FPGA Implementation of 4-Point and 8-Point Fast Hadamard Transform

Ankit Agrawal
M.Tech
Electronics engineering department, MNIT, Jaipur
Rajasthan, INDIA.

Rakesh Bairathi
Associate Professor
Electronics engineering department, MNIT, Jaipur
Rajasthan, INDIA.

Amit Joshi
Assistant Professor
Electronics engineering department, MNIT, Jaipur
Rajasthan, INDIA.

ABSTRACT
Transformation is one of the fundamental blocks of many signal processing applications. The Hadamard transform is useful in variety of application including data encryption methods and latest data compression algorithms such as JPEG extended range (JPEG XR), High Efficiency Video Coding (HEVC) etc. Hadamard transform is multiplier less technique and requires additions and subtractions only. In this paper, we have proposed an efficient method of 4 and 8 points Hadamard transformation using a parallel processing to achieve higher speed. The 8-point DHT has been realized with 4 points DHT implementation. The modules are synthesized using Xilinx ISE 14.2 software with the usage of inbuilt memory core generator for storing co-efficient values. The performance has been verified with area and timing analysis. The proposed implementation shows excellent results and also compared to previous works.

Keywords
Distributed memory, Frequency domain, Real time implementation, Reconfiguarble, Synthesize.

1. INTRODUCTION
In signal processing applications, the transformation methods are the integral computation part where original signal has to be transformed from spatial to frequency domain. In all image or video coding standard, the transformation is the essential part which has made the calculation relatively easy and efficient [1]. With the help of transformation methods, the analysis can be performed very effortlessly. There are various types of transforms such as, Fourier Transform (FT), Laplace Transform (LT), Discrete Wavelet Transform (DWT), Discrete cosine Transform (DCT), used in different signal processing applications [2]. Hadamard Transform is a discrete transform which is known as the Walsh Transform and is considered as a generalized class of Fourier transform. Discrete Hadamard transform (DHT) has a considerable computational advantage over other transformation methods. DHT implementation does not include multiplication operation which reduces the complexity. It has been adopted in H.264/AVC video coding standard for better real time performance [3]. Hadamard transform uses Nlog2N addition/subtraction compare to Fourier transform which requires Nlog2N multiplication and addition which is faster than the Fourier transform. The biggest advantage of the Hadamard transform is that the transformation operation uses +1 and -1 only. This leads to not only an improvement in terms of speed, but avoiding the multiplying that significantly reduces the hardware requirement. The Hadamard transformation has been used in many digital signal processing applications as pattern matching, scene labeling, mouth tracking, path tracking, texture synthesis, augmented reality and many more. It is the one of the important blocks of latest High Efficiency Video Coding (HEVC) standard. It is also useful in the pseudo code generation, error control coding and data encryption. DHT is a useful tool for image analysis and also helps to achieve a good data compression ratio. It also helps in image transformation technique to recognize an object in an image. With the help of DHT, an image is compressed in few coefficients which help to understand energy compaction of the image. It provides greater energy compression for highly correlated images. The objective of the paper is to design an efficient DHT for real time signal processing application. The FPGA provides the advantages in terms of field programmability and processing speed [4]. FPGA implementation outperforms DSP implementation in terms of speed, power and area [5]. FPGA offers flexibility and is less expensive than ASIC therefore DHT architecture has been implemented on an FPGA platform.

The organization of the paper is in following manner: Section II covers the brief literature survey of Hadamard transform implementation. In subsequent sections III, the mathematical model for 4 points and 8 point Hadamard transform has been discussed. Section IV covers the implementation of proposed hardware architecture which is followed by results and discussion in section V. Finally, the work is concluded in Section VI.

2. LITERATURE SURVEY
Hadamard transform has been used for feature selection in the character recognition task. Jurczyk and Loparo have discussed an object recognition system which uses tactile array data using Walsh-Hadamard transform technique [6]. Objects are recognized according to their local features by correlating transformed data from a tactile sample with data obtained from reference images contained in a model library. Several test results are presented to illustrate the performance of the system. P.K. Meher et al.[7] have presented fully pipelined simple modular structure for the efficient hardware realization of discrete Hadamard transform (HT). From the kernel matrix of HT, they had derived four different pipelined modular designs for length N= 4. The architecture was coded using VHDL which simulated by Xilinx ISE tool for functional validation and synthesized for implemented in different FPGA devices. Abbes Amira et al. [8] had proposed a parameterizable and scalable architecture for fast Hadamard transform (FHT) with time and area complexities of O(2(W+1)) and O(2N2) respectively. Porto et al. [9] presented a design space exploration on the forward 4x4 Hadamard Transform in H.264/AVC video coding standard. These implementations were described by VHDL and synthesized on the Altera Stratix device. The speed of the system was

slower compared to other works. Bernhard et al. [10] had investigated a multicarrier transmission scheme using Hadamard transform. The transform has less computational complexity than FFT which is generally used for OFDM. Amira et al. [11] presented two novel architectures for the fast Hadamard transforms using both a systolic architecture and distributed arithmetic techniques. In this paper, we have proposed a simple and efficient method to implement a 4-point DHT and an 8-point DHT.

3. DISCRETE HADAMARD TRANSFORM

3.1 Basic Mathematical Formulations:

The Discrete Hadamard Transform of a sequence x (n) (where n= 0, 1, 2……..) is given by

$$X(k) = \sum_{n=0}^{N-1} H_N(k, n)\, x(n) \qquad (1)$$

Where k varies from 0 to N-1 and HN is a Hadamard matrix of size N x N.

The basis matrix of the Hadamard transform which is known as a Hadamard matrix carries elements +1 or -1 only and is given by

$$H2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

The above is a Hadamard matrix for N=2. Hadamard matrices of higher order can be generated by using the recursive property of Hadamard matrix.

$$H2N = \begin{pmatrix} HN & HN \\ HN & -HN \end{pmatrix}$$

Where HN is an N x N Hadamard matrix.

Hadamard is an orthogonal, symmetric and linear transform and has 2m real numbers.

$$H = HT = H\text{-}1 = H^*$$

Where H* is the conjugate of H and HT is the transpose of H.

3.2 4-point DHT:

To implement 4-point DHT, we have used the look ahead carry adder which is a very fast adder. There are 4 inputs x0, x1, x2, x3 and the 4 outputs y0, y1, y2, y3 in the 4-point DHT. The 4-point DHT can be implemented as a simple matrix multiplication of a 4th order Hadamard matrix and the input matrix is defined as

$$\begin{pmatrix} y0 \\ y1 \\ y2 \\ y3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} x0 \\ x1 \\ x2 \\ x3 \end{pmatrix}$$

x0, x1, x2, x3 are the inputs and y0, y1, y2,y3 are the outputs.

y0 = x0 + x1 + x2 + x3
y1 = x0 - x1 + x2 - x3
y2 = x0 + x1 - x2 - x3
y3 = x0 - x1 - x2 + x3

The proposed architecture has two stage implementation and stage wise process is shown in Table I.

Table 1. Stage wise DHT process

First stage	Second stage
tmp0 = x0 + x1	y0 = tmp0 + tmp1
tmp1 = x2 + x3	y1 = tmp2 + tmp3
tmp2 = x0 – x1	y2 = tmp0 – tmp1
tmp3 = x2 – x3	y3 = tmp2 – tmp3

In the first stage, intermediate results tmp0, tmp1, tmp2, tmp3, are generated and in the second stage final results y0, y1, y2, y3 are generated.

3.3 8-point DHT:

With the help of recursive property of the Hadamard transform, 8- point DHT can be implemented using two 4-point DHT blocks and additional adder/ subtractors. The 4-point DHT is used for implementation of the 8-point DHT. 8-point DHT can be implemented with 3 levels of adder and subtractor. We can implement 8-point Hadamard transform into the form of matrix multiplication as shown below:

$$\begin{pmatrix} y0 \\ y1 \\ y2 \\ y3 \\ y4 \\ y5 \\ y6 \\ y7 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} x0 \\ x1 \\ x2 \\ x3 \\ x4 \\ x5 \\ x6 \\ x7 \end{pmatrix}$$

y0 = x0 + x1 + x2 + x3 + x4 + x5 + x6 + x7
y1 = x0 - x1 + x2 - x3+ x4 - x5 + x6 - x7
y2 = x0 + x1 - x2 - x3+ x4 + x5 - x6 - x7
y3 = x0 - x1 - x2 + x3+ x4 - x5 - x6 + x7
y4 = x0 + x1 + x2 + x3 - x4 - x5 - x6 - x7
y5 = x0 - x1 + x2 - x3- x4 + x5 - x6 + x7
y6 = x0 + x1 - x2 - x3- x4 - x5 + x6 + x7
y7 = x0 - x1 - x2 + x3- x4 + x5 + x6 - x7

The proposed 8- point structure can be implemented in three stages using adder and subtractor or with two stages using 4-point DHT which uses the recursive property of the Hadamard transform.

4. FPGA IMPLEMENTATION OF PROPOSED ARCHITECTURE

The proposed architecture has been implemented using pipelining concept. The architecture is implemented as stage-wise and each stage uses carry look ahead adder and

subtractor module. The Adder Module (AM) and Subtrcator Module (SM) is shown in Fig.1 and Fig.2 respectively.

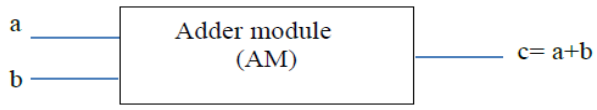

Fig.1 Adder module

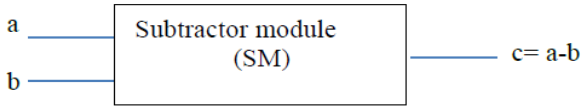

Fig.2 Subtractor module

The proposed architecture of 4 points DHT is shown in Fig. 3 and 8 point DHT is shown in Fig.4. The 4 points DHT requires total 8 adder/subtractor whereas 24 adder/subtractor are needed for 8 point DHT implementation. In general, N point DHT uses Nlog2N adder/subtractor modules.

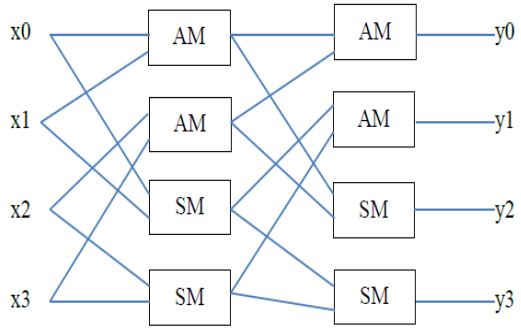

Fig.3 Proposed architecture of 4 points DHT

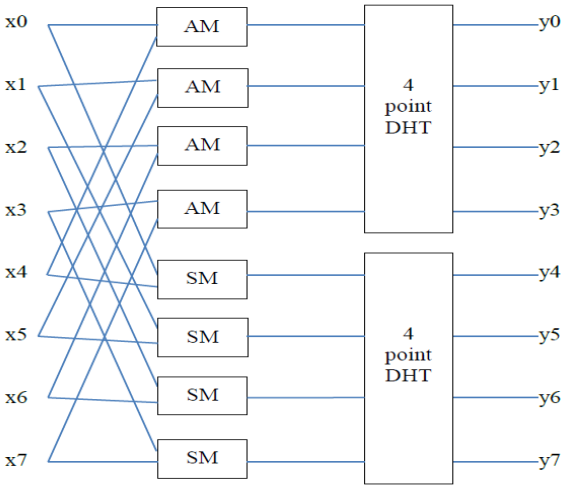

Fig.4 Proposed architecture of 8 point DHT

The design of handmade transform is prototyped on FPGA as shown in Fig. 5. The first step to read the image through MATLAB and then store the pixel values in form of coefficients in the coefficient file (. COE file). Next step is to store these values of coefficients using a memory core generator which helps to get the values to be processed in our program for transformations. The output transformed values of pixels have been stored in a file. Further calculations can be done on this transformed data.

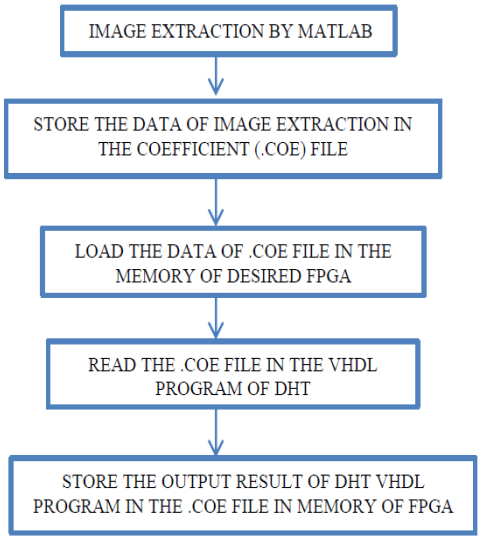

Fig. 5 Design flow diagram of Discrete Hadamard transforms

The Distributed Memory Core Generator is already existed as Intellectual Property (IP) Core in Xilinx Synthesis Technology (XST). The Distributed Memory Generator core is used to create memory structures using LUTs. The depths of the core can be adjusted from 16–65,536 words and bit width has a range of 1 to 1024 bits. The block of Distributed memory core is shown in Fig. 6.

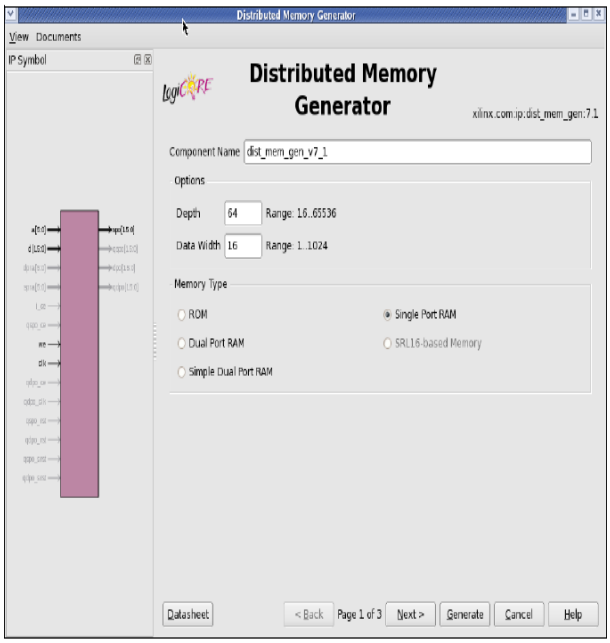

Fig. 6. Distributed Memory core generator IP core

5. RESULTS AND ANALYSIS

For the software validation, DHT has been implemented on MATLAB R2014a and subsequently it is implemented in hardware using Xilinx ISE Design tool.

5.1 Software Validation:

The program of DHT is simulated on MATLAB-R2014a for Intel (R) core (TM) i3 CPU M 370 @ 2.4 GHZ process. The average elapsed time of 8 points DHT is around 23.3 µs (micro second).

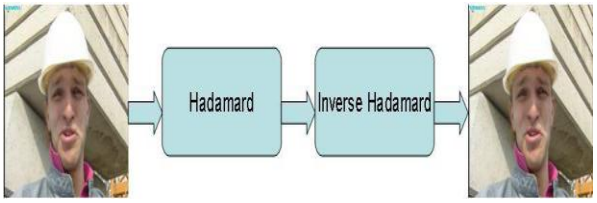

Fig. 7. Software validation of an image

5.2 FPGA Implementation:

The 8-point DHT is implemented on Sparten-3 FPGA family xc3s200-5pq208 board using Xilinx tool 14.2 and the resultant maximum combinational delay is 12.547 ns (nano second). This signifies the hardware performance in terms of speed which is far better in comparison of software implementation.

a). 4-POINT DHT: The proposed 4 points DHT structure is simulated in modelsim with VHDL and is synthesized by Xilinx ISE on Sparten 3 family's xc3s200-5pq208 board. The simulation results are shown in Fig. 8.

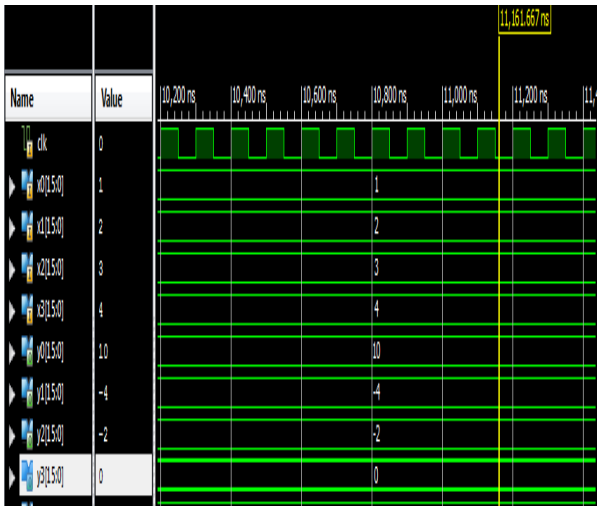

Fig. 8. Simulation result of 4 points DHT

In the simulation results x0, x1, x2, x3 are the inputs of 16 bit and y0, y1, y2, y3 are the outputs of 16 bit. The timing results of 4 point DHT is shown in Table II. Top level RTL schematic and its internal architecture is shown in Fig. 9 and Fig.10 respectively.

Table 2. Timing results of 4- point DHT implementation on SPARTAN -3

Timing Parameter/ Frequency	Resultant Value
Minimum period	3.514ns
Maximum Frequency	284.584MHz
Minimum input arrival time before clock	3.873ns
Maximum output required time after clock	9.065ns
Maximum combinational path delay	9.287ns

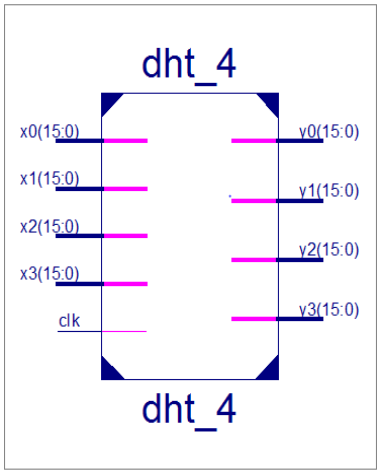

Fig. 9. RTL schematic of 4 point DHT

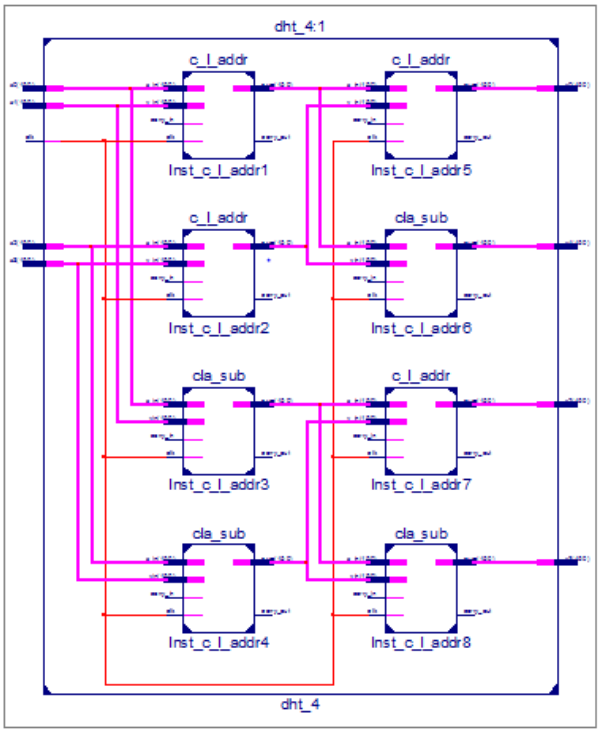

Fig. 10. Internal architecture of 4-point DHT using adders and subtractors

Table 3. Comparison of 4 point DHT Implementation on VIRTEX- 4 FPGA platform

Work	Area (slices)	Maximum frequency (MHz)
Proposed	129	699.276
Meher et al [7]	32	294
Amira et al [11]	82	227

Table 4. Comparison of 4 points DHT Implementation on SPARTAN- 3 FPGA platform

Work	Maximum combinational path delay (ns)
Proposed	9.287
Sridevi et al [12]	19.639

From Table III, the results show that proposed DHT structure is operated as much higher speed (almost 3x times) in comparison with related work but against the compromise in terms of the area. The structure is designed to achieve higher speed for real time applications. Table IV signifies that proposed scheme has almost half combination delay than previous work.

b). 8-POINT DHT: For the 8-point DHT, we used Xilinx ISE 14.2 design tool with xc3s2000-4fg456 board and the results are reported in Table V and Table VI. The simulation results are shown in Fig. 11.

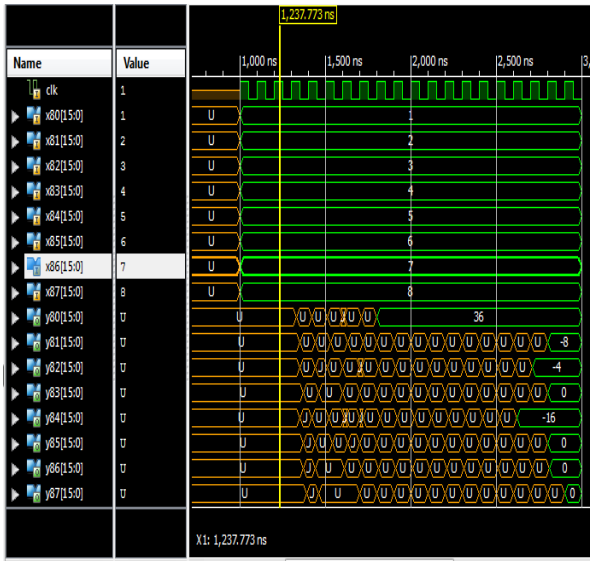

Fig. 11. Simulation result of 8 points DHT

In the simulation results x0, x1, x2, x3, x4, x5, x6, x7, x8 are the inputs of 16 bits and y0, y1, y2, y3, y4, y5, y6, y7 are the outputs of 16 bits. The maximum combinational delay for 8-point DHT is 12.547 ns.

Table 5. Timing results of 8- point DHT implementation on SPARTAN -3

Timing Parameter/ Frequency	Resultant Value
Minimum period	4.209 ns
Maximum Frequency	237.586 MHz
Minimum input arrival time before clock	6.305 ns
Maximum output required time after clock	12.102 ns
Maximum combinational path delay	12.547 ns

Top level RTL schematic and its internal architecture for 8 points Hadamard transform is shown in Fig. 12 and Fig.13 respectively.

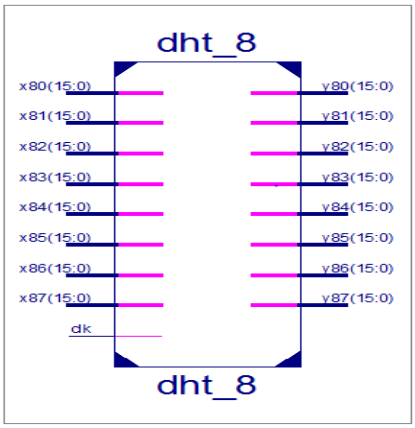

Fig.12. RTL shcematic of 8 point DHT

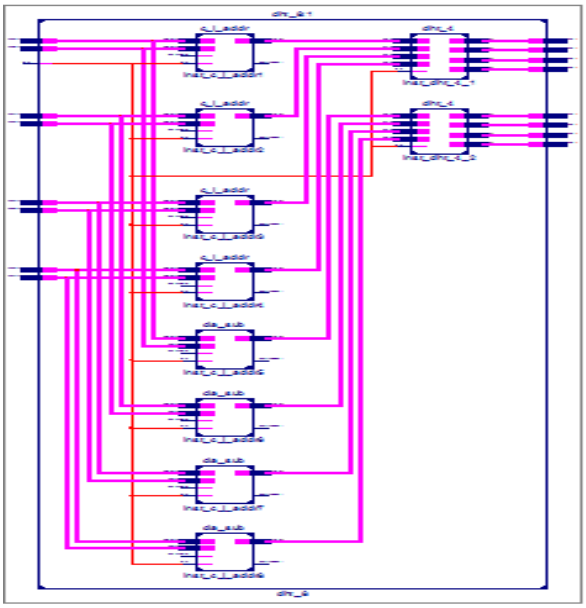

Fig. 13. Internal architecture of 8 points DHT using 4-point DHT

Table 6. Comparison of 8 points DHT Implementation on VIRTEX- 4 FPGA platform

Work	Area (slices)	Maximum frequency (MHz)
Proposed	394	643.708
Meher et al [7]	96	471
Amira et al [11]	163	212

Table 7. Comparison of 8 points DHT Implementation on SPARTAN- 3 FPGA platform

Work	Maximum combinational delay (ns)
Proposed	12.547 ns
Sridevi et al [12]	19.339 ns

Table VI and Table VII show the high performance of 8 points DHT with previously designed structures. This suggests that our proposed architecture works at higher speed than previous work and also has lesser combinational path delay than previous related work.

6. CONCLUSION

P The paper present hardware implementation of Hadamard transforms to verify its real time performance. The elements of the Hadamard matrix have either +1 or -1. The computation of Hadamard transform consists of additions and subtractions only. As it is multiplierless therefore the complexity is in order of N^2 to $N\log_2 N$. The higher orders DHT can be implemented using lower order DHT's. The maximum combinational path delay for 4-point DHT is 9.287 ns and for 8-point is 12.547 ns which show the speed improvement in comparison of previous implementations. The proposed DHT structure is a better choice for real time signal processing applications.

7. REFERENCES

[1] Fan, Chih-Peng, Chia-Wei Chang, and Shun-Ji Hsu. "Cost-Effective Hardware-Sharing Design of Fast Algorithm Based Multiple Forward and Inverse Transforms for H. 264/AVC, MPEG-1/2/4, AVS, and VC-1 Video Encoding and Decoding Applications." *Circuits and Systems for Video Technology, IEEE Transactions on* 24, no. 4 (2014): 714-720.

[2] Joshi, Amit M., Vivekanand Mishra, and R. M. Patrikar. "Design of real-time video watermarking based on Integer DCT for H. 264 encoder." *International Journal of Electronics* 102, no. 1 (2015): 141-155.

[3] Rosenfeld, Azriel, and Avinash C. Kak. Digital picture processing. Vol. 1. Elsevier, 2014.

[4] Joshi, Amit M., Vivekanand Mishra, and R. M. Patrikar. "FPGA prototyping of video watermarking for ownership verification based on H. 264/AVC."*Multimedia Tools and Applications* (2015): 1-24.

[5] Joshi, A., Mishra, V., & Patrikar, R. M., "Real Time Implementation of Digital Watermarking Algorithm for Image and Video Application," InTech, Watermarking/Book2, 2012,pp.64-88.

[6] J. Jurczyk and K. Loparo, ll~athematicatlr ansforms and correlation techniques for object recognition using tactile data," IEEE Transactions on Robotics and ~utomation, vol. 5, No.3, June 1989, pp. 359-362.

[7] Meher, Pramod Kumar, and Jagdish Chandra Patra. "Fully-pipelined efficient architectures for FPGA realization of discrete Hadamard transform." In*Application-Specific Systems, Architectures and Processors, 2008. ASAP 2008. International Conference on*, pp. 43-48. IEEE, 2008.

[8] Amira, Abbes, and Shrutisagar Chandrasekaran. "Power modeling and efficient FPGA implementation of FHT for signal processing." *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 15, no. 3 (2007): 286-295.

[9] Porto, M. S., T. L. Da Silva, R. E. C. Porto, L. V. Agostini, I. V. da Silva, and S. Bampi. "Design space exploration on the H. 264 4× 4 Hadamard transform." In NORCHIP Conference, 2005. 23rd, pp. 188-191. IEEE, 2005.

[10] Bernhard, Michael, and Joachim Speidel. "Multicarrier Transmission using Hadamard Transform for Optical Communications." *ITG-Fachbericht-Photonische Netze* (2013).

[11] Amira, A., A. Bouridane, P. Milligan, and M. Roula. "Novel FPGA implementations of Walsh–Hadamard transforms for signal processing." *IEE Proceedings-Vision, Image and Signal Processing* 148, no. 6 (2001): 377-383.

[12] Sridevi, J., J. E. N. Abhilash, and J. Vasanta Kumar. "Implementation Of Fully-Pipelined 16-Point DHT Architectures Using 8-Point And 4-Point DHTs for FPGA Realization." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 1.9 (2012):pp-256.