Circuits, Systems, and Signal Processing

NOVEL ALGORITHM FOR DISCRETE HARTLEY TRANSFORM AND ITS RECURSIVE STRUCTURE

--Manuscript Draft--

Manuscript Number:	
Full Title:	NOVEL ALGORITHM FOR DISCRETE HARTLEY TRANSFORM AND ITS RECURSIVE STRUCTURE
Article Type:	Original Research
Keywords:	recursive structure; hardware complexity; discrete Hartley transform
Corresponding Author:	Priyanka Jain, Ph.D Delhi Technological University Department of Electronics and Communication Delhi, INDIA
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Delhi Technological University Department of Electronics and Communication
Corresponding Author's Secondary Institution:	
First Author:	Vivek Singh
First Author Secondary Information:	
Order of Authors:	Vivek Singh
	Dhwani Kaushal
	Priyanka Jain, Ph.D
Order of Authors Secondary Information:	
Funding Information:	
Abstract:	The paper presents a new recursive algorithm for the computation of discrete Hartley transform for an input sequence of length N, where N = 2m and m >=4. All the necessary equations for the computation have been derived along with the realization of the recursive structure. The suggested algorithm and the proposed structures have been verified by simulating it in MATLAB Simulink. The proposed folding recursive algorithm can compute DHT with minimum hardware complexity, which is highly useful in parallel VLSI implementations.

NOVEL ALGORITHM FOR DISCRETE HARTLEY TRANSFORM AND ITS RECURSIVE STRUCTURE

Vivek Singh, Dhwani Kaushal, Priyanka Jain

(vsvivek1201@gmail.com), (dhwanikaushal009@gmail.com), (Priyankajain@dtu.ac.in)

Delhi Technological University, Shahbad, Daulatpur, Bawana, Delhi-110042,INDIA

***Abstract* — The paper presents a new recursive algorithm for the computation of discrete Hartley transform for an input sequence of length N, where N = 2^m and m >=4. All the necessary equations for the computation have been derived along with the realization of the recursive structure. The suggested algorithm and the proposed structures have been verified by simulating it in MATLAB Simulink. The proposed folding recursive algorithm can compute DHT with minimum hardware complexity, which is highly useful in parallel VLSI implementations.**

Keywords—recursive structure, hardware complexity, discrete Hartley transform.

I. INTRODUCTION

Over the years, orthogonal transforms have played a major role in signal processing and communication fields. To solve current and upcoming challenges created due to new scientific developments or on the request of new applications, transforms are continuously evolving and the need for new ones has been realized. One such important orthogonal transform was introduced by Bracewell known as the discrete Hartley transform, which progressively became a common substitute for real-valued discrete Fourier transform (RDFT) [1]. DHT is used when a signal is shifted in the time domain and frequency domain [2].

DHT has been popular due to its real-valued and symmetric transform kernel which is identical to that of its inverse, which can be self-inverse or involuntary, to a scale factor of 1/N [3]. DFT properties like shifting and circular convolution theorems, can also be applied to DHT. In general, if the DHT coefficients of a signal are known, the computation of DFT coefficients can be done easily, and vice versa [4]. In FFT algorithms, one complex multiplication is equivalent to four real multiplications, whereas there is no such condition in the computation of DHT [5]. Therefore, the Discrete Hartley transform would be computationally faster than the FFT for computing the Fourier transform of a large data set.

The DHT is used in several signal and image processing applications, like circular error control coding, convolution and interpolation, adaptive filtering, image classification, image encryption, or image compression [3]. Other transforms like, Discrete sine, Discrete cosine Discrete Fourier, and Hilbert can also be calculated using the Discrete Hartley transform [6]. Discrete Hartley Transform when combined with Walsh-Hadamard transform can be used to calculate the Fast Walsh-Hadamard-Hartley transform [7]. Orthogonal frequency division multiplexing (OFDM) technology uses the Hartley transform instead of the Fourier transform, to reduce the computational complexity significantly and it has also reduced the hardware complexity while doubling the spectral efficiency [8]. Since most of the spectrum analysis problems in the real world involve the use of real-valued data only, therefore notable computational improvements can be achieved by adopting DHT [1]. Since DHT has the same form for the forward as

well as the inverse DHT, the same hardware/software can be applied to both transform, making it a cost-effective approach for the industry [9].

In the literature, several algorithms for the computation of DHT have been proposed, aiming to reduce the complexity of the hardware and the number of computations required for calculating DHT coefficients. The number of arithmetic computations required for the direct calculation of DHT, introduced by Bracewell [5], requires N^2 multiplications and additions, therefore a fast Hartley algorithm was developed by Bracewell [10], in which the implementation of DHT was proposed with a complexity proportionate to ($N \log_2 N$) by making the use of radix-2 decimation in-time algorithm, on the cost of complex hardware structure. In [11], Murty has presented a recursive algorithm for 1-D DHT of even length, by using Chebyshev polynomial and single folding of the input data. Fang and Lee have represented 1-D DHT using two new linear systolic architectures, which provide features like modularity, high pipeline ability, and efficient VLSI hardware implementation [9]. The architectures were developed by making use of Horner's rule along with the symmetric property of DHT in the time/frequency domain. In [12], Chang and Lee have proposed two models of linear systolic array, for 1-D DHT, which exhibit several advantages such as simplicity, locality, and a more suitable VLSI implementation than that of FHT. Liu and Chiu have adopted a new scheme of time–recursive approach in [13], to compute DHT, DCT, and DST. By employing such an approach, they have developed a unified parallel lattice structure. In [14], Chiper presented a new VLSI algorithm for DHT of length 2^N with a regular structure, which can be implemented on parallel VLSI architecture with high modularity. The literature discussed so far has proposed algorithms and architectures that contribute towards better hardware complexity, however, as we move forward, the literature discussion focuses on algorithms that provide better computational complexity. In [1], Hamood has developed another new algorithm for FHT which is based on the radix -2/4/8 method. The proposed algorithm has arithmetic complexity comparable to that of the split radix FHT algorithm [15][16][17][18] while preserving the regularity and the simple butterfly structure of the radix-2 algorithm. In [19], Chiper has developed a new split–radix algorithm for computing DHT of length $N=2^N$, which enables the implementation of very efficient parallel hardware by making use of a dual-core system. All the above previously developed structures, as per the best of our knowledge, do not propose a recursive structure, even after giving a recursive algorithm. They require more complex butterfly/cross connections in their structures.

In this paper, a new recursive algorithm for the Discrete Hartley Transform has been developed, along with a recursive generalized structure, which can be used for all N (where $N = 2^m$ and m >=4). It can also be implemented by making use of parallel very-large-scale integration (VLSI) digital filters and can thus provide considerable hardware savings as compared to other previously developed algorithms. This paper has been organized into VII sections. Section II discusses the algorithm for DHT, in section III, the realization of the algorithm through IIR filter structure is presented, and section IV discusses the proposed algorithm's architecture. Performance comparison of the suggested structure along with discussion, with most of the existing work in the literature has been done in sections V and VI, and section VII deals with the conclusion.

II. Algorithm For Recursive Formula For DHT

The DHT of a sequence x(n) with length N is defined as [1]:

$$H[k] = \sum_{m=0}^{N-1} x[m] cas\left(\frac{2\pi mk}{M}\right), \quad (1)$$

where k=0,1,2…N and $\text{cas}\theta = \sin\theta + \cos\theta$

Decimate $x[n]$ into two sequence of $N/2$ samples. Let the two sequences be

$$H_e[k] = \sum_{n=0}^{N/2-1} x[2n]\ cas\left((2n)\frac{2\pi k}{N}\right) \quad (2)$$

and

$$H_o[k] = \sum_{n=0}^{N/2-1} x[2n+1]\, cas\left((2n+1)\frac{2\pi k}{N}\right) \quad (3)$$

where k = 0, 1, 2 ... N/2-1 From (2) and (3) it is clear that

$$H[k] = H_o[k] + H_e[k] \quad (4)$$

Here $H_o[k]$ and $H_e[k]$ are the DHT of size N/2.

Due to symmetry property of DHT, remaining coefficients could be obtained as

$$H\left[\frac{N}{2}+k\right] = H_e[k] - H_o[k]\ , k = 0,1,2 \ldots \frac{N}{2}-1 \quad (5)$$

Case I: Divide-and-conquer algorithm for even samples of the input sequence:

Equation (2) can be further divided as even and odd samples and can be rewritten as

$$H_e[k] = \sum_{n=0}^{N/4-1} x[4n]\, cas\left((4n)\frac{2\pi k}{N}\right) + \sum_{n=0}^{N/4-1} x[4n+2]\, cas\left((4n+2)\frac{2\pi k}{N}\right) \quad (6)$$

Now considering the first part of the above equation separately as

$$H_{ee}[k] = \sum_{n=0}^{N/8-1} x[4n] cas\left((4n)\frac{2\pi k}{N}\right) + \sum_{n=N/8}^{N/4-1} x[4n] cas\left((4n)\frac{2\pi k}{N}\right) \quad (7)$$

Put $n = n' + \frac{N}{8}$ and dropping the dashes in the second term of the equation (7), we get

$$H_{ee}[k] = \sum_{n=0}^{N/8-1} x[4n] cas\left((4n)\frac{2\pi k}{N}\right) + \sum_{n=0}^{N/8-1} x\left[4\left(n+\frac{N}{8}\right)\right] cas\left(4\left(n+\frac{N}{8}\right)\frac{2\pi k}{N}\right) \quad (8)$$

Solving the right term of equation (8) and expanding the cas term by using the following identities

$$\sin(A+B) = \sin A \cos B + \cos A \sin B;\ \cos(A+B) = \cos A \cos B - \sin A \sin B \quad (9)$$

Equation (8) becomes

$$H_{ee}[k] = \sum_{n=0}^{N/8-1} W_{ee}[n] cas\left((4n)\frac{2\pi k}{N}\right), \text{ k = 0, 1, 2 ... N/4-1} \quad (10)$$

Where

$$W_{ee}[n] = x(4n) + (-1)^k x\left(4\left(n+\frac{N}{8}\right)\right) \quad (10a)$$

After doing some mathematical manipulation equation (10) may be written as

$$H_{ee}[k] = (-1)^k . A_i[k] \quad (11)$$

Where

$$A[k] = \sum_{n=0}^{i} W_{ee}[i-n]\, cms\big((n+1)\theta_k\big),\ \text{k} = 0, 1, 2 \ldots \text{N}/4-1 \quad (11a)$$

Where $\theta_k = (8\pi k/N), cms\theta = cos\theta - sin\theta, i = \frac{N}{8} - 1$

We shall use the trigonometric identities

$$\cos(r\theta) = 2\cos[(r-1)\theta]\cos\theta - \cos[(r-2)\theta] \quad (12a)$$

$$\sin(r\theta) = 2\sin[(r-1)\theta]\cos\theta - \sin[(r-2)\theta] \quad (12b)$$

By algebraic manipulations and using equations (12a) and (12b). equation (11a) can be expressed in the following recursive form:

$$A_i[k] = W_{ee}[i]\,\mathrm{cms}(\theta_k) - W_{ee}[i-1] + 2\,A_{i-1}[k]\cos\theta_k - A_{i-2}[k] \quad (13)$$

Step by Step derivation from equation (8) to equation (13) is shown in the **Appendix A**

Now, second part of equation (6) will be solved in a similar manner as for the first part of equation (6)

$$H_{eo}[k] = (-1)^k . B_i[k] \quad (14)$$

$$B_i[k] = \sum_{n=0}^{i} W_{eo}[i-n]\, cms\left(n+\frac{1}{2}\right)\theta_k, k = 0,1,2 \ldots \frac{N}{4} - 1 \quad (14a)$$

Where $W_{eo}[n] = \left[x[4n+2] + (-1)^k x\left[4\left(n+\frac{N}{8}\right)+2\right]\right]$ (14b)

And recursive form of equation (14) is

$$B_i[k] = W_{eo}[i]cms\left(\frac{\theta_k}{2}\right) - W_{eo}[i-1]\, cas\left(\frac{\theta_k}{2}\right) + 2\,B_{i-1}[k]\cos\theta_k - B_{i-2}[k] \quad (15)$$

Case II: Divide-and-conquer algorithm for odd samples of the input sequence:

Dividing the odd samples of the input sequence in first N/4 samples and last N/4 sample in equation (3) and solving the *cas* kernel, equation (3) can be written as

$$H_o[k] = \sum_{n=0}^{N/4-1} W_k[n] cas\left((2n+1)\frac{2\pi k}{N}\right), k = 0,1,2 \ldots \frac{N}{2} - 1 \quad (16)$$

Where

$$W_k[n] = x(2n+1) + (-1)^k x\left(2\left(n+\frac{N}{4}\right)+1\right) \quad (16a)$$

Now equation (16) can be solved depending on *k* as

(1) *(K EVEN)*

When k is even equation (16) can be folded once again and becomes

$$H_{oe}[k] = \sum_{n=0}^{N/8-1} W_{ke}[n] cas\left((2n+1)\frac{2\pi k}{N}\right) + \sum_{n=N/8}^{N/4-1} W_{ke}[n] cas\left((2n+1)\frac{2\pi k}{N}\right) \quad (17)$$

where

$$W_{ke}[n] = x(2n+1) + x\left((2n+1) + \frac{N}{2}\right),\ \mathrm{k} = 0, 2, 4 \ldots \mathrm{N}/2 - 2 \quad (17a)$$

Following the same steps as in equation (6)-(13), equation (17) becomes

$$H_{oe}[k] = (-1)^{\frac{k}{2}} C_i[k],\ \mathrm{k} = 0, 2, 4 \ldots \mathrm{N}/2 - 2 \quad (18)$$

$$C_i[k] = \sum_{n=0}^{i} M_k[i-n]\, cms\left(n + \frac{1}{2}\right)\theta_k,\ \mathrm{k} = 0, 2, 4 \ldots \mathrm{N}/2 - 2 \quad (19)$$

and $\theta_{k1} = (4\pi\mathrm{k}/\mathrm{N})$;

$$M_k[n] = \left[W_{ke}[n] + (-1)^{\frac{k}{2}} W_{ke}\left[n + \frac{N}{8}\right]\right] \quad (19a)$$

By algebraic manipulations equation (19) can be expressed in the following recursive form

$$C_i[k] = M_k[i]cms\left(\frac{\theta_{k1}}{2}\right) - M_k[i-1]\, cas\left(\frac{\theta_{k1}}{2}\right) + 2\, C_{i-1}[k]\cos\theta_{k1} - C_{i-2}[k] \quad (20)$$

(2) *(K ODD)*

When K odd, equation (16) can be further divided into two parts and becomes

$$H_{oo}[k] = \left[\sum_{n=0}^{N/8-1} W_{ko}[n] cas\left((2n+1)\frac{2\pi k}{N}\right) + \sum_{n=N/8}^{N/4-1} W_{ko}[n] cas\left((2n+1)\frac{2\pi k}{N}\right)\right] \quad (21)$$

where

$$W_{ko}[n] = x(2n+1) - x\left((2n+1) + \frac{N}{2}\right), \mathrm{k} = 1, 3 \ldots \mathrm{N}/2 - 1 \quad (21a)$$

Equation (21) could be written as

$$H_{oo}[k] = H_{ooA}[k] + H_{ooB}[k], \mathrm{k} = 1, 3 \ldots \mathrm{N}/2 - 1 \quad (22)$$

Where

$$H_{ooA}[k] = \sum_{n=0}^{N/8-1} W_{ko}[n]\, cas\left((2n+1)\frac{2\pi k}{N}\right), \mathrm{k} = 1, 3 \ldots \mathrm{N}/2 - 1 \quad (23a)$$

and

$$H_{ooB}[k] = \sum_{n=N/8}^{N/4-1} W_{ko}[n]\, cas\left((2n+1)\frac{2\pi k}{N}\right), \mathrm{k} = 1, 3 \ldots \mathrm{N}/2\text{-}1 \quad (23b)$$

Further folding the input samples of $H_{ooA}[k]$,Therefore, equation (23a) becomes

$$H_{ooA}(k) = \sum_{n=0}^{N/16-1} M_k^1[n] cas\left((2n+1)\frac{2\pi k}{N}\right), \mathrm{k} = 1, 3 \ldots \mathrm{N}/2 - 1 \quad (24)$$

where $M_k^1[n] = \left[W_{ko}[n] + (-1)^{\left(\frac{\mathrm{k}-1}{2}\right)} W_{ko}\left[\frac{N}{8} - n - 1\right]\right]$ (24a)

$$H_{ooA}[k] = cas\left(\frac{\pi k}{4}\right) G_j^1[k] - cms\left(\frac{\pi k}{4}\right) G_j^2[k] \quad (25)$$

Where

$$G_j^1[k] = \sum_{n=0}^{j} M_k^1[j-n] \cos\left(n + \frac{1}{2}\right)\theta_{k1}, k = 1, 3 \ldots \mathrm{N}/2 - 1 \quad (26)$$

and

$$G_j^2[k] = \sum_{n=0}^{j} M_k^1[j-n] \left[sin\left(\left(n + \frac{1}{2}\right)\theta_{k1}\right)\right] \quad (26a)$$

With $j = \mathrm{N}/16 - 1$ and $\theta_{k1} = (4\pi\mathrm{k}/\mathrm{N})$

Using the following trigonometric identities in equation (12a) and(12b), recursive relation of equation (23) and (24) are

$$G_j^1[k] = \{M_k^1[j] - M_k^1[j-1]\} cos\left(\frac{\theta_{k1}}{2}\right) + 2G_{j-1}^1[k] \cos\theta_{k1} - G_{j-2}^1[k] \quad (27)$$

And

$$G_j^2[k] = \{M_k^1[j] + M_k^1[j-1]\} \sin\left(\frac{\theta_{k1}}{2}\right) + 2G_{j-1}^2[k] \cos\theta_{k1} - G_{j-2}^2[k] \quad (28)$$

Using equation (25), (26) and (22a), recursive formula for $H_{ooA}[k]$ is obtained.

Further for getting the recursive relation of equation (23b) by putting $n = \frac{N}{8} + n'$ and dropping the dashes in the term of the equation (23b)

$$H_{ooB}(k) = \sum_{n=0}^{N/8-1} W_{ko}\left[\frac{N}{8} + n\right] cas\left(\frac{\pi k}{2} + (2n+1)\frac{2\pi k}{N}\right), \mathrm{k} = 1, 3 \ldots \mathrm{N}/2 - 1 \quad (29)$$

Solving and expanding the cas term, equation (29) reduces to

$$\sum_{n=0}^{N/8-1} W_{ko}^1[n] cms\left((2n+1)\frac{2\pi k}{N}\right) \quad (30)$$

Where

$$W_{ko}^1[n] = (-1)^{\left(\frac{\mathrm{k}-1}{2}\right)} W_{ko}\left[\frac{N}{8} + n\right] \quad (30a)$$

Again, folding the sequence along n,

$$H_{ooB}(k) = \sum_{n=0}^{N/16-1} W_{ko}^1[n] cms\left((2n+1)\frac{2\pi k}{N}\right) + \sum_{n=N/16}^{N/8-1} W_{ko}^1[n] cms\left((2n+1)\frac{2\pi k}{N}\right) \quad (31)$$

where k = 1, 3 ... N/2-1

$$H_{ooB}(k) = \sum_{n=0}^{N/16-1} M_k^2[n] cms\left((2n+1)\frac{2\pi k}{N}\right), \text{ k = 1, 3 ... N/2-1} \quad (32)$$

Where $M_k^2[n] = \left[W_{ko}^1[n] + \left(-1^{\left(\frac{k+1}{2}\right)}\right) W_{ko}^1\left[\frac{N}{8} - n - 1\right]\right]$ (32a)

$$H_{ooB}(k) = cms\left(\frac{\pi k}{4}\right) \sum_{n=0}^{N/16-1} M_k^2\left[\frac{N}{16} - 1 - n\right] cos(2n+1)\frac{2\pi k}{N} + cas\left(\frac{\pi k}{4}\right) \sum_{n=0}^{N/16-1} M_k^2\left[\frac{N}{16} - 1 - n\right] sin(2n+1)\frac{2\pi k}{N} \quad (33)$$

or

$$H_{ooB}(k) = cms\left(\frac{\pi k}{4}\right) G_p^1[k] + cas\left(\frac{\pi k}{4}\right) G_p^2[k], \text{ k = 1, 3 ... N/2-1} \quad (34)$$

After folding the sequence of part B, we will now be building its recursive formula for its recursive structure

$$G_p^1[k] = \{M_k^2[j] - M_k^2[j-1]\} cos\left(\frac{\theta_{k1}}{2}\right) + 2G_{p-1}^1[k] \cos\theta_{k1} - G_{p-2}^1[k] \quad (35)$$

$$G_p^2[k] = \{M_k^2[j] + M_k^2[j-1]\} \sin\left(\frac{\theta_{k1}}{2}\right) + 2G_{p-1}^2[k] \cos\theta_{k1} - G_{p-2}^2[k] \quad (36)$$

After deriving the algorithm, equations (13),(15),(20),(27),(28),(35)and (36) are recursive in nature and these equations can be realized through Infinite Impulse Response (IIR) structure. The structure corresponding to these equations will be discussed next section.

III. Recursive Structure

The recursive structure for the above algorithm is presented in Fig 1. A common structure has been presented along with the variations in the variables Input (In), α, β, γ and θ_k, as represented in table 1. We have developed 5 recursive structures in total to represent our algorithm. (as shown in equation (13), (15), (20), (27), (28), (35) and (36)).

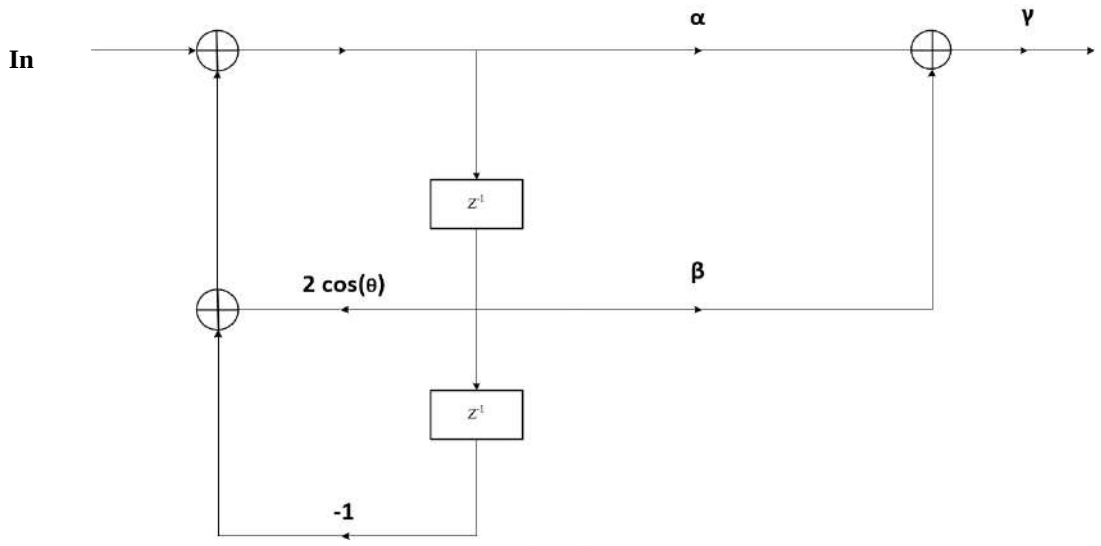

Fig 1. Generalized recursive structure

S. No.	Recursive Structure	In	α	β	γ	θ
1	A	$W_{ee}[n]$	$cms\theta_k$	-1	1	$\theta_k = {8\pi k}/{N}$
2	B	$W_{eo}[n]$	$cms\left(\frac{\theta_k}{2}\right)$	$-cas\left(\frac{\theta_k}{2}\right)$	1	$\theta_k = {8\pi k}/{N}$
3	C	$M_k[n]$	$cms\left(\frac{\theta_k}{2}\right)$	$-cas\left(\frac{\theta_k}{2}\right)$	-1	$\theta_{k1} = {4\pi k}/{N}$
4	G_j^1	$M_k^1[n]$	1	-1	$cos\left(\frac{\theta_{k1}}{2}\right)$	$\theta_{k1} = {4\pi k}/{N}$
5	G_j^2	$M_k^1[n]$	1	1	$sin\left(\frac{\theta_{k1}}{2}\right)$	$\theta_{k1} = {4\pi k}/{N}$
6	G_p^1	$M_k^2[n]$	1	-1	$cos\left(\frac{\theta_{k1}}{2}\right)$	$\theta_{k1} = {4\pi k}/{N}$
7	G_p^2	$M_k^2[n]$	1	1	$sin\left(\frac{\theta_{k1}}{2}\right)$	$\theta_{k1} = {4\pi k}/{N}$

Table 1

IV. Architecture Design

Based on the proposed recursive algorithm, a general architecture for a sequence of length N=32 has been presented. Fig. 2 represents the architecture for odd parts of the original sequence (H_o). The architecture comprises two stages, the pre-processing stage and the filter bank stage. The pre-processing stage/unit is composed of two add/sub-stage and a multiplexer stage. The input sequence when divided into H_e and H_o (shown in equation (2) and (3)), and the odd part is further divided into H_{oe} and H_{oo} in 'k' (equation (17) and (21)) at the multiplexer stage. Starting from H_o, the add/sub stage's purpose is to perform all the necessary additions and subtractions of the input data to fold the input sequence, and the multiplexer stage then selects the data to be processed further, based on whether k/2 is even or odd (for 8 X 1 MUX) and (k-1)/2 and (k+1)/2 is even or odd for (4 X 1 MUX) and sends the data further to the filter bank unit. Once the filter bank receives the data from the multiplexers, it produces the individual outputs from H_{ooA} (equation (27) and (28)) and H_{ooB} (equation (35) and (36)) and H_{oe} from (8 X1 MUX), and after performing the summation of these two outputs, the output H_o is computed.

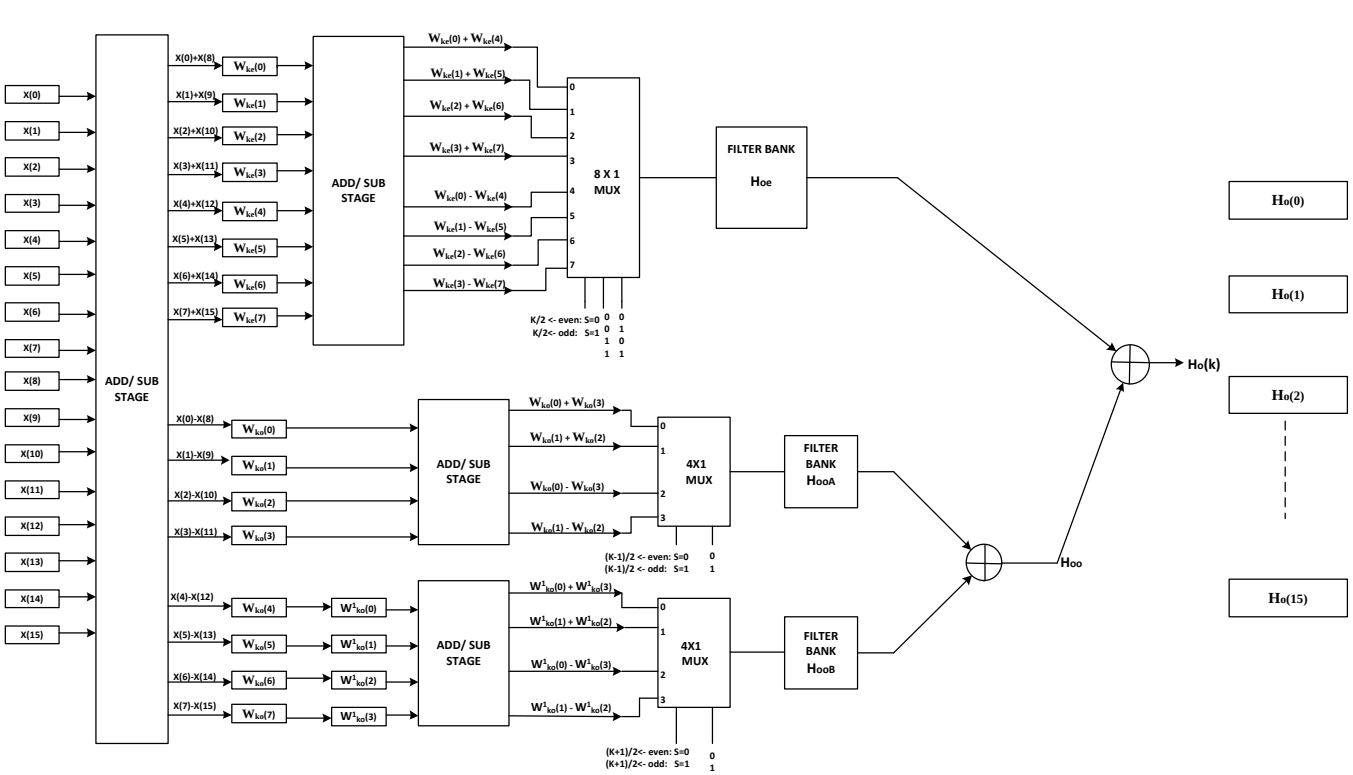

Fig 2. Architecture representing the processing of odd values of original sequence (H_o) (N=32)

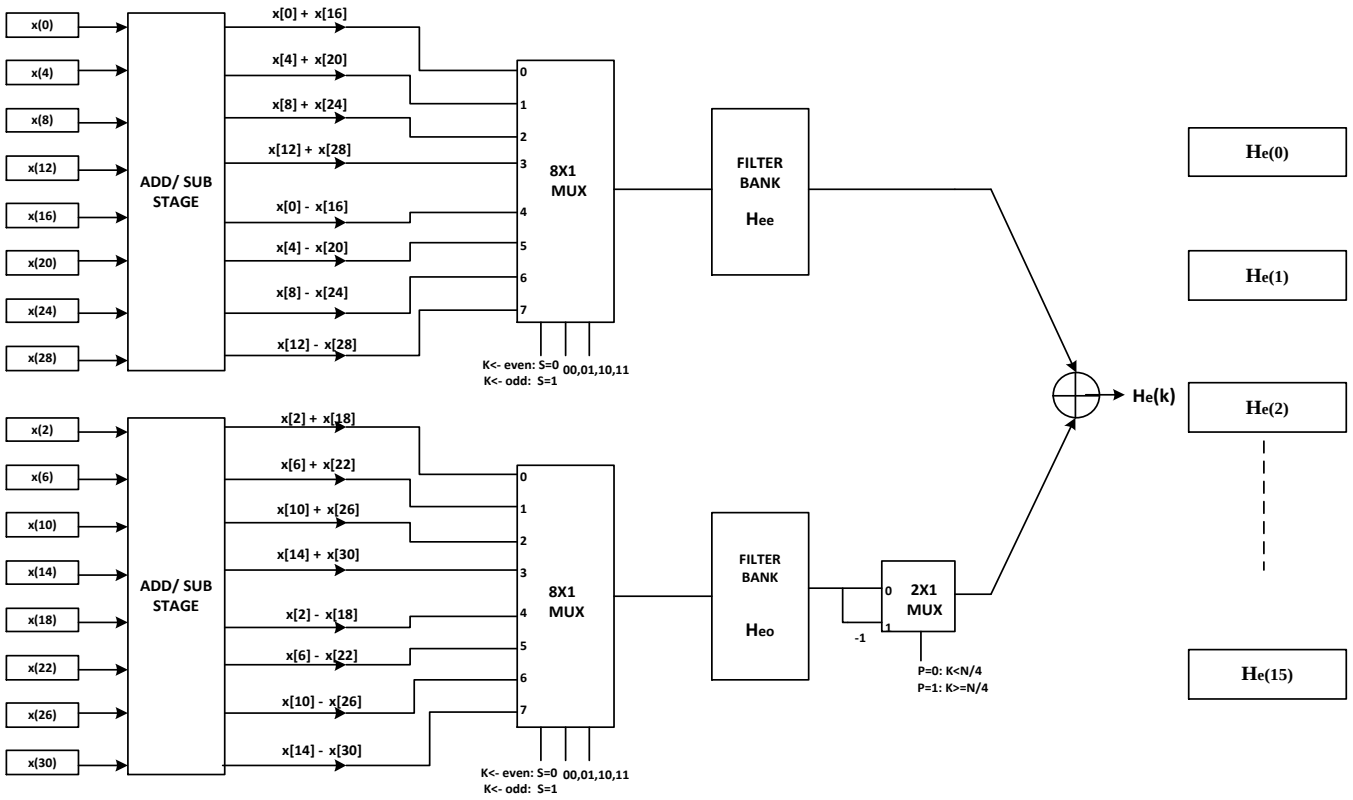

Fig. 3. Architecture representing the processing of even values of original sequence (H_e)

In reference to Fig. 3, for the even input data of the original sequence H_e, a similar approach is used. Here the even length sequence is divided into two sequences of length 8, based on (H_{ee} and H_{eo}) (as shown in equation (6)), and after pre-processing through an add/sub-stage and multiplexer, it is passed on to the filter bank. For H_{eo}, a (2 X 1 MUX) is used to select input based on the value of k (equation (15)), and the respective filter bank outputs are then added to produce an output H_e.

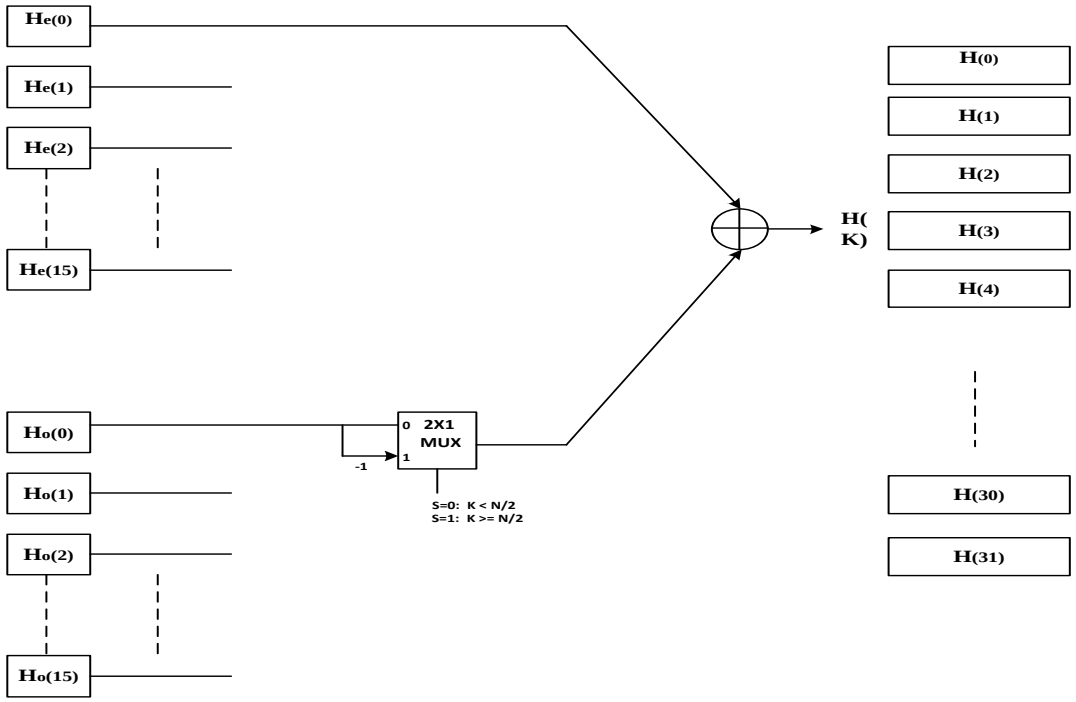

Fig 4. Architecture representing the processing of both H_o and H_e during final addition

In Fig. 4, the final stage, output H_o is passed through a(2X1 MUX), and the selected input values based on the value of k are summated with H_e and the final value of the DHT, H[k] is computed for all k. The proposed architecture requires only adders and subtractors and a few MUX for processing of the algorithm, which reduces the hardware complexity.

V. HARDWARE COMPLEXITY

Length(N)	Proposed	[9]	[12]	[13]	[14]
16	15	35	32	78	67
32	15	67	64	158	205
64	15	131	128	318	553

Table 2. Comparison of algorithm proposed, in terms of number of adders required for DHT computation

Length(N)	Proposed	[9]	[12]	[13]	[14]
16	14	34	64	64	12
32	14	66	128	128	40
64	14	130	256	256	112

Table 3. Comparison of algorithm proposed, in terms of number of multipliers required for DHT computation

VI. Comparison and Discussion

The proposed methodology for computing Discrete Hartley Transform offers better hardware complexity in terms of number of adders and multipliers as compared to [9][12][13][14], as shown in table 2 and 3. The number of adders and multipliers used are significantly lower, and remain constant for different values of N (where $N = 2^m$ and m >=4), making it a feasible, multipurpose, and less bulky hardware structure, but at the cost of higher computational complexity. As discussed in literature, [1] and [19] have proposed FHT algorithms, with which we cannot compare the proposed recursive algorithm's hardware complexity, but since FHT algorithm have cross connections, it is undesirable for high dataset as it becomes highly complex for VLSI implementation. The proposed structure however, requires no complex multiplications and since it does not have any cross-connections, therefore it is highly suitable for VLSI implementation. It can be seen from equations (15) and (20), that the recursive structure for H_{eo} and H_{oe} is similar, therefore only one structure can be used for computing both, giving us a more reduced hardware complexity by compromising the time complexity of the overall structure. Considering the computational complexity, the proposed structure requires more computations as compared to previous work [1], [19]. However, these papers have not proposed a recursive structure for computing DHT, which makes the proposed structure a better structure for real-life VLSI implementations.

VII. Conclusion

In conclusion, a new recursive algorithm for the computation of the Discrete Hartley Transform, along with a recursive structure has been proposed. The proposed divide and conquer algorithm's hardware complexity was found to be much lower, in comparison to the algorithms mentioned in the literature. The algorithm has been verified in appendix B, by simulating it in MATLAB Simulink. The proposed algorithm is also suitable due to its less complex parallel VLSI implementation and in future can also be realized on FPGA by using real time data.

Appendix A

$$H_{ee}[k] = \sum_{n=0}^{N/8-1} x[4n] cas\left((4n)\frac{2\pi k}{N}\right) + \sum_{n=0}^{N/8-1} x\left[4\left(n+\frac{N}{8}\right)\right] cas\left(4\left(n+\frac{N}{8}\right)\frac{2\pi k}{N}\right) \quad \text{(A1)}$$

where k = 0, 1, 2 ... N/4-1

Solving the right term of equation (A1) and expanding the cas term, we obtain,

$$\sum_{n=0}^{N/8-1} x\left[4\left(n+\frac{N}{8}\right)\right]\left(\cos\left(\pi k+(4n)\frac{2\pi k}{N}\right)+\sin\left(\pi k+(4n)\frac{2\pi k}{N}\right)\right) \quad \text{(A2)}$$

where k = 0, 1, 2 ... N/4-1

Using the following identities in equation (A2):

$$\sin(A+B)=\sin A\cos B+\cos A\sin B;\ \cos(A+B)=\cos A\cos B-\sin A\sin B$$

$$\sum_{n=0}^{N/8-1} x\left[4\left(n+\frac{N}{8}\right)\right]\left(\begin{array}{l}\cos\left((4n)\frac{2\pi k}{N}\right)\cos(\pi k)-\sin\left((4n)\frac{2\pi k}{N}\right)\sin(\pi k)\\ +\sin\left((4n)\frac{2\pi k}{N}\right)\cos(\pi k)+\cos\left((4n)\frac{2\pi k}{N}\right)\sin(\pi k)\end{array}\right) \quad \text{(A3)}$$

where k = 0, 1, 2 ... N/4-1 and $\sin(\pi k)=0;\ \cos(\pi k)=(-1^k)$

Equation (A3) reduces to

$$\sum_{n=0}^{N/8-1} x\left[4\left(n+\frac{N}{8}\right)\right]\left((-1)^k\left(cas\left((4n)\frac{2\pi k}{N}\right)\right)\right) \quad \text{(A4)}$$

where k = 0, 1, 2 ... N/4-1

Putting the equation (A4) back in equation (A3):

$$H_{ee}[k]=\sum_{n=0}^{N/8-1} x[4n]cas\left((4n)\frac{2\pi k}{N}\right)+\sum_{n=0}^{N/8-1} x\left[4\left(n+\frac{N}{8}\right)\right](-1)^k cas\left((4n)\frac{2\pi k}{N}\right) \quad \text{(A5)}$$

where k = 0, 1, 2 ... N/4-1

Therefore, equation (A5) becomes

$$H_{ee}[k]=\sum_{n=0}^{N/8-1}\left[x(4n)+(-1)^k x\left(4\left(n+\frac{N}{8}\right)\right)\right]\left(cas\left((4n)\frac{2\pi k}{N}\right)\right) \quad \text{(A6)}$$

where k = 0, 1, 2 ... N/4-1

Assume $W_{ee}[n]=x(4n)+(-1)^k x\left(4\left(n+\frac{N}{8}\right)\right)$

(A7)

$$H_{ee}[k] = \sum_{n=0}^{N/8-1} W_{ee}[n] cas\left((4n)\frac{2\pi k}{N}\right)$$

where k = 0, 1, 2 ... N/4-1

Put $n = \frac{N}{8} - 1 - n$

(A8)

$$H_{ee}[k] = \sum_{n=0}^{N/8-1} W_{ee}\left[\frac{N}{8} - 1 - n\right] cas\left(\pi k - 4(n+1)\frac{2\pi k}{N}\right)$$

where k = 0, 1, 2 ... N/4-1

Using trigonometric identities used above in:

$$\sum_{n=0}^{N/8-1} W_{ee}\left[\frac{N}{8} - 1 - n\right] x \begin{pmatrix} \cos(\pi k)\, cos\, 4(n+1)\frac{2\pi k}{N} + \sin(\pi k)\, sin\, 4(n+1)\frac{2\pi k}{N} \\ + \sin(\pi k)\, cos\, 4(n+1)\frac{2\pi k}{N} - \cos(\pi k)\, sin\, 4(n+1)\frac{2\pi k}{N} \end{pmatrix}$$

(A9)

Where k = 0, 1, 2 ... N/4-1

In equation (A9), $\sin(\pi k) = 0$, $\cos(\pi k) = (-1^k)$ and $cms\theta = \sin\theta - \cos\theta$

(A10)

$$H_{ee}[k] = (-1)^k \sum_{n=0}^{N/8-1} W_{ee}\left[\frac{N}{8} - 1 - n\right] cms\left(4(n+1)\frac{2\pi k}{N}\right)$$

where k = 0, 1, 2 ... N/4-1

(A11)

$$H_{ee}[k] = (-1)^k . G_i[k]$$

where k = 0, 1, 2 ... N/4-1

Let $i = \frac{\mathrm{N}}{8} - 1$ and $\theta_k = \left(\text{“}\frac{8\pi k}{N}\text{”}\right)$

(A12)

$$A_i[k] = \sum_{n=0}^{i} W_{ee}[i-n]\, cms\big((n+1)\theta_k\big)$$

where k = 0, 1, 2 ... N/4-1

(A13)

$$A[k] = \sum_{n=0}^{i} W_{ee}[i-n]\left[\cos\big((n+1)\theta_k\big) - \sin\big((n+1)\theta_k\big)\right]$$

where k = 0, 1, 2 ... N/4-1

Using the following trigonometric identities in equation (A13),

$$\cos(r\theta) = 2\cos[(r-1)\theta]\cos\theta - \cos[(r-2)\theta]$$

$$\sin(r\theta) = 2\sin[(r-1)\theta]\cos\theta - \sin[(r-2)\theta]$$

(A14)

$$G_i[k] = \sum_{n=0}^{i} W_{ee}[i-n]\begin{bmatrix}\big(2\cos(n\,\theta_k)\cos\theta_k - \cos\big((n-1)\theta_k\big)\big)\\ -\big(2\sin(n\,\theta_k)\cos\theta_k - \sin\big((n-1)\theta_k\big)\big)\end{bmatrix}$$

where k = 0, 1, 2 ... N/4-1

(A15)

$$A_i[k] = 2\cos\theta_k \sum_{n=0}^{i} W_{ee}[i-n]\,cms(n\,\theta_k) - \sum_{n=0}^{i} W_{ee}[i-n]\,cms\big((n-1)\theta_k\big)$$

where k = 0, 1, 2 ... N/4-1

(A16)

$$A_i[k] = \left[\begin{array}{c} 2\cos\theta_k\left[W_{ee}[i] + \sum_{n=0}^{i-1} W_{ee}[i-1-n]\,cms\big((n+1)\theta_k\big)\right] \\ -\left[\begin{array}{l} W_{ee}[i]\,cms(-\theta_k) + W_{ee}[i-1].(1) \\ +\sum_{n=0}^{i-2} W_{ee}[i-2-n]\,cms\big((n+1)\theta_k\big)\end{array}\right]\end{array}\right]$$

where k = 0, 1, 2 ... N/4-1

From equation (A16), we can write

(A17)

$$A_{i-1}[k] = \sum_{n=0}^{i-1} W_{ee}[i-1-n]\,cms\big((n+1)\theta_k\big)$$

Where k = 0, 1, 2 … N/4-1

(A18)

$$A_{i-2}[k] = \sum_{n=0}^{i-2} W_{ee}[i-2-n]\,cms\big((n+1)\theta_k\big)$$

Where k = 0, 1, 2 … N/4-1

Using equation. (A17) and equation (A18) in equation (A16), we obtain a recursive from as:

$$A_i[k] = 2\cos\theta_k \left[W_{ee}[i] + A_{i-1}[k] \right] - \begin{bmatrix} W_{ee}[i]\, cms(-\theta_k) \\ + W_{ee}[i-1] \,.\, (1) + G_{i-2}[k] \end{bmatrix} \quad \text{(A19)}$$

where k = 0, 1, 2 ... N/4-1

$$A_i[k] = W_{ee}[i]\cos(\theta_k) - W_{ee}[i]\sin(\theta_k) - W_{ee}[i-1] + 2\, A_{i-1}[k]\cos\theta_k - A_{i-2}[k] \quad \text{(A20)}$$

where k = 0, 1, 2 ... N/4-1

APPENDIX B

B. Computation of DHT from a given sequence x(n) using MATLAB 9.12.0 version

Given a sequence of length N =32,

x(n) = [0 1 2 3 4 5 …. 31]

n = 0,1,2, … 31

The desired H[k] computed using the equation (1) is

H[k] = [496, -178.4507, -96.4374, -68.7449, -54.6274, -45.9339, -39.9457, -35.4961, -32, -29.1309, -26.6909, -24.5522, -22.6274, -20.8535, -19.1826, -17.5759, -16, -14.4241, -12.8174, -11.1465, -9.3726, -7.4478, -5.3091, -2.8691, 0, 3.4961, 7.9457, 13.9339, 22.6274, 36.7449, 64.4374, 146.4507] (B1)

k = 0,1,2, ... 31

B.1 Computation of Even part (H_e), shown in equation (2) using MATLAB Simulink

Using equation (13), we have computed the values for recursive structure (1) (from table 1) by simulating it on Simulink.

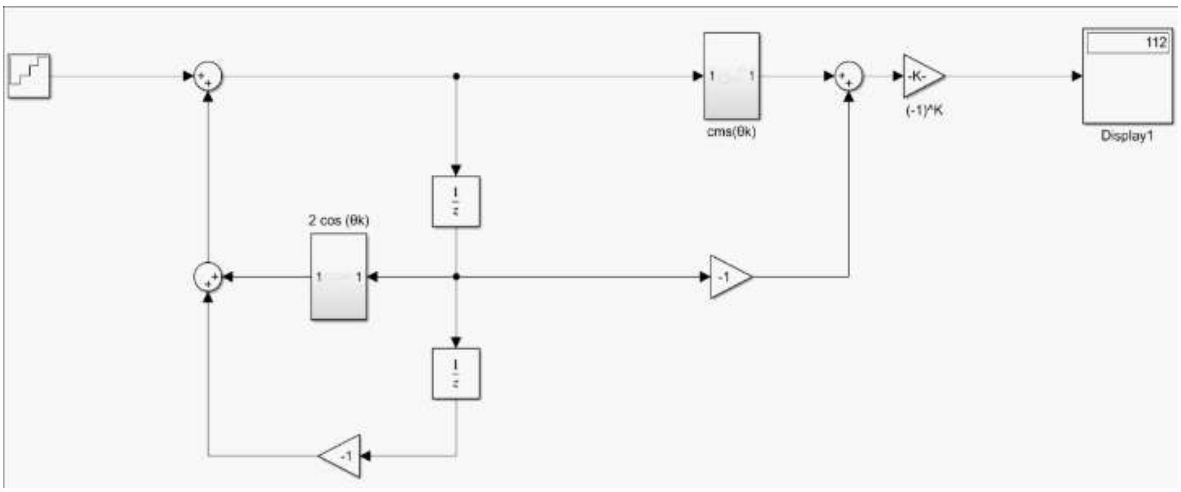

Fig. 5 Showing the output for recursive structure (1) for H_{ee} by simulating it on Simulink

Similarly, all the other values can be computed by running the respective structures in Simulink for different values of k.

$H_{ee}(k) = [112, -54.6274, -32, -22.6274, -16, -9.3726, 0, 22.6274]$ (B2)

k = 0, 1, 2 … 7

Similarly, using equation (15), for recursive structure (2)

$H_{eo}(k) = [128, -41.8100, -22.6274, -17.3183, -16, -17.3183, -22.6274, -41.8100]$ (B3)

k = 0, 1, 2 … 7

By combining (B2) and (B3), we get H_e as,

$H_e(k) = [240, -96.4374, -54.6274, -39.9457, -32, -26.6909, -22.6274, -19.1829,$ (B4)
$-16, -12.8174,$
$-9.3726, -5.3091, 0, 7.9457, 22.6274, 64.4374]$

k = 0, 1, 2 … 7

B.2 Computation of odd part (H_o), shown in equation (3) using MATLAB Simulink

Using equation (20), we have computed the values for recursive structure (3) by simulating it on Simulink.

$H_{oe}(k) = [256, -41.8100, -22.6274, -17.3183, -16, -17.3183, -22.6274, -41.8100]$ (B5)

k = 0, 2, 4 … 14

Similarly, by using equation (27) and (28) for part A; recursive structure (4)

$H_{ooA}(k) = [-82.0133, 0, -19.2430, 0, -16.3135, 0, 028.7992, 0]$ (B6)

k = 1, 3, 5 … 15

And equation (35) and (36) for part B; recursive structure (5)

H_{ooB} (k) = [0, -28.7992, 0, -16.3134, 0, -19.2430, 0, -82.0132] (B7)

k = 1, 3, 5 … 15

By combining (B5), (B6) and (B7), we get H_o as,

H_o (k) = [256, -82.0133, -41.8100, -62.5731, -22.6274, -19.2430, -17.3183, 3.4448, -16, -16.3135, -17.3138, -27.9366, -22.6274, -28.7992, -41.8100, 87.0648] (B8)

k = 0, 1, 2, … 15

B.3 By combining H_o and H_e, the value of DHT H[k] computed by following the proposed structure is found to be:

H[k] = [496, -178.4507, -96.4374, -68.7449, -54.6274, -45.9339, -39.9457, -35.4961, -32, -29.1309, -26.6909, -24.5521, -22.6274, -20.8535, -19.1826, -17.5758, -16, -14.4241, -12.8174, -11.1465, -9.373, -7.4479, -5.3091, -2.8695, 0, 3.4961, 7.9457, 13.9339, 22.6274, 36.7449, 64.4374, 146.4506] (B9)

k = 0, 1, 2, …31

It can be seen from equation B9, that the DHT coefficients computed from the proposed recursive structures, matches with the DHT result computed by following the conventional method, computed in equation B1.

REFERENCES

[1] Hamood, Mounir Taha. "New Efficient Algorithm for the Discrete Hartley Transform." IOP Conference Series: Materials Science and Engineering 978 (2020)

[2] Chiper, Doru Florin. "Fast Radix-2 Algorithm for the Discrete Hartley Transform of Type II." in IEEE Signal Processing Letters, vol. 18, no. 11 (Nov. 2011)

[3] Pyrgas, Lampros. Kitsos, Paris and Skodras, Athanassios. "Compact FPGA Architectures for the Two-Band Fast Discrete Hartley Transform." Microprocessors and Microsystems (2018)

[4] Hamood, Mounir Taha. "New Decimation-In-Time Fast Hartley Transform Algorithm." International Journal of Electrical and Computer Engineering (2016)

[5] Bracewell, R.N. "Discrete Hartley Transform." Journal of the Optical Society of America (1983)

[6] Nascimento, Francisco. "Hartley Transform Signal Compression and Fast Power Quality Measurements for Smart Grid Application." IEEE Transactions on Power Delivery (2023)

[7] Mardan, Suha Suliman and Hamood, Mounir Taha. "New Fast Walsh–Hadamard–Hartley Transform Algorithm." International Journal of Electrical and Computer Engineering (IJECE) (2023)

[8] Wu, Menglong. Xie, Yongfa. Shi, Yongchao. Zhang, Jianwen. Yao, Tianao and Liu, Wenkai. "An Adaptively Biased OFDM Based on Hartley Transform for Visible Light Communication Systems." IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences (2023)

[9] Fang, Wen-Hsien and Lee, J-D. “Efficient CORDIC-based Systolic Architectures for the Discrete Hartley Transform.” Computers and Digital Techniques, IEE Proceedings (1995)

[10] Bracewell, R.N. “The Fast Hartley Transform.” Proceedings of the IEEE 72 (1984)

[11] Murty, M.N. “Novel Recursive Algorithm for Realization of One-Dimensional Discrete Hartley Transform.” International Journal of Research and Reviews in Applied Sciences (2012)

[12] Chang, Long-Wen and Lee, Shen-Wen. “Systolic Arrays for the Discrete Hartley Transform.” in IEEE Transactions on Signal Processing (Nov. 1992)

[13] Liu, K. J. Ray and Chiu, Ching-Te. “Unified Parallel Lattice Structures for Time-Recursive Discrete Cosine/Sine/Hartley Transforms.” in IEEE Transactions on Signal Processing (March 1993)

[14] Chiper, Doru Florin. “A Novel VLSI DHT Algorithm for a Highly Modular and Parallel Architecture.” in IEEE Transactions on Circuits and Systems II: Express Briefs (May 2013)

[15] Bi, Guoan. “A Split Radix Algorithm of Discrete Hartley Transform.” Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference, Brisbane, QLD, Australia (1994)

[16] Bi, Guoan. “New Split-Radix Algorithm for the Discrete Hartley Transform.” in IEEE Transactions on Signal Processing (Feb. 1997)

[17] Bouguezel, S. Ahmad, M.O. and Swamy, M.N.S. “An Efficient Split-Radix FHT Algorithm.” IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512), Vancouver, BC, Canada (2004)

[18] Jiang, Longyu. Shu, Huazhong. Wu, Jiasong. Wang, Lu and Senhadji, Lotfi. “A Novel Split-Radix Fast Algorithm for 2-D Discrete Hartley Transform.” Circuits and Systems I: Regular Papers, IEEE Transactions on. 57. 911 – 924 (2010)

[19] Chiper, Doru Florin. “A Structured Dual Split-Radix Algorithm for the Discrete Hartley Transform of Length 2^N.” Circuits, Systems, and Signal Processing (2017)

Declarations:

Data availability statement: Not-applicable