International Journal For Technological Research In Engineering
Volume 4, Issue 11, July-2017 *ISSN (Online): 2347 - 4718*

VHDL IMPLEMENTATION OF 2D-DWT ARCHITECTURE OF SIGNAL AND MEMORY OPTIMIZATION

Shayna Bareja[1], Tilak Raj[2]
[1]PG Student, [2]Assistant Professor
ECE Department, S(PG)ITM Rewari, Haryana, India

Abstract: The digital data can be transformed using Discrete Wavelet Transform (DWT). The imagesneed to be transformed without loosing of information. The Discrete Wavelet Transform(DWT) was based on time-scale representation, which provides efficient multi-resolution. Here Low Pass filter coefficients and the High Pass filter coefficients filter give lossless mode of information as per the JPEG 2000 Standard. The new JPEG2000 and MPEG4 still image and video compression standards are based upon the DWT and are shown to produce superior results over their previous incarnations that do not use the DWT. The discrete wavelet transform (DWT) is being increasingly used for image coding. This is dueto the fact that DWT supports features like progressive image transmission (by quality, by resolution), ease of transformed image manipulation, region of interest coding, etc. DWT has traditionally been implemented by convolution. Such an Implementation demands both a large number of computations and a large memory features that are not desirable for either high-speed or low-power applications. Recently, Available on chip memory storage and external memory bandwidth determine the range of parallel architecture choices and the degree of design scalability and signal optimization.

Keywords: Discrete wavelet transform; signal and memory optimization; VHDL; 2D architecture.

I. INTRODUCTION

Recently, there has been a tremendous increase in the application of wavelets in many scientific disciplines. Typical applications of wavelets include signal and image processing numerical analysis, with memory optimization ,.While the wavelet transform offers a wide variety of useful features, it is computation intensive. Furthermore, in contrast to other transforms, such as Fourier transform or discrete cosine transform, it is not block based, which makes it difficult to implement in a parallel representation. Several VLSI and FPGA architectural solutions for the discrete wavelet transform have been proposed in order to meet the real time requirements in many applications. These solutions include parallel filter architectures, linear array architectures, multigridarchitectures , and 2D block based architectures . Most of these implementations are special purpose parallel processors developed for specific wavelet filters and/or wavelet decomposition trees that implement high level abstraction of the standard pyramid algorithm. In addition, some are complex designs requiring extensive user control. Knowles proposed systolic-array-based architectures without multipliers for the 1-D and 2D DWT, but these architectures are not suitable for all wavelets. We proposed a systolic-parallel architecture for the 2D DWT based on the recursive pyramid algorithm, but due to the approximations involved these architectures cannot be used when exact reconstruction is required. We proposed a sequential implementation of the polyphase representation of the DWT suitable for the Xilinx Virtex FPGAs. Yong-Hong et al presented a parallel architecture that can compute low pass and high pass DWT coefficients in the same clock cycle. King-Ch et al implemented the operator correlation algorithm of the 2D DWT.

However, these FPGA implementations are aimed at specific filterbanks, do not support block-based transform, or do not handle block boundaries efficiently. There is a clear need for a fast hardware DWT that allows flexibility in customizing the wavelet transform with regard to the filters being used and the structure of the wavelet decomposition. In many image processing applications, including compression, denoising and enhancement, it is critical to compute the 2D wavelet transform in real-time. Field programmable gate arrays (FPGAs) offer a suitable platform (cost effective and highly flexible) for such an implementation. FPGA-based systems represent a new paradigm in the industry – a shift away from a full custom ASIC solutions for each application, to a single hardware assembly (FPGA) that can be reconfigured to accommodate multiple applications.

II. PROPOSED ARCHITECTURE

Working Principle

A. 2D DWT Architectures for Hardware Implementations

The DWT, as represented by the Mallat style multilevel octave-band decomposition system, which uses a two-channel wavelet filter bank, is very computation intensive. This decomposition can be implemented as a pyramidal recursive filtering operation using the corresponding filter banks as shown in Figure 1. We will refer to it as the standard algorithm. The process for the 2D DWT decomposition for each level is implemented with a cascaded combination of two 1- D wavelet transforms. The standard algorithm is constrained by large latency, a high computational cost and the requirement for a large buffer size to store intermediate results, which makes it impractical for real time applications with memory 40 constraints. An alternative representation, requiring fewer computations, is the lifting algorithm which will be the basis of our implementations in this chapter.

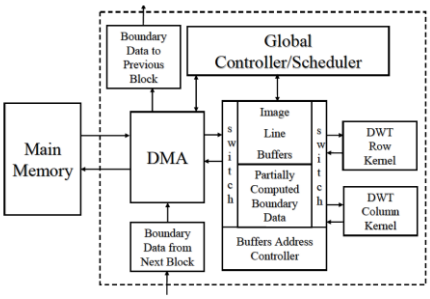

Fig.1 : DWT Unit Block Diagram

B. Existing Architectures for FPGA based parallel implementation of the 2D DWT

Several 2D DWT architectures for parallel implementations were proposed recently, as wavelets gained popularity. Most of these architectures concentrate on saving hardware resources, memory and computations. For example the 1D folded architecture by Chakrabati et al reuses the same logic for both row and column transforms. While it achieves lower hardware resources, it requires high memory bandwidth. For an NxN image, 2N2 read and write operations are needed for the 1st level DWT decomposition. The Partitioned DWT architectures by Ritter et, partitions DWT into small 2D Blocks to achieve lower memory bandwidth and low on-chip storage, but it produces block artifacts along the boundaries between partitions. The recursive pyramid algorithm by Vishwanathet. al., takes advantage of different clock rates at different DWT levels to intermix the next level computations with current calculations. It requires a large on-chip memory and complex scheduling for interleaving the DWT levels. The Generic 2D biorthogonal DWT by Benkrid et al uses separate architectures to calculate each DWT level. It achieves full utilization of memory bandwidth – one write and one read per pixel, but with massive on-chip storage requirements. The Modified folded architectures for SPHIT image compression by Fry and Hauck uses the same filter assembly for both rows and columns with pixels read from one memory port, transposed for the column transform, and written to another memory port. It achieves a DWT runtime of ¾ N2 cycles for an NxN image, but it assumes 64-bit wide memory ports to allow filtering of 4 rows at a time of 16 bit pixels, which may not be practical for all systems.

C. Proposed Parallel Architectures for the 2D DWT

The standard DWT algorithm operates on the whole image in a sequential manner. An improved implementation would partition the image into several blocks and 43 operate on each block independently and in a parallel manner, and then would merge the results to complete the DWT. While this architecture still requires the same intensive computations of the recursive filtering operation and the same memory requirements, the computation can be sped-up if one uses a multi-processor system or identical parallel hardware implementations of the filtering blocks that can operate on multiple image blocks simultaneously. A known disadvantage of such an approach is that it requires data exchanges between neighboring blocks at each decomposition level of the discrete wavelet transform, and hence an additional overhead due to inter-processor communications. We consider three parallel implementations based on the lifting factorization of the DWT. The standard overlapping algorithm shown in Figure eliminates the blocking artifacts and imposes relatively simple control complexity, but has high computational cost and requires high on-chip buffering of data. For an NxN image, using DWT filters of length less than or equal to L, and partitioned into S Blocks, the number of additional filtering operations for a 1 level 2D DWT decomposition vs. a non-overlapped approach is: 2N*L*(S-1).

III. THEORETICAL FRAMEWORK MEMORY OPTIMIZATION

A. Memory Bandwidth Considerations and Storage Calculations

For a DWT filter pair and an image of N x N pixels, denoting:

Fl - length of the longest filter

J - DWT decomposition levels

S - Number of DWT line processors (blocks/stripes)

Consider the stripe-parallel design shown in Figure 3.6. After the completion of 1 level DWT decomposition, the number of transitional boundary states generated at the first boundary of B1 and B2 is:

$M_b1=\lfloor ½ F \rfloor *N$

$M_b2=\lceil ½ F \rceil *N$

From B1

$M_b1=\lfloor ½ F \rfloor *N* ½$

From B2

$M_b2=\lceil ½ F \rceil *N* ½$

where memory is measured here in number of pixels, $\lceil\ \rceil$ and $\lfloor\ \rfloor$ are the ceiling and the floor operators to accommodate odd length DWT filters at the stripe boundaries.

This results from the absence of image data along the boundaries of B1 and B2 required to complete the filtering operations. After the completion of 2 decomposition levels additional transitional boundary states are generated at the same boundary

From B1

$M_b1=\lfloor ½ F \rfloor *N* ½$

From B2

$M_b2=\lceil ½ F \rceil *N* ½$

Hence the memory required to hold transitional boundary states for each boundary is

$M_1=F_l\sum_{i=0}^{j-1} N* (½)^i$

$M_{total}=F_l\sum_{i=0}^{j-1} N* (½)(s-1)$

To minimize external memory I/O bandwidth, the decision was made earlier to use

the cascaded architecture, i.e., pipeline row and column filtering. Assuming a FIFO

buffer length of *N* (image width), the memory required for row buffering is:

for one block

$M_{FIFO} = F_l * N$

and total needed memory for FIFO buffers is

$M_{FIFO\text{- }total} = F_l * N * S$

For example: for an image of 512x512 pixels, a 3 level (9,7) DWT decomposition
with a partition size of 4 stripes, the total required on-chip RAM (BRAM) measured
in pixels is: 9*(512+256+128)*4 + 9*512*4 = 42K bytes
Actual required BRAM may need to be 84K bytes to account for dynamic expansion
in DWT domain.

B. For signal optimization

The pyramidal algorithm for 2-D DWT with separable wavelet bases is given by the following equations.

$X^s(i,j)= \sum_{k=0}^{L-1}\sum_{k=0}^{L-1} h_1(k)h2(k)X^{s-1}(2i-k,2j-k)$

$Y^s(i,j)= \sum_{k=0}^{L-1}\sum_{k=0}^{L-1} g_1(k)h2(k)X^{s-1}(2i-k,2j-k)$

$U^s(i,j)= \sum_{k=0}^{L-1}\sum_{k=0}^{L-1} g_1(k)h2(k)X^{s-1}(2i-k,2j-k)$

$V^s(i,j)= \sum_{k=0}^{L-1}\sum_{k=0}^{L-1} g_1(k)h2(k)X^{s-1}(2i-k,2j-k)$

Where $X^s(i,j)$, $Y^s(i,j)$, $U^s(i,j)$ and $V^s(i,j)$ are the coefficients of s-th stage of 2-D DWT and ,I,j = 0, 1 … … . , N− 1,. L represents the length of the low-pass and high-pass filters used for decomposing the input data matrix of size (N x N) into four sub-bands. The pyramidal algorithm pertaining to the low-low sub-band computation $[X^s(i,j)]$ may be decomposed into two stages of computation as:

$X^s(2i,2j)= \sum_{k=0}^{L-1}\sum_{k=0}^{L-1} h_2(k)W^s2(k)(2i,2j-k)$

$W^s(2i,2j)= \sum_{k=0}^{L-1}\sum_{k=0}^{L-1} h_1(k)X^{s-1}2(k)(2i-K,j)$

in z-domain representation:

$W^s(Z_1,Z_2)=X^{s-1}\ (Z_1,Z_2)\sum_{k=0}^{L-1} h_1(k)(Z_1\text{-}K)$

$X^s(Z_1,Z_2)=W^s\ (Z_1,Z_2)\sum_{k=0}^{L-1} h2(k)(Z_2\text{-}K)$

Where, the input and output of equation are time multiplexed in z-domain. Equations can also be derived for $Y^s(i,j)$, $U^s(i,j)$ and $V^s(i,j)$
in z-domain and the 2-D DWT, therefore may be computed.
Design of an Efficient VLSI Architecture for 2D DWT Wavelet Image Processing
The selected low pass or high pass filter are FIR (finite impulse response) filters. The transfer functions for these filters are as, Transfer function for low pass filter,

$H(z)=h_0H^{-1} + h_1H^{-1} + h_2H^{-1} + h_3H^{-1} + h_{L-1}H^{-L}$

$G(z)=g_0H^{-1} + g_1H^{-1} + g_2H^{-1} + g_3H^{-1} + g_{L-1}H^{-L}$

IV. IMPLEMENTATION RESULTS AND DISCUSSIONS

A. IMPLEMENTATION RESULTS

RESULT OF MEMORY OPTIMIZATION

Table1: Resources Utilization for the Overlap-State Implementation

	ONE DWT MODEL	TWO PARRLEL DWT MODEL
Number of Slice	3380 (10% of total) available slices)	7267 (22%)
Number of Slice Flip Flops	3488 (5%)	7499 (11%)
Number of input LUTs	5455 (8%)	11729 (18%)
Number of BRAMS	26 (8%)	56 (17%)
Number of Multipliers	16 (5%)	34 (10%)

Table2: Resources Utilization and Throughput Comparisons to other Optimized Methods

Architecture	Cast Inc. (LB-2D)	"Software pipelined"	Modified Folded for SPIHT
Device	Virtex II 2V500	Virtex II 2V500	Virtex-E
Number of Slices	2227	986	(62% of chip resources)
Clock 100 Frequency (MHz)	51	98	75
Performance (Msamples/sec)	9	98	100
Image Size	256x256	1024x1024	512x512
DWT Trans. Level	5	5	5

B.Performance Evaluation

(i)for memory optimization

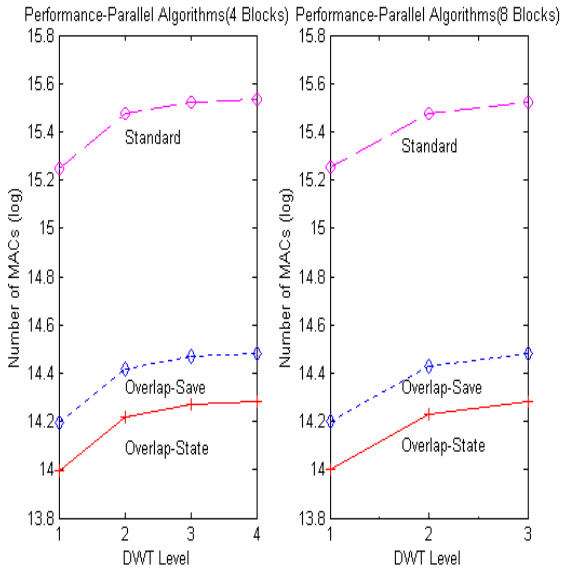

Fig2 . Performance algorithms

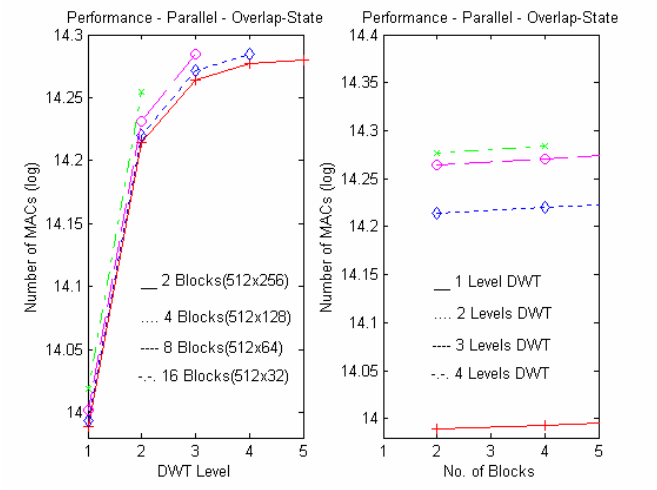

Fig 3. Performance-parallel-overlap-state

(ii) FOR SIGNAL OPTIMIZATION

Fig4 (a)Low-pass filtered output image (b) High-pass filtered output image

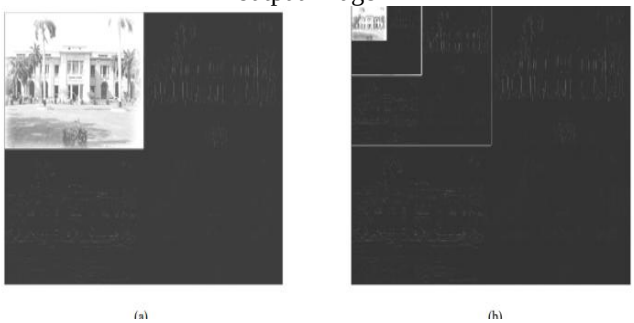

Fig5 .one-level decomposed output image (b) Third-level decomposed output image

V. CONCLUSIONS

We presented, in thesis, a methodology for parallel implementation of memory and signal optimization using DWT on FPGAs. We investigated and analyzed parallel and efficient hardware implementations targeting state-of-the-art FPGAs. We addressed practical considerations and various design choices and decisions at all design stages to achieve an efficient DWT implementation, subject to a given set of constraints and limitations. We presented a specific optimization representation for the DWT that provides architectures suitable for efficient hardware implementation, and a novel data transfer method that provides seamless handling of boundary and transitional states associated with parallel implementations.Also In recent years, several architectures have been proposed for 2-d discrete wavelet transform. However, the hardware of these architectures needs to be further improved. Therefore, in this paper, we have proposed an efficient recursive architecture for 2-D DWT. The advantages of the proposed architecture are saving adders, multipliers, simple control complexity, and complete hardware utilization, making this design suitable for image processing systems.

REFERENCES

[1] M. N. Do and M. Vetterli, “The contourlet transform: an efficient directional multiresolution image representation”, IEEE Trans. Image Processing, vol. 14, pp. 2091-2106, Dec. 2005.

[2] E. L. Pennec and S. Mallat, “Sparse geometric image representations with bandelets”, IEEE Trans. Image Processing, Vol. 14, pp. 423-438, Apr. 2005.

[3] V. Velisavljevic, et al., "Directionlets: anisotropic multidirectional representation with separable filtering", IEEE Trans. Image Processing, Vol. 15, pp.1916-1933, July 2006.

[4] N. G. Kingsbury, “Complex wavelets for shift invariant analysis and filtering of signals”, Applied Computational Harmonic Anal, vol. 10, no. 3, pp. 234-253, May 2001.

[5] S. G. Mallat and Z. Zhang, “Matching pursuits with time frequency dictionaries”, IEEE Trans. Signal Processing, Vol. 41, pp. 3397-3415, Dec. 1993.

[6] S. S. Chen, et al., “Atomic Decomposition by Basis Pursuit”, SIAM J. Scientific Comp., vol. 20, pp. 33-61, 1999.

[7] T. H. Reeves and N. G. Kingsbury, “Overcomplete image coding using iterative projection-based noise shaping”, in Proc. Int. Conf. Image Processing, Rochester, NY, Sept 2002.

[8] N. G. Kingsbury and T. H. Reeves, "Redundant representation with complex wavelets: how to achieve sparsity", in Proc. Int. Conf. Image Processing, Barcelona, Sept. 2003.

[9] B. Wang, et al., “An investigation of 3D dual-tree wavelet transform for video coding”, in Proc. Int. Conf. Image Processing, Singapore, Oct. 2004.

[10] B. Wang, et al., “ Video coding using 3-D dual-tree wavelet transforms”, in Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, Philadelphia, Mar. 2005.

[11] R. W. Buccigrossi and E. P. Simoncelli, “Image compression via joint statistical characterization in the wavelet domain,” IEEE Trans. Image Processing, vol. 8, pp. 1688–1701, Dec. 1999.

[12] J. Liu and P. Moulin, “Information-theoretic analysis of interscale and intrascale dependencies between image wavelet coefficients,” IEEE Trans. Image Processing, vol. 10, no. 11, pp. 1647–1658, 2001.

[13] R.M. Figuerasi Ventura, et al., "Low-rate and

flexible image coding with redundant representations", IEEE Trans. Image Processing, Vol. 15, No. 3, Mar. 2006.

[14] A. Said and W. A. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," IEEE Trans. Circuits and System for Video Tech., vol. 6, pp. 243–250, Jun 1996.

[15] D. Taubman, "High performance scalable image compression with ebcot," IEEE Trans. Image Processing, Vol. 9, pp. 1158–1170, Jul 2000.

