The Nucleus **49**, No. 3 (2012) 181-186

The Nucleus
A Quarterly Scientific Journal of Pakistan
Atomic Energy Commission
NCLEAM, ISSN 0029-5698

A PROCESSOR BASED IMPLEMENTATION OF LAPPED BIORTHOGONAL TRANSFORM FOR JPEG XR COMPRESSION ON FPGA

M. R. REHMAN and *G. RAJA

Department of Electrical Engineering, University of Engineering & Technology, Taxila, Pakistan

(Received May 07, 2012 and accepted in revised form August 06, 2012)

This paper describes a new methodology for implementation of Lapped Biorthogonal Transform (LBT) used in JPEG XR Image compression. Due to sequential nature of LBT, we present a processor based design that executes the instructions of LBT at higher speed. This design can be used in low cost battery operated imaging devices and supports easy up-gradation. We have tested the design on Xilinx Virtex-II Pro FPGA that includes built in PowerPC 405 core which is a 32-bit implementation of RISC PowerPC embedded-environment architecture.

Keywords: JPEG XR, Lapped Biorthogonal Transform, Virtex-II pro

1. Introduction

With the fast development of imaging devices, image size and quality is increasing day by day. Images are almost used in every field of life like medical, commercial, mobile phone, play stations, digital cameras, surveillance, space imaging etc. These applications demands high quality images but that demand leads to the requirement of huge amount of data storage. Thus there is a need to develop such techniques that will decrease the storage size of digital images while keeping the quality of digital image unaffected. Most widely used image compression standard is JPEG which gives high compression ratios at low computational cost. It uses Discrete Cosine Transform (DCT) which is applied on 8x8 image block [1]. However, at low bit rate it produces blocking artifacts. To overcome this limitation a new standard was developed i.e. JPEG2000. This standard uses wavelet transform and provides high compression ratio while image quality is maintained even at low bit rates [2]. But this comes at high computational cost. To preserve the quality of high resolution images, we need high performance image compression technique that will preserve the quality of high definition images while keeping the storage and computational cost as low as possible. To address the limitation of currently used image compression standards, a new image compression standard, JPEG eXtended Range (JPEG XR) has been developed which removes all deficiencies of currently used image compression standards [3]. JPEG XR (ITU-T T.832 | ISO/IEC 29199-2) mainly targets to increase the capabilities of exiting coding techniques and provides high performance at low computational cost [4]. JPEG XR uses lapped biorthogonal transform to convert image samples into frequency domain coefficients [5-6]. LBT is integer transform. It is less computationally expensive than JPEG2000. It reduces blocking artifacts at low bit rates as compared to JPEG. Thus due to less complexity and reduced artifacts, it significantly improves the overall compression performance of JPEG XR [7-8]. To use LBT in real time embedded applications, we need its hardware implementation. Application specific hardware for LBT provides excellent performance but up-gradation of hardware design is very difficult because it requires remodeling of whole hardware design. Pipeline implementation of LBT also provides outstanding performance but requires large amount of memory usage [9]. Large memory requirement increases size and cost of hardware. We have proposed design that will require less memory for LBT operation and easily up-gradable. Rest of the paper is organized as follows. In Section 2, we discuss fundamentals of LBT. Section 3 describes the proposed design for implementation of LBT. Section 4 shows implementation results. Section 5 is about conclusion.

* Corresponding author : gulistan.raja@uettaxila.edu.pk

2. Overview of Lapped Biorthogonal Transform

In JPEG-XR, image is divided into tiles and tiles are consisting of macro blocks. Each macro block is made up of 16 blocks arranged in 4x4 matrix. Block consists of 16 image pixels in dimension of 4x4. Image size should be multiple of 16, if not, we extend height or width of image which can be done by replicating the image sample values at boundaries.

Lapped Biorthogonal transform (LBT) is then applied on image samples. It consists of two key operations.

1. Overlap Pre Filtering (OPF)
2. Forward Core Transform (FCT)

OPF and FCT are applied on image samples in two stages. These stages of LBT are shown in Figure 1.

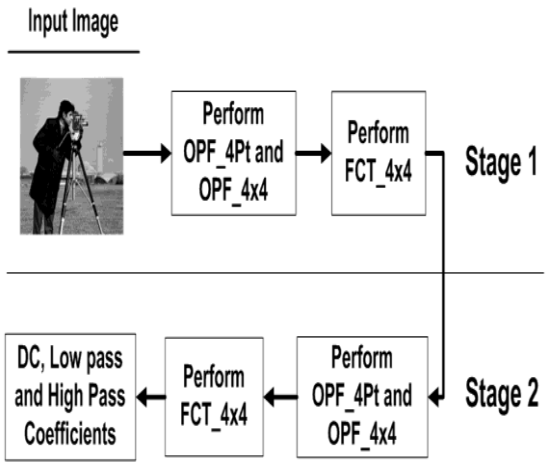

Figure 1. Lapped Biorthogonal Transform Stages.

The various steps performed in OPF and FCT operations are as follows:

1. Overlap pre filter (OPF_4pt) is applied on 2x4 and 4x2 areas between blocks boundaries. Additional filter (OPF_4x4) is also applied on 4x4 area between block boundaries.
2. A forward core transform (FCT_4x4) is applied on 4x4 blocks. This will complete stage 1 of LBT.
3. Each 4x4 block has one DC coefficient. As macro block contains 16 blocks so we have 16 DC coefficients in one macro block. Arrange all 16 DC coefficients of macro blocks in 4x4 DC blocks.
4. In stage 2, Overlap pre filter (OPF_4pt) is applied on 2x4 and 4x2 areas between DC blocks boundaries. Additional filter (OPF_4x4) is also applied on 4x4 area between DC block boundaries.
5. Forward core transform (FCT_4x4) is applied on 4x4 DC blocks to complete stage 2 of LBT. This will results in one DC coefficient, 15 low pass coefficients and 240 high pass coefficients per macro block.

In LBT stages, OPF and FCT can be used in three configurations. In first configuration, OPF is applied in both stages. This will results in the best image quality but complexity of LBT increases. In second configuration OPF is applied in first stage only. In third configuration OPF is not used in any stage resulting in less complexity but quality of image decreases. We use OPF in both stages for the best quality. These three configurations are shown in Figure 2.

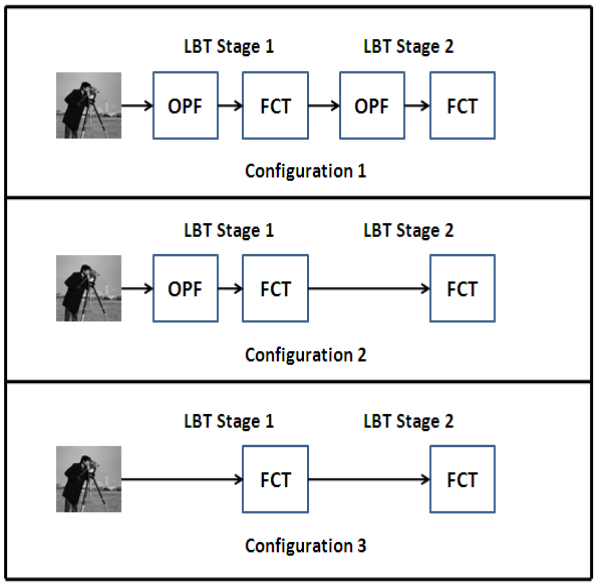

Figure 2. Different Configurations of OPF and FCT.

3. Proposed Design for LBT Implementation

Proposed design is based on high speed embedded processor with On-Chip Memory (OCM) which will be used as cache. Hardware of proposed design is shown in Figure 3. DDR SDRAM is used as off-chip memory for image storage. Off-chip Memory can also provide us a low cost memory solution for storing intermediate processed data

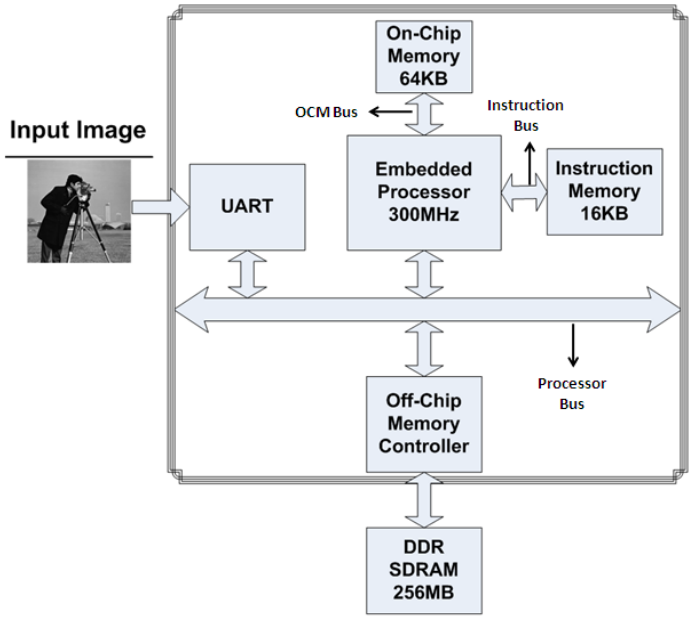

Figure 3. Proposed Design Hardware.

during LBT. Processor bus connects embedded processor with off-chip Memory controller and UART(or image acquisition device). OCM bus and instruction bus connects OCM and instruction memory with embedded processor respectively.

Our proposed design performs series of operations on input image. Acquired image is divided into tiles. Each tile is processed independently. In this way if a tile data is corrupted during transmission, it will not affect the neighboring tiles. Flow diagram shown in Figure 4 shows flow of data operations. Minimum size of input image is 128x128. That's means 64 macro blocks. Partitioning will be performed only for image size greater than 128x128. Large images can be process with this architecture at the cost of increase in computation complexity due to partitioning. Pipelined hardware design takes fix input image size. Larger images can't be processed and up gradation is difficult. In our design, image is first loaded in DDR SDRAM through UART or some image acquisition device. Raw image data can also be fetched from image sensor or imaging device for LBT processing. We can also process image data stored on external memory or network. During execution of LBT, processor first fetch tile data from DDR RAM into OCM and performs OPF_4pt and OPF_4x4 in horizontal and vertical direction on tile samples. After that operation, processed data is written-back to OCM to be used in FCT_4x4. This completes first stage of LBT.

At this stage, block has one DC coefficient and one macro block has 16 DC coefficients. For second stage, OPF_4pt and OPF_4x4 are applied on DC coefficients that are calculated in first stage of LBT and processed data is stored in OCM. After that FCT_4x4 is applied on OCM stored data which will results in DC, Low pass, and high pass coefficients. For each macro block we have 1 DC, 15 low pass and 240 high pass coefficients. After that coefficients are written-back to DDR SDRAM and data for another tile is fetched. This technique will reduce the overall memory required for OCM for LBT. Since LBT processing is sequential in nature, pipelined design requires a lot of registers and memory blocks for processing of LBT [10]. Managing these registers and handling data becomes very difficult as size of image increases. Up gradating of such design is also difficult. Proposed design offers easily up gradation. Processor based design can efficiently process sequential LBT. It requires less registers and memory elements. Handling register and data is easy in processor based design.

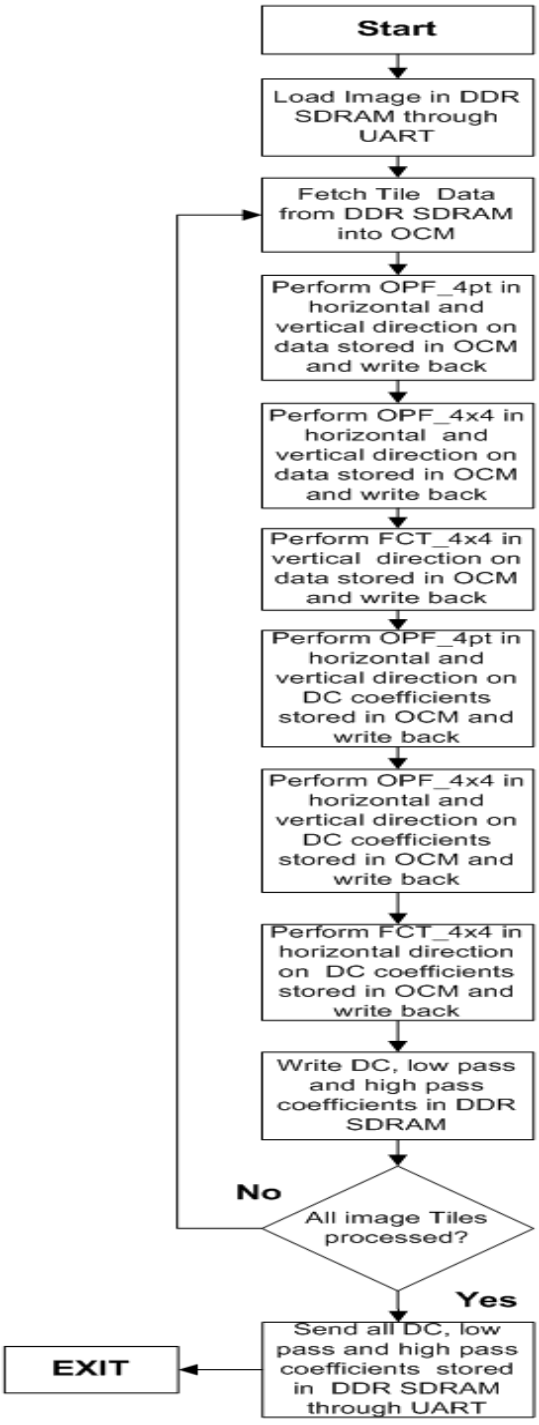

Figure 4. Flow Diagram of Proposed design

4. Design Testing Environment & Results

In order to test our design, we use Xilinx ML310 development board that contains Xilinx Virtex -II pro FPGA. It contains 30,816 logic gates, two PowerPC 405 cores, 2,448 kb Block RAM (BRAM). We use one of the PowerPC 405 core to test our design. Xilinx ML310 development board is shown in Figure 5.

Figure 5. Hardware setup for implementation of LBT.

We used BRAM of FPGA as OCM for data and instruction. DDR SDRAM is connected to processor through memory controller. Design is developed in Xilinx Embedded Development Kit (EDK 10.1) .In EDK, we select Xilinx Virtex-II Pro ML310 Evaluation Platform Revision D for creation of proposed hardware design. After that we select PowerPC processor with desired configurations. RS_232 and DDR SDRAM are configured as I/O interface in our design. We create new software (SW) Application Project in EDK and write the C code for LBT. After that, we implement the embedded processor hardware platform by compiling the system and application software and merge them into a bitstream for download to the board. Xilinx Parallel Cable 4 is used to download bitstream.

In order to verify the functionality of our design, we also implemented LBT in Labview software according to the specifications described in JPEG XR (ITU-T T.832 | ISO/IEC 29199-2) standard. Block Diagram of OPF_4x4 is shown in Figure 6.

We have used same image as an input for both FPGA based design and Labview implementation. We write application in Labview to compare each and every pixel value of both implementations and results from Labview matches with the results of our proposed design in FPGA. This verifies the functionality of LBT according to standard.

Results show that our design requires much less amount of memory. Memory required by our design to store instructions and data is 655,360 bits. Input image size does not affect the memory usage because whatever may be the size of input image, it will be divided into fix size tiles i.e.

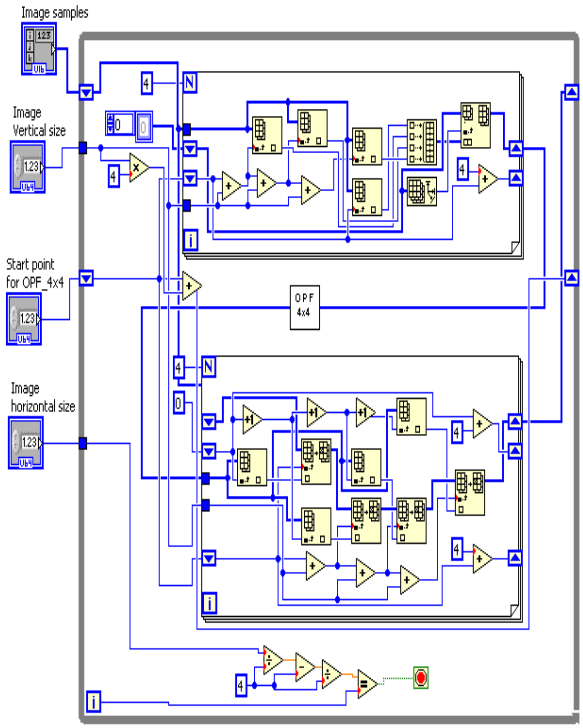

Figure 6. Block Diagram of OPF_4x4 processing in Labview.

128x128. Due to less memory and power consumption it is suitable for low cost battery operated devices. Up-gradation of design only requires change in software specifications of design coded in C language and load the updated instructions in the instruction memory. Test image of size 512x512 is used. Each image sample is unsigned 16 bit. Execution time to process test image is 26.8ms. Compression capability for test image is 37 frames per second. Figure 7 shows original and decompressed image which was compressed by proposed design. As we use lossless compression mode of JPEG XR so recovered image is exactly same as original image. Processor specifications used for implementation are listed in Table 1.

5. Conclusion

In this paper we have proposed a processor based design of LBT that is used in state-of-art image compression algorithm JPEG XR. Proposed design shows that it requires less memory for computation of LBT, easily up-gradable and consumes less power which makes this design suitable for low cost battery operated devices.

(a) **(b)**

Figure 7. Compression using LBT (a) original uncompressed image (b) decompressed image after compression using LBT.

Table 1. Processor Resources for LBT implementation.

Power PC 405	Design Values
Speed	300 MHz
Bus Speed	100 MHz
OCM (BRAM) for Instruction and Data	80 KB
Power Consumption	270 mW

References

[1] De Queiroz and R.L. Fleckenstein, Signal Processing Letters IEEE **7** (2000) 97.

[2] L. Liu, H. Meng, L. Zhang and Z. Wang, An ASIC Implementation of JPEG2000 Codec, Proceedings of IEEE Custom Integrated Circuits Conference (2005) p. 691-694.

[3] ITU-T, JPEG XR Image Coding System – Image Coding Specification, ITU-T Recommendation T.832 (2009).

[4] F. Dufaux, G. Sullivan and T. Ebrahimi, IEEE Signal Processing Magazine **26** (2009) 195.

[5] J. Z. Xu, F. Wu, J. Liang and W. Zhang, IEEE Transactions Image Processing **19** (2010) 85.

[6] A. Maalouf and M-C, Larabi, Low-Complexity Enhanced Lapped Transform for Image Coding in JPEG XR / HD Photo, Proceedings of 16th IEEE International Conference Image Processing, Cairo (2009) p.5-8 ISSN: 1522-4880.

[7] L.N. de Faria, L.M.G. Fonseca and M.H.M. Costa, http://www. hindawi.com/journals/jece/ 2012/471857/.

[8] M.A. Basit and G. Raja, The Nucleus **49**, No. 1 (2012) 11.

[9] C.-Y. Chien, S.-C. Huang, C.-H. Pan, C.-M. Fang and L.-G. Chen, Pipelined Arithmetic Encoder Design for Lossless JPEG XR Encoder, Proceedings of IEEE 13th International Symposium on Consumer Electronics (2009) pp. 144-147.

[10] S. H. Groder and K.W. Hsu, Design Methodolgy for HD Photo Compression Algorithm Targeting a FPGA, Proceedings of IEEE International SOC Conference (2008) pp. 105-108.

[11] Ching-Yen Chien, Sheng-Chieh Huang, Shih-Hsiang Lin, Yu-Chieh Huang, Yi-Cheng Chen, Lei-Chun Chou, Tzu-Der Chuang, Yu-Wei Chang, Chia-Ho Pan and Liang-Gee Chen, A 100 MHz 1920×1080 HD-Photo 20 frames/sec JPEG XR Encoder Design, 15th IEEE International Conference on Image Processing, (2008) p. 1384 – 1387.

[12] Chia-Ho Pan, Ching-Yen Chien, Wei-Min Chao, Sheng-Chieh Huang and Liang-Gee Chen, IEEE Transactions on Consumer Electronics **54** (2008) 963.