International Journal of Electrical, Electronics and Computer Engineering **1**(2): 65-70(2012)

ISSN No. (Online) : 2277-2626

A Novel VLSI based Architecture for Computation of 2D-DCT Image Compression

Prashant Chaturvedi, Prof. Tarun Verma and Prof. M. Zahid Alam

Department of Electronics and Communication Engineering,
LNCT, Bhopal, (MP)
(Received 30 October, 2012, Accepted 16, November, 2012)

ABSTRACT: Data image compression is the reduction of redundancy in data representation in order to achieve reduction in storage cost. In This paper The Implementation and Optimization of FPGA based 2D-DCT (discrete Cosine Transform) with Quantization and Zigzag with parallel pipelining using VHDL. The two 1D-DCT Separability property and calculation by using a Transpose buffer .The coding is simulated using Xilinx 9.2i ISE Synthesized using Spartan-3E XC3S500. The 2D-DCT Architecture uses 615 slices, 74 I/O pins, and 6 multiplier and operating frequency 124MHz and pipeline latency 160 clock cycles.

Keyword: JPEG, discrete cosine transform (DCT), quantization, zigzag, FPGA, model Sim

I. INTRODUCTION

Computer graphics applications, particularly those generating digital photographs and other complex colour images, can generate very large file sizes. Issues of storage space, and the requirement to rapidly transmit image data across networks and over the Internet, have therefore led to the development of a range of image compression techniques, to reduce the physical size of files.

The most common form of image compression is known as JPEG. The joint photographic Expert Group in the late 1980's developed this standard. The committee's first published standard was named as Baseline JPEG. In 1996's, JPEG committee began to investigate possibilities for new image compression standard that can serve current & future applications [1].

The JPEG image compression standard [3, 4] was Developed by Joint Photographic Expert Group [5]. The JPEG compression principle is the use of controllable losses to reach high compression rates. In this context, the information is transformed to the frequency domain through DCT. Since neighbor pixels in an image have high likelihood of showing small variations in color, the DCT output will group the higher amplitudes in the lower spatial frequencies [6]. Then, the higher spatial frequencies can be discarded, generating a high compression rate and a small perceptible loss in the image quality. The JPEG compression is recommended for photographic images, since drawing images are richer in high frequency areas that are distorted with the application of the JPEG compression [2, 7].

Discrete Cosine Transformation (DCT) is the most widely used transformation algorithm. DCT, first proposed by Ahmed [9] *et al*, 1974, has got more importance in recent years, especially in the fields of Image Compression and Video Compression. This chapter focuses on efficient hardware implementation of DCT by decreasing the number of computations, enhancing the accuracy of reconstruction of the original data, and decreasing chip area. As a result of which the power consumption also decreases. DCT also improves speed, as compared to other standard Image compression algorithms like JPEG. Discrete Cosine Transform (DCT) with applications in audio filtering to video compression is a mathematical tool which converts the information from the time or space domains to the frequency domain, such that other tools and transmission media can be run or used more efficiently to reach application goals. The JPEG compression principle is the use of controllable losses to reach high compression rates. In this context, the information is transformed to the frequency domain through DCT. Since neighbor pixels in an image have high likelihood of showing small variations in color, the DCT output will group the higher amplitudes in the lower spatial frequencies [8].

Compression also can come from the way color is represented within pixel data values. Monochrome images simply use one number to indicate the luminance of the sample. [9] In order to represent color within an image several values areused. There are several formats used to represent color, the two most common being RGB and Luminance/Chrominance, which both employ three value pixel representations.

The RGB format represents a colored image in three separate parts. Each of the three samples represents the same image, just the relative proportions of red,
Green, and blue for each given pixel [9]. These are primary colors so any color or shade can be produced by a combination of red, green, and blue intensities. This is known as additive coloring.

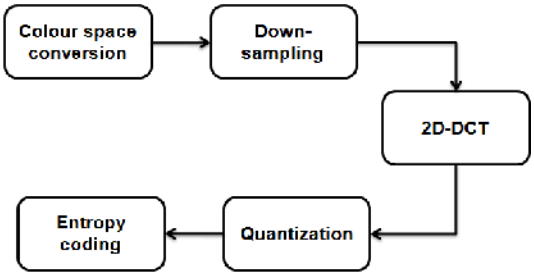

Fig. 1. JPEG Compression Steps for Colour Images.

This paper focuses only in a pipelined hardware
Implementation of the main part of the JPEG standard: the Two Dimensional Discrete Cosine Transform [11-13]. This is the most critical module to be designed in JPEG compressor hardware because of its high algorithm complexity [10].

This work proposes initially a FPGA implementation for flexibility, time-to-finish and development purposes. The results from RTL level VHDL design can be reused for an ASIC implementation in the future and the designed VHDL for 2-D DCT calculation can be used as a core in other designs like video compression. [10]

II. BACKGROUND WORK

JPEG Encoder core is intended to encode raw bitmap images into JPEG compliant coded bit stream [12]. JPEG baseline encoding method is used. The Architecture is given in figure 1; General system architecture consists of encoding chain started by Host Programming interface. Host Data interface shall continuously write BUF_FIFO until FIFO almost full signal is received. Then, it should stop and wait for signal FIFO almost full to deassert. When this is the case it should continue writing and so on.

Encoding is governed by Controller and is a pipelined process where each pipeline stage process 16 x 8 block of samples at a time (16x8 block is so called “data unit”. Following steps are performed: Line Buffering, Chroma subsampling, RGB to YCbCr conversion [14], Discrete Cosine Transform 2-dimensional [12-13], followed by Zig-Zag scan and Quantization.

The 2-D DCT calculation has a high degree of computational complexity. Since many authors have proposed simplifications to this calculation, as, [15-17] and others, this complexity can be minimized according to the application needs. Specifically for image compression applications there are many algorithms to compute the 2-D DCT coefficients and the algorithm chosen in this paper was proposed in [18] and modified in [19].

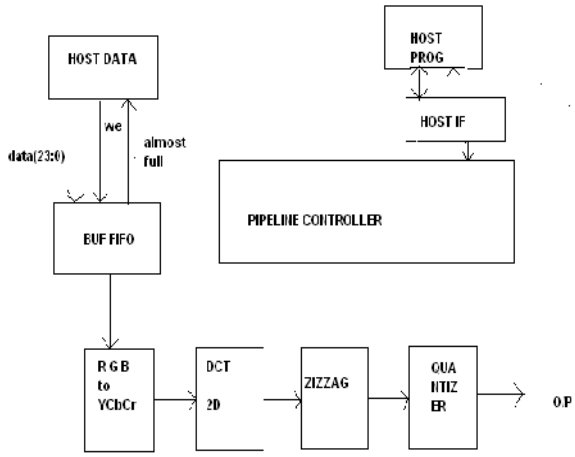

Fig. 2.

III. FPGA IMPLEMENTATION OF JPEG COMPRESSION

A. BUFFIFO

Host Data interface writes input image line by line to BUF_FIFO. This FIFO is intended to minimize latency between raw image loading and encoding start. It performs raster to block conversion.

FDCT is intended to perform functions: RGB to YCbCr conversion, chroma subsampling, input level shift and DCT (discrete cosine transform). Following equation is implemented by means of multipliers and adders to perform conversion:

$$Y = (0.299*R)+(0.587*G)+(0.114*B) \quad ...(1)$$
$$Cb = (-0.1687*R)-(0.3313*G)+(0.5*B)+128 \quad ...(2)$$
$$Cr = (0.5*R)-(0.4187*G)-(0.0813*B)+128 \quad ...(3)$$

B. 2D-DCT

The 2-D DCT (8 x 8 DCT) is implemented by the row-column decomposition technique .We first compute

the 1-D DCT (8 x 1 DCT) of each column of the input data matrix X to yield XtC. after appropriate rounding or truncation, the transpose of the resulting matrix, CtX, is stored in an transpose buffer. We then compute another 1-D DCT (8 x 1 DCT) of each row of CtX to yield the desired 2-D DCT as defined in equation (1). A block diagram of the design is shown in Fig 3.

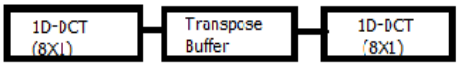

Fig. 3. 2D-DCT Architecture.

2D Discrete cosine transform core combined with level shift. Works on block of 64 samples. Take 8 bit input and produces 12 bit output. First, uncoded image is level shifted from unsigned integers with range [0, 2^P - 1] to signed integers with range [2^ (P-1), 2^ (P-1)-1]. X^p means here x to power of p.

Then 2D DCT is performed using following equation:

$$C_{k,l} = \sqrt{\frac{2}{N}}\, COS\left[\frac{(2k-1)(l-1)\pi}{2N}\right] .....(4)$$

$$For \quad K = 1, 2, 3.....N, \quad l = 2, 3.....N$$

$$and \quad C_{k,l} = N\frac{-1}{2} \quad For \quad l = 1$$

MDCT takes data row-wise but outputs column-wise. To get row-wise order it is necessary to transpose output DCT matrix. FIFO1 has space for 5 8x8 block DCT results. Its size = 5 x 64 x 12 bit. FIFO read controller has internal counter fifo_cnt (6:0) to count number of words read per one start_pb. When start_pb asserts. Dual port RAM 2 x 64 x 12 bit. Double Buffer necessary for pipeline operation. Buf_sel signal is used as MSB in read address of DBUF. ~buf_sel (inverted) is used as MSB of write address to DBUF [10].

C. ZIGZAG

Zig-Zag block is responsible to perform so called zig-zag scan. It is simply reorder of samples positions in one 8 x 8 block.

It means for example that first sample (position 0) is mapped to the same output position 0. Sample 2 is mapped to output position 5 [10]. The architecture of zigzag is shown in Fig. 4.

ZigZag Core is performing reordering of input samples according to zig-zag sequence. For that purpose REORDER ROM is used and ROM should be filled with following values written from address 0 to 63.

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	….	….	….
0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

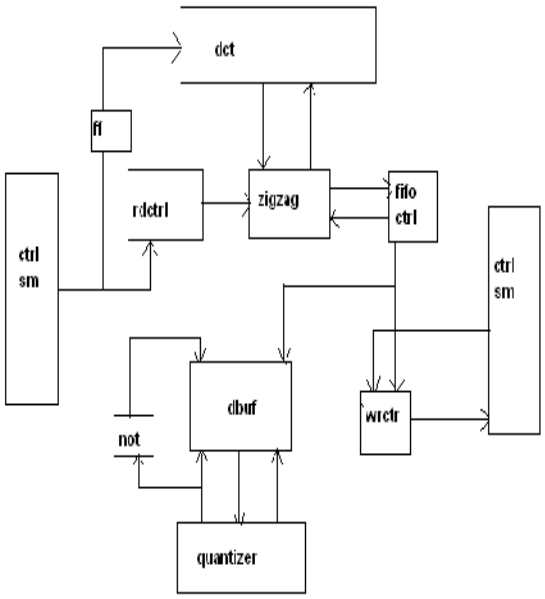

Fig. 4.

D. QUANTIZER

Quantizer performs division of input samples by defined quantization values. Works sample wise. Host can fill in 64 different quantization coefficients to internal RAM (64 x 8 bits).

Following equation is implemented:

FOUT (u, v) = round (FINPUT (u, v)/Q (u, v))... (5)

u,v –rectangular coordinates

FINPUT – input sample to quantizer

FOUTPUT – output sample from quantizer.

Rounding to nearest integer

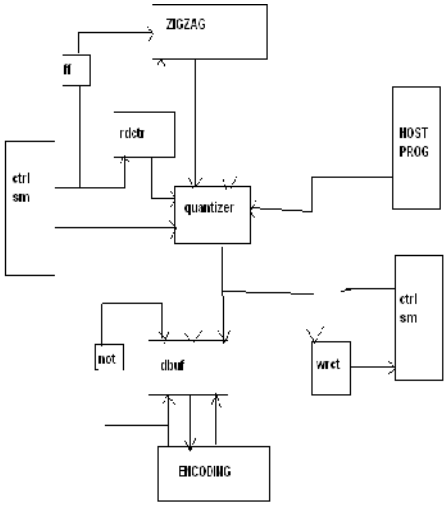

Fig. 5.

IV. SIMULATION RESULTS

The 2-D DCT, Quantization and Zigzag architecture was described in VHDL. This VHDL was synthesized into a Xilinx Spartan 3E family FPGA [15]. The complete synthesis results to Spartan-3E FPGA are presented in table 1, whose hardware was fit in an XCS500E device. The table 2 presents the comparison between [11] and the present work in this paper.

Table 1. Device utilization using Xilinx spartran-3E for total architecture proposed in this paper.

Logic Units	Used	Available	Utilization
Number of slices	1663	4656	36%
Number of Bonded IOBs	37	231	16%
Number of multipliers 18 x18	8	20	40%

According to synthesis result, maximum time delay produced is 8.861 ns. That constraint yields minimum clock period 8.006 ns. Maximum clock frequency can be used is 124.094MHz.

Maximum delay synthesized is much smaller than delay produced in [4]. 1D-DCT designed in [4] yields maximum time delay 76.03 ns.System [4] uses fully parallel processing without clock to compute 8 points 1D-DCT.

Table 2. Present paper's 2D-DCT result is compared with work of 2D-DCT design of [11].

Logic Units	This paper	Presented [1]
Number of slices	615	1891
Number of Bonded IOBs	74	51
Number of multipliers 18 x18	6	8

That system is used as comparison reference because it uses same FPGA with this system. Since the system in paper [1] uses Vertex FPGA that has higher frequency than Spartan, the delay is much smaller than this system and maximum frequency is higher. Figure below show the output results of the different stages

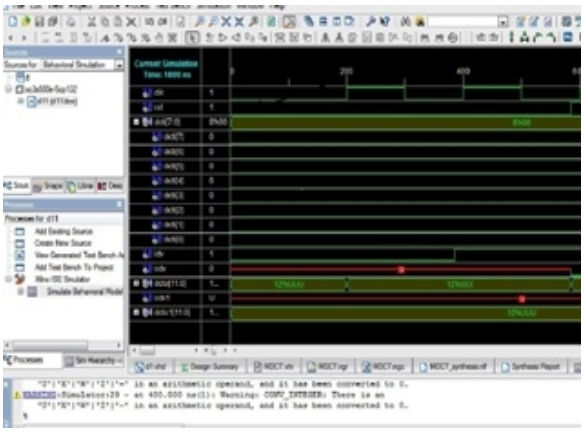

Fig. 6. 2D DCT Simmulation.

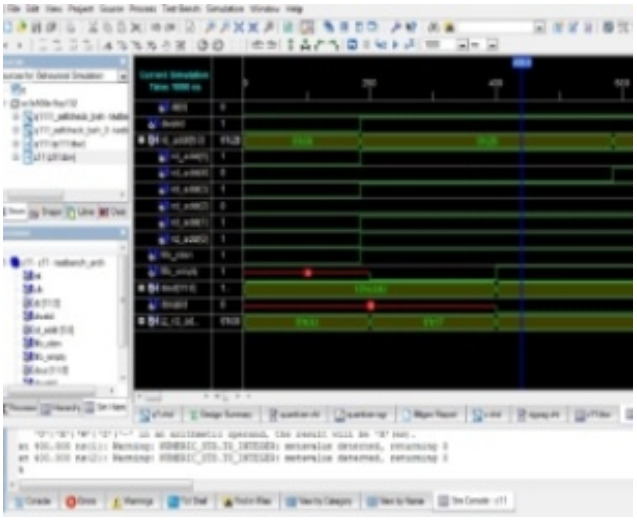

Fig. 7. Zigzag Simulation Result.

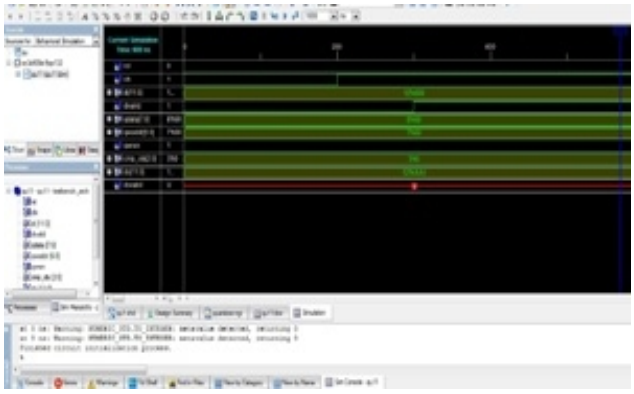

Fig. 8. Quantization Simulation Result.

V. CONCLUSION

The JPEG image compression is designed in VHDL and is tested with colour scale image. VLSI or Parallel pipelined implementation of an algorithm, reducing the no of multiplier is very important. The Compression yield MSE (Mean Square Error Value) =0.059492, which is computed for 64 bit of data is pipeline process in the system. The Latency produced from this system is 160 clock cycle. [20]. In this paper the architecture for the 2D-DCT is proposed. The design takes 615 No. of Slice, 51 No of Bonded IObs and 6 Multipliers and the area is also reduced when analyzed to previous work. The Maximum frequency accomplished by the system is 123 MHz. It is Applicable for implementing on FPGA According to Xilinx XCS500E [12].

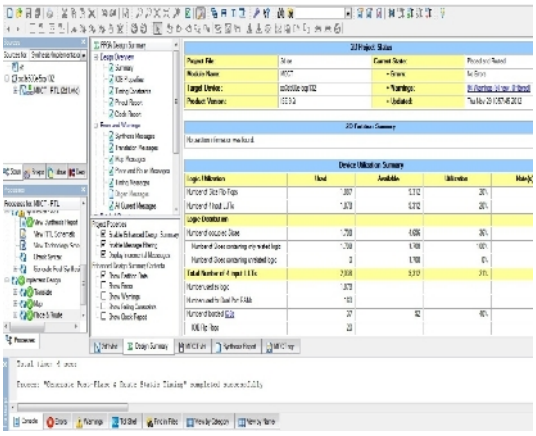

Fig. 9. Design Utilization Summery.

REFERENCES

[1]. Design & Implementation of JPEG2000 Encoder using VHDL ,"Kanchan H. Wagh, Pravin K. Dakhole, Vinod G. Adhau", Proceedings of the World Congress on Engineering 2008 Vol IWCE 2008, July 2 - 4, 2008, London, U.K.

[2]. Pipelined Fast 2-D DCT Architecture for JPEG Image Compression," Luciano Volcan Agostini, Ivan Saraiva Silva and Sergio Bampi", Federal University of Rio Grande do Sul - Microelectronics Group Caixa Postal 15 064 - Porto Alegre – Brazil.

[3]. The International Telegraph and Telephone Consultative Committee (CCITT). "Information Technology – Digital Compression and Coding of Continuous-Tone Still Images –Requirements and Guidelines". Rec. T.81, 1992.

[4]. W. Pennebaker, J. Mitchell. *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, USA, 1992.

[5]. "Home site of the JPEG and JBIG committees"<http://www.jpeg.org/> (21/04/01).

[6]. V. Bhaskaran, K. Konstantinides. *Image and Video Compression Standards Algorithms and Architectures – Second Edition Kluwer Academic Publisher USA 1999*

[7]. J. Miano. *Compressed Image File For*mats – JPEG, PNG, GIF, XBM, BMP, Addison Wesley Longman Inc, USA, 1999.

[8]. Design and FPGA Implementation of CORDIC-based 8-point 1D DCT Processor," Rohit Kumar Jain", 2010.

[9]. Iain Richardson. *Video Codec Design - Developing Image and Video Compression Systems*. Copyright by John Wiley & Sons Ltd. 2002 (ISBN0471485535).

[10]. Fpga Implementation of 2-D DCT Architecture for JPEG Image Compression," *Prashant Chaturvedi ,Prof. Tarun Verma ,Dr. Rita Jain* "*International Journal Of Advanced Electronics And Communication Systems*". Dec 2012.

[11]. T. Pradeepthi1 and Addanki Purna Ramesh "PIPELINED ARCHITECTURE OF 2D-DCT, QUANTIZATION AND ZIGZAG PROCESS FOR JPEG IMAGE COMPRESSION USING VHDL PIPELINED ARCHITECTURE OF 2D-DCT, QUANTIZATION AND ZIGZAG PROCESS FOR JPEG IMAGE COMPRESSION USING VHDL" *INTERNATIONAL JOURNAL OF VLSI DESIGN & COMMUNICATION SYSTEMS* (VLSICS) VOL.**2**, NO.3, SEPTEMBER 2011.

[12]. Trang T.T. Do, Binh P. Nguyen "A High-Accuracy and High-Speed 2-D 8x8 Discrete Cosine Transform Design". *Proceedings of ICGCRCICT* 2010, vol. **1**, 2010, pp. 135-138.

[13]. Xilinx, Inc., "2D Discrete Cosine Transform (DCT) V2.0 ", Logicore Product Specification, Xilinx Corporation, 2002.

[14]. The International Telegraph and Telephone Consultative Committee (CCITT). "Information Technology – Digital Compression and Coding of Continuous-Tone Still Images – Requirements and Guidelines". Rec. T.81, 1992.

[15]. "Home site of the JPEG and JBIG committees" <http://www.jpeg.org/> (21/04/01).

[16]. W. Pennebaker, J. Mitchell. *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, USA, 1992.

[17]. A. Shams, A. Chidanandan, W. Pan, and M. Bayoumi, "NEDA: A low power high throughput DCT architecture", *IEEE Transactions on Signal Processing*, vol.**54**(3), Mar. 2006.

[18]. B.G. Lee, — A new algorithm to compute the discrete cosine] transform‖ —*IEEE Trans. Acoustic., Speech, Signal Processing*, vol ASSp-**32** pp 1243-1245 Dec-1984.

[19]. H.S Hou, —A fast recursive algorithms for computing the] discrete cosine transform, —IEEE Trans. Acoust., Speech signal processing vol, ASSP-**35**,pp 1455-1461 oct 1987

[20]. I. Basri, B. Sutopo, "Implementation 1D-DCT Algoritma Feig- Wino grad di FPGA Spartan-3E (Indonesian)". *Proceedings of CITEE 2009*, vol. **1**, 2009, pp. 198-203.

[21]. J. Miano. *Compressed Image File For*mats – JPEG, PNG, GIF, XBM, BMP, Addison Wesley Longman Inc, USA, 1999.

[22]. N.I Cho and S.U.Lee, —DCT algorithms for VLSI parallel implementation, —*IEEE Trans. Acoust., Speech, Signal Processing*, vol **38**. pp. 121-127, Jan.1990.