A Simple Architecture of 8- point Discrete Hartley Transform using DGA Algorithm

Nibedita Patra[*], Dr. Sudhansu Sekhar Nayak[**]

[*]Research Scholar,
Centurion university of Technology and Management,
Paralakhemendi,
Odisha.761211.
[**]Professor in Physics,
Centurion University of Technology and Management,
Paralakhemundi, Odisha.

Abstract - The discrete Hartley transform is a real-valued transform, comparable to the complex Fourier transform, which has many applications in disciplines such as pattern recognition, signal and image processing. In this paper, a new very large scale integration (VLSI) algorithm for an 8-point discrete Hartley transform (DHT) is presented, named as Dual group adaptive DHT algorithm which can be efficiently implemented on a highly modular and parallel VLSI architecture with a regular structure and consists of two blocks: a multiplication block and an addition/subtraction block. The proposed DHT algorithm may be efficiently divided into many parallel components that can be run concurrently. Furthermore, the proposed algorithm lends itself well to the sharing approach, which may be utilised to greatly decrease the hardware complexity of a highly parallel VLSI implementation. The proposed architecture employs only two multipliers and 36 adders. Using the benefits of the proposed algorithm, the number of multipliers has been considerably decreased, resulting in a very small number of multipliers when compared to the existing algorithms. Furthermore, constant multipliers can be effectively implemented in VLSI. As a result, the proposed design is an effective solution for space-constrained applications.

Keywords - Discrete Hartley Transform, Multiplier, Adder, Real-valued Transform, VLSI, Systolic arrays

I. INTRODUCTION

The discrete Hartley transform was first introduced in 1983 [1], and the first fast DHT was introduced in 1984 [2]. The Hartley transform (DHT) is a real-valued alternative to the discrete Fourier transform (DFT) that leads to more efficient computation of the DFT as well as other unitary transforms such as cosine and sine transforms [3,4]. The DHT computation is extremely computationally demanding, necessitating the use of highly sophisticated hardware. For its hardware computation, a variety of architectures have been proposed, ranging from memory-based to FPGA to digital signal processors (DSP) [5, 6, 7]. The DHT discovers signal and image processing applications such as error control coding, adaptive filtering, multi-carrier modulation, image classification, image encryption, and image compression [8-12]. Since then, ongoing efforts have been made to reduce its computational complexity and, as a result, increase the speed of computation in software and hardware [13–18]. There are some quick methods for DHT calculation and some techniques for extended DHT computation. There are also numerous split-radix methods available for calculating DHT with a cheap arithmetic cost. As a result, [19, 20] developed split-radix algorithms for DHT with minimal arithmetic cost. [21] Developed another split-radix technique in which the odd-indexed transform outputs are calculated indirectly. Due to its irregular computational structure and the fact that the butterflies change substantially from stage to stage, the conventional split-radix method is challenging to execute on VLSI. As a result, it is important to develop new such algorithms that are appropriate for a parallel VLSI system. There are also numerous fast recursive methods in the literature, such as those in [22] for discrete cosine transform (DCT) and [23] for extended DHT. Because DHT is computationally expensive, specialised hardware solutions based on VLSI technology are required. Systolic arrays are one type of VLSI implementation. There are several DHT systolic array implementations and that designs are modular and regular, but they rely on pipelining rather than parallel processing to provide high-speed computing. Multipliers in a VLSI framework use a substantial amount of chip space and cause significant delays. To efficiently build multipliers using lookup-table-based solutions, one operand must be a constant. If one of the operands is constant, all of the partial results can be stored in a ROM. The amount of memory words has been decreased from 22N to 2N.

This work proposes a novel DHT algorithm that is ideally suited for implementation on a highly parallel and modular architecture. The hardware complexity can be considerably decreased by using the common constant sharing approach and use the same multipliers. The proposed strategy is interesting due to its high level of parallelism as well as its use of a modular and regular structure. It may also be utilised to achieve low hardware complexity by substantially sharing common blocks. This work present a simple architecture for DHT that will applicable in area constrained applications in which very less number of multipliers has been used as compared to the other existing works.

The remainder of this research report is divided into several sections. Section II review the existing works which is related to the DHT algorithms. In Section III, describe a novel proposed algorithm for calculating an M-point DHT. In Section IV, provide an algorithm for an 8-point DHT. Section V will describe the new architecture for 8 point DHT, and Section VI will conclude the research.

II. Related Works

This section briefly explains the existing works that has been done which is related to DHT algorithms and simple architectures.

Chiper et al. [24] proposed a novel radix-2 fast algorithm for a VLSI parallel DHT-III implementation. This method is based on a direct recursive decomposition of a DHT-III of length N into two DHT-IIIs with length N/2. Aside from a minimal number of arithmetic operations, the proposed method features a basic and regular structure that makes it well suited for VLSI implementation.

Shu et al. [25] developed a novel fast radix-3 algorithm for computing the length-N generalised discrete Hartley transform of type II. For length N ≥ 9, the computational complexity of the proposed solution is lower than that of the conventional approach. The arithmetic operations may be avoided from 19% to 29% for N = 3m varying from 9 to 243 and from 17% to 29% for N = 3x2m changing from 12 to 384.

Pyrgas et al. [26] developed an FPGA design for 64-point Two-Band Fast DHT processing. Because of its streamlined and symmetric form, the proposed design is implemented on a highly parallel architecture with low hardware complexity. The architecture computes the 64-point Two-Band Fast Discrete Hartley Transform in 57 clock cycles and can process up to 103.82 million samples per second at a clock frequency of 92 MHz.

Lampros et al. [27] established two flexible FPGA designs for 32-point Two-Band Fast DHT processing. Because of the streamlined, regular, and symmetric structure of the Two-Band Fast DHT, the suggested designs are implemented in a highly parallel manner with little hardware complexity. The findings of the synthesis show that the designs are an efficient alternative for area-constrained applications with high operating frequency requirements.

The upcoming section briefly explains about the novel proposed algorithm to compute the DHT for area constraint applications.

III. Novel Algorithm for M-point DHT

The DHT is a linear real-to-real transform with forward and inverse DHT transforms that are identical. The DHT transform of a set of N real-valued input data $x(0), x(1), \dots, x(M-1)$ is defined as follows:

$$\boldsymbol{D_M(k) = \sum_{m=0}^{M-1} x(m)\, \mathrm{cas}\left(\frac{2\pi km}{M}\right)} \quad (1)$$

Where k =0, 1, 2,…,M – 1 and cas (·) = cos (·) + sin(·) (2)

The Dual Group Adaptive DHT algorithm is based on splitting input data into low-band $x_l(m)$ and high-band $\boldsymbol{x_h(m)}$ values. We can construct an N-length DHT by employing a new algorithm defined by the following relationships:

$$\boldsymbol{x_l(m) = \frac{x(2m)+x(2m+1)}{2}} \quad (3)$$

$$\boldsymbol{x_h(m) = \frac{x(2m)-x(2m+1)}{2}} \quad (4)$$

$$x(2m) = \boldsymbol{x_l(m)} + \boldsymbol{x_h(m)} \quad (5)$$

$$x(2m+1) = \boldsymbol{x_l(m)} - \boldsymbol{x_h(m)} \quad (6)$$

Following the splitting of the dual groups, the core DHT transform is applied to the summations and differences in the low-band and high-band values of neighbouring samples. This is also supported by the characteristics of the trigonometric function cas(•).

$$D_M(k) = D_l(k) + [\cos\theta\, D_h(k) + \sin\theta\ D_h(-k)] \quad (7)$$

$$\text{Where } \theta = {2\pi k}/{M} \quad (8)$$

$$D_l(k) = D^l_{M/2}(k) + D^h_{M/2}(k) \quad (9)$$

$$D_h(k) = D^l_{M/2}(k) - D^h_{M/2}(k) \quad (10)$$

and

$$D^l_{M/2}(k) = \boldsymbol{\sum_{m=0}^{M/2-1} x_l(m)\, \mathrm{cas}\left(\frac{2\pi km}{M/2}\right)} \quad (11)$$

$$D^h_{M/2}(k) = \boldsymbol{\sum_{m=0}^{M/2-1} x_h(m)\, \mathrm{cas}\left(\frac{2\pi km}{M/2}\right)} \quad (12)$$

The DHT coefficients of (7) that are M/2 places away from k are calculated as follows:

$$D_M\left(k+\frac{M}{2}\right) = D_l(k) - [\cos\theta\, D_h(k) + \sin\theta\ D_h(-k)] \quad (13)$$

$$D_h(-k) = D_h\left(\frac{M}{2}-k\right) \quad (14)$$

The M-point DHT is transformed into,

$$D_M(k) = D_l(k) + [\cos\theta\, D_h(k) + \sin\theta\ D_h\left(\frac{M}{2}-k\right)] \quad (15)$$

$$D_M\left(k+\frac{M}{2}\right) = D_l(k) - [\cos\theta\, D_h(k) + \sin\theta\ D_h\left(\frac{M}{2}-k\right)] \quad (16)$$

Where k= 0, 1,…, $\frac{M}{2}-1$

The resulting method may be utilised as a Dual Group Adaptive Algorithm (DGA), with the number of multipliers considerably decreased by sharing the multipliers with the same constant. The number of multipliers can be lowered even further by employing sub-expression sharing methods and sharing multipliers with the same constant.

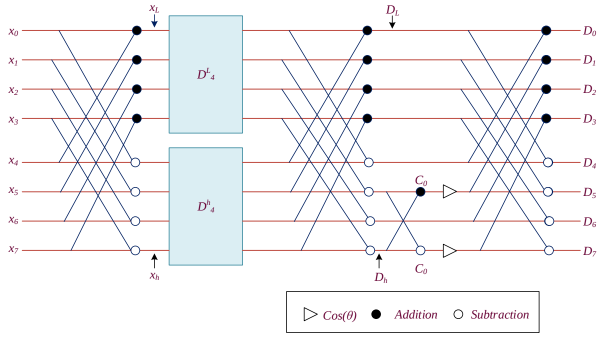

Figure 1: Flow graph of 8-point DHT

Figure 1 depicts an 8-point Dual Group Adaptive DHT flow graph. It is made up of two 4-point Dual Group Adaptive DHTs, extremely symmetrical butterflies, and two twiddle factor multiplications. There are butterflies that can compute using simply addition or subtraction. Their output nodes are represented by full or empty cycles. There are butterflies that require cosine multiplications before addition and subtraction in order to compute. Their output nodes are represented as full rectangles or empty rectangles. This implies that the node's two inputs should be multiplied by Hartley coefficient before being added or subtracted.

IV. Algorithm for an 8-point DHT

8-point DHT is a sequence {xm; m= 0, 1, 2,..,7} is

$$\boldsymbol{D_8(k) = \sum_{m=0}^{7} x(m)\, \mathrm{cas}\left(\frac{2\pi km}{8}\right)} \quad (17)$$

The 8-point dual group adaptive DHT as given below,

$$D_8(k) = D_l(k) + [\cos\theta\, D_h(k) + \sin\theta\ D_h(4-k)] \quad (18)$$

$$D_8(k+4) = D_l(k) - [\cos\theta\, D_h(k) + \sin\theta\ D_h(4-k)] \quad (19)$$

Where $\theta = {}^{2\pi k}/_{8}$,

$$D_l(k) = D_4^l(k) + D_4^h(k)$$
$$D_h(k) = D_4^l(k) - D_4^h(k)$$
$$D_4^l(k) = \sum_{\boldsymbol{m=0}}^{\boldsymbol{3}} \boldsymbol{x_l(m)\ \mathrm{cas}\left(\frac{2\pi km}{4}\right)}$$
$$D_4^h(k) = \sum_{\boldsymbol{m=0}}^{\boldsymbol{3}} \boldsymbol{x_h(m)\ \mathrm{cas}\left(\frac{2\pi km}{4}\right)}$$

Where k = 0,1,2,3 and multiplication coefficient will be c0 = 0.707107.

The 8 point DHT described below,

$$D_8(0) = x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7$$
$$D_8(1) = x_0 + x_1\sqrt{2} + x_2 - x_4 - x_5\sqrt{2} - x_6$$
$$D_8(2) = x_0 + x_1 - x_2 - x_3 + x_4 + x_5 - x_6 - x_7$$
$$D_8(3) = x_0 - x_2 + x_3\sqrt{2} - x_4 + x_6 - x_7\sqrt{2}$$
$$D_8(4) = x_0 - x_1 + x_2 - x_3 + x_4 - x_5 + x_6 - x_7$$
$$D_8(5) = x_0 - x_1\sqrt{2} + x_2 - x_4 + x_5\sqrt{2} - x_6$$
$$D_8(6) = x_0 - x_1 - x_2 + x_3 + x_4 - x_5 - x_6 + x_7$$
$$D_8(7) = x_0 - x_2 - x_3\sqrt{2} - x_4 + x_6 + x_7\sqrt{2}$$

The structure is basic, regular, modular, and scalable, making software or hardware implementations easier. The structure is similar to fast transformations' 'constant geometry.' Constant geometry methods avoid the area and time overhead associated with multiplexing various registers, which is advantageous in high-throughput systems.

The proposed design which uses 2 multipliers and 36 adders/sub tractors. Because multiplying with the same constant "c" and sharing the same multiplier reduces the number of multipliers even further.

Computational Complexity: We can simply calculate the amount of operations required to compute the dual group M-point fast DHT.

$$Number\ of\ Additions = 2Number\ of\ Additions_{M/2} + 3M + {}^{M}/_{2} - 8$$

$$Number\ of\ Multiplication = 2Number\ of\ Multiplication_{M/2} + M - 4$$

V. Proposed Architecture for 8-Point DHT using Dual group Adaptive Computation

This section provides the simple and compact architecture of 8-Point DHT using dual group adaptive algorithm. It uses 2 multipliers and 36 adder/ sub tractor to implement. We chose designs with a modest increase in multiplication cost to enhance symmetry and resource reusability while keeping multipliers (M) and adders (A) to a minimum. In terms of number of multiplications, the proposed algorithm beats the existing designs mentioned in the related works. However, in terms of hardware implementation, our proposed design employs a constant number of multipliers and adders regardless of the number of points.

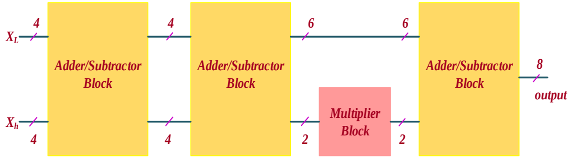

Figure 2: Proposed Architecture of 8-Point DHT

Since the proposed architecture shown in figure 2 have been built in a highly parallel and symmetric structure with limited hardware resources, taking use of some design approaches such as multiplier and addition sharing and some FPGA features such as DSP components Furthermore, the proposed designs' high parallelism with regular structure necessitates the usage of a small number of multiplexers. This is critical for space-constrained applications such as embedded or ubiquitous computing.

The below tables shows the synthesis results and the comparison of the arithmetic complexity of the 8-Point DHT

Table 1: Synthesis Results of the 8-Point Dht

FPGA device	Xilinx Zynq 7z020clg484-1
Units in Add/ Sub Block	2
Slice LUTs	491
Slice Registers	186
Maximum Frequency	100MHz
Clock cycle	14

Table 2: Arithmetic Complexity of the Existing Works with the Proposed Design.

Methods	**Multiplication**	**Additions/ Subtraction**
Fast Radix-2 Algorithm [23]	10	28
Dual spilt Radix Algorithm [17]	2	16
Radix-2 Fast Algorithm [24]	12	23
Novel VLSI DHT algorithm [16]	2	16
Two band DHT Algorithm[27]	2	42
Proposed Algorithm	2	36

VI. Conclusion

This research provides simple architectures for the 8-point dual group adaptive DHT computing. Because of the reduced, regular, and symmetric structure of the dual group adaptive DHT, the suggested designs are implemented in a highly parallel manner with little hardware complexity. The proposed design employs two multipliers and 36 adders/sub tractors, and it has been demonstrated that the proposed architectures are an efficient solution for area-constrained applications requiring high operating frequencies.

Reference

[1] R. N. Bracewell, "Discrete Hartley transform" *J. Opt. Soc. Am.*, vol. 73, no. 12, pp. 1832–1835, 1983.

[2] R. N. Bracewell, "The fast Hartley transform," *Proc. IEEE*, vol. 72, no. 8, pp. 1010–1018, 1984.

[3] H. V. Sorensen, D. L. Jones, C. S. Burrus, M. Heideman, "On computing the discrete Hartley transform," IEEE Trans. Acoust. Speech Signal Process, vol. 33, no. 5, pp. 1231–1238, 1985.

[4] H. J. Meckelburg, D. Lipka, “Fast Hartley transform algorithm,” *Electron. Lett.*, vol. 21, no. 8, pp. 311–313, 1985.

[5] A. Amira, “An FPGA-based parameterisable system for discrete Hartley transforms implementation,” *Proceedings of the International Conference on Image Processing (ICIP’03),* vol. 2, pp. 567–570, 2003.

[6] K. M. M. Prabhu, R. S. Sundaram, “Fast Hartley transform implementation on DSP chips,” *Microprocessors Microsyst.*, vol. 20, pp. 233–240, 1996.

[7] A. Amira, A. Bouridane, “An FPGA implementation of discrete Hartley transforms,” *Proceedings of the 7th International Symposium Signal Processing and Its Applications,* vol. 1, pp. 625–628, 2003.

[8] S. Bouguezel, M. O. Ahmad, M. N. S. Swamy, “A new split-radix FHT algorithm for length-q×2m DHTs,” *IEEE Trans. Circuits Syst.*, vol. I 51, no. 10, pp. 2031–2043, 2004.

[9] P. K. Meher, J. C. Patra, M. N. S. Swamy, “High-throughput memory-based architecture for DHT using a new convolutional formulation,” *IEEE Trans. Circuits Syst. II: Express Briefs,* vol. 54, no. 7, pp. 606–610, 2007.

[10] M. T. Hamood, S. Boussakta, “New radix-based FHT algorithm for computing the discrete Hartley transform,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1581–1584, 2011.

[11] M. Tamilselvi, M. P. Gomathi, A. Lakshminarayanan, “Resourceful fast DHT algorithm for VLSI implementation by split radix algorithm,” *Int. J. Res. Eng. Technol.,* vol. 3, no. 14, pp. 44–48, 2014.

[12] P. Yu, H. Hua, “A new fast recursive algorithm for computing discrete Hartley transform and its implementation,” *Signal Process.*, vol. 81, no. 10, pp. 2235–2241, 2001.

[13] P. Saha, P. Bhattacharyya, D. Kumar, A. Dandapat, “ASIC implementation of high speed processor for computing fast Hartley transformation,” *Proceedings of the International Conference on Advanced Electronic Systems,* pp. 334–336, 2013.

[14] A. N. Skodras, M. F. Aburdene, A. K. Nandi, “Two-band fast Hartley transform,” *Electron. Lett.*, vol. 51, no. 1, pp. 57–59, 2015.

[15] R. C. d Oliveira, H. M. d Oliveira, R. C. d Souza, E. J. P. Santos, “A flexible implementation of a matrix Laurent series-based 16-point fast Fourier and Hartley transforms,” *2010 VI Southern Programmable Logic Conference (SPL), Ipojuca,* pp. 175–178, 2010. http://dx.doi.org/10.1109/SPL.2010.5483017.

[16] D. F. Chiper, “A novel VLSI DHT algorithm for a highly modular and parallel architecture,” *IEEE Trans. Circuits Syst. II: Express Briefs,* vol. 60, no. 5, 2013.

[17] D. F. Chiper, “A structured dual split-radix algorithm for the discrete Hartley transform of length 2^N,” *Circuits Syst. Signal Process.,* 2017. http://dx.doi.org/10.1007/s00034-017-0552-3.

[18] W. L Hsue, W. C. Chang, “Real discrete fractional Fourier, Hartley, generalized Fourier and generalized Hartley transforms with many parameters,” *IEEE Trans. Circuit and Syst. - I: Regul. Pap.,* vol. 62, no. 10, 2015.

[19] H. V. Sorensen, D. L. Jones, C. S. Burrus and M. T. Heideman, “On computing the discrete Hartley transform,” *IEEE Trans. Acoust., Speech, Signal Process.,* vol. ASSP-33, no. 5, pp. 1231–1238, Oct. 1985.

[20] H. S. Malvar, “Signal Processing With Lapped Transforms”, *Norwood, MA, USA: Artech House*, 1992.

[21] G. Bi, “New split-radix algorithm for the discrete Hartley transform,” *IEEE Trans. Signal Process.,* vol. 45, no. 2, pp. 297–302, Feb. 1997.

[22] C. W. Kok, “Fast algorithm for computing discrete cosine transform,” *IEEE Trans. Signal Process.,* vol. 45, no. 3, pp. 757–760, Mar. 1997.

[23] D. F. Chiper, “Fast radix-2 algorithm for the discrete Hartley transform of type II,” *IEEE Signal Process. Lett.,* vol. 18, no. 11, pp. 687–689, Nov. 2011.

[24] Chiper, Doru Florin, “Radix-2 fast algorithm for computing discrete Hartley transform of type III,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 5, pp. 297-301, 2012.

[25] Shu, Huazhong, Jiasong Wu, Chunfeng Yang and Lotfi Senhadji, “Fast radix-3 algorithm for the generalized discrete Hartley transform of type II,” *IEEE Signal Processing Letters,* vol. 19, no. 6, pp. 348-351, 2012.

[26] Pyrgas, Labros, Paris Kitsos and Athanassios N. Skodras, “An FPGA Design for the Two-Band Fast Discrete Hartley Transform,” *In 2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), IEEE,* pp. 295-299, 2016.

[27] Pyrgas, Lampros, Paris Kitsos and Athanassios Skodras, “Compact FPGA architectures for the two-band fast discrete Hartley transform,” *Microprocessors and Microsystems,* vol. 61, pp. 117-125, 2018.