ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

Vlsi Implementation of Haar DWT with Modified Matrix Multiplication Algorithm

[1]R. Shanmuganathan and [2]Dr. S. Dhanalakshmi

[1]PG Scholar, Easwari Engineering College, ECE Department, Chennai, Tamilnadu, India.
[2]Asst.Professor, Easwari Engineering College, ECE Department, Chennai, Tamilnadu, India.

ARTICLE INFO

Article history:
Received 10 March 2015
Received in revised form 20 March 2015
Accepted 25 March 2015
Available online 10 April 2015

Keywords:
Discrete Wavelet Transform(DWT), Matrix Multiplication (MM), Processing Element(PE)

ABSTRACT

Haar (DWT), the best among all DWTs, has numerous applications in signal and image process fields. The prevailing approach for 2-D Haar DWT is by performing 1D row operation then 1D column operation. Anyhow, this method is infeasible for its high computational requirements for processing large sized images. Here the segmented matrix algorithm with improved matrix multiplier design to achieve speed up in computation is implemented. The efficiency of the proposed design is measured and compared with previous algorithms. This improved algorithm is simulated using XILINX software which shows better speed and consumes less number of multipliers than previous algorithms which has been commonly used today.

To Cite This Article: R. Shanmuganathan and Dr. S. Dhanalakshmi., VLSI Implementation of Haar DWT with Modified Matrix Multiplication Algorithm. ***Aust. J. Basic & Appl. Sci.,*** *9(15): 142-147, 2015*

INTRODUCTION

Discrete wavelet transforms (DWTs) has been used in a wide range of signal and image processing applications such as – image and video coding (MPEG-4 or JPEG 2000), image watermarking, pattern recognition, medical image analysis etc. In traditional approach, 2D (two-dimensional) Haar DWT is performed in two phase- one row operation, one column operation, where column operation cannot be performed as far as the row operation is finished. So, the speed of computation degrades significantly. To address this problem, Chen and Liao proposed the segmented matrix algorithm where computation is performed by data rearrangements and one matrix multiplication. Hence, this simple algorithm can produce the same results as traditional 2D Haar DWT with a much faster speed. Furthermore, it is highly suitable for parallel implementation as only two rows are involved in computation per cycle.

Nowadays vast size images are common due to the availability and advancement of image capturing technology. Thus more wavelet based applications have to manage large scaled image processing. A significant amount of works have already been done for all sorts of high performance computers, for special purpose hardware (Bravo, I., 2007; Graves, C. and C. Gloster, 1998; Idris, M., 2010), for FPGAs (Haapala, K., 2000; Martina, M., 2000) for SIMD architectures. In this paper we have implemented the segmented matrix algorithm for 2D Haar wavelet transform. Parallel computing is a direct way of speeding up these high computation requirements. Our objective is to achieve computation speed to process large scaled images without increasing computational complexity and cost. It is achieved by rearranging the matrix element into a two-dimensional array of processing elements interconnected as a mesh. The functionality of the circuitry was verified and the performance parameters for example, propagation delay were calculated.

Traditional Computation:

Haar DWT is the simplest since it only uses two low pass filter coefficients (1,1) and two high pass filter coefficients (1,-1). Haar wavelet transform in frequency domain can be obtained by addition and subtraction of the pixels of images [1]. 2D haar DWT decomposes an input image into four sub bands, one average component (W_{LL}) and three detail components (W_{LH}, W_{HL}, W_{HH}).

Traditionally, 2D Haar wavelet transform can be accomplished by one row and one column operations where the result of row transform is the input of column transform. The 2D Haar wavelet transforms of a 4×4 image is represented in Fig. 1.

Corresponding Author: R. Shanmuganathan, (PG Scholar, Department Of ECE, Easwari Engineering College, Pin code: 600 078 Chennai, Tamilnadu, India.
Ph: +91 9094397822, E-mail: shanmuga1991@yahoo.com.

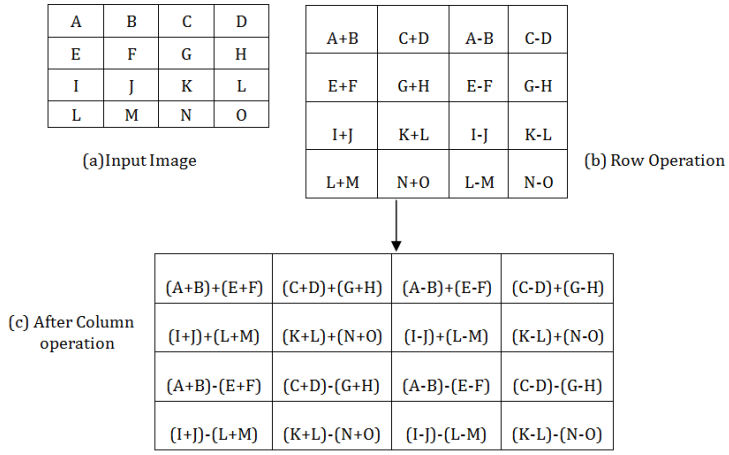


Fig. 1: 2D Haar DWT of a 4×4 image.

Segmented Matrix Algorithm:

Chen and Liao proposed a computationally fast algorithm called "segmented matrix algorithm" where 2D Haar DWT can be performed by only one matrix multiplication instead of two separate 1D transforms. The step by step process of this algorithm is as follows.

Step 1: Consider I as the input image of size m×n. Form B=2×2 sub-blocks from original image I where i=1…m/2 and j=1…n/2. For example,

$$B_{11} = \begin{bmatrix} I_{11} & I_{12} \\ I_{21} & I_{22} \end{bmatrix}, B_{12} = \begin{bmatrix} I_{13} & I_{14} \\ I_{23} & I_{24} \end{bmatrix}, \ldots B_{1,n/2} = \begin{bmatrix} I_{1,n-1} & I_{1,n} \\ I_{2,n-1} & I_{2,n} \end{bmatrix}$$

Step 2: Z-scan each B_{ij} and generate m×n row vectors A. For example.

$$B_{11} = \begin{bmatrix} I_{11} & I_{12} \\ I_{21} & I_{22} \end{bmatrix} = A_{11} = [\, I_{11}\ I_{12}\ I_{21}\ I_{22}\,]$$

Step 3: Express these row matrices as an intermediate matrix M.

$$M = \begin{pmatrix} A_{11} \\ A_{12} \\ . \\ . \\ A_{m/2,n/2} \end{pmatrix} = \begin{pmatrix} I_{11} & I_{12} & I_{21} & I_{22} \\ I_{13} & I_{14} & I_{23} & I_{24} \\ . & . & . & . \\ . & . & . & . \\ I_{m-1,n-1} & I_{m-1,n} & I_{m,n-1} & I_{m,n} \end{pmatrix}$$

Step 4: Consider filter coefficient matrix

$$C = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

FIND H=M×C.

Step 5: Haar wavelet transform can be divided into four sub-matrices of size $\frac{m}{2} \times \frac{n}{2}$,

$$W = \begin{pmatrix} W_{LL} & W_{HL} \\ W_{LH} & W_{HH} \end{pmatrix}$$

The rearrangement of the elements of H into four sub-matrices will produce the resultant Haar wavelet transform matrix W.

The rearrangements are as follows

a) The elements in the first column of H are filled in W row by row.

b) The elements in the second column of H are filled in W row by row.

c) The elements in the third column of H are filled in W row by row.

d) The elements in the fourth column of H are filled in W row by row.

Improved MM Algorithm:

In the previous section, it can be envisaged that for large matrix multiplications the operation cycles

will get increased. The operation cycle for conventional method (Matthias Hopf and Thomas Ertl, 2000) takes 'n^{2}' whereas systolic array (Chen, P.Y. and E.C. Liao, 2002) architecture requires only '2n+1' cycle to implement MM algorithm. The algorithm used for matrix multiplication here takes only (n-1) cycles for rearrangement of matrix elements and 'n' cycles for 'n×n' MM was designed (Prabir, S., A. Banerjee, 2014). Here, rotation of matrix elements and matrix multiplication will take simultaneously to produce resultant matrix. The proposed architecture also reduces the layout area for hardware implementation. This parallel MM is simple and optimised for the following reasons:

- Each PE is dedicated to calculate a part of result matrix, a sub-matrix or a block.
- Each PE is interconnected as array.
- Each block has equal memory even though if it has any necessity for buffers and equal delay for all the blocks thus avoiding 'convey effect'.

A. Multiplication algorithm :

Assume two 'n × n' matrices A and B are partitioned into *p* blocks $A_{i,j}$ and $B_{i,j}$ ($0 \le i, j \le \sqrt{p}$) of size ($n/\sqrt{p}$) each. The sub-blocks of 'A' and 'B' residing with the processor (i, j) are denoted by A_{ij} and B_{ij}, respectively, where $0 \le i \le \sqrt{p}$ and $0 \le j \le \sqrt{p}$. In the first phase of the execution of the algorithm, the data in the two point matrices are aligned in such a manner that the corresponding sub-matrices at each processor can be multiplied together locally. That is done by sending the block A_{ij} to processor (i, (j + 1) mod $\sqrt{p}$), and the block B_{ij} to processor ((i + j) mod $\sqrt{p}$, j). The copied sub-blocks are then multiplied together. Now, the A sub-blocks are rolled one step to the left and B sub-blocks are rolled one step up and the newly copied sub-blocks are multiplied and the results are added to the partial results in C sub-blocks. The multiplication of 'A' and 'B' is complete after $\sqrt{p}$ steps of rolling sub-blocks of 'A' and 'B' left and up, respectively, and multiplying the incoming sub-blocks in each processor.

B. Implementation stages:

In this algorithm, the computations of $\sqrt{p}$ processes of the i^{th} row and j^{th} column is scheduled in a manner such that, any time, each process is using different blocks $A_{i,j}$ and $B_{i,j}$. These blocks can be systematically rotated among the processes after every sub-MM so that every process obtains a fresh $A_{i,j}$ and $B_{i,j}$ after each rotation.

Step 1: Align the blocks of A and B in such a way that each process multiplies its local sub-matrices. This is done by shifting all sub-matrices $A_{i,j}$ to the left (with wraparound) by 'i' steps and all sub-matrices $B_{i,j}$ up (with wraparound) by 'j' steps.

Step 2: Perform local block multiplication.

Step 3: Each block of A moves one step left and each block of B moves one step up (again with wraparound).

Step 4: Perform next block multiplication; add to partial result, repeat until all blocks have been multiplied.

C. Flowchart diagram:

A flowchart diagram of the modified MM algorithm is shown in Fig. 2. Through this algorithm, matrix multiplication is implemented following these steps.

(i) Initialize the matrices A and B, with first and second matrices, respectively, and initialize the result matrix R with zeros.

(ii) Perform the shift operations for the first matrix (A). To perform the shift operation of the first matrix, the number of rows has been considered. If we want to perform the shift operation for second row, then consider only (k − 1) left shift, where k is number of rows.

(iii) Perform the shift operations for the second matrix (B). To perform the shift operation of the second matrix, considers only (k − 1) up shifting, where k is number of columns.

(iv) Perform the multiplication for each element, of every individual position. That is, perform the multiplication of elements like first row first column of a matrix with first row first column of another matrix, first row second column of a matrix with first row second column of another matrix and so on.

(v) Perform one left shift operation for each element of first (A) matrix and up shift for one time of each element of second (B) matrix.

(vi) Perform the multiplication like step (iv) and add the partial results.

(vii) Perform shift operation like step (v).

(viii) Perform the multiplication like step (iv) and add the partial results with previously generated results.

(ix) The process continues until all the elements multiplication.

RESULTS AND DISCUSSIONS

A. Hardware Estimation:

Let N×N be image size and the number of segments will be $\frac{N}{4} \times \frac{N}{4}$. For image size 64×64, the image is segmented in a 4×4 matrix. So that it is split up into 256 separate segments, the number of multipliers and adders required for each segment will be 48 respectively.

B. Comparison:

Table I shows the comparison of the existing architectures and the proposed architecture for 2-D Haar DWT in terms of multipliers, adders and computation time. Compared with the design in (Baofeng Li, 2009), the proposed design uses the same number of adders, 16 less number of multipliers and the calculations are performed with less computational cycles.

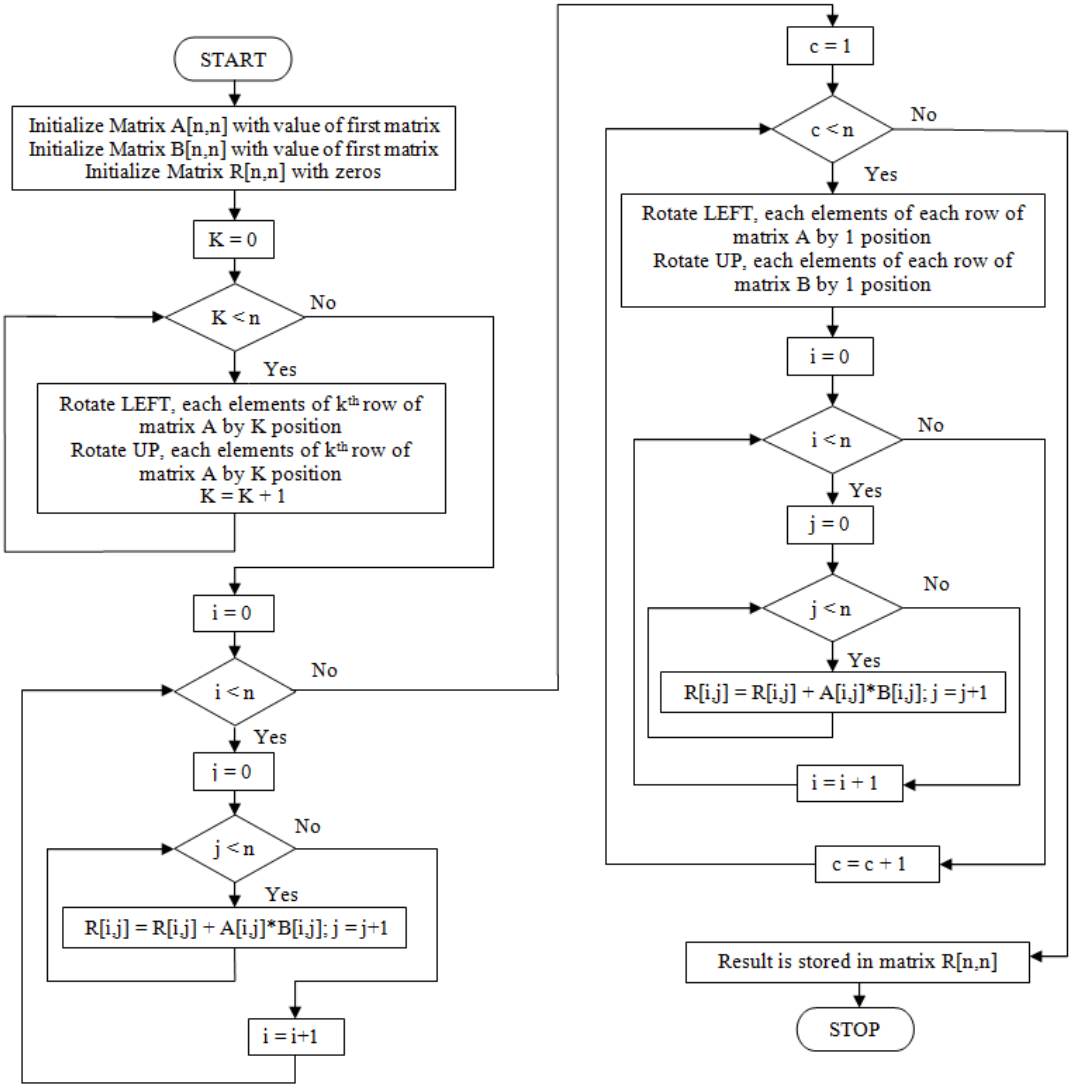
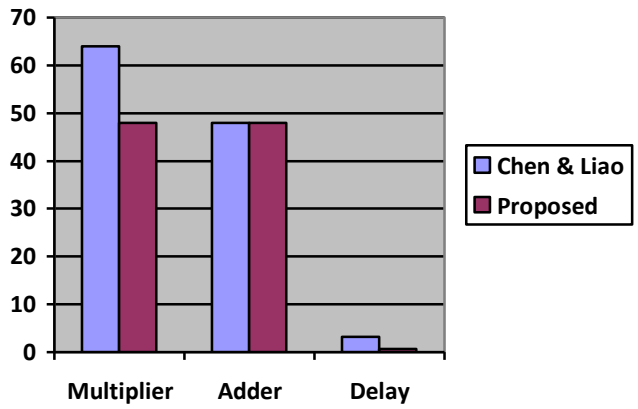


Fig. 2: Flowchart diagram of MM Operation.

Table 1: Comparison of Synthesis Results.

Designs	Multiplier	Adder	Delay
Chen & Liao	64	48	3.153 ηs
Proposed	48	48	0.605 ηs



Fi g. 3: Comparison Chart of Synthesis Results for Existing and Proposed Architecture.

C. RTL View Of Proposed Architecture:

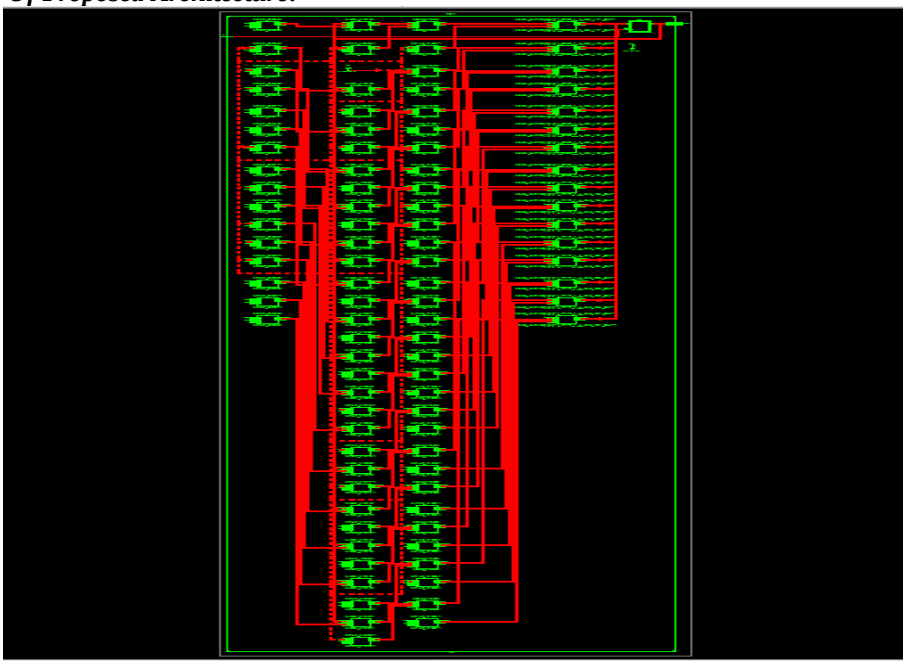


Fig. 4: RTL view of Proposed architecture.

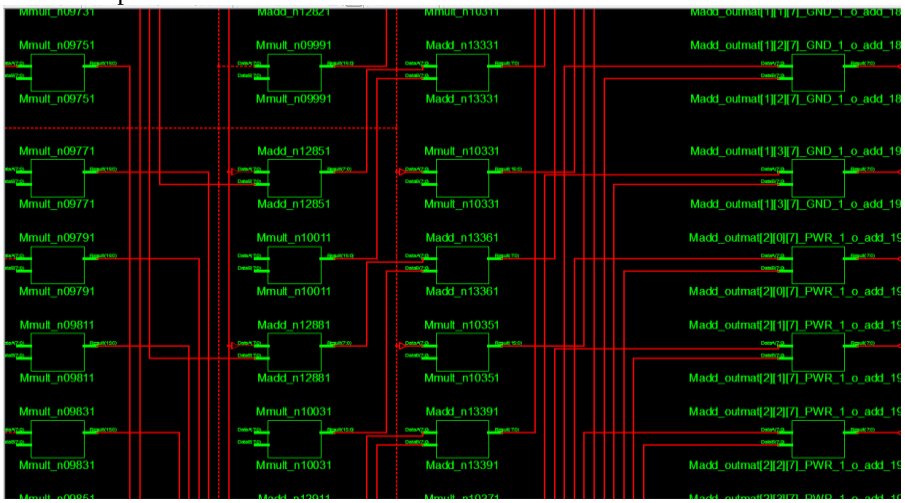


Fig. 5: Elaborated View Of RTL Schematic.

The RTL schematic view of proposed architecture is shown in Fig. 4 and Fig. 5.

The device utilization summary is given below:

HDL Synthesis Report:

Multipliers	-	48
Adder/subtractor	-	48
128 bit D type Flipflop	-	1

Advanced HDL synthesis Report:

2x2-to-8-bit MAC	-	13
2x3-to-8-bit MAC	-	14
3x2-to-8-bit MAC	-	11
3x3-to-8-bit MAC	-	10

Conclusions:

With our proposed architecture, the architecture for 2-D Haar DWT consumes less number of multipliers and computational cycles. Based on this method, a new algorithm is used for matrix multiplication. Furthermore, the proposed system will be developed using Verilog-HDL and synthesized in Cadence RTL compiler using typical libraries of TSMC 0.18 um technology. The synthesized Verilog netlist and their respective design constraints file (SDC) are imported to Cadence SoC Encounter and are used to generate automated layout from standard cells and placement and routing.

REFERENCES

Baofeng Li, Yong Dou, Haifang Zhou and Xingming Zhou, 2009. “FPGA accelerator for wavelet-based automated global image registration,” *EURASIP J. Embedded Syst.*, pp: 1-10.

Bravo, I., P. Jimenez, M. Mazo, J.L. Lazaro, J.J. De las Heras, A. Gardel, 2007. 'Different proposal to matrix multiplication based on FPGAs'.Proc. IEEE Int. Symp. Industrial Electronics (ISIE), pp: 1709-1714.

Graves, C. and C. Gloster, 1998. "Use of dynamically reconfigurable logic in adaptive wavelet packet applications," *Proc. of the 5th* Canadian Workshop on Field-Programmable Devices.

Idris, M., N.M. Noor, E.M. Tamil, Z. Razak, H. Arof, 2010. 'Parallel matrix multiplication design for monocular SLAM'. Proc. IEEE Fourth Asia Int. Conf. Mathematical/Analytical Modeling and Computer Simulation (AMS), pp: 492-497.

Haapala, K., P. Kolinummi, T. Hamalainen and J. Saarinen, 2000. "Parallel DSP implementation of wavelet transform in image compression," Proc. of ISCAS IEEE International Symposium *on Circuits and Systems*, 5: 89-92.

Martina, M., G. Masera, G. Piccinini and M. Zamboni, 2000. "A VLSI Architecture for IWT (Integer Wavelet Transform)," *Proc. of 43* Midwest Symposium on Circuits and Systems, pp: 1174-1177.

Matthias Hopf and Thomas Ertl, 2000. "Hardware Accelerated Wavelet Transformations," Proc. of EG/IEEE TCVG Symposium on Visualization VisSym, pp: 93-103.

Chen, P.Y. and E.C. Liao, 2002. "A new algorithm for Haar discrete wavelet transform," IEEE International Symposium on Intelligent Signal Processing and Communication Systems, pp: 453-457.

Prabir, S., A. Banerjee, 2014. 'Improved matrix multiplier design for high-speed digital signal processing applications'. IET Circuits Devices Syst., 8(1): 27-37.

Snopce, H., L. Elmazi, 2008. 'The importance of using the linear transformation matrix in determining the number of processing elements in 2-dimensional systolic array for the algorithm of matrix-matrix multiplication'. Proc. IEEE Int. Conf. Information Technology Interfaces, pp: 885-892.