A New Fast Radix-2 DIF Algorithm and Architecture for Computing the DHT

Gautam A. Shah
Department of Electronics and Telecommunications
St. Francis Institute of Technology, Mumbai University
Mumbai, India
e-mail: gautamshah@ieee.org

Tejmal S. Rathore
Department of Electronics and Telecommunications
St. Francis Institute of Technology, Mumbai University
Mumbai, India
e-mail: tsrathor@ee.iitb.ac.in

***Abstract* — The radix-2 decimation-in-time (DIT) fast Hartley transform algorithm for computing the Discrete Hartley Transform (DHT) was introduced by Bracewell. DIT and decimation-in-frequency (DIF) algorithms were further developed by Meckelburg and Lipka, Prado, Sorenson et al, Kwong and Shiu. In these algorithms, the stage structures perform all the additions and multiplications and utilize stage dependent sine and cosine coefficients. A new fast radix-2 DIF algorithm for computing the DHT is proposed, which introduces multiplying structures in the signal flow diagram that perform all the multiplications with the stage independent cosine coefficients and their related additions leading to simplification of the stage structures which now have to perform only the additions. This leads to a reduction in the number of multiplications. An architecture utilizing current feedback operational amplifiers which implements the algorithm in hardware has been proposed. It has been tested by simulating it with the help of PSpice.**

Keywords – algorithm, analog architecture, decimation-in-frequency, discrete Hartley transform, radix-2

I. INTRODUCTION

The fast Hartley transform (FHT) algorithm introduced by Bracewell [1] performs the DHT in a time proportional to $N \log_2 N$ using decimation-in-time (DIT). Meckelburg and Lipka presented the decimation-in-frequency (DIF) FHT algorithm [2] claiming it to be faster than the one in [1]. Sorenson et al. [3] using the index mapping approach analyzed the FHT having the same decomposition as [1], implemented the algorithms for both DIT and DIF, and verified their operational complexities to be the same. Prado [4] presented an in-place version of the FHT in [1] along with its operational complexity. Kwong and Shiu [5] restructured the signal flow diagram originally proposed in [1] for clarity, and applied the transposition theorem to obtain the DIF algorithm with the same operational complexity. The above approaches utilize both the cosine coefficients (CCs) and sine coefficients (SCs) which are stage-dependent. Hou [6] stressed that the FHT algorithm, in essence, is a generalization of the Cooley-Tukey fast Fourier transform (FFT) algorithm to compute the discrete Fourier transforms (DFT), but it requires only real arithmetic computations as compared to complex arithmetic operations in any standard FFT. Hao [7] examined both the pre- and post-permutation algorithms in [1] and [2], and suggested improvements to make them faster by use of fast rotation to reduce the multiplications and by incorporating in-place or distributed permutation. Malvar [8] presented a new factorization of the DHT which involves the discrete cosine transform (DCT). His algorithms minimize the multiplications at the expense of an increased number of additions. Rathore [9] reported that, for both the DIT in [1] and the DIF in [2], the operational complexity involved is the same. He further utilized the matrix approach, derived some properties of the DHT [10], obtained the relations for computational complexity and presented DHT-based DFT and DFT-based DHT algorithms.

Various architectures have been reported in the literature to compute the DHT. Chakrabarti and Jaja [11] proposed a modular bit-level systolic architecture. Dhar and Banerjee [12] employed a set of linear arrays of Givens rotors. Chang and Lee [13] derived two models of linear systolic arrays and suggested the use of cordic algorithms to make the systolic arrays more efficient in computation. Hsiao et al. [14] modified the above cordic processor and obtained a higher throughput and cost effective architecture. Kar and Rao [15] proposed a unified systolic architecture for sliding window computation of discrete transforms. Nayak and Meher [16] implemented a bit-level systolic architecture for discrete orthogonal transforms using a serial–parallel vector–matrix multiplication scheme based on the Baugh–Wooley algorithm. Guo [17, 18] presented two architectures; one using parallel adders and the other using a distributed arithmetic based array that utilizes identical ROM modules and eliminates the accumulation loop in the processing elements. Amira and Bouridane [19, 20] developed architectures to implement the DHT on field programmable gate arrays. Meher et al. [21] presented a design framework for scalable and modular memory based implementation of the DHT in systolic hardware. These architectures compute the DHT using digital VLSI techniques.

The role of analog integrated circuits in modern electronic systems remains important, even though digital circuits dominate the market for VLSI solutions. Analog systems have always played an essential role in interfacing digital electronics to the real world. An important advantage of digital integrated circuits has been their relative ease of design over analog circuits. In particular, since digital circuit

DOI 10.5013/IJSSST.a.12.01.01

ISSN: 1473-804x online, 1473-8031 print

design is amenable to automation, several CAD-compatible digital integrated circuit design methodologies have been developed, including design-for-testability, design optimization and rapid prototyping in field-programmable gate arrays (FPGAs). However, there are architectures which compute the DHT based on analog blocks. Culhane et al. [22] presented an analog circuit which utilizes a linear programming neural net to compute the DHT. Raut et al. [23] presented basic switched capacitor building blocks in systolic array architecture to implement the DFT. A two dimensional DCT structure proposed by Kawahito et al. [24] is designed with fully differential switched-capacitor circuits. Digitally controlled analog circuits proposed by Chen et al. [25] utilize the principle of charge scaling for computing the DCT and DFT. Mal and Dhar [26] proposed an analog sampled data architecture for the DHT.

The growing computational demand for complex information processing has motivated significant research in the design of power efficient signal processing systems. One method for achieving low-power designs is to move processing on system inputs from the digital processor to analog hardware. However, for analog systems to be desirable to digital signal processing engineers, they need to provide a significant advantage in terms of size and power and yet still remain relatively easy to use and integrate into a larger digital system. Reconfigurable analog arrays, dubbed field-programmable analog arrays (FPAAs), can speed the transition of systems from digital to analog by providing the ability to rapidly implement advanced, low-power signal processing systems [27]. This has demanded the development of high performance analog circuits that are reconfigurable and suitable for CAD methodologies. Analog circuits based on the current feedback operational amplifier (CFA) technique are suitable for high frequency applications [28]. The CFA combines high bandwidth and very fast large signal response with excellent dc performance. It is optimized for use in current to voltage applications and as an inverting mode amplifier. It can be used in place of traditional operational amplifiers (OA) and its current feedback architecture results in much better ac performance, high linearity and an exceptionally clean pulse response. Its closed-loop bandwidth is determined by the feedback resistor and is almost independent of the closed-loop gain unlike OA-based circuits, which are limited by a constant gain-bandwidth product. It can be used in ways similar to a conventional OA while providing performance advantages in wideband applications [29, 30].

The proposed algorithm overcomes the stage dependencies of the CCs and SCs. It utilizes only CCs which are stage independent. It introduces multiplying structures (MSs), and results in a signal flow diagram (SFD) with butterflies similar in each stage structure (SS). It leads to simplification of the stage computations and reduces the operational complexity. A simple and versatile basic analog circuit based on CFAs is designed, which can be easily reconfigured as a stage structure or multiplying structure circuit. An architecture which is modular and can be scaled for higher values of N is proposed to implement the algorithm.

II. DISCRETE HARTLEY TRANSFORM

An N-point DHT X_H of a sequence $x(n)$ is defined as

$$X_H(k) = \sum_{n=0}^{N-1} x(n)\,\mathrm{cas}\left(\frac{2\pi kn}{N}\right),\ k = 0, 1, \ldots, N-1$$

where cas (.) = cos (.) + sin (.). Using the matrix approach [10], the DHT can be expressed as

$$\begin{bmatrix} X_H(0) \\ X_H(1) \\ \vdots \\ X_H(N-1) \end{bmatrix} = \begin{bmatrix} h_{0,0} & h_{0,1} & \cdots & h_{0,N-1} \\ h_{1,0} & h_{1,1} & \cdots & \vdots \\ \vdots & \cdots & \cdots & \vdots \\ h_{N-1,0} & \cdots & \cdots & h_{N-1,N-1} \end{bmatrix} \cdot \begin{bmatrix} x(0) \\ \\ \\ x(N-1) \end{bmatrix}$$

In the expression $[X_H] = [H_N]\cdot[x]$, H_N is the $N \times N$ Hartley matrix and its elements are given by

$$h_{i,j} = \mathrm{cas}\left(\frac{2\pi ij}{N}\right) \tag{1}$$

where indices i and j are integers from 0 to $N-1$ [22].

The FHT algorithm by Bracewell [1] performs the DHT of a data sequence of N elements in a time proportional to $N\log_2 N$ where $N = 2^P$. The algorithm in [2] requires an operation count given by N_A additions and N_M multiplications

$$N_A = \frac{(3N\log_2 N - 3N + 4)}{2} \tag{2}$$

$$N_M = N\log_2 N - 3N + 4\,. \tag{3}$$

Sorenson et al [3] have analyzed the FHT algorithm in [1] using the index mapping approach. Their radix-2 DIT and DIF programs implement the FHT algorithm with the same operation count as given by (2) and (3). Prado [4] in his paper presents an in-place version of the FHT in [1] and gives a table for the number of non-trivial real operations which tally with those obtained by using (2) and (3). Kwong and Shiu [5] have restructured the signal flow diagram originally proposed in [1] for clarity, and applied the transposition theorem to obtain the DIF algorithm with the same operational complexity. However, Rathore [9] has observed that for all these radix-2 algorithms [1]-[5], the operational complexities involved are the same. The above approaches utilize both the cosine coefficients (CCs) and sine coefficients (SCs) which are stage-dependent. The proposed algorithm computes only stage independent CCs.

III. Proposed Algorithm

A view of the proposed algorithm operation may be obtained as a sequence of matrix operations on the data. The DHT

$$X_H = N^{-1} P_r L_P L_{(P-1)} L_{(P-2)M} L_{(P-2)} \cdots L_{2M} L_2 L_{1M} L_1 x(n),$$

where L_S is a SS matrix (SSM) and L_{SM} is a MS matrix (MSM).

For each SSM, there is a matrix L_Y having dimensions $Y \times Y$, where $Y = 2^{P+1-S}$ which repeats itself along the forward diagonal N/Y times. If L_Y is split into 4 quadrants $\begin{array}{c|c} Q_1 & Q_3 \\ \hline Q_2 & Q_4 \end{array}$, Q_1 Q_2 and Q_3 have the same elements and are identity matrices, whereas $Q_4 = -Q_1$. Fig. 1 shows the SSMs involved in the stages 1 to 4 for $N = 16$.

MSMs are introduced for the stages $1 \le S < (P-1)$. For each L_{SM}, matrix L_Z having dimensions $Z \times Z$, where $Z = 2^{P+1-S}$, repeats itself along the forward diagonal N/Z times. Splitting L_Z into 4 quadrants $\begin{array}{c|c} Q_1 & Q_3 \\ \hline Q_2 & Q_4 \end{array}$, only Q_1 and Q_4 have some non-zero elements. Q_1 is an identity matrix, whereas elements which appear in Q_4 are related to the CCs of the elements in the first row or column of H_N. The CCs appear along the forward diagonal of Q_4. They also appear along the inverse diagonal of the sub-matrix formed within Q_4 after deleting the 0th row and 0th column of Q_4. The intersection element of both these elements is a unity factor. Due to the cosine-sine symmetry, there is a reduction in the number of these elements to only a few CCs and a unity factor.

$$L_1 = \left[\begin{array}{c|c} I_8 & I_8 \\ \hline I_8 & -I_8 \end{array}\right] \quad (L_Y)$$

$$L_2 = \left[\begin{array}{c|c} \boxed{\begin{array}{c|c} I_4 & I_4 \\ \hline I_4 & -I_4 \end{array}}_{\,L_Y} & O_8 \\ \hline O_8 & L_Y \end{array}\right]$$

$$L_3 = \left[\begin{array}{c|c} \begin{array}{c|c} \boxed{\begin{array}{c|c} I_2 & I_2 \\ \hline I_2 & -I_2 \end{array}}_{\,L_Y} & O_4 \\ \hline O_4 & L_Y \end{array} & O_8 \\ \hline O_8 & \begin{array}{c|c} L_Y & O_4 \\ \hline O_4 & L_Y \end{array} \end{array}\right]$$

$$L_4 = \left[\begin{array}{c|c} \begin{array}{c|c} \begin{array}{c|c} \boxed{\begin{array}{c|c} I_1 & I_1 \\ \hline I_1 & -I_1 \end{array}}_{\,L_Y} & O_2 \\ \hline O_2 & L_Y \end{array} & O_4 \\ \hline O_4 & \begin{array}{c|c} L_Y & O_2 \\ \hline O_2 & L_Y \end{array} \end{array} & O_8 \\ \hline O_8 & \begin{array}{c|c} \begin{array}{c|c} L_Y & O_2 \\ \hline O_2 & L_Y \end{array} & O_4 \\ \hline O_4 & \begin{array}{c|c} L_Y & O_2 \\ \hline O_2 & L_Y \end{array} \end{array} \end{array}\right]$$

$I_N = N \times N$ Identity Matrix, $O_N = N \times N$ Null Matrix

Figure 1. Matrices L_S and L_Y for $N = 16$, $S = 1$ to 4.

The first stage has the maximum number of CCs. The number of CCs for the other stages is lesser and their values belong to the set of CCs for the first stage. While computing L_Z it is sufficient to compute Q_4. Hence, only $(N/4) - 1$ stage independent CCs have to be computed. Fig. 2 shows the MSMs involved in the stages 1 and 2 for $N = 16$.

The permutation matrix shown in Fig. 3 is involved in the computation and applied to the outputs of the final stage to rearrange them in the normal order.

A succession of P stage operations followed by permutation leads stage by stage to the outputs, $x_1(n)$ through $x_P(n)$, and finally to the transform output X_H. For any stage, SS performs only additions, and MS performs the multiplications with the CCs and the related additions.

$$L_{1M} = \left[\begin{array}{c|c} I_8 & O_8 \\ \hline O_8 & C_8 \end{array}\right] \quad (L_Z) \qquad I_N = N \times N \text{ Identity Matrix}, \quad O_N = N \times N \text{ Null Matrix}$$

$$L_{2M} = \left[\begin{array}{c|c} \boxed{\begin{array}{c|c} I_4 & O_4 \\ \hline O_4 & C_4 \end{array}}_{\,L_Z} & O_8 \\ \hline O_8 & L_Z \end{array}\right] \qquad C_4 = \begin{bmatrix} 1 & & & \\ & C_2 & & C_2 \\ & & 1 & \\ & C_2 & & -C_2 \end{bmatrix}$$

$$C_8 = \begin{bmatrix} 1 & & & & & & & \\ & C_1 & & & & & & C_3 \\ & & C_2 & & & & C_2 & \\ & & & C_3 & & C_1 & & \\ & & & & 1 & & & \\ & & & C_1 & & -C_3 & & \\ & & C_2 & & & & -C_2 & \\ & C_3 & & & & & & -C_1 \end{bmatrix}$$

Figure 2. Matrices L_{SM} and L_Z for $N = 16$, $S = 1, 2$.

$$P_r = \begin{bmatrix} 1&&&&&&&&&&&&&&& \\ &&&&&&&&1&&&&&&& \\ &&&&1&&&&&&&&&&& \\ &&&&&&&&&&&&1&&& \\ &&1&&&&&&&&&&&&& \\ &&&&&&&&&&1&&&&& \\ &&&&&&1&&&&&&&&& \\ &&&&&&&&&&&&&&1& \\ &1&&&&&&&&&&&&&& \\ &&&&&&&&&1&&&&&& \\ &&&&&1&&&&&&&&&& \\ &&&&&&&&&&&&&1&& \\ &&&1&&&&&&&&&&&& \\ &&&&&&&&&&&1&&&& \\ &&&&&&&1&&&&&&&& \\ &&&&&&&&&&&&&&&1 \end{bmatrix}$$

Figure 3. Permutation Matrix for $N = 16$.

DOI 10.5013/IJSSST.a.12.01.01

ISSN: 1473-804x online, 1473-8031 print

Figure 4 depicts a partial SFD showing a generalized SS and MS. The SFD for the SS is simple, follows a regular pattern and performs only additions. For each MS, the CCs α_i, β_i or 0.707 are multiplied with different elements as shown in Fig. 4. For each stage S from 1 to P - 2, there are CCs in the corresponding MS.
Elements from 0 to $m/2$ have no CCs.
Each element $(m/2) + i$ has no CC for $i = m/4$, the CC 0.707 for $i = m/8$, and CCs α_i and β_i for i = 1 to $(m/8) - 1$ and $(m/8) + 1$ to $(m/4) - 1$.
Each element $m - i$ has the CC 0.707 for $i = m/8$, and CCs $-\alpha_i$ and β_i for i = 1 to $(m/8) - 1$ and $(m/8) + 1$ to $(m/4) - 1$,

where $\alpha_i = \cos\frac{2\pi i}{m}$ and $\beta_i = \cos\frac{2\pi}{m}\left(\frac{m}{4} - i\right)$.

Further $\alpha_i = \beta_{\left(\frac{m}{4}-i\right)}$ and $\beta_i = \alpha_{\left(\frac{m}{4}-i\right)}$.

Hence, for all the MSs, only $(N/4) - 1$ stage independent CCs have to be computed.

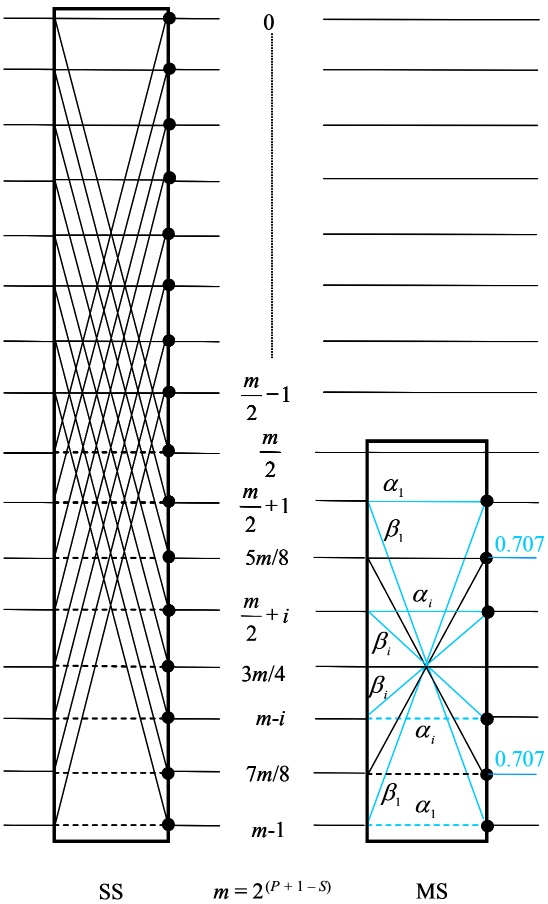


Figure 4. A SFD for the generalized structure, $N = 2^P$

In the proposed algorithm the operational complexity is calculated as follows:
The additions are performed by both the SS and MS.
Each SS requires N additions, for $N = 2^P$, the number of stages are P, hence, total number of additions for all the SSs $= NP$.

Number of MSs required per stage $= \frac{N}{m}$, where $m = 2^{(P+1-S)}$.

Each MS requires $(m/2) - 2$ additions.

Hence additions per stage $= \left(\frac{m}{2} - 2\right)\frac{N}{m} = \frac{N}{2} - \frac{2N}{m}$.

The number of additions for all the MSs are

$$\sum_{S=1}^{P-2}\left(\frac{N}{2} - \frac{2N}{2^{P+1-S}}\right).$$

For the entire SFD including all the SSs and MSs

$$N_A = NP + \sum_{S=1}^{P-2}\left(\frac{N}{2} - \frac{N}{2^{P-S}}\right) = \frac{(3N\log_2 N - 3N + 4)}{2} \tag{4}$$

It is clear from (2) and (4) that N_A remains the same.
The multiplications are performed within the MSs.
Each MS requires $m - 6$ multiplications.

Hence multiplications per stage $= (m-6)\frac{N}{m} = N - \frac{6N}{m}$.

The number of multiplications for the entire SFD are

$$N_M = \sum_{S=1}^{P-2}\left(N - \frac{3N}{2^{P-S}}\right) = N\log_2 N - 3.5N + 6 \tag{5}$$

From (3) and (5) it is seen that N_M is lesser for $N \geq 8$ in the proposed algorithms as compared to the existing algorithms in [1] – [5]. It is evident that the number of non-trivial arithmetic operations is reduced by 2 multiplications (M) for each MS introduced. The reduction of $2M$ at the first stage is due to one MS corresponding to stage 1 and a reduction of $4M$ at the second stage is due to two MSs corresponding to stage 2. As the values of P and N increase, the MSs for the corresponding stages also increase, leading to a further reduction in the M. The total number of M reduces by $\frac{N-4}{2}$ for $N \geq 8$. The comparison of the operational complexities of the existing radix-2 algorithms with the proposed algorithm for various transform lengths is shown in Table I.

TABLE I. COMPARISON OF OPERATIONAL COMPLEXITIES

Length	Radix-2 FHT algorithms [1]-[5]			Proposed Radix-2 Algorithm		
N	N_M	N_A	*Total*	N_M	N_A	*Total*
8	4	26	30	2	26	28
16	20	74	94	14	74	88
32	68	194	262	54	194	248
64	196	482	678	166	482	648
128	516	1154	1670	454	1154	1608
256	1284	2690	3974	1158	2690	3848
512	3076	6146	9222	2822	6146	8968
1024	7172	13826	20998	6662	13826	20488
2048	16388	30722	47110	15366	30722	46088
4096	36868	67586	104454	34822	67586	102408

Fig. 5 shows the SFD for the proposed algorithm with N = 16, P = 4. It is regular and well structured. The first stage consists of the SS and MS. The matrix for L_1 shown in Fig. 1 is directly mapped as SS and that for L_{1M} shown in Fig. 2 is directly mapped as MS for stage 1 in the SFD. This stage has the maximum number of stage independent CCs which should be equal to $(N/4) - 1$. As $N = 16$, these should be 3. Although in the SFD it appears to be more, as $\alpha_1 = \beta_3$ and $\alpha_3 = \beta_1$, they may be treated as α_1, β_1, and 0.707.

The second stage also consists of the SS and MS. The matrix for L_2 shown in Fig. 1 is directly mapped as SS and that for L_{2M} shown in Fig. 2 is directly mapped as MS for stage 2 in the SFD. This stage has only one stage independent CC 0.707 which belongs to the set of CCs for the first stage.

The third and fourth stages consist only of the SSs. The matrices for L_3 and L_4 shown in Fig. 1 are directly mapped as SSs for stages 3 and 4 respectively. The fourth stage followed by permutation results in the transformed output.

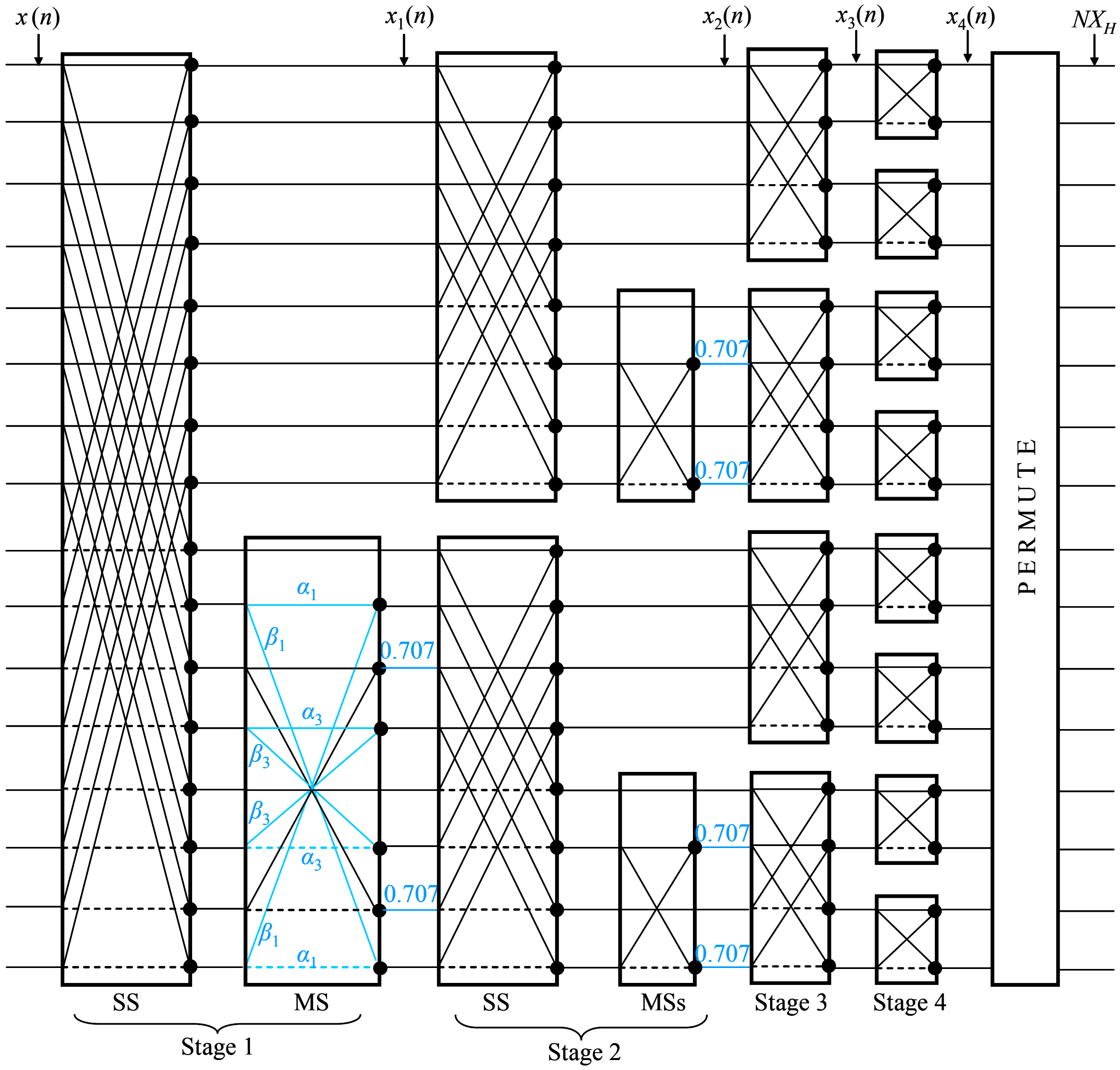


Figure 5. SFD for proposed algorithm with N = 16, P = 4.

IV. CFA BASED ANALOG CIRCUIT

Consider the circuit shown in Fig. 6.

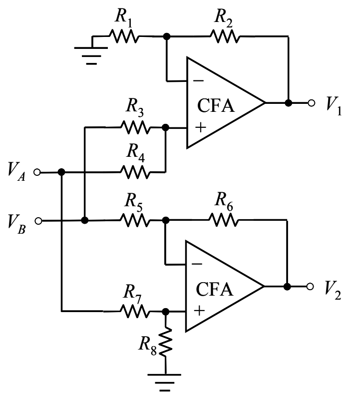


Figure 6. Basic analog circuit

The outputs are

$$V_1 = \left(\frac{R_1 + R_2}{R_3 + R_4}\right)\left[\frac{R_3}{R_1}V_A + \frac{R_4}{R_1}V_B\right] \tag{6}$$

$$\text{and } V_2 = \frac{R_6}{R_5}\left(\frac{1+\dfrac{R_5}{R_6}}{1+\dfrac{R_7}{R_8}}\right)V_A - \frac{R_6}{R_5}V_B \tag{7}$$

Thus, the circuit acts as a weighted summer and subtractor.

Case (i): Choosing

$$R_1 = R_2 = R_3 = R_4 \text{ and } \frac{R_6}{R_5} = \frac{R_8}{R_7} = 1,$$

$$V_1 = V_A + V_B \tag{8}$$

$$\text{and } V_2 = V_A - V_B. \tag{9}$$

Thus the circuit may be utilized in the stage structure.

Case (ii): Choosing

$$\frac{R_1 + R_2}{R_3 + R_4} = 1,\ \frac{R_3}{R_1} = \frac{R_6}{R_5} = \alpha_i,\ \frac{R_4}{R_1} = \beta_i$$

$$\text{and } \frac{R_7}{R_8} = \left(\frac{\alpha_i + 1}{\beta_i}\right) - 1,$$

$$V_1 = \alpha_i V_A + \beta_i V_B \tag{10}$$

$$\text{and } V_2 = \beta_i V_A - \alpha_i V_B. \tag{11}$$

Thus the circuit may be utilized in the multiplying structure.

V. ANALOG ARCHITECTURE

An architecture to obtain the radix-2 DHT for N = 2, 4, 8 and 16 shown in Fig. 7 have been implemented utilizing the basic analog circuit.

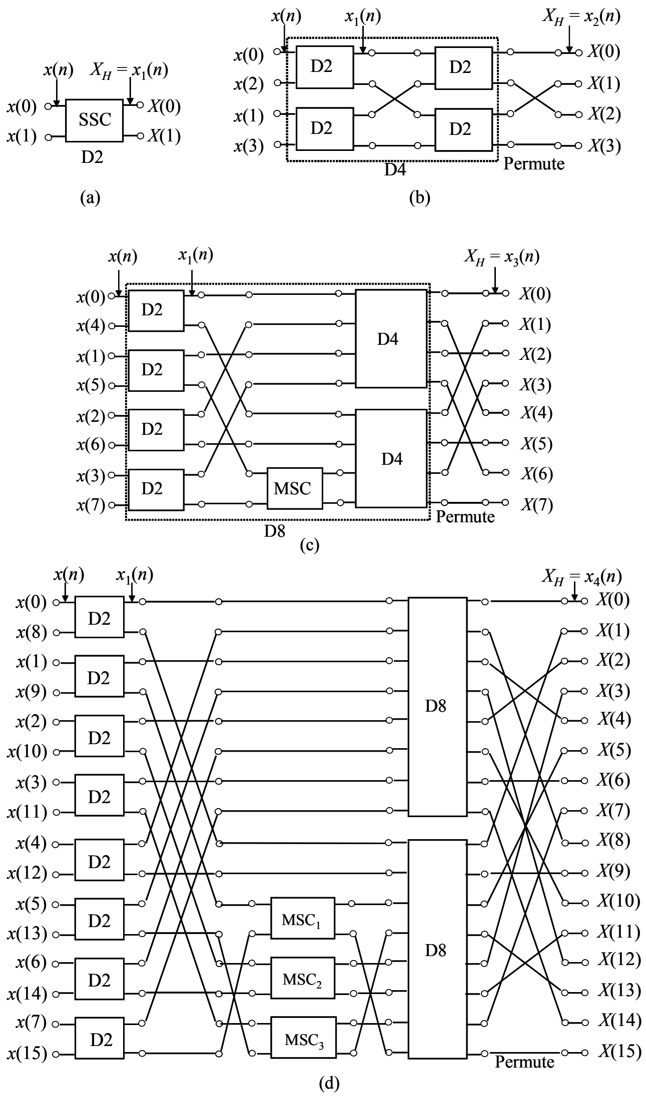


Figure 7. Architecture to obtain the DHT for (a) N = 2, (b) N = 4, (c) N = 8 and (d) N = 16.

Each butterfly in the SFD of the SS shown in Fig. 4 is implemented by a single analog stage structure circuit (SSC). Each stage has $N/2$ butterflies and hence requires $N/2$ SSCs to implement it. Similarly, each butterfly in the SFD of the MS shown in Fig. 4 is implemented by a single multiplying structure circuit (MSC). For each stage S from 1 to $P - 2$, there are $2^{(S-1)}$ MSs and each MS has $(2^{(P-S-1)} - 1)$ MSCs. Hence, $2^{(S-1)} \cdot (2^{(P-S-1)} - 1)$ are the total number of MSCs for each stage. The recursive structure allows generating the architecture for next higher order transform length from two identical lower order ones. It can be directly mapped into the SFD and provides a regular structure for easy implementation.

VI. SIMULATION RESULTS

The architecture has been tested by simulating it with the help of Orcad PSpice. The forward and the inverse transformations have been tested. It has been tested by applying different types of sequence patterns such as step, impulse, sinusoidal, ramp and found to obtain the desired output sequences. These output sequences are applied as input to the inverse transformation to retrieve back the original sequence. The theoretically calculated values and the outputs obtained by simulation for the forward and inverse transformations for these patterns are tabulated in Tables II to V.

TABLE II. RESULTS FOR STEP PATTERN

n	**Input $x(n)$ (mV)**	**Forward transformation output X_H (mV)**		**Inverse transform-ation output $x(n)$ (mV)**
		Theoretical Values	***Simulation Results***	
0	1000	1000	1000	1000
1	1000	0	0	1001
2	1000	0	0	998
3	1000	0	0	998
4	1000	0	0	1000
5	1000	0	0	1001
6	1000	0	0	1001
7	1000	0	0	1000

TABLE III. RESULTS FOR IMPULSE PATTERN

n	**Input $x(n)$ (mV)**	**Forward transformation output X_H (mV)**		**Inverse transform-ation output $x(n)$ (mV)**
		Theoretical Values	***Simulation Results***	
0	200	25	25	200
1	0	25	24	0
2	0	25	25	1
3	0	25	26	0
4	0	25	25	2
5	0	25	25	0
6	0	25	26	0
7	0	25	25	0

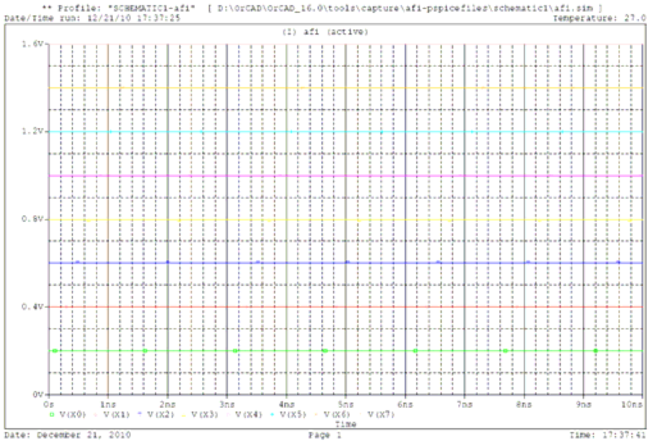
TABLE IV. RESULTS FOR SINUSOIDAL PATTERN

n	**Input $x(n)$ (mV)**	**Forward transformation output X_H (mV)**		**Inverse transform-ation output $x(n)$ (mV)**
		Theoretical Values	***Simulation Results***	
0	0	0	0	0
1	707	500	500	709
2	1000	0	0	1000
3	707	0	0	709
4	0	0	0	2
5	-707	0	0	-705
6	-1000	0	0	-998
7	-707	-500	-500	-705

TABLE V. RESULTS FOR RAMP PATTERN

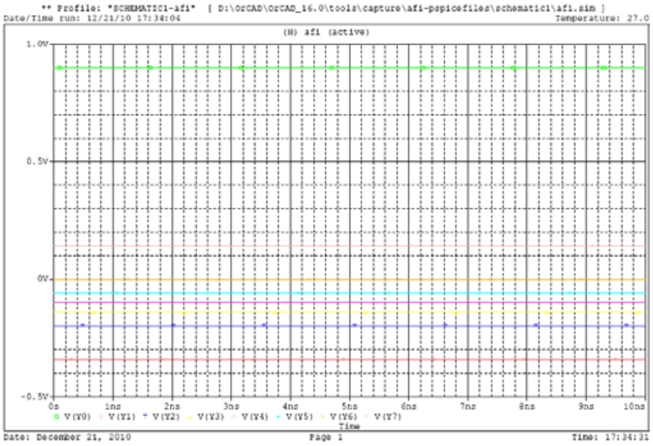
n	**Input $x(n)$ (mV)**	**Forward transformation output X_H (mV)**		**Inverse transform-ation output $x(n)$ (mV)**
		Theoretical Values	***Simulation Results***	
0	200	900	900	201
1	400	-341	-340	398
2	600	-200	-200	600
3	800	-141	-140	798
4	1000	-100	-100	998
5	1200	-58	-59	1196
6	1400	0	0	1396
7	1600	141	140	1596

The input sequence $V(XN)$ corresponding to a ramp sequence pattern is shown in Figure 8 (a), the simulation results for the output sequence $V(YN)$ obtained after the forward transformation is shown in Figure 8 (b) and the retrieved sequence $V(ZN)$ after the inverse transformation is shown in Figure 8 (c). They are in good agreement with those obtained theoretically.



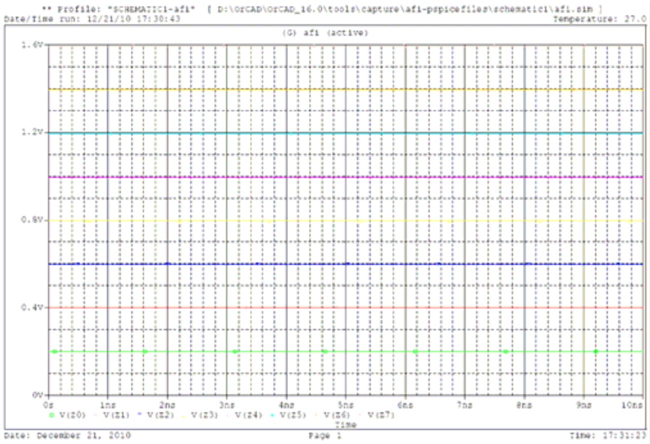
(a)

Figure 8. (a) Input sequence



(b)

Figure 8. (b) Output sequence after forward transformation



(c)

Figure 8. (c) Retrieved sequence after forward and inverse transformations

VII. CONCLUSIONS

In the existing algorithms [1]-[5], the stage structures perform all the additions and multiplications. They require the computation of stage dependent cosine and sine coefficients for each stage. The proposed algorithm introduces multiplying structures in the signal flow diagram which perform all the multiplications with the cosine coefficients and their related additions. This leads to simplification of the stage structures which are now similar in nature and perform only the additions. The proposed algorithm computes lesser number of stage independent cosine coefficients. The distinct advantage is that the number of multiplications is reduced without affecting the number of additions. Its recursive nature allows generating the next higher order transform from two identical lower order ones. It can be directly mapped into the SFD and provides a regular structure for easy implementation using the proposed basic analog circuit. The architecture utilizing this circuit is modular and can be scaled for large values of *N* unlike the neural net approach in [22]. It processes the data simultaneously at each stage and speeds up the transformation as compared to those which employ a multiply and accumulate approach as in [26]. Both the forward and inverse transformations have been tested by performing the simulation on Orcad PSpice. The architecture could prove suitable for signal processing applications using FPAAs [27]-[28].

REFERENCES

[1] R. N. Bracewell, “The fast Hartley transform,” *Proc. IEEE*, vol. 72, no. 8, pp. 1010-1018, Aug. 1984.

[2] H. J. Meckelburg and D. Lipka, “Fast Hartley transform algorithm,” Electronics Letters, vol. 21, no. 8, pp. 311-313, Apr. 1985.

[3] H. V. Sorensen, D. L. Jones, C. S. Burrus and M. T. Heideman, “On computing the discrete Hartley transform,” IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-33, no. 4, pp. 1231-1238, Oct. 1985.

[4] J. Prado, Comments on “The fast Hartley transform,” Proc. IEEE, vol. 73, no. 12, pp. 1862-1863, Dec. 1985.

[5] C. P. Kwong and K. P. Shiu, “Structured fast Hartley transform algorithms,” IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-34, no. 4, pp. 1000-1002, Aug. 1986.

[6] H. S. Hou, “The Fast Hartley Transform Algorithm,” IEEE Trans. Computers, vol. C-36, no. 2, pp. 147-156, Feb 1987.

[7] H. Hao, “On fast Hartley transform algorithms,” Proc. IEEE, vol. 75, no. 7, pp. 961-962, July 1987.

[8] H. S. Malvar, “Fast computation of the discrete cosine transform and the discrete Hartley transform,” IEEE Trans. Acoustics, Speech, and Signal Processing, vol. ASSP-35, no. 10, pp. 1484-1485, Oct. 1987.

[9] T. S. Rathore, “Recursive relations for complexities of Hartley transform algorithms,” IETE Journal of Research, vol. 35, no. 6, pp. 357-359, Nov.-Dec 1989.

[10] T. S. Rathore, “Hartley transform – Properties and algorithms,” in Proc. National Conf. Real Time Systems, Indore, pp. 21-30, Nov. 1990.

[11] C. Chakrabarti and J. Jaja, “Systolic architectures for the computation of the discrete Hartley and discrete cosine transforms based on prime factor decomposition,” IEEE Trans. Comp., vol. 39, no. 11, pp. 1359–1368, Nov. 1990.

[12] A. S. Dhar and S. Banerjee, “An array architecture for fast computation of discrete Hartley transform,” IEEE Trans. Circuits Syst., vol. 38, no. 9, pp. 1095–1098, Sep. 1991.

[13] L. W. Chang and S. W. Lee, “Systolic arrays for the discrete Hartley transform,” IEEE Trans. Signal Process., vol. 39, no. 11, pp. 2411–2418, Nov. 1991.

[14] J. H. Hsiao, L. G. Chen, T. D. Chiueh, and C. T. Chen, “Novel systolic array design for the discrete Hartley transform with high throughput rate,” in Proc. IEEE Int. Conf. Circuits Syst., pp. 1567–1570, 1993.

[15] D. C. Kar and V. V. Bapeswara Rao, “A cordic-based unified systolic architecture for sliding window applications of discrete transforms,” IEEE Trans. Signal Process., vol. 44, no. 2, pp. 441–444, Feb. 1996.

[16] S. S. Nayak and P. K. Meher, “High throughput VLSI implementation of discrete orthogonal transforms using bit-level vector-matrix multiplier,” IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 46, no. 5, pp. 655–658, May 1999.

[17] J. I. Guo, “An efficient design for one-dimensional discrete Hartley transform using parallel additions,” IEEE Trans. Signal Process., vol. 48, no. 10, pp. 2806–2813, Oct. 2000.

DOI 10.5013/IJSSST.a.12.01.01

ISSN: 1473-804x online, 1473-8031 print

[18] J. I. Guo, “A new DA-Based array for one-dimensional discrete Hartley transform,” in Proc. Int. Symp. Circuits Syst, (ISCAS 2001), vol. 4, pp. 662-665, May 2001.

[19] A. Amira and A. Bouridane, “An FPGA implementation of discrete Hartley transforms,” in Proc. 7th Int. Symp. Signal Process. Appl., vol. 1, pp. 625–628, Jul. 2003.

[20] A. Amira and A. Bouridane, “An FPGA-based accelerator for discrete Hartley and fast Hadamard transforms,” in Proc. Int. Symp. Micro-NanoMechatronics and Human Science (MWCAS’03), vol. 2, pp. 860–863, 2003.

[21] P. K. Meher, T. Srikanthan and J. C. Patra, “Scalable and modular memory-based systolic architectures for discrete Hartley transform,” IEEE Trans. Circuits Syst. I, vol. 53, no. 5, pp. 1065–1077, May 2006.

[22] A. D. Culhane, M. C. Peckerar and C. R. K. Marrian, “A neural net approach to discrete Hartley and Fourier transforms,” IEEE Trans. Circuits Syst., vol. 36, no. 5, pp. 695-703, May 1989.

[23] R. Raut, B. B. Bhattacharya and S. M. Faruque, “A discrete Fourier transform using switched-capacitor circuits in systolic array architecture,” IEEE Trans. Circuits Syst., vol. 37, no. 12, pp. 1578-1580, Dec. 1990.

[24] S. Kawahito et al.,“A CMOS image sensor with analog two-dimensional DCT based compression circuits for on-chip cameras,” IEEE J. Solid-State Circuits, vol. 32, pp. 2030-2041, Dec. 1997.

[25] J. Chen, G. Shou and C. Zhou, “Digital-controlled analog circuits for weighted-sum operations,” IEICE Trans. Fundamentals, vol. E82-A, pp. 2505-2513, Nov 1999.

[26] A. K. Mal and A. S. Dhar, “Analog Sampled Data Architecture for Discrete Hartley Transform,” in Proc. Tenth Int. Conf. on Convergent Technologies for Asia-Pacific Region, Bangalore, India, vol. 3, pp. 1035–1039, Oct. 2003.

[27] P. K. Meher, T. Srikanthan and J. C. Patra, “Large-Scale Field-Programmable Analog Arrays for Analog Signal Processing,” IEEE Trans. Circuits Syst. I, vol. 52, no. 11, pp. 2298-2307, Nov. 2005.

[28] A. H. Madian, S. A. Mahmoud and A. M. Soliman, “Field Programmable Analog Array based on CMOS CFOA and its Application,” in Proc. IEEE Int. Conf. Electr. Circuits Syst. (ICECS’08), pp. 1042–1046, 2008.

[29] AD-844 data sheet, Monolithic operational amplifier, Analog Devices, Rev-C.

[30] T. S. Rathore and U. P. Khot, “CFA-based grounded-capacitor operational simulation of ladder filters,” Int. J. Circ. Theor. Appl., vol. 36, pp. 697-716, 2008.

DOI 10.5013/IJSSST.a.12.01.01 ISSN: 1473-804x online, 1473-8031 print