ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2014

FPGA Implementation of CORDIC Based DHT for Image Processing Applications

Shaik Waseem Ahmed[1], Sudhakara Reddy.P [2]

P.G. Student, Department of Electronics and Communication Engineering, SKIT College, Srikalahasti, India[1]

Associate Professor, Department of Electronics and Communication Engineering, SKIT College, Srikalahasti, India[2]

ABSTRACT: Digital image processing(DIP) is the use of computer algorithms to perform image processing on digital images. The basic operation performed by a simple digital camera is, to convert the light energy to electrical energy, then the energy is converted to digital format and a compression algorithm is used to reduce memory requirement for storing the image. This compression algorithm is frequently called for capturing and storing the images. This leads us to develop an efficient compression algorithm which will give the same result as that of the existing algorithms with low power consumption. Image compression is useful as it helps in reduction of the usage of expensive resources, such as memory, or the transmission bandwidth required. But on the downside, compression techniques result in distortion and also additional computational resources are required for compression-decompression of the medical image data.

In this, we proposed DHT based image compression technique. The computational complexity is further reduced by using pipelined CORDIC architecture for computation of *sin* and *cos* terms in DHT. The CORDIC based DHT is partially simulated and synthesized by using Xilinx ISE design suite and the same was implemented on targeted FPGA. The hardware requirements and gate delay values are observed and noted down.

KEYWORDS: CORDIC, DHT, FPGA, Image Processing.

I. INTRODUCTION

Compression, the art and science of reducing the amount of data required to represent an image, is one of the most useful and commercially successful technologies in the field of digital image processing. Digital image and video compression is now very essential. Bio-Medical Image Compression would not be feasible unless a high degree of compression is achieved. Compression is useful as it helps in reduction of the usage of expensive resources, such as memory (hard disks), or the transmission bandwidth required. In today's Age of competition where everything is reducing its size every minute, the smaller is the better. But on the downside, compression techniques result in distortion and also additional computational resources are required for compression-decompression of the data. Compression ratio (C) is defined as the ratio of the size of compressed data to that of the uncompressed data.

$$C = \frac{size\ of\ compressed\ \ data}{size\ of\ uncompressed\ \ data} \tag{1}$$

Redundancy is the reduction in size in comparison of the uncompressed size.

$$R = 1 - C \tag{2}$$

In image compression process the image $f(x,y)$ is mapped to a format to reduce spatial redundancy. The Discrete Hartley transform is used for mapping Next quantization is done, where the loss of information takes place. Since it is an irreversible process, we can omit this step for a lossless coding technique. The final step is symbol coding, where various coding techniques can be used to represent the information in minimum possible number of bits[1] This is shown in figure 1.

The discrete Hartley transform is a linear, invertible function H: $\boldsymbol{R} \rightarrow \boldsymbol{R}$ (where **R** denotes the set of real numbers). The N real numbers $x_0 x_1 \ldots x_{N-1}$ are transformed into N real numbers $H_0 H_1 \ldots H_{N-1}$ according to formula for $k = 0, 1 \ldots N-1$

$$H_K = \sum_{n=0}^{N-1} x_n \left(cos\tfrac{2\pi nk}{N} + sin\tfrac{2\pi nk}{N}\right) \qquad (3)$$

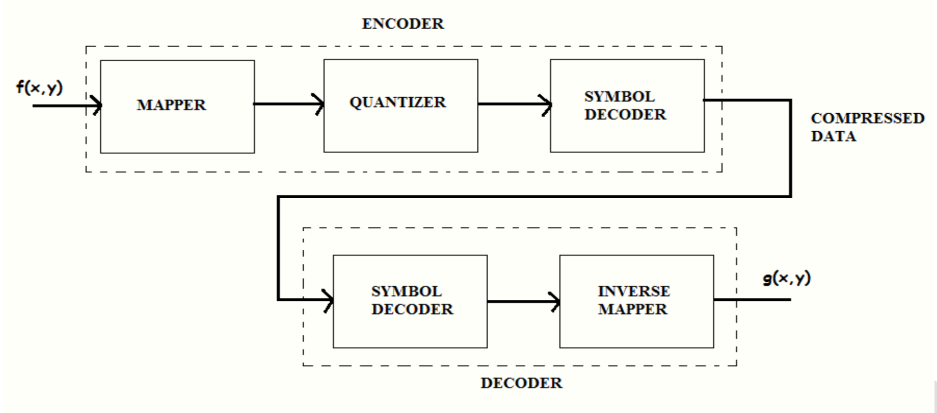


Fig.1. Functional block diagram of a general image compression system

II. IMAGE PROCESSING

Image processing involves minimizing the size in bytes of a graphics file without degrading the quality of the image to an unacceptable level. The reduction in file size allows more images to be stored in a given amount of disk or memory space. It also reduces the time and bandwidth required for images to be sent over the Internet or downloaded from Web pages.

There are several different ways in which image files can be compressed. For Internet use, the two most common compressed graphic image formats are the JPEG format and the GIF for mat. The JPEG method is more often used for photographs, while the GIF method is commonly used for line art and other images in which geometric shapes are relatively simple.

The steps involved in image compression are as follows:

1. First of all the image is divided into blocks of 8x8 pixel values. These blocks are then fed to the encoder from where we obtain the compressed image.
2. The next step is mapping of the pixel intensity value to another domain. The mapper transforms images into a (usually non- visual) format designed to reduce spatial and
3. Temporal redundancy. It can be done by applying various transforms to the images. Here discrete Hartley transform is applied to the 8x8 blocks.
4. Quantizing the transformed coefficients results in the loss of irrelevant information for the specified purpose.
5. Source coding is the process of encoding information using fewer bits (or other information- bearing units) than an unencoded representation would use, through use of specific encoding schemes.

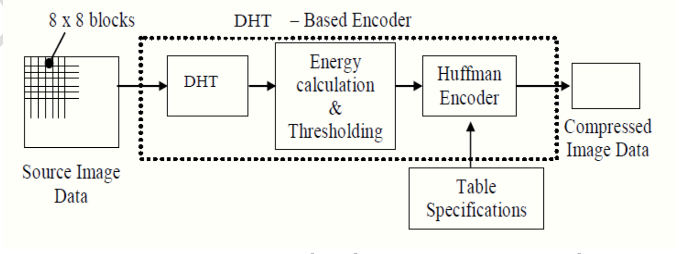


Fig .2. Energy quantization based image compression encoder

For retrieving the image back, the steps have to be reversed from the forward process. First the data is decoded using the decoder. Next inverse transform (IDHT) is calculated to get the 8x8 blocks. These blocks are then connected to form the final image. From the reconstructed image pixel values it is clear that some of the high frequency components are preserved. This indicates that the edge property of the image is preserved.

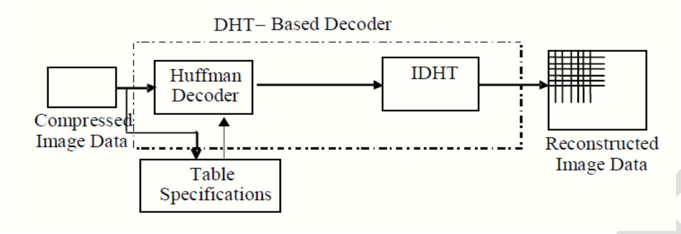


Fig .3.Energy quantization based image compression decoder

III. MATHEMATICAL REPRESENTATION OF DHT

The Hartley transform is an integral transform closely related to the Fourier transform, but which transforms real- valued functions to real-valued functions. It was proposed as an alternative to the Fourier transform by R. V. L. Hartley in 1942. Compared to the Fourier transform, the Hartley transform has the advantages of transforming real functions to real functions and of being its own inverse. The discrete version of the transform, the Discrete Hartley transform, was introduced by R. N. Bracewell in 1983. Discrete Hartley Transform is the real valued transform which gives only real transform coefficients for real input stream. It has the main advantage over DCT of reducing the memory content up to 50% since the inverse transform is identical to the forward transform. Also, it retains the higher frequency components, which restores the detailing of the image. Since it is a real valued function unlike DFT, the computational complexities are also lower than in DFT algorithms [2].

Formally, the discrete Hartley transform is a linear, invertible function $H: \boldsymbol{R} \rightarrow \boldsymbol{R}$ (where $\boldsymbol{R}$ denotes the set of real numbers). The N real numbers $x_0 x_1 \ldots x_{N-1}$ are transformed into N real numbers $H_0 H_1 \ldots H_{N-1}$ according to the formula: for $k = 0, 1 \ldots N-1$

$$H_K = \sum_{n=0}^{N-1} x_n \left(cos\frac{2\pi nk}{N} + sin\frac{2\pi nk}{N}\right) \tag{4}$$

The inverse transform is given by: for $n = 0,1 \ldots N-1$

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} H_n \left(cos\frac{2\pi nk}{N} + sin\frac{2\pi nk}{N}\right) \tag{5}$$

The *cas* function is given by:

$$cas\frac{2\pi nk}{N} = cos\frac{2\pi nk}{N} + sin\frac{2\pi nk}{N} \tag{6}$$

and one of the properties of *cas* function is:

$$2cas(a+b) = cas(a)cas(b) + cas(-a)cas(b) + cas(a)cas(-b) - cas(-a)cas(-b) \tag{7}$$

2 –Dimensional DHT of an array $x\,(m,n)$ of size MxN may be defined as:

$$\mathrm{X(k,l)} = \sum_{\mathrm{m=0}}^{\mathrm{M}} \sum_{\mathrm{n=0}}^{\mathrm{N}} \mathrm{x(m,n)cas}\left(\frac{2\pi \mathrm{mk}}{\mathrm{m}} + \frac{2\pi \mathrm{nl}}{\mathrm{n}}\right) \tag{8}$$

For $k = 0,1 \ldots M-1$ & $1 = 0,1 \ldots N-1$

The inverse transform is given by the same formula along with a scaling factor of $1/MN$ i.e.

$$X(k,l) = \frac{1}{MN}\sum_{m=0}^{M} \sum_{n=0}^{N} x(m,n)cas\left(\frac{2\pi mk}{m} + \frac{2\pi nl}{n}\right) \tag{9}$$

for $k = 0, 1 \ldots M - 1$ & $l = 0,1 \ldots\ldots N - 1$

The real and imaginary parts of the Fourier transform are given by the even and odd parts of the Hartley transform, respectively

$$F(W) = \frac{H(w)+H(-w)}{2} - \frac{i(H(w)-H(-w))}{2} \tag{10}$$

There is also an analogue of the convolution theorem for the Hartley transform. If two functions $x\,(\,t\,)$ and $y\,(\,t\,)$ have Hartley transforms$X\,(\,)$ *and* $Y\,(\,)$, respectively, then their convolution $z\,(\,t\,) = x * y$ has the Hartley transform:

$$Z(w) = \{H(x*y)\} = \frac{\sqrt{2pi}\,(X(w)[Y(w)+Y(-w)]+X(-w[Y(w)-Y(-w)])}{2} \tag{11}$$

Similar to the Fourier transform, the Hartley transform of an even/odd function is even/odd, respectively.

IV. 8-POINT DHT SIGNAL FLOW

The signal-flow graph for 8- point DHT with pipelined stages with delays is shown in figure.4. The DHT of a real-valued-point input vector$x_0 x_1 \dots x_{N-1}$, may be defined as

$$X_k = \sum_{n=0}^{N-1} x_n C_N(k,n) \quad (12)$$

Where

$$C_N(k,n) = cas\left(\frac{2\pi nk}{N}\right) = cos\frac{2\pi nk}{N} + sin\frac{2\pi nk}{N} \quad (13)$$

for $k,n = 0,1 \dots . N-1$, Supposing N to be an even number, the sequence x_n is divided into two sub-sequences x_1 and x_2 length $N/2$ each, such that $x_1 = \{x_0, x_2 \dots . x_{N-2}\}$ contains all even-indexed terms and $x_2 = \{x_1, x_3 \dots . x_{N-1}\}$ contains all odd-indexed terms of the input sequence x .
Then the DHT can be defined as:

$$X_k = \sum_{n=0}^{\frac{N}{2}-1} x_1 C_N(k,2n) + \sum_{n=0}^{\frac{N}{2}-1} x_2 C_N(k,2n+1) \quad (14)$$

Let X_{1k} and X_{2k} represent the $(N/2)$ –point DHT coefficients of sequences x_1 and x_2 of length (N/2) respectively. Using the symmetry properties of sine and cosine functions, the N-point DHT may be expressed as the following set of equations:

$$X_k = X_{1k} + E_k \quad (15)$$

$$X_{m+k} = X_{1k} - E_k \quad (16)$$

$$E_k = X_{2k} \cos\left(\frac{\pi k}{M}\right) + X_{2(M-k)} sin(\frac{\pi k}{M}) \quad (17)$$

For $k = 1,2 \dots M-1$ where $M = \frac{N}{2}$.Hence for computing 8-point DHT from 4-point DHT the set of equations are shown in equations 18 to 25, which are obtained from equation 17

1. $X_0 = X_{10} + X_{20}$ (18)
2. $X_1 = X_{11} + (X_{21} + X_{23})/\sqrt{2}$ (19)
3. $X_2 = X_{12} + X_{22}$ (20)
4. $X_3 = X_{13} + (X_{21} - X_{23})/\sqrt{2}$ (21)
5. $X_4 = X_{10} - X_{20}$ (22)
6. $X_5 = X_{11} - (X_{21} + X_{23})/\sqrt{2}$ (23)
7. $X_6 = X_{12} - X_{22}$ (24)
8. $X_7 = X_{13} - (X_{21} - X_{23})/\sqrt{2}$ (25)

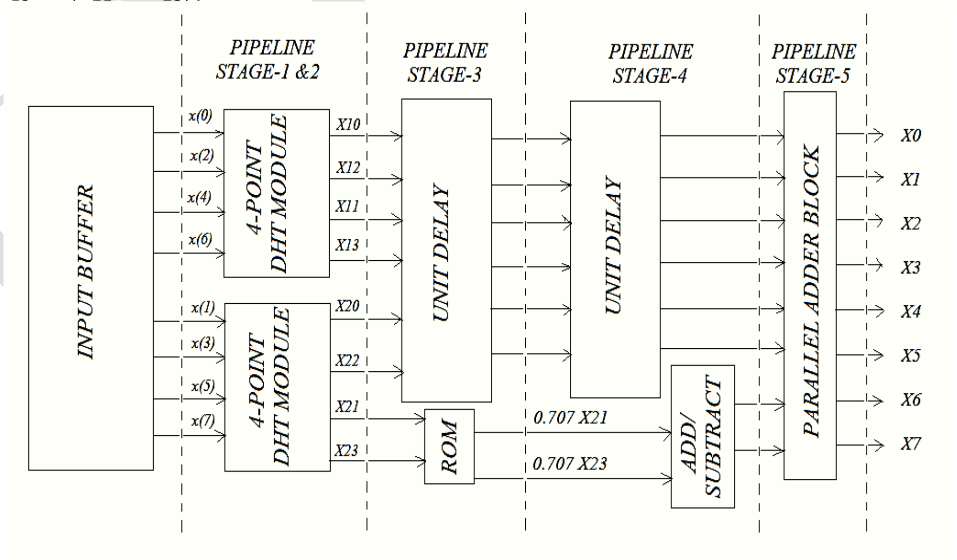


Fig.4. Flow chart of the 8-point DHT in pipelined approach with delays

So, for computing 8-point DHT the multiplication with $1/\sqrt{2}$ can be read from a ROM, while a block of pipelined

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2014

adders perform the addition. It computes DHT in 5 pipelined stages. For first two stages, it consists of two 4-pointDHT modules that receive the odd and even indexed subsequence x_1 and x_2 and from the input buffer. In the third pipelined stage, multiplication with $1/\sqrt{2}$ is done for the required coefficients i.e. X_{21} *and* X_{23}. Next they are added and subtracted in the fourth stage. During 3^{rd} and 4^{th} stages the rest of the coefficients are passed through a delay. Delay consists of simply registers i.e. they are stored in different registers and passed to the next stage. Finally the fifth pipelined stage is a parallel adder block which adds/subtracts the coefficients to give the desire output.[3] The block diagram of the described method is given in figure 2.

V. CORDIC BASED DHT

The CORDIC means Coordinate Rotation Digital Computer. CORDIC use simple shift and add operations for several computing tasks. It is generally faster than the other approaches when no hardware multiplier is available. In recent years, the CORDIC algorithms have been in use extensively for various applications, especially in FPGA implementation. The CORDIC provide an iterative solutions to perform vector rotations by arbitrary angles using only shifts and adds. The CORDIC algorithm can be operated in either **vectoring mode** or **rotation** mode.[4] Generalization of the CORDIC algorithm: The generalized CORDIC is formulated as follows

$$x_{i+1} = x_i - m\sigma_i . 2^{-i} . y_i \tag{26}$$
$$y_{i+1} = y_i + \sigma_i . 2^{-i} . x_i \tag{27}$$
$$z_{i+1} = z_i - \sigma_i . \alpha_i \tag{28}$$

Where

$$\sigma_i = \begin{cases} sign(z_i) & for\ rotation\ mode \\ -sign(y_i) & for\ vectoring\ mode \end{cases} \tag{29}$$

For m= 1,0 or -1 which is suitable to perform rotations in circular , hyperbolic & linear coordinate system and α represents the angle. We have proposed an 8-point CORDIC based DHT with six CORDIC rotations to realize multiplier less approximation. They are circular rotation mode, circular vectoring mode, linear rotation mode, linear vectoring mode, hyperbolic rotation mode and hyperbolic vectoring mode

The CORDIC is hardware-efficient algorithms for computation of trigonometric and other elementary functions that use only shift and add to perform. The CORDIC set of algorithms for the computation of trigonometric functions was designed by Jack E. Volder in 1959 Later, J. Walther in 1971 extended the CORDIC scheme to other functions. Depending on the configuration, the resulting module implements pipelined parallel-pipelined, word-serial, or bit-serial architecture in one of two modes: rotation or vectoring. In rotation mode, the CORDIC rotates a vector by a certain angle. This mode is used to convert polar to Cartesian coordinates. For eg consider the multiplication of two complex numbers $x + jy$ and $(cos(\theta) + j\ sin(\theta))$ .The result $u + jv$, can be obtained by calculating the final coordinate after rotating a 2x2 vector [x y]T through an angle (θ) and then scaled by a factor r. This is achieved in CORDIC via a three-stair procedure: angle conversion, Vector rotation and scaling. The radix 2 system is taken because it avoids the use of multiplications while implementing the above equation. Hence a CORDIC iteration can be realized using shifters and adders only.

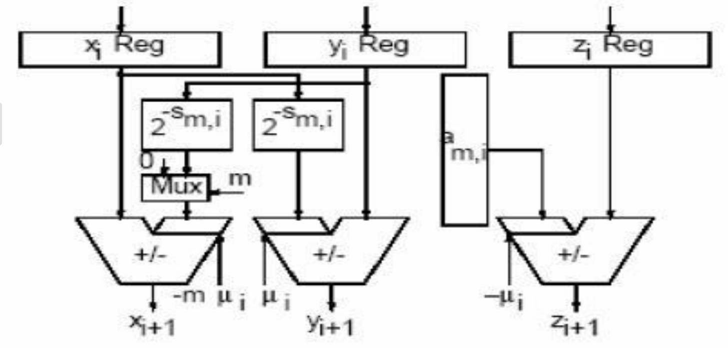


Fig.5. Structure of a processing element for one CORDIC iteration

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2014

The figure 5 shows the structure of a processing element which implements one CORDIC iteration the rotation mode and vectoring mode are two schemes for the CORDIC algorithm. In rotation mode, the aim is to rotate the given input vector $(x\,,y)t$ with a given angle. After n no's of iterations, Z_n is driven to zero and the total accumulated rotation angle is equal to desired angle Parallel pipelined architecture for CORDIC represents a version of the sequential CORDIC algorithm. Instead of reusing the same hardware for all iteration stages, the parallel architecture provides a separate processor for every iteration.

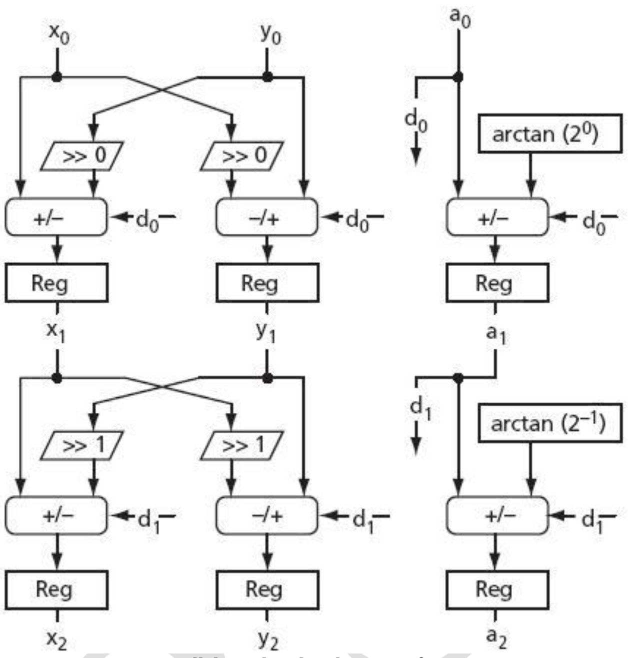


Fig.6. Parallel pipelined architecture for CORDIC

An example of the parallel CORDIC architecture for rotation mode is shown in figure 6. Each of the n processors present in the block performs a specific iteration, and a particular processor always performs the same iteration. All the shifters perform the fixed shift, so that it can be implemented in FPGA.[5] Every processor utilizes a individual arc tan value that can also be hardwired to the input of every angle accumulator in the absence of a state machine which provides simplicity to this type of architecture. The parallel architecture is much faster than the sequential architecture described in the “iterative Word-serial architecture” in figure 6. It takes new input data and puts out the results at every clock cycle, introducing a latency of n clock cycles. The architecture which is used in the design of the DHT is this parallel-pipelined architecture because this architecture which provides high throughput and low power consumption.
Then we can apply the above conditions to the DHT equation.
The DHT is given

$$H_K = \sum_{n=0}^{N-1} x_n\,(cos^{\frac{2\Pi nk}{N}} + sin^{\frac{2\Pi nk}{N}}),\ k = 0,1 \ldots N-1 \tag{30}$$

In this equation we have we have one multiplication and adder.

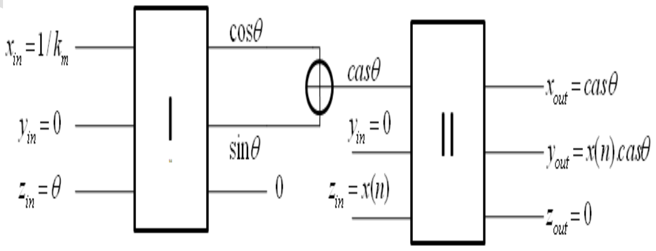


Fig.7. $cas\theta$ generation using circular and linear mode

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2014

The equation (30) can be implemented by CORDIC as follows. In order to compute the $cas\theta$ term makes the initial condition $x_{in} = 1/k_m$, $y_{in} = 0$ and $z_{in} = \theta$. In this block diagram apply the above condition we get the $Cas\theta$ term as shown in figure 7.
$x[n]$ Multiplied with θ , and its summation according to above equation are shown figure 8.

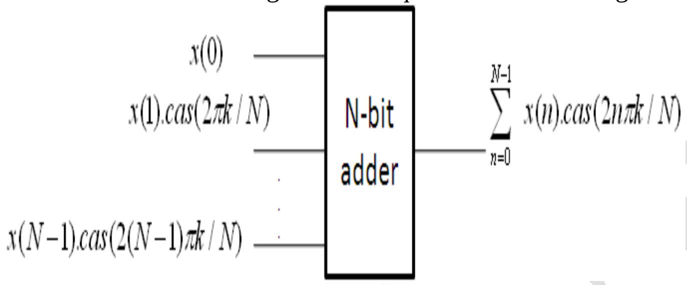

$x(0)$
$x(1).cas(2\pi k/N)$
$x(N-1).cas(2(N-1)\pi k/N)$
N-bit adder
$\sum_{n=0}^{N-1} x(n).cas(2n\pi k/N)$

Fig.8. Block diagram of a N-bit adder

VI. RESULTS ANALYSIS AND IMPLEMENTATION

The real and imaginary parts of the Fourier transform are given by the even and odd parts of the Hartley transform, respectively

$$F(W) = \frac{H(w)+H(-w)}{2} - \frac{i(H(w)-H(-w))}{2} \tag{31}$$

There is also an analogue of the convolution theorem for the Hartley transform. If two functions $x(t)$ and $y(t)$ have Hartley transforms $X()$ and $Y()$, respectively, then their convolution $z(t) = x * y$ has the Hartley transform:

$$Z(w) = \{H(x*y)\} = \frac{\sqrt{2pi}\,(X(w)[Y(w)+Y(-w)]+X(-w[Y(w)-Y(-w)])}{2} \tag{32}$$

Similar to the Fourier transform, the Hartley transform of an even/odd function is even/odd, respectively. Hence DHT is a better option for compression algorithms and is used for mapping the input image pixels and quantization Normally the performance of a data compression scheme can be measured in terms of three parameters. These are Compression efficiency, Complexity and Distortion measurement for lossy compression. The Mean square error for a 1-D DHT data is given by:

$$MSE = \frac{1}{N}\sum_{n=0}^{N-1} |x(n) - x'(n)|^2 \tag{33}$$

Where N is the number of pixels in the image, $x(n)$ is the original data and $x'(n)$ is the compressed data. The Peak Signal to Noise ratio (PSNR) is given by:

$$PSN10log_{10}\frac{255^2}{MSE'} \tag{34}$$

Where MSE is calculated for 2-D block as:

$$MSE' = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} |x(m,n) - x'(m,n)|^2 \tag{35}$$

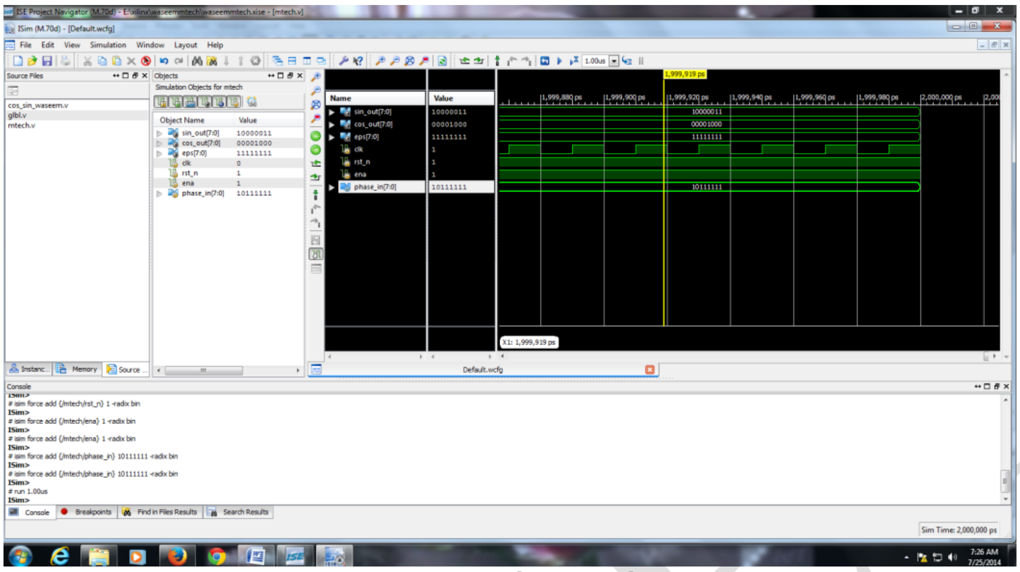
Initially the DHT was decomposed in terms of COS and SINE terms by using Euler's formula, then for the computation of these trigonometric components we use CORDIC processor. For hardware implementation, we developed Verilog code and compiled using ModelSim software. Further simulated and synthesized by using Xilinx ISE design suite version 12.0 and implemented on Spartan 6.0 FPGA. Finally synthesis report and delay report are noted down. From the results it is observed that, the total real time taken for execution is 1.00secs, the total CPU time taken for execution is 0.94 sec. The macro statistics of CORDIC requires only single ROM, one 4x8-bit ROM, 20-adders and subtractions, three 8-bit adders, one 8-bit subtraction, 32 registers, eight 2-bit registers, 24 8-bit registers and 2 – multiplexers. The simulation results are as shown in figure 9.

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2014



Object Name	Value
sin_out[7:0]	10000011
cos_out[7:0]	00001000
eps[7:0]	11111111
clk	0
rst_n	1
ena	1
phase_in[7:0]	10111111

```
ISim>
# isim force add {/mtech/rst_n} 1 -radix bin
ISim>
# isim force add {/mtech/ena} 1 -radix bin
ISim>
# isim force add {/mtech/ena} 1 -radix bin
ISim>
# isim force add {/mtech/phase_in} 10111111 -radix bin
ISim>
# isim force add {/mtech/phase_in} 10111111 -radix bin
ISim>
# run 1.00us
ISim>
```

Fig 9 Simulation results

VII. CONCLUSION

In the present work, Discrete Hartley Transform for input matrix was implemented in FPGA using VHDL as the synthesis tool. The DHT was also calculated for 8-point input using two algorithms and their effectiveness were discussed, this primarily focuses on image compression with less computation and low power. The simulation results and design summary of DHT was obtained and it was shown that the architecture implemented is an efficient method which uses limited space and time. The hardware utilization is quite optimum and power analysis shows that the power requirement is also optimum. However if the input contents are large, they tend to overflow from the registers and hence error occurs. It can be rectified by saving the transformed coefficients in larger registers. Also due to quantization in the contents of the ROM, even-number outputs are more deviated from the desired results than the odd-numbered outputs.

REFERENCES

[1] S.K.Pattanaik and K.K.Mahapatra, "DHT Based JPEG Image Compression Using a Novel Energy Quantization Method" *IEEE International conference on industrial technology*, pp. 2827 – 2832, Dec 2006.
[2] RN.Bracewell, 0.Buneman, H. Hao and **J.** Villasenor, "Fast two-dimensional hartley transform*" Proceedings of IEEE,* Vol 74, No. 9, Sept1986.
[3] C. H. Paik and M. D. Fox, "Fast Hartley transform for image processing," *IEEE Trans. Med. Image, vol. 7* , no. 6, pp. 149–153, Jun. 1988
[4] P.K.Meher, S. Thambipillai and J.C. Patra, "Scalable and modular memory-based systolic architectures for discrete hartley transform" *IEEE Transactions on circuits and systems-I: regular papers, Vol53*, pp. 1065-1077, May 2006
[5] A. Amira, "An FPGA based system for discrete hartley transforms. "*IEEE publication*, pp. 137-140, 2003
[6] R.C. Gonzalez, R.E. Woods, Digital Image Processing, Pearson Education 3rd Edition 2008
[7] CR. Baugh and BA. Wooley, "A two's complement parallel array multiplication algorithm" *IEEE Transactions on computers,* Vol C-22, pp. 1045-1047, Dec 1973
[8] Bracewell, Ronald N. "The Hartley transform" *New York: Oxford university press* 1986
[9] Ranjan Bose, Information theory coding and Cryptography, Tata McGraw-Hill 2003.
[10] F.Vahid and T. Givargis, Embedded system design: A unified hardware/software introduction, Wiley India (P.) Ltd, 3 rd edition 2009.
[11] H.S. Hou, "The fast Hartley transform algorithm," *IEEE Transactions on Computers*, vol. C-36, no. 2, pp. 147–156, Feb. 1987.
[12] L.W. Chang and S.W. Lee, "Systolic arrays for the discrete Hartley transform," *IEEE Transactions Signal Processing*, vol. 39, no. 11, pp. 2411–2418, Nov. 1991.
[13] Miodrag Popović and Dragutin Šević, "A new look at the comparison of the fast Hartley and Fourier transforms," *IEEE Trans. Signal Processing* 42 (8), 2178-2182 (1994)