iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: Issue **Issue:** ssue-1 **Month of publication:** October 2014

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology(IJRASET)

Computational Performances of OFDM using Different Pruned Radix FFT Algorithms

Alekhya Chundru[1], P.Krishna Kanth Varma[2]

M.Tech Student, Asst Professor Department Of Eelectronics and Communications,
SRKR Engineering College,
Andhra Pradesh, India

Abstract- The Fast Fourier Transform (FFT) and its inverse (IFFT) are very important algorithms in signal processing, software-defined radio, and the most promising modulation technique i.e. Orthogonal Frequency Division Multiplexing (OFDM). From the standard structure of OFDM we can find that IFFT/FFT modules play the vital role for any OFDM based transceiver. So when zero valued inputs/outputs outnumber nonzero inputs/outputs, then general IFFT/FFT algorithm for OFDM is no longer efficient in term of execution time. It is possible to reduce the execution time by "pruning" the FFT. In this paper we have implemented a novel and efficient input zero traced radix FFT pruning (algorithm based on radix-2 DIF FFT, radix-4 DIF FFT, radix-8 DIF FFT). An intuitive comparison of the computational complexity of orthogonal frequency division multiplexing (OFDM) system has been made in terms of complex calculations required using different radix Fast Fourier transform techniques with and without pruning. The different transform techniques are introduced such as various types of Fast Fourier transform (FFT) as radix-2 FFT, radix-4 FFT, radix-8 FFT, mixed radix 4/2, mixed radix 8/2 and split radix 2/4. With intuitive mathematical analysis, it has been shown that with the reduced complexity can be offered with pruning, OFDM performance can be greatly improved in terms of calculations needed.

Index terms- OFDM (Orthogonal frequency division multiplexing), Fast Fourier Transform (FFT), Pruning Techniques, MATLAB.

I. INTRODUCTION

Orthogonal Frequency Divisional Multiplexing (OFDM) is a modulation scheme that allows digital data to be efficiently and reliably transmitted over a radio channel, even in multi-path environments [1]. In OFDM system, Discrete Fourier Transforms (DFT)/Fast Fourier Trans- forms (FFT) are used instead of modulators. FFT is an efficient tool in the fields of signal processing and linear system analysis. DFT isn't generalized and utilized widely until FFT was proposed. But the inherent contradiction between FFT's spectrum resolution and computational time consumption limits its application. To match with the order or requirement of a system, the common method is to extend the input data sequence x(n) by padding number of zeros at the end of it and which is responsible for a increased value of computational time. But calculation on undesired frequency is unnecessary. As the OFDM based cognitive radio [2] has the capability to nullify individual sub carriers to avoid interference with the licensed user. So, that there could be a large number of zero valued inputs/outputs compare to non-zero terms. So the conventional radix FFT algorithms are no longer efficient in terms of complexity, execution time and hardware architecture. Several researchers have proposed different ways to make FFT faster by "pruning" the conventional radix FFT algorithms.

In this paper we have proposed an input zero traced radix DIF FFT pruning algorithm for different radix FFT algorithms, suitable for OFDM based transceiver. The computational complexity of implementing radix-2, radix-4, radix-8, mixed radix and split radix Fast Fourier Transform with and without pruning has been calculated in an OFDM system and compared their performance. Result shows IZTFFTP of radix algorithms are more efficient than without pruning.

II. OFDM SYSTEM MODEL

OFDM is a kind of FDM (Frequency Division Multiplexing) technique in which we divide a data stream into a number of bit streams which are transmitted through sub-channels [3].

The characteristics of these sub-channels are that they are orthogonal to each other. As the data that are transmitted through a sub-channel at a particular time are only a portion of the data transmitted through a channel so bit rate in a sub-channel can be kept much low. After splitting the data in *N* parallel data streams each stream is then mapped to a tone at a

unique frequency and combined together using the Inverse Fast Fourier Transform (IFFT) to yield the time domain waveform to be transmitted [4]. After IFFT is done, the time domain signals are then converted to serial data and cyclic extension is added to the signal. Then the signal is transmitted. At the receiving side we do the reverse process to get original data from the received one [4,5].

In case of deep fade, several symbols in single carrier is damaged seriously, but in parallel transmission each of *N* symbol is slightly affected. So even though the channel is frequency selective, the sub-channel is flat or slightly frequency selective. This is why OFDM provide good protection against fading [6].

In an OFDM system there are *N* numbers of sub-channels. If *N* is high then it will be very complex to design a system with *N* modulators and demodulators. Fortunately, it can be implemented alternatively using DFT/FFT to reduce the high complexity. A detailed system model for OFDM system is shown in Figure 1 [5,6].

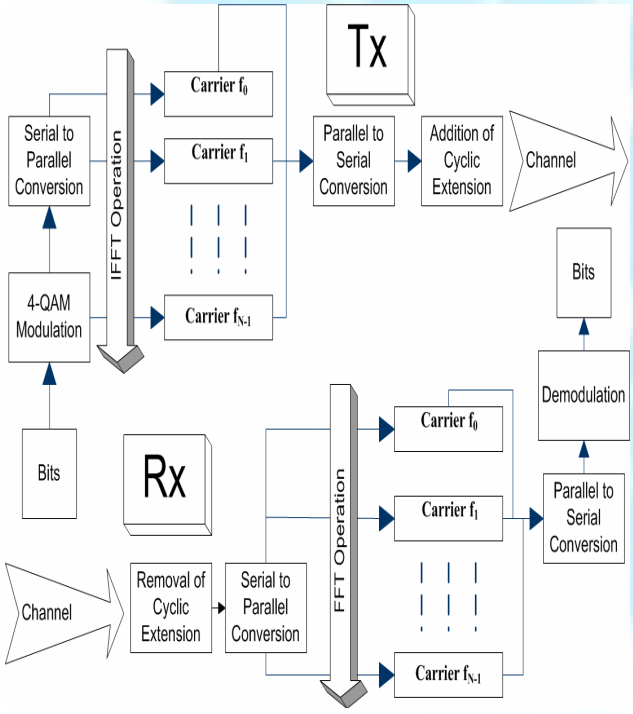


Figure1: OFDM System Model

III. FOURIER TRANSFORM ALGORITHM

Discrete Fourier Transform (DFT) computational complexity is so high that it will cause a long computational time and large power dissipation in implementation. Cooley and Tukey provided a lot of ways to reduce the computational complexity. From that, many fast DFT algorithms have been developing to reduce the large number of the computational complexity, and these fast DFT algorithms are named fast Fourier transform (FFT) algorithms. Decomposing is an important role in the FFT algorithms. There are two decomposed types of the FFT algorithm. One is decimation-in-time (DIT), and the other is decimation-in-frequency (DIF). There is no difference in computational complexity between these two types of FFT algorithm. Different Radix DIF algorithms we used are

A. Radix-2 DIF FFT Algorithm

Decomposing the output frequency sequence $X[k]$ into the even numbered points and odd numbered points is the key component of the Radix-2 DIF FFT algorithm [6]. We can divide $X[k]$ into 2r and 2r+1, then we can obtain the following equations：

$$X[2r] = \sum_{n=0}^{N-1} x[n] W_N^{n(2r)} \quad (1)$$

$$X[2r+1] = \sum_{n=0}^{N-1} x[n] W_N^{n(2r+1)} \quad (2)$$

$$r = 0,1,2,..,\left(\frac{N}{2}\right) - 1$$

Because the decomposition of the Equation (1) and Equation (2) are the same, we only use Equation (1) to explain as shown in Equation (3).

$$X[2r] = \sum_{n=0}^{\frac{N}{2}-1} x[n] W_N^{n(2r)} + \sum_{n=\frac{N}{2}}^{N-1} x[n] W_N^{n(2r)} \quad (3)$$

Finally, by the periodic property of twiddle factors, we can get the even frequency samples as

$$X[2r] = \sum_{n=0}^{\frac{N}{2}-1} (x[n] + x[n+N/2])\ W_N^{n(2r)} \quad (4)$$

$$r = 0,1,2,..,\left(\frac{N}{2}\right) - 1$$

Similarly, the odd frequency samples is

$$X[2r+1] = \sum_{n=0}^{\frac{N}{2}-1} \left(x[n] - x\left[n+\frac{N}{2}\right] \right) W_N^{n(2r)} W_N^{2n}$$

$r = 0,1,2,..,\left(\frac{N}{2}\right) - 1$ (5) From Equation (4) and (5), we can find out the same components, x[n] and x[n+N/2], so we can combine the two equations as one basic butterfly unit shown in Figure 2. The solid line means that $x[n]$ adds $x[n + N / 2]$, and the meaning of the dotted line is that $x[n]$ subtracts $x[n + N / 2]$.

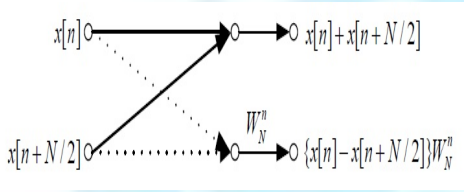


Figure 2: The butterfly signal flow graph of radix-2 DIF FFT

We can use the same way to further decompose N-point DFT into even smaller DFT block. So from the radix-2 dif FFT, there is a reduction of number of multiplications, which is about a factor of 2, showing the significance of radix-2 algorithm for efficient computation. So this algorithm can compute *N*-point FFT in *N/2* cycles.

B.Radix-4 DIF FFT

In case *N*-data points expressed as power of 4^M, we can employ radix-4 algorithm [9] instead of radix-2 algorithm for more efficient estimation. The FFT length is 4^M, where M is the number of stages. The radix-4 DIF fast Fourier transform (FFT) expresses the DFT equation as four summations then divides it into four equations, each of which computes every fourth output sample. The following equations illustrate radix-4 decimation in frequency.

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk} \quad (6)$$

$$= \sum_{n=0}^{\frac{N}{4}-1} x(n)\, W_N^{nk} + \sum_{n=\frac{N}{4}}^{\frac{2N}{4}-1} x(n)\, W_N^{nk} + \sum_{n=\frac{2N}{4}}^{\frac{3N}{4}-1} x(n)\, W_N^{nk} + \sum_{n=\frac{3N}{4}}^{N-1} x(n)\, W_N^{nk} \quad (7)$$

Equation (7) can thus be expressed as

$$X(k) = \sum_{n=0}^{\frac{N}{4}-1} \left[\begin{matrix} x(n) + (-j)^k x(n+N/4) + (-1)^k \\ x(n+N/2) + (j)^k x(n+3N/4) \end{matrix} \right] W_N^{nk} \quad (8)$$

So, Equation (8) can then be expressed as four N/ 4 point DFTs. The simplified butterfly signal flow graph of radix-4 DIF FFT is shown in Figure 3.

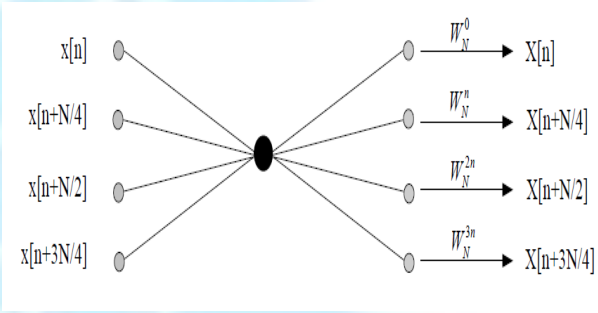


Figure 3: The simplified butterfly signal flow graph of radix-4 DIF FFT

This algorithm results in $(3/8)N \log_2 N$ complex multiplications and $(3/2)N \log_2 N$ complex additions. So the number of multiplications is reduced by 25%, but the number of addition is increased by 50%.

C.Radix-8 DIF FFT

Comparing with the conventional radix-2 FFT algorithm and radix-4 FFT algorithm, the advantage of developing radix-8 FFT algorithm is to further decrease the complexities, especially the number of complex multiplications in implementation. We can split Equation (2.1) and replace index k with eight parts, including 8r, 8r+1,8r+2, 8r+3, 8r+4, 8r+5, 8r+6, and 8r+7. Hence, we can rewrite Equation (6) and obtain the Equation (9).

$X(8r + l) =$

Special Issue-1, October 2014
ISSN: 2321-9653

$$= \sum_{r=0}^{N-1} \left\{ \begin{pmatrix} x[n] \\ +x[n+{}^{2N}/_{8}]W_4^{l} \\ +x[n+{}^{4N}/_{8}]W_4^{2l} \\ +x[n+{}^{6N}/_{8}]W_4^{-l} \end{pmatrix} + \begin{pmatrix} x[n+{}^{N}/_{8}] \\ +x[n+{}^{3N}/_{8}]W_4^{l} \\ +x[n+{}^{5N}/_{8}]W_4^{2l} \\ +x[n+{}^{7N}/_{8}]W_4^{-l} \end{pmatrix} W_8^{l} \right\} W_N^{nl} W_{N/8}^{nr}$$

(9)

The butterfly graph can be simplified as shown in Figure 4

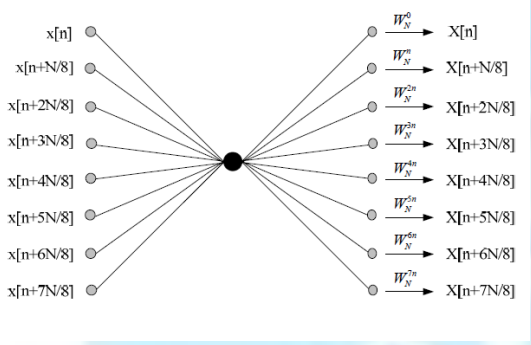


Figure 4: The simplified butterfly signal flow graph of radix-8 DIF FFT

D. Mixed radix DIF FFT There are two kinds of mixed-radix DIF FFT algorithms. The first kind refers to a situation arising naturally when a radix-*q* algorithm, where $q = 2^m > 2$, is applied to an input series consisting of $N = 2^k \times q^s$ equally spaced points, where $1 \le k < m$. In this case, out of necessity, *k* steps of radix-2 algorithm are applied either at the beginning or at the end of the transform, while the rest of the transform is carried out by *s* steps of the radix-*q* algorithm.

For example if $N = 2^{2m+1} = 2 \times 4^m$, the mixed-radix algorithm [7][8] combines one step of the radix-2 algorithm and m steps of the radix-4 algorithm. The second kind of mixed-radix algorithms in the literature refers to those specialized for a composite $N = N0 \times N1 \times N2 \ldots \times Nk$. Different algorithms may be used depending on whether the factors satisfy certain restrictions. Only the 2×4^m of the first kind of mixed-radix algorithm will be considered here.

The mixed-radix 4/2 butterfly unit is shown in Figure5.

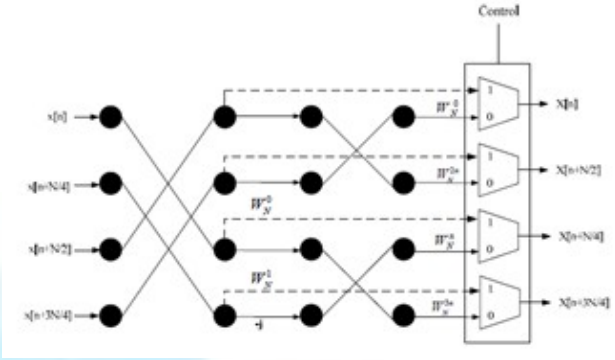


Figure 5: The butterfly signal flow graph of mixed-radix-4/2 DIF FFT

It uses both the radix-2^2 and the radix-2 algorithms can perform fast FFT computations and can process FFTs that are not power of four. The mixed-radix 4/2, which calculates four butterfly outputs based on X(0)~X(3). The proposed butterfly unit has three complex multipliers and eight complex adders.

E. Split-Radix FFT Algorithms

Split-radix FFT algorithm assumes two or more parallel radix decompositions in every decomposition stage to fully exploit advantage of different fixed-radix FFT algorithm. As a result, a split-radix FFT algorithm generally has fewer counts of adder and multiplication than the fixed-radix FFT algorithms, while retains applicability to all power-of-2 FFT length.

More computational complexity of the odd frequency terms than the even frequency terms, so we can further decompose the odd terms to reduce complexities. If we use radix-2 DIF FFT algorithm for the even frequency terms and the radix-2^2 DIF FFT algorithm for the odd parts, we can obtain the split-radix 2/4 algorithm [10,11] as shown in the equation
in the Equation (10).

$$X[2k] = \sum_{n=0}^{\frac{N}{2}-1} \left(x[n] + x\left[n + \frac{N}{2}\right] \right) W_N^{n(2k)} \quad (10)$$

$$k = 0,1,2,\ldots,\left(\frac{N}{2}\right) - 1$$

$$X(4k+1) =$$

$$\sum_{n=0}^{\frac{N}{4}-1} \left[\begin{matrix} x(n) + (-j)x(n+N/4) \\ +(-1)x(n+N/2) + (j)x(n+3N/4) \end{matrix} \right] W_N^n W_N^{nk} \quad (11)$$

$$X(4k+3) =$$

$$\sum_{n=0}^{\frac{N}{4}-1}\begin{bmatrix} x(n)+(j)x(n+N/4) \\ -x(n+N/2)+(-j)x(n+3N/4)\end{bmatrix} W_N^{3n} W_N^{nk}$$

(12)

Thus the *N*-point DFT is decomposed into one *N/2* -point DFT without additional twiddle factors and two *N/4* -point DFTs with twiddle factors. The *N*-point DFT is obtained by successive use of these decompositions up to the last stage. Thus we obtain a DIF split-radix-2/4 algorithm. The signal flow graph of basic butterfly cell of split-radix-2/4 DIF FFT algorithm is shown in Figure 6

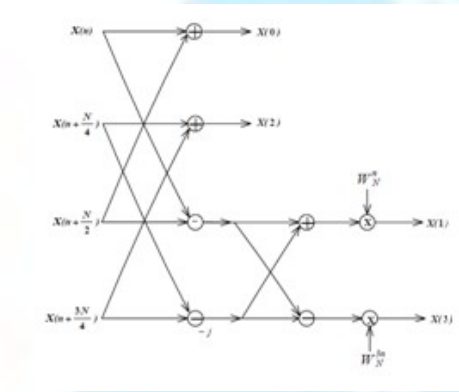


Figure 6: The butterfly signal flow graph of mixed-radix-2/4 DIF FFT

we have $X(0) = x(n) + x\left(n+\frac{N}{2}\right)$

$$X(2) = x\left(n+\frac{N}{4}\right) + x\left(n+\frac{3N}{4}\right)$$

$$X(1) = \begin{bmatrix} x(n)+(-j)x(n+N/4) \\ +(-1)x(n+N/2)+(j)x(n+3N/4)\end{bmatrix} W_N^n$$

$$X(3) = \begin{bmatrix} x(n)+(j)x(n+N/4) \\ -x(n+N/2)+(-j)x(n+3N/4)\end{bmatrix} W_N^{3n}$$

(13)

As a result, even and odd frequency samples of each basic processing block are not produced in the same stage of the complete signal flow graph. This property causes irregularity of signal flow graph, because the signal flow graph is an "L"-shape topology.

IV PRUNING TECHNIQUES

To increase the efficiency of the FFT technique several pruning and different other techniques have been proposed by many researchers. In this paper, we have implemented a new pruning technique i.e. IZTFFTP by simple modification and some changes and also includes some tricky mathematical techniques to reduce the total execution time.

Zero tracing- as in wide band communication system a large portion of frequency channel may be unoccupied by the licensed user, so no. of zero valued inputs are much greater than the non-zero valued inputs in a FFT/IFFT operation at the transceiver. Then this algorithm will give best response in terms of reduced execution time by reducing the no. of complex computation required for twiddle factor calculation. IZTFFTP have a strong searching condition, which have an array for storing the input & output values after every iteration of butterfly calculation. In a input searching result whenever it found "zero" at any input, simply omit that calculation by considering useful condition based on radix algorithm used.

A Input Zero Traced Radix-2 DIF FFT Pruning

In radix-2 since we couple two inputs to obtain two outputs, we therefore have 4 combinations of those two inputs at radix-2 butterfly. Now there exist three conditions only based upon zeros at the input.

- No zero at input: No pruning happens in this case, butterfly calculations are same as conventional radix-2.
- Any one input zero: Output will be only the copied version of input available, butterfly calculations are reduced compared to conventional radix-2.
- All zero input: Output is zero and is obtained from mathematical butterfly calculations is zero.

B. Input Zero Traced Radix-4 DIF FFT Pruning

In radix-4 since we couple four inputs to obtain four outputs, we therefore have 16 combinations of those four inputs at radix-4 butterfly. Now therefore for radix-4 pruning there exist five conditions only based upon zeros at the input.

- No zero at the input: No pruning takes place, butterfly calculations are same as radix-4
- Any one input zero: Output will be only the copied version of remaining inputs available, butterfly calculations are reduced compared to radix-4.

- Any two inputs are zeros: Output will be only the copied version of that remaining two inputs available, butterfly calculations are reduced compared to radix-4 pruning with one zero at input.
- Any three inputs are zeros: Output will be only the copied version of that remaining single input available, butterfly calculations are reduced compared to radix-4 pruning with two zero at input.
- All zeros input: Output is zero and is obtained from mathematical calculations is zero.

C. Input Zero Traced Radix-8 DIF FFT Pruning In radix-8 since we couple eight inputs to obtain eight outputs, we therefore have 256 combinations of those eight inputs at radix-8 butterfly. Now therefore for radix-8 pruning there exist seven conditions only based upon zeros at the input. Similarly to radix-4 pruning, output is the version of non zero input. The more the number of zeros at input leads to less mathematical calculations compared to radix-8.

D. Input Zero Traced Mixed radix DIF FFT Pruning If we consider mixed radix 4/2, it uses the combination of radix-2 pruning and radix-4 pruning. Similarly mixed radix 8/2 uses the combination of radix-2 pruning and radix8 pruning.

E. Input Zero Traced Split radix DIF FFT Pruning If we consider spilt radix 2/4, it uses the combination of radix-2 pruning and radix-4 pruning.

V RESULTS

In order to compare the computational complexities among the different radix DIF FFT algorithms on OFDM, the calculations based on the OFDM block sizes have been performed which are given in Table 1 and with pruning comparison in Table 2.

The speed improvement factors from without to with pruning of different radix algorithms are seen in Table 3.

OFDM Block Size	Radix -2		Radix-4		Radix-8		Mixed Radix-4/2		Mixed Radix-8/2		Split Radix-2/4	
	cm	cadd	cm	cadd	cm	cadd	cm	cadd	cm	cadd	cm	cadd
2	1	2	-	-	-	-	-	-	-	-	-	-
4	4	8	3	8	-		-	-	-	-	0	8
8	12	24	-	-	7	24	10	24	-	-	4	24
16	32	64	24	64	-	-	28	64	22	64	12	64
32	80	160	-	-	-	-	64	160	60	160	36	160
64	192	384	144	384	112	384	160	384	152	384	92	384

Table 2: Comparison of complex additions(cadd) and complex multiplications(cm) of different radix algorithms without pruning

OFDM Block Size	Radix -2		Radix-4		Radix-8		Mixed Radix-4/2		Mixed Radix-8/2		Split Radix-2/4	
	cm	cadd	cm	cadd	cm	cadd	cm	cadd	cm	cadd	cm	cadd
2	0	2	-	-	-	-	-	-	-	-	-	-
4	0	8	3	8			-	-	-	-	0	8
8	12	24	-	-	7	24	8	24	-	-	4	24
16	31	64	24	64	-	-	26	64	22	64	12	64
32	76	160	-	-	-	-	64	160	60	160	36	160
64	179	384	141	384	112	384	157	384	152	384	90	384

Table 2: Comparison of complex additions(cadd) and complex multiplications(cm) of different radix algorithms with pruning

FFT Size	Radix - 2	Radix -4	Radix -8	Mixed Radix -4/2	Mixed radix-8/2	Split radix -2/4
8	1	-	1	1.25	-	1
16	1.03	1	-	1.07	1	1
32	1.05	-	-	1	1	1
64	1.07	1.02	1	1.01	1	1

Table 3: Speed Improvement Factor without to with pruning in terms of Multiplications

Output shows the significant reduction of computational complexity by reducing the total no. of complex operation i.e. both the multiplications and additions compare to the ordinary radix FFT operations. The complex multiplications and additions are compared for different radix and pruned algorithms. The comparison of complex multiplications for different radix DIF FFT algorithms is shown in Figure 7 and for different input zero traced radix DIF FFT pruned algorithms are shown in Figure 8.

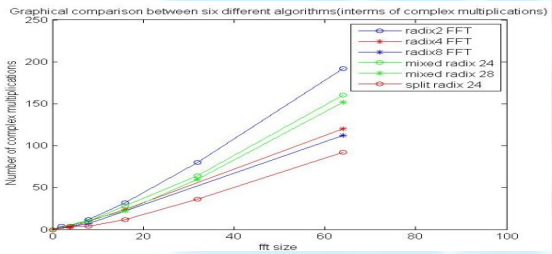


Figure 7: Comparison of complex multiplications for different radix DIF FFT

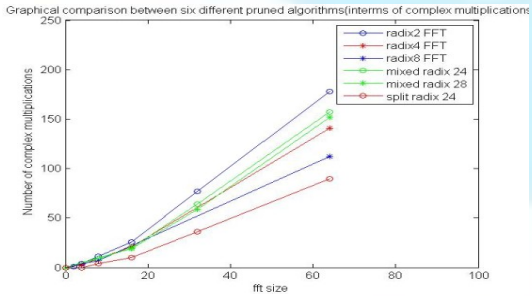


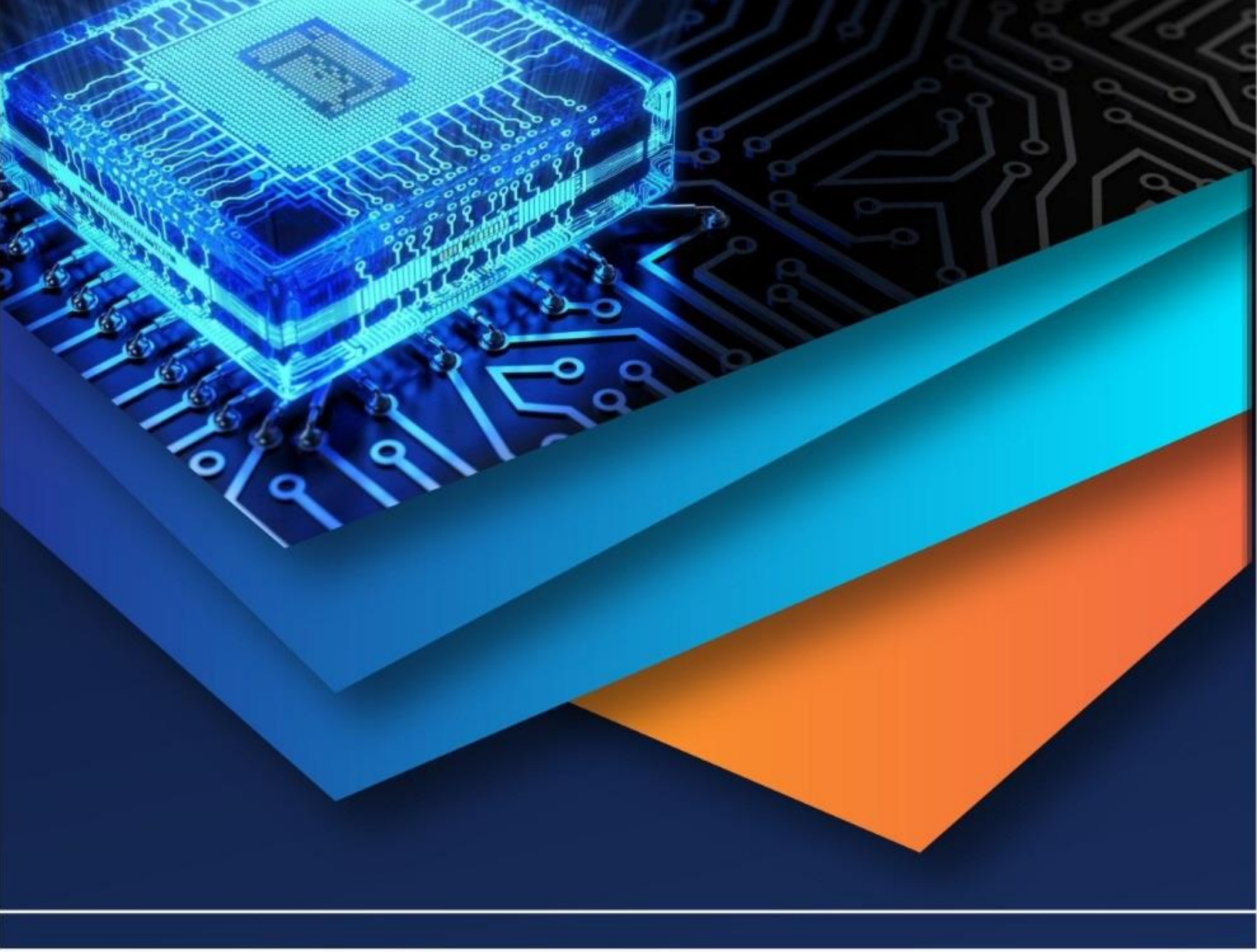
Figure 8: Comparison of complex multiplications for different Radix DIF FFT pruned algorithms

VI CONCLUSION

The computational performance of an OFDM system depends on FFT as in an OFDM system. FFT works as a modulator. If the complexity decreases, then the speed of OFDM system increases. Results shows input zero traced radix DIF FFT pruned algorithms are much efficient than the Radix DIF FFT algorithms as it takes very less time to compute where number of zero valued inputs/outputs are greater than the total number of non zero terms, with maintaining a good trade-off between time and space complexity, and it is also independent to any input data sets.

REFERENCES

[1] B. E. E. P. Lawrey, "Adaptive Techniques for Multi-User OFDM," Ph.D. Thesis, James Cook University, Townsville,2001, pp. 33-34.
[2] J. Mitola, III, "Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio," Thesis (PhD), Dept. of Teleinformatics, Royal Institute of Technology (KTH), Stockholm Sweden, May 2000.
[3] S. Chen, "Fast Fourier Transform," Lecture Note, Radio Communications Networks and Systems, 2005.
[4] "OFDM for Mobile Data Communications," The International Engineering Consortium WEB ProForum Tutorial, 2006. http://www.iec.org.
[5] Andrea Goldsmith," Wireless Communications" Cambridge university press, 2005, ISBN: 978052170416.
[6] J.G. Proakis and D.G. Manolakis, "Digital Signal Processing: Principles, Algorithms and Edition, 2002, pp. 448-475.
[7] E. Chu and A. George, *Inside the FFT Black Box :Serial & Parallel Fast FourierTransform Algorithms*. CRC Press LLC, 2000.
[8] B. G. Jo and M. H. Sunwoo, " New Continuous-Flow Mixed-Radix (CFMR) FFT Processor Using Novel In-Place Strategy," *Electron Letters*, vol. 52, No. 5, May 2005.
[9] Charles Wu, **"**Implementing the Radix-4 Decimationin Frequency (DIF) Fast Fourier Transform (FFT) Algorithm Using aTMS320C80 DSP", Digital Signal Processing Solutions,January 1998.
[10] P. Duhamel and H. Hollmann, "Split-radix FFT Algorithm," *Electron Letters*, vol. 20, pp 14-16, Jan. 1984.
[11] [4] H. V. Sorensen, M. T. Heideman and C. S. Burrus, "On Computing the Split-radixFFT," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 152-156,Feb. 1986.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY