International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 06, June - 2017

An Efficient Multi Precision Floating Point Complex Multiplier Unit in FFT

Mrs. Yamini Gayathri T
Assistant Professor,
ACS College of Engineering,
Department of ECE,
Bangalore-560074, India

Abstract- **Discrete Fourier Transform (DFT) is a fundamental digital signal processing domain transformation technique used in many applications for frequency analysis and frequency domain processing. Fast Fourier Transform (FFT) is an algorithm to compute discrete transform and its inverse. FFT computation involves addition, multiplications and subtraction. Multipliers are slow performing units. Multiplier has the disadvantages of large area, long propagation delay and consumes power. Low power multiplier design has an important role in the design of low power VLSI systems. As twiddle Factor contains a floating-point complex number, the FFT needs a Floating-point complex multiplier for computations. In some biomedical applications, apart from power, area and speed, the importance has given to the accuracy. The main aim of our study is to improve the accuracy of the multipliers used by using a precision floating point complex multiplier in the unit. For FFT analysis, theverilog code should framed for the various combinations given in the methodology and the ASIC design Flow(Front end and Backend simulation) is carried out using Cadence Verilog ASIC Simulator using 90nm technology to find out power, area and delay.**

Keywords – Double-Precision, twiddle factor, IEEE754 Format, katarsubha urdhva tiryagbhyam algorithm, Mixed-Radix, Floating point

1. INTRODUCTION

The Discrete Fourier Transform (DFT) decomposes a sequence of values into components of different frequencies (Time domain to Frequency domain). The Fast Fourier transform (FFT) is the efficient algorithm to compute the DFT and its inverse. The FFT was proposed by Cooley and tukey to efficiently reduce the time complexity to $O(N\log_2 N)$, where N denotes the FFT size. The hardware implementation of FFT classified into memory-based architecture and pipeline architecture.The Memory based architecture design composes of main processing element (PE) and several memory units. It consumes less power and less hardware cost comparing to other styles. The main disadvantage is long latency, low throughput and can't be parallelized. The Pipeline architecture has get rid of the disadvantages of the memory based style at the cost of an acceptable hardware overhead. The types of the pipeline architecture are single path delay feedback (SDF), single path delay commutator (SDC) and multiple delay commutator (MDC). In addition, pipeline structure is highly regular, which can be easily scaled and parameterized when Hardware Description Language (HDL) is used in the design.

2. OBJECTIVE OF THE STUDY

The main aim of the study is to implement an efficient Floating point complex multiplier in FFT for biomedical applications. The tool used for implementation is Cadence Verilog ASIC simulator with 90nm technology. As the twiddle factor has a complex value, the complex addition and complex multiplication is used. The complex multiplication method is varied and used in twiddle factor multiplication of N point FFT. Then the comparison of power, area and speed has to make for the implemented design of an N point FFT. In addition to power area and speed, Accuracy is improved for the Multiplier by increasing the significant figures for the twiddle factor..

3. LITERATURE SURVEY

Booth encoded Wallace tree multiplier uses, booth encoding to increase the speed of algebra by reducing the number of partial products and Wallace tree for decreasing the number of levels of additions [1]. The three term fused dot product unit implemented in the paper achieved a reduction in area compared to conventional multiplier [2].Golub method can reduce the complexity with a slight increase in latency [2]. To obtain a high throughput, Mixed radix and delay Feedback style is used for implementing the 128 point FFT Processor [3].CORDIC is an efficient and economic approach to compute trigonometric functions using addition, subtraction and shifting operations [3]. The Number of multipliers in complex multipliers has reduced from 4 to 3 and the design using this multiplier proved to be superior to the existing system with complex multipliers of 4 relating to area and time consumption [4]. The Urdhva tiryagbhyam method used in the paper [5] can reduce the time delay by adding the partial products concurrently with the multiplication operations.The Multi-functional floating point multiplier design proposed in the paper [6] has a advantage of Accuracy at the cost of area. The pipelined architecture has advantage of high data throughput, relatively small area and a relatively simple control [7]. As Accuracy requirement decreases, the width of the multiplier decreases [8]. The multiplication of lower bit length mantissa consumes less amount of power [8]. Karatsubha Algorithm is best suited for higher bit length [8].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 06, June - 2017

4. PROPOSED METHODOLOGY

4.1 Complex Multiplication method

4.1.1 Conventional Complex Multiplication

The two complex value a+jb and c+jd can be multiplied as follows.

(a+jb) X (c+jd) =(ac-bd) + j(bc+ad) (1)

Then, Conventional complex multiplier needs 4 multiplications and 1 subtraction and 1 addition.

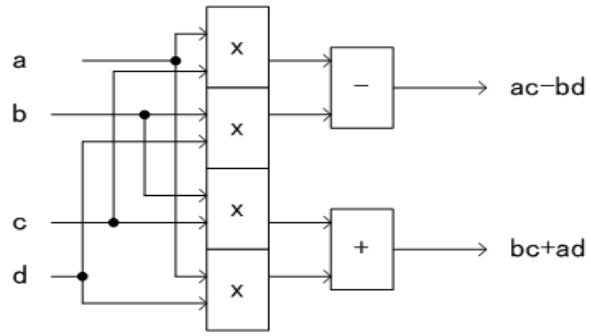


Figure 1. Conventional Complex Multiplication

4.1.2 Golub Complex Multiplication

The two complex value a+jb and c+jd can be multiplied as follows.

(a+jb) X (c+jd) =(ac-bd) + j(bc+ad) (2)

The complex multiplication can be reduced as follows

Real =ac-bd –ad+ad

=a(c-d)+d(a-b) (3)

Imaginary=ad+bc-bd+bd

=d(a-b)+b(c+d) (4)

Then, Golub complex multiplier needs 3 multiplication and 2 subtractions and 3 additions.

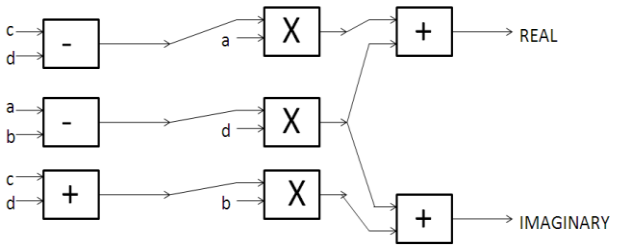


Figure 2: Golub Complex Multiplication

4.1.3 Three term fused dot product unit:

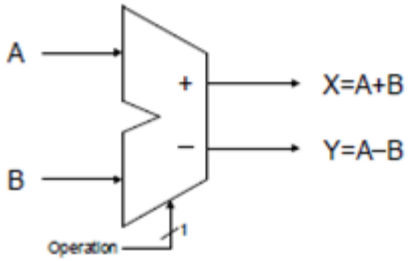


Figure 3. Fused ADD-SUBTRACT unit

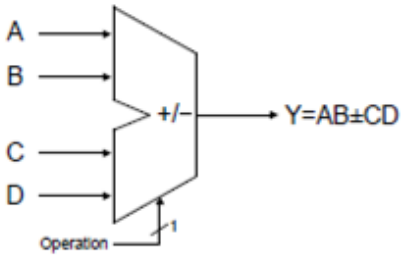


Figure 4. Fused DOT-PRODUCT unit

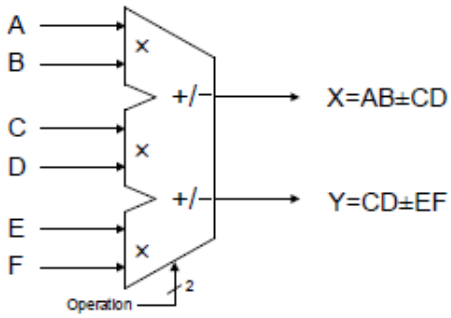


Figure 5. Three term fused dot product unit

Fig (3) shows the add and subtract unit using MUX reducing the hardware complexity.

Fig (4) shows the multiplication, add and subtract unit using MUX reducing the hardware complexity.

Fig (5) shows the three multiplications fused with add and subtract unit using MUX reducing the hardware complexity.

The Three terms fused dot product unit is used for conventional complex multiplications. If fused unit is used for golub complex multiplication, the complexity of architecture will become complex.

4.2 Multi precision floating point Multiplier

Floating point multiplication is one of the crucial operations in many application domains such as image processing, signal processing etc. Every application requires different working features like Precision, low power consumption, low latency, low area etc.

The proposed work based on the multi precision floating point complex multiplier involving six modes of operations.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 06, June - 2017

Mode1: Auto mode

Mode2: double-precision floating multiplier having mantissa size of 8 bit

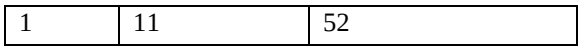
Mode3: double-precision floating multiplier having mantissa size of 16 bit

Mode4: double-precision floating multiplier having mantissa size of 23 bit

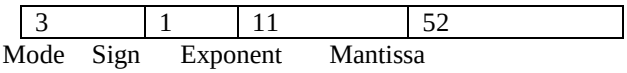
Mode5: double-precision floating multiplier having mantissa size of 23 bit

Mode6: double-precision floating multiplier having mantissa size of 52 bit

The mode with less number of mantissa bits consumes less amount of power. The mode is chosen according to the accuracy needed for the application. If the application needs low power, mode2/mode3 is chosen. If accuracy is given priority mode4, mode5 and mode6 is used in the increasing order of accuracy.
The double precision floating point IEEE754 format is as follows.



The multi precision Floating point format for the proposed work is as below



The different mode select bit combinations for different modes is shown below:

Mode	Mode select bits
Mode1	000
Mode2	001
Mode3	010
Mode4	011
Mode5	100
Mode6	101

Table 1. Modes of Multi-precision floating point multiplier

A Floating-point number is represented in IEEE-754 format as
$\pm s \times b^{e}$ (or) $\pm$significant x base^{exponent}

4.2.1 Sign calculation:
The MSB of floating point number represents the sign bit. The sign of the product will be positive if both the numbers are of same sign and will be negative if numbers are of opposite sign. This logic can be implemented using XOR gate.
4.2.2 Addition of exponents:
The input exponents are added together to get the product exponent. The simple ripple carry adder and ripple borrow subtractor is optimal for exponent addition.

4.2.3 Mantissa Multiplications:
The mantissa multiplication is the most important and complex part of the floating point multiplication. Multiplication operation requires more time compared to addition. As the no of bits increases, there will be an increase in delay and power. As double precision multiplier contains 53 bits of mantissa, it requires 53 X 53 multiplier. For the implementation of 53 X 53 multiplier requires much time for the operations.

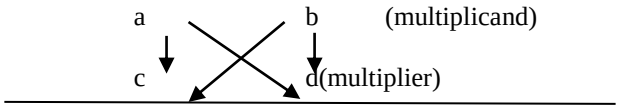
4.3 Karatsuba urdhva tiryagbhyam algorithm
It is the combination of the karatsubha algorithm and urdhva tiryagbhyam algorithm.Kartsubha algorithm is used for the operands of higher bit lengths, But at lower bit lengths it is not suited as it has higher bit lengths. So urdhva tiryagbhyam algorithm is used for lower bit lengths and provides less area and low power advantages when it is used for the lower bit length binary multiplications as the partial products are added in ripple manner. To increase the efficient, ripple adder can be replaced by the carry select and carry save adders. Kartsubha algorithm uses divide and conquer approach where it breaks down the inputs into Most significant half and least significant half and then multiplication is performed. It reduces the number of multipliers required by replacing multiplication operations by addition operations. The Kartsubha algorithm is optimal if width of the inputs is more than 16 bits.In urdhva tiryagbhyam algorithm speed of the multiplications is increased by reducing the number of steps for multiplication using the concept of vertically and crosswise use in Vedic mathematics.

Katarsubha Algorithm:
$X=10^{\wedge(n/2)} * a+ b$ (5)
$Y=10^{\wedge(n/2)} * c+d$ (6)
$X*Y=10^{\wedge n} *ac + 10^{\wedge(n/2)}*(ad+bc) +bd$ (7)
Example:
X= 5678, a=56 b=78
Y=1234, c=12, d=34
Step1: ac=672
Step2: bd=2652
Step3: (a+b)(c+d)=134*46=6164
Step4: ad+bc= step3 value - step2 value - step1 value= 2840
Step5: $10^{\wedge n}$ *ac 6720000
+ bd 2652
$+10^{\wedge(n/2)}$*(ad+bc) 284000
ADD three values= 70006652 (which is X*Y)

Urdhva tiryagbhyam algorithm:

- Actual meaning is vertically and crosswise
- Example:
- X=ab; Y=cd (8)
- X*Y=ab x cd (9)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 06, June - 2017



Left:a*b=l ;

Middle:(a*d)+(b*c)=m

Right:b*d =n

X*Y = lmn

4.4 Booth Encoded Wallace Tree multiplier

The Booth Encoded Wallace Tree multiplier is a combination of booth tree multiplier and Wallace tree multiplier. This algorithm shows a better performance in terms of power and area than booth multiplier. The booth encoding is used to increase the speed of algebra by reducing the number of partial products and Wallace tree module for decreasing number of levels of addition.

Example: -21 X -25 = ???
-21 = 1 0 1 0 1 1 -25 = 1 0 0 1 1 1
1 0 1 0 1 1 -21 (Multiplicand)
1 0 0 1 1 1 -25 (Multiplier)
Booths Recoded Multiplier:
-1 0 +1 0 0 -1

-2 +2 -1 (Bit Pair recoding)
2^4 2^2 2^0 (power of 2)

0 0 0 0 0 0 0 1 0 1 0 1 -1(-21)(2^0)=+21->A
1 1 1 1 0 1 0 1 1 0 0 0 +2(-21)(2^2)=-168->B 0 0 1 0 1 0 1
0 0 0 0 0 -2(-21)(2^4)=+672->C

0 0 1 0 0 0 0 0 1 1 0 1
Final Result = +525 (add the values)

Steps involved:
1. Booths Recoded Multiplier: Take -1 as LSB. Next Lsb: If the values of the multiplier are same, the next LSB is Zero. If the values changes from right to left , then it is +1 and the next change of value will give -1.
2. Bit Pair recoding: The values are taken from the recoding table for booth recoded multiplier values.
3. Bit Pair recoding, respective power of 2 values and multiplicand values are multiplied. The sum values for the three combinations A, B and Care added gives the final result.

Booth Pair		Recoded Bit pair
0	0	0
0	+1	+1
0	-1	-1
+1	0	+2
+1	+1	----
+1	-1	+1
-1	0	-2
-1	+1	-1
-1	-1	----

Table 2. Modified booth Recoding

4.5 Mixed-Radix Algorithm

Radix FFT Algorithm has less number of complex multiplications compared with radix-2 FFT algorithm which is the simplest form of all FFT algorithms. In order to save the number of complex multiplications, Radix-8 Algorithm is chosen. Since the 128-point FFT is not a power of 8, the Mixed-Radix FFT Algorithm, including Radix-2 and Radix-8 FFT algorithm is used.
Hence the hardware can be reduced by using the Mixed-Radix Algorithm.

4.6 DIT vs DIF

The two main types of FFTs are the Decimation In Time (DIT) and Decimation In Frequency (DIF) varieties. Both calculate two butterfly outputs, X and Y , from two butterfly inputs, A and B, and a complex coefficient W. The DIT approach calculates the outputs using the equations: X = A+BW and Y = A−BW, while the DIF approach calculates its outputs using: X = A + B and Y = (A − B)W.

4.7 Twiddle Factor of a 64 Point FFT

The 64 points FFT design needs $(W_{64})^0$ to $(W_{64})^{63}$ 64 complex values as shown in figure 4. The real (I) component is cosine and the imaginary components (Q) is -sine.

$W_{64}=e^{-j(2\pi/64)}$ (10)

$W_{64}=\cos(2\pi/64)-j\sin(2\pi/64)$ (11)

Real part = $\cos(2\pi/64)$ (12)

Imaginary part = $-\sin(2\pi/64)$ (13)

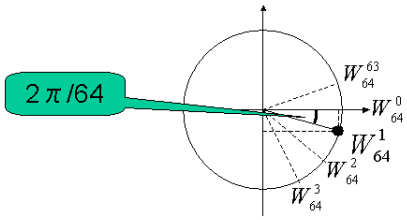


Figure 4.Twiddle Factor for 64 FFT.

TWIDDLE FACTOR	REAL PART	IMAGINARY PART
W_{64}^0	1	0
W_{64}^1	0.99518	-0.00980
W_{64}^2	0.98070	-0.19509
W_{64}^3	0.95694	-0.29028
W_{64}^4	0.92387	-0.38268

Table 3. Few Samples of Twiddle Factor for 64 point FFT

The twiddle factor is of floating point complex number. In DIT Algorithm the twiddle factor has to multiply with the various outputs of the stages. To implement this, floating point complex multipliers is used. The values for twiddle factor are given in the table 1. The value here is restricted

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 6 Issue 06, June - 2017

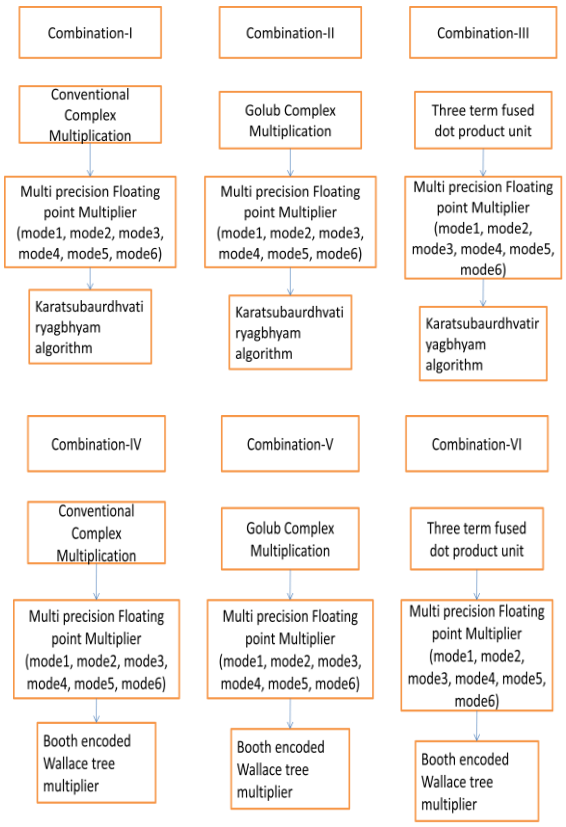
to 4digits.If the significant figures are increased for the twiddle factors, the Accuracy of the FFT analysis can be increased. The floating point numbers as to be converted into an IEEE 754 format.

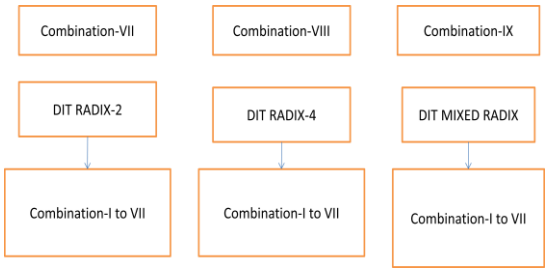
4.8 FFT in Biomedical Applications

Electroencephalography (EEG) is a mechanism of measuring electrical activity of the brain. Upon studying EEG signals, various health conditions can be monitored and diagnosed ex: brain diseases like Alzheimer, tumors, epilepsy, human behavior etc. These signals are recorded from various positions on scalp through electrodes and conductive media. Diagnostic results are made from the spectral content of EEG signal analysis. The sample must be given as an input to the DIT-FFT and the outputs is obtained using the Mixed Radix for 128 point FFT.

5. EXPECTED RESULTS

The FFT is implemented using a DIT algorithm. The power, area, delays and Accuracy must be compared for the following combinations.





From all the above Combinations, an efficient combination is applied for the applications for analysis according to the requirement of power, area, speed and Accuracy because always there will be a tradeoff between these parameters.

REFERENCES

[1] Arathi Ajay, Dr.R.Mary Loudre , BITS Pilani, Dubai, UAE "VLSI Implementation of an improved multiplier for FFT computation in Biomedical Applications"-2015 IEEE computer society Annual Symposium on VLSI.

[2] Sangho Yun and Gerald. E. Sobelman, *University of Minnesota, Minneapolis, MN 55455 USA,* "A Low Complexity Floating-Point Complex Multiplier with a Three-term Dot-Product Unit" -2014 IEEE

[3] Namarata sarode, Rajeev Atluri, P.K Dakhole, "Mixed-Radix and CORDIC Algorithm for implementation of FFT"-2015 IEEE

[4] Dr.Uma B V, Harsha R Kamath, Mohith s, Sreekar V, Shravan Bhagirath, ECE, RVCE, Bangalore,India "Area and time optimized Realization of 16 point FFT and IFFT Blocks by using IEEE754 Single precision complex floating point multiplier"- 2015 ICSCTI, IEEE

[5] Srinivasa Rao, M.Kamaraju,Gudlavalleru Engineering College, AP, India- "An FPGA implementation of high speed and area efficient Double-precision Floating point multiplier using Urdhva tiryagbhyam technique"-2015 IEEE. Conference on PCCCTSG, Kurnool, AP,India.

[6] De Liu, Mingjiang Wang, Yiwen Wang, Hang Su, Harbin Institute of technology Shenzhen graduate school Shenzhen, china "A multi-functional Floating Point Multiplier"- 2015 IEEE.

[7] Guihua Liu1,Quanyuan Feng, Institute of Microelectronics, Southwest Jiaotong University, Southwest University of Science and Technology, **"**ASIC Design of Low-power Reconfigurable FFT Processor" in 2007 IEEE pp 44-46.

[8] Arish S, R K Sharma,NIT, Kurushetra, India, "Run-time reconfigurable multi-precision floating point multiplier design for high speed, low-power applications"-IEEE 2015 SPIN.