International Journal of Engineering Sciences & Research Technology

(A Peer Reviewed Online Journal)
Impact Factor: 5.164

IJESRT

RESEARCHERID
THOMSON REUTERS

[Kumar *et al.*, 9(7): July, 2020]
IC™ Value: 3.00

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

DESIGN ANALYSIS OF 2-D DWT BASED IMAGE COMPRESSION USING FPGA FOR MEMORY SPEED OPTIMIZER

Vinit Kumar*, D S Gangwar, Amit Kumar & Monika Gupta
*M. Tech Scholar, Dept. of Electronics Engineering
Assistant Professor, Dept. of Electronics Engineering
Faculty of Technology, Uttarakhand Technical University, Dehradun, UK, India

DOI: https://doi.org/10.29121/ijesrt.v9.i7.2020.22

ABSTRACT

Wavelet Transform is successfully applied a number of fields, covering anything from pure mathematics to applied science. Numerous studies, done on wavelet Transform, have proven its advantages in image processing and data compression and have made it a encoding technique in recent data compression standards along with multi- resolution decomposition of signal and image processing applications. Pure software implementations for the Discrete Wavelet Transform (DWT), however, seem the performance bottleneck in real-time systems in terms of performance. Therefore, hardware acceleration for the DWT has developed into topic of contemporary research.

On the compression of image using 2-Dimensional DWT (2D-DWT) two filters are widely-used, a highpass as well as a lowpass filter. Because filter coefficients are irrational numbers, it's advocated that they must be approximated with the use of binary fractions. The truth and efficiency with that your filter coefficients are rationalized within the implementation impacts the compression and critical hardware properties just like throughput and power consumption. An expensive precision representation ensures good compression performance, but at the expense of increased hardware resources and processing time. Conversely, lower precision with the filter coefficients ends up with smaller, faster hardware, but at the expense of poor compression performance.

KEYWORDS: DWT, Xilinx, Image Compression.

1. INTRODUCTION

Image compression is different from binary data compression. When binary data compression techniques are applied to images, the results are not optimal. In lossless compression, the data (such as executables, documents, etc.) are compressed such that when decompressed, it gives an exact replica of the original data. They need to be exactly reproduced when decompressed. For example, the popular PC utilities like WinZip or and Adobe Acrobat perform lossless compression. On the other hand, images need not be reproduced exactly. A 'good' approximation of the original image is enough for most purposes, as long as the error between the original and the compressed image is tolerable. Lossy compression techniques can be used in this application. This is because images have certain statistical properties, which can be exploited by encoders specifically designed for them. Also, some of the finer details in the image can be sacrificed for the sake of saving bandwidth or storage space.

In digital images the neighboring pixels are correlated and therefore contain redundant information. Before the image is compressed, the pixels, which are correlated is to be found. The fundamental components of compression are redundancy and irrelevancy reduction. Redundancy means duplication and irrelevancy means the parts of signal that will not be noticed by the signal receiver, which is the Human Visual System (HVS).

1.1 Principles of Image Compression

The system consists of three main components, namely, the source encoder, the quantizer, and the entropy encoder. The input signal (image) has a lot of redundancies that needs to be removed to achieve compression.

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

These redundancies are not obvious in the time domain. Therefore, some kind of transform such as discrete cosine, Fourier, or wavelet transform is applied to the input signal to bring the signal to the spectral domain. The spectral domain output from the transformer is quantized using some quantizing scheme. The signal then undergoes entropy encoding to generate the compressed signal.

1.2 Short Time Fourier Transform vs. Wavelet Transform

The STFT is a modified version of the Fourier Transform. The Fourier Transform separates the waveform into a sum of sinusoids of different frequencies and identifies their respective amplitudes. Thus it gives us a frequency-amplitude representation of the signal. In STFT, the non-stationary signal is divided into small portions, which are assumed to be stationary. This is done using a window function of a chosen width, which is shifted and multiplied with the signal to obtain the small stationary signals. The Fourier Transform is then applied to each of these portions to obtain the Short Time Fourier transform of the signal.

The problem with STFT goes back to the Heisenberg uncertainty principle which states that it is impossible for one to obtain which frequencies exist at which time instance, but, one can obtain the frequency bands existing in a time interval. This gives rise to the resolution issue where there is a trade-off between the time resolution and frequency resolution. To assume stationarity, the window is supposed to be narrow, which results in a poor frequency resolution, i.e., it is difficult to know the exact frequency components that exist in the signal; only the band of frequencies that exist is obtained. If the width of the window is increased, frequency resolution improves but time resolution becomes poor, i.e., it is difficult to know what frequencies occur at which time intervals. Also, choosing a wide window may violate the condition of stationarity. Consequently, depending on the application, a compromise on the window size has to be made. Once the window function is decided, the frequency and time resolutions are fixed for all frequencies and all times.

The Wavelet Transform solves the above problem to a certain extent. In contrast to STFT, which uses a single analysis window, the Wavelet Transform uses short windows at high frequencies and long windows at low frequencies. This results in multi-resolution analysis by which the signal is analyzed with different resolutions at different frequencies, i.e., both frequency resolution and time resolution vary in the time-frequency plane without violating the Heisenberg inequality. In Wavelet Transform, as frequency increases, the time resolution increases; likewise, as frequency decreases, the frequency resolution increases. Thus, a certain high frequency component can be located more accurately in time than a low frequency component and a low frequency component can be located more accurately in frequency compared to a high frequency component.

1.3 The Need for Efficient DWT Architecture

The properties of Wavelet Transform allow it to be successfully applied to non-stationary signals for analysis and processing, e.g., speech and image processing, data compression, communications, etc. Due to its growing number of applications in various areas, it is necessary to explore the hardware implementation options of the Discrete Wavelet Transform (DWT). An efficient design should take into account aspects such as area, power consumption, throughput, etc. Techniques such as pipelining, distributed arithmetic, etc., help in achieving these requirements. For most applications such as speech, image, audio and video, the most crucial problems are the memory storage and the global data transfer. Therefore, the design should be such that these factors are taken into consideration.

In this thesis, Field Programmable Gate Arrays (FPGAs) are used for hardware implementation of the DWT. FPGAs have application specific integrated circuits (ASICs) characteristics with the advantage of being reconfigurable. They contain an array of logic cells and routing channels (called interconnects) that can be programmed to suite a specific application. At present, the FPGA based ASIC market is rapidly expanding due to demand for DSP applications. FPGA implementation could be challenging as they do not have good arithmetic capabilities when compared with the general purpose DSP processors. However, the most important advantage of using an FPGA is because it is reprogrammable. Any modifications can be easily accomplished and additional features can be added at no cost which is not the case with traditional ASICs.

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

2. MATERIALS AND METHODS

As already stated lifting scheme is one of the most efficient algorithm for the implementation of discrete wavelet transform. But one of the major shortcoming with this scheme is that the lifting coefficients obtained for the implementation of biorthogonal 9/7 wavelet transformation are irrational numbers. Hence the direct irrational coefficient implementation requires lot of hardware resources and the processing time at the cost of slight improvement in the compression performance. On the other hand, lower precision in filter coefficients results in smaller and faster hardware at the cost of compression performance. In addition to this rationalization also determines other critical hardware properties such as throughput and power consumption. Hence it is suggested that they should be optimally rationalized without much affecting the compression performance.

	Irrational value	Rational value
α	-1.5861343420…..	-3/2
β	-0.0529801185…..	-1/16
γ	0.8828110755…..	4/5
δ	0.4435068520…..	15/32
ζ	1.1496043988…..	4√2/5

Irrational and rational lifting coefficients for 9/7 wavelet transform

Table shows the irrational and approximated rational counterpart for 9/7 filter which are considered as a very good alternative to irrational coefficients coefficients are applied to image coding, the compression performance is almost same as that of irrationalized filter coefficient implementation, while the computational complexity is reduced remarkably. For this implementation only two floating point operations are required while other remaining are integer operations whose implementation is quite straightforward.

DWT Architecture

In the classical 2-D wavelet transform, for each stage of the transformation, the process is divided into two steps. The first step is usually 1-D wavelet transform of each row and the second step is 1-D wavelet transform of the columns of the results obtained in the first step. One issue of concern in this case is storage of the intermediate results. Hence the previous parallel architecture's problem is due to the sequential processing of row, and column direction transforms. In the conventional method, the row calculation performance limits the reduction of initial latency, because a column direction transform needs the results of row direction transform. The initial latency means that a start latency of column direction transform requires buffer memory to store the data of row direction results. This problem's solution depends on the architecture of the processor pair and the data scheduling.

For other schemes; i.e., other than lifting, however, the problem of intermediate memory requirement is more complex. Proper operation of other structures requires that for each stage of the 2-D wavelet transform, column processing follows the corresponding row processing. One approach for this case, which is more systematic, is to incorporate some form of intermediate memory buffer to store intermediate results of the processed rows. Sizes of buffers do not need to be as big as the total size of the intermediate results. In this case, after collection of enough data, in each buffer, the corresponding column processing can start. In this way, the intermediate memory is used as a pipeline for the column processing. In this work, we propose an improved architecture focused on the increasing the row calculation performance and reducing the buffer memory size. The row calculation ability can be increased by using lifting algorithm, interlacing, parallel processing and optimized shift-add operations for multiplications, we can improve the column start latency under the column processor's performance besides reducing buffer memory requirement and increasing the speed of computation.

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

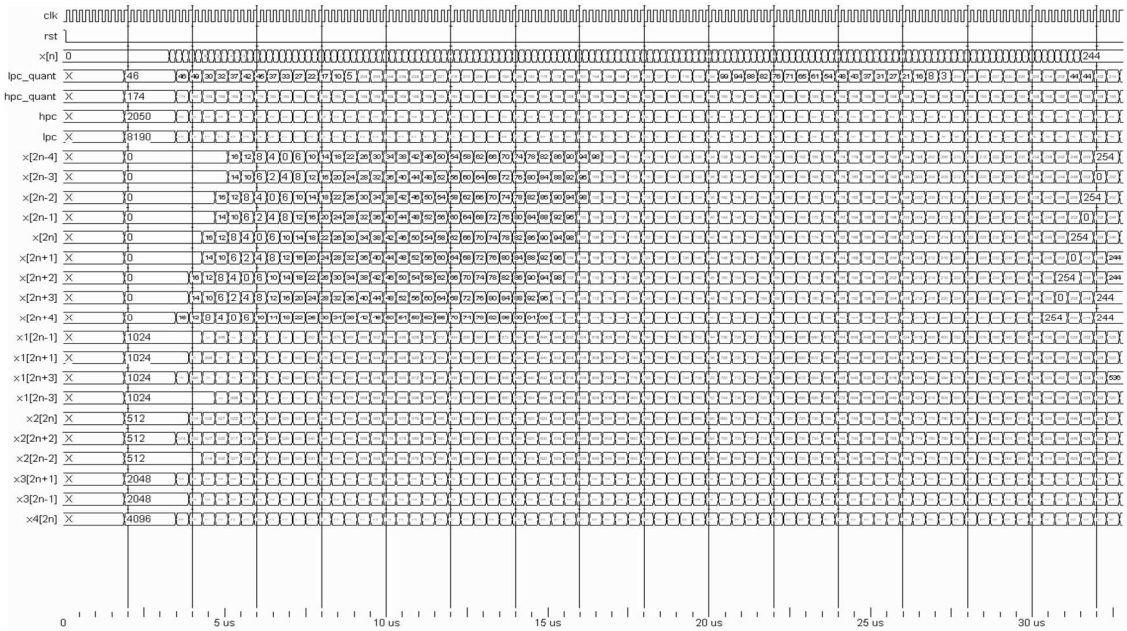
3. RESULTS AND DISCUSSION

Implementation Results and Discussion

Based on proposed memory speed optimized 2-D DWT algorithm based on 9/7 Daubechies filter using lifting scheme is designed and implemented using Xilinx® ISE 9.1 design tools. The entire code is written in VHDL and compilation of code is done on ISE simulator as well as Modelsim SE 6.0d simulator. The whole code is developed using structural based design to tailor the hardware utilization and delay at each step.

As already stated the main component of this design is 1-D DWT processor. This component accepts one pixel per clock cycle which will reduce the number of pins required for an FPGA implementation which is one of the basic design constraint. As soon as it accumulates the sufficient number of pixels, the row processor, starts processing. The results of the row processor output are shown in for clear elaboration.

The calculation of set of wavelet coefficients for pixel 'A', four set of pixels on either side are required. Hence total nine pixels are required for generation of coefficient set, one high pass coefficient and one low pass counterpart. Apart from that symmetric boundary extension is applied to the input data on either side of the image boundary. For synthesis of output DWT coefficients for the first set of input we require four extra pixels across the boundary. This technique is adopted in our design otherwise the first set of output generated by the processor will correspond to the fifth pixel. Thus, in absence of boundary extension the total number of output data will differ from that of input. The padded data boundary extension scheme can also be employed over here. Under this scheme all the extended coefficients are assigned to zero. But this will generate the DWT coefficients of high value for the first input pixel and will require more number of bits to encode which doesn't contain any useful information. This high value will also require more number of bits to encode it which clearly we don't want.



Output waveforms for 1-D DWT decomposition with intermediate results

The outputs of the row processor thus generate are then taken as input of the column processor. For that we require that first of all it accepts the LL set of coefficients and then HH set of coefficients should be given as input to the 1-D processor which is not in accordance with the figure 4.2. This figure clearly shows that LL and HH coefficients are arranged column-wise and we require it row-wise. Hence we need to transpose the set of output results. This implementation will cost us lot of hardware resources which is of the order of N^2. To overcome this issue we have designed the row processor in such a manner that it calculates the set of coefficients column-wise and the corresponding results are stored in row-wise manner for direct calculation

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

of 2-D coefficients. The whole set of functioning is shown in figure 4.5. This will save us the memory required to transpose the input matrix of column processor and hence saves the lot of hardware resources. We have designed the row processor which accepts input as a pixel which is 8-bit wide and hence a constraint. When this component is replicated as column processor will also require input size to be 8-bit wide as same design is reused. Thus the output of the row processor has to be quantized to maintain the same data width. Both the high pass and low pass components are quantized so that their width is modified to eight bits. This will help in easier cascading of the column processor with the row processor. The quantized output is shown in the figure 5.1 under the label 'lpc_quant' and 'hpc_quant'.

Design of a DWT processor using lifting scheme require *predict, update* and *scale* operations to be implemented on hardware. This approach requires equations through given in the previous chapter to be implemented without much hardware complexity. For this purpose the irrational lifting coefficients are rationalized without much affecting compression performance and accelerating the computational speed. As clearly seen from the equations that two output coefficients, one highpass and another lowpass, are generated as a result of DWT decomposition. These coefficients can be seen in the output waveform labeled as 'hpc' and 'lpc' respectively. The other modules which we have designed in VHDL are different adders and shifters which in turn are basic building blocks of multipliers. The different multipliers implemented are $\alpha, \beta, \gamma, \delta, \zeta$ and $1/\zeta$ multipliers as shown in equation. Their VHDL codes are given in appendix provided with this thesis. All these multiplier codes are synthesizable individually and they are implemented via shift-add operation. These multipliers are designed using structural design approach as can be clearly seen in the code. These all multiplier blocks are cascaded together to obtain the overall 1-D DWT implementation and the 2-D DWT component thereafter. The size of input, which the each multiplier accepts, and the output it generates is different for each multiplier and is decided according to the architecture requirement.

In the whole implementation of multiplier modules, 2's compliment is used as standard for data representation and multiplication. Wherever it is required to divide the negative number, the number is first converted into positive number, divided, and again converted back into the negative number. This approach is adopted because it requires minimal hardware (since we have to take 2's complement only for two times, one for converting negative number into positive number and other for converting it back into negative number after division) as compared to other implementations. To check the efficiency of our design we have also calculated the mean square error and root mean square error for very large chunk of data. The whole calculation is presented in Appendix B for clear elaboration. The mean square error value is calculated out to be 3.4823 and that of root mean square error is 1.8661 which indicates the efficient implementation of the Lifting algorithm. We have also calculated the energy preserved by the transformation, which is a direct measure of compression quality and inherent property of biorthogonal transform, and its value comes out to be 99.9326%.

The 2-D DWT implementation is simple extension of 1-D DWT. The data in case of 2-D DWT is picked from image file which is in hexadecimal format. This hexadecimal file 'data_hex.txt' is obtained by MATLAB implementation and this file is stored in the same directory as that of project. The 1-D row processor picks up one pixel at a time from this file. This data is then converted into logic levels for further processing. Once it generates the output set of coefficients it store the result into buffer memory. The whole set of 1-D coefficients are also stored in file 'dwt_1d.txt' for clear elaboration. Once the sufficient number of coefficients are collected, the column processor starts working and it stores its results again in a text file named 'dwt_2d.txt'.All these results of various levels of decomposition is stored in hexadecimal format. The 'dwt_2d.txt' file can be invoked back for next level of decomposition, if required. Proposed 9/7 lifting scheme utilizes only 42% of the total resources of the Spartan-II chip available out of the 50K available. The chip used for the implementation is XC2S50TQ144-5. The memory requirement for this kind of architecture and data flow is only N, i.e. the length of the column required for the storage of the DWT coefficients, for the input image size of N*N. The maximum clock frequency reported by the timing analysis tool is 112.435 MHz. The whole implementation code is given in appendix A for clear elaboration.

4. CONCLUSION

Synthesis filter banks to compute the inverse DWT, i.e., IDWT can be implemented using similar architectures for the corresponding analysis filter banks. Another useful extension to this research can be the modification of

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

the design in order to implement other sorts of filters than the lifting scheme. Considering the encouraging result of the performance estimations, the popular 5/3 filter can be a good candidate for this purpose. Besides this the proposed work is done for monochrome image compression. The work can be extended for colour image implementation either by multiplexing the bits or by multiplying the hardware three times, one for each colour. Further improvements in hardware performance can be obtained by addressing hardware architecture issues such as pipelining, placement and routing and memory access. FPGA implementations of the cascade-polyphase and the rational lifting implementation can be tweaked and optimized to improve hardware performance.

Multiple levels of DWT computation present the problem of growing signal bit widths. Starting with 8-bit image data, the input to each successive DWT level will have wider bit widths, making it impractical to save all bits of precision in memory till the final decomposition level. Intermediate DWT coefficients will have to be truncated after every level, so that data read from and written to memory will have fixed bit widths. For the lifting implementation, where bit widths grow after every filter within a single lifting stage, intermediate signals may have to be truncated within a level. Truncation of DWT coefficients and other intermediate values during the computation of the DWT further impacts performance, and presents another interesting topic for further study.

REFERENCES

[1] Daubechies, "Ten Lectures on Wavelets," *SIAM*, Philadelphia, 1992.
[2] S. Saha, "Image compression from DCT to wavelets," *http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html*
[3] S.G. Mallat, 'A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.11, No. 7, pp. 674-693, July 1989.
[4] W. Sweldens, "The Lifting Scheme: A custom-design construction of biorthogonal Wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, issue 2, pp. 186-200, article no. 15, April 1996.
[5] I. Daubechies, W. Sweldens, "Factoring wavelet transforms into lifting steps,"*Journal of Fourier Analysis and Applications*, vol. 4, No. 3, pp. 247-269, 1998.
[6] R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet transforms that map integers to integers," *Applied and Computational Harmonic Analysis*, vol. 5, issue 3, pp. 332– 369, article no. HA970238, July 1998.
[7] W. Sweldens, "The Lifting Scheme: A construction of second generation Wavelets,"*SIAM Journal on Mathematical Analysis*, vol. 29, No. 2, pp. 511-546, March 1998.
[8] P. Schröder and W. Sweldens, "Spherical wavelets: Efficiently representing functions on the sphere," *Computer Graphics Proceedings (SIGGRAPH 95)*, pp. 161-172, 1995.
[9] J. Kovacevic, W. Sweldens, "Wavelet Families of Increasing Order in Arbitrary Dimensions," *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 480-496, March 2000.
[10] K. Aadra, C. Chakrabarti, T. Acharya, "A VLSI Architecture for Lifting-Based Forward and inverse Wavelet Transform," *IEEE Transactions on Signal Processing*, vol. 50, No. 4, pp. 966 - 977, April, 2002.
[11] M. Vishwanath, R.M. Owens, and M.N. Irwin, "VLSI Architecture for the Discrete Wavelet Transform," *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 42, No. 5, pp. 305-316, 1995.
[12] M. Ferretti, and D. Rizzo, "A Parallel Architecture For The 2-D Discrete Wavelet Transform with Integer Lifting Scheme," *Journal of VLSI signal processing*, vol. 28, pp. 165-115, 2001.
[13] R. C. Gonzalez, R. E. Woods, "*Digital Signal Processing*," Second Edition, Prentice Hall Publications, 2002.
[14] S. Mallat, "*A Wavelet Tour of Signal Processing*," Academic Press, Second Edition,1999.
[15] M. Vetterli, J. Kovačevicć, "*Wavelets and Subband Coding*," Prentice Hall, 1995.
[16] A. Mertins, "*Signal Analysis: Wavelets, Filter banks, Time-Frequency Transforms and Applications*," John Wiley & Sons, 1999.
[17] H.–G. Stark, "*Wavelets and Signal Processing: An Application-Based Introduction*," Springer-Verlag Berlin Heidelberg, 2005.
[18] J. S. Walker, "*A Primer on Wavelets and Their Scientific Applications*," Chapman & Hall/CRC, 1999.
[19] S.G. Mallat, "Multi-frequency Channel Decompositions of Images and Wavelet Models," *IEEE*

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

Transactions on Acoustics, Speech and Signal Processing, vol. 37, No.12, pp. 2091-2110, December 1989.

[20] M.D. Adams, F. Kossentine, "Reversible Integer-To-Integer Wavelet Transform For Image Compression: Performance Evaluation and Analysis," *IEEE Transactions on Image Processing*, vol. 9, No. 6, pp. 1010-1024, 2000.

[21] Z. Guangjun, C. Lizhi and C. Huowang, "A Simple 9/7-Tap Wavelet Filter Based On Lifting Scheme," *IEEE Proceedings of International Conference on Image Processing*, vol. 2, pp. 249-252, October 2001.

[22] S.-M. Kang, Y. Leblebici, "*CMOS Digital Integrated Circuits Analysis & Design*," Tata McGrew Hill Publications, 2004.

[23] http://www.xilinx.com/esp

[24] http://www.altera.com/end-markets/end-index.html

[25] "Spartan-II 2.5V FPGA Family: Complete Data Sheet," *http://direct.xilinx.com/bvdocs/publications/ds001.pdf*, August 2, 2004.

[26] file:///C:/Xilinx/doc/usenglish/help/iseguide/iseguide.htm#html/ise_b_overview.htm. [27] S. Brown, Z. Vranesic, "*Fundamentals of Digital Logic with VHDL Design*," Tata McGraw-Hill Publications, 2005.

[27] D. L. Perry, "*VHDL*," Tata McGrew Hill Publications, 1991.

[28] T. Acharya, P.-S. Tsai, "*JPEG2000 Standard for Image Compression: Concepts, Algorithms and VLSI Architectures*," Wiley Interscience, 2005.