FPGA Implementation of 2-D DCT for JPEG Image Compression

R.Uma
Electronics and Communication Engineering
Rajiv Gandhi College of Engineering and Technology
Puducherry, India
uma.ramadass1@gmail.com

Abstract— **Data (image) compression is the reduction or elimination of redundancy in data representation in order to achieve reduction in storage and communication cost. Many algorithms and VLSI architectures for the fast computation of DCT have been proposed [10,11,12]. For the fast computation of 2-D DCT, the conventional approach is the row-column method. This method requires 2N 1-D DCT's for the computation of NxN DCT. For hardware parallel implementation using the conventional approach, a complicated matrix transposition architecture and 2N 1-D DCT modules is required which increases the area as well as delay. Thus for more efficient computation or parallel implementation of the 2-D DCT, the algorithm that work directly on the 2-D DCT is the direct polynomial approach [13] in which the number of multiplications is reduced to 50 to 75% of the conventional approach and the number of 1-D DCT required will be N so area minimization can be achieved. This paper presents a linear, highly pipelined direct polynomial fast 2-D DCT algorithm hardware implementation in which the number of multiplication is reduced to 50% of the conventional row-column approach. The coding is simulated using Xilinx 9.1 ISE synthesized using Spartan II. The 1-D and 2-D DCT implementation is done using 18-bit floating point adders, subtractors and multipliers [15]. The DCT processor saves hardware using module reusability concept and achieves high performance. The chip of 8x8 DCT processor is fabricated in a 180 nm CMOS technology. Power analysis is analyzed using CADENCE tool.**

Keywords- **Discrete Cosine Transform (DCT), Field Programmable Gate Array (FPGA), Joint Photographic Expert Group (JPEG), Register Transfer Logic (RTL).**

I. INTRODUCTION

Image compression is the art / science of efficiently coding digital images to reduce the number of bits required in representing an image. Recent advances in digital technology have led to communication media in which visual information plays the key role. Some of the emerging applications include high definition TV, videoconferencing, video telephony, medical imaging, virtual reality, video wireless transmission and video server. The raw digital image and video signal usually contain huge amount of information and therefore require a large channel or storage capacity. In spite of advances in communications channel and storage capacity, the implementation cost often put constraint on capacity. Generally, the transmission or storage cost increase with increase in bandwidth requirement. To meet the channel or storage capacity requirement, it is necessary to employ compression techniques, which reduce the data rate while maintaining the subjective quality of the decoded image or video signal. Image compression techniques achieve compression by exploiting statistical redundancies in the data and eliminating or reducing data to which the human eye is less sensitive.

There are two types of redundancies that occur in images. They are spatial and spectral redundancy. Spatial redundancy is due to the correlation between neighboring pixels. Spectral redundancy is due to correlation between different color planes. Both spatial and as well as spectral redundancies can be removed using subband coding or transform coding (Discrete Cosine Transform) which are two well-known techniques. There is other type of redundancy called temporal redundancy which is due to the correlation of different frames in a sequence of images such as in videoconferencing applications or broadcast images. These temporal redundancies are removed by an interframe coding called Motion Compensated Predictive Coding.

DCT is a computation intensive operation. Since DCT approaches the statistically optimal Karhunen-Loeve Transform (KLT) for highly correlated signals [13]. DCT is used in most of the digital processing application because of its energy compaction characteristics. The development of efficient algorithms for the computation of DCT (more specifically DCT –II) began soon after Ahmed et al (1974) [14] reported their work on DCT. It was natural for initial attempts to focus on the computation of DCT by using Fast Fourier Transform (FFT) algorithms. Many algorithms for fast computation of DCT are reported in the literature [10-13]. General approach used in DCT is converting the image pixels of block 8x8 or 16x16 into series of coefficients that define spectral composition of the block. Its direct implementation requires large number of adders and multipliers. Conventional approach used for 2-D DCT is row-column method. This method requires 2N 1-D DCT‘s for the computation of NxN DCT and a complex matrix transposition architecture which increases the computational complexity as well as area of the chip. On the other hand if the DCT processor is designed using polynomial approach [13,16] reduces the order of computation as well as the number of adders and multipliers used in the DCT processor will be reduced and area reduction can be considerably achieved. Thus the main objective of this paper work is to design a DCT processor in which area reduction is achieved using pipelined hardware architecture using reusability concept.

This paper is organized as follows. Section II gives an overview of JPEG encoding procedure. Following section III gives the computation of DCT. Section IV explains the design flow of DCT algorithm. Section V presents the hardware architecture of 1-D DCT along with the design of floating point adder/subtractor, multiplier. The module is functionally tested using Xilinx ISE 9.1 and its simulation and synthesis results are presented. Section VI presents the implementation of 2-D DCT and its simulated result is shown. Section VII gives the synthesis, power and area estimation and conclusion is given in section VIII.

II. JPEG IMAGE COMPRESSION OVERVIEW

The JPEG image compression standard [1,2] was developed by Joint Photographic Expert Group [3]. JPEG is a sophisticated lossy/lossless compression method for color or gray scale still images. It also works best on continuous tone images, where adjacent pixels have similar colors [4].
One of the advantages of JPEG is the use of many parameters, allowing the user to adjust the amount of data lost (and thus also the compression ratio) over a very wide range. JPEG unlike other formats like PPM (Predict, Probability Model), GIF (Graphics Interchange Format) and PGM, is a lossy compression technique, this means some visual information is lost permanently. The key is making JPEG work is choosing what data to throw away. The JPEG encoder block diagram is shown in Figure (1).

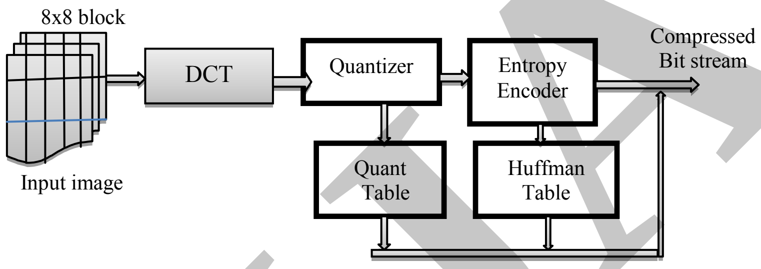

Figure 1 Block Diagram for JPEG Encoder

Image is divided into 8x8 blocks. The pixels of each color component are organized in groups of 8x8 pixels called data units. If the number of image rows or columns is not a multiple of 8, the bottom row and the rightmost column are duplicated as many time as necessary[17]. The discrete transform (DCT) is then applied to each data unit to create a 8x8 maps of frequency components. They represent the average pixel value and successive higher-frequency changes within the group.

For a given 2-D data sequence { X_{ij}: i,j = 0,1 – N-1}, the 2-D DCT sequence { Y_{mn}: m,n = 0,1-N-1} is given by

$$Y_{mn} = \frac{4}{N^2} u(m)u(n) \sum_{i=0}^{N-1}\sum_{i=0}^{N-1} X_{ij} \cos\frac{(2i+1)m\pi}{2N} \cos\frac{(2i+1)n\pi}{2N} \quad (1)$$

Where u(m) = u(n) = {$1/\sqrt{2}$, m = 0, n = 0
1 otherwise

X_{ij} is the pixel intensity.
The DCT can be expressed in a matrix form

$$DCT = C\ X\ C^T \quad (2)$$

Where X is the input matrix, C is the Cosine coefficient matrix and C^T its transpose with constants (1/2)u(m) and (1/2) u(n) absorbed in C and C^T matrix respectively.
Each of the 64 frequency components in a data unit is divided by a separate number called its Quantization Coefficient (QC), and then rounded to an integer. Quantizer step size is determined by the acceptable visual quality of image. After quantization, coefficients are arranged in increasing frequency order. For each non-zero DCT coefficients, JPEG records the number of zeros that preceded the number of bits needed to represent the number's amplitude. To consolidate the runs of zero, JPEG processes DCT coefficients in the Zig –Zag pattern[18]. The top left coefficient is called DC coefficient and the rest of the 63 coefficients are called AC coefficients. AC coefficients are coded by entropy coding whereas difference of DC coefficients from previous block is coded by entropy coding [19]. JPEG committee provides a standard table to be used for entropy coding.

III. DCT COMPUTATION

Discrete Cosine Transform (DCT) is a computation intensive algorithm has a lot of electronic applications [5]. DCT transforms the information from the time or space domains to the frequency domain to provide compact representation, fast transmission, memory saving and so on. DCT algorithm is very effective due to its symmetry and simplicity [6]. DCT computation requires a large number of adders and multipliers for direct implementation. Multipliers consume more power and hence distributed arithmetic (DA) is used to implement multiplication without multiplier [7]. DCT implementation using DA is done in literature [8,9]. A number of Fast Cosine Transform (FCT) algorithms like Lee, Chen, Smith, Loefflar have been mentioned in the literature [10,11,12]. Among them most efficient is the Nam Ik Cho (Chan and Ho) algorithm [13] is used in this paper for the implementation of 2-D DCT because this algorithm is not scaled. Therefore it is not multiplied by any scalar factor consequently no shift in transformation. Mathematical computation is simple. Hence reduces the complexity for DCT calculation and gives better resolution. This algorithm follows a similar logic to that of FFT algorithm, except that it is preceded with a reordering scheme of the input data. The assumption made in this derivation is that N is a multiple of 2.
The one-dimensional discrete cosine transform (DCT) is given by

$$X(k) = \frac{2\varepsilon_k}{N} \sum_{n=0}^{N-1} x(n) \cos\left(\frac{\pi(2n+1)k}{2N}\right) \quad (3)$$

Where ε_k = $1/\sqrt{2}$ for k = 0
= 1 otherwise and k = 0,1,2,…….,N

The input data x(n) is reordered as follows

$$\left.\begin{aligned}\tilde{X} &= x(2)\\ \tilde{X}(N-n-1) &= x(2n+1)\end{aligned}\right| \quad n=0,1,2,\ldots,N/2-1$$

Substituting in (3) the DCT equation becomes

$$X(k) = \sum_{n=0}^{N/2-1} x(2n)\cos\pi \frac{(4n+1)k}{2N} + \sum_{n=0}^{N/2-1} x(2n+1)\cos\pi \frac{(4n+3)k}{2N} \qquad (4)$$

Following the decimation-in-frequency approach utilized in the derivation of FFT, X(k) can be expressed as even and odd indexed elements. The various stages (butterfly operations, bit reversal, and recursive additions) are combined in a single flow graph as shown in Figure (2). From this flow graph the hardware implementation is preceded using VHDL (Very High speed Hardware Description Language). The algorithm development procedure is almost identical to the FFT with the exception that a recursive addition stage is needed. Lookup tables for the cosine values should have been used, as well as lookup tables for bit reversal are used.

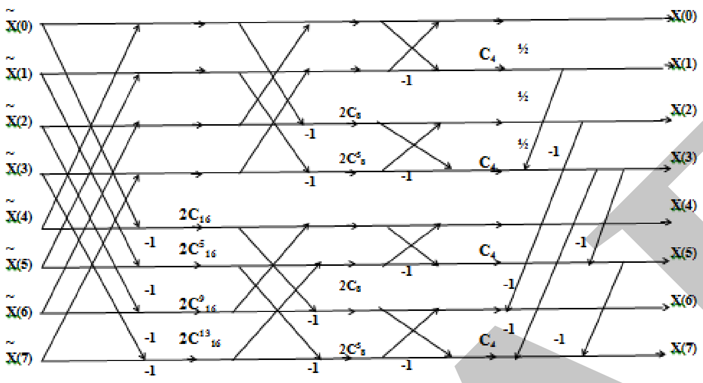

Figure 2 Final butterfly Network

IV. DESIGN FLOW FOR DCT ALGORITHM

The design flow starts with the MATLAB environment for the computation of image pixels. The image is segmented into 8x8 or 16x16 block. After determining the pixel values the datas are stored in the RAM for the Computation of DCT. The time domain coefficients are converted into spectral coefficient. The architecture proposed in this work is based on parallel polynomial approach with butterfly computation. Core is implemented to take 8-bit input data and produces 18-bit signed output coefficient for each stage of 1-D DCT. The implementation requires floating point adder/subtractor and multiplier. The advantages of this method are regular structure, simple control, interconnection and good balance between performance and complexity of implementation. The implemented design functionality is tested using Xilinx simulator and synthesized using Xilinx Spartan II. Similarly the computation of 2-DCT is performed and its functional verification is carried out. The output of 8x8 block produces 64 DCT coefficients which is stored in a RAM. Finally the DCT core‘s area and power analysis is performed using cadence 180 nm TMSC library. The design flow for 2-DCT algorithm is shown in Figure (3).

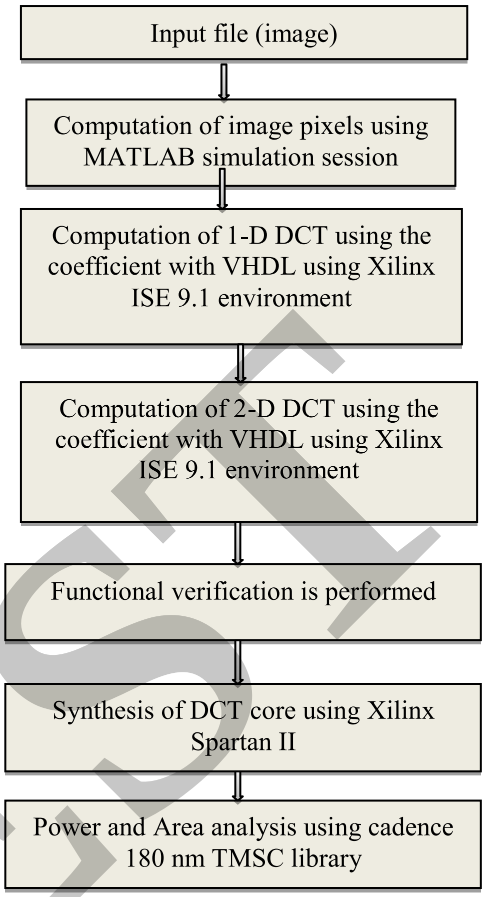
Input file (image)
Computation of image pixels using MATLAB simulation session
Computation of 1-D DCT using the coefficient with VHDL using Xilinx ISE 9.1 environment
Computation of 2-D DCT using the coefficient with VHDL using Xilinx ISE 9.1 environment
Functional verification is performed
Synthesis of DCT core using Xilinx Spartan II
Power and Area analysis using cadence 180 nm TMSC library

Figure 3 Design flow for DCT algorithm

V. IMPLEMENTATION OF 1-D DCT

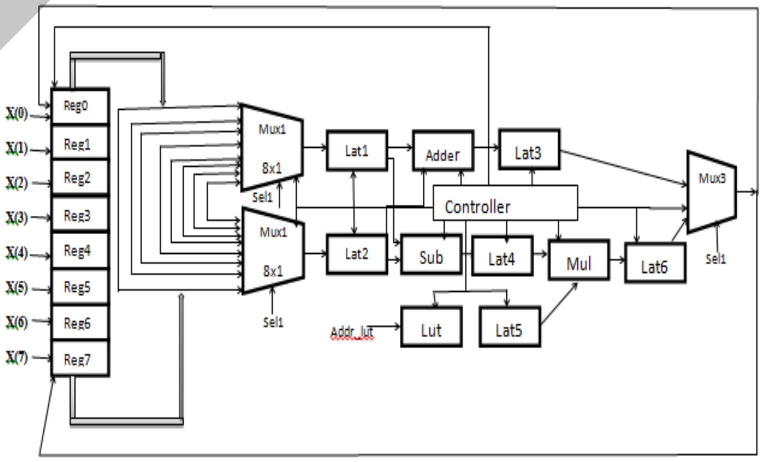

Figure 4 Architecture of 1-D DCT

The architecture of 1-D DCT is implemented from the butterfly network from Figure (2). This architecture consist of eight 18-bit register, two 18-bit multiplexers (8x1), one 18-bit multiplexer (2x1), one floating point adder (18-bit), one floating point subtractor (18-bit), one floating point multiplier (18-bit), six 18-bit latch and a controller. The design considerations made in this architecture are: Exploiting parallelism, pipelining and reusability concept is incorporated so that area minimization can be achieved considerably. Since the direct implementation of butterfly network needs 12 adders, 21 subtractors and 12 multipliers whereas the architecture shown in the Figure (4) uses only single adder, subtractor and multiplier repeatedly reused there by reducing the area and cost of the chip.

A. Sequence of operation

The sequence of operation carried over in this architecture are: Initially the 8 inputs [x(0),x(1)………..x(7)] i.e., the pixel image values are serially loaded into eight 18-bit registers. All the outputs of registers are given to the multiplexer1 and multiplexer2. The register values are selected depending upon the control that is given by the controller. The controller is designed using a Finite State Machine (FSM) to control the overall operation of the architecture. Therefore the value in one register is selected by the multiplexer at a time. The outputs of multiplexer1 and multipexer2 are latched at latch1 and latch2. Correspondingly the values from latch1 and latch2 are given to adder and subtractor module. The output of adder and subtractor are stored in latch3 and latch4. The cosine values are stored using lookup table and the controller provides the necessary cosine terms to the multiplier using latch5. Finally the outputs of adder and multiplier are stored in the selected registers depending upon the selection of multiplexer3.

B. Controller architecture

The architecture of controller is shown in Figure (5). The controller is designed using Finite State Machine (FSM). The total number of states required to complete this 1-D DCT will be 200. The controller uses clk, reset, start, addflag, subflag and mulflag signals as its input. The enables of registers, the enables of mux1, mux2, mux3, adder, subtractor, lut, multiplier, sel1, sel2, addr_lut are outputs of this controller. Operation of controller for single stage of butterfly network is presented in Table 1.

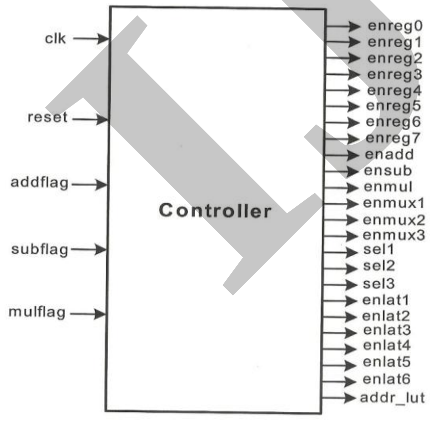

Figure 6 Architecture of controller

Table 1 Operation of controller for single stage of butterfly network

States	Operation
State1 to state8	The input values are loaded in the eight 18-bit register
State 9	Reg0 and Reg4 selected by mux1 and mux2
State 10	Reg0 + Reg4, Reg0 – Reg4
State 11	Output of Reg0 – Reg4 is multiplied with constant cosine value
State 12	The output of adder is selected by mux3 when sel3 is ‗0ƒ and the corresponding output value is stored in Reg0
State 13	The output of multiplier is selected by mux3 when sel3 is ‗1ƒ and the corresponding output is stored in Reg4

C. Design of floating point adder/subtractor and multiplier circuit

The 18-bit floating point format proposed in this work is a IEEE 754 standard [14] used to store floating point numbers. The format is shown in Figure (7).

S	E	F
17	16 10	9 0

Figure 7 18 - bit floating point representation

Where
S = sign bit (MSB)
E = exponent part (totally 7 bits)
F = mantissa part (totally 10 bits)
The floating point value (V) is computed by

$$V = (-1)^s \, 2^{(e-63)} \, (1.f) \qquad (5)$$

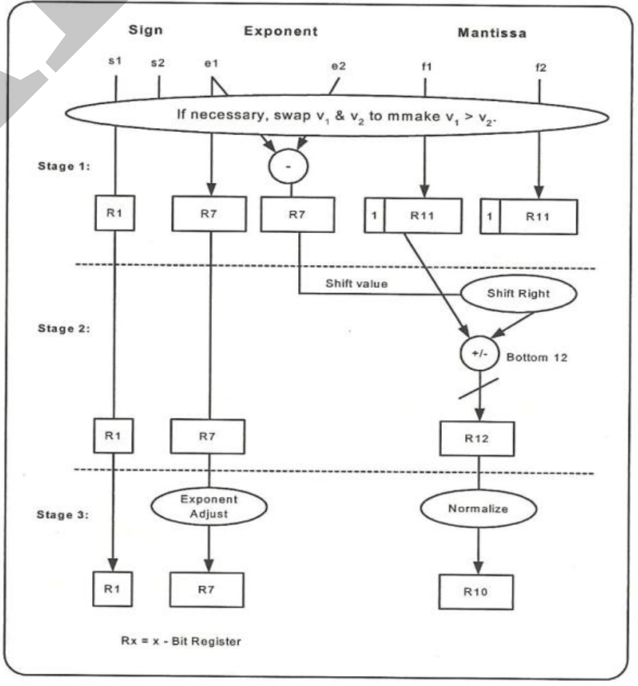

Figure 8 Floating point add/sub

The implementation of floating point adder/subtractor is shown in Figure 8. The computation is performed in three stages. The notation s_i, e_i, f_i are used to represent the sign, exponent and mantissa fields of the floating point number v_i.

Stage1: Swapping of v1 and v2 if the absolute value of v1 is less than v2 by checking the exponent and mantissa of each value. Subtract e2 from e1 in order to calculate the number of positions to shift f2 to the right so that the decimal points are aligned before addition or subtraction in stage2.

Stage2: shift 1.f2 to the right (e2-e1) places calculated in the previous stage. Add 1.f1 to 1.f2 if s1 equals s2 else subtract 1.f2 from 1.f1.

Stage3: Normalization of f3 is done by shifting it to the left until the high order bit is a one. Adjusting exponent of the result, e3, is done by subtracting it by the number of positions that f3 was shifted left.

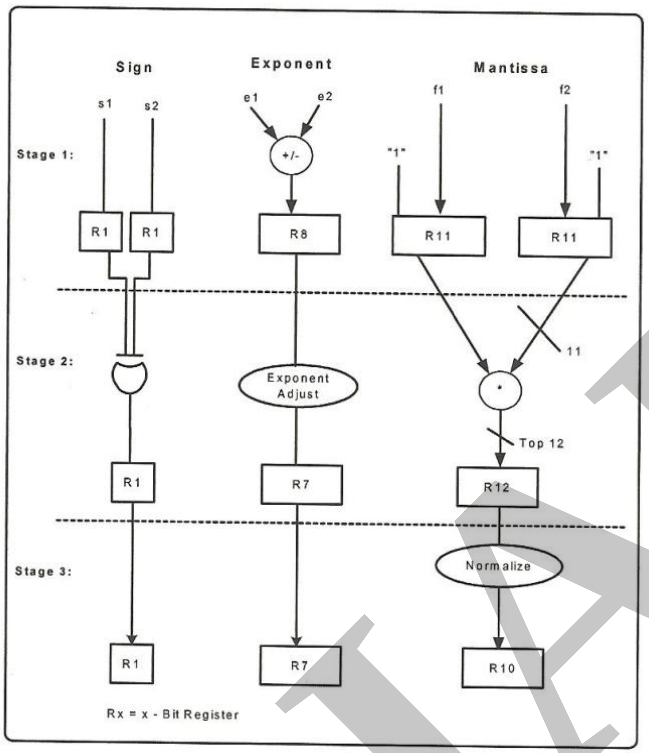

Figure 9 Floating point multiplier

The block diagram for the three stage 18 bit floating point multiplier is shown in Figure (9).

Stage1: The exponents, e1 and e2 are added and the result along with the carry bit is stored in an 8-bit register. The maximum represent is -63 under overflow. If there is over flow the result is set to the maximum number. For non-zero floating point concatenate f1 and f2 terms. Register only the sign bit in this stage.

Stage2: Multiply 1.f1 and 1.f2 each of 11 bit quantity. Store the 22 bit result in the register. Adjust the exponent part. Compare the sign bits of two operands if they are same assign +ve sign else assign –ve sign.

Stage3: Normalize the mantissa part. The resulting sign, exponent and mantissa fields placed into an 18-bit floating point word.

D. Result

The above 1-D DCT module is simulated using Xilinx ISE 9.1. Functional testing and timing analysis were carried out for this module. The results are verified and synthesized using Spartan II. The proposed architecture was tested with 9000 ns (9µs) clock for 1-D DCT block and was found to be working satisfactorily. The simulated and synthesized results are shown in Figure 10, 11.

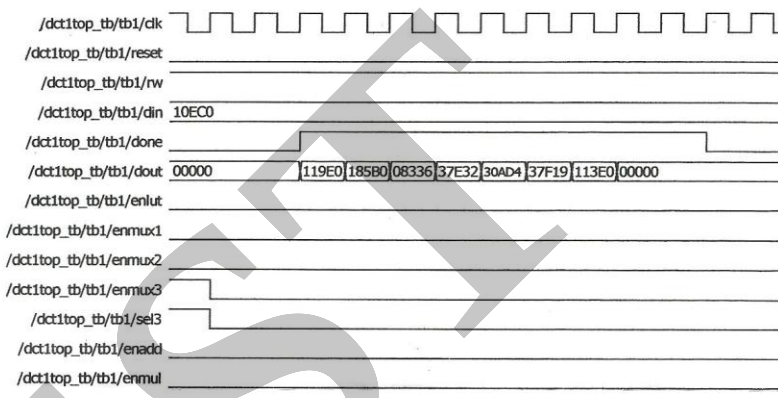

Figure 10 Simulated result of 1-D DCT

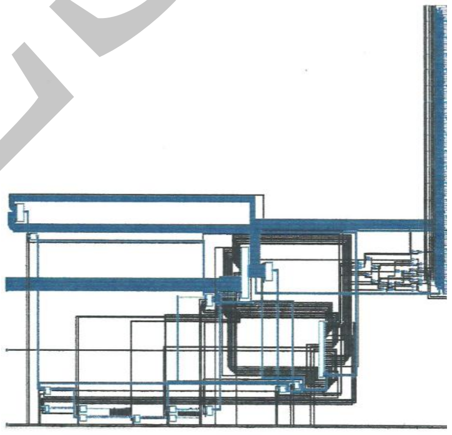

Figure 11 Synthesis result of 1-D DCT

VI IMPLEMENTATION OF 2-D DCT

--The implementation of 2-D DCT is show in Figure (12, 13, 14). The signal flow graph is separated into three parts for convenience. Figure (12) is the signal flow graph from x_{ij}'s to f_{pi}'s and g_{pi}'s. Figure (13) is from f_{pi}'s to Y_{mn}'s where n is even and Figure (14) is from g_{pi}'s to Y_{mn}'s when n is odd. From this flow graph it seen that the the 2-D DCT requires only 8 1-D DCT's. From Figure (13 and 14) it is seen that the addition operations after the 1-D DCT stages can be implemented in the butterfly form. The broken lines represent transfer factors -1 and the full lines represent unity transfer factor. O represents adder and $\rightarrow$ with ½ represents multiplication by ½ which is equivalent to shift operation.

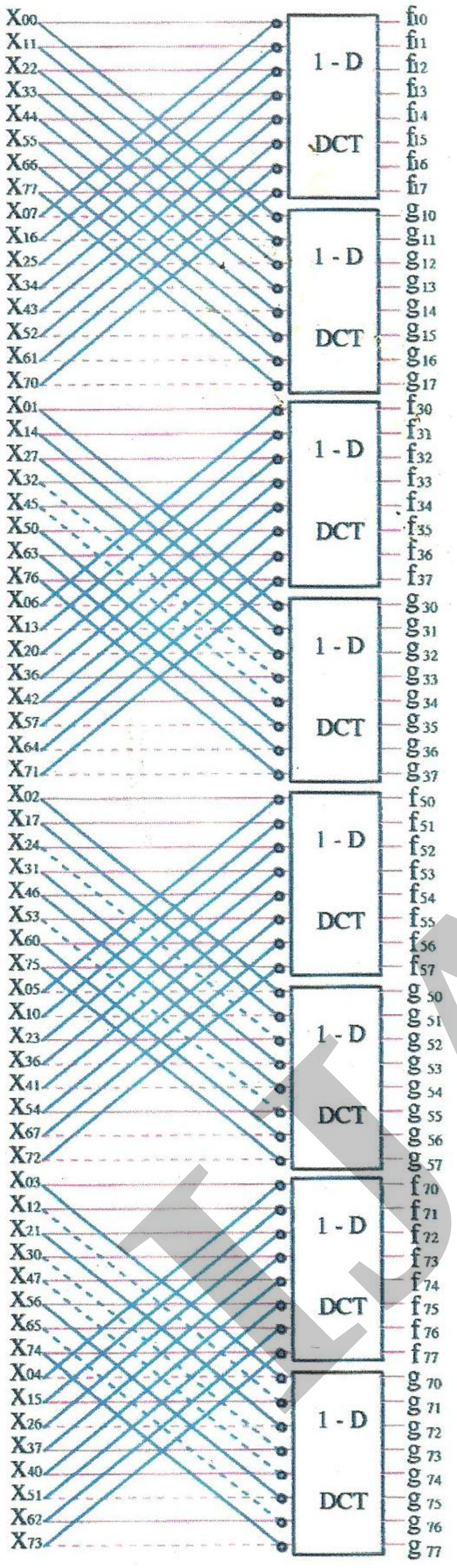

Figure 12 Signal flowgraph or 8x8 DCT

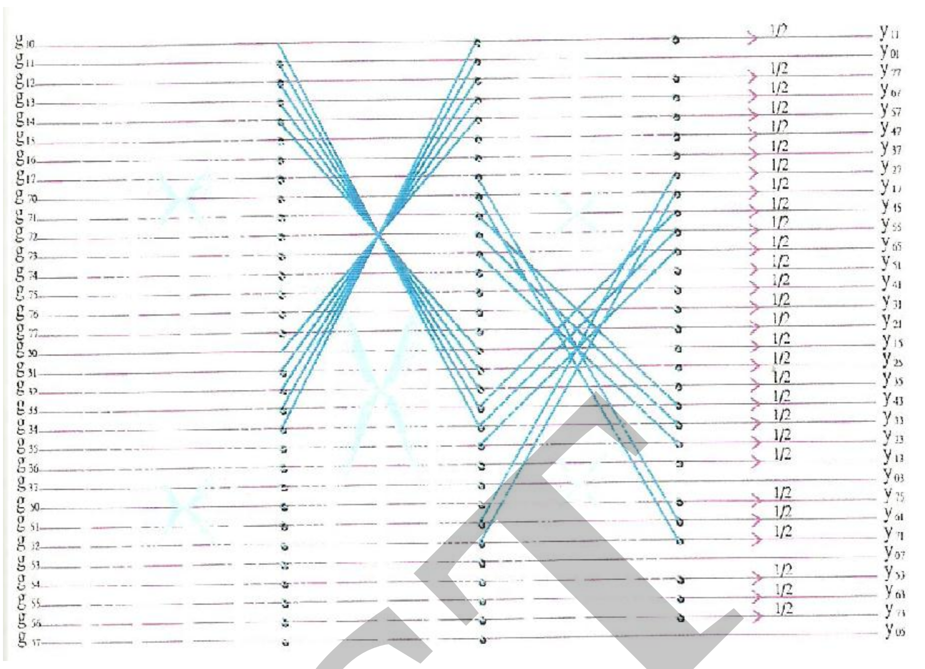

Figure 13 Signal flow graph from f_{pi} to Y_{mn} where n is even

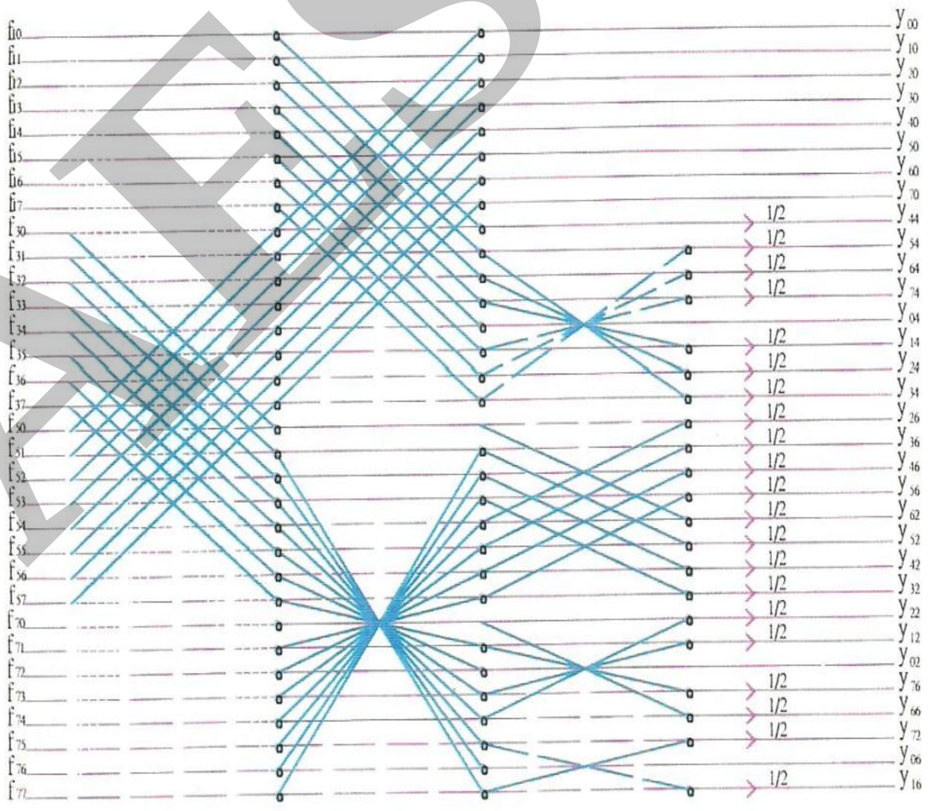

Figure 14 Signal flow graph from f_{pi} to Y_{mn} where n is odd

E. Achitecture of 2-D DCT

The architecture of 2-D DCT is shown in Figure (15). This architecture consists of RAM to store the 64 pixel values. The datas are loaded serially to the processing element PE1. The ouptut of PE1 is loaded in 64 bytes (18 bit) register. Totally 8 1-D DCT are used for 8x8 block computation. Ouputs of 1-DCTs are given to network1 and network2. The output of network1 and network2 are stored in 64x18 bits register.

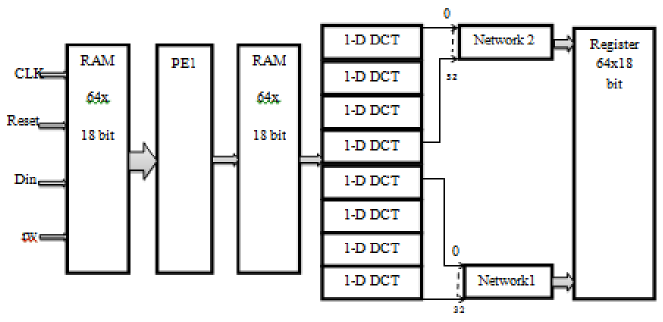

Figure 15 Architecture of 2-D DCT

F. Architecture of PE1

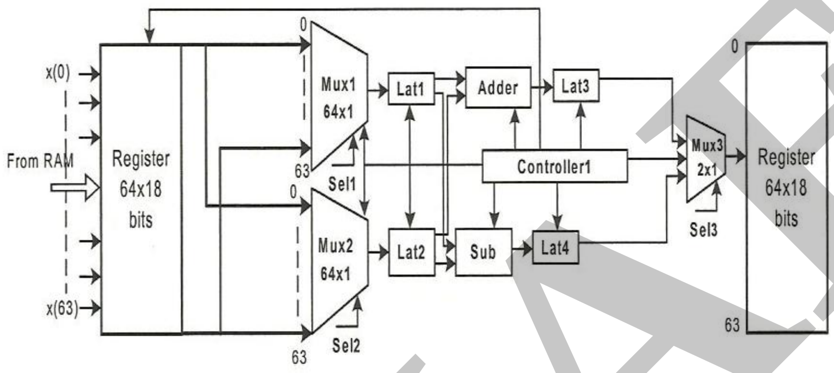

Figure 16 Architecture of Processing Element

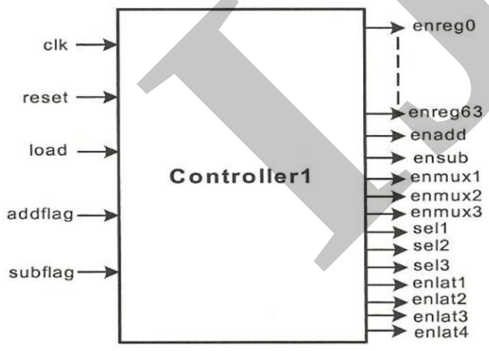

Figure 17 Controller of PE1

The architecture of PE1 and controller are shown in Figure (16 and 17). The processing element consists of 64x18 bits register where the 64 input datas are loaded through RAM. The outputs of registers are fed into multiplexer1 and multiplexer2. The mux1 and mux2 select one values depending upon the select signal provided by the controller. The outputs of mux1 and mux2 is latched at lat1 and lat2. The outputs of lat1 and lat2 are the inputs to adder and subtractor. The output of adder and subtractor are given to mux3 and depending upon the select line the outputs are placed at the register. Overall operation is controlled by controller1.

G. Architecture of Network1/Network2

The architecture of Network1 and Network2 are similar but the only difference is that the selection of 1-D DCT's values and the sequence of operation provided by the controller. The controller and architecture of Network1/Network2 are shown in Figure (18 and 19)

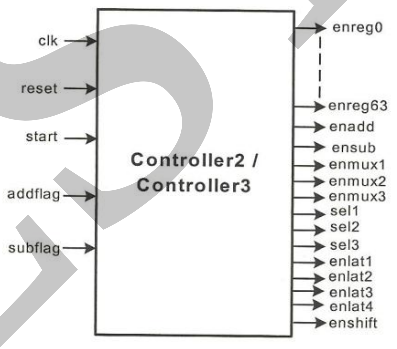

Figure 18 Controller of Network/Network2

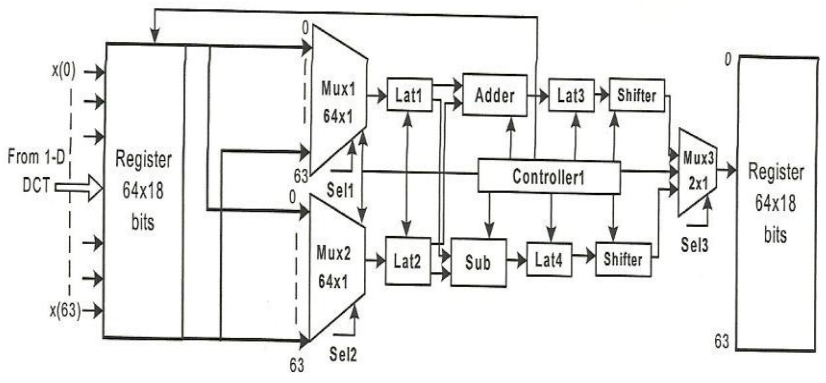

Figure 19 Architecture of Network1/Network2

H. Sequence of operation

Initially the 64 pixel values are loaded into 64x18 bit register through RAM. The process of loading is activated when rw and load signal is enabled. The computation of addition/subtraction is carried in the processing element (PE1). The 64 outputs are stored in 64x18 bit register. Through this register 8 values are loaded into each 1-D DCT module. The 1-D DCT module performs its operation. The outputs of 1-D DCT is loaded to Network1/Network2 through registers. Network1/Network2 does the computation of addition, subtraction and shifting operation of the signals. When done flag is set to one the final resulted value of 64 DC co-efficient will be stored in 64x18 bit register. The overall operation is controlled by controller1, controller2 and controller3 of PE, Network1 and Network2.

I. Result of 2-D DCT

The above 2-D DCT module is simulated using Xilinx ISE 9.1. Functional testing and timing analysis were carried out for this module. The results are verified and synthesized using Xilinx Spartan II. The proposed architecture was tested with 40000 ns (40 μs) clock for 2-D DCT block and was found to be working satisfactorily. Figure 20 shows the simulated result of 2-D DCT.

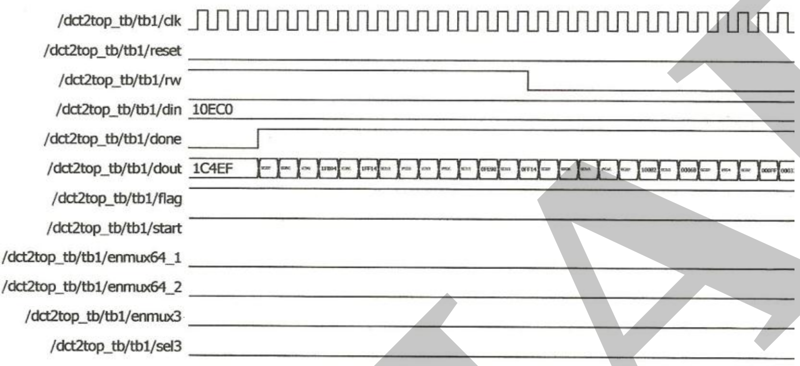

Figure 20. Simulated result of 2-D DCT

VII SUMMARY

For VLSI or hardware parallel implementation of an algorithm, reducing the number of multipliers is very important, because they occupy a large area of the chip. Also important considerations are regularity, modularity in the computation structure, and the complexity of data access scheme. In this context, the architecture proposed in this scheme reduces the number of multipliers being used for parallel implementation. Table 2 shows the hardware utilization summary implemented in Xilinx Spartan II. A considerable hardware savings is achieved using this architecture. After the module is synthesized the power and area analysis is done on to 180 nm TSMC library using RTL compiler and its characteristics is tabulated below in Table 3.

Table 2 Device utilization for Xilinx Spartan II

Features	2-D DCT core
Number of latches	5760
Number of function generators	5982
Number of MUX CARRYs	864
Number of MUXF5	187
Number of ports	40
Number of nets	1639

Table 3 Power and area analysis

Features	2-D DCT core
Power	2.134m
Area	05431 mm^2
Latency	85 cycles
Block size	8x8
Number of cells	5442

VIII CONCLUSION

In this paper, the architecture for the 2-D DCT is proposed. It is shown that the NxN DCT is obtained from N 1-D DCT‘s with some additions and shift operations. Thus the total number of multiplications required for the NxN DCT is N times that for the 1-D DCT, which is only half of that required for the conventional row-column approach. This architecture exploits hardware parallelism and explores the reusability concept. The image coefficients are obtained by using MATLAB and are applied as inputs to DCT core. The DC and AC coefficients are determined through the proposed architecture and resultant is stored in RAM. Finally the power analysis was done using RTL compiler with 180 nm TSMC library and low power is achieved.

REFERENCES

[1] 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.
[2] W. Pennebaker and J. Mitchell. *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, USA, 1992.
[3] ―Home site of the JPEG and JBIG committees” http://www.jpeg.org
[4] R. C. Gonzalez, R. E. Woods, Digital Image Processing, 2nd.Ed.,Prentice Hall, 2002.
[5] Roger Endrigo Carvalho Porto, Luciano Volcan Agostini ―Project Space Exploration on the 2-D DCT Architecture of a JPEG Compressor Directed to FPGA Implementation” IEEE, 2004
[6] VijayaPrakash and K.S.Gurumurthy.”*A Novel VLSI Architecture for Digital Image Compression Using Discrete Cosine Transform and” Quantization* IJCSNS September 2010.
[7] S. A. White, ―Applications of distributed arithmetic to digital signal processing: a tutorial review,” *IEEE ASSP Magazine*, vol.6, no.3, pp.4- 19, Jul.1989.

[8] 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.
[9] Peng Chungan, Cao Xixin, Yu Dunshan, Zhang Xing, ―A 250MHz optimized distributed architecture of 2D 8x8 DCT," 7th International Conference on ASIC, pp. 189 – 192, Oct. 2007.
[10] B.G. Lee, ―A new algoritm to compute the discrete cosine transform" ―IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-32, pp. 1243-1245, Dect.1984.
[11] 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.
[12] 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.
[13] Nam Ik Cho, Sang Uk Lee ―Fast Algorithm and Implementation of 2-D Discrete Cosine Transform, ―IEEE Transaction on Circuits and Systems", Vol.38,No.3, March 1991.
[14] N. Ahmed, T.Natarajan, and K.R. Rao, ―Discrete Cosine Transform," IEEE Trans. Commun., vol, COM-23, pp. 90-93, Jan. 1974
[15] Nabeel Shirazi, Al Walters, and Peter Athanas ―Quantitative Analysis of Floating Point Arithmetic on FPGA Based Custom Computing Machines" presented at the IEEE Symposium on FPGAs for Custom Machines, Napa Valley, California, April 1995
[16] M. Vetterli, ―Fast 2-D Discrete Cosine Transform,"in Proc. ICASSP'85. Mar.1985.
[17] Jongsun Park Kaushik Roy *A Low Complexity Reconfigurable DCT Architecture to Trade off Image Quality for Power Consumption* Received:2 April 2007 / Revised: 16 January 2008 / Accepted: 30 April 2008 /Published online: 3 June 2008.
[18] Jin-Maun Ho, Ching Ming Man, ―The Design and Test of Peripheral Circuits of Image Sensor for a Digital Camera," *IEEE International Conference on Industrial Technology, 2004. IEEE ICIT '04*, vol.3, pp.1351 – 1356, 8-10 Dec. 2004.
[19] Junqing Chen, Kartik Venkataraman, Dmitry Bakin, Brian Rodricks, Robert Gravelle, Pravin Rao, Yongshen Ni, ―Digital Camera Imaging System Simulation," *IEEE Transactions on Electron Devices,*vol.56(11), pp.2496 – 2505, Nov. 2009.
[20] A. Kassem, M. Hamad, E. Haidamous, ―Image Compression on FPGA using DCT," *International Conference on Advances in Computational Tools for Engineering Applications, 2009*, *ACTEA '09*, pp.320-323, 15-17 July 2009.

AUTHOR PROFILE

She is graduated B.E (EEE) from Bharathiyar University Coimbatore in the year 1998, Post graduated in M.E (VLSI Design) from Anna University Chennai in the year 2004. Currently she has been working as Assistant Professor in Electronics and Communication Engineering, Rajiv Gandhi College of Engineering and Technology, Puducherry. She has been teaching VLSI Design, Embedded Systems, Microprocessor and Microcontrollers for PG and UG students. She authored books on VLSI Design. She has published several papers on national conference and symposium. She is the guest faculty for Pondicherry University for M.Tech Electronics. He has been actively guiding PG and UG students in the area of VLSI, Embedded and image processing. She has received the best teacher award for the year 2006 and 2007. Her research interests are Analog VLSI Design, Low power VLSI Design, Testing of VLSI Circuits, Embedded systems and Image processing. She is a member of ISTE