New VLSI Architecture for Audio Compression Using Parallel Computation

Suganthi s [1], Thiruveni M[2]
[1]*PG Student, VLSI Design, PSNA College of Engineering and Technology, Dindigul, Tamil Nadu, India*
Corresponding author: suganthiraj50@gmail.com
[2]*Assistant Professor, Department of Electronics and Communications, PSNA College of Engineering and Technology, Dindigul, Tamil Nadu, India*

Abstract -**In this paper, we propose a new very large scale integration (VLSI) algorithm for a 2*N*-length discrete Hartley transform (DHT) that can be efficiently implemented on a highly modular and parallel VLSI architecture. In the existing system the conventional discrete Hartley transformation is used. In that system has more number of multipliers which leads to more cost and high complexity in VLSI Architectures. To overcome this demerit, New Discrete Hartley transformation is going to be proposed. The proposed DHT components can be separated as even and odd components and the computation can be done parallely.The DHT algorithm can be efficiently split on several parallel parts that can be executed concomitantly. The proposed DHT architecture is well-matched for the sub expression sharing technique. Sub expression technique is used when one data is multiplied with many constants or sum of product. So it reduces the number of multiplications which in turn reduce the hardware complexity, delay and cost.**

Keywords: **Discrete Hartley transform (DHT), DHT domain processing, fast algorithms.**

I.INTRODUCTION

The Discrete Fourier transform (DFT) is used in many digital signal processing applications as in signal and image compression techniques, filter banks, signal representation, or harmonic analysis. The discrete Hartley transform (DHT) can be used to efficiently replace the DFT when the input sequence is real. In the literature, there are some fast algorithms for the computation of DHT and some algorithms for the computation of generalized DHT. There are also several split-radix algorithms for computing DHT with a low arithmetic cost. Thus, Sorensen *et al.* and Malvar proposed split-radix algorithms for DHT with a Low arithmetic cost. Bi proposed another split-radix algorithm where the odd-indexed transform outputs is computed using an indirect method. The classical split-radix algorithm is difficult to implementation of VLSI due to its irregular computational structure and due to the fact that the butterflies significantly differ from stage to stage. Thus, it is necessary to derive new such algorithms that are suited for a parallel VLSI system .There are also in the literature several fast algorithms that use a recursive strategy for discrete cosine transform (DCT) and for generalized DHT. Since DHT is computationally intensive, it is necessary to derive dedicated hardware implementations using the VLSI technology. One category of VLSI implementations is represented by systolic arrays. There are many systolic array implementations of DHT. Systolic array architectures are modular and regular, but they use particularly pipelining and not parallel processing to obtain a high-speed processing. In the literature, highly parallel solutions were also proposed.

The DHT can be computed from the DFT of a real-valued sequence, so FFT algorithms optimized for real-valued sequences must also be considered viable fast algorithms for computation of the discrete Hartley transform. The operation counts for the FHT algorithms are determined and compared to the counts for corresponding real-valued FFT algorithms. A unified development of algorithms for the real-valued FFT and the DHT gives insight into both the discrete Hartley transform and fast algorithms. Using this filter collateral a desired output is generated, it can be used as a part for mass updating to be observe using conventional LMS adaptive algorithm. It is suitable in many real time DSP processing such as adaptive filters. The conventional LMS adaptive algorithm, is used for real-time operation and this algorithm used to the conventional existing systems.

We have a highly parallel solution for the implementation of DHT based on a direct implementation of fast Hartley transform (FHT).
It is worth to note that hardware implementations of FHT are rare. Multipliers in a VLSI structure consume a large portion of the chip area and introduce significant delays. This is the reason why memory-based solutions to implement multipliers have been more and more used in the literature. To efficiently implement multipliers with lookup-table-based solutions, it is necessary that one operand to be a constant. When one of the operands is constant, it is possible to store all the partial results in a ROM, and the number of memory words is significantly reduced from 22*L* to 2*L*.

In this paper we proposed a new VLSI DHT algorithm that is well suited for a VLSI implementation on a highly parallel and Modular architecture . It can be used for designing a completely novel VLSI architecture for DHT. Moreover, using sub expression sharing technique and sharing the multipliers with the same constant, the hardware complexity can be significantly reduced the number of multipliers being very small, significantly less than that in. In the proposed solution, we have used only multipliers with a constant that can be efficiently implemented in VLSI. The proposed solution is not only appealing by its high level of parallelism and by using a modular and regular structure but it can be also used to obtain a small hardware complexity by extensively sharing the common blocks.

II. ALGORITHM FOR DHT

The discrete Hartley transform is a linear, invertible function H : $\mathbf{R}^n$ -> $\mathbf{R}^n$ (where **R** denotes the set of real numbers). The N real numbers x_0, ..., x_{N-1} are transformed into the N real numbers H_0, ..., H_{N-1} according to the formula

$$H_K = \sum_{n=0}^{N-1} x_n[\cos(\frac{2\pi}{N}nk) + \sin(\frac{2\pi}{N}nk)] \qquad (1)$$

Where K = 0,……,N-1

cos(x) + sin(x) = cas(x).

.

The cas function is the sum of sine and cosine function. We can compute a *N*-length DHT using a new algorithm given by the following relations:

$X_N(k)\{x(i)\} = X_{N/2}(k)\{x(2i)\} + u(0)\cdot\sin(2k\pi/N) + [X_{N/2}(k)\{u(i)\} - u(0)/2]$
$\cdot 2\cdot\cos(2k\pi/N)$ (2)

$X_N(N/2 + k)\{x(i)\} = X_{N/2}(k)\{x(2i)\} - u(0)\cdot\sin(2k\pi/N) - [X_{N/2}(k)\{u(i)\} - u(0)/2]$
$\cdot 2\cdot\cos(2k\pi/N)]$
for $k = 0, 1, \ldots, N/4 - 1$ (3)

XN(N/2 − k) {x(i)} = XN/2(N/2 − k) {x(2i)}
+ u(0) · sin(2kπ/N -[XN/2(N/2 − k) {u(i)} − u(0)/2]· 2 · cos(2kπ/N) (4)

XN(N − k) {x(i)} = XN/2(N/2 − k) {x(2i)} − u(0) · sin(2kπ/N)+[XN/2(N/2 − k) {u(i)} − u(0)/2]
· 2 · cos(2kπ/N) for k = 1, . . . , N/4 (5)

Where

XN/2(k) {x(2i)} =

$$\sum_{t=0}^{N/2-1} x(2i).cas[2ki\frac{\pi}{N/2}\ ] \qquad (6)$$

XN/2(k) {u(i)} =

$$\sum_{t=0}^{N/2-1} u(i).cas[2ki\frac{\pi}{N/2}] \qquad (7)$$

are DHT of length *N*/2, with $\{u(i) : i = 0, 1, \ldots, N/2 - 1\}$ an auxiliary input sequence given by

u(N/2 − 1) =x(N − 1) (8)

u(i) =x(2i + 1) − u(i + 1)

for i = N/2 − 2, . . . , 1, 0. (9)

For the computation of (2)–(5), there are necessary extra7*N*/4 additions and *N*/2 multiplications, if we share the multipliers with the same constant. For the computation of the auxiliary input sequence using (8) and (9), there are necessary extra *N*/2 − 1 additions. The obtained algorithm can be used as a VLSI algorithm where the number of multipliers can be significantly reduced by sharing the multipliers with the same constant.

III. PROPOSED STRUCTURE

An efficient implementation of a fast DHT algorithm closely depends on an efficient algorithm for a small DHT. We present here an efficient DHT algorithm for a length N = 8

$$X(0) = [(x(0) + x(4)) + (x(2) + x(6))] + [(x(1) + x(5)) + (x(3) + x(7))]$$

$$X(2) = [(x(0) + x(4)) - (x(2) + x(6))] + [(x(1) + x(5)) - (x(3) + x(7))]$$

$$X(4) = [(x(0) + x(4)) + (x(2) + x(6))] - [(x(1) + x(5)) + (x(3) + x(7))]$$

$$X(6) = [(x(0) + x(4)) - (x(2) + x(6))] - [(x(1) + x(5)) - (x(3) + x(7))]$$

$$X(1) = [x(0) - x(4)] + [x(2) - x(6)] + c\,[x(1) - x(5)]$$

$$X(3) = [x(0) - x(4)] - [x(2) - x(6)] + c\,[x(3) - x(7)]$$

$$X(5) = [x(0) - x(4)] + [x(2) - x(6)] - c\,[x(1) - x(5)]$$

$$X(7) = [x(0) - x(4)] - [x(2) - x(6)] - c\,[x(3) - x(7)]$$

Where $c = \sqrt{2}$

IV. ARITHMETIC COST

Let *A*DHT(*N*) and *M*DHT(*N*) denote the number of additions and multipliers for computing DHT(*N*). We have

$$MDHT(N) = 2MDHT(N/2) + (1/2)N \quad (10)$$

$$ADHT(N) = 2ADHT(N/2) + (9/4)N - 1 \quad (11)$$

where *M*DHT(8) = 2 and *A*DHT(8) = 16.

Solving the recursions (10) and (11), we obtain

$$M_{DHT(N)} = \frac{1}{2} N(\log_2 N - 5) \quad (12)$$

$$A_{DHT(N)} = \frac{9}{4} N \log_2 N - \frac{39}{8} N + 1 \quad (13)$$

Table 1: Computational Complexity

	Radix 2 [13]*		Radix-2 [13]**		Radix-2 [11]		Proposed	
N	M	A	M	A	M	A	M	A
8					4	26	2	16
16	10	62	10	62	20	74	12	67
32	40	168	34	174	69	194	40	205
64	118	418	98	438	196	482	112	553
128	320	1008	258	1070	516	1154	288	1393
256	806	2354	642	2518	1284	2690	704	3361

Table I lists the required number of multiplications and additions for the proposed algorithm, the Sorensen one and Bialgorithm, where rotations are implemented with four multiplications and two addition and with three multiplications and three additions.The values of *M* in the proposed algorithm are computed considering that the multipliers with the same constant are shared. The number of multipliers is significantly greater than that in the proposed one. The number of multipliers for Bi algorithm where rotations are implemented with four multiplications and two additions is greater than the necessary number of multipliers for our algorithm and slightly smaller when the rotations are implemented with three multiplications and three additions.

However, the split-radix algorithm has an irregular structure and is difficult to be implemented in hardware as opposed to our algorithm that has a regular and modular structure and can be very easily implemented in parallel as it will be shown in Section VI for a DHT of length *N* = 32.

V.PARALLEL ARCHITECTURE

In order to clearly illustrate the features and advantages of the proposed algorithm, the VLSI architecture for a DHT of length *N* = 32 is presented in Fig. 1(a) and (b). It can be seen that the proposed architecture is highly parallel and has a modular and regular structure being formed of only a few blocks: U, MUL, ADD/SUB, XCH, and a few additional adders/subtracters. The"U" blocks implement (20), XCH blocks interchange the values and are simply implemented in hardware by appropriate wiring, and MUL blocks are used to implement the shared multipliers with a constant.

This block contains four multipliers with a constant. Each multiplier is shared by four input sequences that are multiplied with the same constant in an interleaved manner using multiplexers and demultiplexers controlled by two clocks. One of the advantages of this algorithm and architecture is the fact that the

multiplications with the same constant are share the MUL blocks.

Thus, the number of multipliers is significantly less than the value 40 given in Table I which has become now only 16.

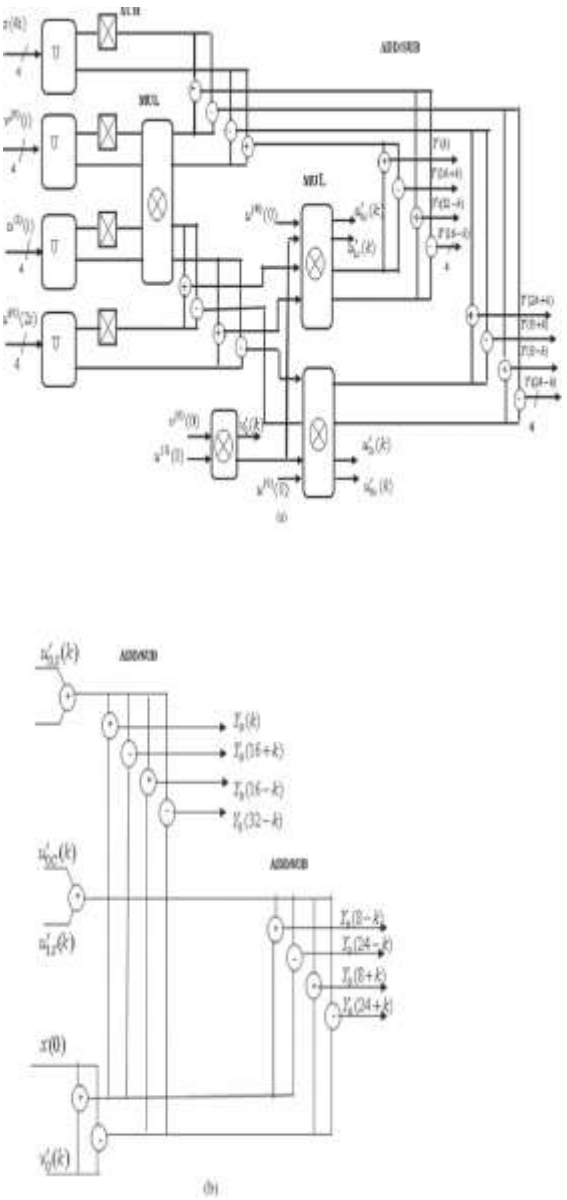

Figure.1. (a) VLSI architecture for DHT of length N = 32 . (b) VLSI architecture for DHT of length N = 32.

The proposed architecture has a high throughput of 32 samples per clock and can be pipelined. It is highly parallel using a low hardware complexity structure. The multipliers with a constant in MUL blocks can be efficiently implemented in hardware.

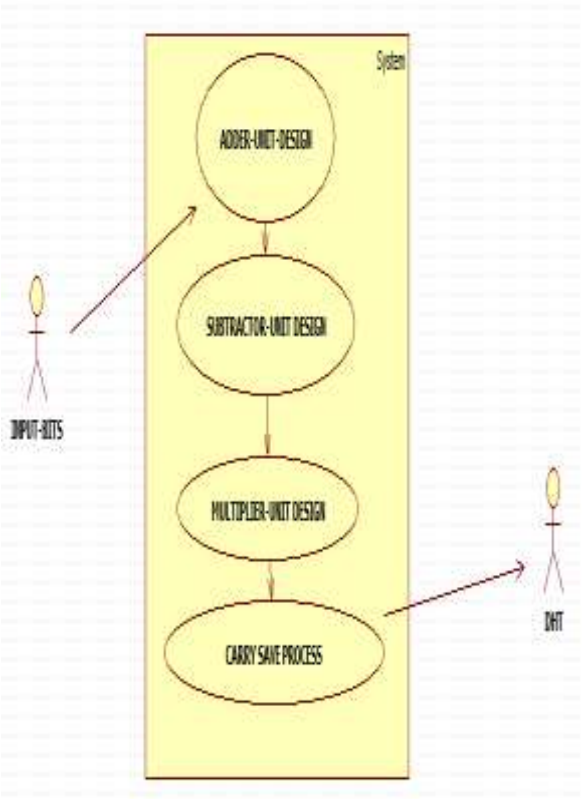

Figure.2. Case Diagram

The input bits are applied to the DHT transform architecture to optimize the internal operation. The DHT transform is used to optimize the adder , subtractor and the multiplier unit. Basically the subtraction operation consists more number of logic gates. The subtractor unit is used to reduce the logic gates function and to optimize the subtractor unit for DHT transform process.

VI. CONCLUSION

In this brief, a new highly parallel VLSI algorithm for the computation of a length-$N = 2n$ DHT having a modular and regular structure has been presented. Moreover, this algorithm can be implemented on a highly parallel architecture having a modular and regular structure with a low hardware complexity by extensively using a subexpression sharing technique and the sharing of multipliers having the same constant.

ACKNOWLEDGEMENTS

The authors thank the Management and Principal of P.S.N.A College of Engineering and Technology for providing good platform and encouragement.

REFERENCES

[1] P. K. Meher, "LUT optimization for memory-based computation," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 57, no. 4, pp. 285–289, Apr. 2010.

[2] P. K. Meher, "New approach to look-up table design and memory-based realization of FIR digital filters," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 3, pp. 592–603, Mar. 2010.

[3] D. F. Chiper and P. Ungureanu, "Novel VLSI algorithm and architecture with good quantization properties for a high-throughput area efficient systolic array implementation of DCT," EURASIP J. Adv. Signal Process., vol. 2011, no. 1, pp1–14, Jan. 2011.

[4] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "New parametric discrete Fourier and Hartley transforms, and algorithms for fast compu-tation," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 58, no. 3, pp. 562– 575, Mar. 2011.

[5] J. S. Wu, H. Z. Shu, L. Senhadji, and L. M. Luo, "Radix 3 × 3 algorithm for the 2-D discrete Hartley transform," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 55, no. 6, pp. 566–570, Jun. 2008.

[6] S. Bouguezel, M. O. Ahmad, and M. N. S. Swamy, "A split vector-radix algorithm for the 3-D discrete Hartley transform," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 53, no. 9, pp. 1966–1976, Sep. 2006.

[7] D. F. Chiper, "Radix-2 fast algorithm for computing discrete Hartley transform of type III," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 59, no. 5, pp. 297–301, May 2012.

[8] H. Z. Shu, J. S. Wu, C. F. Yang, and L. Senhadji, "Fast radix-3 algorithm for the generalized discrete Hartley transform of type II," IEEE Signal Process. Lett., vol. 19, no. 6, pp. 348–351, Jun. 2012.

[9] 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.

[10] R I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.,vol. 43, no. 10, pp. 677–688, Oct. 1996.

