# DISCRETE FOURIER TRANSFORM

Discrete Signals and Systems- A Review – Introduction to DFT – Properties of DFT – Circular Convolution - Filtering methods based on DFT – FFT Algorithms –Decimation in time Algorithms, Decimation in frequency Algorithms – Use of FFT in Linear Filtering.

## 2.1 Introduction to DFT:

DFT is an important mathematical tool which can be used for the software implementation of certain digital signal processing algorithms.DFT gives a method to transform a given sequence to frequency domain and to represent the spectrum of the sequence using only k frequency values, where k is an integer that takes N values, k=0, 1, 2,…..N-1.

The advantages of DFT are:
1. It is computationally convenient.

2. The DFT of a finite length sequence makes the frequency domain analysis much simpler than continuous Fourier transform technique.

The (DFT discrete Fourier transform) is one of the specific forms of Fourier analysis. As such, it transforms one function into another, which is called the frequency domain representation, or simply the DFT, of the original function (which is often a function in the time domain). But the DFT requires an input function that is discrete and whose non-zero values have a limited (finite) duration. Such inputs are often created by sampling a continuous function, like a person's voice. And unlike the discrete-time Fourier transform (DTFT), it only evaluates enough frequency components to reconstruct the finite segment that was analyzed. Its inverse transform cannot reproduce the entire time domain, unless the input happens to be periodic (forever).
The Fourier transform of a discrete signal x(t) may be defined as

$$X(K)=\sum_{n=0}^{N-1} x(n)e^{\frac{-j2\pi kn}{N}} \quad ,K=0,1,2\ldots N-1$$

Since the input function is a finite sequence of real or complex numbers, the DFT is ideal for processing information stored in computers. In particular, the DFT is widely employed in signal processing and related fields to analyze the frequencies contained in a sampled signal, to solve partial differential equations, and to perform other operations such as convolutions. The DFT can be computed efficiently in practice using a fast Fourier transform (FFT) algorithm.

Since FFT algorithms are so commonly employed to compute the DFT, the two terms are often used interchangeably in colloquial settings, although there is a clear distinction: "DFT" refers to a mathematical transformation, regardless of how it is computed, while "FFT" refers to any one of several efficient algorithms for the DFT. The sequence of *N* complex numbers $x_0$, ..., $x_{N-1}$ is transformed into the sequence of $N$ complex numbers $X_0$, ..., $X_{N-1}$ by the DFT according to the formula:

**The inverse discrete Fourier transform (IDFT):**

A simple description of these equations is that the complex numbers $X_k$ represent the amplitude and phase of the different sinusoidal components of the input "signal" $x_n$. The DFT computes the $X_k$ from the $x_n$, while the IDFT shows how to compute the $x_n$ as a sum of sinusoidal components $X_k\exp(2\pi ikn / N) / N$ with frequency $k / N$ cycles per sample. By writing the equations in this form, we are making extensive use of Euler's formula to express sinusoids in terms of complex exponentials, which are much easier to manipulate.
Inverse Fourier transform can be represented as given below.

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1} x(k)e^{\frac{j2\pi kn}{N}} \quad ,n=0,1,2\ldots.N-1+$$

Let us define WN=$e^{\frac{-j2\pi}{N}}$

This is called twiddle factor. Hence DFT and IDFT equation can be written as,

$$X(K)=\sum_{n=0}^{N-1} x(n)w_N^{kn} \quad ,K=0,1,2\ldots\ldots\ldots\ldots\ldots N-1$$

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1} X(K)w_N^{-kn} \quad ,n=0,1,2\ldots\ldots\ldots N-1$$

Note that the normalization factor multiplying the DFT and IDFT (here 1 and 1/$N$) and the signs of the exponents are merely conventions, and differ in some treatments. The only requirements of these conventions are that the DFT and IDFT have opposite-sign exponents and that the product of their normalization factors be 1/$N$. A normalization of for both the DFT and IDFT makes the transforms unitary, which has some theoretical advantages, but it is often more practical in numerical computation to perform the scaling all at once as above (and a unit scaling can be convenient in other ways).

**2.2 Relationship between DFT and DTFT:**

The DTFT is given as

$$X(\Omega)=\sum_{n=-\alpha}^{\alpha} x(n)e^{-j\Omega n}$$

Evaluate this DTFT of $\Omega=2\pi k/N$, Where k=0,1,2….N-1

Then we get,

$$X(2\pi k/N)=\sum_{n=-\alpha}^{\alpha} x(n)e^{\frac{-j2\pi kn}{N}}$$

Here note that X(2πk/N) gives the DFT coefficients of periodic sequence.

$$xp(n)=\sum_{l=-\alpha}^{\alpha} x(n-lN)$$

Thus xp(n) is the sequence which is periodic with period N. Hence if sequence x(n) has the length less than N, then

xp(n)=x(n)

X(2πk/N)=X(K) i.e., DFT of x(n)

If the length of x(n) is greater than N, then DFT and DTFT will have no direct relationship.

## 2.3 Properties of DFT:

1.Periodicity Property:

$x(K+N)=X(K)$

2.Linearity

$DFT[a_1x_1(n)+a_2x_2(n)]=a_1x_1(K)+a_2X_2(K)$

3. Symmetry properties:

$x(n)=X_R(n)+jX_I(N)$

$X(K)=X_R(K)+jX_I(K)$

4.Symmetry property for real valued x(n):

$x(N-K)=X^*(K)=X(-K)$

5.Circular convolution :

$DFT[x_1\ N\ x_2(n)]=X_1(K)X_2(K)$

5.Time reversal of a sequence:

$DFT[x(n)]=X(K)$

$x((-n))_N=x(N-n)\overset{DFT}{\longleftrightarrow} X((-K))_N = X(N-K)$

6.Circular time shift of a sequence:

$x((n-l))_N\overset{DFT}{\longleftrightarrow} x(k)e^{\frac{-j2\pi kl}{N}}$

7.Circular frequency shift:

$DFT[x(n)]=X(K)$

$DFT[x(n)e^{\frac{j2\pi ln}{N}}] = X((K-l))_N$

8.Complex conjugate property:

$$x^*(n)\overset{DFT}{\longleftrightarrow}X^*((-K))_N=X^*(N-K) \text{ and } x^*((-n))_N=x^*(N-K)\overset{DFT}{\longleftrightarrow}X^*(K)$$

9.Circular convolution:

$$\bar{\gamma}_{xy}(l)\overset{DFT}{\longleftrightarrow}\bar{R}_{xy}(K)=X(K).Y^*(K)$$

10.Multiplication of two sequence:

$$x1(n)x2(n)\overset{DFT}{\longleftrightarrow}\frac{1}{N}X_1(K)\ N\ X_2(K)$$

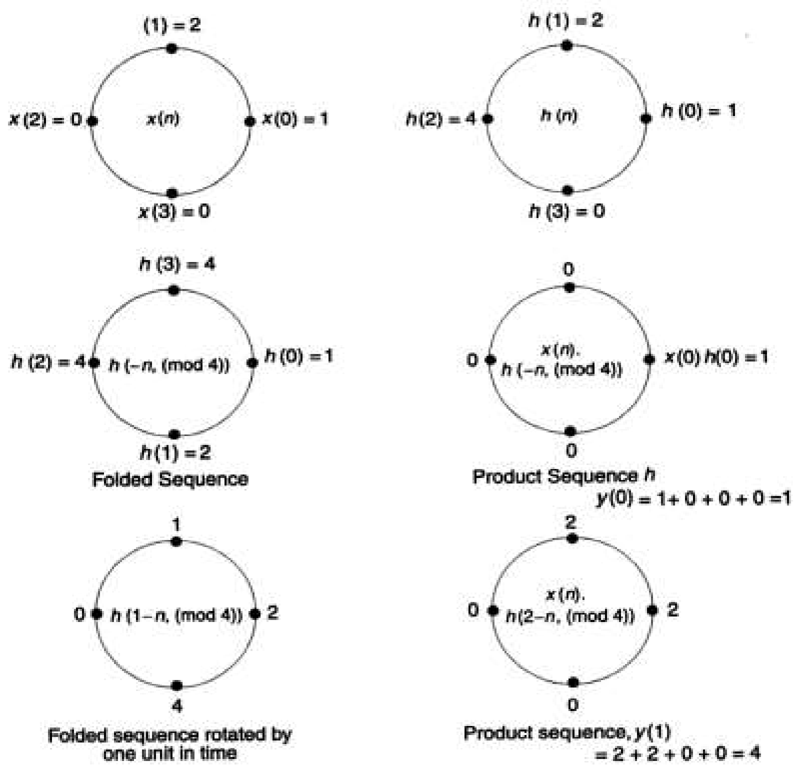
11.Parsevals theorem:

$$\sum_{n=0}^{N-1}x(n)y^*(n)=\frac{1}{N}\sum_{K=0}^{N-1}X(K)Y^*(K)$$

$$\sum_{n=0}^{N-1}|x(n)|^2=\frac{1}{N}\sum_{K=0}^{N-1}|X(K)|^2$$

## 2.4 Circular Convolution:

1.Find the response of the given sequence using circular convolution h(n)=(1,2,4} and x(n)=(1,2}



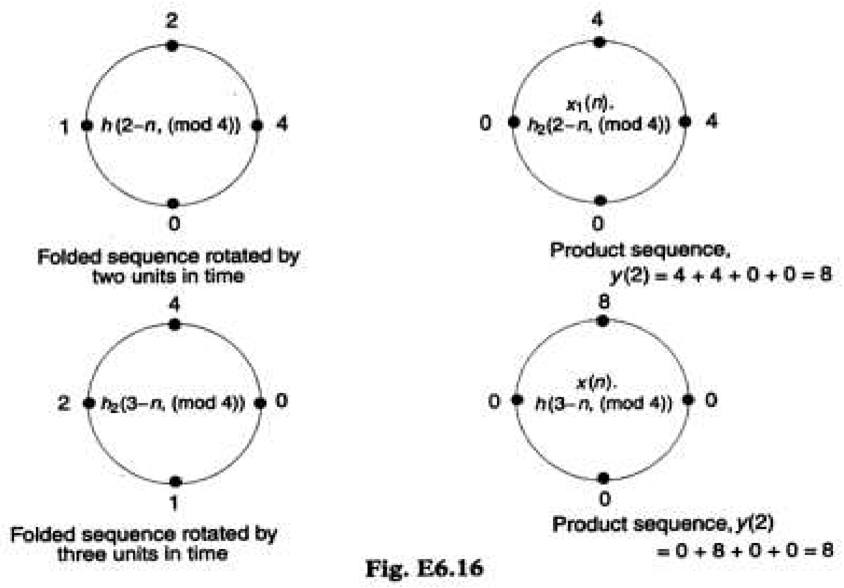

2
1
$h(2-n, (\text{mod } 4))$
4
0
Folded sequence rotated by two units in time
4
0
$x_1(n)$.
$h_2(2-n, (\text{mod } 4))$
4
0
Product sequence, $y(2) = 4 + 4 + 0 + 0 = 8$
4
2
$h_2(3-n, (\text{mod } 4))$
0
1
Folded sequence rotated by three units in time
8
0
$x(n)$.
$h(3-n, (\text{mod } 4))$
0
0
Product sequence, $y(2) = 0 + 8 + 0 + 0 = 8$

**Fig. E6.16**

## 2.5 FFT COMPUTATION USING DECIMATION IN TIME AND DECIMATION IN FREQUENCY ALGORITHMS:

### 2.5.1 Decimation-in-time (DIT) Radix-2 FFT:

The radix-2 algorithms are the simplest FFT algorithms. The decimation-in-time (DIT) radix-2 FFT recursively partitions a DFT into two half-length DFTs of the even-indexed and odd-indexed time samples. The outputs of these shorter FFTs are reused to compute many outputs, thus greatly reducing the total computational cost.

We know that DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-\frac{j2\pi nk}{N}}, \qquad k=0,1,2\ldots\ldots\ldots\ldots N-1$$

From the formula of DFT,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{kn}, \qquad k=0,1,2\ldots\ldots\ldots\ldots N-1$$

$$W_N = e^{-\frac{j2\pi}{N}} \qquad \Longrightarrow \text{Twiddle factor or phase factor}$$

Let N=8

$$X(k) = \sum_{n=0(EVEN)}^{\frac{N}{2}-1} x(n)\, W_N^{kn} + \sum_{n=0(ODD)}^{\frac{N}{2}-1} x(n)\, W_N^{kn}$$

$$X(k) = \sum_{r=0}^{N/2-1} x(2r)\, W_N^{2r} + \sum_{r=0}^{N/2-1} x(2r+1)\, W_N^{(2r+)k}$$

$$X(k) = \sum_{r=0}^{N/2-1} x(2r)\, W_N^{2rk} + \sum_{r=0}^{N/2-1} x(2r+1)\, W_N^{2rk} W_N^{k}$$

$$X(k) = \sum_{r=0}^{N/2-1} x(2r)\, W_N^{2rk} + W_N^{K} \sum_{r=0}^{N/2-1} x(2r+1)\, W_N^{2rk}$$

$$X(k)= \sum_{r=0}^{\frac{N}{2}-1} x(2r)\,(W_N^2)^{rk} + W_N^k \sum_{r=0}^{N/2-1} x(2r+1)\,(W_N^2)^{rk}$$

$$W_N^2 = W_{\frac{N}{2}}$$

$$X(k)= \sum_{r=0}^{\frac{N}{2}-1} x(2r)\,(W_{\frac{N}{2}})^{rk} + W_N^k \sum_{r=0}^{N/2-1} x(2r+1)\,(W_{\frac{N}{2}})^{rk}$$

X(K)=G(K)+$W_N^K H(K)$……………………(1)

| | |
|---|---|
| X(0)=G(0)+ $W_N^0 + H(0)$ | X(4)=G(0)+ $W_N^4 + H(0)$ |
| X(1)=G(1)+ $W_N^1 + H(1)$ | X(5)=G(1)+ $W_N^5 + H(1)$ |
| X(2)=G(2)+ $W_N^2 + H(2)$ | X(6)=G(2)+ $W_N^6 + H(2)$ |
| X(3)=G(3)+ $W_N^3 + H(3)$ | X(7)=G(3)+ $W_N^7 + H(3)$ |

Figure 2.1shows the Flow graph of the first stage Decimation –in-Time FFT Algorithm for N=8

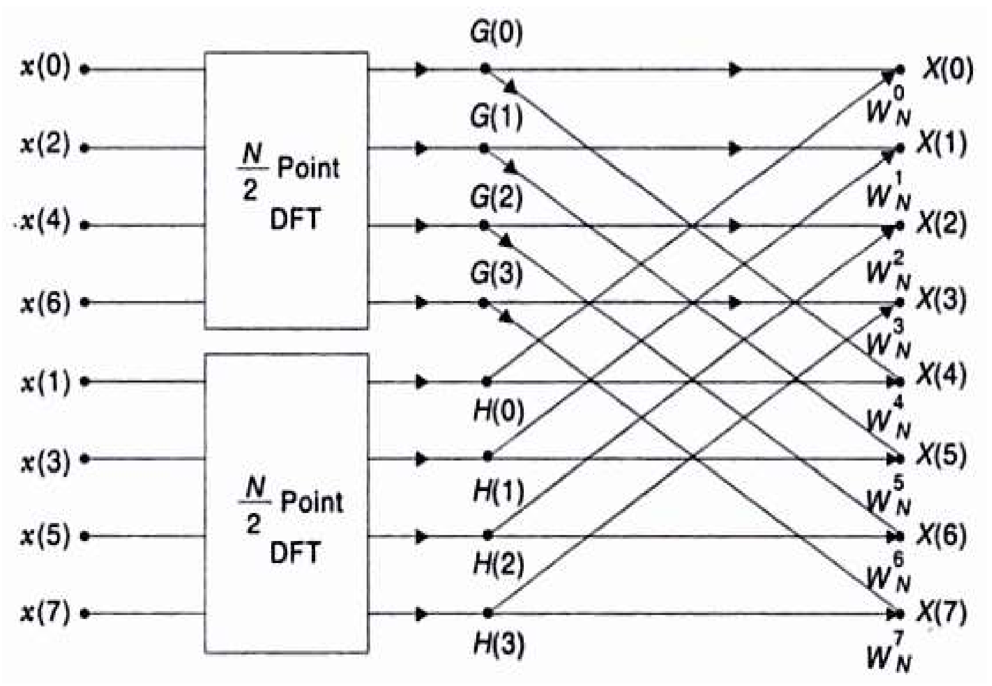


Fig 2.1:Flow graph of the first stage Decimation –in-Time FFT Algorithm for N=8

From equation (1) we know that

$$G(k)= \sum_{r=0}^{N/2-1} g(r)\,(W_{\frac{N}{2}})^{rk}$$

$$G(k)= \sum_{l=0}^{\frac{N}{4}-1} g(2l)\,(W_{\frac{N}{2}})^{2lk} + \sum_{l=0}^{\frac{N}{4}-1} g(2l+1)\,(W_{\frac{N}{2}})^{(2l+1)k}$$

$$G(k)=\sum_{l=0}^{\frac{N}{4}-1} g(2l)\,(W_{\frac{N}{4}})^{lk} + W_{\frac{N}{2}}^{k}\sum_{l=0}^{\frac{N}{4}-1} g(2l+1)\,(W_{\frac{N}{4}})^{lk}$$

$G(K)=A(K)+ W_{\frac{N}{2}}^{k}B(K)$……………………(2)

Figure 2.2 shows the Flow graph of the second stage of DIT-FFT algorithm for N=8

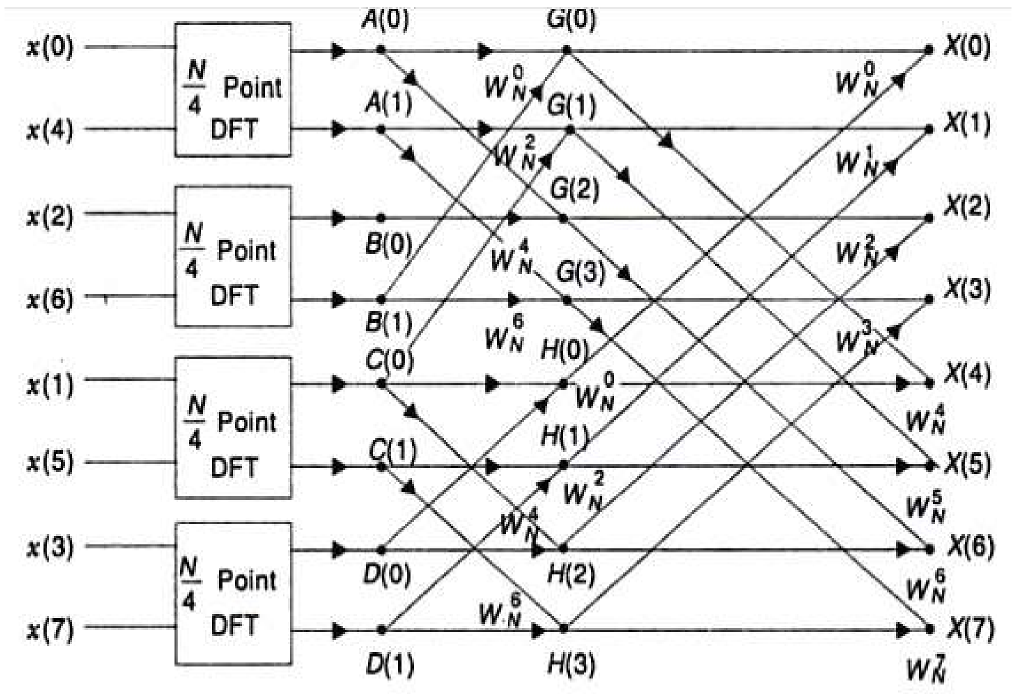


Fig 2.2 : Flow graph of the second stage of DIT-FFT algorithm for N=8

From equation (1) we know that

$$H(k)=\sum_{l=0}^{\frac{N}{4}-1} h(2l)\,(W_{\frac{N}{4}})^{lk} + \sum_{l=0}^{\frac{N}{4}-1} h(2l+1)\,(W_{\frac{N}{4}})^{lk}$$

$H(K)=C(K)+ W_{\frac{N}{2}}^{k}D(K)$ …………………..(3)

From equation (2)

For k=0, $G(0)=A(0)+ W_{\frac{N}{2}}^{0}B(0) = A(0) + W_{N}^{0}B(0)$

For k=1, $G(1)=A(1)+ W_{\frac{N}{2}}^{1}B(1) = A(1) + W_{N}^{2}B(1)$

For k=2, $G(2)=A(2)+ W_{\frac{N}{2}}^{2}B(2) = A(0) + W_{N}^{4}B(0)$

For k=3, $G(3)=A(3)+ W_{\frac{N}{2}}^{3}B(3) = A(1) + W_{N}^{6}B(1)$

From equation (3)

$H(0) = C(0) + W_N^0 D(0)$

$H(1) = C(1) + W_N^2 D(1)$

$H(2) = C(0) + W_N^4 D(0)$

$H(3) = C(1) + W_N^6 D(1)$

This is called decimation in time because the time samples are rearranged in alternating groups, and a radix -2 algorithm because there are two groups. Because of the periodicity with N=2, frequency samples of these length-N DFTs, G(k) and H(k) can be used to compute two of the length-N DFT frequencies.

Computational cost of radix-2 DIT FFT

- $N^2 \log_2 N$ complex multiplies
- $N \log_2 N$ complex adds

This is a remarkable savings over direct computation of the DFT. For example, a length-1024 DFT would require 1048576complex multiplications and 1047552 complex additions with direct computation, but only 5120 complex multiplications and 10240 complex additions using the radix-2 FFT, a savings by a factor of 100 or more. The relative savings increase with longer FFT lengths, and are less for shorter lengths. Figure 2.3 shows the Flow graph of the Decimation-in-Time FFT Algorithm for N=8

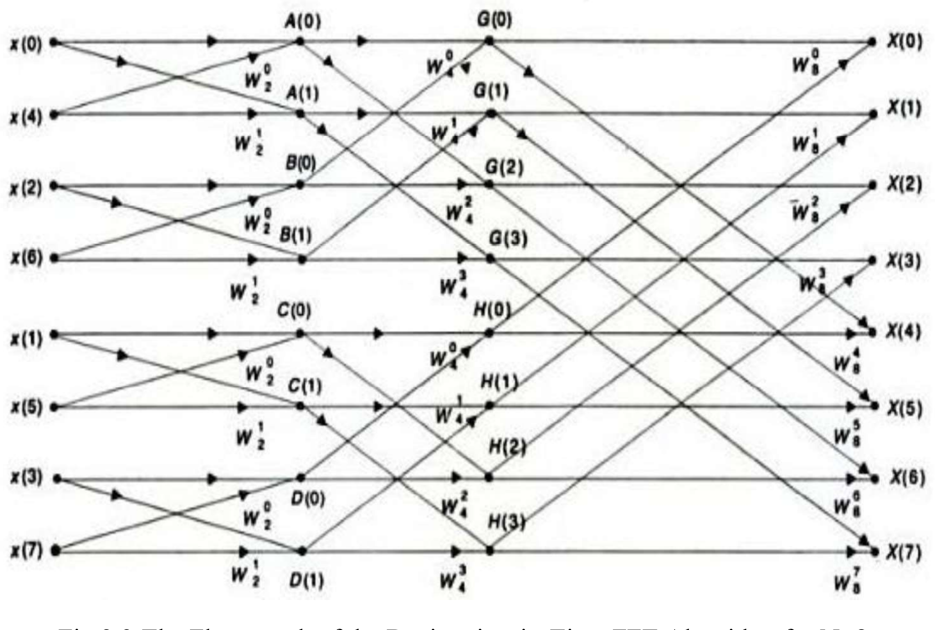


Fig 2.3:The Flow graph of the Decimation-in-Time FFT Algorithm for N=8

In a decimation-in-time radix-2 FFT the input is in bit-reversed order (hence "decimation-in-time"). That is, if the time-sample index n is written as a binary number, the order is that binary number reversed. The bit-reversal process is illustrated for a length-N=8. Figure2.4 shows the Basic butterfly flow graph for the computation in the DIT-FFT algorithm and Figure 2.5 shows the Reduced flow graph for an 8 point DIT-FFT.

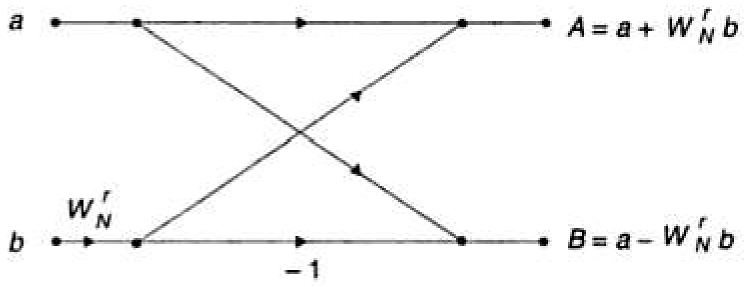


Fig 2.4: Basic butterfly flow graph for the computation in the DIT-FFT algorithm

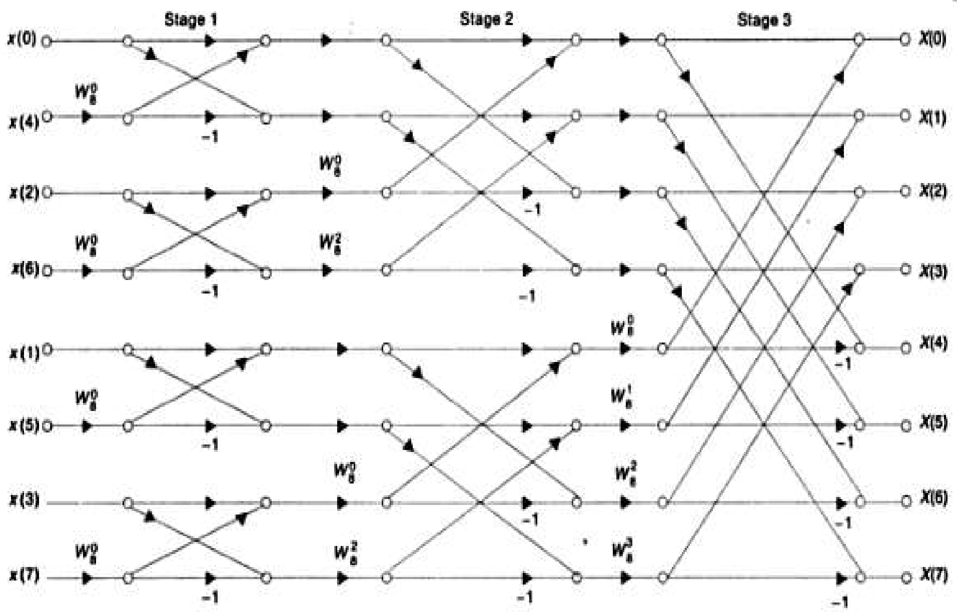


Fig 2.5: Reduced flow graph for an 8 point DIT-FFT

## 2.5.2 Decimation-in-frequency (DIF) Radix-2 FFT:

The decimation-in-frequency (DIF) radix-2 FFT partitions the DFT computation into even-indexed and odd-indexed outputs, which can each be computed by shorter-length DFTs of different combinations of input samples. Recursive application of this decomposition to the shorter-length DFTs results in the full radix-2 decimation-in-frequency FFT.

We know that DFT, $X(k)=\sum_{n=0}^{N-1} x(n)e^{-\frac{j2\pi nk}{N}}$

From the formula of DFT,

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

$W_N = e^{-\frac{j2\pi}{N}}$ $\Longrightarrow$ Twiddle factor or phase factor

Let N=8

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

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

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

We know that, $W_N = e^{-\frac{j2\pi}{N}}$

$W_N^{k\frac{N}{2}} = e^{-\frac{j2\pi}{N}*\frac{N}{2}k}$

$= e^{-j\pi k}$

$= \cos \pi k - j \sin \pi k$

$=(-1)^k$

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

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

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

$X(2r)= \sum_{n=0}^{\frac{N}{2}-1} [x(n) + x(n+\frac{N}{2})]\,W_{\frac{N}{2}}^{rn}$ …………………(1)

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

$X(2r+1)= \sum_{n=0}^{\frac{N}{2}-1} [x(n) - x(n+\frac{N}{2})]\,W_N^{n} W_{\frac{N}{2}}^{rn}$ …………………(2)

From equation (1) and (2)

g(n)=x(n)+x(n+N/2) and h(n)= x(n)-x(n+N/2)

g(0)=x(0)+x(4) h(0)=x(0)-x(4)

g(1)=x(1)+x(5) h(1)=x(1)-x(5)

g(2)=x(2)+x(6) h(2)=x(2)+x(6)

g(3)=x(3)+x(7)    h(0)=x(3)+x(7)

Figure 2.6 shows the Flow graph of the first stage Decimation –in-Frequency FFT Algorithm for N=8

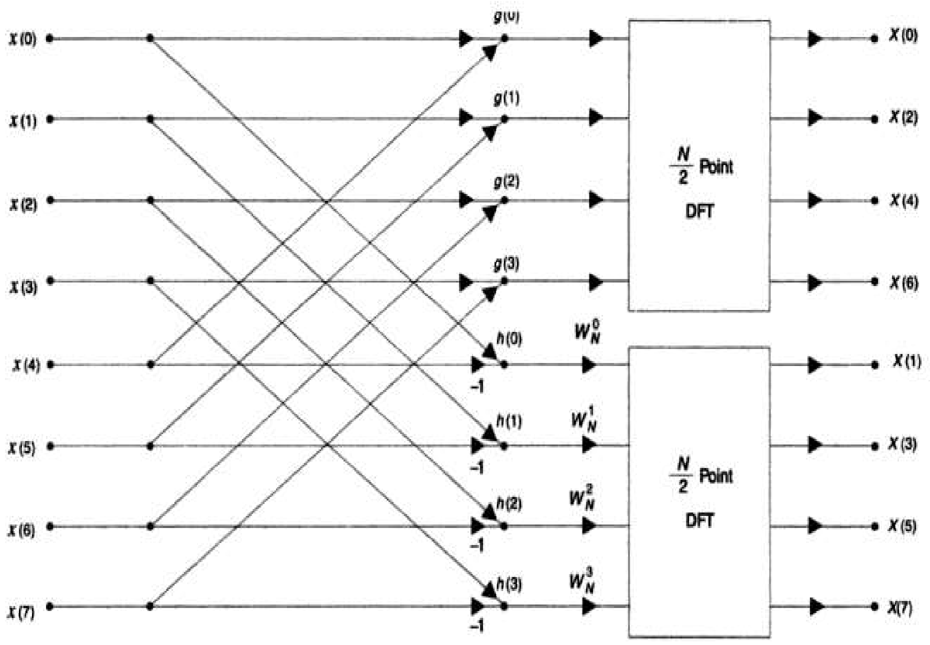


Fig 2.6:Flow graph of the first stage Decimation –in-Frequency FFT Algorithm for N=8

From equation (1) we know that

$X(2r)= \sum_{n=0}^{\frac{N}{2}-1} g(n)\, W_N^{2rn}$

g(n)=x(n)+x(n+N/2)

$X(2r)= \sum_{n=0}^{\frac{N}{4}-1} g(n)\, W_N^{2rn} + \sum_{n=N/4}^{\frac{N}{2}-1} g(n)\, W_N^{2rn}$

$X(2r)= \sum_{n=0}^{\frac{N}{4}-1} g(n) W_N^{2rn} + \sum_{n=0}^{\frac{N}{4}-1} g(n+N/2)\; W_N^{2r(n+\frac{N}{4})}$

$X(2r)= \sum_{n=0}^{\frac{N}{4}-1} g(n) W_N^{2rn} + W_N^{2r(\frac{N}{4})} \sum_{n=0}^{\frac{N}{4}-1} g(n+N/2)\; W_N^{2rn}$

$$W_N^{\frac{N}{2}} = -1$$

$X(2r)= \sum_{n=0}^{\frac{N}{4}-1} [g(n) +(-1)^r\; g(n+\frac{N}{4})]\, W_N^{2rn}$

When r=2l(even),

$X(4l)= \sum_{n=0}^{\frac{N}{4}-1} [g(n) +\; g(n+\frac{N}{4})]\, W_N^{4ln}$

X(4l)= $\sum_{n=0}^{\frac{N}{4}-1}[A(n)]\, W_N^{4ln}$ ,where A(n)=g(n)+g(n+N/4)

A(0)=g(0)+g(2)

A(1)=g(1)+g(3)

When r=2l+1(odd)

X(4l+2)= $\sum_{n=0}^{\frac{N}{4}-1}[g(n)-g(n+\frac{N}{4})]\, W_N^{2n(2l+1)}$

$= \sum_{n=0}^{\frac{N}{4}-1}[B(n)]\, W_N^{2n} W_N^{4ln}$

Where, B(n)=g(n)-g(n+N/4)

B(0)=g(0)-g(2)

B(1)=g(1)-g(3)

Similarly, From equation (2) we know that

X(2r+1)= $\sum_{n=0}^{\frac{N}{2}-1}[h(n)]\, W_N^{2rn} W_N^{n}$

X(2r+1)= $\sum_{n=0}^{\frac{N}{4}-1} h(n)\, W_N^{2rn} W_N^{n} + \sum_{n=N/4}^{\frac{N}{2}-1} h(n)\, W_N^{2rn} W_N^{n}$

X(2r+1)= $\sum_{n=0}^{\frac{N}{4}-1} h(n)\, W_N^{2rn} W_N^{n} + \sum_{n=0}^{\frac{N}{2}-1} h(n+N/4)\, W_N^{(n+\frac{N}{4})} W_N^{2r(n+\frac{N}{4})}$

X(2r+1)= $\sum_{n=0}^{\frac{N}{4}-1} h(n)\, W_N^{2rn} W_N^{n} + \sum_{n=0}^{\frac{N}{2}-1} h(n+N/4) W_N^{n}\, W_N^{2rn} W_N^{(2r+1)\frac{N}{4}}$

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

X(2r+1)= $\displaystyle\sum_{n=0}^{\frac{N}{4}-1}[h(n) + (-1)^r W_{\frac{N}{2}}\, h(n+\frac{N}{4})] W_N^{(2r+1)n}$

Figure 2.7 shows the Flow graph of the second stage of DIF-FFT algorithm for N=8 and Figure 2.8 shows the Reduced Flow Graph of Radix-2 decimation-in-frequency FFT for a length-8 signal

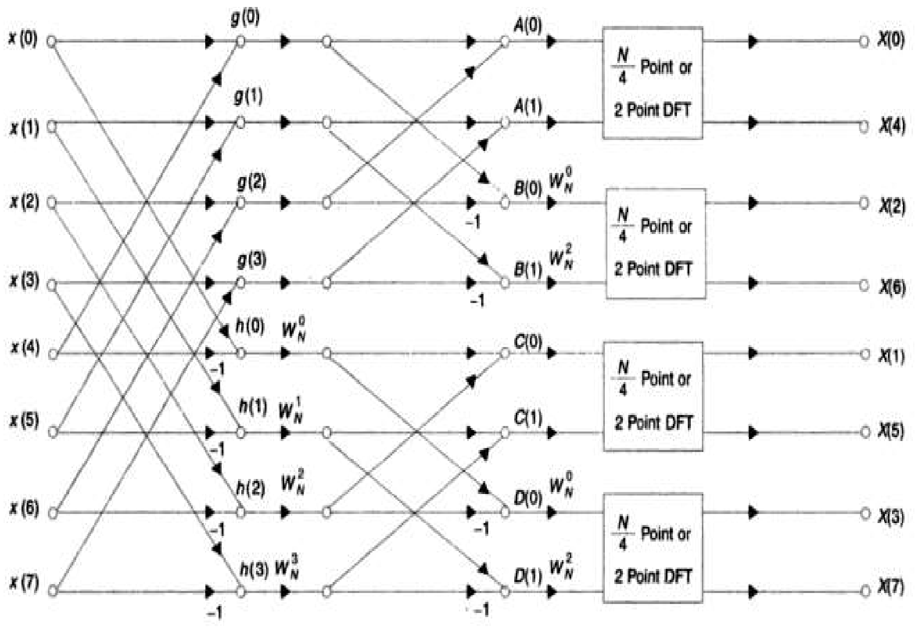


Fig 2.7 : Flow graph of the second stage of DIF-FFT algorithm for N=8

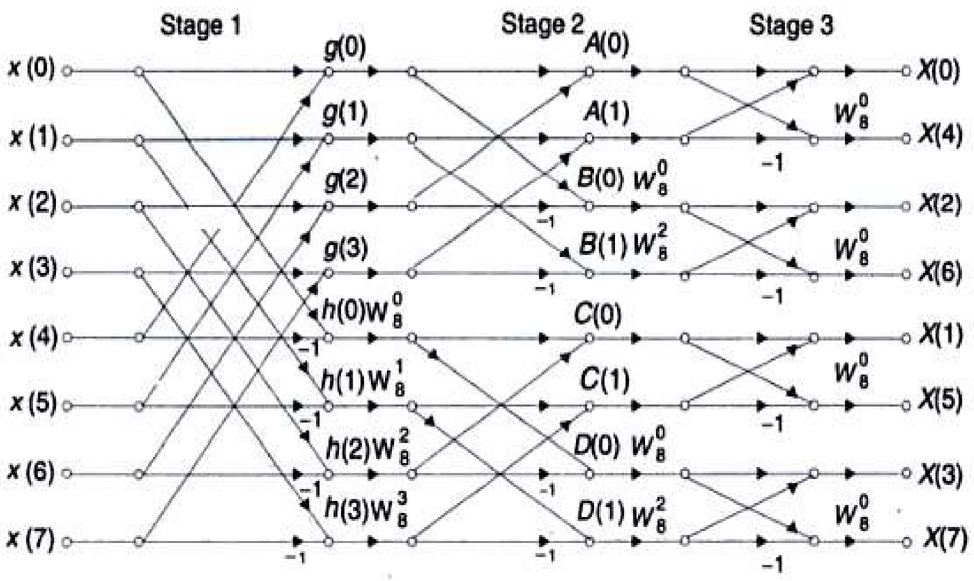


Fig 2.8: Reduced Flow Graph of Radix-2 decimation-in-frequency FFT for a length-8 signal

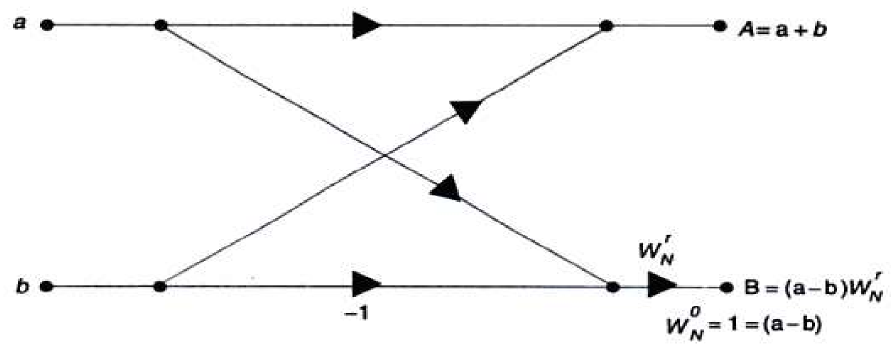
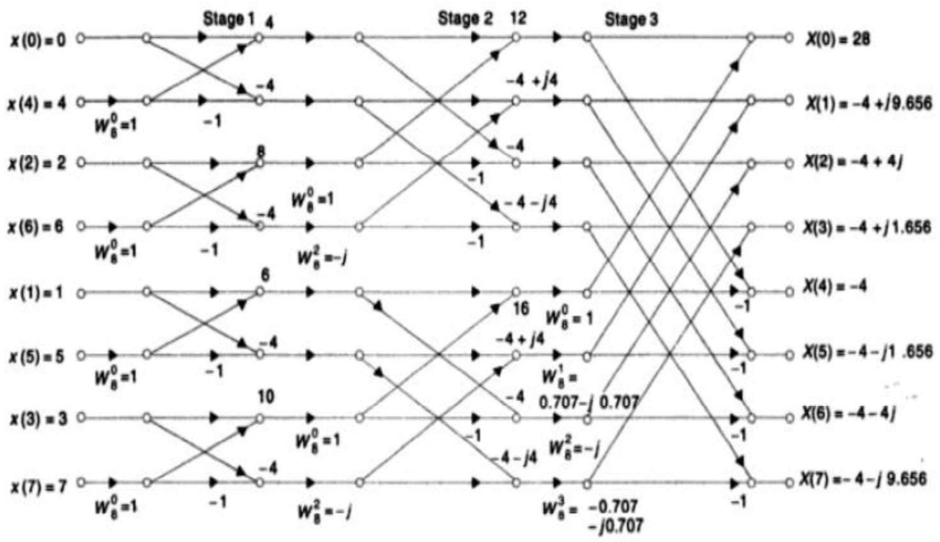


Fig 2.9: Basic butterfly flow graph for the computation in the DIF-FFT algorithm

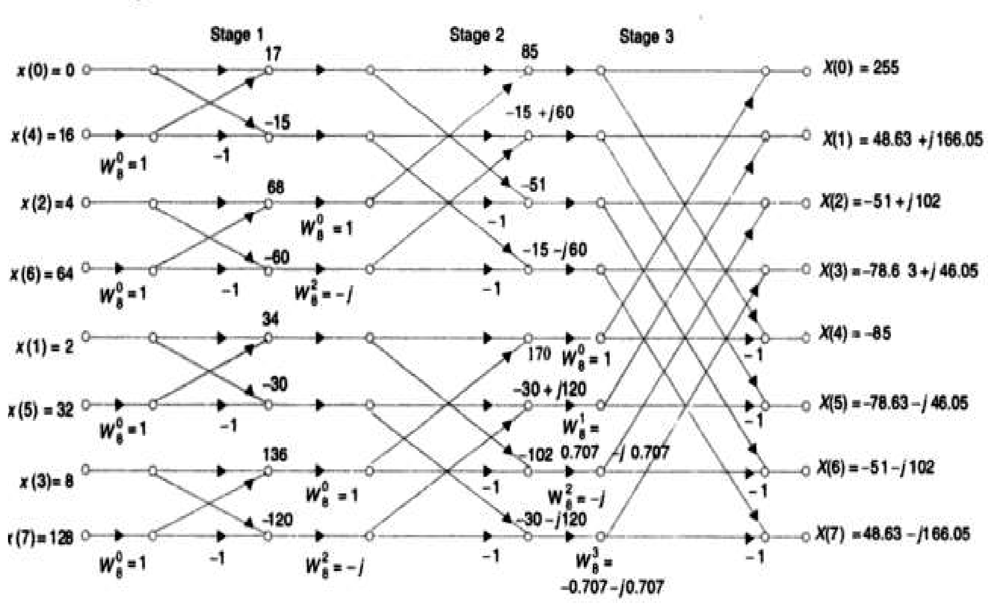
This is called decimation in frequency because the frequency samples are computed separately in alternating groups, and a radix-2 algorithm because there are two groups. The same radix-2 decimation in frequency can be applied recursively to the two length-$N^2$ DFTs to save additional computation. When successively applied until the shorter and shorter DFTs reach length-2, the result is the radix-2 decimation-in-frequency FFT algorithm. The full radix-2 decimation-in-frequency decomposition illustrated in given Figure requires M=$\log^2 N$ stages.

**Problems**:

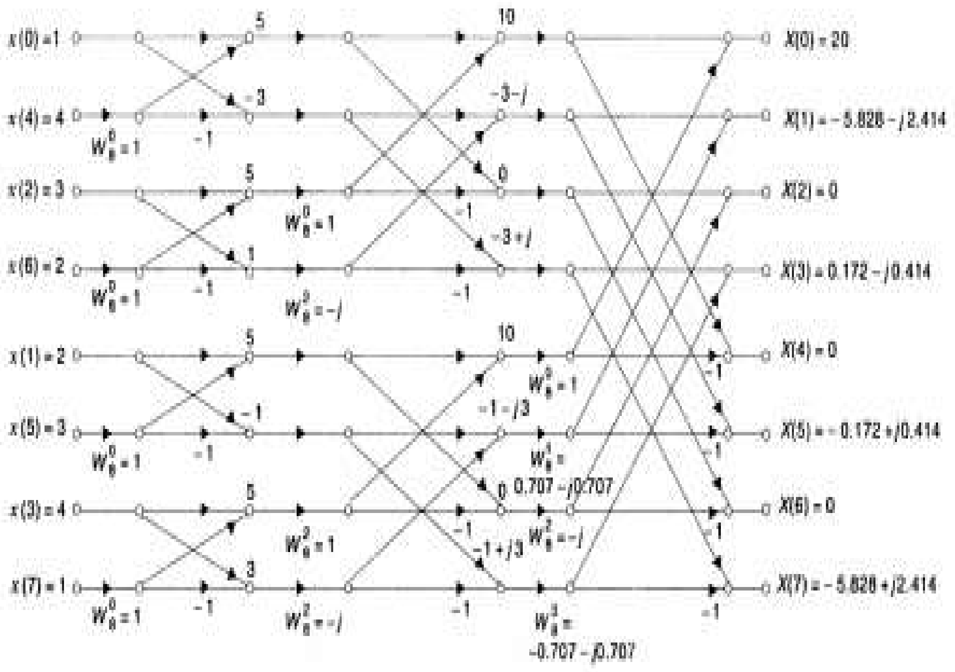
1. Given x(n)={0,1,2,3,4,5,6,7},find X(K) using DIT FFT algorithm.



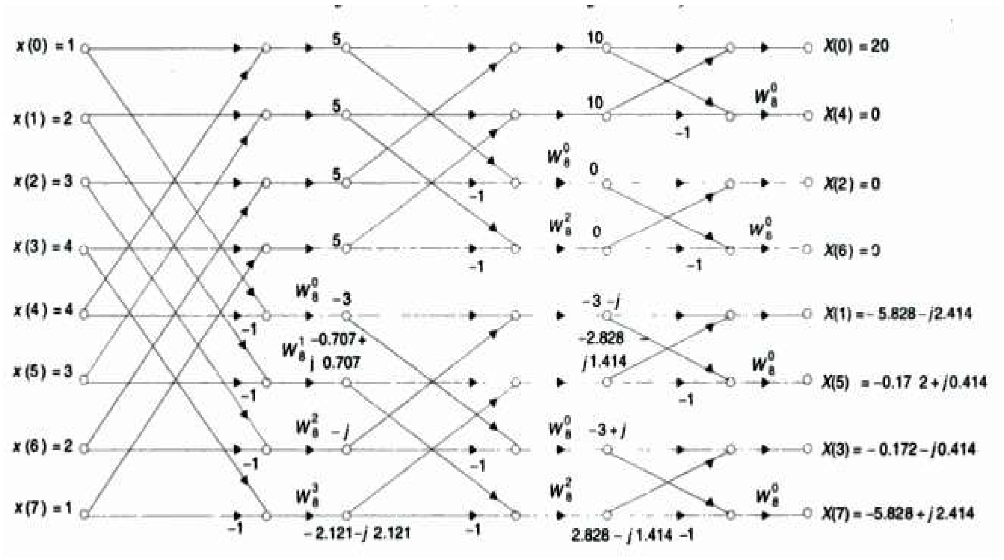
2.Given x(n)=$2^n$ ,find X(K) using DIT FFT algorithm.



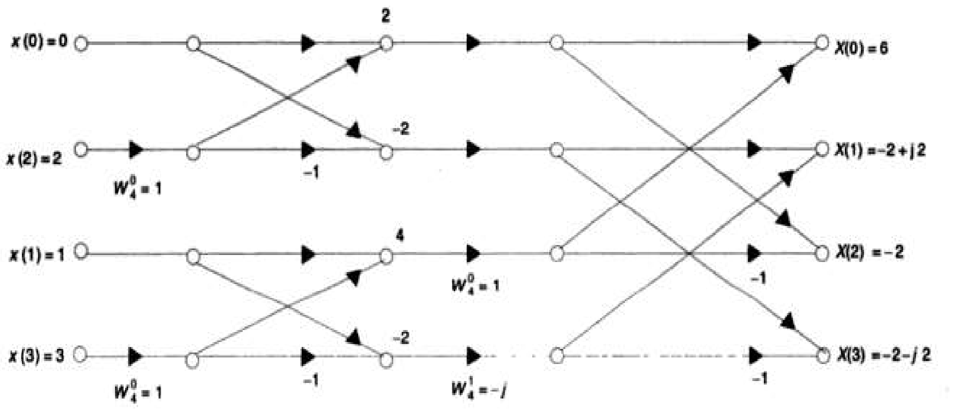
3.Given x(n)={1,2,3,4,4,3,2,1} ,find X(K) using DIT FFT algorithm.



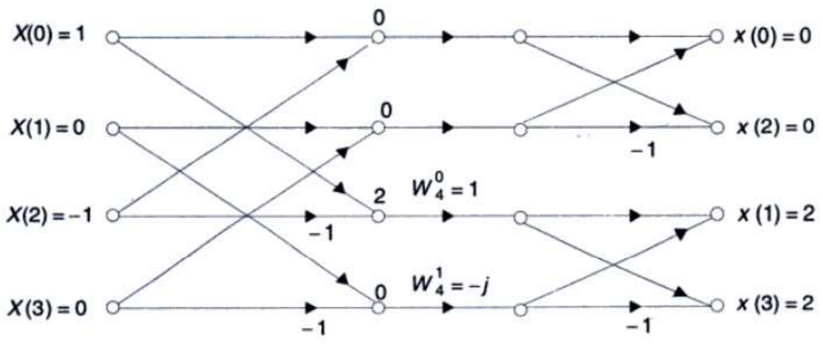
4.Given x(n)={1,2,3,4,4,3,2,1} ,find X(K) using DIF FFT algorithm.



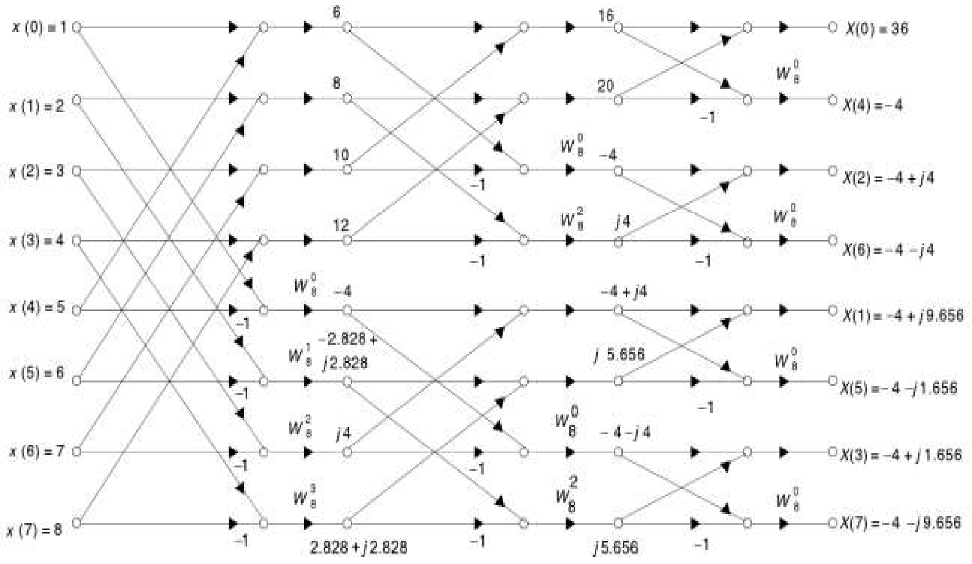
5.Given x(n)={0,1,2,3} ,find X(K) using DIT FFT algorithm.



6.Given x(n)=cos(nπ/2)find X(K) using DIF FFT algorithm

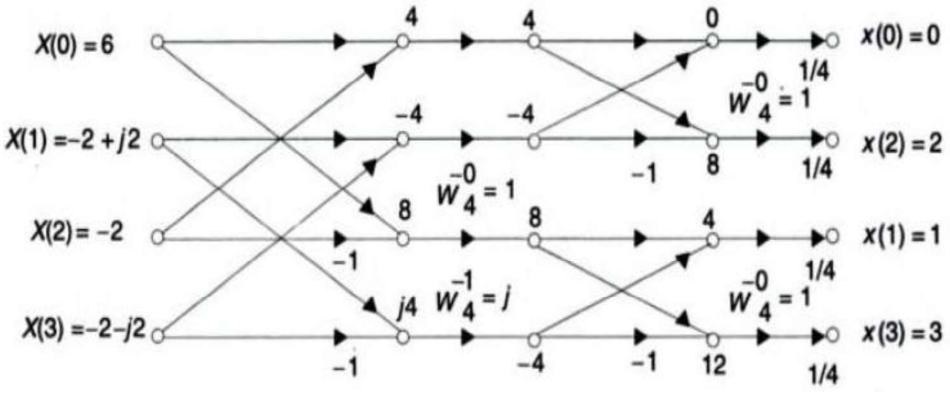


7. Given x(n)=n+1find X(K) using DIF FFT algorithm.

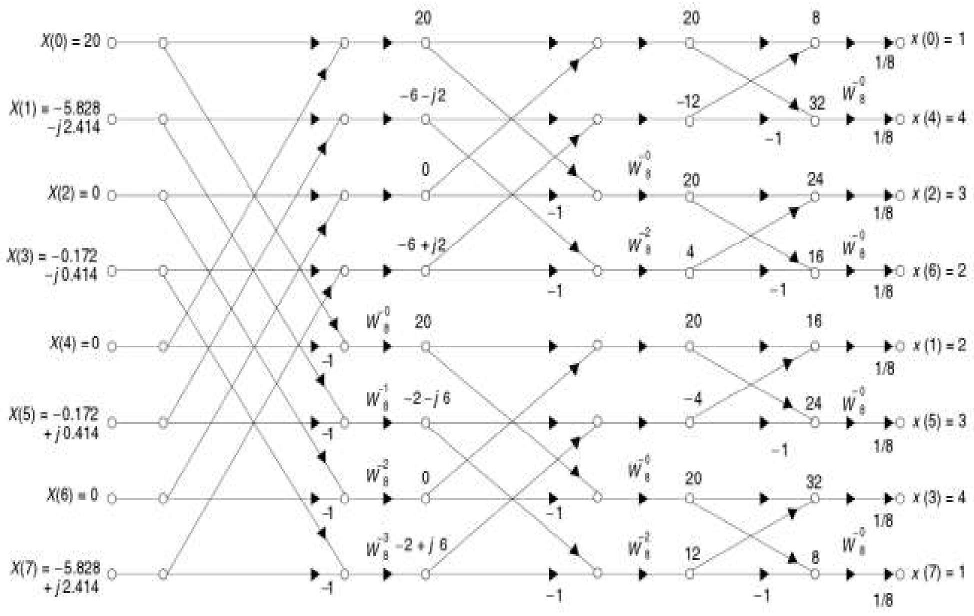


Inverse DFT Problems:

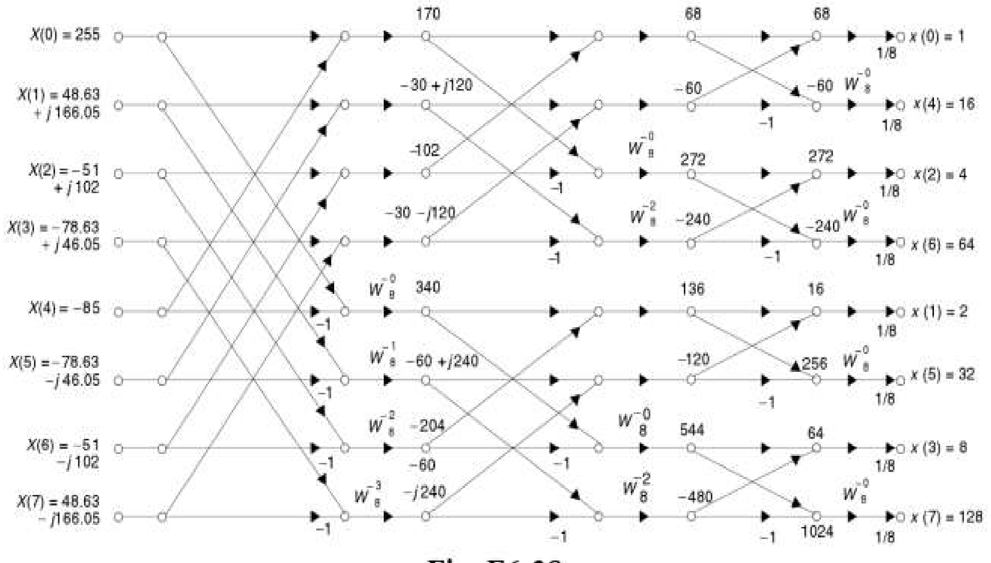
1.Use the four point inverse FFT find x(n).X(K)={6,-2+i2,-2,-2-i2}



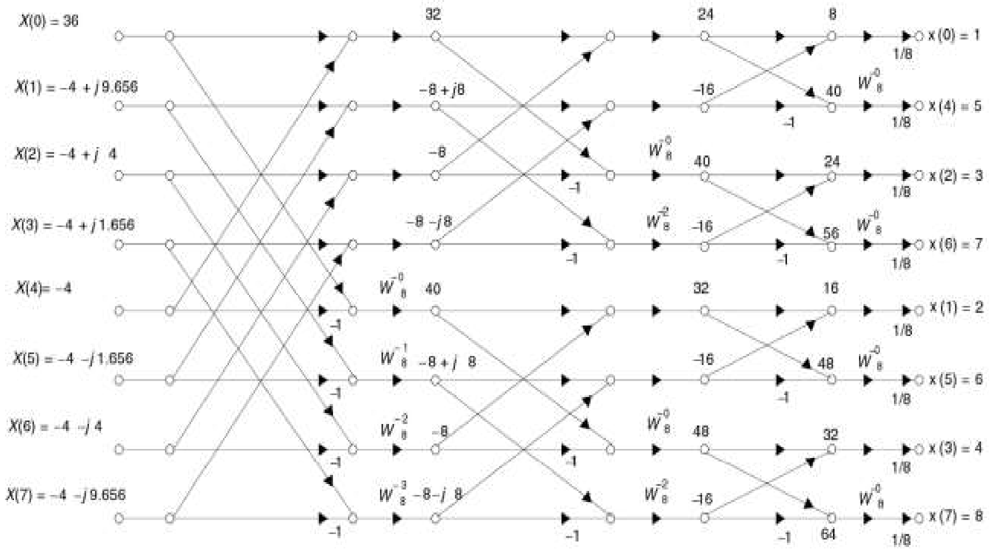
2.Given X(K)={20,-5.828-i2.414,0,-0.172-i0.414,0,-0.172+i0.414,0,-5.828+i2.414},find x(n)



3.Given X(K)={255,48.63+i166.05,-51+i102,-78.63+i46.05,-85,-78.63-i46.05,-51-i102,48.63-i166.05},find x(n)



4.Given X(K)={36,-4+i9.656,-4+i4,-4+i1.656,-4,-4-i1.656,-4-i4,-4-i9.656},find x(n)



## 2.6 Fast Convolution:

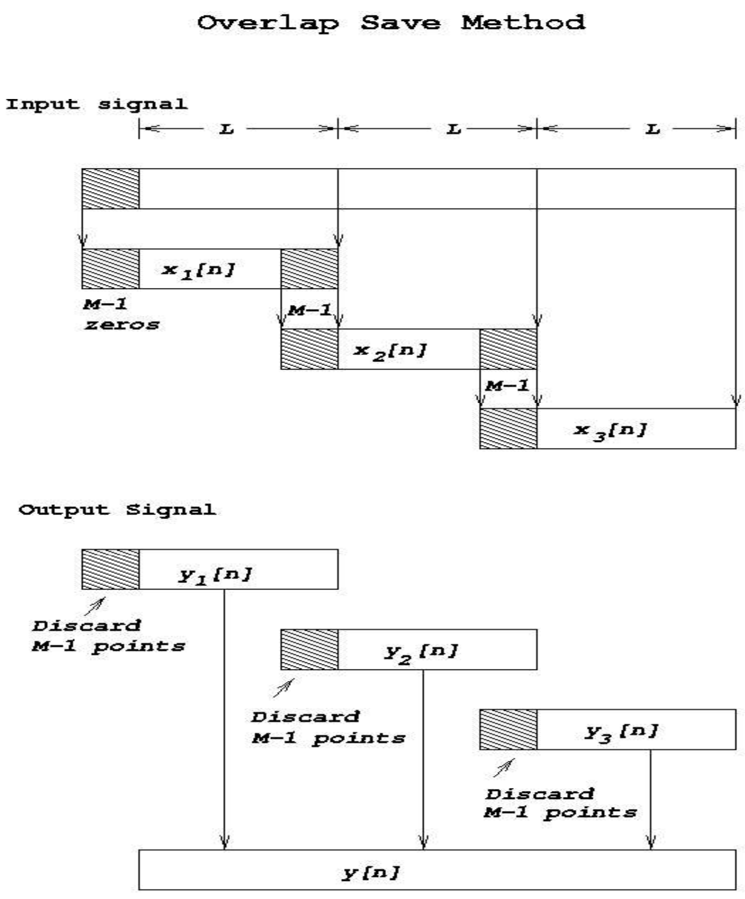
In a LTI system the system response is got by convoluting the input with the impulse response. In the frequency domain their respective spectra are multiplied. These spectra are continuous and hence cannot be used for computations. The product of 2 DFT s is equivalent to the circular convolution of the corresponding time domain sequences. Circular convolution cannot be used to determine the output of a linear filter to a given input sequence. In this case a frequency domain methodology equivalent to linear convolution is required. Linear convolution can be implemented using circular convolution by taking the length of the convolution as N >= n1+n2-1 where n1 and n2 are the lengths of the 2 sequences.

### 2.6.1.Overlap–save method:

In this method x (n) is divided into blocks of length N with an overlap of k-1 samples. The first block is zero padded with k-1 zeros at the beginning. H (n) is also zero padded to length N. Circular convolution of each block is performed using the N length DFT .The output signal is obtained after discarding the first k-1 samples the final result is obtained by adding the intermediate results.

In this method the size of the I/P data blocks is N= L+M-1 and the size of the DFts and IDFTs are of length N. Each data block consists of the last M-1 data points of the previous data block followed by L new data points to form a data sequence of length N= L+M-1. An N- point DFT is computed from each data block. The impulse response of the FIR filter is increased in length by appending L-1 zeros and an N-point DFT of the sequence is computed once and stored.

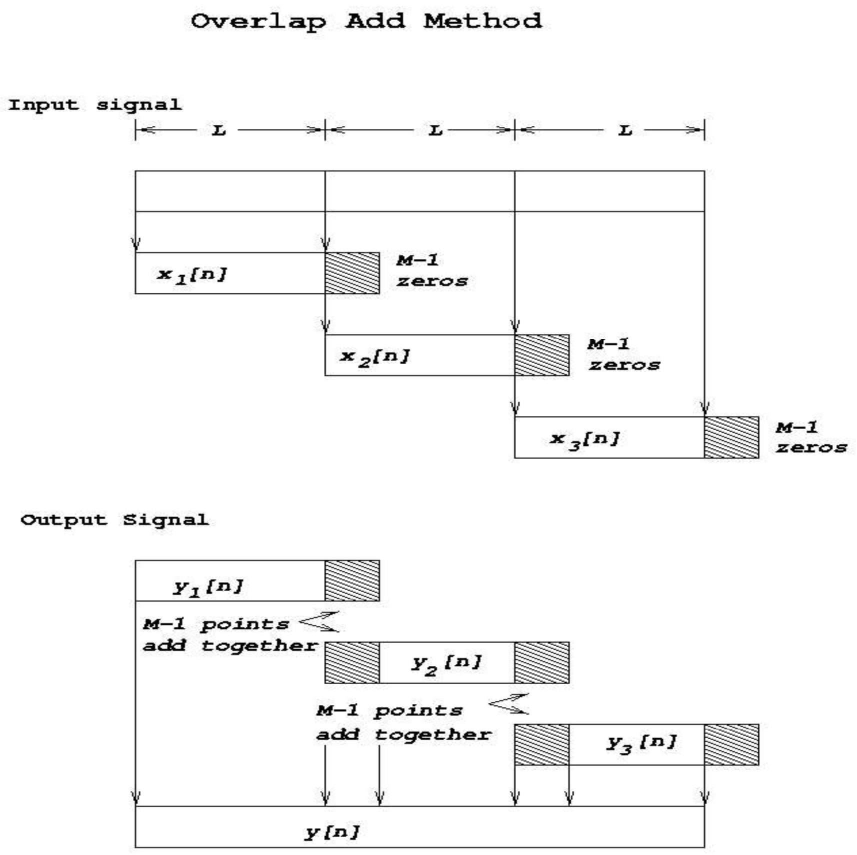
The multiplication of two N-point DFTs {H(k)} and {Xm(k)} for the mth block of data yields Since the data record is of the length N, the first M-1 points of Ym(n) are corrupted by aliasing and must be discarded. The last L points of Ym(n) are exactly the same as the result from linear convolution and as a consequence we get,



**2.6.2.Overlap-add method:**

In order to convolve a short duration sequence with a long duration sequence x(n) ,x(n) is split into blocks of length N x(n) and h(n) are zero padded to length L+M-1 . circular convolution is performed to each block then the results are added. These data blocks may be represented as

The IDFT yields data blocks of length N that are free of aliasing since the size of the DFTs and IDFT is N = L+M -1 and the sequences are increased to N-points by appending zeros to each block. Since each block is terminated with M-1 zeros, the last M-1 points from each output block must be overlapped and added to the first M-1 points of the succeeding block. Hence this method is called the overlap method. This overlapping and adding yields the output sequences given below.



**Two Mark Questions and Answers:**

1. What is a continuous and discrete time signal?

Continuous time signal: A signal x(t) is said to be continuous if it is defined for all time t. Continuous time signal arise naturally when a physical waveform such as acoustics wave or light wave is converted into a electrical signal. This is affected by means of transducer. (e.g.) microphone, photocell.

Discrete time signal: A discrete time signal is defined only at discrete instants of time. The independent variable has discrete values only, which are uniformly spaced. A discrete time signal is often derived from the continuous time signal by sampling it at a uniform rate.

2. Give the classification of signals?

Continuous-time and discrete time signals

Even and odd signals