An Efficient VLSI Implementation of Lifting Based Forward Discrete Wavelet Transform Processor for JPEG200

M.S.BHUYAN [1], NOWSHAD AMIN [2], MD.AZRUL HASNI MADESA [1]
MD.SHABIUL ISLAM [1]
Faculty of Engineering [1], Department of Electrical, Electronic and System Engineering, Solar Energy Research Institute Faculty of Engineering [2]
Multimedia University [1], University Kebangsaan Malaysia [2]
63100 Cyberjaya, Selangor [1], *43600 UKM, Bangi, Selangor* [2]
MALAYSIA [1, 2]

Abstract: - This paper describes the hardware design flow of lifting based two-dimensional (2-D) Forward Discrete Wavelet Transform (FDWT) processor for JPEG 2000. In order to build high quality image of JPEG 2000 codec, an effective 2-D FDWT algorithm has been performed on input image file to get the decomposed image coefficients. The Lifting Scheme reduces the number of operations execution steps to almost one-half of those needed with a conventional convolution approach. In addition, the Lifting Scheme is amenable to "in-place" computation, so that the FDWT can be implemented in low memory systems. Initially, the lifting based 2-D FDWT algorithm has been developed using Matlab. The developed codes are then translated into behavioral level of FDWT algorithm in VHDL. The FDWT modules were simulated, synthesized, and optimized using Altera design tools. The final design was verified with VHDL test benches and Matlab image processing tools. Comparison of simulation results between Matlab and VHDL was done to verify the proper functionality of the developed module. The motivation in designing the hardware modules of the FDWT was to reduce its complexity, enhance its performance and to make it suitable development on a reconfigurable FPGA based platform for VLSI implementation. Results of the decomposition for test image validate the design. The entire system runs at 215 MHz clock frequency and reaches a speed performance suitable for several real-time applications.

Key Words: - VLSI**,** DWT, Lifting, JPEG 2000, Synthesis.

1 Introduction

The importance of visual communications has increased tremendously in the last few decades. The development of new technologies and communication networks creates new needs and stimulates the introduction of new functionalities. The current standards in the field of still image coding are inadequate for producing the best quality of performance. To address this concern, ISO committee came up with a new coding system, JPEG2000 [1]. JPEG2000 is intended to provide subjective image quality performance superior to other existing standard image file formats viz. JPEG, BMP, and GIF etc. In addition, JPEG2000 includes many modern features such as lossless to lossy coding with the same algorithm, scalability, etc. This new ISO/ITU-T standard has shown to provide superior coding efficiency to the previous standards. The techniques enabling all new features of JPEG 2000 are a Discrete Wavelet Transform (DWT) followed by an arithmetic coding [1]. The full-frame nature of DWT de-correlates the image over a larger scale and eliminates blocking artifacts at high compression ratios. The advantage of DWT over Fourier Transform is DWT performs multi-resolution analysis of signal with localization in both time and frequency. The DWT decomposes a digital image into different subbands so that the lower frequency sub bands have finer frequency resolution and coarser time resolution compared to the higher frequency subbands. The DWT is being increasingly used for image compression due to the fact that the DWT supports features like progressive image transformation (by quality and resolution) ease of compressed image manipulation, region of interest coding etc. The interest in DWT is growing tremendously in recent years due to its adoptions in JPEG2000 and MPEG-4 [1]. The multi-resolution representation derived from DWT time-frequency decomposition demonstrates extraordinary advantages in signal analysis and compression widely used in image processing, communication

and robotics. As a result, efficient VLSI implementations of DWT processor become more and more important recent years.

Since the discovery of DWT by Mallat [2], several architectures for DWT processor have been proposed. The most common one is the filter-bank implementation [2].Recently an alternative implementation has been proposed, known as the Lifting Scheme (LS) [2]. In addition to providing, a significant reduction in memory uses and computational complexity, lifting provides "in-place" computation of the wavelet coefficients by overwriting the memory locations that contain the input sample values. The wavelet coefficients calculated by lifting are identical to those computed by direct filter-bank convolution. Consequently, the number of multiplications and additions compared to the filter-bank approach are reduced, resulting in a more efficient use of power and chip area. Its modular structure is well suitable for hardware implementation. Because of these advantages, the specification of the DWT kernels in JPEG 2000 is only provided in terms of the lifting coefficients [3].

Fig.1 shows an input image is forward transformed before entropy coded in JPEG 2000. To decode the original image, the inverse transform is applied in exactly the reverse order. For simplicity, we have only described the design details of the 2-D FDWT module in this paper. The focus is on the development of the hardware architecture and the corresponding VHDL models. In our design, we have chosen the (5/3) Le Gall wavelet filters. The JPEG2000 standard committee has recommended using these wavelet filters for integer mode operation [3].

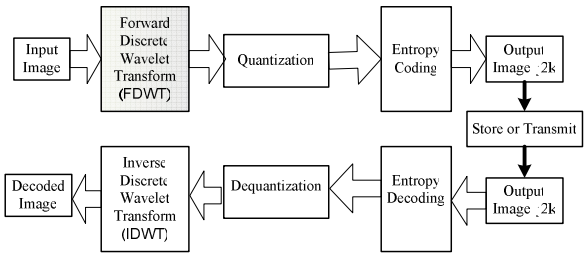


Fig.1 Building block diagram of JPEG 2000

2 Discrete Wavelet Transform and Lifting Scheme

The DWT has been traditionally implemented by means of the Mallat filter bank scheme [4] that includes two main steps: signal decimating and filtering with a pair of Quadrate Mirror Filters (QMFs). The filter-bank can be realized using FIR filters. The process consists of performing a series of dot products between the two filter masks and the signal.

2.1 Lifting Scheme realization

Sweldens [2] proposed the LS where all the operations can be performed in parallel, hence the possibility of a fast implementation. The LS is composed of mainly three steps, namely Split, Predict (P), and Update (U). Fig. 2 describes the LS process. The first step is splitting the input signal x_i into even and odd indexed samples. Then we try to predict the odd samples based on the evens. If the original signal has local correlation, then the prediction should be of high accuracy.

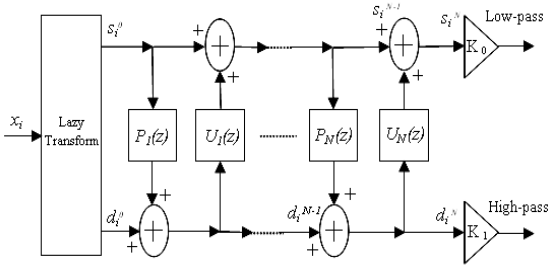


Fig. 2 General diagram of the lifting process

For designing the FDWT module, we have chosen the LS approach. Fig. 3 represents the basic building block of the 1D-FDWT using the Le Gall wavelet filters.

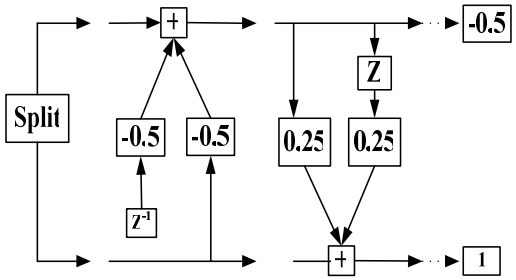


Fig. 3 Implementation of Le Gall wavelet filters

The LS based realization allows Integer Wavelet Transform (IWT). The transform coefficients of the IWT are exactly represented by finite precision numbers, thus allowing for truly lossless encoding. This helped us in reducing number of bits for the sample storage and to use simpler filtering units. IWT is achieved by rounding off the output of filters, before addition or subtraction. This led to a very beneficial class of transforms, in terms of computational complexity and memory requirements. Yet, they are highly dependant on the choice of the factorization of the

polyphase matrix [4]. Equation (1) shows the LS for the FDWT design.

$$y_{2i+1} = \lfloor -0.5(x_{2i} + x_{2i+2}) \rfloor + x_{2i+1}$$
$$y_{2i} = \lfloor 0.25(y_{2i+1} + y_{2i+3}) \rfloor + x_{2i}, where..0 \le i < N/2 \quad (1)$$

3 Design Architecture for the FDWT

In our design a scalable LS based FDWT module has been implemented in FPGA. Design acceleration has been achieved through parallel processing of independent sub-modules, re-usability of image pixel data. Input pixels are accessed through a four sample register (temporary storage), to activate two concurrent predict and update as shown in Fig.4

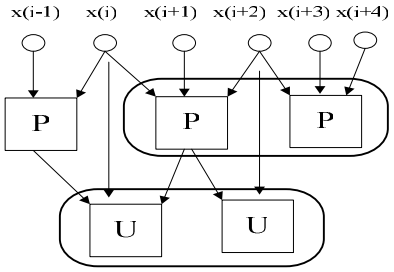


Fig. 4 Parallel processing of input data samples

Registers were used for temporary storage, and reusability of temporary data. Fig 5 and Fig. 6 show the block diagram for the predict and update modules respectively. A fixed precision of 8 bits per pixels were selected. The error introduced by this precision has been proved, through comparing software and hardware implementations, to be within an accepted range. Since all coefficients are multiplies of 2, all multiplications and divisions were replaced by shifting operations.

Starting with the specifications outlined in [1], the DWT module is supposed to have following parameters: picture width, picture height, levels of decomposition and mode of operation. Since we limited our implementation to only one wavelet family, thus we will have only one mode of operation. Our module has parameterized with picture width (512), height (512), and levels of decomposition (maximum levels 7).

4 Matlab Simulation

The approximation and the detailed coefficients of test image were computed first in Matlab platform. We calculated approximation and detailed Coefficients for all the rows of the 256 × 256 input image referred as the 1-D FDWT transform.

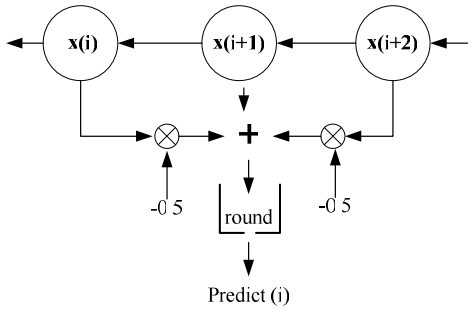


Fig.5 Predict Filter module in FDWT

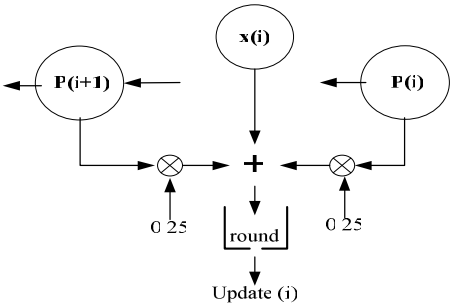


Fig. 6 Update Filter module in FDWT

Next, the same Matlab routine was implemented on this coefficients column wise to obtain the 2-D FDWT coefficients. By this consecutive row and column wise operation on the input image data, we get the level-1 decomposition coefficients. Table 1 show few example coefficients extracted from Matlab simulation.

Table 1
Transformed Coefficients First Level

Approximation Coefficients 1st Level (Column)	Detail Coefficients 1st Level (Column)	Approximation Coefficients 1st Level (Row)	Detail Coefficients 1st Level (Row)
0	0	0	0
156	6	117	-156
158	-3	120	-160
156	6	120	-159
154	0	119	-158
155	3	120	-160
154	0	117	-155
159	3	119	-158
156	1	119	-158
154	-3	121	-161

After level-1 decomposition, approximation and the detailed coefficients were assembled together for the level-2 decomposition. Fig. 7 shows the image of level-1 transform coefficients after the assembling process. The same Matlab routine was again applied on the lowest-frequency subband (LL_1) to get the level-2 approximation and detailed coefficients shown in Fig. 8. Matlab routine continued to get the level-3 transformation shown in Fig. 9.

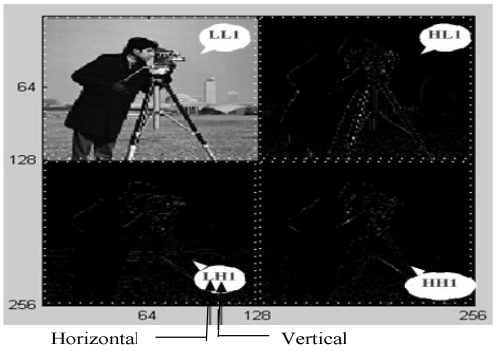


Fig. 7 Level-1 Transform.

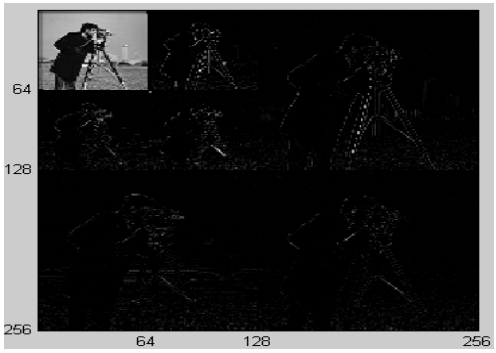


Fig. 8 Level-2 Transform

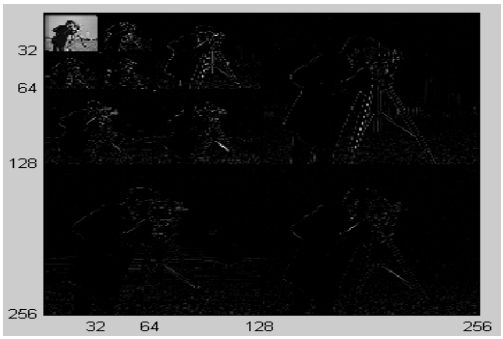


Fig. 9 Level-3 Transform.

5 Modeling 2-D DWT in VHDL

The developed 2-D FDWT module is composed of four major building blocks: DWT_1D_Control, DWT_2D_control, FDWTCore, and Memory. Detailed architecture is shown in Fig. 10. DWT_1D_Control module performs the 1-D transform row/column wise. After all the rows of the image are transformed, two dimensional controller initiates the transformation process in column wise thus complete the level-1 transformation. Level of computation is controlled by DWT_2D_control module through parameter (NL) signal. The memory consists of only one process. First, an array for the input data and the transformed coefficients are created. Input image data are initialized by the memory read (rd) signal for simulation purposes. The memory is able to process a request only if the wr (*write*) or the rd (read) signal is active (high). The incoming addr signal (address*)* is converted into an integer value for accessing the memory array. The FDWTCore block does the actual computation on the image data. Four input and two output registers are used to hold four input data and two (approximation and detail) output data simultaneously. The data are read sequentially from the memory. Computations are based on equation (1). The higher the level of computation (*nNL*), the higher the number of pixels to be processed thereby requiring increased number of computations. For example, when the level of computation is two, computations are first done for all 65536 pixels (256 × 256) in the original block corresponding to level-1 computation and the results are written to the memory. Then computations are done for all the 1024 pixels (128 × 128) in the LL_1 block of the transformed image. Therefore, the total number of computations in this case is 81920. For each computation level, pixel values are first read in row-by-row fashion. This continues until all pixel values from all rows are read and the transformed coefficients are stored in the memory.

6 Functional and Timing simulation of the design

Simulation is critical in verifying developed design behavior. Functional and timing simulation of the FDWT design was done with Mentor Graphics ModelSim-Altera using developed test bench and appropriate stimuli to validate the design

The number of clock cycles required for various levels of computation is shown in Fig. 11. These are the number of cycles starting from the time the start signal is asserted until the ready signal

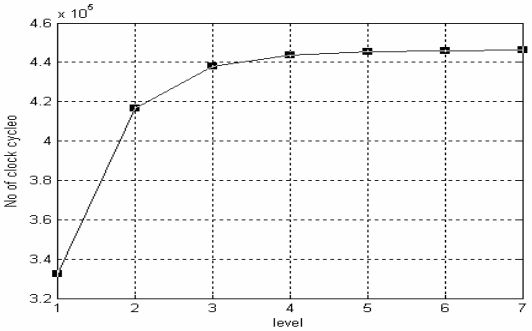


Fig. 11 Clock cycles required for various levels

is issued. Clearly increasing the number of computation levels increases the number of clock cycles required for performing the tasks.

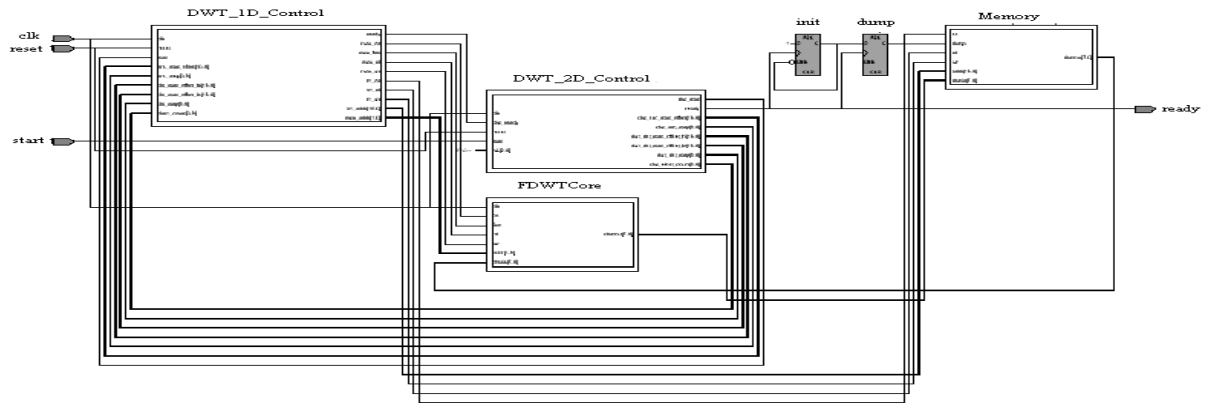
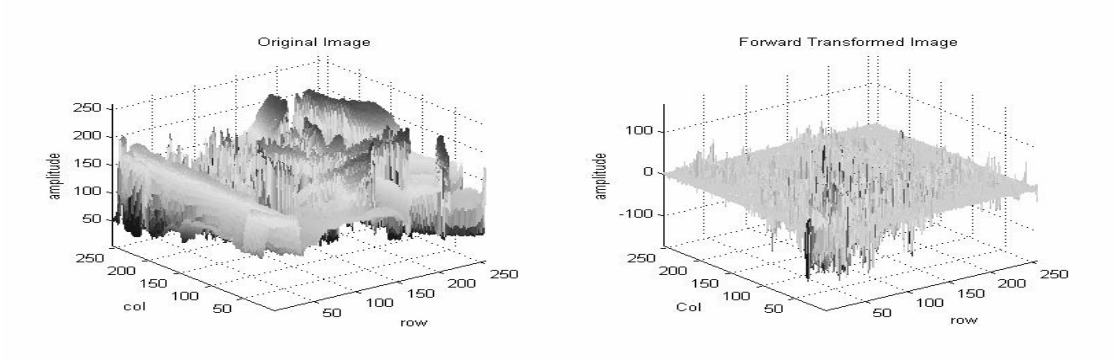


Fig.10 RTL view of the FDWT



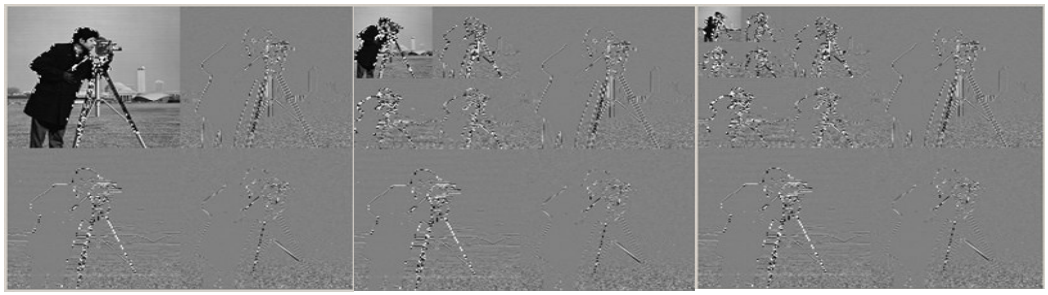
(a) (b)

Fig. 12 Plot showing (a) pixel values for the original input image (b) Transformed output resulting from a level-3 FDWT decomposition of the input image.

Fig 12(a) shows a three-dimensional plot of the pixel intensity values for the input image. It is important to note that the image energy is distributed across the pixel array. Fig 12(b) shows input image after level-3 FDWT transformation. Note that, the *x* and *y-axis* of the plot correspond to the rows and columns of the 256 × 256 output array. The graph shows that the signal energy from the original image has been decorrelated and then concentrated in a much smaller region corresponding to LL_3 subband as expected. This concentration of signal energy shows how large compression ratios can be achieved by removing the extraneous information contained in the less-important sub bands. Thus, only small portions of coefficients are needed to encode for compression.

7 Comparison of Simulation Results

The coefficients extracted from different levels of decomposition in Matlab ware compared with the VHDL simulation. First, we did the comparison for 1-D case. Eight arbitrary pixel data were fed into the 1-D Matlab module. The same data were fed into the 1-D VHDL module. The resulting coefficients (decimal) for three different pixel data sets (Test 1, Test 2, Test 3) shown in Table 2. We can observe the Matlab and the VHDL coefficients for the approximation and the details coefficients are almost the same. Then, we perform the comparison between the coefficients for the complete 2-D FDWT module. The input image consisting of 256 × 256 pixels were fed into the VHDL module. The approximation and the detailed



(a) (b) (c)

Fig.13 Forward Transform (a) Level-1 (b) Level-2 (c) Level-3 in VHDL simulation

coefficients for three different levels of decomposition were compared with the corresponding level of decomposition of the Matlab platform. Fig.13 (a), (b), and (c) were constructed from the 1st, 2nd, and 3rd level of decomposition respectively.

TABLE 2
Coefficients of Matlab vs. VHDL 1-D FDWT

Coeff. No	Matlab Test1	VHDL Test1	Matlab Test2	VHDL Test2	Matlab Test3	VHDL Test3
1	30	31	30	30	25	23
2	29	29	31	30	28	29
3	29	31	32	32	30	28
4	0	28	0	32	0	23
5	1	0	-1	1	-3	0
6	-0	-1	-0	0	2	0
7	2	0	0	0	1	1
8	0	0	-1	-2	-29	-30

8 Synthesis and Implementation Results

Quartus II Integrated Synthesis (QIS) tool was used to synthesize the FDWT design codes into gate-level schematic. QIS includes advanced synthesis options and compiler directives (attributes) to guide the synthesis process to achieve optimal results. Fig 14 shows the schematic view of the FDWTCore module of the 2-D FDWT. Fig 15 shows slice of analysis and synthesis flow summary report generated by Quartus-II for the 2-D FDWT module.

9 Conclusion

The hardware architecture of a 2-D FDWT processor using 256 × 256 pixels block of 8-bit image information as input has been presented. This architecture employs no multiplier and therefore is an attractive alternative to processors employing computationally expensive multipliers. The processor has been modeled in VHDL and validated

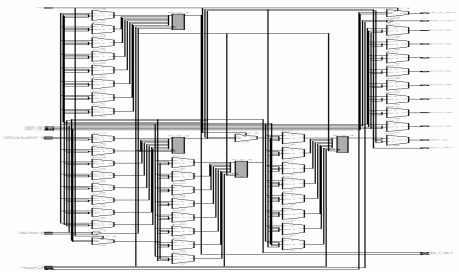


Fig. 14 RTL view of FDWTCore

Flow Status	Successful - Sat Jun 23 01:25:30 2007
Quartus II Version	6.1 Build 201 11/27/2006 SJ Full Version
Revision Name	FDWT_2D_TOP
Top-level Entity Name	FDWT_2D_TOP
Family	Stratix II
Device	EP2S60F1020C4
Timing Models	Final
Met timing requirements	Yes
Logic utilization	< 1 %
Combinational ALUTs	82 / 48,352 (< 1 %)
Dedicated logic registers	79 / 48,352 (< 1 %)
Total registers	79
Total pins	4 / 719 (< 1 %)

Fig. 15. Analysis and Synthesis report of FDWT

using simulation and in-system debugging. The present model is a foundation for development of a comprehensive modeling framework for DWT processors starting from abstract high-level system models to synthesizable models.

References:

[1] ISO/IEC.ISO/IEC.JPEG2000 coding system.
[2] Sweldens, The lifting scheme: A new philosophy in biorthogonal wavelet constructions', Proceedings of SPIE, pp.68-79, 1995.
[3] ISO/IEC.JTC/SC29/WG11Generic Coding of Audio Visual Objects: Final Draft, 1998.
[4] M.S. Bhuyan, Md. Shabiul Islam, Md.Azrul Hasni Madesa, and Masuri Othman. Design of Lifting Based 2-D Discrete Wavelet Transform in VHDL, ISBN: 978-983-42747-7-7, page 224, 2007.