The International Journal Of Engineering And Science (IJES)
||Volume||2 ||Issue|| 4 ||Pages||22-26||2013||
ISSN(e): 2319 – 1813 ISSN(p)**:** 2319 – 1805

FFT Algorithms: A Survey

[1,]Pavan Kumar K M, [2,]Priya Jain, [3,]Ravi Kiran S, [4,]Rohith N, [5,]Ramamani K
[1,2,3,4,5,]*Department of E&CE, Bahubali College of Engineering, Shravanabelagola, INDIA*

----------**Abstract**----------
FFT is an efficient tool in the field of signal processing in linear system analysis. It is widely used in OFDM and wireless communication system in today's world. Complex arithmetic modules like multiplier and powering units are now being extensively used in design. The parallel pipelined technology is introduced to increase the throughput of the circuit at low frequency. Based on low power technology of 16-pt. FFT up to 45% of power saving is achieved. The radix-2 is the fastest method for calculating FFT. A FFT is an efficient algorithm to compute the DFT and its inverse. The Fourier transform is the method of changing time representation to frequency representation.

Keywords: *Butterfly architecture, radix-2, FFT, OFDM, Xilinx, FPGA.*

Date of Submission: 12, February, 2013 Date of Publication: 10 April 2013

I. INTRODUCTION

FFT is an extremely robust algorithm that lends itself well to machine computation and is efficiently applied in wireless communication. The invention of FFT is attributed to Cooley and Tukey in 1965. There are two main families of FFT algorithm: The Cooley-Tukey algorithm and the prime factor algorithm. These differ in the way they map the full FFT into smaller sub-transforms. Of, the Cooley-Tukey algorithm there are two types of routines in common use: mixed-radix algorithm and radix-2 algorithm. FFT compute the DFT with greatly reduced number of operations. FFT and its inverse play a significant role in many DSP applications. FFT algorithm started a new era in DSP by reducing the order of complexity of DFT from N^2 to $N\log_2 N$ reduces the number of required complex multiplication compared to the normal DFT. The FFT core is able to perform an N-point FFT in approximately 2.4*N clock cycles. It is applied in a large range of fields and applications such as: Asymmetrical digital subscriber line (ADSL), Digital audio broadcasting (DAB), Digital video broadcasting (DVB) and OFDM systems and it also plays a significant role in radar, medical imaging, spectral analysis and acoustics.

The smallest transform used in 2-point DFT which is known as radix-2. It processes a group of two samples. Radix-2 is the fastest method for calculating FFT. These are amongst the one of large number of FFT algorithm being developed. Radix-2 algorithm are useful if N is a regular power of 2(N=2^p). The term FFT is actually slightly ambiguous because there are several commonly used FFT algorithms. There are two different radix-2 algorithms, they are: decimation-in-time (DIT) and decimation-in-frequency(DIF) algorithms. A butterfly unit block consisting of N/2 butterflies. Each one containing two (N/2)*16-bits ROMs to store the sine and cosine of the twiddle factors, four 16*16 multipliers in 2's complement, six 32-bits accumulators and two special operators to adequate the data format.

A lot of work has been done in the field of FFT. In this paper a survey has been made on the design to build a simple and efficient architecture to keep the area and power consumption as low as possible while meeting the timing constraints.

II. FAST FOURIER TRANSFORMS

2.1 A Modified Fast FFT Algorithm for OFDM:

OFDM is a digital multicarrier modulation technology. It uses a large number of closely spaced orthogonal subcarrier. The benefit of OFDM approach rather than other modulation approaches is the efficient use of bandwidth using overlapping property [1]. A typical OFDM system consists of two parts: receiver and transmitter. The receiver has four important blocks which are serial-to-parallel block. The common method applied to match with the order or requirement of a system is to extend the input data sequence x(n) by padding number of zeroes at the end of it. In general most of the FFT or pruning algorithms are literally proposed. Some of them are implemented either in MATLAB or in FORTRAN. The main constraints of those techniques are that, they are not dynamically efficient enough for any type of input dataset [1]. It is very rare to find an algorithm implemented in high level computer program, which is able to show the required actual execution time for an FFT operation.

An input zero traced FFT pruning (IZTFFTP) algorithm, suitable for NC-OFDM based transceiver has been proposed. Result shows IZTFFTP is more efficient than ordinary FFT. To increase the efficiency of the FFT technique several pruning and different other techniques have been proposed by many researchers.

2.2 Optimized FFT Design using Constant Co-efficient Multiplier:

Multiplier designs require large area and consume a considerable amount of power per computation [2]. Use of general purpose multipliers lead to power being wasted. Thus it is proposed to use dedicated powering units which performs a specific function. The advantage with using dedicated KCM is that they consume less power compared to general purpose multipliers.

Constant Co-efficient Multiplier (KCM) is a special case of multiplication. By using dedicated resources one can save a considerable amount of power which allows designers to remain inside their power budgets. Recently, lot of research has been conducted in order to develop different methodologies [2] to implement KCM giving more importance to improve delay & reducing area constraints.The use of dedicated optimizing KCM units which is will be advantageous in field of area and power requirements is presented. The salient features are easy and simple to implement, low power consumption, less area, better timing can be achieved.

2.3 Design of Low Power High Performance 16-Point 2-Parallel Pipelined FFT Architecture:

Parallel pipelined FFT processors are required to meet the growing demand of high processing rate as they can increase the FFT processing rate greatly high throughput and high power efficiency applications [3]. These can operate at a lower frequencies compared to that of pipelined FFT's thus resulting in lower power consumption.

A lower power butterfly structure for 16 point 2-parallel pipelined FFT architecture implementation has been proposed [3]. This structure reduces the total power dissipation of FFT along with high performance IDR commutator and also power butterfly structure. Thus the technology to minimize the power consumption of FFT architecture by reducing number of functional blocks used to implement FFT processor has been presented.

2.4 Implementing FFT Algorithms on FPGA:

FPGA technology is quite mature for DSP applications due to fast processing VLSI technology. The FPGA devices provide fully programmable system on chip environment by incorporating the probability of programmable logic devices and the architecture of gate arrays [4]. The introduction of verilog hardware descriptive language provided a modeling and simulation environment for fast prototyping digital circuits and system on FPGA.

Some practical FFT algorithm including Cooley-Tukey, Good-Thomas, radix-2 and radar methods are modeled by VHDL and their performance and compared in terms of chip area utilization and maximum frequency operation. The result demonstrated that the good-Thomas method was faster than Cooley-Tukey and the radar had worst operating frequency on FPGA between all the proposed FFT approaches [4]. The utilized FPGA chip area increased by increasing the number or FFT points for all methods.

2.5 Efficient Design and Implementation of FFT:

The Fourier transform is the method of changing time representation to frequency representation. The DFT is one of the Fourier transform used in Fourier analysis [5]. The DFT of a given sequence x[n] can be computed using

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \qquad 0 \le k \le N-1$$

$W_N = e^{-i2\pi n/N}$

Where W_N is a twiddle factor,

FFT uses a standard three loop structure for the main FFT computation. It is an efficient algorithm to compute the DFT and its inverse. The Fast Fourier Transform is an optimized computational algorithm to implement the Discrete Fourier Transform to an array of 2^N samples where, N is the length of samples. It allows determining the frequency of a discrete signal, representing the signal in the frequency domain, convolution, etc. This algorithm has a complexity of O(N*log2(N)).The ordering minimizes the number of fetches or computations of the twiddle-factor values. By considering simple butterfly first for FPGA implementation [5] then the result of complete FFT is observed. As a result the time taken by each block in between language is obtained.

2.6 Low Power Hardware Implementation of High Speed FFT Core:

Fourier transforms play an important role in many digital signal processing applications including speech, signal and image processing. The FFT algorithm, first explained by Cooley and Tukey, opened a new area in digital signal processing by reducing the order of complexity of DFT from $N2$ to $2\ N \log N$. Parallel-pipelined FFTs are preferred for both high throughput and low power consumption. In real-time applications, input data is a sequential stream. Implementing commutator with no switching activity, hence achieving a significant power saving as compared to previous commutator architectures has been focused.

A parallel pipelined architecture for 16 point radix-4 DIF FFT in fixed point representation is proposed and implemented has been proposed [6]. Several novel low power techniques: multiplier-less, DR commutator and LB butterfly are implemented.

III. ALGORITHMS FOR FFT COMPUTATION

In this section we discuss some of the popular methods to compute FFT.

3.1 Cooley-Tukey:

The Cooley-Tukey algorithm was published in 1965. It has been the most widely used FFT algorithm. The basic idea of the algorithm is to divide the N-point DFT into M, N/M point DFTs [7, 12], hence if M=2 then it is divided into two N/2 DFTs. These are called the radix-2. Similarly we have radix-4, 8, 16…etc. Although the basic idea is recursive, most traditional implementations rearrange the algorithm to avoid explicit recursion. The Cooley–Tukey algorithm divides the DFT into smaller DFTs, so it can be combined arbitrarily with any other algorithm for the DFT [7, 12].

3.2 Winograd Algorithm:

In this algorithm the basic idea behind is that it factorizes Z^N -1 into various polynomials having coefficients of 1, 0, or −1, and thus it require few multiplications for its operation, so Winograd can be used to obtain minimal-multiplication FFTs. It is often used to find efficient algorithms for small factors[8]. Winograd showed that the DFT can be computed with only irrational multiplications, hence reducing the number of multiplications considerably, but at the cost of hardware. The modern hardware architecture consist of multiplier blocks, hence this is not considered to be a weakness so far. It is generally used with Rader's algorithm [8, 12].

3.3 Rader- Brenner Algorithm:

In this the complex multiplications are replaced by multiplication of complex number by purely real or imaginary number. It is realized by computing an N-point DFT with $N=2^t$ [9, 12].

3.4 Brunn's Algorithm:

Bruun's algorithm [10] is a fast Fourier transform (FFT) algorithm based on an unusual recursive polynomial-factorization approach. It was proposed for powers of two by G. Bruun in 1978.Its operation involves only real co-efficient until the last computation stage. Because of this fact, it was initially proposed as a way to efficiently compute the discrete Fourier transform (DFT) of real data [10].

3.5 Prime factor Algorithm [Good-Thomas]:

The prime-factor algorithm is a fast Fourier transform algorithm. It is also called as the Good-Thomas algorithm that re-expresses the discrete Fourier transform of a size $N = N_1N_2$ as a two-dimensional $N_1 \times N_2$ DFT, but *only* for the case where N_1 and N_2 are relatively prime. These smaller transforms of size N_1 and N_2 can then be evaluated by applying PFA recursively or by using some other FFT algorithm.

IV COMPARATIVE ANALYSIS OF FFT ALGORITHMS

A comparative analysis of the FFT algorithms discussed in section III has been tabulated as follows. The table I give a performance comparison of the algorithms discussed in the paper.

TABLE 1: COMPARATIVE ANALYSIS OF DIFFERENT FFT ALGORITHMS

Algorithms	Cost	Complexity	Operating Frequency	Operation
Cooley-Tukey algorithm	Low	Less	Worst	Divide N-pt DFT into M,N/M pt DFT's
Winograd algorithm	High	More	Good	Factorizes Z^N-1 into various polynomials
Rader-Brenner algorithm	High	More	Worst	Computes N-pt DFT with $N=2^t$
Brunn's algorithm	Moderate	Less	Good	Computes DFT of real co-efficient
Prime factor algorithm	Low	Moderate	Better	Re-expresses the DFT but only for the case where N_1 and N_2 are relatively prime.

V CONCLUSION

We have described the basic algorithms for one-dimensional radix-2 and mixed-radix FFTs. Modeling and hardware description of some FFT approaches such as Cooley-Tukey, Good-Thomas, radix-2 and Rader FFT algorithms by Verilog hardware description language and realization of them on Xilinx FPGA chip was proposed. The power and area impact parameters of each technique has been observed and compared with the conventional FFT blocks to analyze the performance. Based on this analysis the proposed low power techniques of 16-point FFT, up to 45% power saving is achieved. In this paper a survey has been made on the comparative study of the different FFT algorithms. Thus we can conclude that Good-Thomas algorithm is better in comparison to the remaining algorithms.

REFERENCES

[1] Niladri Mandal, Souragni Ghosh (2012). *A Modified Fast FFT Algorithm for OFDM*. International Journal of Soft Computing and Engineering (IJSCE), Vol.1, Issue-6, January 2012.

[2] Chandan.M,S.L.Pinjare, Chandra Mohan Umapthy Chandan M, S.L Pinjare (2012). *Optimized FFT Design using Constant Co-efficient Multiplier*. International Journal of Emerging Technology and Advanced Engineering, Vol. 2, Issue 6, June 2012.

[3] Senthil Sivakumar M & Banupriya M & Arockia Jayadhas S (2012). *Design of Low Power High Performance 16-Point 2-Parallel Pipelined FFT Architecture*. International Journal of Electronics,`Communication& Instrumentation Engineering Research and Development (IJECIERD). Vol. 2, Issue 3 September 2012.

[4] Arman Chahardahcherik, Yousef S. Kavian, Otto Strobel, and Ridha (2011). *Implementing FFT Algorithms on FPGA* (IJCSNS). International Journal of Computer Science and Network Security, Vol. 11, No.11, November 2011.

[5] Sneha N.Kherde, Meghana Hasamnis(2011). *Efficient Design and Implementation of FFT* . International Journal of Engineering Science & Technology, Vol. 3 Issue Sup, p10, February 2011.

[6] M. Kannan and S.K. Srivatsa(2007). *Low Power Hardware Implementation of High Speed FFT Core*. Journal of Computer Science. Vol. 3, Issue 6, 2007.

[7] Cooley, James W., and John W. Tukey (1965) *An algorithm for the machine calculation of complex Fourier series, Math. Computer*. Vol. 19, 1965.

[8] C. Rader(1965), *Discrete Fourier Transform when the Number of Data Samples is Prime*, *Proceedings of the IEEE* 56 , 1968, Vol. 19, No. 90 (April 1965).

[9] Brenner, N.; Rader, C. (1976). *A New Principle for Fast Fourier Transformation*. IEEE Acoustics, Speech & Signal Processing, Vol. 23, Issue 3, Jun 1976.

[10] Aniket Shukla, Mayuresh Deshmukh(2012), *Comparative Study of Various FFT Algorithm Implementation on FPGA*, International Journal of Emerging Trends in Signal Processing Vol.1, Issue 1, November 2012.