academicJournals

Vol. 8(9), pp. 352-358, 4 March, 2013
DOI 10.5897/SRE12.059
ISSN 1992-2248 © 2013 Academic Journals
http://www.academicjournals.org/SRE

Scientific Research and Essays

Full Length Research Paper

Field-Programmable Gate Array (*FPGA*) Implementation of Lapped Biorthogonal Transform for JPEG XR Compression

Muhammad Riaz ur Rehman and Gulistan Raja*

Department of Electrical Engineering, University of Engineering and Technology, Taxila, Pakistan.

Accepted 15 November, 2012

This paper describes the hardware implementation of Lapped Biorthogonal Transform (LBT) on Field-Programmable Gate Array (FPGA) for JPEG XR Image compression. The implementation is based on dividing image into 128×128 dimension tiles with each tile processed independently. Two main operations that is, overlap pre-filtering and forward core transform are applied on each tile. The proposed design has small memory requirement due to fix 128×128 tile size processing. The hardware design is tested on Xilinx Virtex-II Pro FPGA. The design utilizes 262,144 memory bits, 5824 number of slices and maximum speed is 107.308 MHz.

Keywords: Lapped Biorthogonal Transform (LBT), Field-Programmable Gate Array (FPGA), image compression, implementation.

INTRODUCTION

High quality imaging devices are used in applications like medical imaging, surveillance and space imaging. These applications demands high quality images which requires large storage. Therefore, compression is required to reduce the size of stored image. JPEG and JPEG 2000 are most commonly used image compression standards (Queiroz and Fleckenstein, 2000; Liu et al., 2005), however, both have some drawbacks. JPEG produces blocking artifacts at low bit rates and JPEG 2000 is computationally intensive technique. To address the limitation of currently used image compression standards, a new image compression technique JPEG eXtended Range (JPEG XR) is introduced (ITU-T, 2009; Dufaux et al., 2009). 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. JPEG XR uses Lapped Biorthogonal Transform (LBT) to convert image samples from spatial domain to frequency domain (Xu et al., 2010; Maalouf and Larabi, 2009). To use Lapped Biorthogonal transform in real time embedded environment, its hardware implementation is needed (Chien et al., 2009). Application specific hardware for LBT provides excellent performance. Due to sequential nature of LBT, it requires large amount of memory in case of pipeline implementation (Groder and Hsu, 2008; Chien et al., 2008; Pan et al., 2008). We propose a hardware design and implementation of LBT on FPGA that requires less amount of memory. Rest of the paper is organized as follows: In this paper an overview of LBT is described, and the proposed architecture of LBT elaborated. Thereafter the implementation results are discussed and

*Corresponding author. E-mail: gulistan.raja@uettaxila.edu.pk.

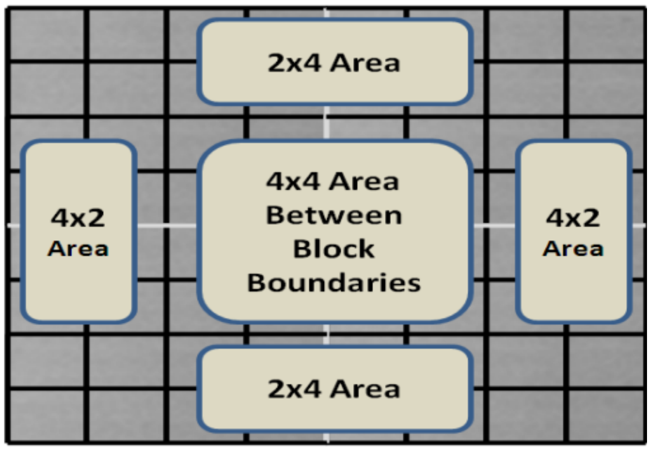

Figure 1. Areas between block boundaries.

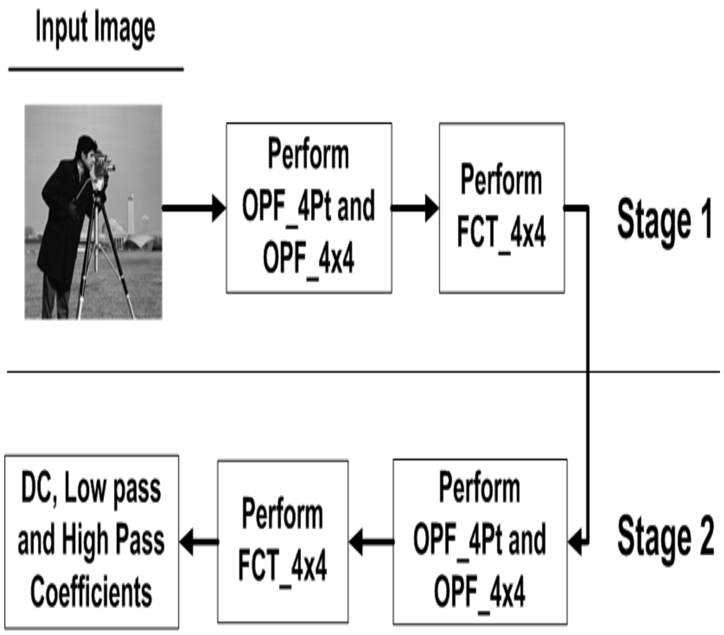
Input Image
Perform OPF_4Pt and OPF_4x4
Perform FCT_4x4
Stage 1
DC, Low pass and High Pass Coefficients
Perform FCT_4x4
Perform OPF_4Pt and OPF_4x4
Stage 2

Figure 2. Lapped Biorthogonal Transform stages (Rehman and Raja, 2012).

the paper concluded.

OVERVIEW OF LAPPED BIORTHOGONAL TRANSFORM (LBT)

The image is divided into tiles and tiles into macro blocks in JPEG-XR. Each macro block is collection of 16 blocks. A block is a collection of 16 image pixels. The image size should be multiple of 16, if not, we extend height or width of image by replicating the image sample values at boundaries. Areas of 4×4, 4×2 and 2×4 between image block boundaries are shown in Figure 1.

Lapped Biorthogonal Transform consists of two key operations: Overlap Pre Filtering (OPF) and Forward Core Transform (FCT) as explained below:

Overlap Pre Filtering (OPF)

Its main purpose is to exploit the correlation and redundancy across block boundaries. It also helps migrating blocking artifacts in compressed image especially at low bitrates. This operation is optional in LBT and can be switched off to reduce computational complexity at cost of reduce in compressed image quality.

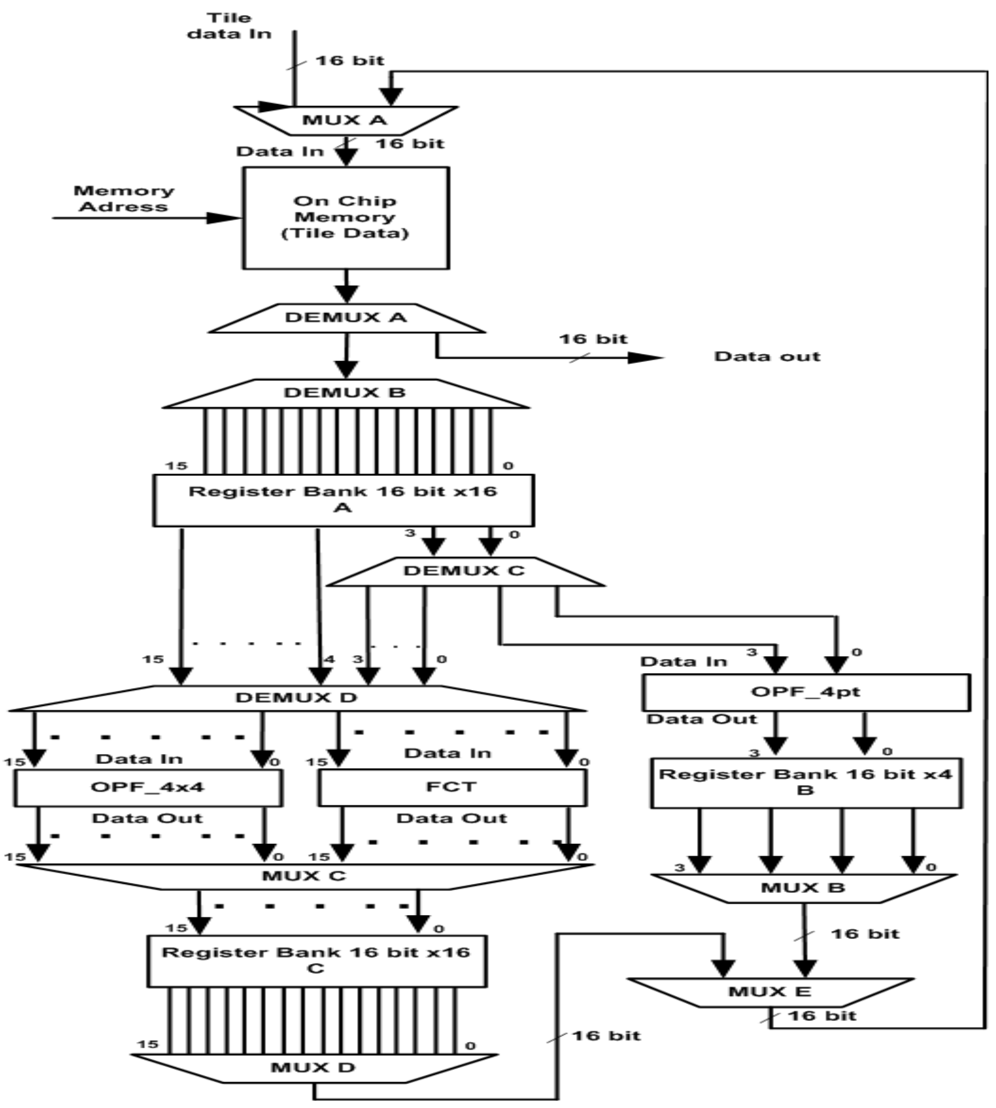

Figure 3. Hardware implementation architecture of LBT.

Forward Core Transform (FCT)

This operation is similar to Discrete Cosine Transform (DCT) which is used in original JPEG compression standard. It exploits the spatial correlation within blocks, and also has the same drawbacks of DCT. This operation is compulsory in LBT processing. The details of operations in LBT are shown in Figure 2 (Rehman and Raja, 2012).

1) In stage 1, Overlap pre filter (OPF_4pt) is applied to 2×4 and 4×2 areas between blocks boundaries. Additional filter (OPF_4×4) is also applied to 4×4 area between block boundaries.
2) A forward core transform (FCT_4×4) is applied to 4×4 blocks. This will complete stage 1 of LBT.
3) Each 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 to 2×4 and 4×2 areas between DC blocks boundaries. Additional filter (OPF_4×4) is also applied to 4×4 area between DC block boundaries.
5) Forward core transform (FCT_4×4) is applied to 4×4 DC blocks to complete stage 2 of LBT. This results in one DC coefficient, 15 low pass coefficients and 240 high pass coefficients per macro block.

PROPOSED ARCHITECTURE AND IMPLEMENTATION OF LBT

The proposed architecture of LBT is shown in Figure 3. The implementation consists of On-Chip Memory (OCM) for storing tile data, Register banks, two overlap pre

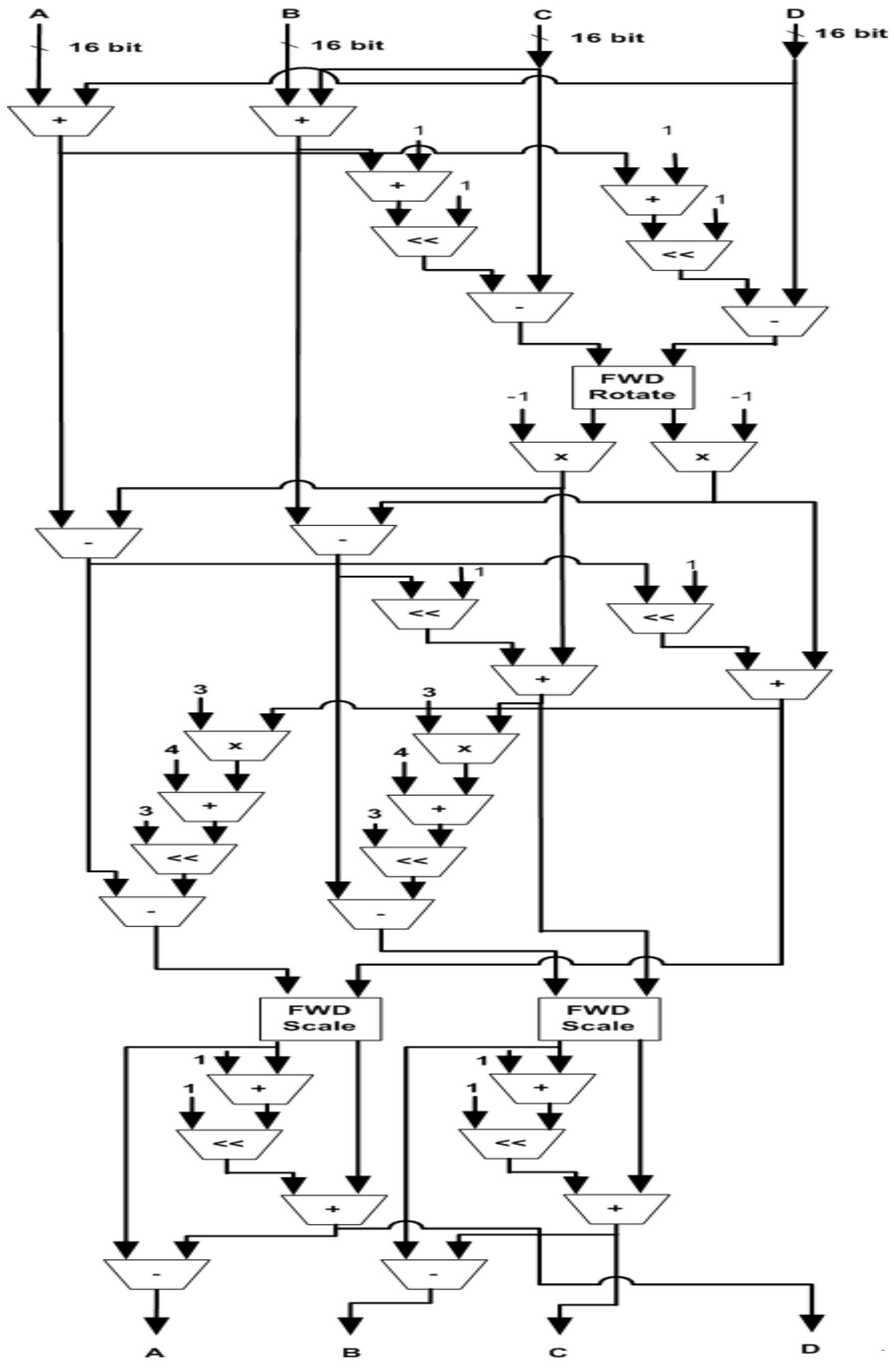

Figure 4. OPF 4pt processing block diagram.

filtering blocks and Forward core transform processing blocks. Minimum input image size is 128×128 and the image tile of size 128×128 is loaded in OCM. Block RAM (BRAM) of FPGA is used as OCM. OCM is also used to store intermediate process data.

Since memory used in design can only read or write at one time against a particular memory address, register banks are used that will buffer 16 image samples and passes these image samples to processing blocks in parallel. For processing of LBT, image tile data is loaded into OCM by selecting MUX A. Then overlap pre-filtering is performed on image tile boundary area of size 4×2 and 2×4. Four image samples are buffered in Register Bank A by selecting DEMUX A and DEMUX B. These image samples are processed by OPF 4pt. OPF 4pt block diagram is shown in Figure 4.

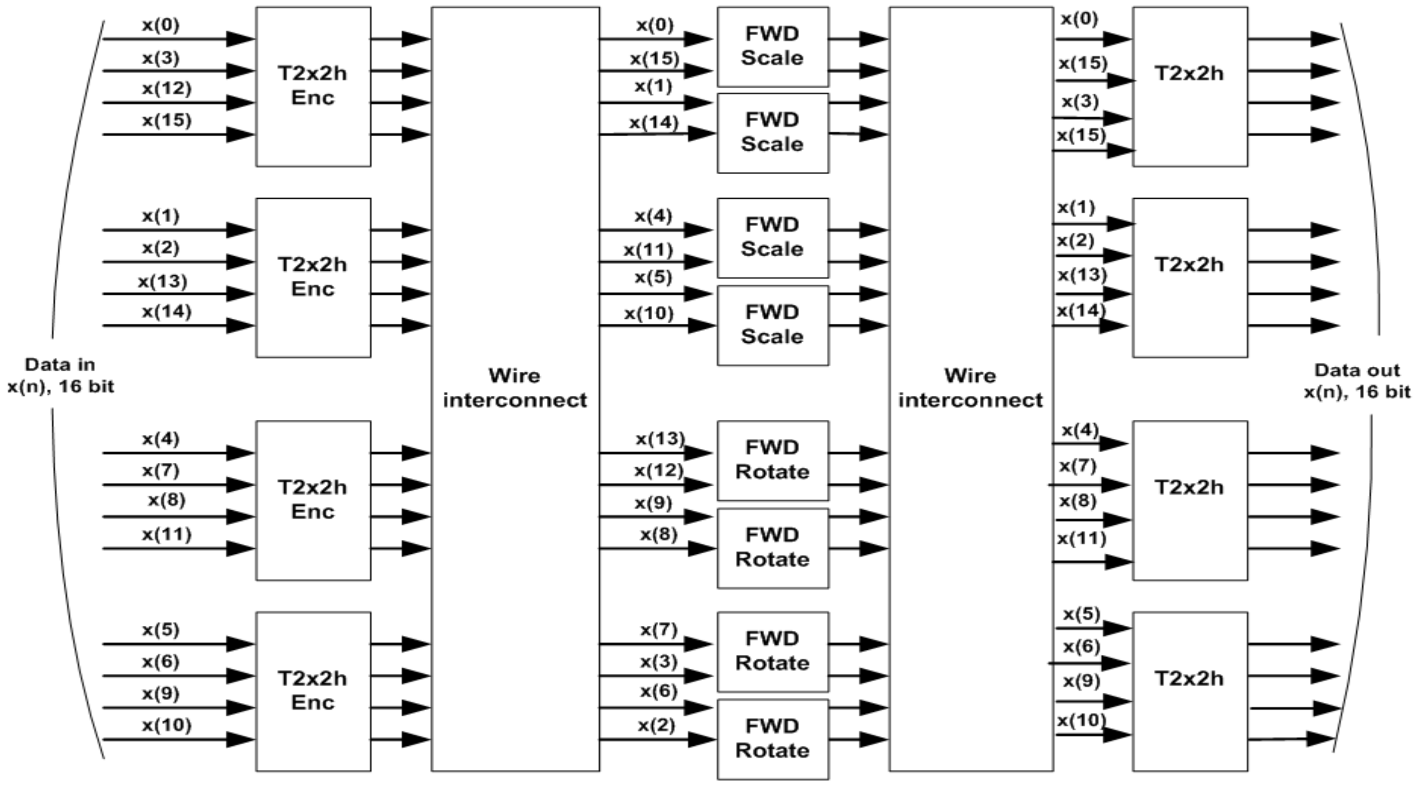

Figure 5. OPF 4x4 processing block diagram.

OPF 4pt performs a series of mathematical operation such as addition, subtraction, multiplication and logical shifting. OPF 4pt is applied on tile boundary areas of size 4×2, 2×4 along vertical and horizontal dimensions of tile. After OPF 4pt, processed image samples are loaded in Register Bank B and written back to OCM by selecting MUX B and MUX E. OPF 4×4 operation is performed on block boundary areas of size 4×4. For this operation, 16 image samples are loaded in Register Bank A from OCM. These 16 image samples are processed by OPF 4×4 block. Operational block diagram of OPF 4×4 is shown in Figure 5.

After OPF 4×4, processed image samples are load in Register Bank C and written back to OCM by selecting MUX C, MUX D, MUX E. After operation of OPF 4×4, FCT is applied on image blocks. This operation results in one DC coefficient per image block. This completes the stage 1 of LBT. Similar to OPF 4×4, FCT processed image samples are written back to OCM. Block diagram of FCT processing is shown in Figure 6. For stage 2 of LBT, DC coefficients are arranged in blocks of size 4×4. Same operations of stage 1 of LBT are applied on these DC blocks. OPF 4pt is performed on DC block boundaries of 4×2, 2×4 areas. After that operation OPF 4×4 is applied on DC block boundary area of size 4×4. Thereafter FCT is applied on 4×4 DC block and stage 2 of LBT is completed. This result in 1 DC coefficient, 15 low pass coefficients and 240 high pass coefficients per macro block.

SIMULATION RESULTS

In order to implement our design, Xilinx ML310 development board is used which has Xilinx Virtex -II pro FPGA. It contains 30,816 logic gates, 2,448 kb Block RAM (BRAM). Functionality of design is verified according to the specifications described in JPEG XR (ITU-T T.832 | ISO/IEC 29199-2) standard (ITU-T, 2009). Functionality of each module of LBT is tested by writing test benches in ModelSim 6.5. The processing of image on tile data basis in proposed design reduces the memory requirements for operation of LBT. As a result, memory required by our design to store tile of size 128×128 is 262,144 bits. Input image size does not affect the memory usage because image size greater than 128×128 will be divided into fix size tiles that is, 128×128. Due to less memory requirement, it is suitable for low cost embedded applications. The implementation resource utilization is shown in Table 1. The implementation utilizes 5824 number of slices, 6381 number of slice flip flops and 9508 number of 4-input LUTs to map proposed architecture on FPGA. The number of BRAMs used is 16. Design uses 6695 16 bit

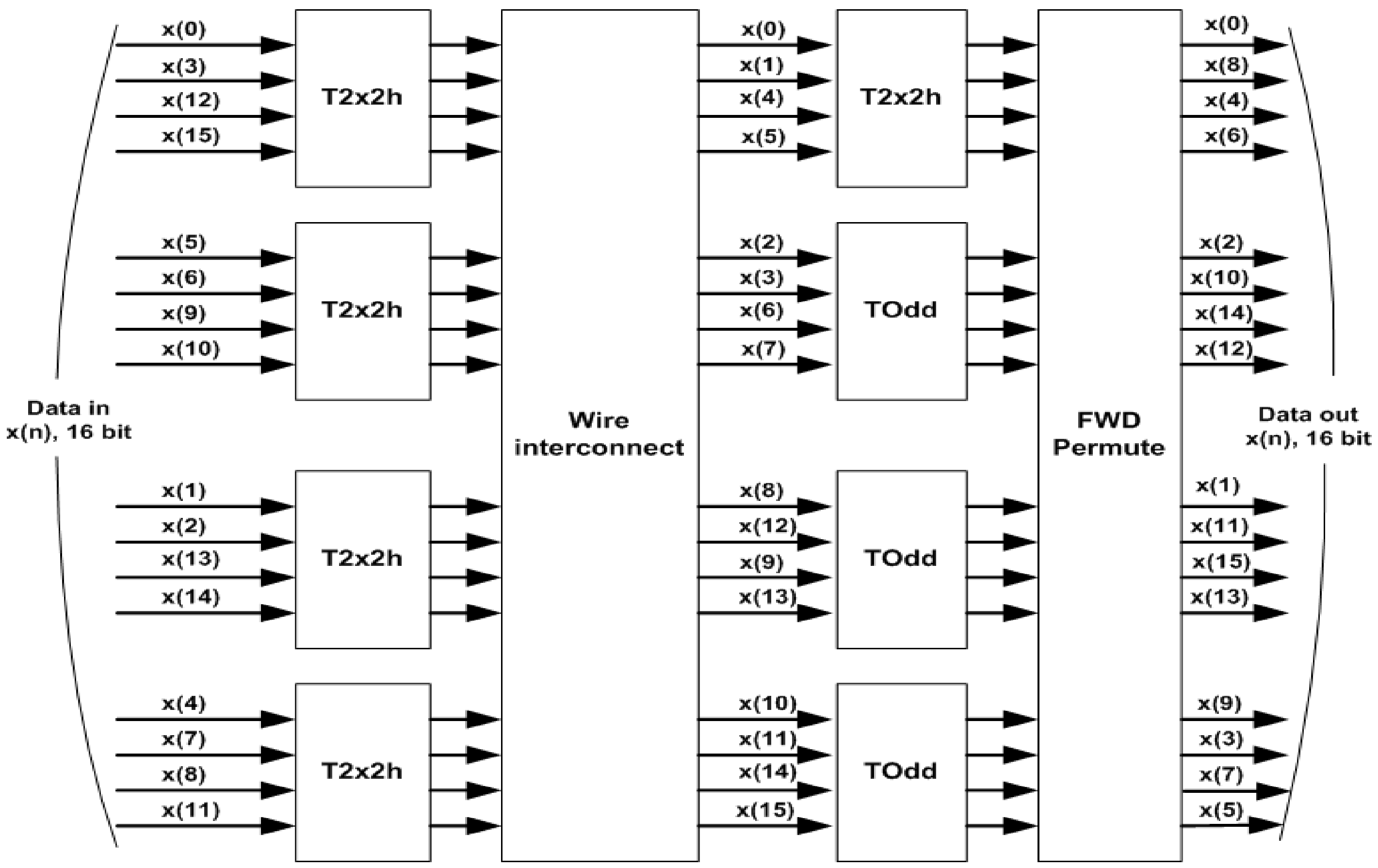

Figure 6. FCT processing block diagram.

Table 1. FPGA implementation results.

Technology	Xilinx Virtex-II Pro
Number of slices	5824
Number of slice flip flops	6381
Number of 4 input LUTs	9508
OCM	262,144 bits
Number of BRAMs	16
Number of MULT18X18s	4
16 bit Adders/Subtractors	66, 95
Maximum frequency	107.308 MHz

adders/ subtractors. OCM required for LBT operation is 262144 bits. The design can operate on maximum frequency of 107.308 MHz.

Conclusion

In this paper we have proposed architecture for processing of LBT in hardware for the state-of-art image compression algorithm JPEG XR. The proposed implementation is based on On-Chip Memory that is used for storing and processing image tiles. Input image is divided into tiles of fixed size that is, 128×128. The independent processing of each tile of image can isolate errors in tiles that may occur during transmission of image over channel. Proposed architecture is tested on Xilinx Virtex -II pro FPGA. The design can operate at maximum frequency of 107.308 MHz. Architecture of this design is simple and required optimum resources of FPGA. Proposed design also reduces the amount of memory required for processing of LBT. The design can be used in low cost embedded system.

REFERENCES

Chien CY, Huang SC, Pan CH, Fang CM, Chen LG (2009). Pipelined arithmetic encoder design for lossless JPEG XR encoder. In Proc. IEEE 13th Int. Symp. Consum. Electron. pp. 144-147.

Chien CY, Huang SC, Lin SH, Huang YC, Chen YC, Chou LC, Chuang TD, Chang YW, Pan CH, Chen LG (2008). A 100 MHz 1920×1080 HD-photo 20 frames/sec JPEG XR encoder design. In Proc. 15th IEEE Int. Conf. Image Process. pp. 1384-1387.

Dufaux F, Sullivan G, Ebrahimi T (2009). The JPEG XR image coding standard. IEEE Sig. Process. Mag. 26(6):195-199.

Groder SH, Hsu KW (2008). Design methodolgy for HD photo compression algorithm targeting a FPGA. In Proc. IEEE Int. Conf. SOC. pp. 105-108.
ITU-T (2009). JPEG-XR image coding system – Image coding specification. ITU-T Recommendation T.832.
Liu L, Meng H, Zhang L, Wang Z (2005). An ASIC implementation of JPEG2000 codec. In Proc. IEEE Custom Integrated Circuits Conf. pp. 691-694.
Maalouf A, Larabi MC (2009). Low-complexity enhanced lapped transform for image coding in JPEG XR / HD Photo. In Proc. IEEE Intl. Conf. Image Process. pp. 5-8.
Pan CH, Chien CY, Chao WM, Huang SC, Chen LG (2008). Architecture design of full HD JPEG XR encoder for digital photography applications. IEEE Trans. Consum. Electron. 54(3):963-971.
Queiroz D, Fleckenstein RL (2000). Very fast JPEG compression using hierarchical vector quantization. IEEE Sig. Process. Lett. 7(5):97-99.
Rehman MR, Raja G (2012). A processor based implementation of lapped biorthogonal transform for JPEG XR Compression on FPGA. Nucleus. 49(3):181-186.
Xu JZ, Wu F, Liang J, Zhang W (2010). Directional lapped transforms for Image coding. IEEE Trans. Image Process. 19(1):85-97.