partial products *p*[7:6].

VI. 8 BIT VEDIC MULTIPLIER

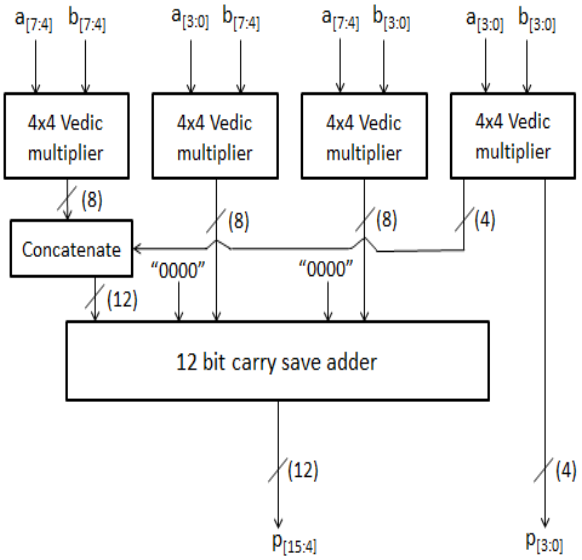


Figure 4: 8x8 Vedic multiplier

In the above multiplier, we have used 4x4 bit Vedic multipliers and 12 bit carry save adder. A carry save adder can add 3 bits at a time. The output of 4x4 bit Vedic multipliers are C_0[7:0], C_1[7:0], C_2[7:0] and C_3[7:0]. C_0[3:0] is directly considered as partial products *p*[3:0]. C_0[7:4] is concatenated with C_4[7:0] to make it a 12 bit vector. C_1[7:0] and C_2[7:0] are zero padded in the MSB to make them a 12 bit vector each. These three 12 bit vectors are given as input to the 12 bit carry save adder. The outputs of 12 bit carry save adder are the partial products*p*[15:4].

VII. 16 BIT VEDIC MULTIPLIER

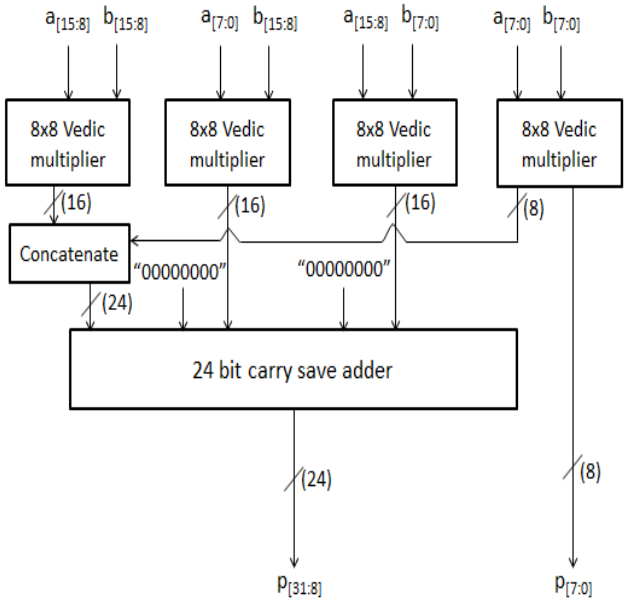


Figure 5: 16x16 Vedic multiplier

In the above multiplier, we have used 8x8 bit Vedic multipliers and 24 bit carry save adder. A carry save adder can add 3 bits at a time. The output of 4x4 bit Vedic multipliers are C_0[15:0], C_1[15:0], C_2[15:0] and C_3[15:0]. C_0[7:0] is directly considered as partial products *p*[7:0]. C_0[15:8] is concatenated with C_4[15:0] to make it a 24 bit vector. C_1[15:0] and C_2[15:0] are zero padded in the MSB to make them a 24 bit vector each. These three 24 bit vectors are given as input to the 24 bit carry save adder. The outputs of 24 bit carry save adder are the partial products *p*[31:8].

VIII. LINEAR CONVOLUTION

Convolution takes two signals and produces a third signal. In linear systems, convolution is used to describe the relationship between three signals of interest: the input signal, the impulse response and the output signal.

If the input and impulse response of a system are $x(n)$ and $h(n)$ respectively, then the convolution is given by the expression,

$$y(n) = x(n) * h(n)$$

$$y(n) = \sum_{-\infty}^{\infty} x(k)h(n-k)$$

In this equation, $x(k)$, $h(n\text{-}k)$ and $y(n)$ represent the input to and output from the system at time n. Here we could see that one of the inputs is shifted in time by a value every time it is multiplied with the other input signal. Linear Convolution is quite often used as a method of implementing filters of various types.

If, $x(n)$ is a M point sequence and $h(n)$ is a N point sequence then, $y(n)$ is a $(M+N\text{-}1)$ point sequence.

Figure 6: Convolution of two sequences

In the above diagram, $x(n)$ is 8 point sequence and $h(n)$ is 2 point sequence. Thus $y(n)$ is 9 point sequence.

Convolution is used for filtering operations and has been implemented using 16 bit vedic multipliers and carry look ahead adder.

IX. DESIGN HAAR FILTER USING VEDIC MULTIPLIER

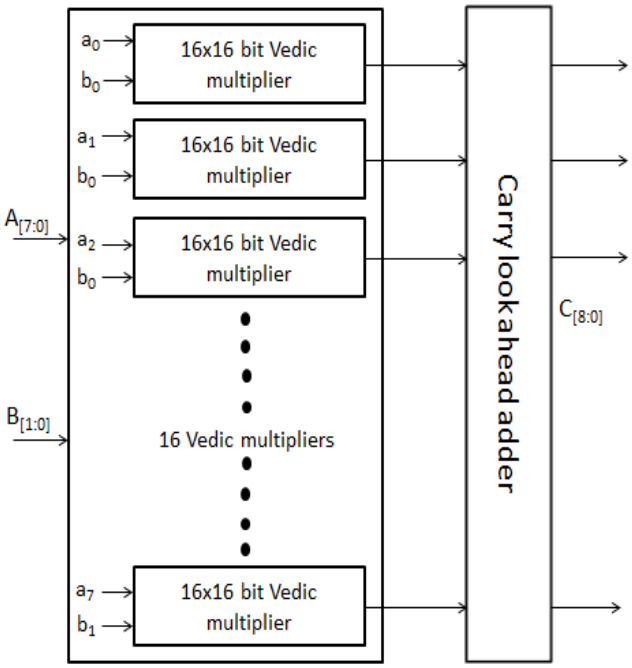


Figure 7: Convolution using 16 bit Vedic multiplier

In the above block diagram, length of input sequence is 8 and length of filter coefficient is 2.Thus the length of output sequence will be 9. Each sample is of 16 bits where the first 8 bits represent the numbers before the decimal point (whole numbers) and the last 8 bits represent the numbers after the decimal point (fraction).We have assigned a separate bit for sign. Thus the above design is applicable for floating point numbers. In total 16 multipliers are required as observed from figure 5. The output of 16 bit multipliers is given to carry look ahead adder for addition of product terms as shown in figure 5. Thus the output of carry look ahead adder is the output sequence of length 9.

From figure 1 it can be observed that the input sequence is given to both High pass filter and low pass filter. The length of high pass filter coefficients and low pass filter coefficients is 2. Thus the length of output sequence obtained from each of the filters is 9.

X. DOWN SAMPLING

After the filtering operation, the samples obtained are down sampled by 2.Down sampling by two refers to dropping every alternate sample of the signal. Thus we have dropped the odd samples and retained the even ones. These even samples are level-1 DWT coefficients.

XI. RESULT

The DWT architecture has been designed in Xilinx ISE Design suite 14.7 software using VHDL.

Example:

The input sequence $x(n)$ = {1, 2, 1, 2, 1, 2, 1, 2}

Low pass filter coefficients $g(n)$ = {0.707, 0.707}

High pass filter coefficients$h(n)$ = {0.707, -0.707}

Output sequence from low pass filter after down sampling by 2 = {0.707, 2.828, 2.828, 2.828, 1.414}

Output sequence from high pass filter after down sampling by 2 = {0.707, -0.707, -0.707, -0.707, -1.414}

Thus the above output sequences are the level-1 DWT coefficients.

Simulation results are as follows:

Device Utilization Summary

Logic Utilization	Used	Available	Utilization
No. of slice registers	570	408000	0%
No. of slice LUTs	10315	204000	5%
No. of fully used LUT-FF pairs	558	10327	5%
No. of bonded IOBs	554	600	90%
BUFGS	6	32	18%

Total Combinational Delay

```
                          Gate    Net
Cell:in->out      fanout  Delay   Delay  Logical Name (Net Name)
----------------------------------------  ------------
 IBUF:I->O            70   0.000   0.659  af5_1_IBUF (af5_1_IBUF)
 LUT4:I0->O            3   0.043   0.534  q11/11/n1/r1/z<3>1 (q11/11/n1/q0<3>)
 LUT4:I0->O            2   0.043   0.500  q11/11/n1/ca<12>1 (q11/11/n1/ca<12>)
 LUT5:I2->O            2   0.043   0.355  q11/11/n1/ca<11>1 (q11/11/n1/ca<11>)
 LUT4:I3->O            2   0.043   0.618  q11/11/n1/ca<10>1 (q11/11/n1/ca<10>)
 LUT6:I0->O            2   0.043   0.410  q11/11/n1/Mxor_p<6>_xo<0>1 (q11/11/q1<6>)
 LUT3:I1->O            2   0.043   0.608  q11/11/n5/FA11/Mxor_S_xo<0>11 (q11/11/n5/FA
 LUT5:I0->O            3   0.043   0.362  q11/11/n5/FA11/Cout1 (q11/11/n5/C3)
 LUT5:I4->O            3   0.043   0.362  q11/11/n5/FA13/Cout1 (q11/11/n5/C5)
 LUT5:I4->O            2   0.043   0.608  q11/11/n5/FA15/Cout1 (q11/11/n5/C7)
 LUT5:I0->O            6   0.043   0.378  q11/11/n5/FA19/Mxor_S_xo<0>11 (q11/11/n5/FA
 LUT6:I5->O            3   0.043   0.362  q11/11/n5/FA17/Mxor_S_xo<0>1 (q11/q1<13>)
 LUT3:I2->O            2   0.043   0.527  q11/15/FA6/Mxor_S_xo<0>1 (q11/15/Y<4>)
 LUT5:I1->O            3   0.043   0.362  q11/15/FA21/Cout1 (q11/15/ca<5>)
 LUT5:I4->O            3   0.043   0.362  q11/15/FA23/Cout1 (q11/15/ca<7>)
 LUT5:I4->O            3   0.043   0.362  q11/15/FA25/Cout1 (q11/15/ca<9>)
 LUT5:I4->O            4   0.043   0.367  q11/15/FA27/Cout1 (q11/15/ca<11>)
 LUT5:I4->O            5   0.043   0.373  q11/15/FA29/Cout1 (q11/15/ca<13>)
 LUT5:I4->O            2   0.043   0.527  q11/15/FA31/Cout1 (q11/15/ca<15>)
 LUT5:I1->O            8   0.043   0.444  q11/15/FA35/Mxor_S_xo<0>11 (q11/15/FA35/Mxo
 LUT3:I1->O            3   0.043   0.625  q11/15/FA34/Mxor_S_xo<0>1 (z11<26>)
 LUT6:I0->O            3   0.043   0.625  sum31/c<27>3_SW0 (N98)
 LUT6:I0->O            3   0.043   0.534  sum31/c<27>3_SW4 (N560)
 LUT5:I1->O            2   0.043   0.410  sum31/c<25>3_SW4_SW1 (N1005)
 LUT5:I3->O            1   0.043   0.495  sum31/c<25>3_SW5 (N903)
 LUT5:I2->O            5   0.043   0.428  sum31/c<29>1 (sum31/c<29>1)
 LUT6:I4->O            2   0.043   0.355  sum31/c<31>_SW0 (N6)
 LUT6:I5->O           64   0.043   0.475  sum31/c<32>1 (sum31/c<32>)
 LDE:GE                    0.161          sum31/cc_29
----------------------------------------
Total                     14.353ns (1.322ns logic, 13.031ns route)
                                   (9.2% logic, 90.8% route)
```

XII. FUTURE SCOPE

This architecture is used in real time applications like speech recognition and data compression.

This architecture can be modified for image compression by giving the input as a gray level matrix and accordingly performing the filtering operations.

XIII. REFERENCES

[1] C.S. Avinash and John Sahaya Rani Alex, "FPGA Implementation of Discrete Wavelet Transform using Distributed Arithmetic Architecture" *2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, T.N., India.6 - 8 May 2015. pp.326-330.

[2]ApekshaA.Narayane and Prof. YogeshWatile, "Review on High Speed Convolution and Deconvolution Algorithm Using Indian Vedic Mathematics"*International Journal of Innovative Research in Computer and Communication Engineering*(An ISO 3297: 2007 Certified Organization) Vol. 4, Issue 2, February 2016

[3] Shivshankar, Pattemsampathkumar "Convolution and Deconvolution Algorithm using Vedic Mathematics with Wallace Adder " *International Journal of Emerging Engineering Research and Technology* Volume 3, Issue 7, July 2015, PP 122-128 ISSN 2349-4395 (Print) & ISSN 2349-4409 (Online)

[4]SayaliShembalkar,Samiksha Dhole, TirupatiYadav and Prof. PrasheelThakre, "Vedic Mathematics Sutras -A Review" *International Conference on Recent Trends in Engineering Science and Technology* (ICRTEST 2017) ISSN: 2321-8169 148 – 155 Volume 5 Issue 1(*Special Issue 21-22 January 2017*)

[5]Samiksha Dhole, SayaliShembalkar, TirupatiYadav, Prof. PrasheelThakre,"Design and FPGA Implementation of 4x4 Vedic Multiplier using Different Architectures"*International Journal of Engineering Research & Technology (IJERT)* ISSN: 22780181I JERTV 6IS040673 Vol. 6 Issue 04, April 2017

[6]AnkurMaloo, JyotiprakashChoudhary and Prof. PrasheelThakre "Design and Analysis of 4x4 Vedic Multiplier using Carry Save Adder and Vertical Adder"*Journal of Integrated Circuits in Electrical Devices* Volume 4, Issue 1, March 2018