INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

DESIGN AND IMPLEMENTATION OF SPLIT RADIX ALGORITHM FOR LENGTH - 6^M DFT USING VLSI AND FPGA

G.Naresh[1], S.Sruthi[2]

[1] Assistant Professor
Department Of E.C.E
Sree Vidyanikethan Engineering College, Tirupati, A.P
ganekantinaresh@gmail.com
[2] P.G. Student Scholar, M.Tech (VLSI),
Department of E.C.E
Sree Vidyanikethan Engineering College, Tirupati, A.P
sruthiseasapuram@gmail.com

Abstract

The Discrete Fourier Transform (DFT) is a transform used for Fourier Analysis of a sampled sequence of a signal. The Fast Fourier Transform (FFT) is a numerically efficient algorithm used to compute the DFT. In this paper we propose to design a 12-point DFT and also find its computation time. The 12-point DFT can be calculated by radix-3 and radix-6 FFT with decimation in time. It is a variant of split radix and can be flexibly implement a length of $2^r X 3^m$ DFT. Novel order permutations of Sub-DFT's and reduction of the number of arithmetic operations improve the viability of the proposed algorithm. It provides a wider choice of accessible FFT's lengths. The six point DFT and the twelve point DFT can be simulated in Modelsim using System Verilog language. The algorithm can evaluate a non-power-of-six DFT, as long as its length can be divisible by 6. In order to reduce the number of operations, all sub-DFT's are reordered satisfactorily. The proposed algorithm shows that its implementation requires fewer operations as compared with the earlier known algorithms.

Keywords: Discrete Fourier Transforms (DFT), Fast Fourier Transforms (FFT), Inverse Fast Fourier Transforms (IFFT), General Split Radix, Radix-6

1. Introduction

The Discrete Fourier Transform (DFT) is a transform used for Fourier analysis of a finite sampled sequence of a signal. The DFT is widely employed in signal processing and related fields to analyze the frequencies contained in a signal. The Fast Fourier Transform (FFT) is a numerically efficient algorithm used to compute the DFT. For a complex N-point Fourier transform, the FFT reduces the number of complex multiplications from the order of N2 to the order of N logN. Mixed radix-2/3/4/5 FFTs can be used to implement the DFT algorithm with reduced computation if the number of DFT point N = 2k 3l 4m 5n (k, l, m, and n are positive integers).

Vol.2 Issue.7, Pg.: 36-45
July 2014

It is a Fourier transform for a finite domain, discrete time periodic function, which is suitable for processing data stored in computers. Basically it converts discrete time data into discrete frequency data and vice versa. The need for this conversion is that our signals can be viewed in different domain, inside which different difficult problems become simple to analyze. The increasing application of digital equipment caused the computation of discrete Fourier transform to become an important problem.

In the past few years, a number of algorithms have been proposed for computing the discrete Fourier transform. To determine the DFT more quickly and with less complexity, Fast Fourier transform algorithms have been developed which are generally known as FFT. Most of these algorithms deal with power-of-2 sequence lengths. The first widely known achievement in this area was the radix-2 FFT. The number of arithmetic operations required for calculating the FFT is one of the important factors in evaluating any FFT algorithm. The radix-2 FFT algorithm is in the long list of practical DFT algorithms with reduced arithmetical complexity for data sizes $N=2^r$, r being an integer.

The increased usage of FFTs made us concentrate on the complexity, memory usage, and power consumption of the algorithms when used in digital signal processing applications. This lead to the improvement of FFT for different length sequences such as radix-3, radix-6, and radix-12 DFTs. These FFT algorithms are developed from radix-2 FFTs and they are found to be better than the existing algorithms. Simultaneously, the researches on the algorithms for computing length- $n=k^m$ DFT have resulted in the presentation of the methods for k=3 and k=6. Due to the poor efficiency, the algorithms for k^m are of trivial practical meanings when k=2. However, there exist many applications in which the sequence lengths are 3^m and 6^m [1]. So an algorithm for sequence length-$N=6^m$ have been developed which shows increased performance than the existing algorithms.

2. FFT algorithms

2.1 Radix-2/8 FFT Algorithm for Length $qx2^m$ DFTs

A new radix-2/ 8 fast Fourier transform (FFT) algorithm have been proposed for computing the discrete Fourier transform of an arbitrary length $N=qx2^m$, where m is an odd integer [2]. It reduces substantially the operations such as data transfer, address generation, and twiddle factor evaluation or access to the lookup table, which contribute significantly to the execution time of FFT algorithms. It is shown that the arithmetic complexity (multiplications, additions) of the proposed algorithm is, in most cases, the same as that of the existing split-radix FFT algorithm. The basic idea behind the proposed algorithm is the use of a mixture of radix-2 and radix-8 index maps. The algorithm is expressed in a simple matrix form, thereby facilitating an easy implementation of the algorithm, and allowing for an extension to the multi dimensional case. For the structural complexity, the important properties of the Cooley–Tukey approach such as the use of the butterfly scheme and in-place computation are preserved by the proposed algorithm. It is suitable only for DFT of sequence length $N=qx2^m$.

2.2 Radix 2/6 Split- Radix FFT Algorithm

A radix-2/16 decimation-in-frequency (DIF) fast Fourier transforms (FFT) algorithm and its higher radix version, namely radix-4/16 DIF FFT algorithm, have been proposed by suitably mixing the radix-2, radix-4 and radix-16 index maps, and combing some of the twiddle factors [3]. It is shown that the proposed algorithms and the existing radix-2/4 and radix-2/8 FFT algorithms require exactly the same number of arithmetic operations (multiplications, additions). By using this technique, it can be shown that all the possible split-radix FFT algorithms of the type radix- $2^r/2^{rs}$ for computing a 2^m point DFT require exactly the same number of arithmetic operations. This algorithm is suitable only for sequence of length $N=2^m$, m is integer.

2.3 New Radix-6 FFT algorithm

A new radix-6 FFT algorithm suitable for multiply-add instruction have been proposed. The new radix-6 FFT algorithm requires fewer floating-point instructions than the conventional radix-6 FFT algorithms on processors that have a multiply-add instruction. Techniques to obtain an algorithm for computing radix-6 FFT with fewer floating-point instructions than conventional radix-6 FFT algorithms have been proposed [3]. The

Vol.2 Issue.7, Pg.: 36-45
July 2014

number of floating-point instructions for the new radix-6 FFT algorithm is compared with those of conventional radix-6 FFT algorithms on processors with multiply-add instruction.

2.4 The 2-Point Radix 3/6 Split Radix Algarithm

2.4.1 The Radix-3 and Radix-6 FFT Approach

The proposed Radix 3/6 algorithm is based on mixture of Radix-3 and Radix-6 FFT algorithms. The definition of DFT is given by,

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N}$$

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

$$W_N^{nk} = e^{-j2\pi nk/N} = \cos\left(\frac{2\pi nk}{N}\right) - j\sin\left(\frac{2\pi nK}{N}\right) \quad (2)$$

In (1) and (2) N is the number of data, j= √-1 and is the twiddle factor. W_N^{nk} is called the N-point DFT of the sequence of x(n). For each value of k, the value of X (k) represents the Fourier transform at the frequency.

The Radix-3 DIT-FFT can be derived as,

$$X(k) = \sum_{n=0}^{N-1} x_n W_N^{nk}$$

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

$$= \sum_{n=0}^{\frac{N}{2}-1} x(3n) W_{\frac{N}{3}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x(3n+1) W_{\frac{N}{3}}^{nk} + W_N^{2k} \sum_{n=0}^{\frac{N}{3}-1} x(3n+2) W_N^{nk}$$

$$= P(k) + W_N^k Q(k) + W_N^{2k} R(k) \quad (3)$$

Each of the sums, P(k), Q(k), and R(k), in (3) is recognized as an N/3-point DFT. The transform *X(k)* can be broken into three parts as shown in (4).

$$X(k) = P(k) + W_N^k Q(k) + W_N^{2k} R(k)$$

$$X(k + \frac{2N}{3}) = P(k) + W_N^{k+\frac{2N}{3}} Q(k) + W_N^{2(k+\frac{2N}{3})} R(k) = P(k) + W^{2/3} W_N^k Q(k) + W^{1/3} W_N^{2k} R(k) \quad (4)$$

$$k = 0,1,2.....\frac{N}{3} - 1$$

In (4), the periodicity property $W_N^{k+N} = W_N^k$ is used to simplify $W_3^4 = W_3^1$ .The complex numbers W_3^1 *and* W_3^2 can be expressed as shown in (5)

$$W_3^1 = e^{-j2\pi/3} = -\frac{1}{2} - \frac{\sqrt{3}}{2} j$$

$$W_3^2 = e^{-j2\pi\times 2/3} = -\frac{1}{2} + \frac{\sqrt{3}}{2} j \qquad (5)$$

The Radix-6 DIT-FFT can be derived as,

$$X(k) = \sum_{n=0}^{N-1} x_n W_N^{nk}$$

$$= \sum_{n=0}^{\frac{N}{6}-1} x(6n) W_{N/6}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{6}-1} x(6n+1) W_{N/6}^{nk} + W_N^{4k} \sum_{n=0}^{\frac{N}{6}-1} X(6n+4) W_{N/6}^{nk} +$$

$$W_N^{2k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+2) W_{\frac{N}{6}}^{nk} + W_N^{3k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+3) W_{N/6}^{nk} + W_N^{5k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+5) W_N^{6nk} + W_N^{6k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+6) W_{N/6}^{nk}$$

$$= P(k) + W_N^k Q(k) + W_N^{2k} R(k) + W_N^{3k} S(k) + W_N^{4k} T(k) + W_N^{5k} U(k) + W_N^{6k} V(k) \qquad (6)$$

2.4.2 Split Radix 3/6 FFT Approach

The Algorithm decomposes a DFT of size-N=6m into one length-N/3 and four length-N/6 sub DFTs. The flexibility of the decomposition enables the algorithm to be competent at the implementation of a non-power-of-six DFT, while its length can exactly divided by 6. Appropriate permutations are used for sub DFT input sequences to reduce the computational capacity.

The definition of DFT is

$$X_k \sum_{n=0}^{N-1} x_n w_N^{nk} \qquad (7)$$

Where $W_N = e^{-j2\pi/N}$, $j = \sqrt{-1}$, the length N of sequence *x(n)* is assumed as an integer, which is divisibly by six. For lengths N of DFT, powers-of-six would be best for the proposed algorithm. Obviously, the DFT can be divided into three length N/3 sub-DFTs. In order to derive a best possible algorithm, we continue to decompose the three sub-DFTs. Due to no scaling factor in front of it, the first sub-DFT should be let as it is and directly go into the recursive decomposition of the next stage. The other two sub DFTs are divided into four sub-DFTs of length-N/6. Actually, if the length of a DFT can be divided by 6, the DFT can be decomposed by the algorithm. The generalized length-N can be assumed as $N=2^r x3^m$, where $r \geq m-1$. The decomposition of a DFT of size $N=2^r x3^m$ is denoted by,

$$X(k) = \sum_{n=0}^{\frac{N}{3}-1} x_{3n} W_{N/3}^{nk} + W_{2^r}^k w_{3^m}^k \sum_{n=0}^{\frac{N}{6}-1} x_{6n+2^r+3^m} W_{N/6}^{nk} +$$

$$w_3^K m \sum_{n=0}^{\frac{N}{6}-1} x_{6n+2^r} W_{N/6}^{nk} + w_{3^m}^{-k} \sum_{n=0}^{\frac{N}{6}-1} x_{6n-2^r} W_{N/6}^{nk} + w_{2^r}^{-k} w_{3^m}^{-k} \sum_{n=0}^{\frac{N}{6}-1} x_{6n-2^r-3^m} W_{N/6}^{nk} \qquad (8)$$

Where the four length- N/6 sub DFTs are reordered. To simplify the description, (8) can be expressed by,

Vol.2 Issue.7, Pg.: 36-45
July 2014

$$X(k) = A_k + w_{2^r}^k w_{3^m}^k B_k + C_k + w_{3^m}^{-k} E_k + w_{2^r}^{-k} w_{3^m}^{-k} F_k \quad (9)$$

Where,

$$A_k = \sum_{n=0}^{\frac{N}{3}-1} x_{3n} W_{N/3}^{nk}$$

$$B_k = \sum_{n=0}^{\frac{N}{6}-1} x_{6n+2^r+3^m} W_{N/6}^{nk}$$

$$C_k = \sum_{n=0}^{\frac{N}{6}-1} x_{6n+2^r} W_{N/6}^{nk}$$

$$E_k = \sum_{n=0}^{\frac{N}{6}-1} x_{6n-2^r} W_{N/6}^{nk}$$

$$F_k = \sum_{n=0}^{\frac{N}{6}-1} x_{6n-2^r-3^m} W_{N/6}^{nk} \quad (10)$$

In (9), $W_{2^r}^k w_{3^m}^k B_k$ and $w_{2^r}^{-k} w_{3^m}^{-k} F_k$ can be treated in pairs, since $w_{2^r}^{-k} w_{3^m}^{-k} F_k$ and $w_{2^r}^{-k} w_{3^m}^{-k} F_k$ is a conjugate-pair. In the similar way, and can be handled, with in pairs. The direct implementation of (9) performs many unnecessary operations, since the computations of $X_k, X_{\frac{N}{6}+k}, X_{\frac{2N}{6}+k}, X_{\frac{3N}{6}+k},$ $X_{\frac{4N}{6}+k,}$ $X_{\frac{5N}{6}+k}$ turn out to share many calculations each other. In particular, if we add to, the size-N/6 to *k* DFT are not changed (because they are periodic in k), while the size-N/3 DFT is unchanged if we add to *2N/6* to *k*. So, the only things that change are $W_2^k W_3^k m, W_{2r'}^{-k} W_{3m}^{-k}$ and W_{3m}^{-k} terms. In order to reduce the number of the operations, the following six identities are necessary,

$$X_k = A_k + (w_{2^r}^k w_{3^m}^k B_k + w_{2^r}^{-k} w_{3^m}^{-k} F_k)$$

$$+(w_{3^m}^k C_k + w_{3^m}^{2^r} w_{2^r}^k w_{3^m}^k B_k + w_3^{-2^r} w_{2^r}^{-k} F_k) \quad (11)$$

$$X_{k+\frac{2N}{6}} = A_k + (w_3^{2^r} w_{2^r}^k w_{3^m}^k B_k + w_3^{-2^r} w_{2^r}^{-k} w_{3^m}^{-k} F_k) + (w_3^{2^r} w_{3^m}^k C_k + w_3^{-2^r} w_{3^m}^{-k} E_k) \quad (12)$$

$$X_{k+\frac{4N}{6}} = A_k + (w_3^{2^r+1} w_{2^r}^k w_{3^m}^k B_k + w_3^{-2^r+1} w_{2^r}^{-k} w_{3^m}^{-k} F_k) + (w_3^{2^{r+1}} w_{3^m}^k C_k + w_3^{-2^{r+1}} w_{3^m}^{-k} E_k) \quad (13)$$

$$X_{k+\frac{N}{6}} = A_{k+N/6} - (w_3^{2^{r-1}} w_{2^r}^k w_{3^m}^k B_k + w_3^{-2^{r-1}} w_{2^r}^{-k} w_{3^m}^{-k} F_k) + (w_3^{2^{r-1}} w_{3^m}^k C_k + w_3^{-2^{r-1}} w_{3^m}^{-k} E_k) \quad (14)$$

$$X_{k+\frac{3N}{6}} = A_{k+N/6} - (w_{2^r}^k w_{3^m}^k B_k + w_3^{-2^r} w_{2^r}^{-k} w_{3^m}^{-k} F_k) + (w_{3^m}^k C_k + w_{3^m}^{-k} E_k) \quad (15)$$

Vol.2 Issue.7, Pg.: 36-45
July 2014

$$X_{k+\frac{5N}{6}} = A_{k+N/6} - (w_3^{2^r} w_{2^r}^k w_{3^m}^k B_k + w_3^{-2^r} w_{2^r}^{-k} w_{3^m}^{-k} F_k) + (w_3^{2^r} w_{3^m}^k C_k + w_3^{-2} w_{3^m}^{-k} E_k) \qquad (16)$$

A complete output set $\{X_k\}$ can be obtained if we let range from 0 to N/6 1in the above six equations. We now summarize the scheme of the proposed radix-3/6 FFT algorithm. The initial input sequence of length- is decomposed into five sub-sequences. This process is repeated successively for each of new sub-sequences, until the sizes of all sub DFTs are indivisible by 6. Figs 2, 3, 4 illustrate the flow graph of 3, 6 and 12 point radix 3/6 algorithm (2-points and 4-points FFT can be performed with SRFFT).

2.4.3 Performance Analysis of the Algorithm

In this section, we consider the performance of the proposed algorithm by analysing the computational complexity and comparing it with existing algorithms. Let M_N and A_N be, respectively the number of multiplications and additions. We assume that a 3-point DFT requires 4 real multiplications and 12 real additions (some algorithm assumes that a 3-point DFT is calculated with 2 real multiplication and 12 real additions, since one need not multiply ½ and the multiplication by 1/2 can be evaluated with bit shift).

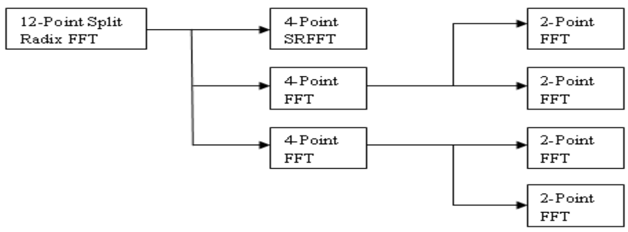

Figure 1: Block diagram of split radix 3/6FFT

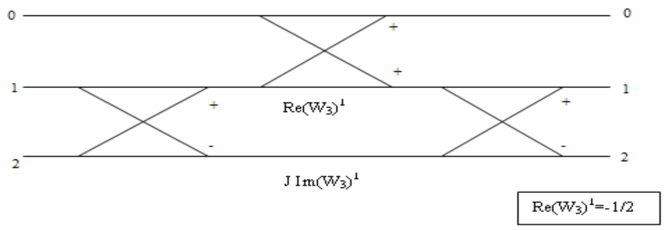

Figure 2: Flow graph of 3-point FFT

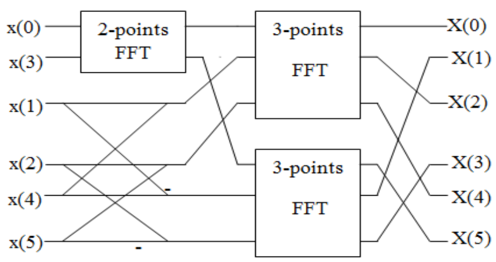

Figure 3: Flow graph of 6-point 3/6 FFT

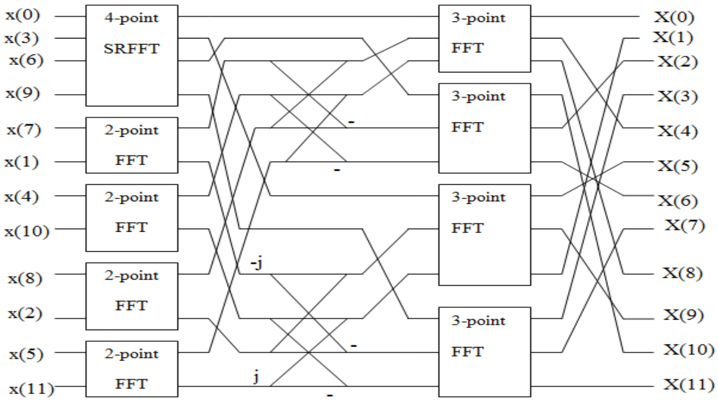

Figure 4: Flow graph of 12-point 3/6 FFT

The decomposition in the proposed algorithm is conducted recursively until the lengths of all sub DFTs cannot be exactly divided by 6. In general, there are only 1 the first special butterfly (if r≥1and m≥1), 1 the second special case butterfly (if r≥2and m≥1), 1 the third special case butterfly and 1 the fourth special case butterfly (if r≥3and m≥1). The total number of the fifth and sixth type of butterflies is 2^{r-1} -4. Thus, the arithmetic complexity of the proposed algorithm can be given as follows,

$$M_N = \begin{cases} \dfrac{M_{N/3} + 4M_{N/6} + 8N}{3-8} r = 1, m \geq 1 \\ \dfrac{M_{N/3} + 4M_{N/6} + 8N}{3-16} r = 2, m \geq 1 \\ \dfrac{M_{N/3} + 4M_{N/6} + 8N}{3-24} r \geq 3, m \geq 1 \end{cases} \tag{19}$$

$$A_N = \begin{cases} \dfrac{A_{N/3} + 4A_{N/6} + 20N}{3-8} r = 1, m \geq 1 \\ \dfrac{A_{N/3} + 4A_{N/6} + 20N}{3-16} r = 2, m \geq 1 \\ \dfrac{A_{N/3} + 4A_{N/6} + 20N}{3-2^{r+1}-8} r \geq 3, m \geq 1 \end{cases} \tag{20}$$

3. IFFT ALGORITHMS

3.1 Performance Analysis of the Algorithm

Implementation of Radix 3/6 IFFT Design and verification using system verilog will be done. Due to being an irregular integer for the sequence lengths; it is difficult to gain a completely accurate formula of computational complexity. The IFFT can be performed by first swapping the real and imaginary parts of the incoming data and then performing the forward FFT on them and once again swapping the real and imaginary parts of the data at the output. This methods allows one to perform the IFFT without changing any internal coefficients and thus, resulting in more efficient hardware implementation.

Vol.2 Issue.7, Pg.: 36-45
July 2014

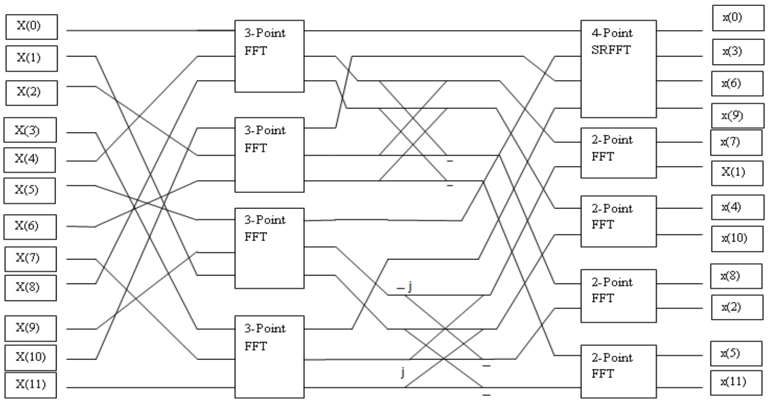

Figure 5: 12-Point Split Radix 3/6 IFFT

4. Simulation Results

The 12 point DFT sequence has been implemented in VLSI and simulated using modelsim based on radix 3/6 FFT algorithm. Fig. 6, 7 shows the simulation results of 12 point DFT sequence. Fig.8, 9 shows the simulation results of 12-Point Split Radix 3/6 IFFT.

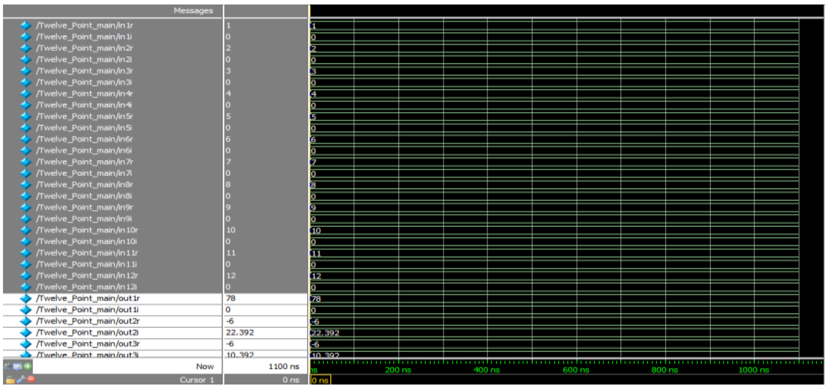

Figure 6: Input to 12-Point DFT

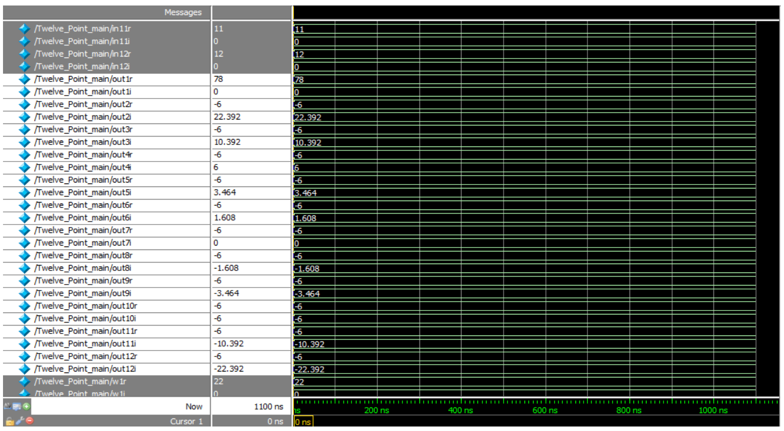

Figure 7: Output of 12-Point DFT

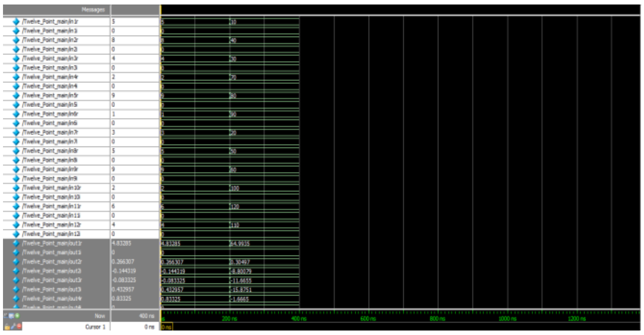

Figure 8: Input to 12-Point IFFT

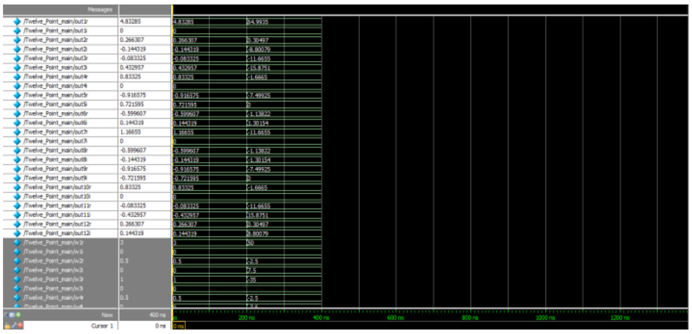

Figure 9: Output of 12-Point IFFT.

5. Conclusion

In this paper, a radix 3/6 FFT algorithm is presented for length-6^m DFT. The proposed algorithm is a mixture of radix-3 and radix-6 algorithm. It can evaluate a non-power-of-six DFT, as long as its length- can be divided by 6. In order to reduce the number of operations, all sub DFTs are reordered favourably. The proposed algorithm shows that its implementation requires less real operations as compared with the published algorithms. Computational complexity is approximately 4Nlog2N-6N+8. Due to being an irregular integer for the sequence lengths; it is difficult to gain a completely accurate formula of computational complexity. The device utilization summary shows that the area occupied by the algorithm is very low. The proposed algorithm can be used in OFDM systems, Signal and Image Processing, Discrete time Signal processing, and DCT applications. In the future, since the proposed algorithm shows advantages for computing length-N=2q X 3m DFT, we will do some works in this way. The algorithms presented in can be improved by scaled DFT, and the comparison of its computational complexity with the proposed algorithm needs to make.

References

[1] Weihua Zheng and Kenli *, "Split Radix Algorithm for Length 6m DFT",* IEEE signal processing Letters, *VOL. 20, NO. 7, July 2013*.

[2] Wen-Chang Yeh and Chein-Wei Jen*,"High-Speed and Low-Power Split-Radix FFT",* IEEE Transactions on Signal Processing, *Vol. 51, No. 3, March 2003.*

[3] Saad Bouguezel, M. Omair Ahmad, and M.N.S. Swamy, **"***An Efficient Split-Radix FFT Algorithm"*–conference proceedings of International symposium on circuits and systems ISCAS'*03,Vol.4,2003*

[4] M.Frigo and S.Johnson, "The design and implementation of ff3," Proc. IEEE, vol. 93, o. 2, pp. 216-231, 2005.

[5] J.Keiner S. Kunis, and D. Potts, "Using nfft3-A software library for various non equispaced fast Fourier transforms," ACM Trans. Math. Softw. (TOMS), vol.36, no.4, pp. 19-19, 2005.

[6] S. Bouguezel, M. Ahmad, and M. Swamy, "A new radix-2/8 FFT algorithm for length-DFTs," IEEE Trans. Circuits Syst. I, vol.51, no. 9, pp. 1723–1732, Sep. 2004.

[7] I. Kamar and Y. Elcherif, "Conjugate pair fast Fourier transform," Electron. Lett., vol. 25, no.5, pp. 324–325, Apr. 1989.

[8] Y. Suzuki, T. Sone, and K. Kido, "A new FFT algorithm of radix 3, 6, and 12," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-34,no. 2, pp. 380–383, Apr. 1986.
[9] S. Prakash and V. Rao, "A new radix-6 FFT algorithm," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-29, no. 4, pp. 939–941,Aug. 1981.
[10] E. Dubois and A. Venetsanopoulos, "A new algorithm for the radix-3 FFT," IEEE Trans. Acoust., Speech, Signal Process,, vol. ASSP-26, pp.222–225, Jun. 1978.

AUTHORS PROFILE

G. NARESH Working as Assistant Professor in Sree Vidyanikethan Engineering College, Tirupati. He completed M.Tech from Sathyabhama University, Chennai and Completed B.Tech from SVCET, Chittoor. His areas of interest are Low Power VLSI and FPGA. **Email:ganekantinaresh@gmail.com**

S. SRUTHI Completed her B.Tech in Electrical and Electronics Engineering from Annamacharya Institute of Technology and Sciences, Tirupati. Her interests are Low Power VLSI and Digital System Design. Presently, she is pursuing M.tech in Sree Vidyanikethan Engineering College, Tirupati, Andhra Pradesh.

Email:sruthiseasapuram@gmail.com