ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE
International Journal of Advanced Research in Computer and Communication Engineering

NCDSPICE 2016
National Conference on Digital Signal Processing, Information and Communication Engineering
Govt. Polytechnic College, Kasaragod
Vol. 5, Special Issue 4, November 2016

A High Speed VLSI Architecture for Lifting Based Discrete Wavelet Transform

Nikhitha M[1], Mohamed Salih K.K[2]

ECE Department, Government Engineering College, Trichur, Thrissur, Kerala[1, 2]

Abstract: The Discrete Wavelet Transform (DWT) is well known for its application in image and video compression. Due to its remarkable advantage over the discrete cosine transform (DCT) in image compression, 2-D DWT has been accepted for the JPEG-2000 compression standard. The implementation of 2-D DWT, however, is highly computation-intensive. The lifting scheme is an efficient implementation of wavelet transform. Compared with convolution based DWT, lifting-based architectures not only have lower computation complexity but also require less memory. Modifications are made to lifting scheme in order to reduce critical path delay and size of temporal buffer.

Keywords: Lifting scheme, Discrete Wavelet Transform, Discrete Cosine Transform, VLSI architecture

INTRODUCTION

Wavelet means ‘small wave’. So wavelet analysis is about analyzing signal with short duration finite energy functions. They transform the signal under investigation into another representation which presents the signal in a more useful form. Unlike Fourier transform, we have a variety of wavelets that are used for signal analysis. Choice of a particular wavelet depends on the type of application in hand. Wavelet analysis is a new development in the area of applied mathematics. They were first introduced in seismology to provide a time dimension to seismic analysis that Fourier analysis lacked. In Wavelet Transform, dilations and translations of a mother wavelet are used to perform a spatial frequency analysis on the input data. For spatial analysis, contracted versions of the mother wavelets are used. These contracted versions can be compared with high frequency basis functions in the Fourier based transforms. The relatively small support of the contracted wavelets makes them ideal for extracting local information like positioning discontinuities, edges and spikes in the data sequence, which makes them suitable for spatial analysis. Dilated versions of the mother wavelet, on the other hand, have relatively large supports (the length of the dilated mother wavelet). The larger support extracts information about the frequency behaviour of data. Varying the dilation and translation of the mother wavelet, therefore, produces a customizable time/frequency analysis of the input signal. If the process is done in a smooth and continuous fashion (ie. If scale and position is varied very smoothly) then the transform is called Continuous Wavelet Transform (CWT). All the wavelet functions used in transformation are derived from the mother wavelet through translation (shifting) and scaling (dilation or compression). If the scale and position are changed in discrete steps, the transform is called Discrete Wavelet Transform (DWT).

In the last decade, discrete wavelet transform (DWT) has been successfully used in a wide range of applications, including numerical analysis, signal analysis, image and video coding, pattern recognition, statistics, and physics. Still image compression technique based on 2-D discrete wavelet transform (DWT) has already gained superiority over traditional JPEG based on discrete cosine transform and is standardized in forms like JPEG 2000. Quite similarly, the application of its 3-D superset, i.e., 3-D DWT on video, outperforms the current predictive coding standards, like H.261-3, MPEGl-2,4 Successful application of 3-D DWT has been reported in the literature in emerging fields like medical image compression, hyper-spectral and space image compression, etc.

After the advent of the lifting scheme in 1994, the computation of DWT has experienced a sea change. While providing facilities like reduced computational complexity, in-place computation, ease in building non-linear and inverse wavelets, the lifting also reduces the memory requirement. Thus, it has become a powerful tool to the researchers for computation of both 2-D and 3-D DWT in several applications. The Lifting Scheme is an efficient implementation of Wavelet Transform Using the Lifting Scheme, it is easy to use integer arithmetic without encountering problems due to finite precision or rounding. Applying the inverse transform in the Lifting Scheme is very easy and, as long as the transform coefficients are not quantized, will always result in a perfect reconstruction of the original picture, regardless of the precision of the applied arithmetic.

Recently, several novel architectures based on the lifting scheme have been proposed. Shi et al. [2] achieved an efficient folded architecture (EFA) with low hardware complexity. However, its critical path delay is Tm + Ta,

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering

NCDSPICE 2016

National Conference on Digital Signal Processing, Information and Communication Engineering

Govt. Polytechnic College, Kasaragod

Vol. 5, Special Issue 4, November 2016

where Tm and Ta are the delay of a multiplier and an adder, respectively, and the computation time of EFA is quite long. Through optimizing the lifting scheme, Wu and Lin [3] proposed a pipelined architecture to reduce the critical path to one multiplier and limit the size of the temporal buffer to 4N for an image of size NxN, but it has one input and one output and cannot achieve high processing speed.

Based on Wu and Lin's design, Lai et al. [4] implemented the parallel 2-D DWT. The design is a pipelined two-input/two-output architecture, and a 2x2 transposing module with four registers was developed. In addition, the critical path delay is one Tm. Nevertheless, it needs eight pipelining stages to complete the 1-D DWT and makes the total number of registers reach 22.

The flipping structure is another important DWT architecture that was proposed by Huang et al. [5]. With a five-stage pipeline, the critical path can be also reduced to one multiplier. However, the flipping structure has a large temporal buffer, and fewer pipelining stages lead to longer critical path delay.

In this brief, further optimization on the lifting scheme is proposed by Zhang et al. [1] to overcome shortages in previous works and minimize sizes of the logic units and the memory without loss of the throughput. By recombining the intermediate results of the row and column transforms, the number of pipelining stages and registers can be reduced, while keeping the critical path delay as Tm.

In this paper, Modifications are made to lifting scheme in order to reduce critical path delay and size of temporal buffer. The paper is organized as follows: The equation implementation of (9,7) filter using lifting scheme is described in section II. Section III discusses the overall architecture of a 2-D DWT. Simulation results are presented in Section IV and the paper is concluded in Section V.

LIFTING BASED DWT

The lifting scheme is a method for constructing wavelets by spatial approach. Any DWT filter bank of perfect reconstruction can be decomposed into a finite sequence of lifting steps. This decomposition can factorize the ploy-phase matrix of the target wavelet filter into a sequence of alternating upper and lower triangular matrices and a constant diagonal matrix, which can be expressed as follows

$$h(z) = h_e(z) + z^{-1} h_o(z) \quad (1)$$

$$g(z) = g_e(z) + z^{-1} g_o(z) \quad (2)$$

$$P(z) = \begin{bmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{bmatrix} \quad (3)$$

$$P(z) = \prod_{i=1}^{m} \begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & \frac{1}{K} \end{bmatrix} \quad (4)$$

where $h(z)$ and $g(z)$ are the low-pass and high-pass analysis filters, respectively, equations (3) and (4) shows the poly-phase decomposition, and $P(z)$ is the poly-phase matrix.

The (9,7) filter can be decomposed into four lifting stages as follows:

$$P(z) = \begin{bmatrix} 1 & \alpha(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \beta(1+z) & 1 \end{bmatrix} \begin{bmatrix} 1 & \gamma(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \delta(1+z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & \frac{1}{K} \end{bmatrix} \quad (5)$$

where α = -1.586134342, β = -0.052980118, γ = 0.882911076, δ = 0.43506852, and K = 1.149604398. The whole lifting scheme of the (9,7) filter is shown in Fig.1.

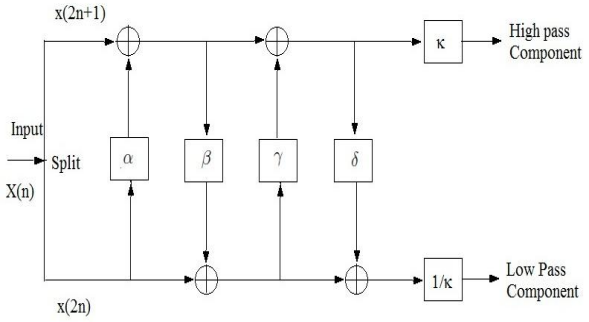


Fig 1.Lifting scheme of (9,7) filter

$$\frac{1}{\alpha} y(2n+1) = \frac{1}{\alpha} x(2n+1) + x(2n) + x(2n+2) \quad (6)$$

$$\frac{1}{\beta} y(2n) = \frac{1}{\beta} x(2n) + y(2n-1) + y(2n+1) \quad (7)$$

$$\frac{1}{\gamma} H(2n+1) = \frac{1}{\gamma} y(2n+1) + y(2n) + y(2n+2) \quad (8)$$

$$\frac{1}{\delta} L(2n) = \frac{1}{\delta} y(2n) + H(2n-1) + H(2n+1) \quad (9)$$

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering

NCDSPICE 2016

National Conference on Digital Signal Processing, Information and Communication Engineering

Govt. Polytechnic College, Kasaragod

Vol. 5, Special Issue 4, November 2016

Where α , β , γ , δ are the lifting coefficients. $x(2n)$ is the input signal. H(n) and L(n) represent high pass and low pass components respectively. Based on (6)-(9), the flipping structure can achieve one multiplier delay by pipelining. However, the above flipping based algorithm also shows obvious limitations. It needs the temporal buffer with the size of 11N to cache the intermediate data. Substituting (6) into (7)

$$\frac{1}{\alpha\beta}y(2n)=\frac{1}{\alpha\beta}x(2n)+\frac{1}{\alpha}y(2n-1)+\frac{1}{\alpha}y(2n+1) \quad (10)$$

Reordering the expression with the associative law, we can get

$$\frac{1}{\alpha\beta}y(2n)=\left\lfloor\left(\frac{1}{\alpha\beta}+1\right)x(2n)+\frac{1}{\alpha\beta}x(2n-1)+x(2n-2)\right\rfloor + \left\lfloor\frac{1}{\alpha\beta}x(2n+1)+x(2n)+x(2n+2)\right\rfloor \quad (11)$$

Four intermediate variables, namely, $D_1^k(n)$, $D_2^k(n)$, $D_3^k(n)$, $D_4^k(n)$ are defined as below, where k stands for different values in the row and column transforms. In the row transform, k means the number of rows in progress, whereas in the column transform, k represents the number of scans, and one scan means finishing parallel scan of two adjacent rows in the column transform.

$$D_1^k(n)=\frac{1}{\alpha}x(2n+1)+x(2n) \quad (12)$$

$$D_2^k(n)=\left(\frac{1}{\alpha\beta}+1\right)x(2n)+\frac{1}{\alpha}x(2n-1)+x(2n-2) \quad (13)$$

$$D_3^k(n)=\frac{1}{\gamma}y(2n+1)+y(2n) \quad (14)$$

$$D_4^k(n)=\left(\frac{1}{\gamma\delta}+1\right)y(2n)+\frac{1}{\gamma}y(2n-1)+y(2n-2) \quad (15)$$

Thus rearranging (9) with the method deriving to (11), then substituting (12)-(15) into (6),(8),(11) and the expression rearranged from (9) the following expressions can be derived:

$$\frac{1}{\alpha}y(2n+1)=D_1^k(n)+x(2n+2) \quad (16)$$

$$\frac{1}{\alpha\beta}y(2n)=D_2^k(n)+D_1^k(n)+x(2n+2) \quad (17)$$

$$\frac{1}{\gamma}H(2n+1)=D_3^k(n)+y(2n+2) \quad (18)$$

$$\frac{1}{\delta\gamma}L(2n)=D_4^k(n)+D_3^k(n)+y(2n+2) \quad (19)$$

Compared with the flipping-based lifting scheme, this modified algorithm suggests a novel way in data combination with different coefficients in even data and simplifies the computation process. The algorithm combines the predictor with the updater. The high-pass signal and the low-pass signal can be calculated in parallel through the two-input/two-output architecture. At the same time, the coefficients of even items are changed by inversion of the factors. As a result, (11) can be divided into two asymmetrical parts, and data with coefficients $(1+\frac{1}{\alpha\beta})$ and $\frac{1}{\alpha}$ can be obtained from the first pipelining stage by dividing multiplication and addition into two pipelining stages. Intermediate variables $D_1^k(n)$ and $D_2^k(n)$, as shown in (12) and (13), are introduced into (11) to calculate $\frac{y(2n+1)}{\alpha}$ and $\frac{y(2n)}{\alpha\beta}$ together in the second pipelining stage with a critical path delay of T_m. Simultaneously, $D_1^k(n)$ and $D_2^k(n)$ are updated. The highpass signal and the low-pass signal can be directly outputted from the third pipelining stage. Therefore, the pipelining stages of the 1-D processing element (PE) are limited to three, and the number of registers is further reduced.

ARCHITECTURE FOR THE 2-D DWT

A. Overall Architecture

A novel architecture for the 2-D DWT based on the modified algorithm is shown in Fig.2.

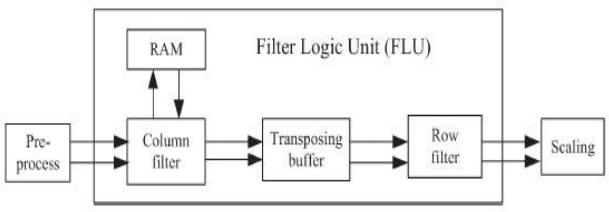


Fig.2.Overall architecture of a 2-D DWT

First, the serial-parallel conversion for the original data in the preprocessing module is carried out. After that, data are sent into the column filter for the column transform. Next, the output data of the column filter are sent into the transposing buffer, where the data transposition is operated to meet the order of the data flow required by the row filter. Then, the row filter begins to read the data from

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE
International Journal of Advanced Research in Computer and Communication Engineering

NCDSPICE 2016
National Conference on Digital Signal Processing, Information and Communication Engineering
Govt. Polytechnic College, Kasaragod
Vol. 5, Special Issue 4, November 2016

the transposing buffer for the row transform. Finally, the scaling module is used to finish the scaling computation.

B. Parallel Scan of Preprocessing and Raw Image Data

Since parallel scanning is adopted, as shown in Fig.3., the data of each even row and odd row of the columns are alternately read. This way, the column filter can process the column transform for the data of adjacent columns alternately. With the preprocessing module, raw image data are factorized into odd and even parts for the following filter logic unit (FLU) module.

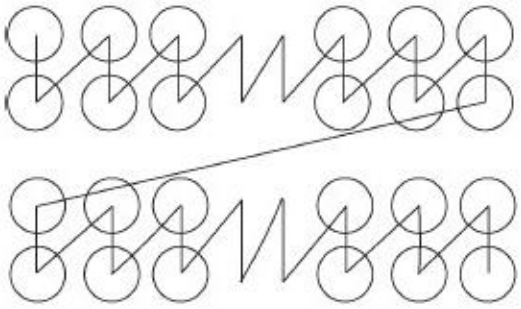


Fig.3. Parallel scanning of the input data

C. One-Dimensional PE

The architecture of the 1-D PE is shown in Fig.4. This architecture can be applied in the column and row filters by selecting the RAM or Buffer properly. In order to reduce the size of the transposing buffer between the column and row filters and to improve the processing speed, this design adopts the architecture of two-input/two-output.

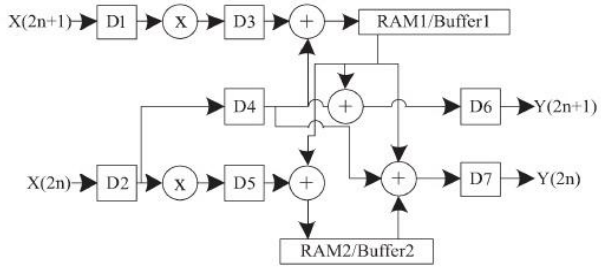


Fig.4. Processing element

When the column filter begins to work, the input data from the preprocessing module, the odd term $x_i(2n+1)$ and the even term $x_i(2n)$, are sent into the column filter in each clock cycle simultaneously. In this case, i means the column index, and n has the same meaning as k, which represents the number of scans in the column.

Then, each input is multiplied by the corresponding coefficient. The result of multiplication is used in the computation of the intermediate variables $D^k_{1(i)}(n)$ and $D^k_{2(i)}(n)$,which will be stored in the temporal buffers RAM1 and RAM2, respectively. At the same time, if k >1 , $D^{k-1}_{1(i)}(n)$ and $D^{k-1}_{2(i)}(n)$ will be read for the computation of $y_i(2n-1)$ and $y_i(2n-2)$.

Otherwise, the boundary extension will be processed. Because of parallel scanning, $D^{k-1}_{1(i)}(n)$ and $D^{k-1}_{2(i)}(n)$ read from RAM1 and RAM2 in current clock cycle are calculated and saved at the corresponding places in the previous two rows.

Then, they are used by the new data to carry out the current column transform. RAM1 and RAM2 are both dual-port RAM devices with the depth of 2N. Fig.5. shows the data flow of the first lifting step in the column filter. Such architecture not only has low hardware complexity but also reduces the critical path between every two registers to one multiplier.

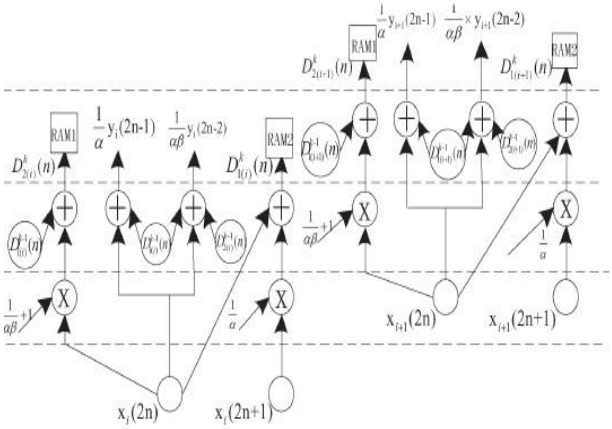


Fig.5. Data flow of the firstlifting stepin the column filter

For the row filter, two buffers with the size of two registers are enough to store the intermediate variables. In the row filter, the input data will take part in the computation of intermediate variables $D^k_1(n+1)$ and $D^k_2(n+1)$ after two clock cycles. Then, $D^k_1(n+1)$ and $D^k_2(n+1)$ are fed into Buffer1 and Buffer2, which are used to meet the requirement of the alternate reading intermediate variables in the odd and even rows and to make the computation in the odd and even rows independent without disturbing each other.

$D^k_1(n)$ and $D^k_2(n)$, which are stored in another addresses in Buffer1 and Buffer2 are read out to calculate $y(2n+1)$ and $y(2n)$ at the same time. It should be noted that n is the variable to judge the boundary extension in the row filter instead of the variable k in the column filter.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

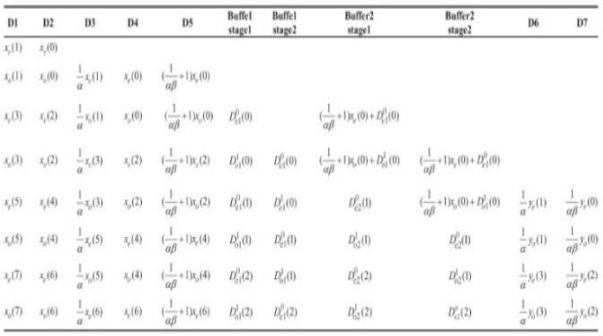
International Journal of Advanced Research in Computer and Communication Engineering

NCDSPICE 2016

National Conference on Digital Signal Processing, Information and Communication Engineering

Govt. Polytechnic College, Kasaragod

Vol. 5, Special Issue 4, November 2016



D1	D2	D3	D4	D5	Buffe1 stage1	Buffe1 stage2	Buffer2 stage1	Buffer2 stage2	D6	D7
$x_e(1)$	$x_o(0)$									
$x_e(1)$	$x_o(0)$	$\frac{1}{\alpha}x_e(1)$	$x_o(0)$	$(\frac{1}{\alpha\beta}+1)x_o(0)$						
$x_e(3)$	$x_o(2)$	$\frac{1}{\alpha}x_e(1)$	$x_o(0)$	$(\frac{1}{\alpha\beta}+1)x_o(0)$	$D^0_{o1}(0)$		$(\frac{1}{\alpha\beta}+1)x_o(0)+D^0_{e1}(0)$			
$x_e(3)$	$x_o(2)$	$\frac{1}{\alpha}x_e(3)$	$x_o(2)$	$(\frac{1}{\alpha\beta}+1)x_o(2)$	$D^1_{o1}(0)$	$D^0_{e1}(0)$	$(\frac{1}{\alpha\beta}+1)x_o(0)+D^1_{o1}(0)$	$(\frac{1}{\alpha\beta}+1)x_o(0)+D^0_{e1}(0)$		
$x_e(5)$	$x_o(4)$	$\frac{1}{\alpha}x_e(3)$	$x_o(2)$	$(\frac{1}{\alpha\beta}+1)x_o(2)$	$D^0_{o1}(1)$	$D^1_{e1}(0)$	$D^0_{e2}(1)$	$(\frac{1}{\alpha\beta}+1)x_o(0)+D^1_{o1}(0)$	$\frac{1}{\alpha}y_e(1)$	$\frac{1}{\alpha\beta}y_o(0)$
$x_e(5)$	$x_o(4)$	$\frac{1}{\alpha}x_e(5)$	$x_o(4)$	$(\frac{1}{\alpha\beta}+1)x_o(4)$	$D^1_{o1}(1)$	$D^0_{e1}(1)$	$D^1_{e2}(1)$	$D^0_{e2}(1)$	$\frac{1}{\alpha}y_e(1)$	$\frac{1}{\alpha\beta}y_o(0)$
$x_e(7)$	$x_o(6)$	$\frac{1}{\alpha}x_e(5)$	$x_o(4)$	$(\frac{1}{\alpha\beta}+1)x_o(4)$	$D^0_{o1}(2)$	$D^1_{e1}(1)$	$D^0_{e2}(2)$	$D^1_{e2}(1)$	$\frac{1}{\alpha}y_e(3)$	$\frac{1}{\alpha\beta}y_o(2)$
$x_e(7)$	$x_o(6)$	$\frac{1}{\alpha}x_e(6)$	$x_o(6)$	$(\frac{1}{\alpha\beta}+1)x_o(6)$	$D^1_{o1}(2)$	$D^0_{e1}(2)$	$D^1_{e2}(2)$	$D^0_{e2}(2)$	$\frac{1}{\alpha}y_e(3)$	$\frac{1}{\alpha\beta}y_o(2)$

Fig.6. Data flow of the first lifting step in the row filter

Fig.6. shows the data flow of the first lifting step in the row filter, where subscripts e and o represent the even and odd rows, respectively. In the row filter, k represents the number of row in which the current calculation is going on.

Set k = 0 for the first even row and k = 1 for the first odd row. The following cases can be obtained in the same manner as above. Note that the aforementioned row refers to the flow order of the input data for the transposing buffer.

D. Transposing Module

The architecture of the transposing module is shown in Fig.7. Three registers and two multiplexers are used to make the output data meet the order of the data flow required by the row filter. Fig.7. also shows the orders of the input and the output for the transposing module, where L and H represent the low-pass signal and the high-pass signal of the column transform output, respectively.

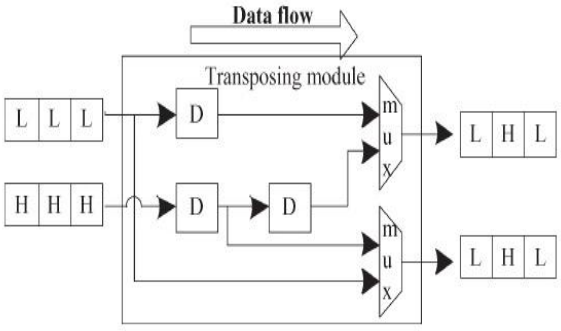


Fig.7. Transposing module

SIMULATION RESULTS

The input image of size 64x64, used for compression is shown in Fig.8. The image is converted into pixel values. These pixel values are stored in a file.

Then read pixel values from file to test bench memory. The computed DWT result is stored into a memory. Then it was written to a file.

Fig.8.Lena image

Fig.9.DWT result

The simulation result of program for DWT using lifting based (9,7) filter is shown in Fig.10.

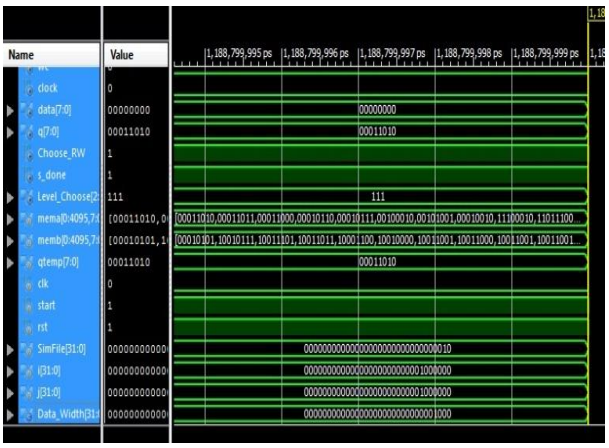


Fig.10.Simulation Result

V. CONCLUSION

The Discrete Wavelet Transform is a multi-resolution analysis tool with excellent characteristics in the time and frequency domains. The DWT is widely used in signal processing and image compression, such as JPEG 2000. Compared with convolution based DWT lifting based architectures not only have lower computation complexity but also require less memory. In this work the architecture is developed for (9,7) filter based on modified lifting scheme. The modified one lifting step circuit can work within three pipelining stages with fewer registers, and the critical path delay is Tm.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering

NCDSPICE 2016

National Conference on Digital Signal Processing, Information and Communication Engineering

Govt. Polytechnic College, Kasaragod

Vol. 5, Special Issue 4, November 2016

REFERENCES

[1] 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, pp. 158-162, March 2012.

[2] G. Shi, W. Liu, and L. Zhang, "An efficient folded architecture for lifting based discrete wavelet transform,"IEEE Trans. Circuits Syst. II, Exp.Briefs, vol.56, no. 4, pp. 290-294, Apr. 2009.

[3] B.F.Wu and C.F.Lin, "A high-performance and memory-efficient pipeline architecture for the 5/3 and 9/7 discrete wavelet transform of JPEG2000 codec,"IEEE Trans. Circuits Syst. Video Technol., vol. 15, no. 12, pp. 1615-1628, Dec. 2005.

[4] Y. K. Lai, L. F. Chen, and Y. C. Shih, "A high-performance and memory-efficient VLSI architecture with parallel scanning method for 2-D lifting based discrete wavelet transform," IEEE Trans. Consum. Electron., vol. 55, no. 2, pp. 400-407, May 2009.

[5] C.T.Huang,P.C.Tseng, and L.G.Chen, "Flippingstructure: An efficient VLSI architecture for lifting-based discrete wavelet transform,"IEEE Trans. Signal Process., vol. 52, no. 4, pp. 1080-1089, Apr. 2004.

[6] I.Daubechies and W.Sweldens, "Factoring wavelet transform into lifting steps," J. Fourier Anal. Appl., vol. 4, no. 3, pp. 245-267, Mar. 1998.

[7] C.Y. Xiong, J.W. Tian, and J. Liu, "A note on flipping structure: An efficient VLSI architecture for lifting-based discrete wavelet transform," IEEE Trans. Signal Process., vol. 54, no. 5, pp. 1910-1916, May 2006.

[8] Xiong, J. Tian, and J. Liu, "Efficient architectures for two-dimensional discrete wavelet transform using lifting scheme,"IEEE Trans. Image Process., vol. 16, no. 3, pp. 607-614, Mar. 2007.

[9] G.Xing, J. Li, andY. Q.Zhang, "Arbitrarily shaped video-objectcoding by wavelet,"IEEE Trans. Circuits Syst. Video Technol., vol. 11, no. 10, pp. 1135-1139, Oct. 2001.

[10] K. K. Parhi and T. Nishitani, "VLSI architecture for discrete wavelet transforms,"IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 1, no. 2, pp. 191-202, Jun. 1993.

[11] H. Liao, M. K. Mandal, and B. F. Cockburn, "Efficient architectures for 1-D and 2-D lifting-based wavelet transforms,"IEEETrans.SignalProcess.,vol.52,no.5,pp.1315-1326, May 2004.

[12] Cheng and K. K. Parhi, "High-speed VLSI implement of 2-D discrete wavelet transform," IEEE Trans. Signal Process., vol. 56, no. 1, pp. 393- 403, Jan. 2008.