DHT algorithm based on encoding algebraic integers

R. Baghaie and V. Dimitrov

A novel algorithm for computing the discrete Hartley transform (DHT) is presented. The proposed algorithm is based on the algebraic integers encoding scheme. With the aid of this scheme, an error-free representation of the *cas* function becomes possible. Furthermore, for the implementation of the algorithm a fully pipelined systolic architecture with $O(N)$ throughput is proposed.

Introduction: The discrete Hartley transform (DHT) is an attractive alternative to the discrete Fourier transform (DFT) because of its real-valued computation and properties similar to those of the DFT [1]. Another interesting property of the DHT is that the same kernel is used for both the transform and its inverse transform. Consequently, since its introduction the DHT has found its way to many digital signal processing applications [2]. The 1-D DHT of a N-point sequence $\{x_n,\ n = 0,\ldots,N\text{-}1$ and $N = 2^m\}$ is defined as:

$$X_k = \sum_{n=0}^{N-1} x_n cas(2\pi \frac{kn}{N}) ,\ 0\le k\le N\text{-}1 \qquad (1)$$

where $\text{cas}\,\theta = \cos\theta + \sin\theta$.

Since the introduction of the DHT, a number of systolic architectures have been proposed, many of which are based on the direct implementation of algorithm [3,4]. In these implementations for the calculation of the *cas* function, different types of approximations have been introduced. Processing with algebraic integers, in which the signal sample is represented by a set of small integers, was introduced in [5]. Algebraic integers are roots of monic polynomials that have integer coefficients with leading coefficient equal to unity. The motivation for introducing this new mapping of real numbers is to drastically reduce the dynamic range of each of the independent computations. In this letter, we illustrate how with the aid of the algebraic integers scheme, an efficient error-free systolic implementation of DHT can be obtained.

Algebraic-integer interpretation: Consider the 16-point DHT. The kernel of this transformation is $\text{cas}(2kn\pi/16)$ where $0\le k,\ n \le N\text{-}1$. The classical method for calculating the *cas* function is to approximate the function in binary number system, which leads to rounding off errors. In this paper, we adopt the algebraic integers encoding scheme. Consider the first nonzero angle of the *cas* function that is $2\pi/16$. We can represent the *cos* and *sin* functions of this angle as:

$$\cos(2\pi/16) = \sqrt{2+\sqrt{2}}\Big/2 \qquad (2)$$

$$\sin(2\pi/16) = \sqrt{2-\sqrt{2}}\Big/2 \qquad (3)$$

The other needed angles can be represented in a similar manner. Now without compromising the calculations, we omit the “2” from the denominator of eqns. 2 and 3. Denoting z as $z = 2\cos(2\pi/16) = \sqrt{2+\sqrt{2}}$, where z is a root of eqn. 4.

$$x^4 - 4x^2 + 2 = 0 \qquad (4)$$

Consider now the polynomial of eqn. 5:

$$f(z) = \sum_{i=0}^{3} a_i z^i \qquad (5)$$

where a_i are integers. By assigning (0,1,0,0) to a_i we have an *exact code* for z, and thus, $2(\cos 2\pi/16)$. Proceeding in the same manner as in [6], we can obtain an *error-free* representation of the *cas* function necessary to evaluate the 16-point 1-D DHT. Table 1 presents the corresponding coefficients of every required angle. Other remaining angles can be obtained by changing the signs of the given coefficients.

Table 1: Representation of the *cas* function for 16-point DHT

	a_0	a_1	a_2	a_3	*Error*
$2\text{cas}(0\cdot\pi/16)$	2	0	0	0	0
$2\text{cas}(2\cdot\pi/16)$	0	-2	0	1	0
$2\text{cas}(4\cdot\pi/16)$	-4	0	2	0	0
$2\text{cas}(6\cdot\pi/16)$	0	-2	0	1	0

Systolic array implementation: Although fast Hartley transform (FHT) requires less computation, however, similar to the FFT, in order to support perfect shuffling between different stages the FHT requires global communication. On the other hand, the DHT enjoys simple communication and is much more suitable for VLSI systolic arrays. For the implementation of the proposed DHT algorithm, a fully pipelined systolic architecture is presented. The proposed systolic array enjoys the advantages of simplicity, modularity, regularity, and locality.

Fig. 1 illustrates the case where N = 16. In Fig. 2, the cell function of each Processor Element (PE) is presented. In PE1, a_{ij} $\{i=I,\ldots,0$ and $0\le j\le N\text{-}1\}$ correspond to the coefficients in Table 1 and they are stored in the local registers. Note that index i in a_{ij} corresponds to the position of the i^{th} PE1 column, $i = I,\ldots,0$, where I is the degree of the polynomial of eqn. 5.

In Fig. 1, the input data is first pipelined into the array of PE1s through an output-buffered demultiplexer. The pipelined data x_{in} should then be multiplied by a_{ij}. However, it is easy to see that the required multiplication can be replaced with only one shift operation (Table 1). In PE1, the $shift_{aij}\,(x_{in})$ operation means that x_{in} is shifted once and the type of shift is determined by a_{ij}. This can be implemented with a shifter such as a barrel shifter.

Up to this point, for the computation of the DHT algorithm, we have utilized an error-free format. However, the accuracy of the final reconstruction depends on the precision used to represent z. Therefore, one can estimate the precision that is needed to insure the required accuracy. As an example, if the word-length of data stream is 8-bit, then $\hat{z} = 2 - 2^{-5} - 2^{-7}$ is a very good approximation of z. For the final reconstruction, by utilizing the Horner’s rule, eqn. 5 can be rewritten as:

$$f(z) = ((a_3 z + a_2)z + a_1)z + a_0 \qquad (6)$$

For the implementation of eqn. 6, a simple linear array consisting of PE2 and PE3 processors is utilized.

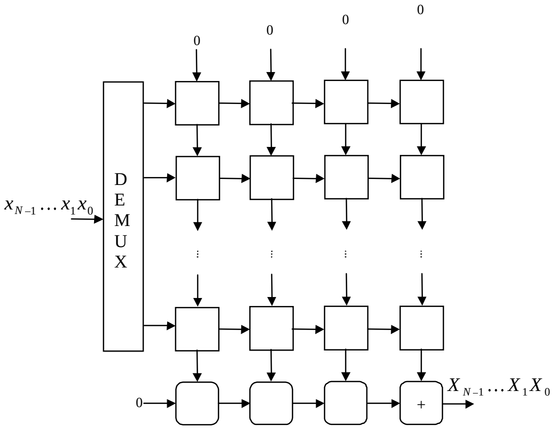
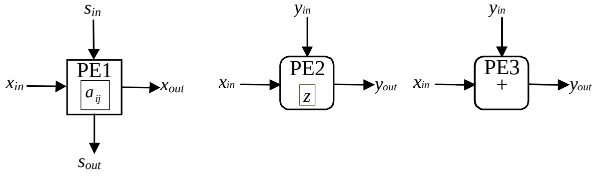


Fig. 1 *Systolic implementation of the 16-point DHT algorithm*



$$x_{out} \leftarrow x_{in} \qquad y_{out} \leftarrow (x_{in} + y_{in})\cdot z \qquad y_{out} \leftarrow x_{in} + y_{in}$$

$$s_{out} \leftarrow s_{in} + shift_{a_{ij}}(x_{in})$$

Fig. 2 *Input-output ports of the PEs and their cell functions*

Hardware and throughput consideration: The proposed architecture requires $N(I + 1)$ PE1s, I PE2s and one PE3. For evaluation of the throughput, we first assume that one time step of the global clock corresponds to one operation in PE2. For the computation of the first set of 1-D DHT $(2N + I)$ time steps are required. The successive sets are computed in an interval of N steps. Therefore, $O(N)$ time complexity is achieved.

In the proposed method, only I multipliers are required. On the other hand, in [3] N CORDICs and in [4] N multipliers are needed. Table 2 illustrates the performance comparison of various systolic implementations of 1-D DHT. From Table 1 it is obvious that 7/16 of operations in PE1s are actually simple data transfer operations $s_{out} \leftarrow s_{in}$. In order to further reduce the complexity, the proposed method can be combined with *any* other DHT algorithms.

Table 2: Comparison of various systolic arrays for 1-D DHT

N	Multiplications Additions		Total Number of multipliers or CORDICs		
	Proposed	Direct [3]	Proposed	Chang & Lee[3]	Pan & Park [4]
8	8 M 72 A	128 M 64 A	1	8	8
16	48 M 496 A	512 M 256 A	4	16	16

For an error-free implementation, as N gets larger I becomes larger as well. However, by utilizing different approximation techniques we can get a very good estimate of the *cas* function as can be seen from Table 3. As an example, the vector (3,-12,10,2) is a good approximation of $\mathrm{cas}(2\pi/32)$ with an error of 0.000012.

Table 3: Approximation of cas(2πk/32) for 32-point DHT

k	4-bit dynamic range					6-bit dynamic range				
	a_0	a_1	a_2	a_3	*Error*	a_0	a_1	a_2	a_3	*Error*
0,8	2	0	0	0	0	2	0	0	0	0
1,7	-4	4	-4	2	0.000281	3	-12	10	2	0.000012
2,6	0	-2	0	1	0	0	-2	0	1	0
3,5	-7	3	-8	5	0.001291	-20	-18	-15	17	0.000003
4	-4	0	2	0	0	-4	0	2	0	0
9,15	-1	-6	4	0	0.001090	5	-8	-17	11	0.000008
10,14	0	-4	0	1	0	0	-4	0	1	0
11,13	-7	-7	6	0	0.000831	-25	17	26	-15	0.000003
12	0	0	0	0	0	0	0	0	0	0

Conclusions: In this letter, we proposed a novel approach that is aimed at efficient implementation of the DHT algorithm. One of the advantages of the proposed algorithm is an *error-free* implementation of the DHT computation up until the final reconstruction. The proposed method can be combined with *any* DHT algorithms to achieve further hardware reduction. Finally, for the implementation of the algorithm, a fully pipelined systolic architecture with $O(N)$ throughput is proposed.

14 May 1999
Electronics Letters Online No: XXXXXX

R. Baghaie and V. Dimitrov (*Helsinki University of Technology, Department of Electrical and Communications Engineering, PO Box 3000, FIN-02015 HUT, FINLAND*)

References

1 Bracewell, R.N.: ‘Discrete Hartley transform’, *Journal of Optical Society of America*, 1983, **73**, (12), pp. 1832-1835

2 Bracewell, R.N.: ‘Aspects of Hartley transform’, *Proceedings of the IEEE*, 1994, **82**, (3), pp. 381-387

3 Chang, L.W., and Lee, S.W.: ‘Systolic arrays for the discrete Hartley transform’, *IEEE Trans. Signal Proc.*, 1991, **39**, (11), pp. 2411-2418

4 Pan, S.B., and Park, R.-H.: ‘Unified systolic arrays for computation of DCT/DST/DHT’, *IEEE Trans. Circuits and Systems for Video Technology*, 1997, **7**, (2), pp. 413-419

5 Cozzens, J.H., and Finkelstein, L.A.: ‘Computing the discrete Fourier transform using residue number systems in a ring of algebraic integers’, *IEEE Trans. Info. Theory*, 1985, **31**, (5), pp. 580-588

6 Dimitrov, V., Jullien, G.A., and Miller, W.C.,: ‘A new DCT algorithm based on encoding algebraic integers’. Proc. ICASSP'98, Seattle, 1998, vol. 3, pp. 1377-1380