International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

FPGA Implementation of 2D-DCT for Image Compression

Pradnya P. Parate[1], **Nilesh A. Mohota**[2]

[1]Resercher, J.D college of engineering & Management, Nagpur, Maharashtra, India

[2]Assistant Professor, J.D college of engineering & Management, Nagpur, Maharashtra, India

Abstract: ***Image compression is the application of data compression on digital images. Image compression can be lossy or lossless. Compressing an image is significantly different than compressing raw binary data. Of course, general purpose compression programs can be used to compress images, but the result is less than optimal. DCT has been widely used in signal processing of image. The one-dimensional DCT is useful in processing one-dimensional signals such as speech waveforms. For analysis of two dimensional (2D) signals such as images, we need a 2D version of the DCT data, especially in coding for compression, for its near-optimal performance.***

Keywords: DCT, FPGA, Image Compression, JPEG

1. Introduction

A. Image Compression.

JPEG is the most common form of image compression. JPEG stands for Joint Photographic Expert Group[1], this standard as developed in late 80's.

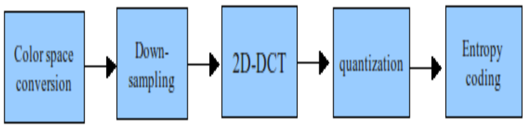

Figure 1: compression steps of color images

Color Image compression consist of five steps as shown in the fig1.Steps consist of color space conversion, down sampling, 2D-DCT, quantization, entropy coding.

2. Discrete Cosine Transform (DCT)

The DCT basis functions are for size 8 x 8 .The mapping between the mathematical values and the colors (gray levels) is the same as in the DFT case[1]. Each basis function occupies a small square; the squares are then arranged into as 8 x 8 matrix.

Definition (2-D DCT)
Assume that the data array has finite rectangular support on $[0, N_1, -1] x [0, N_2 - 1]$, then the 2-D DCT is given as

$$X_{C(K1K2)} = \sum_{k1=0}^{N1-1}\sum_{k2=0}^{N2-1} 4X(n1n2)\cos\frac{\Pi K1}{2N1}(2n1+1).\cos\frac{\Pi K2}{2N2}(2n2+1)$$

The inverse of DCT is given as

$$X_{(K1K2)} = \frac{1}{N1N2}\sum_{k1=0}^{N1-1}\sum_{k2=0}^{N2-1} w(K1)w(K2)Xc(K1K2)\cos\frac{\Pi K1}{2N1}(2n1+1).\cos\frac{\Pi K2}{2N2}(2n2+1)$$

Computation of 2D-DCT is shown in Fig 2.

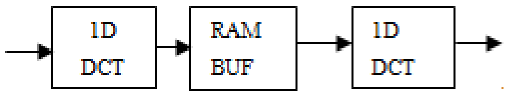

Figure 2: 2D-DCT

In case of 2D-DCT, first we perform the operation of 1D-DCT and stored that output to RAM after that we again perform the operation of 1D-DCT.

3. 1D-Discrete Cosine Transform(DCT)

Operation of 1D-DCT can be perform in several ways. In this paper we are using Butterfly algorithm for the computation of DCT [2]. DCT coefficient which we are getting after computation is not the scaled computation So, for obtaining scaled computation

$$y' = Cx \qquad (1)$$

Where x is a 8 point vector. C is DCT matrix and y' is vector of scaled DCT coefficient. To get real DCT

$$y = s.* y' \qquad (2)$$

Constant s is a vector of post scaling factor. Element by element multiplication is shown by "*" operator. By using above formulas of 1D-DCT we can find the DCT.

Step 1:	$a_0 = x_0 + x_7$ $a_1 = x_1 + x_6$ $a_2 = x_3 - x_4$ $a_3 = x_1 - x_6$ $a_4 = x_2 + x_5$ $a_5 = x_3 + x_4$ $a_6 = x_2 - x_5$ $a_7 = x_0 - x_7$
Step 2:	$b_0 = a_0 + a_5$ $b_1 = a_1 - a_4$ $b_2 = a_2 + a_6$ $b_3 = a_1 + a_4$ $b_4 = a_0 - a_5$ $b_5 = a_3 + a_7$ $b_6 = a_3 + a_6$ $b_7 = a_7$
Step 3:	$d_0 = b_0 + b_3$; $d_1 = b_0 - b_3$; $d_2 = b_2$; $d_3 = b_1 + b_4$;

	$d_4 = b_2 - b_5$; $d_5 = b_4$; $d_6 = b_5$; $d_7 = b_6$; $d_8 = b_7$;
Step 4:	$e_0 = d_0$; $e_1 = d_1$; $e_2 = m_3 * d_2$; $e_3 = m_1 * d_7$; $e_4 = m_4 * d_6$; $e_5 = d_5$; $e_6 = m_1 * d_3$; $e_7 = m_2 * d_4$; $e_8 = d_8$;
Step 5:	f0 = e0; f1 = e1; f2 = e5 + e6; f3 = e5 - e6; f4 = e3 + e8; f5 = e8 - e3; f6 = e2 + e7; f7 = e4 + e7;
Step 6:	$y'_0 = f_0$; $y'_1 = f_4 + f_7$; $y'_2 = f_2$; $y'_3 = f_5 - f_6$; $y'_4 = f_1$; $y'_5 = f_5 + f_6$; $y'_6 = f_3$; $y'_7 = f_4 - f_7$;

4. Two Dimensional Discrete Cosine Transform (2D-DCT)

2D-DCT is made from two 1D-DCT using DCT matrix

$$Y' = [C]\,[X]\,[C]^T \qquad (3)$$

Where,
X= 8x8 input matrix
Y'=Scaled 2D-DCT 8x8 output matrix
C=DCT matrix

To get 2D-DCT real value, Y' must be element by element multiplied by post scaler as shown in the below equation.

$$Y = [S].*[Y'] \qquad (4)$$

With S= post scaled matrix and
Y= real DCT coefficients.
Post scaled matrix S is

$$S = s\,s^T \qquad (5)$$

a) Quantization

Quantization output is made by divide each DCT coefficient by quantization value. It is done by doing element by element matrix multiplication between DCT output and quantization matrix.

$$Yq = [Q].*[Y] \qquad (6)$$

$$Yq = Q.*[S].*[Y] \qquad (7)$$

The quantization matrix is only use for the luminance. The matrix is modified and post scaled by the matrix S. the scaled quantization matrix is shown below

$$Q=\begin{bmatrix} 32 & 34 & 39 & 27 & 21 & 16 & 19 & 30 \\ 31 & 22 & 20 & 17 & 14 & 8 & 11 & 24 \\ 28 & 22 & 19 & 14 & 10 & 9 & 10 & 25 \\ 31 & 18 & 15 & 13 & 9 & 6 & 10 & 23 \\ 28 & 17 & 11 & 8 & 8 & 61 & 9 & 24 \\ 27 & 13 & 9 & 9 & 8 & 8 & 11 & 26 \\ 19 & 11 & 9 & 9 & 9 & 19 & 15 & 34 \\ 26 & 15 & 15 & 16 & 17 & 24 & 33 & 68 \end{bmatrix}$$

FPGA Implementation

b) Entire System

Block diagram of entire system to be implemented in FPGA is shown in figure below. Input data is inserted into the system every 8 bit sequentially. Actually, many DCT designs insert the input to the DCT in parallel. For example is 8 x 8 bit [1],[3],. This is ideal for DCT computing because it only consumes a clock cycle to insert data to 1D-DCT unit. With sequential manner, it takes 8 clock cycles to insert a set of data (8 points) to the DCT unit. The sequential architecture is chosen to save I/O port in FPGA chip. Some 2D-DCT intellectual property designs from *Xilinx* also use 8-bit input

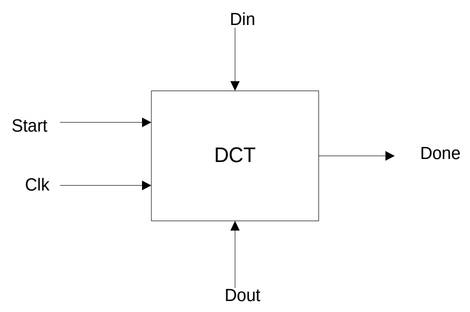

Figure: Block representation of entire system

c) 1D- DCT pipeline process

Algorithm defined in table 1 from [1] is used to compute 1D-DCT. The algorithm itself has 6 steps. Since the DCT input/output has 8 points and data has to be entered and released in sequential manner, it takes 8 clock cycles for each input and output process. Totally, 8 points 1D-DCT computation needs 22 clock cycles. Design for data input and output in this paper is inspired by design from [1]. The input and output process visualization is shown in figure. The difference from system [2] is that it computes every single operation in a clock cycle, so every step needs 8 – 9 clock cycles. In this paper, system computes every step in a clock cycle, so DCT computation can be done faster.

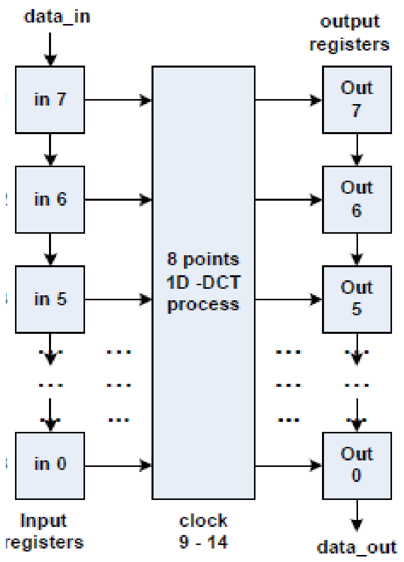

Figure: Data input/output process visualization of 8 points 1D-DCT

1D-DCT computation is done in pipeline process. The pipeline process is made in three stages. Thus, one pipeline stage is done in 8 clock cycles.

d) Transpose Buffer

Transpose buffer is static RAM, designed with two set of data and address bus. It has input and output data address buses. The module block symbol is shown in fig.

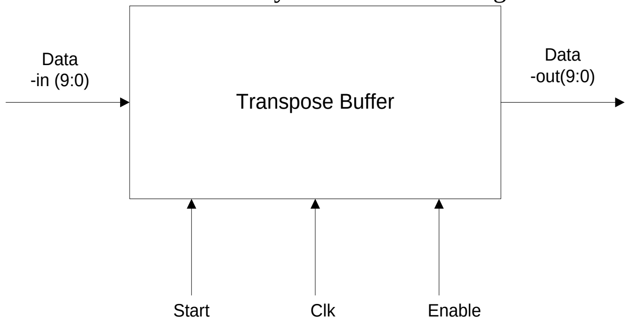

Figure: Transpose buffer primitive symbol

e) Implementation Result

The 2-D DCT architecture was described in VHDL.
This VHDL was synthesized into an Xilinx Vertex 5 family FPGA. There is a requirement to use this architecture. The FPGA must have at least 11 multipliers, because the system utilizes only internal multiplier.

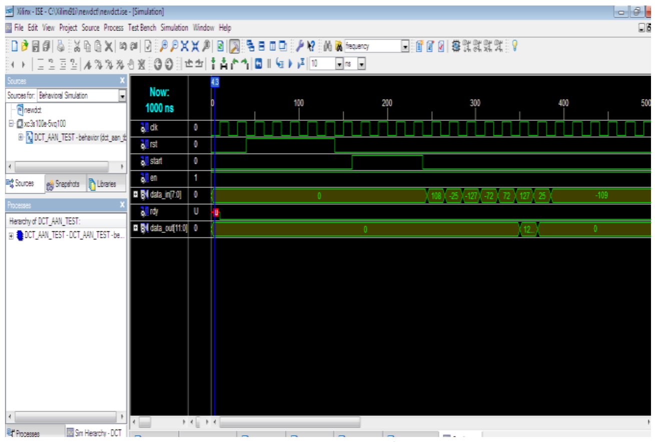

Figure: Simulation of 2D-DCT

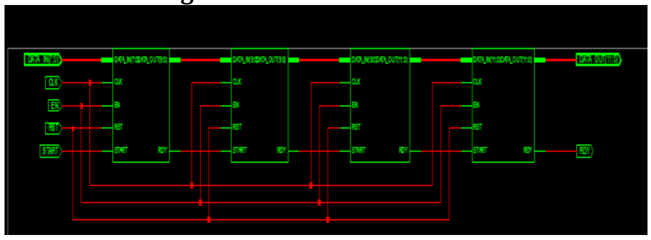

Figure: RTL view of 2D-DCT

Table: Device Utilization Using Xilinx VERTEX 5

Logic Unit	Used	Available	Utilization
Number of Slices	718	960	74%
Number of Slices FFs	578	1920	30%
Number of 4 input LUTs	1152	1920	60%
Number of Bonded IOBs	25	66	37%
Number of Multipliers 18X18	4	4	100%

Table – Device utilization comparison between this work and 2D-DCT designed in [1]

Logic Unit	Used	Present in [1]
Number of Slices	718	1145
Number of Slices FFs	578	1696
Number of 4 input LUTs	1152	1750
Number of Bonded IOBs	25	21
Number of Multipliers 18X18	4	11

5. Conclusion

The JPEG image compression is designed in VHDL. We proposed a new direct 2D DCT architecture for a fast efficient scaled transform. The preprocessors predict the zero quantized coefficients and can reduce up to 50% of the total operations. Discrete Cosine Transform 2-dimensional, followed by Zig Zag scan, Quantizer has been performed.

References

[1] Enas Dhuhri Kusuma, Thomas Sri Widodo "FPGA Implementation of Pipelined 2D-DCT and Quantization Architecture for JPEG Image Compression" 978-1-4244-6716-7/10/$26.00 ©2010 IEEE
[2] Sanjeevannanavar, S. Nagamani, A.N. "Efficient design and FPGA implementation of JPEG encoder using verilog HDL"
[3] Zulkalnain Mohd-Yusof, Ish& Suleiman, Zulfakar Aspar "Implementation Of Two Dimensional Forward DCT And Inverse DCT Using FPGA". **0**-7803-6355-8/00/$10.00©2000IEEE

[4] Hatim Anas, Said Belkouch, M. El Aakif "FPGA Implementation Of A Pipelined 2D-DCT And Simplified Quantization For Real-Time Applications". 978-1-61284-732-0/11/$26.00 ©2010 IEEE

[5] Prashant Chaturvedi, Prof. Tarun Verma and Prof. M. Zahid Alam," A Novel VLSI based Architecture for Computation of 2D-DCT Image Compression",*International Journal of Electrical, Electronics* **ISSN No.** (Online) : 2277-2626 *and Computer Engineering* **1**(2): 65-70(2012)

[6] K. Deepthi, R. Ramprakash "Design and Implementation of JPEG Image Compression and Decompression" International Journal of Innovations in Engineering and Technology (IJIET) Vol. 2 Issue 1 February 2013 ISSN: 2319 – 1058.