ISSN:2321-1156

International Journal of Innovative Research in Technology & Science(IJIRTS)

Implementation of Split Radix algorithm for length 6^m DFT using VLSI

J.Nancy, PG Scholar,PSNA College of Engineering and Technology; S.Bharath,Assistant Professor,PSNA College of Engineering and Technology;J.Wilson,Assistant Professor,MAR College of Engineering and Technology

Abstract

This paper uses structured design to implement and simulate radix 12-point DFT by vhdl language. 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 flexibly implement a length of 2^r x 3^m DFT. Novel order transformation of sub-DFTs and reduction of the number of real addition and multiplication operations improve the viability.The six point DFT can be simulated in modelsim using system verilog language and the 12 point DFT can be simulated in vhdl. 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 DFTs are reordered satisfactorily. The proposed algorithm shows that its implementation requires less real operations as compared with the published algorithms. The pending update to system Verilog contains several new packages and functions. The new packages include support for both fixed-point and floating-point binary math. These fully Non-synthesizable packages will raise the level of abstraction in System Verilog. DSP applications, which previously needed an independent processor core, or required very difficult manual translation, can now be performed within your system verilog source code. In addition, Schematic-based DSP algorithms can now be translated directly to System Verilog.

Index Terms - Discrete Fourier transform (DFT), Fast Fourier Transform (FFT), general Split Radix, radix 6, System Verilog language

.

Introduction

Discrete Fourier Transform (DFT) plays a very important role in digital signal processing. 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, 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 $k \neq 2$. However, there exists 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.

Literature Survey

A.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 multidimensional 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 $N=qx2^m$.

B.Radix 2/16 Split-Radix FFT Algorithms

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^mpoint DFT require exactly the same number of arithmetic operations. This algorithm is suitable only for sequence of length $N=2^m$, m is integer.

C.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 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.

The 12-pointRadix 3/6 Split Radix Algorithm

A.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,

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

$$W_N^{nk} = e^{-j2\pi nk/N} = \cos\left(\frac{2\pi nk}{N}\right) - jsin\left(\frac{2\pi nk}{N}\right) \qquad (2)$$

In (1)and(2) N is the number of data, j= √ -1 and W_N^{nk} is the twiddle factor. (1) 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 $\frac{2\pi k}{N}$.

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

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

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

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

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

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

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).

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

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

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

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

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)

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

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

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

$= \sum_{n=0}^{\frac{N}{6}-1} x(6n) W_N^{6nk} + \sum_{n=0}^{\frac{N}{6}-1} x(6n+1) W_N^{(6n+1)k} + \sum_{n=0}^{\frac{N}{6}-1} x(6n+2) W_N^{(6n+2)k} + \sum_{n=0}^{\frac{N}{6}-1} x(6n+3) W_N^{(6n+3)k} + \sum_{n=0}^{\frac{N}{6}-1} x(6n+4) W_N^{(6n+4)k} + \sum_{n=0}^{\frac{N}{6}-1} x(6n+5) W_N^{(6n+5)k} + \sum_{n=0}^{\frac{N}{6}-1} x(6n+6) W_N^{(6n+6)k}$

$= \sum_{n=0}^{\frac{N}{6}-1} x(6n) W_N^{6nk} + W_N^k \sum_{n=0}^{\frac{N}{6}-1} x(6n+1) W_N^{6nk} + W_N^{2k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+2) W_N^{6nk} + W_N^{3k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+3) W_N^{6nk} + W_N^{4k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+4) W_N^{6nk} + W_N^{5k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+5) W_N^{6nk} + W_N^{6k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+6) W_N^{6nk}$

$= \sum_{n=0}^{\frac{N}{6}-1} x(6n) W_{N/6}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{6}-1} x(6n+1) W_{N/6}^{nk} + W_N^{2k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+2) W_{\frac{N}{6}}^{nk} + W_N^{3k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+3) W_{N/6}^{nk} + W_N^{4k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+4) W_{N/6}^{nk} + W_N^{5k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+5) W_{N/6}^{nk} + W_N^{6k} \sum_{n=0}^{\frac{N}{6}-1} x(6n+6) W_{N/6}^{nk}$

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

Each of the sums, P(k), Q(k), and R(k), in (6) is recognized as an N/6-point DFT. The transform *X(k)* can be broken into three parts as shown in (7).

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

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

$X\left(k+\frac{2N}{6}\right) = P(k) + W^{\frac{1}{6}} W_N^k Q(k) + W^{\frac{2}{6}} W_N^{2k} R(k) + W^{3/6} W_N^{3k} S(k + W^{4/6} W_N^{4k} T(k) + W^{5/6} W_N^{5k} U(k) + W_N^{6k} V(k)$
(7)

$$k = 0,1,2 \ldots\ldots \frac{N}{6} - 1$$

The complex numbers $W_6^1, W_6^2, W_6^3, W_6^4, W_6^5$ can be expressed as shown in (8)

$$W_6^1 = e^{-j2\pi/6} = -j$$
$$W_6^2 = e^{-j2\pi\times 2/6} = -\frac{1}{6}$$
$$W_6^2 = e^{-j2\pi\times 3/6} = -1$$
$$W_6^2 = e^{-j2\pi\times 4/6} = -\frac{1}{6}$$
$$W_6^2 = e^{-j2\pi\times 5/6} = -\frac{1}{6}$$
(8)

B. 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 intension.

The definition of DFT is

$$X_k = \sum_{n=0}^{N-1} x_n w_N^{nk} \qquad (9)$$

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 $N=2^r x 3^m$, where $r \geq m-1$. The decomposition of a DFT of size $N=2^r x 3^m$ is denoted by,

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

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

Where the four length- N/6 sub DFTs are reordered. To simplify the description, (10) can be expressed by,

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

Where,

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

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

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

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

$$F_k = \sum_{n=0}^{\frac{N}{6}-1} x_{6n-2^r-3^m}W_{N/6}^{nk} \quad (12)$$

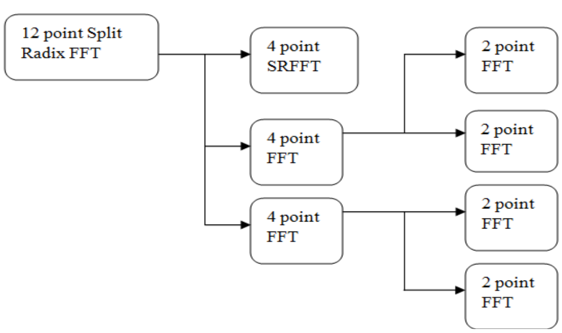


Fig 1. Block diagram of Split Radix 3/6 FFT

In (11), $w_{2^r}^{k}w_{3^m}^{k}B_k$ 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,$w_{3^m}^{k}$ and $w_{3^m}^{-k}$can be handled with in pairs. The direct implementation of (11) 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 changes are the $w_{2^r}^{k}w_{3^m}^{k}$, $w_{2^r}^{-k}w_{3^m}^{-k}$ and $w_{3^m}^{-k}$ terms. In order to reduce the number of the operations, the following six identities are necessary,

$$X_k = A_k + (w_{2^r}^{k}w_{3^m}^{k}B_k + w_{2^r}^{-k}w_{3^m}^{-k}F_k) + (w_{3^m}^{k}C_k + w_{3^m}^{-k}E_k) \quad (13)$$

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

$$X_{k+\frac{4N}{6}} = A_k + (w_3^{2^{r+1}}w_{2^r}^{k}w_{3^m}^{k}B_k + w_3^{-2^{r+1}}w_{2^r}^{-k}w_{3^m}^{-k}F_k) + (w_3^{2^{r+1}}w_{3^m}^{k}C_k + w_3^{-2^{r+1}}w_{3^m}^{-k}E_k) \quad (15)$$

$$X_{k+\frac{N}{6}} = A_{k+N/6} - (w_3^{2^{r-1}}w_{2^r}^{k}w_{3^m}^{k}B_k + w_3^{-2^{r-1}}w_{2^r}^{-k}w_{3^m}^{-k}F_k) + (w_3^{2^{r-1}}w_{3^m}^{k}C_k + w_3^{-2^{r-1}}w_{3^m}^{-k}E_k) \quad (16)$$

$$X_{k+\frac{3N}{6}} = A_{k+N/6} - (w_{2^r}^{k}w_{3^m}^{k}B_k + w_{2^r}^{-k}w_{3^m}^{-k}F_k) + (w_{3^m}^{k}C_k + w_{3^m}^{-k}E_k) \quad (17)$$

$$X_{k+\frac{5N}{6}} = A_{k+N/6} - (w_3^{2^r}w_{2^r}^{k}w_{3^m}^{k}B_k + w_3^{-2^r}w_{2^r}^{-k}w_{3^m}^{-k}F_k) + (w_3^{2^r}w_{3^m}^{k}C_k + w_3^{-2^r}w_{3^m}^{-k}E_k) \quad (18)$$

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).

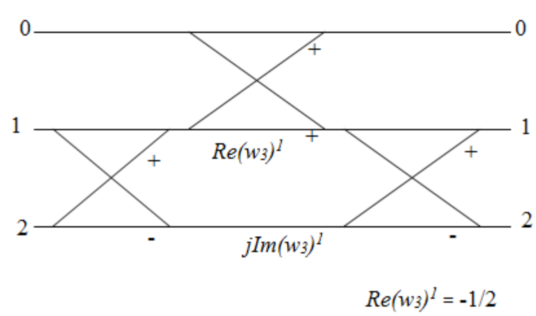


Fig 2. Flow graph of 3-point FFT

C.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).

The general butterfly of the proposed algorithm requires 16 real multiplications and 40 real additions. In general butterfly we evaluate (13) with 8 real multiplication and 16 real additions. Because $w_{2^m}^{k} w_{3^m}^{k} = w_{2^m}^{-k*} w_{3^m}^{-k*}$ and $w_{3^m}^{2k} = w_{3^m}^{-2k*}$ We calculate (14) with 8 real multiplications and 8 real additions because we share real additions with which have been undertaken in evaluating (13). We evaluate (13) with only 4 real additions, because 1+u+u*=0. Furthermore, we perform (16)–(18) at cost of 12 real additions, because all multiplications and some additions have been calculated in (13)–(15).

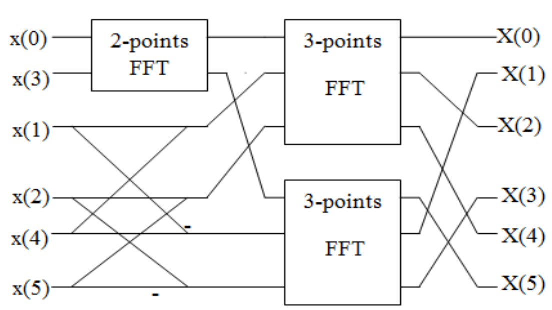


Fig: 3 Flow graph of 6-point 3/6 FFT

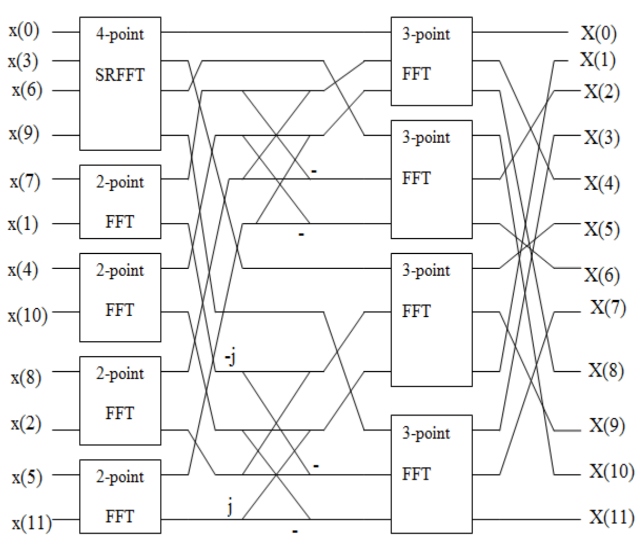


Fig 4. Flow graph of 12-point 3/6 FFT

There are six special cases. The first special case, when k=0 requires 8 real multiplications and 32 real additions. In this case, (13) is evaluated with 8 real additions (one need not multiply 1), (14) is implemented with 4 real multiplications and 6 real additions because we use real additions which have been undertaken in evaluating above calculation, (15) can be calculated with only 2 real additions, because we need not add the duplicate portion between u and u*. In the same way, (16)–(18) can be performed by only 4 real multiplications and 16 real additions. This special butterfly is illustrated in Fig. 3. The second special case, when $k=2^{r-2} \times 3^{m-1}$, requires the number of operations equals that of the first case. In this case, all rotator factors of sub DFTs in (15) can be omitted, so it can be evaluated with 8 real additions, (13)can be implemented with 4 real multiplications and 6 real additions,(15) can be calculated with only 2 real additions. Similarly,(16)–(18) can be performed by only 4 real multiplications and 16real additions.

The third special case is when $k=2^{r-3} \times 3^{m}$. This butterfly requires 12 real multiplications and 36 real additions. In this case, (13) requires extra 4 real multiplication and 4 real additions over the first case. The computations of the rest ones are similar with that of the first case. The fourth special case is when $k=2^{r-3} \times 3^{m-1}$. This butterfly requires also 12 real multiplications and 36 real additions. In this case, (15) requires extra 4 real multiplication and 4 real additions over that of (15) in the second case. The computations of the rest equations are similar with that of the second case. The fifth special case is when k mod 3^m and k mod $2^{r-3} \neq 0$. This butterfly requires 16 real multiplications and 36 real additions. In this case, (13) requires extra 8 real multiplication and 4 real additions over the first case. The sixth special case is when, k mod $3^{m-1}=0$, k mod $3^m \neq 0$ and k mod 2^{r-3}. This butterfly requires 16 real multiplications and 36 real additions. In this case, (15) requires extra 8 real multiplication and 4 real additions over the second case.

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. In additions, there are 2^{r-1} $(3^{m-1}- 1)$ general butterfly. Thus, the arithmetic complexity of the proposed algorithm can be given as follows,

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

(19)

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

(20)

Simulation Results

The 12 point DFT sequence has been implemented in VLSI and simulated using modelsim based on radix 3/6 FFT algorithm. The output is checked using the 12 point radix 3/6 flow graph theoretically and it matches with the simulated results. Fig 5, 6, 7 shows the simulation results of 12 point DFT sequence. Fig 8 shows the device utilization summary of 12 point DFT sequence in Xilinx XSE.

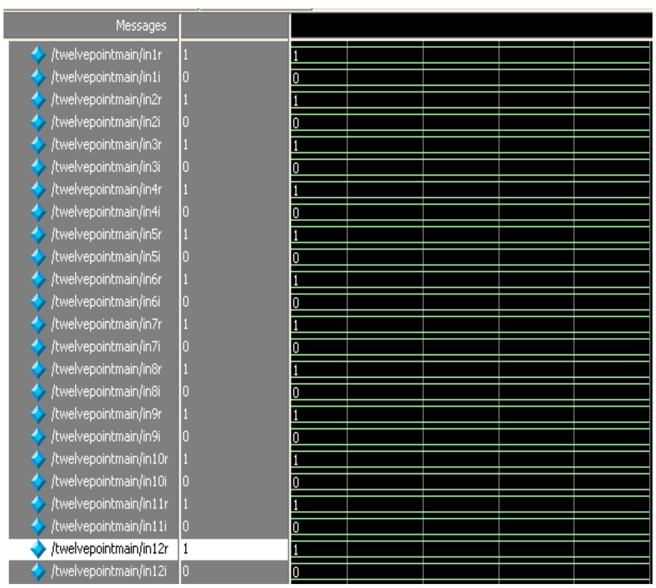


Fig 5.Simulation screenshot 1

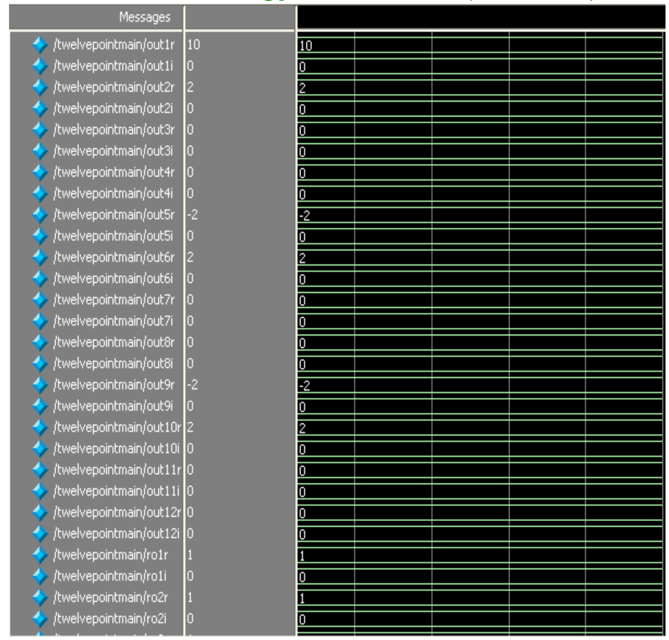


Fig 6.Simulation screenshot 2

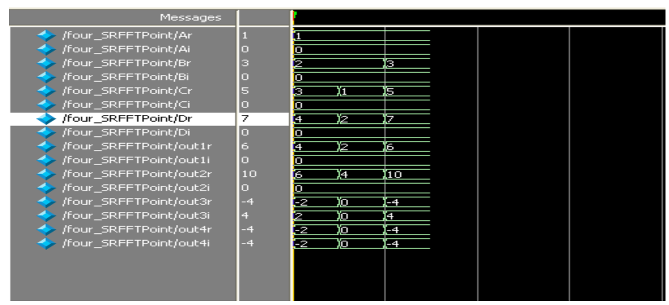
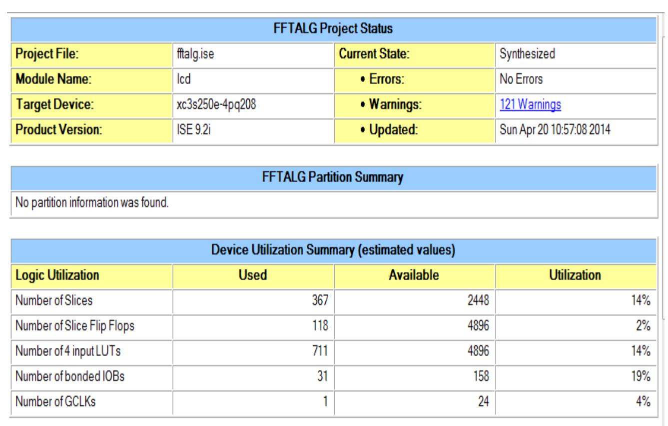


Fig 7.Simulation screenshot 3



FFTALG Project Status			
Project File:	fftalg.ise	Current State:	Synthesized
Module Name:	lcd	• Errors:	No Errors
Target Device:	xc3s250e-4pq208	• Warnings:	121 Warnings
Product Version:	ISE 9.2i	• Updated:	Sun Apr 20 10:57:08 2014

FFTALG Partition Summary
No partition information was found.

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	367	2448	14%
Number of Slice Flip Flops	118	4896	2%
Number of 4 input LUTs	711	4896	14%
Number of bonded IOBs	31	158	19%
Number of GCLKs	1	24	4%

Fig 7.Simulation screenshot 4

Conclusion

A radix 3/6 FFT algorithm is presented for length-6m 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 favorably. 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.

References

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

[2] Saad Bouguezel, M. Omair Ahmad and M.N.S. Swamy, "A New Radix-2/8 FFT Algorithm for Length-q x 2^m DFTs", IEEE Transactions on circuits and systems-I: Regular papers, VOL. 51, NO. 9, September-2004.

[3] Saad Bouguezel, M.Omair Ahmad, M.N.S. Swamy, "Arithmetic Complexity of the Split-Radix FFT Algorithms" by - IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05).

[4] Daisuke Takahashi, "A new radix-6 FFT algorithm suitable for multiply and add instruction",IEEE International Conference on Acoustics, Speech, and Signal Processing, 2000.

[5] G. Bi and Y. Chen, "Fast DFT algorithms for length $N=qx2^m$" *IEEE* Trans. Circuits Syst. II: Analog Dig. Signal Process. IEEE Transactions, vol. 45, no. 6, pp. 685–690, 1998.

[6] 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.

[7] 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

[8] Daisuke Takahashi, "An Extended Split-Radix FFT Algorithm" IEEE Signal Processing Letters, Vol. 8, NO. 5, May 2001.

[9] Ryszard Stasilisk , "Radix-K FFT's Using K-Point Convolutions" IEEE Transactions on Signal Processing, Vol. 42, No. 4, April 1994.

[10] S. Prakash and V. Rao, "A new radix-6 FFT algorithm," IEEE Trans.Acoust., Speech, Signal Pro-cess., vol. ASSP-29, no. 4, pp. 939–941,Aug. 1981.