Optimization Result for 16-bit Discrete Hartley Transform using Partition Multiplier and 16-bit RCA with KSA Adder

Anil Kumar Yadav
M. Tech. Scholar
Dept. of Electronics and Communication
TRUBA, Bhopal, India

Prof. Nishi Pandey
Assistant Professor
Dept. of Electronics and Communication
TRUBA, Bhopal, India

Prof. Abhishek Agwekar
Head of Dept.
Dept. of Electronics and Communication
TRUBA, Bhopal, India

Abstract— **A discrete Hartley transform (DHT) algorithm can be efficiently implemented on a highly modular and parallel architecture having a regular structure is consisting of multiplier and adder. DHT is one of the transform used for converting data in time domain into frequency domain using only real values. We have proposed a new algorithm for calculating DHT of length 2^N, where N=3 and 4. We have implemented 16-bit DHT using partition multiplier and 16-bit ripple carry adder (RCA) with kogge stone adder (KSA). Partition multiplier based on KSA as an improvement in place of simple multiplication used in conventional DHT. This paper gives a comparison between conventional DHT algorithm and proposed DHT algorithm in terms of delays and area. It is also comparison of 16-bit conventional adder and proposed RCA with KSA adder in term of delay and area.**

Keywords— ***Discrete Hartley Transform (DHT), Partition Multiplier, KSA, Number of Slice***

I. INTRODUCTION

Signal processing involves analyzing, extraction of information and synthesis of the signal. It depends on the type of signal and the nature of information carried by the signal. In general, it is a mathematical and a graphical tool for operations on signal either for data compression, data transmission, elimination of noise, filtering, or smoothing and identification etc. The signal could be sound, image, a time varying sensed data or control signal, to mention a few. The mathematical representation of a signal, if function of time, is known as time domain representation. The same signal can also be expressed as a sum of sine and cosine functions of frequencies and this representation of signal is called the frequency domain representation. The method of converting a time domain signal into the frequency domain is known as transformation. A time domain signal depicts how the signal changes with time, whereas the frequency domain representation gives information regarding the various frequency components present in the signal. Frequency domain representation is commonly used for visualizing the real world signal.

Digital image compression is the most important part in the multimedia applications which aims to reduce the number of bits in an image data for its efficient storage (less storage area). JPEG based still image compression follows three steps i.e. transform, quantization and coding to compress an image [1]. Reverse process comprising decoding, dequantization and inverse transform is used for image de-compression [2]. Discrete cosine transform (DCT) is used to transform the image from spatial domain to frequency domain. During quantization less important frequencies are discarded and is termed as lossy image compression. Bracewell has drawn attention to the discrete Hartley transform (DHT) as a substitute for the discrete fourier transform (DFT) [3]. Many applications of DHT in signal processing and communications have been presented in the literatures [4]. DHT is used in JPEG based image compression with DHT replacing the DCT in [5]. PSNR comparable to DCT based image transform has been obtained. The advantage of using DHT over DCT is that a dequantizer is not required at the decoder side and hence facilitates saving of hardware resources. Hardware implementation of DHT requires a large number of multipliers. Since multiplier requires much more hardware as compared to adder and additionally consumes more power, it is not recommended in a battery running portable device [6]. Distributed Arithmetic (DA) is a digital signal processing technique that implements multiplication without the use of multiplier. Two approaches have been emerged for DA. One is look–up table based [7] and other is without look-up table. Field programmable gate array (FPGA) is an emerging technology in VLSI design because of its many advantages like reprogramability, low cost, fast design cycle, in-circuit programmability etc [8].

The separable 2-D DHT (SDHT) is proposed in place of DCT for image transform. The image transformed by SDHT is reconstructed after taking inverse SDHT (same as forward with a constant division). No error in the reconstructed image indicates that SDHT can be used for image transform [9]. SDHT is implemented in Xilinx FPGA XC2VP30 device using memory based DA and proposed DA for DHT. Proposed DA for DHT requires very less hardware for its implementation as compared to memory based DA [10]. It also requires less computation as compared to DCT.

II. DISCRETE HARTLEY TRANSFORM

Discrete Hartley Transform is abbreviated for DHT and this transform was proposed by R. V. L. Hartley in 1942. DHT is the analogous to Fast Fourier transform which provides the only real value at any cost. The main difference from the DFT is that it transforms the real inputs to real outputs with no intrinsic involvement of complex value. DFT can be used to compute the DHT, and vice versa.

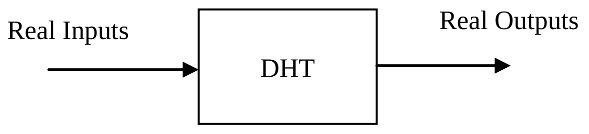


Figure 1: Block Diagram of DHT

In other words, Discrete Hartley transform is used to convert real values into real ones. It requires decomposition of data into stages using butterfly similar to FFT. But the butterfly used in DHT is quite different in terms of coefficients or multipliers. With the increase in number of DHT sequence length the number of coefficients is also increased simultaneously.

Let $N \geq 4$ be a power of two. For any real input sequence $\{x(i) : i = 0,1,2,\dots\dots\dots N-1\}$

$$X(k) = DHT(N)\{x((i)\}$$

$$= \sum_{i=0}^{N-1} x(i).cas[2ki\pi / N] \qquad for\, k = 0,1\dots\dots\dots N-1$$

Where $cas(x) = \cos(x) + \sin(x)$

- ALGORITHM FOR 16 POINT DHT-

We present an implementation of fast DHT algorithm for a length N=1. There are six stages required to complete the butterfly design of N=16 length DHT. These stages include summing stages and coefficient multiplying stages. Before first stage the data sequence are arranged in bit reversed pattern by using any method like permutation. Then in the first stage the pairs of bit reversed patterns are added to form eight terms. In the second stage, one third of the terms are again added and subtracted to form further three terms.

First two stages do not include any multiplication. Remaining terms are multiplied by the first coefficient. In the next stage again two new coefficients are introduced which is multiplied by the lower half of the third stage. In each stage multiplying of coefficients stage precedes its summing stage. After coefficient multiplication it is preceded by its summing stage to form the common terms used in the final stage. Last stage includes only summing of terms. Finally we get the transformed data sequence in order and do not need any permutation.

$$X(0) = (x(0) + x(8)) + (x(4) + x(12)) + (x(2) + x(10)) + (x(6) + x(14)) + (x(1) + x(9)) + (x(5) + x(13)) + (x(3) + x(11)) + (x(7) + x(15))$$

$$X(1) = (x(0) + x(8)) + (x(4) - x(12)) + c_1(x(2) - x(10)) + (x(5) + x(13)) + c_3\{(x(3) - x(11)) + (x(7) - x(15))\} + c_2\{(x(3) - x(11)) - (x(7) - x(15))\}$$

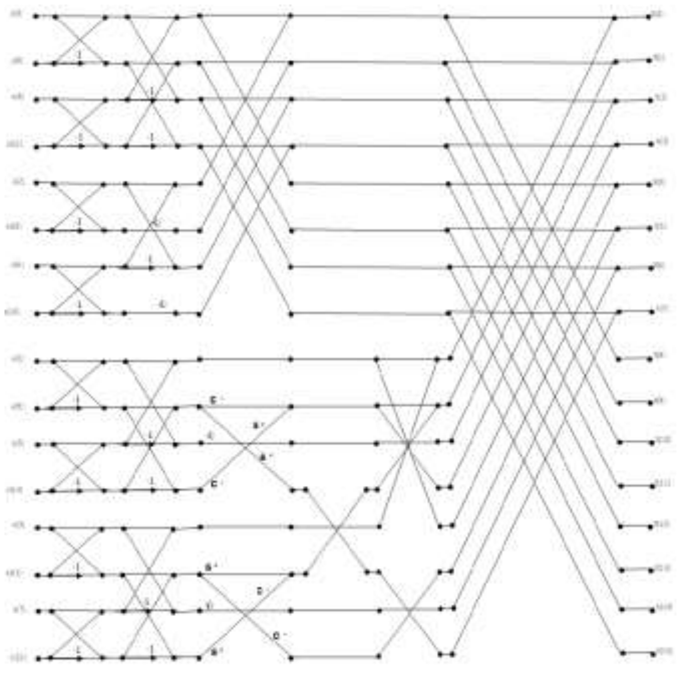


Figure 2: 16-point DHT Butterfly

$$X(2) = (x(0) + x(8)) - \{(x(4) + x(12)) + (x(6) + x(14)\} + c_1\{(x(1) + x(9)) - (x(5) + x(13))\} + (x(2) + x(10)) (x(3) + x(11)) + (x(7) + x(15))$$

$$X(3) = (x(0) - x(8)) - (x(4) - x(12)) + c_1(x(6) - x(14)) + (x(5) + x(13)) + c_3\{(x(3) + x(11)) + (x(7) - x(15))\} + c_2\{(x(3) - x(11)) - (x(7) - x(15))\}$$

$$X(4) = (x(0) + x(8)) + (x(4) + x(12)) + (x(2) + x(10)) + (x(6) + x(14)) + (x(1) + x(9)) + (x(5) + x(13)) - \{(x(3) + x(11)) + (x(7) + x(15))\}$$

$$X(5) = (x(0) + x(8)) + (x(4) - x(12)) + c_1(x(2) - x(10)) + (x(5) + x(13)) + c_3\{(x(3) - x(11)) + (x(7) - x(15))\} - c_2\{(x(5) - x(13)) - (x(1) - x(9))\}$$

$$X(6) = (x(0)+x(8)) + (x(6)+x(14)) - \{(x(2)+x(10)) + (x(4)+x(12))\} + c_1\{(x(3)-x(11)) + (x(7)-x(15))\}$$

$$X(7) = (x(0)+x(8)) - (x(4)-x(12)) + c_1(x(6) - x(14)) - c_3\{(x(5)+x(13)) - (x(1)-x(9))\} + c_2\{(x(5) - x(13)) + (x(1)-x(9))\}$$

$$X(8) = (x(0)+x(8)) + (x(4)+x(12)) + (x(2)+x(10)) + (x(6)+x(14)) - (x(1)+x(9)) - (x(5)+x(13)) - (x(3)+x(11)) - (x(7)+x(15))$$

$$X(9) = (x(0)-x(8)) + (x(4)-x(12)) + c_1(x(2)-x(10)) + (x(5)+x(13)) - c_3\{(x(3)-x(11)) + (x(7)-x(15))\} + c_2\{(x(3)-x(11)) - (x(7)-x(15))\}$$

$$X(10) = (x(0)+x(8)) - \{(x(4)+x(12)) + (x(6)+x(14)\} - c_1\{(x(1)+x(9)) - (x(5)+x(13))\}$$

$$X(11) = (x(0)-x(8)) - (x(4)-x(12)) + c_1(x(6)-x(14)) + c_3\{(x(3)-x(11)) + (x(7)-x(15))\} + c_2\{(x(3)-x(11)) - (x(7)-x(15))\}$$

$$X(12) = (x(0)+x(8)) + (x(4)+x(12)) + (x(2)+x(10)) + (x(6)+x(14)) - \{(x(1)+x(9)) + (x(5)+x(13))\} + (x(3)+x(11)) + (x(7)+x(15))$$

$$X(13) = (x(0)-x(8)) + (x(4)-x(12)) - c_1(x(2)-x(10)) - c_3\{(x(3)-x(11)) + (x(7)-x(15))\} + c_2\{(x(1)-x(9)) - (x(5)-x(13))\}$$

$$X(14) = (x(0)+x(8)) - (x(4)+x(12)) - (x(2)+x(10)) + c_1(x(3)+x(11)) - (x(7)+x(15))$$

$$X(15) = (x(0)-x(8)) - (x(4)-x(12)) - c_1(x(6)-x(14)) - c_3\{(x(13)-x(5)) - (x(1)-x(9))\} + c_2\{(x(1)-x(9)) - (x(5)-x(13))\}$$

III. PROPOSED TECHNOLOGY

Here, the DHT of length N=16 point requires 67 additions and 12 multiplications. We have used a different technique called partition multiplier of ancient Vedic times for multiplication. This multiplication technique reduces the delay and complexity. It converts multiplication into simple logical AND operation using associated circuits of full, half adders and AND gate. Further, the area is reduced by using various compressors for adding the partial products of multiplication.

Partition Multiplier:-

Algorithm for partition method

```
t1 : in  STD_LOGIC_VECTOR (7 downto 0);
t2 : in  STD_LOGIC_VECTOR (7 downto 0);
t3 : out  STD_LOGIC_VECTOR (15 downto 0));
h1<=t1(3 downto 0);
h2<=t1(7 downto 4);
h3<=t2(3 downto 0);
h4<=t2(7 downto 4);
su1<=h1*h3;
su2<=h1*h4;
su3<=h2*h3;
su4<=h2*h4;
ad1<=("00000000" & su1);
ad2<=("0000" & su2 & "0000");
ad3<=("0000" & su3 & "0000");
ad4<=(su4 & "00000000");
t3<=ad1 + ad2 + ad3 + ad4;
```

16-bit adder

A 16-bit carry select adder can be developed in two different sizes namely uniform block size and variable block size. Similarly a 32, 64 and 128-bit can also be developed in two modes of different block sizes. Ripple-carry adders are the simplest and most compact full adders, but their performance is limited by a carry that must propagate from the least-significant bit to the most-significant bit. The various 16, 32, 64 and 128-bit CSLA can also be developed by using ripple carry adders. The speed of a carry-select adder can be improved upto 40% to 90%, by performing the additions in parallel, and reducing the maximum carry delay. Figure 3.6 shows the Regular structure of 16-bit SQRT CSLA. It includes many ripple carry adders of variable sizes which are divided into groups. Group 0 contains 2-bit RCA which contains only one ripple carry adder which adds the input bits and the input carry and results to sum [1:0] and the carry out.

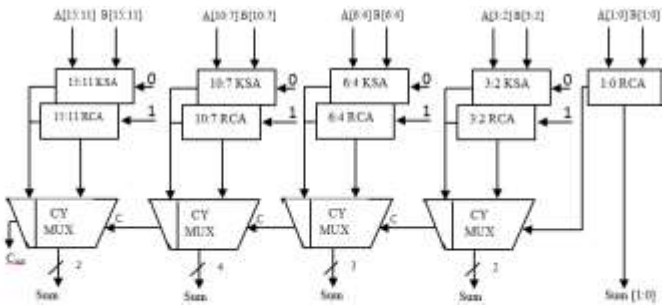


Figure 3: Block Diagram of 16-bit Regular 16-bit SQRT CSLA using KSA

The carry out of the Group 0 which acts as the selection input to mux which is in group 1, selects the result from the corresponding KSA (Cin=0) or RCA (Cin=1). Similarly the remaining groups will be selected depending on the Cout from the previous groups. In Regular CSLA, there is only one RCA to perform the addition of the least significant bits [1:0]. The remaining bits (other than LSBs), the addition is performed by using two KSA corresponding to the one assuming a carry-in of 0, the other a carry-in of 1 within a group. In a group, there are two RCAs that receive the same data inputs but different Cin. The upper adder has a carry-in of 0, the lower adder a carry-in of 1. The actual Cin from the preceding sector selects one of the two RCAs. That is, as shown in the figure 3.6, if the carry-in is 0, the sum and carry-out of the upper KSA is selected, and if the carry-in is 1, the sum and carry-out of the lower RCA is selected. For this Regular CSLA architecture, the implementation code, for the Full Adders and Multiplexers of different sizes (6:3, 8:4, 10:5 up to 12:6) were designed initially. The regular 64-bit, 128-bit CSLA were implemented by calling the regular 16-bit SQRT CSA and all multiplexers.

In the process of structuring a fast convolution the main challenges are to design high speed combinational logic circuit and provide the fast and less time consuming multiplication computation technique. Generally we are having so many types of adders to design high speed multiplier such as Ripple carry, Look ahead carry etc. But these are not efficient with the concern of path delay and number of slices and gates.

Apart from that whenever we construct a digital device we always put the main point is that circuit should have low propagation delay, less number of gates, less number of slices and LUTs. To fulfill this requirement we are using Kogge-Stone adder that is also called prefix adder which is better than other types of adder [2]. For the low bit addition like 2 and 4bit gate delay will be same as other parallel adder but as soon as we design the adder beyond the 4 bit propagation delay or gate delay will be decreased than to other types of adder.

IV. SIMULATION RESULT

Device utilization summary for 8-point, 16-point discrete Hartley transform using Urdhwa parallel multiplier and partition multiplier technique are shown in table 5.6 and table 5.7 respectively. It is observed from the table that the processing unit for 8-point DHT uses 306 number of slice, 588 number of 4-input look up table (LUTs), 256 number of input output bounds (IOBs), 25.343 ns maximum combination path delay using Urdhwa parallel multiplier and uses 289 number of slice, 562 number of 4-input look up table (LUTs), 256 number of input output bounds (IOBs), 24.076 ns maximum combination path delay using partition multiplier method.

Table I: Device Utilization Summary for 8-point DHT

Device	Spantan-3E 3s100evq100-5	
Architecture	DHT using Urdhwa Parallel Multiplier	DHT using Partition Multiplier Method
Number of Slice	530	385
Number of 4 input LUTS	940	693
Number of IOs	264	264
Maximum Combinational Path Delay	30.340 ns	26.076 ns

It is observed from the table that the processing unit for 16-point DHT uses 1834 number of slice, 3233 number of 4-input look up table (LUTs), 536 number of input output bounds (IOBs), 35.027 ns maximum combination path delay using Urdhwa parallel multiplier and uses 1039 number of slice, 1843 number of 4-input look up table (LUTs), 536 number of input output bounds (IOBs), 29.547 ns maximum combination path delay using partition multiplier method.

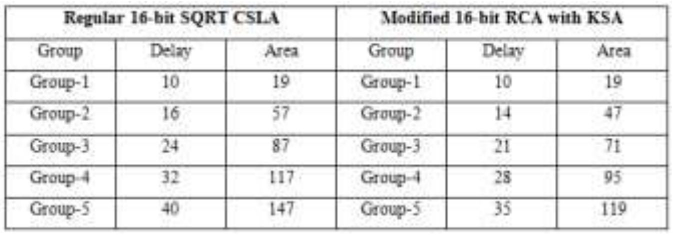
Table II: Device Utilization Summary for 16-point DHT

Device	Spantan-3E 3s100evq100-5	
Architecture	DHT using Urdhwa Parallel Multiplier	DHT using Partition Multiplier Method
Number of Slice	1834	1039
Number of 4 input LUTS	3233	1843
Number of IOs	536	536
Maximum Combinational Path Delay	35.027 ns	29.547 ns

Based on this approach, the regular 16-bit SQRT CSLA and modified 16-bit SQRT CSLA split the structure into five groups. The delay and area estimation of each group are shown in table 5.8. The steps leading to the evaluation are given below:- The group 2 has one 2-bit RCA which has 1 full adder and 1 half adder for C_{in}=0. Instead of another 2-bit RCA with C_{in}=1 a 3-bit KSA is used which adds one to the output from 2-bit RCA. The Area and delay count of group-2 is determined as follows:

Regular 16-bit SQRT CSLA is Gate count = 57(full adder + half adder + MUX)

Table III: Comparison Result



Regular 16-bit SQRT CSLA			Modified 16-bit RCA with KSA		
Group	Delay	Area	Group	Delay	Area
Group-1	10	19	Group-1	10	19
Group-2	16	57	Group-2	14	47
Group-3	24	87	Group-3	21	71
Group-4	32	117	Group-4	28	95
Group-5	40	147	Group-5	35	119

Table IV shows the hardware constraints obtained for a 16-bit adder. Area constraints were calculated for least data arrival times and power was reported for a common clock pulse of 50 ns. The DHT shows reduction in MCPD as well as number of slice consumption for all proposed designs over the design.

Table IV: Comparison Result for Existing and Proposed Design

Parameters	Previous Design	Proposed Design
Area	427	351
Delay	8.894 ns	5.776 ns

V. CONCLUSION

DHT is a new transform used for real value to real value conversion. Urdhwa Triyambakam is an ancient technique for multiplication. DHT is used in various fields such as image processing, space science, scientific applications etc. Delay provided and area required by hardware are the two key factors which are need to be consider. Here we present DHT with 8×8 partition multiplier by using full adder, OR and XNOR gates in different types of adder.

REFRENCES

[1] Nikhil Advaith Gudala, Trond Ytterdal, John J. Lee and Maher Rizkalla, "Implementation of High Speed and Low Power Carry Select Adder with BEC", IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), 2021.

[2] Krishna Sravani Nandam;K. Jamal;Anil Kumar Budati;Kiran Mannem; Manchalla. O. V. P. Kumar, "Design of Multiplier with Dual Mode Based Approximate Full Adder", 5th International Conference on Communication and Electronics Systems (ICCES), IEEE 2020.

[3] Muteen Munawar;Talha Khan;Muhammad Rehman;Zain Shabbir;Kamran Daniel;Ahmed Sheraz;Muhammad Omer, "Low Power and High Speed Dadda Multiplier using Carry Select Adder with Binary to Excess-1 Converter", International Conference on Emerging Trends in Smart Technologies (ICETST), IEEE 2020.

[4] Maytham Allahi Roodposhti;Mojtaba Valinataj, "A Novel Area-Delay Efficient Carry Select Adder Based on New Add-one Circuit", 9th International Conference on Computer and Knowledge Engineering (ICCKE), IEEE 2019.

[5] N. M. Hossain;M. A. Abedin, "Implementation of an XOR Based 16-bit Carry Select Adder for Area, Delay and Power Minimization", International Conference on Electrical, Computer and Communication Engineering (ECCE), IEEE 2019.

[6] Deepti Gautam and Anshuman Singh, "Implementation of DHT Algorithm for a VLSI Architecture" Journal of Basic and Applied Engineering Research, Volume 5, Issue 7; October-December, 2018.

[7] Low Power Approximate Adders for General Computing Using Differential Transmission Gate" on 28.03.2018 National Conference on Innovations in Communication and Computing NCICC" 18 SNS College of Technology.

[8] Omid Akbari, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram, "Dual-Quality 4:2 Compressors for Utilizing in Dynamic Accuracy Configurable Multipliers", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 34, Issue 7, 2017.

[9] G.Challa Ram and D.Sudha Rani, "Area Efficient Modified Vedic Multiplier", 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT).

[10] G. Gokhale and P. D. Bahirgonde, "Design of Vedic Multiplier using Area-Efficient Carry Select Adder", 4th IEEE International Conference on Advancesin Computing, Communications and Informatics (ICACCI-2015), Kochi, August10-13, 2015, India.

[11] G. Gokhale and Mr. S. R. Gokhale, "Design of Area and Delay Efficient Vedic Multiplier Using Carry Select Adder", 4th IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI-2015), Kochi, August 10-13, 2015, India.

[12] Shirali Parsai, Swapnil Jain and Jyoti Dangi, "VHDL Implementation of Discrete Hartley Transform using Urdhwa Multiplier", 2015 IEEE Bombay Section Symposium (IBSS).

[13] Doru Florin Chiper, Senior Member, IEEE, "A Novel VLSI DHT Algorithm for a Highly Modular and Parallel Architecture", IEEE Transactions on Circuits and Systems—Ii: Express Briefs, Vol. 60, NO. 5, May 2013.

[14] Sushma R. Huddar and SudhirRao, Kalpana M., "Novel High Speed Vedi Mathematics Multiplier using Compressors", 978-1 - 4673-5090-7/13/$31.00 ©2013 IEEE.