High Performance VLSI Architecture for 2-D DWT Using Lifting Scheme

Mithun R,Ganapathi Hegde
Dept. of ECE, Amrita Vishwa Vidyapeetham
Amrita School of Engineering, Bangalore 560 035, INDIA
mithunchembra@gmail.com
ganapathihegde1@gmail.com
Phone: +91 80 25183700, Fax: +91 80 28440092

***Abstract*—Reduced area and high speed 2-D DWT structural design is presented here. To decrease the delay in critical path with one multiplier, minimum stages of pipeline stage required are four for a lifting step. To reduce the pipeline stages, here short modification is adopted by recombining and storing the intermediate stages of result value. By adopting this work, we can reduce register number without critical path extension and scanning architecture adopted is parallel two inputs/ two outputs architecture, and it increases the speed and critical path can be decreased to a delay of a multiplier by 3 pipeline stages. By adopting shift add method for multiplication, the proposed method able to decrease the delay in critical path to delay of a adder. The number of registers between column and row filter requires is three for this proposed architecture for NXN size 2-DWT.**

***Index Terms*—Discrete wavelet transform (DWT), lifting scheme, pipeline, VLSI architecture.**

I. Introduction

The compensation of the wavelet transform to conventional transforms, like fourier transform, are recognized fine. Since it is having good locality in time-frequency domain, wavelet transform is broadly used for analysis and compression of the signal. Mallat introduced prospect of its implementation. The discrete wavelet transform (DWT) perform a multi-resolution signal analysis, which has adjustable locality in both the space (time) and frequency domains. The decomposition of signals in to various sub bands with frequency and time information can be possibly by using DWT. Comparing to DCT, image restoration quality and coding efficiency is high for DWT. More over DWT has high compression ratio. So DWT is widely using for image compression and signal processing such as JPEG2000, MPEG-4 and so on. By using FIR filters and then sub sampling is the usual implementation method of DWT. As a result of its huge number of computations, many research works to build up innovative algorithms[1].A DWT using lifting scheme can be simply implemented due to significantly fewer Computations[2].This process is fully based on a spatial justification of the wavelet transform. Moreover, it is having the ability of producing new mother wavelets. DWT implementation on field programmable gate array (FPGA) and DSP chips has been widely developed. The structural processing elements are set successively in the lifting scheme. These processing elements include multipliers also. Thus at this point the amount of multipliers in every pipeline phase decides the clock rate of the structure. According to[3],the key challenges in the hardware building for 1-D DWT are the dispensation rate and the amount of multipliers.But in the case of 2-D DWT the issue is mainly the memory and it dominates the hardware cost and complexity in hardware. The limitation of the on-chip memory and the power utilization[3][4] is the reason. An efficient folded (EFA) structural design is discussing about low complexity in hardware. But there is a Tm +Ta of critical path. Where Ta and Tm are adder and multiplier delay and quite long computation time is required for EFA. Lin and Wu [5] presented a pipeline structure that achieves the delay of critical path to Tm and temporal buffer size is limited to 4N. But processing speed cannot achieve in to high because of one input one output method of scanning. Parallel 2-D DWT is implemented by Lai et al.[6] . Although it is one of the pipelined design with two inputs/ two outputs with Tm critical path, it requires eight number of pipeline stages for the completion of 1-DWT.

Huang et al [7] proposed a flipping DWT architecture. Here pipeline stages used is five with critical path of one multiplier delay. However, few pipelining stages lead to long delay in critical path and it has a temporal buffer of large size. The number of stages in pipeline and number of register can be reduced by recombining the intermediate values of the result. The proposed lifting scheme can overcome the shortages of previous work and both memory size and logic size can be minimized without throughput loss. Here delay in critical path is Tm and this can be reduced to Ta by using shift add method. More over for the buffer size reduction, the parallel scanning method is adopted. Consequently, from our design higher effectiveness can be achieved.

The remains of this paper are planned as follows. Section II reviews flipping and lifting architecture of DWT. Then our adapted DWT algorithm is presented . Section III presents 2-D DWT architecture for this work and Section IV presents implementation and comparison of this architecture with previous architectures. Section V concluded this paper.

II. PROPOSED ALGORITHM

In 1996, Sweldens presented a lifting scheme DWT with high computational speed, which can be implemented easily by hardware due to broadly reduced computations[2],[4] and [8]. It tells that each FIR filters bank or wavelet that can be factored in to lifting steps. That is the decomposition of the poly phase matrices in to a upper and lower triangular matrices multiplied with a diagonal normalization matrix. 9/7 filter lifting scheme has one scaling step and two lifting steps. By changing the lifting coefficient in [7] a adapted algorithm is employed.

$$y(2n+1) = x(2n+1) + a[x(2n) + x(2n+2)] \quad (1)$$

$$y(2n) = x(2n) + b[x(2n-1) + y(2n+1)] \quad (2)$$

$$H(2n+1) = y(2n+1) + c[y(2n) + y(2n+2)] \quad (3)$$

$$L(2n) = y(2n) + d[H(2n-1) + H(2n+1)] \quad (4)$$

Here for 5/3 filter value of a = 1/2 and b =1/4, and that of 9/7 filter value of a= -1.58613, b= -0.052980,c =0.882911075 and d=1.230174104

We can rearrange (1) to (4) as

$$1/a * y(2n+1) = 1/a * x(2n+1) + x(2n) + x(2n+2) \quad (5)$$

$$1/b * y(2n) = 1/b * x(2n) + x(2n-1) + y(2n+1) \quad (6)$$

$$1/c * H(2n+1) = 1/c * y(2n+1) + y(2n) + y(2n+2) \quad (7)$$

$$1/d * L(2n) = 1/d * y(2n) + H(2n-1) + H(2n+1) \quad (8)$$

One multiplier delay in flipping structure can be achieved as a result of pipelining. But the algorithm given has restrictions. It wants high temporal buffer size for doing the cache of the in-between data.

From the above substitute (5) to (6) and reordering the expression we will get,

$1/ab * y(2n) = 1/ab * x(2n) + 1/c * y(2n+1)$

and

$$1/ab * y(2n) = [(1/ab+1) * x(2n) + 1/a * x(2n-1) + x(2n-2)] + [1/a * x(2n+1) + x(2n) + x(2n+2)] \quad (9)$$

$R_1^k(n)$,$R_2^k(n)$,$R_3^k(n)$,$R_4^k(n)$ are intermediate variables as given below and different values of k in the row and column transform. K representing the number of the rows in progress in the row transforms and in column transform it is the number of scans, where one scan is parallel scan of two neighboring rows finish in the column transform.Thus

$$R_1^k = 1/a * x(2n+1) + x(2n) \quad (10)$$

$$R_2^k = (1/ab+1) * x(2n) + (1/a) * x(2n-1) + x(2n-1) \quad (11)$$

$$R_3^k = 1/c * y(2n+1) + y(2n) \quad (12)$$

$$R_4^k = (1/cd+1) * y(2n) + (1/c) y(2n-1) + y(2n-2) \quad (13)$$

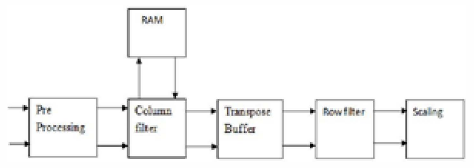

Fig. 1. Two Dimensional DWT Architecture

Rearrange equation (8), and then Substitute (10) to (13) in to the equation (5), (7), (9) and then rearrange form (8) then we will get,

$$(1/a) y(2n+1) = R_1^k + x(2n+2) \quad (14)$$

$$(1/ab) y(2n) = R_2^k + R_1^k + x(2n+2) \quad (15)$$

$$1/c * H(2n+1) = R_3^k + y(2n) \quad (16)$$

$$(1/cd) * L(2n) = R_4^k + R_3^k + y(2n) \quad (17)$$

Comparing to the flipping based structure, this modified algorithm tells data grouping way by different values of coefficients in even sample and hence computation process simplified. Predictor is combined with update in this proposed algorithm. With parallel two-input/ two-output structural design, the high-pass values and low-pass values can be possible to calculate together. The inversion factor changes coefficients of the even items. So equation (9) is possible to divide in to asymmetrical parts of two, and data is possible to obtain with (1+1/ab) and (1/a) coefficients from the primary pipeline stage. Here multiplication and addition is possible to divide in to two pipeline phase. The intermediate variables as shown in equation (10) and equation (11) can be substituted in to equation (9) for the calculation of y (2n+1)/a and y (2n)/ab jointly in the second pipeline phase with a Tm delay in critical path. At the same time the intermediate values are updated.

That is $R_1^k(n)$ and $R_2^k(n)$.The high pass value and low pass value can be getting as an output in the third phase of pipeline. There for we can limit the pipeline stages required for a one dimensional processing element in to in to three with reduced number of registers.

III. PROPOSED 2-D DWT ARCHITECTURE

A. *Overall architecture of 2-D DWT*

From adapted algorithm, Fig.1 tells about 2-D DWT architecture. Here preprocessing stage carry out serial-parallel translation of the original sample sequence and then data are given to column processor for doing the operation of column transform.

Then data output of column filter are given to transposing buffer, where transposition of data occurs to meet the dataflow order for the operation of row filter. Finally, scaling computation is done by scaling module. Fig.2 which helps to understand the series of operation concerned in this process.

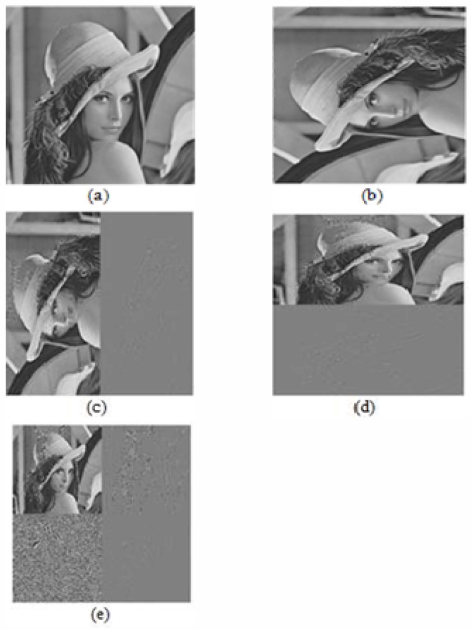

Fig. 2. (a) Original image. (b) Pre -processed image. (c) After column process. (d)After transposing of column processed output. (e) After row processing.

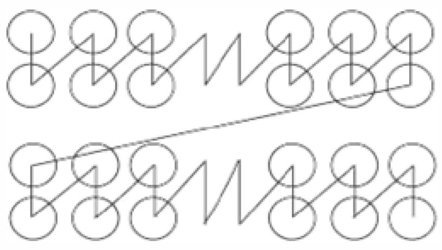

Fig. 3. Parallel scan method

B. Scanning method

Every even and odd row of sample is reading alternatively because of parallel scanning method as shown in Fig. 3. This way, column transform can be processed by column filter for the sample of neighboring column alternatively.

C. One-Dimensional processing element (1D PE)

The proposed 1-D processing element architecture is shown in Fig. 4. It is possible to apply this architecture in the row and column filter by the proper selection of RAM or buffer properly. By adopting the two input/two output structural design, it is possible to decrease the transpose buffer size among column processor and row processor and also improvement in speed of operation. When column filter starts its duty, the input sample getting from preprocessing module,the odd sample $x_i(2n+1)$ and the even sample $x_i(2n)$ are sending to column filter at the same time in every cycle. Here i is representing column index and n is representing the number of scans in the column, and n means same as k.

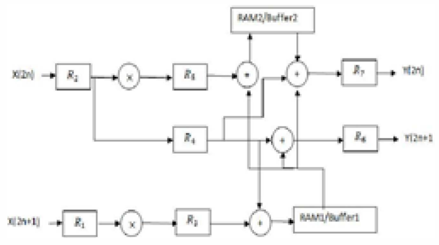

Fig. 4. Proposed 1-D Processing Element

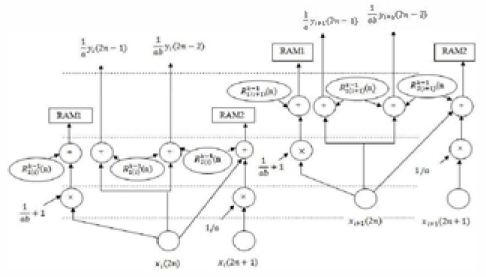

Fig. 5. Data flow of first lifting step in the column filter

Then every input is multiplied with equivalent filter coefficient. The multiplication result used for the computation $R_1^k(n)$ and $R_2^k(n)$ of intermediate variables , which will store in RAM1 and RAM2 respectively where, RAM1 and RAM2 are temporal buffer. If k¿1, at the same time, computation of Yi(2n-1) and Yi(2n-2) are processing by reading $R_1^{k-1}(n)$ and $R_2^{k-1}(n)$.Otherwise, the processing of boundary extension will happens. The dual port RAM devices RAM1 and RAM2 are having the depth of 2N. First lifting step in the column filter is shown in Fig 5. Such architecture has low complexity in hardware and also decreases the critical path to one multiplier.

Two buffers are sufficient for storing the intermediate variables and buffer size is that of a register. $R_1^k(n+1)$ and $R_2^k(n+1)$ for the row filter operation. In the row filter, here for the computation of intermediate variables input will take part after two clock cycles. $R_1^{k-1}(n)$ and $R_2^{k-1}(n)$, which are intermediate variables stored in buffer1 and buffer2 with another addresses are reading out for the y(2n+1) calculation and y(2n) calculation at same time. Here boundary extension is judged by variable n instead of k in column filter.

D. Transposing Module

Fig. 5 shows transposing module architecture .The sequence of the sample flow needed for row filter can be met by using three registers and two multipliers. In the fig. 6 tells about the transposing module and here, L and H is representing low pass value and high pass value respectively.

E. Shift and add unit for multiplier

Two multiplications can be implemented by using shifter to save the hardware. It needs adders, unit for complement and

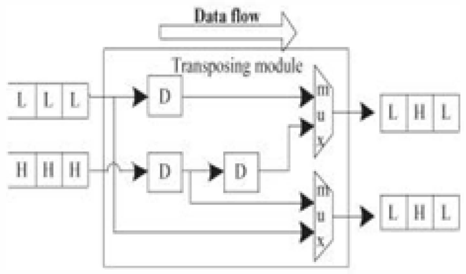

Fig. 6. Transposing module

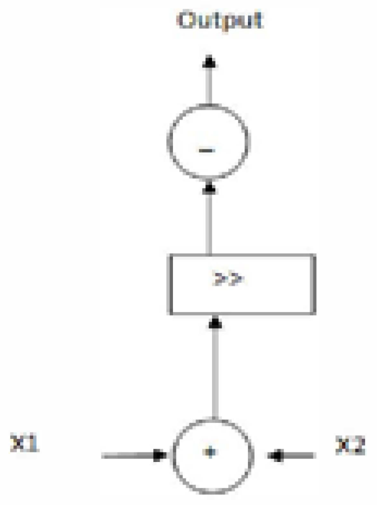

Fig. 7. Shift and add unit for multiplier

shifter. The block diagram for shift adder multiplier is given in Fig. 7. Here X1 and X2 are inputs, -denotes 2s complement and $>>$ denotes right shift

IV. PERFORMANCE COMPARISON AND FPGA IMPLEMENTATION

In this section performance analysis of previous designs compared with proposed one and we implemented this work as behavioral level Verilog model and output simulated. We compared the FPGA implemented result of proposed one with Weizhang architecture.

A. Performance comparison

Here we compared the performance of proposed architecture with previously existing architecture and it is given in Table I. Here performance comparison is evaluated in terms of delay in critical path, throughput rate and hardware complexity (measured by number of multiplier, adder and register).
From Table I we can find that, the less number of arithmetic resources is used in the proposed architecture and here number of multiplier used is zero with throughput rate of two inputs/ two outputs per cycle. The delay in critical path is Ta, which is less than the previous architectures given in table.

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT 2-D DWT ARCHITECTURES

Architecture	**Multiplier**	**Adder**	**Citical Path Delay**	**Throughput**
Generic RAM[10]	10	16	$4T_m + 8T_a$	1 input/output
Flipping[7]	10	16	$4T_m + 8T_a$	1 input/output
Flipping[7]+5pipeline	10	16	$4T_m + 8T_a$	1 input/output
DSA[11]	12	16	$4T_m + 8T_a$	1 input/output
Wu[5]	6	8	$4T_m + 8T_a$	1 input/output
PLSA+Pipiline	10	16	$4T_m + 8T_a$	1 input/output
FA[12]	10	16	$4T_m + 8T_a$	2 input/output
HA[12]	18	32	$4T_m + 8T_a$	2 input/output
Cheng(s=1)[13]	10	40	$4T_m + 8T_a$	2 input/output
Mohanthy(s=17)[14]	388	688	$4T_m + 8T_a$	2 input/output
Wi Zhang[9]	10	16	$4T_m + 8T_a$	2 input/output
Proposed	0	20	$4T_m + 8T_a$	2 input/output

TABLE II
DEVICE UTILIZATION OF WEIZHANG ARCHITECTURE IN VERTEX-4

Logic Utilization	**Used**	**Available**
Numbers of Slices	1658	6144
Numbers of Slices Flip flops	553	12288
Number of 4 input LUTs	2942	12288
Number of bounded IOBs	50	240
Number of RAMB16s	11	48
Number of GCLKs	1	32

B. FPGA Implementation

We implemented this work as behavioral level Verilog model and output simulated. The proposed work is synthesized and implemented for Xilinxs Vertex-4 and compared the result with Weizhang [9] architecture. The synthesized result is shown in table II and table III.

V. CONCLUSION

In this work we introduced a new structural design for DWTs by using shift method for multiplication. Here we need three pipelining phase for one lifting step with fewer number of registers and the delay in critical path is Ta. This structure used only the 4N temporal buffer space for two dimensional wavelet transform. A thorough study is carried out to evaluate the presented structural design with existing architecture. Thus the complexity in hardware, delay in critical path, and throughput of various architectures are compared. From the results, this work achieves better speed with lesser complexity in hardware and lesser storage space.

TABLE III
DEVICE UTILIZATION OF PROPOSED ARCHITECTURE IN VERTEX-4

Logic Utilization	**Used**	**Available**
Numbers of Slices	1544	6144
Numbers of Slices Flip flops	525	12288
Number of 4 input LUTs	2939	12288
Number of bounded IOBs	50	240
Number of RAMB16s	10	48
Number of GCLKs	1	32

References

[1] S. C. B. Lo, H. Li, and M. Freedman, "Optimization of wavelet decomposition for image compression and feature preservation," *IEEE Trans. Med. Image*, vol. 22, no. 9, pp. 1141–1151, Sept.2003.

[2] K. K. Parhi and T. Nishitani, "Vlsi architecture for discrete wavelet transforms," *IEEE Trans. Very Large Scale Integer. (VLSI)syst.*, vol. 2, no. 2, pp. 191–202, Jun 1993.

[3] Wu and L. Chen, "An efficient architecture for two-dimensional discrete wavelet transform," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 4, pp. 536–545, Apr 2001.

[4] W. Sweldens, "The new philosophy in biorthogonal wavelet constructions," *Proc. SPIE*, vol. 2569, pp. 68–79, 1995.

[5] 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.

[6] Y. K. Lai, L. F. Chen, and Y. C. Shih

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

[8] K. Andra, C. Chakrabarti, and T. Acharya, "A vlsi architecture for lifting-based forward and inverse wavelet transform," *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 966–977, Apr 2002.

[9] W. Zhang and Z. Jiang, "An efficient vlsi architecture for lifting based dwt," *IEEE Trans. Circuit and System- II*, vol. 59, no. 13, 2012.

[10] P-C.Tseng, C-T.Huang, and L.-G. Chen, "Generic ram-based architecture for two dimensional discreet wavelet transform with line based method," *Proc. Asia-Pacific conf. Circuits Syst*, vol. 2, pp. 363–366, 2002.

[11] H.Liao, M. K. Mandal, and B. F. Cockbum, "Efficient architecture for 1-d and 2-d lifting based wavelet transform," *IEEE Trans. Signal process*, vol. 52, no. 5, pp. 1315–1326, May 2004.

[12] C.Xiong, J. Tian, and J.Liu, "Efficient architecture for two dimensional discrete wavelet transform using lifting scheme," *IEEE Trans. Image Process*, vol. 16, no. 13, pp. 607–614, Mar 2007.

[13] C. 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.

[14] B. K. Mohanty and P. K. Meher, "Throughput-scalable hybrid pipeline architecture for multilevel lifting 2-d dwt of jpeg2000 coder," *IEEE int. Conf. Appl. Specific Syst*, pp. 305–309, 2008.