International Journal of Computer Applications (0975 – 8887)
International Conference on Quality Up-gradation in Engineering, Science and Technology (ICQUEST2015)

FPGA Implementation of Image Compression Algorithm using Angular Domain

Pravin B. Pokle
Research scholar,
member IEEE B,D,C.E.Sewagram,
Dist.Wardha.(India)

N.G. Bawane, PhD
Principal and senior member IEEE
S.B.J.I.T.M.R., Nagpur (India)

ABSTRACT
Image compression is the science of reducing the size of image file in bytes by reducing the redundancy between pixels in an image without degrading the quality of image so that it can store more images in a given amount of disk or memory space and also it tends to reduces the time required to send the images over the network.Many hardware efficient techniques exist,inspired from it this paper, propose an image compression technique based on the pixel-wise fidelity and its FPGA implementation. The proposed method is used to reduce the bit rate of the pixels for better image compression by using angular transformation. Here propose an hardware efficient FPGA architecture using angular domain concept based on CORDIC algorithm is presented. In this paper, the architecture is first simulated in MATLAB for calculating PSNR, MMSE and compression ratio and then it simulated and synthesized using Xilinx ISE tool and verify the parameters such as area, power and delay required for compressing the image with visual appearance of the output compressed image.

Keywords
Image compression, Angular transformation, VHDL, FPGA, CORDIC, Bit plane slicing, MMSE, PSNR, RTL.

1. INTRODUCTION
In day to day life, image compression is very essential for transmission and storage of data. Recently, many researchers have been proposed different techniques and algorithms for image compression such as JPEG, MPEG and H.263 so as to improve the performance measures in last decades in order to achieve the better performance and quality. The key factor in image compression is to remove similar pixels by discarding the redundant pixels in an image [1].Almost all the methods proposed are either define in frequency domain or in time domain. The major drawback of using frequency domain is that the low frequencies pixels are discarded whereas the time domain reduces the size of an image this will degrade the visual appearance of the image [2]. However every algorithm has its advantages and disadvantages due to nature of each scheme. Amongst all the schemes proposed till now are based on discrete cosine transform. Vector Quantization is the recent technique developed for image compression. The performance of VQ is directly proportional to the vector size and codebook size [3,4]. This means with increase in vector size, the codebook size also increases which results in exponential increase in encoding complexity. On the other hand, microprocessor and microcontroller based systems can process image data very easily with less complexity, but these are not able to target onto actual hardware. Thus the efficient technique is come into existence using HDL [5]. This paper proposesa hardware efficient FPGA architecture scheme based on angular transformationfor the better image compression. This scheme considers angular position of each pixel under a sine wave by converting every pixel into angle by using CORDIC algorithm. For achieving further compression of image data, bit plane slicing is used. The main objective of this paper is to develop an efficient hardware on FPGA chip for compression of an image using angular domain in order to optimize area, power and delay and to achieve high compression ratio without degrading the original image quality.

2. PROPOSED METHOD
This paper presenteda new concept for image compression using angular transformation based on CORDIC (coordinate rotation digital computer) algorithm. The compression is achieved in three steps i.e. angular transformation using CORDIC, divider block and Bit plane slicing. Before processing an input image for HDL simulation, first step is to convert all pixels into text IO file format and stored it into memory block in which all the pixels are properly arranged so that further it will process pixel by pixel. Figure 1 below shows the complete top level image compression system.

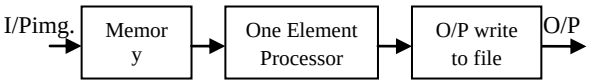

Fig.1: Top level entity for proposed scheme.

In proposed scheme, the input image is first stored in memory block in column and row matrix form because VHDL language cannot read image directly. It needs to convert first the image data into equivalent text I/O file format. In which if column matrix is equal to row matrix is greater than the size of image then go to the next row, thus the pixel wise image data in readable form is obtained[6]. Now this data is process pixel by pixel for converting into corresponding angle using one element processor. Once one element processor has finished processing all pixels then output writer could write the image into file after output. This will results in compression of image at the output. In this paper the top level block is further divided into smaller subsystem so that it can be easily implemented on FPGA.

2.1 Memory Block
In memory block the input image for the compression is passed. Image is created with the multiple pixels. For the angular transformation it requires single pixel by pixel input. In memory block the image is decomposed in the number of rows and columns according to the pixels resolution of the image. For every column and row pixel first check whether its size is greater than size of image, if it's true then go to next row for the column and repeat the process and if the size of pixel is less then image size then save the value of pixel in memory and pass this pixel to the one element processor for the angular transformation process. The block diagram of the memory block shown in figure 2.

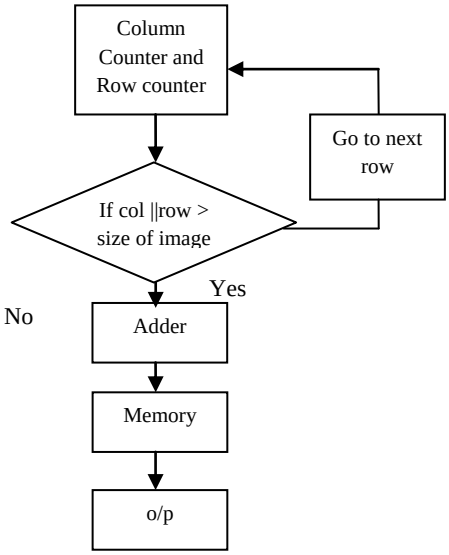

Fig. 2 Block Diagram of memory block

3. SYSTEM BLOCK DIAGRAM

Top level entity is further divided into subsystem as shown in figure 3 below: One element processor consists of sine coder, divider block and bit plane slicing.

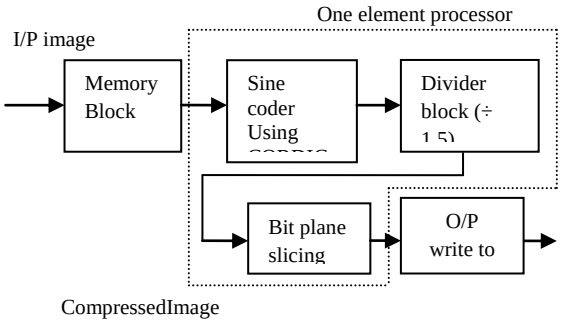

Fig.3: Complete Image Compression.

In this work, the CORDIC processor is used to achieve angular transformation in VHDL for converting pixels into respective phase angle then performs division so as to reduce bit rate and finally bit plane slicing is used for further compression of image at the compressor.

3.1 Angular transform

According to sine transform, the sine transform has the property that for two different angles, it has same gray level value.i.e. Pixels at an angle 89^0 an 91^0 have the same gray level value:

$$\text{i.e. } \sin x \quad ,-\frac{\pi}{2} \le x \le \frac{\pi}{2} \text{.....(1)}$$

For the assigned sequence P(n), angular transform is given by:

$$P(j) = \sqrt{\frac{2}{N+1}} \sum_{i=1}^{N} p(i) \sin \frac{ji\pi}{N+1} \text{..... (2)}$$

So, in this work first all the pixels are converted into corresponding angle under sine wave which tends to reduce the one bit and hence it requires only seven bits instead of eight bits.

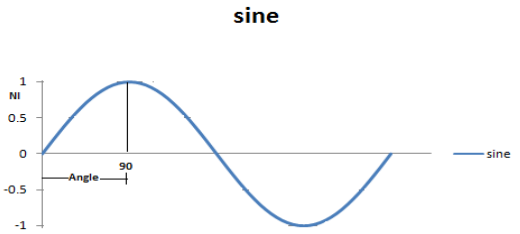

Fig. 4: Sine wave of the Normalized image

3.2 Cordic Algorithm

CORDIC, Coordinate rotation digital computer is a fast and efficient algorithm for trigonometric calculations. The algorithm was first proposed by Jack E. Volder [7]. It has two modes (vector mode and rotation mode) and has three resisters, namely P, Q and Z(Angle). In this paper, rotation mode is use to find sine or cosine of an angle iteratively using simple arithmetic such as add, subtract, shift ,compare table look-up. In this the resister Z is used to stored the angle whose sine or cosine is found and resisters P and Q are set to 1 and 0 in order to represent horizontal vector. The angle in Z is changed successively smaller in steps with a view to finally make it zero. The corresponding rotation of the vector can be represented after updating P and Q register with each change in Z. After sufficient iterations, Z nearly equal to 0 and register P represent cosine and register Q represent Sine of the original angle [8].

On the other hand the vector mode is used to find arc tangent of the number. In this the Z register is set to 0 and P and Q registers are initialized such that $\tan\Phi = Q/P$, where Φ is the corresponding angle. After sufficient iterations Q becomes 0 and Z containsΦ.

Figure 5 shows the relation between angle of rotation and corresponding change in x, y coordinates. From the figure it is seen that at an angle Φ_1 the vector is originally inclined to the x axis. This will results in p_1 and q_1 coordinates respectively [9]. Further it is rotated by an angle Φ in a counter clockwise direction so that it is now inclined to x axis at an angle Φ_2.Thus the new coordinates are p_2 and q_2 respectively.

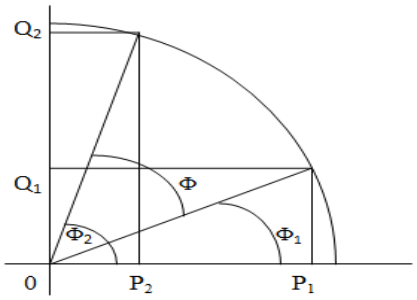

Fig.5: Coordinate rotation

Now as per the trigonometric equation for sin(P+Q) and cos(P+Q):

$P_2 = P_1 * \cos(\Phi) - Q_1 * \sin(\Phi)$ …….(3)

$Q_2 = P_1 * \sin(\Phi) + Q_1 * \cos(\Phi)$ …….(4)

This becomes,

$P_2 = \cos(\Phi) * [P_1 - Q_1 * \tan(\Phi)]$ ……..(5)

$$Q_2 = \cos(\Phi) * [P_1 * \tan(\Phi) + Q_1] \quad \text{.....(6)}$$

In rotation mode, the initial angle is θ which can be rotated in steps Φi to become 0. Now at i^{th} step, choose tanΦi as a fractional power of 2 such that we can replace multiplication by tan(Φ) by right shift. Thus we have,

$$\Phi i = \tan^{-1}(2^{-i}) \quad \text{.....(7)}$$

Here multiplication by cos(Φ) tends to a constant value K irrespective to the initial values of P,Q and Z thus we can ignore it.Where K is defined as:

$$K = \cos(\tan^{-1}(2^{-0}))* \cos(\tan^{-1}(2^{-1}))* \cos(\tan^{-1}(2^{-2}))* \quad \text{.....(8)}$$

$$K = \prod_{N=0}^{\infty} \frac{2^N}{\sqrt{1+2^{2N}}} = 0.607 \quad \text{.....(9)}$$

Thus if we consider cosine is equal to p_{in}, sine equal to q_{in} and angle equals to z_{in} then iterations per pixels are as follows:

Q = q = 0, P = p = 0.607 and Zin= θ Then

p_0 = pi + q when Zin < 0 else pi – q

q_0 = qi - q when Zin < 0 else qi + q

The architecture required for one step cordic iteration is shown in figure 6. Depending upon the number of iterations required for a given image above architecture is required in pipelined manner. In VHDL it can be done with the help of generate statement by simply passing generic parameter which is equal to the number of iterations as shown in following figure.

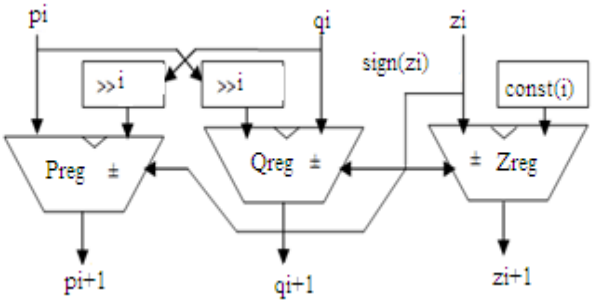

Fig.6: one step cordic iteration

This means that multiplying by K tends to initialized X with 0.607 Instead of 1. Thus all pixels are converted into angular values in the range 0 to 90 degrees [10].

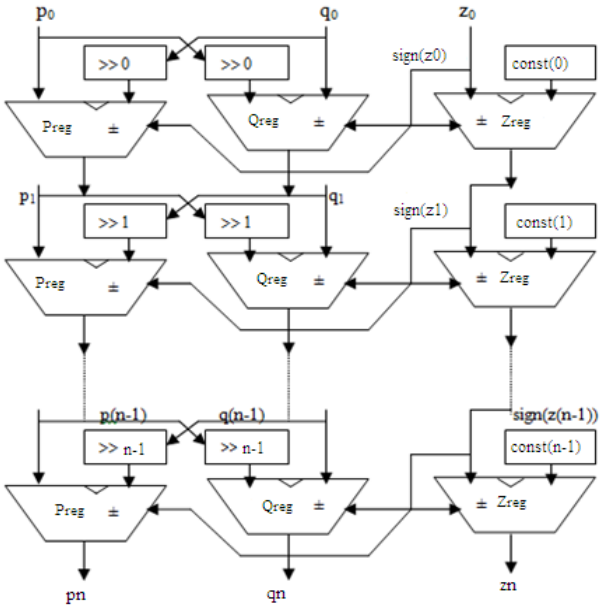

Fig.7: N step cordic iteration

By iteratively rotating Φ towardsθ, $\sin^{-1}$ (θ) can be calculated

Step1. Set $\Phi = 45^0$

Step2. If θ>= Φ then

$\Phi = \Phi + (45/2)^0$ else

$\Phi = \Phi - (45/2)^0$

Step3. If θ>= Φ then

$\Phi = \Phi + (45/4)^0$ else

$\Phi = \Phi - (45/4)^0$

Continue by halving step size

For example, for 5 x 5 images as shown below:

$$P_{ij} = \begin{bmatrix} 180 & 200 & 197 & 250 & 198 \\ 165 & 243 & 176 & 199 & 220 \\ 120 & 189 & 221 & 245 & 195 \\ 180 & 176 & 229 & 212 & 147 \\ 167 & 134 & 172 & 178 & 189 \end{bmatrix}$$

After sufficient number of cordic iterations, all the gray level values are get converted into corresponding angular values. Thus the resulting matrix is given as:

$$Q_{ij}' = \begin{bmatrix} 46^\circ & 55^\circ & 52^\circ & 90^\circ & 52^\circ \\ 51^\circ & 76^\circ & 44^\circ & 52^\circ & 62^\circ \\ 29^\circ & 48^\circ & 62^\circ & 78^\circ & 51^\circ \\ 46^\circ & 44^\circ & 65^\circ & 58^\circ & 36^\circ \\ 42^\circ & 32^\circ & 44^\circ & 45^\circ & 48^\circ \end{bmatrix}$$

Here P_{ij} is the gray level values of pixels in an image and $Q_{ij}'$is the respective angular values.Thus for representing the original image the value of pixels are 0 to 255 which requires 8 bit, but after applying the above algorithm, we need 7 bits for representation as it have the angular values from 0 to 90[11].

3.3 Reduction of bit rate of q_{ij}' image

For further compression of an image the output is again processed by divider block so that it will convert the image into 6 bit form for further compression, now for further compression of an image, divide the image by 1.5 because the range of 0 to 90 after division will become 0 to 60 this will reduce the bit rate of the image[12].

Thus we have; $Q_{ij}'' = \frac{Q_{ij}'}{1.5}$ .....(10)

$$Q_{ij}'' = \begin{bmatrix} 31^\circ & 36^\circ & 34^\circ & 60^\circ & 34^\circ \\ 34^\circ & 50^\circ & 29^\circ & 34^\circ & 41^\circ \\ 30^\circ & 21^\circ & 27^\circ & 34^\circ & 22^\circ \\ 30^\circ & 29^\circ & 43^\circ & 38^\circ & 24^\circ \\ 28^\circ & 21^\circ & 29^\circ & 30^\circ & 32^\circ \end{bmatrix}$$

This will results the maximum value of angle is 60, for which $2^6 = 64$ and hence we can represent it in 6 bits.

3.4 Bit plane slicing

In this approach, bit plane slicing is used for further compression of an image. Using one bit plane or two bit plane slicing the output is further compressed. A bit plane is a set of bits corresponding to a given bit position in each of the binary numbers in an image. It is used to determine the adequacy of numbers of bits used to quantize each pixel in the image [13].From figure 8, it is seen that instead of highlighting gray level images, it is desired to highlight the contribution made to total image appearance by specific bits. Suppose that each pixel in an image is represented by 8 bits[14]. Now assume that the image is having eight, 1-bit planes ranging from bit plane1-0 (LSB) to bit plane 7 (MSB). In Q_{ij}"it have 6 bit image in compressed form, now apply the bit plane slicing technique on Q_{ij}"image.

In Q_{ij}"(1,1) the gray level pixel with value 46(00101110) will be present in the 6th, 4th, 3rd and 2nd bit plane. After applying 2 bit plane slicing and 1 bit plane slicing on the binary number to get the image in 4 bit and 5 bit form as shown below.

46 ⟶ 101110 ⟶ 1011 (4 bit form)

46 ⟶ 101110 ⟶ 10111(5 bit form)

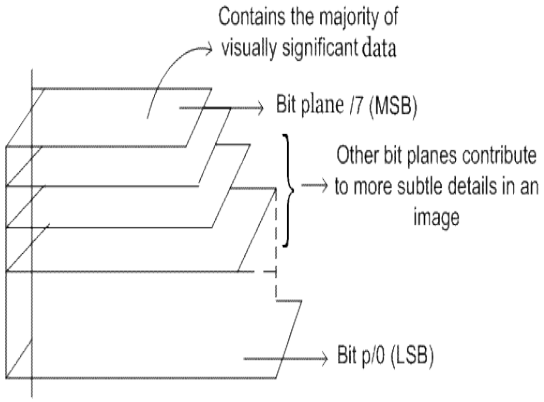

Fig.8: Bit Plane slicing for 8 bit image

Suppose the original 8 bit input image is of size 100kb. So after applying the 0 bit bit plane slicing on Q_{ij}" image the output image will be 2/8 of the original image i.e. this will able to compressed the size of image to 25%. For 1 bit bit plane slicing the output image will be 3/8 of the original image i.e. this will able to compress the image to 37.5%. For 2 bit bit plane slicing the output image will be 4/8 of the original image i.e. this will able to compress the image to 50%.Thus the proposed method is able to compressed an image upto 50 %[15].

4. MODELING RESULTS

In this work, MATLAB codes are developsfor simulation and verifying PSNR, MSE and compression ratio and compare the results with other existing methods such as DCT, Huffmans, RLE and Wavelet with different filters. Another set of experiment involves FPGA implementation for performing simulation and synthesis using Xilinx ISE for the verification of area, power and delay of the proposed method. The set of experiments evaluate the effect of differentmethods on the quality of the reconstructed image. Experiments were conducted using the standard data base such as images 'lena', 'Football' and Cutebaby. The performance measures for analysis used is mainly Mean square Error (MSE), Peak Signal to Noise Ratio (PSNR), Compression Ratio and Image Quality[16]. Where MSE is thecumulative squared error between the compressed and the original image. Whereas PSNR is a measure of the peak error. For evaluating mean square error and peak signal to noise ratio, following formulae were used.

$$MSE = \frac{1}{XY}\sum_{i=1}^{X}\cdot\sum_{j=1}^{Y}(Q_{ij} - Q_{ij}'')^2 \quad \text{.....(11)}$$

$$PSNR = 20^* \quad \log_{10}\left(\frac{255}{\sqrt{MSE}}\right)$$

(12)Where Q(i,j) is the original image and Q(i,j)" is the compressed version of the image and X,Y are the dimensions of given image, 255 is the peak signal value.

Figure9:(a) (b) (c) (d) (e) (f)

Fig. 9: (a) Original lena image (b) Reconstructed image using DCT (c) Reconstructed image using RLE (d) Reconstructed image using Huffmans (e) Reconstructed image using wavelet dB2 filter (f) Reconstructed image using proposed method.

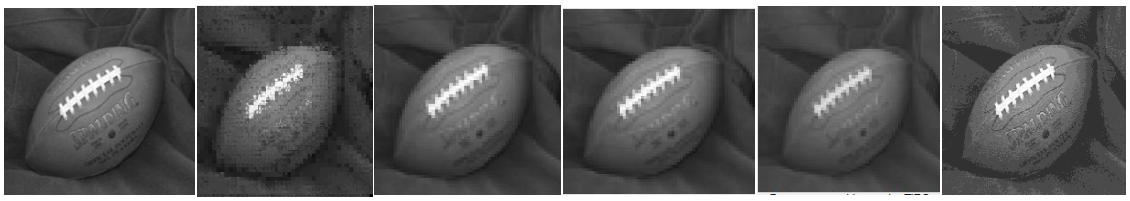

Figure10: (a) (b) (c) (d) (e) (f)

Fig. 10: (a) Original Football image (b-f) Reconstructed image using DCT, RLE, Huffmans, using wavelet dB2 filter and using proposed method.

Fig.11: (a) (b) (c) (d) (e) (f)

Fig. 11:(a) Original Cutebaby image (b) Reconstructed image using DCT (c) Reconstructed image using RLE (d)Reconstructed image using Huffumans (e) Reconstructed image using wavelet dB6 filter (f) Reconstructed image using proposed method.

Figure 12 shows the compressed images after VHDL simulation for the proposed architecture applied on different standard test images. After simulation the proposed method is synthesized using Xilinx ISE suite13.2 and tested it on Spartan-3 FPGA. Figure 13 shows the RTL (register transfer level) level diagram for proposed scheme and Table 2 shows the micro statistics of the proposed method. Figure 14 shows

Fig.12: Compressed images after VHDL simulation. Figure (a) represents compressed Football image (b) compressed cutebaby image (c) compressed Lena image

Table I: Comparison of results for different images using MATLAB

	Parameters	**DCT**	**RLE**	**Huffman**	**Wavelet (dB2/ dB6)**	**Proposed Method**
Image 1 LenaImage	MMSE (dB)	3.75	26.52	25.26	2.62	**1.29**
	PSNR (dB)	42.39	23.84	23.70	43.93	**45.89**
	Compression Ratio (%)	74.90	47.22	7.7	12.16	**50**
	Time Elapsed (s)	0.11	0.13	0.32	1.89	**0.085**
Image 2 Football	MMSE (dB)	1.04	37.32	39.90	2.07	**0.67**
	PSNR (dB)	47.98	27.96	27.92	44.96	**51.54**
	Compression Ratio (%)	64.12	48.79	16.98	48.72	**50**
	Time Elapsed (s)	0.55	0.25	0.43	0.79	**0.041**
Image 3 Cuteby Image	MMSE (dB)	1.35	40.21	38.32	12.73	**0.245**
	PSNR (dB)	46.82	20.11	20.82	37.08	**60.31**
	Compression Ratio (%)	72.70	48.76	15.54	28.55	**50.28**
	Time Elapsed (s)	0.58	0.19	0.45	2.54	**0.039**

Table2: Micro statistics of proposed method using Spartan 3 FPGA

Method / **Parameters**	**Proposed**	**DCT**
No.of ROM	1	2
Multiplier	1	1
Add/Sub	43	36
No.of Slices	66/768	75/768
Registers	16	24
Power(mW)	0.18	0.27
Delay(nSec)	6.216	13.714

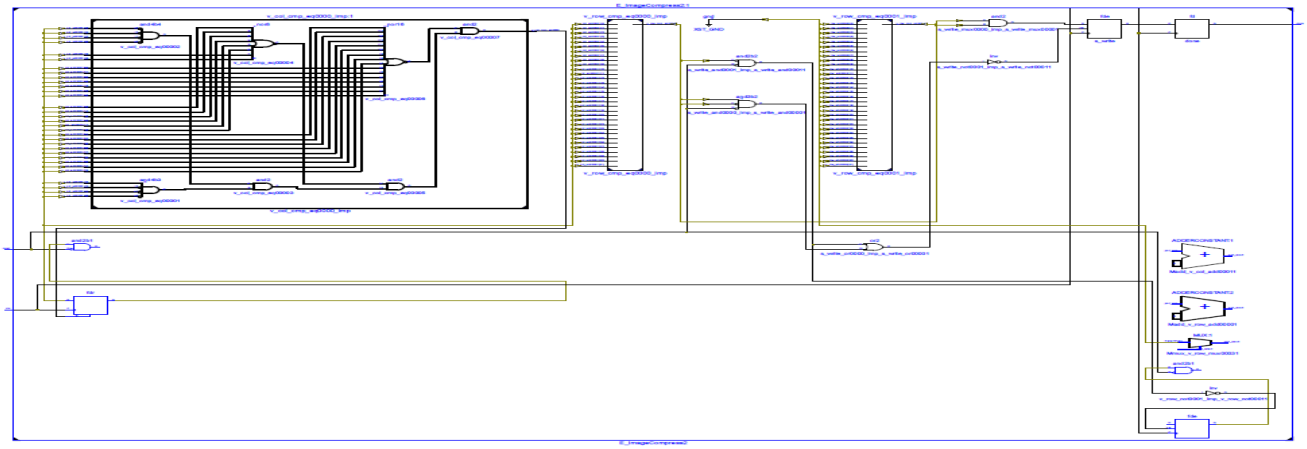

Fig.13: RTL level diagram for proposed method after synthesis using Xilinx ISE tool.

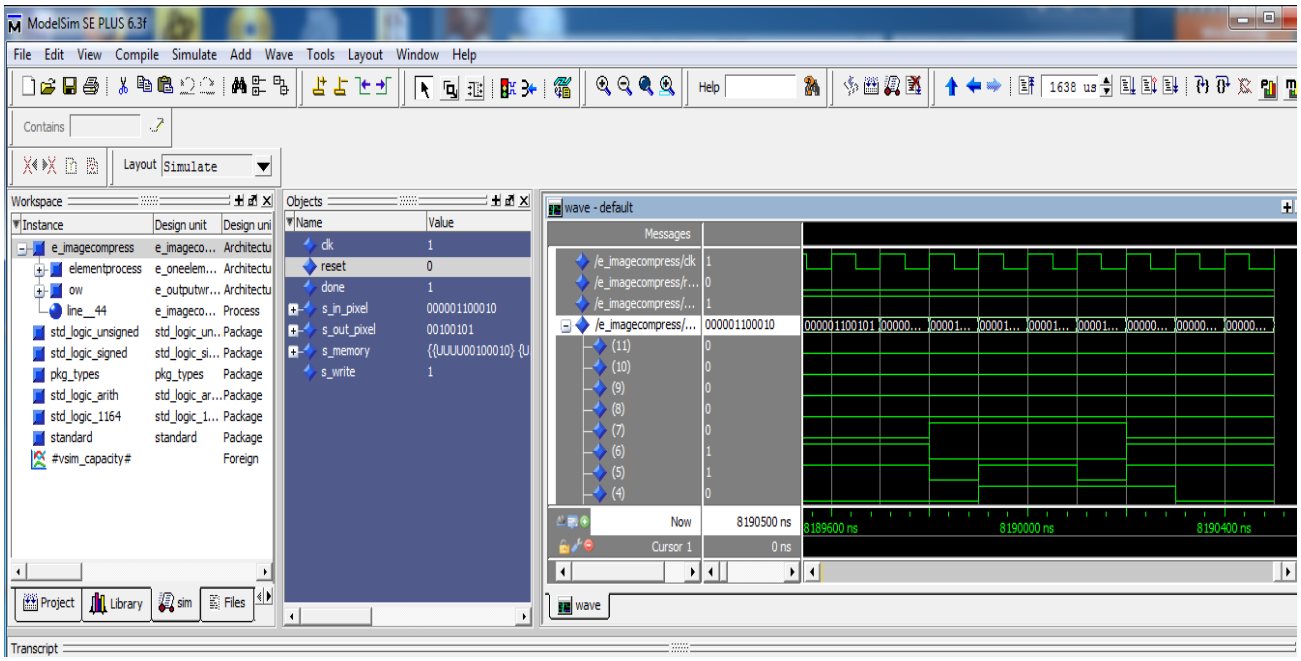

Fig.14: Simulation waveforms for proposed method

5. CONCLUSION

This paper presented a new method based on angular domain using CORDIC algorithm. In this work, MATLAB codes as well as VHDL codes were develops. In first set of experiment the algorithm is simulated for MMSE, PSNR and compression ratio using MATLAB. The results are compared with standard existing methods such as DCT, RLE, Huffmans and Wavelet with dB62 and dB6 filters. From the results it is observed that the proposed method works very efficiently and effectively and the results obtained are extremely good. The proposed method able to compress an image 50.00 % which is a very high compression rate. Also the visual quality of output image is intact and hence is exactly as same as input image. Moreover, the Peak Signal to Noise ratio (PSNR) value obtained is very high and is in between 50 dB to 60 dB and minimum mean square error (MMSE) value is very low and comes out to be 0.245 for image cutebaby. The other set of experiment shows the VHDL simulation and synthesis for analyzing area, power and delay for the proposed method. Table 2 summarizes the Micro statistics of proposed method using Spartan 3 FPGA the results of the proposed method were compared with standard DCT. It is seen that the propose method consumes very small power of 0.18 mWatts with only one ROM. Also the proposed method consumes only 9% of total area with small delay of about 6 nSec. Moreover, from the table 1 and table 2, it is clear that performance measures obtained are very good enough to show the high success rate of the research being done by means of this paper.

6. REFERENCES

[1] M. Tuceryan and A. K. Jain, “Texture analysis,” in The Handbook of Pattern Recognition and Computer Vision, C. H. Chen, L. F. Pau, and P. S. P. Wang, Eds., 2nd ed. Singapore: World Scientific, 1998, pp. 207–248.

[2] D.Malarvizhi, Dr.K.Kuppusamy, ”A new entropy encoding algorithm for image compression using DCT”, International Journal of Engineering Trends and Technology- Volume3Issue3- 2012.

[3] S. Rooij, P. Vitanyi, “Approximating Rate-Distortion Graphs of individual Data: Experiments in Lossy Compression and Denoising”, IEEE Transaction on Computers, vol.61, N° 3, March 2012, pp. 395-407.

[4] Harjeetpal singh, Sakhi Sharma, "Hybrid Image Compression Using DWT, DCT & Huffman Encoding Techniques",International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 10, October 2012.

[5] Shantanu D. Rane and Guillermo Sapiro, *Member, IEEE*,"Evaluation of JPEG-LS, the New Lossless and Controlled-Lossy Still Image Compression Standard, for Compression of Hi gh-Resolution Elevation Data ", IEEE Transactions on Geoscience and Remote sensing, VOL. 39, NO. 10, Oct. 2001

[6] Ali M. Reza, Robert D. Turneyi, "FPGA Implementation of 2D Wavelet Transform", IEEE Trans. On Pattern recognition-0-7803-5700-000,1999

[7] J.Holder,"The CORDIC trigonometric computing technique",IRE trans. Electron. Comput. Vol.No.8, pp-330-334, 1959

[8] S.S.Limaye,"VHDL-A design oriented approach", The Mc-Graw-Hill companies, 5th reprient 2011

[9] Chih-Hsiu Lin, "Mixed-scaling-rotation CORDIC (MSR-CORDIC) algorithm and architecture for high-performance vector rotational DSP applications", Regular Papers, IEEE Transactions on Circuits and Systems (Volume:52 , Issue: 11,2012)

[10] Loay E. George and Azhar M. Kadim, "Color Image Compression Using Fast VQ with DCT Based Block Indexing Method", ICIAR 2011, part II, LNCS6754, pp.253-263, Springer-Verlag Berlin Heidelberg 2011

[11] M. Tuceryan and A. K. Jain, "Texture analysis," in *The Handbook of Pattern Recognition and Computer Vision*, C. H. Chen, L. F. Pau, and P. S. P. Wang, Eds., 2nd ed. Singapore: World Scientific, 1998, pp. 207–248.

[12] Sung-Hsien Sun and Shie-Jue Lee, "A JPEG Chip for Image Compression and Decompression", Journal of VLSI Signal Processing 35, 43–60, 2003

[13] Mr. N S T Sai and R.C.Patil,"Image retrieval using Bit plane pixel distribution", International Journal of computer science and Information tech.,Vol 3, June-2011.

[14] Kadono, S, Tahara O and Okamoto N (2001) "Encoding ofcolor still pictures wavelet transform and vector quantization", *Canadian Conference on Electrical andComputer Engineering* 2:931–936.

[15] B. Krill, A.Ahmad, A.Amira, H.Rabah, "An efficient FPGA-based dynamic partial reconfiguration design flow and environment for image and signal processing IP cores", Signal Processing: Image Communication 25 (2010) 377–387 Elsevier.

[16] Kalaiyarasi .K, Deepika .S et al, "Fast DCT Computation Using Cordic Algorithm for Image Processing Application" International Journal for Research and Development in Engineering pp- 342-347-2014.