Minimum Path Delay for Discrete Hartley Transform using XOR-XNOR Technique

Ritu Singh
M. Tech. Scholar
Dept. of Electronics and Communication
LNCT, Bhopal

Prashant Chaturvedi
Professor
Dept. of Electronics and Communication
LNCT, Bhopal

Abstract— A discrete Hartley transform (DHT) algorithm can be efficiently implemented on a highly modular and parallel architecture having a regular structure is consisting of multiplier and adder. Discrete Hartley Transform (DHT) is one of the transform used for converting data in time domain into frequency domain using only real values. We have proposed a new algorithm for calculating DHT of length 2^N, where N=3 and 4. We have implemented Urdhwa multiplier based on compressor as an improvement in place of simple multiplication used in conventional DHT. This paper gives a comparison between conventional DHT algorithm and proposed DHT algorithm in terms of delays and area.

Keywords— *Discrete Hartley Transform (DHT), Urdhwa Multiplier, 4:2 Compressor, 7:2 compressor and Xilinx Vertex family.*

I. INTRODUCTION

Digital signal processing (DSP) includes processing of data in various domains based on their applications.DSP has vast applications in various fields such as space, medical, commercial, industrial and scientific. Each requires processing of vast data for collecting useful information. Transform is a technique used in DSP for converting one form of data in another. A family of transform is available in DSP for data processing .Fourier analysis one of the oldest technique used in this family. Fourier analysis is named after Jean baptiste joseph Fourier (1768-1830) a French mathematician and physicist. It was used for periodic continuous signals. Fourier series is a technique which decomposes a signal in time domain into a no. of sine and cosine waves in frequency domain. But it was not applicable for non-periodic signals Then came Fourier transform into existence which removes the drawback of Fourier series and thus can be used for non-periodic continuous signals. Fourier transform is a mathematical tool using integrals. But Fourier transform is not suitable for non-stationary signals. Since both transforms are not applicable for discrete signals, so there is a need for new transform for discrete signals. Discrete time Fourier transform (DTFT) is used for signals that extend from positive to negative infinity but are not periodic. DTFT is not used for periodic discrete signals so discrete Fourier transform (DFT) can into existence. DFT is a discrete numerical equivalent of FT using summation instead of integrals. DFT is used for signals that repeat themselves in periodic fashion extending from positive to negative infinity. FFT is improvement of DFT in which computation has becomes faster. All the family members of Fourier till now works on complex values which requires large storage space and computationally complex in nature. So, now comes a new member of transform called Discrete Hartley transform (DHT) which converts real values into real values. Therefore, it needs lesser storage space and less computational complexity.

In this paper, DHT is computed using simple addition and multiplication is done by using Urdhwa multiplier uses different 4:2 and 7:2 compressors along with half adder, full adder, XOR Gate and XNOR Gate for complete multiplication process.

II. URDHWA MULTIPLIER

Multipliers are the blocks in DSP processors which requires large amount of storage and processing complexity. Vedic mathematics has well known an ancient method for mathematical operations which are easy to use Urdhwa triyambakam is a method used for multiplication calculations. It takes the advantage of vertical and crosswise calculations. In this technique multiplication of terms is obtained by using simple AND logical operations, full adders and half adders.

In this paper a 8 bit Urdhwa multiplier is used. For this purpose of addition, we have been used various designs of adders and compressors. Here, these adders and compressors are deployed in such a way so as to provide optimized design of multiplier in terms of delay. Four half adders, two full adders, five 7: 2 compressor and ten 4:2 compressors has been used.

We have presented three designs for Urdhwa multiplier. These three designs vary in terms of 4:2 compressor designs. First design uses the simple 4:2 compressor which uses two full adders. Second design uses four XOR gates and two multiplexers (2×1). Third design of 4:2 compressors is two XOR-XNOR gates and four multiplexers (2×1). First design of 4:2 compressors is simplest one, but more delay. Next design has lesser delay as compared to the first design at the cost of higher complexity. But the last which is our proposed design has least amount of delay but has greatest complexity and occupies maximum space among all of them.

We have designed 7:2 compressor using different designs of 4:2 compressor thus we got three variations of 7:2 compressor.

- **4:2 Compressor**

To add binary numbers with minimal carry propagation we use compressor adder instead of other adder. Compressor is a digital modern circuit which is used for high speed with minimum gates requires designing technique. This compressor becomes the essential tool for fast multiplication adding technique by keeping an eye on fast processor and lesser area.
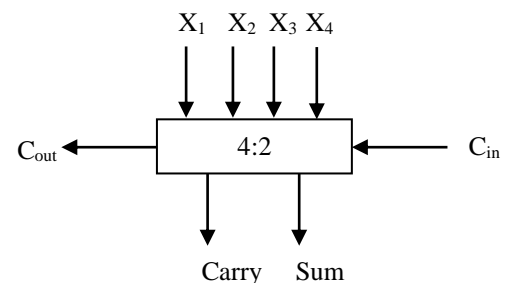

Figure 1: Block Diagram of 4:2 Compressors

4:2 compressors are capable of adding 4 bits and one carry, in turn producing a 3 bit output. The 4-2 compressor has 4 inputs X_1, X_2, X_3 and X_4 and 2 outputs Sum and Carry along with a Carry-in (Cin) and a Carry-out (Cout) as shown in Figure 1. The input Cin is the output from the previous lower significant compressor.

The Cout is the output to the compressor in the next significant stage. The critical path is smaller in comparison with an equivalent circuit to add 5 bits using full adders and half adders. Similar to the 3-2 compressor the 4-2 compressor is governed by the basic equation the standard implementation of the 4-2 compressor is done using 2 Full Adder cells as shown in Figure 2(a). When the individual full Adders are broken into their constituent XOR blocks, it can be observed that the overall delay is equal to 4*XOR.

$$X_1 + X_2 + X_3 + X_4 + C_{in} = sum + 2*(Carry + C_{out})$$

$$Sum = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{in}$$

$$C_{out} = (X_1 \oplus X_2).X_3 + (X_1 \oplus X_2).X_1$$

$$C_{arry} = U.C_{in} + \overline{(X_1 \oplus X_2 \oplus X_3 \oplus X_4)}.X_4$$

Where

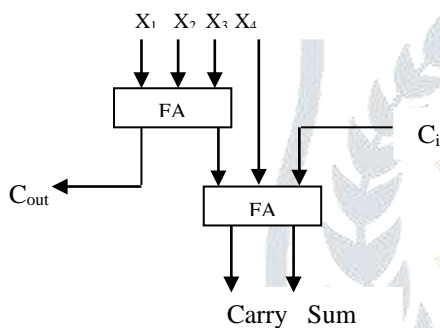$$U = X_1 \oplus X_2 \oplus X_3 \oplus X_4$$

Figure 2(a): 4:2 Compressors using Full Adder

The standard implementation of the 4-2 compressor is done using 2 Full Adder cells as shown in Figure 2(a). When the individual full Adders are broken into their constituent XOR blocks, it can be observed that the overall delay is equal to 4*XOR. The block diagram in Figure 2(b) shows the existing architecture for the implementation of the 4-2 compressor with a delay of 3*XOR. The equations governing the outputs in the existing architecture are shown below
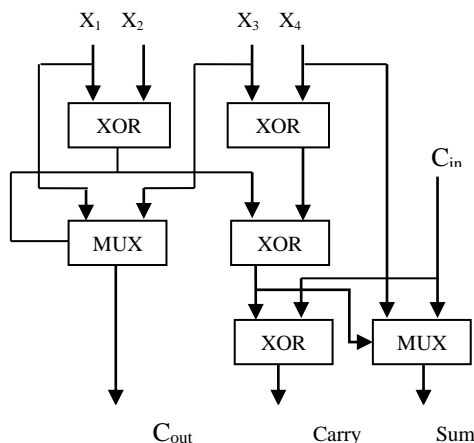

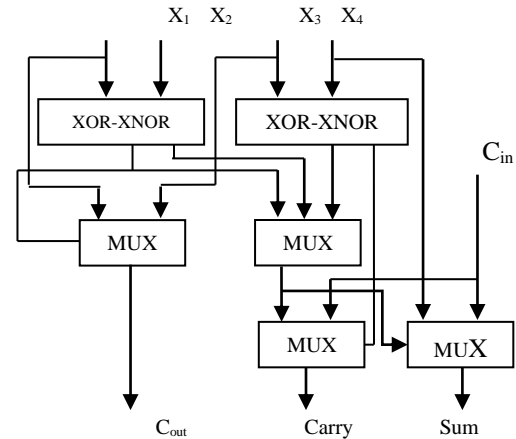Figure 2(b): 4:2 Compressors using XOR Gate

Figure 2(c): 4:2 Compressors using XOR-XNOR Gate

Thus replacing some XOR blocks with multiplexer's results in a significant improvement in delay. Also the MUX block at the SUM output gets the select bit before the inputs arrive and thus the transistors are already switched by the time they arrive. This minimizes the delay to a considerable extent. This is shown in Figure 2(c).

The equations governing the outputs in the proposed architecture are shown below

$$Sum = (X_1 \oplus X_2).X_3 + (X_1 \oplus X_2).(X_3 \oplus X_1).C_{in}$$

$$C_{out} = (X_1 \oplus X_2).X_3 + (X_1 \oplus X_2).X_1$$

All the designs have been captured by VHDL and the functionality is verified by RTL and gate level simulation shown in table 1. Designs are implemented on a Xilinx Spartan 3 FPGA using VHDL as the RTL language with the help of Xilinx ISE design suite 14.1.

TABLE 1: Device utilization summary (Vertex-4) of 4:2 Compressors using full adder, 4:2 Compressor using XOR Gate and 4:2 Compressor using XOR-XNOR Gate

Design	No. of Slices	No. of 4 input LUTs	MCPD (ns)
4:2 Compressor	3	6	6.257
4:2 Compressor using XOR gate	2	4	5.663
4:2 Compressor using XOR-XNOR gate	2	3	5.280

- **7:2 Compressor**

Similar to its 4:2 compressor counterpart, the 7:2 compressors as shown in Figure 3, is capable of adding 7 bits of input and 2 carry's from the previous stages, at a time. In our implementation, we have designed a novel 7:2 compressor utilizing two 4:2 compressors, two full adders and one half adders. We have designed 7:2 compressor using different designs of 4:2 compressor thus we got three variations of 7:2 compressor.
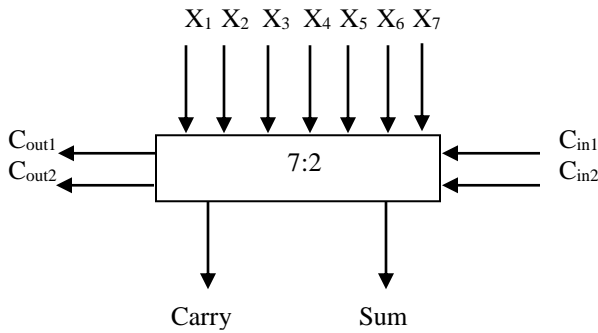
$$X_1 \ X_2 \ X_3 \ X_4 \ X_5 \ X_6 \ X_7$$

Figure 3: Block Diagram of 7:2 Compressors

III. DISCRETE HARTLEY TRANSFORM

Discrete Hartley Transform is abbreviated for DHT and this transform was proposed by R. V. L. Hartley in 1942. DHT is the analogous to Fast Fourier transform which provides the only real value at any cost. The main difference from the DFT is that it transforms the real inputs to real outputs with no intrinsic involvement of complex value. DFT can be used to compute the DHT, and vice versa.

Figure 4: Block Diagram of DHT

In other words, Discrete Hartley transform is used to convert real values into real ones. It requires decomposition of data into stages using butterfly similar to FFT. But the butterfly used in DHT is quite different in terms of coefficients or multipliers. With the increase in number of DHT sequence length the number of coefficients is also increased simultaneously.

Let $N \geq 4$ be a power of two. For any real input sequence $\{x(i) : i = 0,1,2,\ldots\ldots\ldots\ldots N-1\}$

$$X(k) = DHT(N)\{x((i)\}$$

$$= \sum_{i=0}^{N-1} x(i).cas[2ki\pi / N] \qquad for\, k = 0,1\ldots\ldots N-1$$

Where $cas(x) = \cos(x) + \sin(x)$

- ALGORITHM FOR 16 POINT DHT-

We present an implementation of fast DHT algorithm for a length N=1. There are six stages required to complete the butterfly design of N=16 length DHT. These stages include summing stages and coefficient multiplying stages. Before first stage the data sequence are arranged in bit reversed pattern by using any method like permutation. Then in the first stage the pairs of bit reversed patterns are added to form eight terms. In the second stage, one third of the terms are again added and subtracted to form further three terms.

First two stages do not include any multiplication. Remaining terms are multiplied by the first coefficient. In the next stage again two new coefficients are introduced which is multiplied by the lower half of the third stage. In each stage multiplying of coefficients stage precedes its summing stage. After coefficient multiplication it is preceded by its summing stage to form the common terms used in the final stage. Last stage includes only summing of terms. Finally we get the transformed data sequence in order and do not need any permutation.
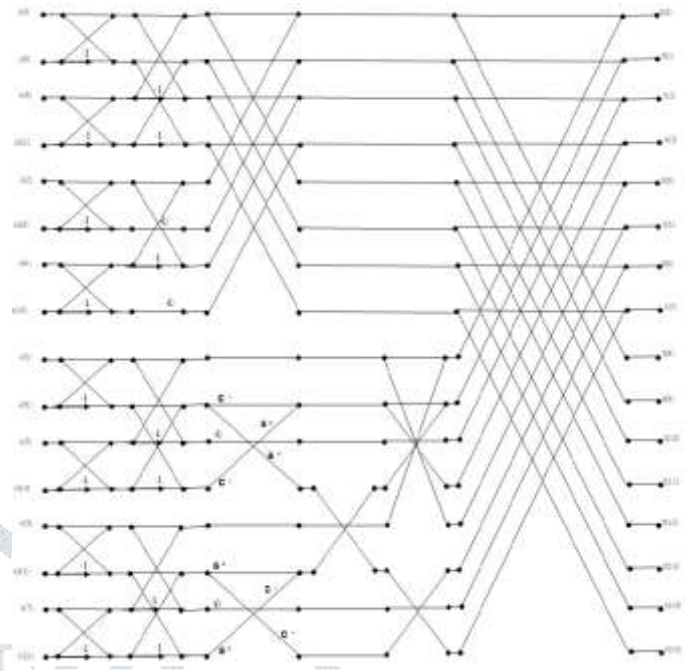

Figure 5: 16-point DHT Butterfly

$$X(0) = (x(0) + x(8)) + (x(4) + x(12)) + (x(2) + x(10))$$
$$+ (x(6) + x(14)) + (x(1) + x(9)) + (x(5) + x(13)) +$$
$$(x(3) + x(11)) + (x(7) + x(15))$$

$$X(1) = (x(0) + x(8)) + (x(4) - x(12)) + c_1(x(2) - x(10))$$
$$+ (x(5) + x(13)) + c_3\{(x(3) - x(11)) + (x(7) - x(15))\} +$$
$$c_2\{(x(3) - x(11)) - (x(7) - x(15))\}$$

$$X(2) = (x(0) + x(8)) - \{(x(4) + x(12)) + (x(6) + x(14)\}$$
$$+ c_1\{(x(1) + x(9)) - (x(5) + x(13))\} + (x(2) + x(10))$$
$$(x(3) + x(11)) + (x(7) + x(15))$$

$$X(3) = (x(0) - x(8)) - (x(4) - x(12)) + c_1(x(6) - x(14))$$
$$+ (x(5) + x(13)) + c_3\{(x(3) + x(11)) + (x(7) - x(15))\} +$$
$$c_2\{(x(3) - x(11)) - (x(7) - x(15))\}$$

$$X(4) = (x(0) + x(8)) + (x(4) + x(12)) + (x(2) + x(10))$$
$$+ (x(6) + x(14)) + (x(1) + x(9)) + (x(5) + x(13)) -$$
$$\{(x(3) + x(11)) + (x(7) + x(15))\}$$

$$X(5) = (x(0) + x(8)) + (x(4) - x(12)) + c_1(x(2) - x(10))$$
$$+ (x(5) + x(13)) + c_3\{(x(3) - x(11)) + (x(7) - x(15))\} -$$
$$c_2\{(x(5) - x(13)) - (x(1) - x(9))\}$$

$$X(6) = (x(0) + x(8)) + (x(6) + x(14)) - \{(x(2) + x(10))$$
$$+ (x(4) + x(12))\} + c_1\{(x(3) - x(11)) + (x(7) - x(15))\}$$

$$X(7) = (x(0) + x(8)) - (x(4) - x(12)) + c_1(x(6) -$$
$$x(14)) - c_3\{(x(5) + x(13)) - (x(1) - x(9))\} + c_2\{(x(5)$$
$$- x(13)) + (x(1) - x(9))\}$$

$X(8) = (x(0) + x(8)) + (x(4) + x(12)) + (x(2) + x(10))$
$+ (x(6) + x(14)) - (x(1) + x(9)) - (x(5) + x(13)) -$
$(x(3) + x(11)) - (x(7) + x(15))$

$X(9) = (x(0) - x(8)) + (x(4) - x(12)) + c_1(x(2) - x(10))$
$+ (x(5) + x(13)) - c_3\{(x(3) - x(11)) + (x(7) - x(15))\} +$
$c_2\{(x(3) - x(11)) - (x(7) - x(15))\}$

Parameter	Previous Technique	Modified DHT	Proposed DHT
No. of Slices	601	553	548
No. of 4 input LUTs	1125	1032	1029
No. of bounded IOBs	256	256	256
Maximum combinational path delay(in ns)	29.62	24.97	21.96

$X(10) = (x(0) + x(8)) - \{(x(4) + x(12)) + (x(6) + x(14))\}$
$- c_1\{(x(1) + x(9)) - (x(5) + x(13))\}$

$X(11) = (x(0) - x(8)) - (x(4) - x(12)) + c_1(x(6) - x(14))$
$+ c_3\{(x(3) - x(11)) + (x(7) - x(15))\} + c_2\{(x(3) - x(11))$
$- (x(7) - x(15))\}$

$X(12) = (x(0) + x(8)) + (x(4) + x(12)) + (x(2) + x(10))$
$+ (x(6) + x(14)) - \{(x(1) + x(9)) + (x(5) + x(13))\} +$
$(x(3) + x(11)) + (x(7) + x(15))$

$X(13) = (x(0) - x(8)) + (x(4) - x(12)) - c_1(x(2) - x(10))$
$- c_3\{(x(3) - x(11)) + (x(7) - x(15))\} + c_2\{(x(1) - x(9))$
$- (x(5) - x(13))\}$

$X(14) = (x(0) + x(8)) - (x(4) + x(12)) - (x(2) + x(10))$
$+ c_1(x(3) + x(11)) - (x(7) + x(15))$

$X(15) = (x(0) - x(8)) - (x(4) - x(12)) - c_1(x(6) - x(14))$
$- c_3\{(x(13) - x(5)) - (x(1) - x(9))\} + c_2\{(x(1) - x(9))$
$- (x(5) - x(13))\}$

IV. PROPOSED TECHNOLOGY

Here, the DHT of length N=16 point requires 67 additions and 12 multiplications. We have used a different technique called Urdhwa tiryakbhyam of ancient Vedic times for multiplication. This multiplication technique reduces the delay and complexity. It converts multiplication into simple logical AND operation using associated circuits of full, half adders and AND gate.

Further, the area is reduced by using various compressors for adding the partial products of multiplication.
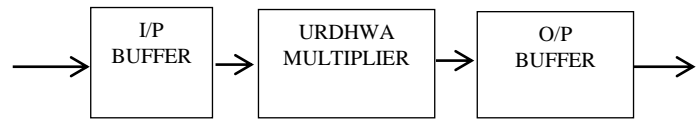

Figure 6: Proposed Architecture

V. SIMULATION RESULT

In this paper, we have designed three architectures of DHT transform. First architecture is designed using simple Urdhwa multiplier. In simple Urdhwa multiplier, we have used 4:2 compressor designed by using full adder and half adder i.e. the basic design of 4:2 compressor. Second architecture is designed using modified Urdhwa multiplier. In modified Urdhwa multiplier, we have used 4:2 compressor designed by using XOR gates and multiplexer.

Third architecture is designed using modified Urdhwa multiplier. In modified Urdhwa multiplier, we have used 4:2 compressor designed by using XNOR gates and multiplexer.

All three architecture are designed by using Xilinx 14.2i.The simulated results are displayed in table 2.In the simulation results proposed DHT gives the best performance.

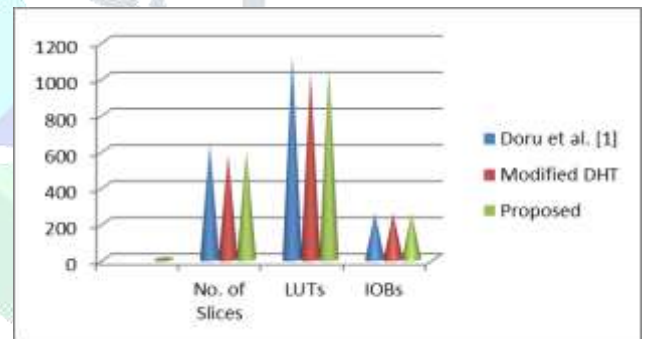TABLE II: Comparisons Result for 8-point Discrete Hartley Transform (DHT) with word length 16.

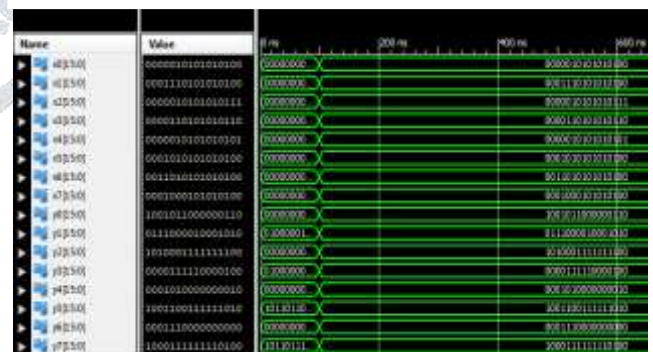Figure 7: Show the bar graph of 8-bit DHT with word length 16

Figure 8: Output Waveform of 8-bit DHT with word length 16

VI. CONCLUSION

DHT is a new transform used for real value to real value conversion. Urdhwa Triyambakam is an ancient technique for multiplication. DHT is used in various fields such as image processing, space science, scientific applications etc. Delay provided and area required by hardware are the two key factors which are need to be consider. Here we present DHT with 8×8 Urdhwa multiplier by using full adder, OR and XNOR gates in different types of compressors.

Among all three compressors, compressor provides the least amount of delay. Also, it takes least number of slices i.e. Occupy least area among all three compressors.

REFRENCES

[1] N. Hamed S. Timarchi "Low-Power and Fast Full Adder by Exploring New XOR and XNOR Gates" IEEE Transactions on Very Large Scale Integration (VLSI) Systems 2018.

[2] Low Power Approximate Adders for General Computing Using Differential Transmission Gate" on 28.03.2018 National Conference on Innovations in Communication and Computing NCICC" 18 SNS College of Technology.

[3] Omid Akbari, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram, "Dual-Quality 4:2 Compressors for Utilizing in Dynamic Accuracy Configurable Multipliers", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 34, Issue 7, 2017.

[4] Doru Florin Chiper, *Senior Member, IEEE,* "A Novel VLSI DHT Algorithm for a Highly Modular and Parallel Architecture", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, VOL. 60, NO. 5, MAY 2013.

[5] Sushma R. Huddar and Sudhir Rao, Kalpana M., "Novel High Speed Vedic Mathematics Multiplier using Compressors ", 978-1 - 4673-5090-7/13/$31.00 ©2013 IEEE.

[6] S. S. Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur and Girish V A, "Implementation of Vedic multiplier for Digital Signal Processing", International Conference on VLSI, Communication & Instrumentation (ICVCI) 2011, Proceedings published by International Joural of Computer Applications® (IJCA), pp.1 -6.

[7] Himanshu Thapaliyal and M.B Srinivas, "VLSI Implementation of RSA Encryption System Using Ancient Indian Vedic Mathematics", Center for VLSI and Embedded System Technologies, International Institute of Information Technology Hyderabad, India.

[8] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja, "Vedic Mathematics: Sixteen simple Mathematical Formulae from the Veda", Delhi (2011).

[9] Sumit Vaidya and Depak Dandekar. "Delay-power performance comparison of multipliers in VLSI circuit design". International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010.

[10] P. D. Chidgupkar and M. T. Karad, "The Implementation of Vedic Algorithms in Digital Signal Procesing", Global J. of Eng. Edu, Vol.8, No.2, 204, UICEE Published in Australia.

[11] Asmita Haveliya, "Design and Simulation of 32-Point FFT Using Radix-2 Algorithm for FPGA Implementation", Second International Conference on Advanced Computing & Communication Technologies IEEE 2012.

[12] S. Correa, L. C. Freitas, A. Klautau and J. C. W. A. Costa, "VHDL Implementation of a Flexible and Synthesizable FFT Processor", IEEE LATIN AMERICA TRANSACTIONS, VOL. 10, NO. 1, JAN. 2012.