19ECE211 Digital Signal Processing

Module 1

Prepared by,

Dr.J.Aravinth

Outline

- Linear filtering using DFT
- Linear and Circular convolution
- Efficient computation using of DFT Algorithms

TIME-FREQUENCY ANALYSIS OF FOURIER SERIES AND FOURIER TRANSFORMS

	Time Domain	Frequency Domain
Continuous-Time Fourier Series (CTFS)	Continuous & Periodic	Discrete & Aperiodic
Continuous-Time Fourier Transform (CTFT)	Continuous	Continuous & Periodic
Discrete Fourier Series (DFS)	Discrete & Periodic	Discrete & Aperiodic
Discrete-Time Fourier Transform (DTFT)	Discrete	Continuous & Periodic
Discrete Fourier Transform (DFT)	Discrete & Periodic	Discrete & Periodic

∞

DFT Vs DTFT

- $X(e^{jw})$ is the FT transform of DT-signal x(n).The range of ‗w‘ is from 0 to 2π (or) - π to π.Hence it is not possible to compute $X(e^{jw})$ on digital computer.
- Because in expression the range of summation is from $-\infty$ to ∞.
- If we make the range is finite, then it is possible to do the calculations on computer

Discrete-Time Fourier Transform

- Many sequences can be expressed as a weighted sum of complex exponentials as

$$X\left(e^{j\omega}\right) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n} \quad (\text{forward transform})$$

- Where the weighting is determined as

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X\left(e^{j\omega}\right) e^{j\omega n} d\omega \qquad (\text{inverse transform})$$

  - $X\left(e^{j\omega}\right)$ is the Fourier spectrum of the sequence x[n]
  - The phase wraps at 2π hence is not uniquely specified
  - The frequency response of a LTI system is the DTFT of the impulse response

$$H\left(e^{j\omega}\right) = \sum_{k=-\infty}^{\infty} h[k] e^{-j\omega k} \quad \text{and} \quad h[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} H\left(e^{j\omega}\right) e^{j\omega n} d\omega$$

DISCRETE FOURIER TRANSFORM (DFT)

- DFT-Defines a relationship between a signal in the Time-Domain and its representation in Frequency Domain.
- The DFS provides a mechanism for numerically computing the DTFT. It also alerted us to a potential problem of aliasing in the time domain.
- Sampling of the DTFT result in a periodic sequence $\tilde{x}(n)$. But most of the signals in practice are not periodic. They are likely to be of finite duration.

Possibility of numerically computable Fourier representation

for such signals?

- From the theoretical conclusion, define a periodic signal whose primary shape is that of the finite-duration signal and then using the DFS on this periodic signal.
- In Practical, define a new transform called the *discrete Fourier transform* (DFT), which is the primary period of the DFS.
- This DFT is the ultimate numerically computable Fourier transform for arbitrary finite-duration sequences.

DISCRETE FOURIER TRANSFORM (DFT)

- The discrete-time Fourier transform provided the frequency domain (ω) representation for absolutely summable sequences. The transforms (DTFT and ZT) have two features in common.
    1. The Transforms are defined for infinite-length sequences. (ie. Numerical computation viewpoint or from MatLab's viewpoint)
    2. They are functions of continuous variables (ω or z).

- These two features are troublesome because one has to evaluate *infinite sums* at *uncountably infinite* frequencies.

- To compute these transforms using software(MatLab, Scilab) the sequence have to truncated and then evaluate the expressions at finitely many points.(Evaluations approximated to the exact calculations).

- Therefore the DTFT and the ZT are *not numerically computable* transforms.

DISCRETE FOURIER TRANSFORM (DFT)

- Numerically computable transform –

  by sampling the DTFT in the frequency domain (or the *z*-transform on the unit circle).

- This transform is developed by analyzing periodic sequences.

- In Fourier analysis, a periodic function (or sequence) represented by a linear combination of harmonically related complex exponentials (which is a form of sampling). This gives us the *discrete Fourier series* (DFS) representation.

- The extension of DFS to *finite-duration sequences*, which leads to a new transform, called the *discrete Fourier transform* (DFT).

- **The DFT avoids the TWO PROBLEMS mentioned and is a numerically computable transform that is suitable for computer implementation.**

- Now consider $\omega = 2\pi k/N$

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

$$\begin{aligned} X\left(\frac{2\pi}{N}k\right) &= \cdots + \sum_{n=-N}^{-1} x(n)e^{-j2\pi kn/N} + \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N} \\ &\quad + \sum_{n=N}^{2N-1} x(n)e^{-j2\pi kn/N} + \cdots \\ &= \sum_{l=-\infty}^{\infty} \sum_{n=lN}^{lN+N-1} x(n)e^{-j2\pi kn/N} \end{aligned}$$

- By changing the index in the inner summation from n to $n - lN$ and interchanging the order of summation

$$\begin{aligned} X\left(\frac{2\pi}{N}k\right) &= \sum_{n=0}^{N-1}\left[\sum_{l=-\infty}^{\infty} x(n-lN)\right] e^{-j\frac{2\pi}{N}k(n-lN)} \\ &= \sum_{n=0}^{N-1}\left[\sum_{l=-\infty}^{\infty} x(n-lN)\right] e^{-j\frac{2\pi}{N}kn} e^{-j2\pi kl} \end{aligned}$$

$$e^{-j2\pi kl} = 1 \because \text{both k and l integers}$$

$$X\left(\frac{2\pi}{N}k\right) = \sum_{n=0}^{N-1}\left[\sum_{l=-\infty}^{\infty} x(n-lN)\right] e^{-j2\pi kn/N} \qquad k=0,1,2,\ldots N-1$$

$$Let \qquad x_p(n) = \sum_{l=-\infty}^{\infty} x(n-lN)$$

- The term $x_p(n)$ is obtained by the periodic repetition of $x(n)$ every N samples hence it is a periodic signal. This can be expanded by Fourier series as

$$x_p(n) = \sum_{k=0}^{N-1} c_k e^{j2\pi kn/N} \qquad n=0,1,\cdots N-1$$

- where c_k is the fourier coefficients expressed as

$$c_k = \frac{1}{N}\sum_{n=0}^{N-1} x_p(n) e^{-j2\pi kn/N} \qquad k=0,1,\cdots N-1$$

- Upon comparing

$$c_k = \frac{1}{N} X\left(\frac{2\pi}{N}k\right) \qquad k=0,1,\cdots N-1$$

$$x_p(n) = \frac{1}{N}\sum_{k=0}^{N-1} X\left(\frac{2\pi}{N}k\right) e^{j2\pi kn/N} \quad n = 0, 1, \cdots N-1$$

- $x_p(n)$ is the reconstruction of the periodic signal from the spectrum $X(\omega)$ (IDFT).
- The equally spaced frequency samples $X\left(\frac{2\pi}{N}\right) \quad k = 0, 1, \cdots N-1$ do not uniquely represent the original sequence when $x(n)$ has infinite duration. When $x(n)$ has a finite duration then $x_p(n)$ is a periodic repetition of $x(n)$ and $x_p(n)$ over a single period is

$$x_p(n) = \begin{cases} x(n) & 0 \le n \le L-1 \\ 0 & L \le n \le N-1 \end{cases}$$

- For the finite duration sequence of length L the Fourier transform is:

$$X(\omega) = \sum_{n=0}^{L-1} x(n)e^{-j\omega n} \quad 0 \le \omega \le 2\pi$$

- When $X(\omega)$ is sampled at frequencies $\omega_k = 2\pi k/N \quad k = 0, 1, 2, \ldots N-1$ then

$$X(k) = X\left(\frac{2\pi k}{N}\right) = \sum_{n=0}^{L-1} x(n)e^{-j2\pi kn/N} = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}$$

- The upper index in the sum has been increased from $L-1$ to $N-1$ since x(n)=0 for $n \ge L$

DISCRETE FOURIER TRANSFORM PAIRS

Analysis Equation

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}$$

Synthesis Equation

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}$$

$$x[n] \overset{DFT}{\longleftrightarrow} X[k]$$

DISCRETE FOURIER TRANSFORM (DFT)

- **The DFT provides uniformly spaced samples of the Discrete-Time Fourier Transform (DTFT).**

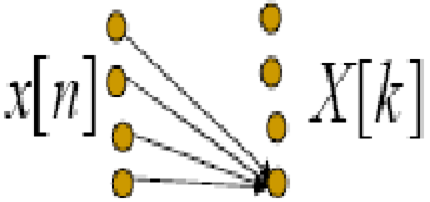
Discrete Fourier transform (DFT) of a discrete-time signal $x[n]$ with finite extent n □ $[0, N-1]$

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi nk}{N}} \qquad x[n] = \frac{1}{N}\sum_{n=0}^{N-1} X[k] e^{j\frac{2\pi nk}{N}}$$

Twiddle factor:

$$W_N^k = e^{-j(2\pi/N)k} = \cos(2\pi k/N) + j\sin(2\pi k/N), \quad k = 0,1,\ldots,N-1$$

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



DISCRETE FOURIER TRANSFORM (DFT)

With these definitions, the N-point DFT may be expressed in matrix form as,

$$\mathbf{X}_N = \mathbf{W}_N \mathbf{x}_N$$

The inverse DFT

$$\mathbf{x}_N = \mathbf{W}_N^{-1} \mathbf{X}_N$$

Or

$$\mathbf{x}_N = \frac{1}{N} \mathbf{W}_N^* \mathbf{X}_N$$

$$\mathbf{W}_N^{-1} = \frac{1}{N} \mathbf{W}_N^* \qquad \mathbf{W}_N \mathbf{W}_N^* = N\mathbf{I}_N$$

I_N is an NxN identity matrix.

DISCRETE FOURIER TRANSFORM (DFT)

Let N-point vector x_N of the signal sequence x(n) , n = 0, 1, 2, ….(N-1) and N-point vector X_N of frequency samples and NxN matrix W_N as

$$\mathbf{x}_N = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}, \qquad \mathbf{X}_N = \begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix}$$

$$\mathbf{W}_N = \begin{bmatrix} 1 & 1 & 1 \cdots & 1 & \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots\vdots\vdots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}$$

EXAMPLE

Compute the DFT of the 4-point sequence

$x(n) = (0 \quad 1 \quad 2 \quad 3)$

$$\mathbf{W}_4 = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^0 & W_4^2 \\ 1 & W_4^3 & W_4^2 & W_4^1 \end{bmatrix} \qquad W_N = e^{-j2\pi/N}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

then

$$\mathbf{X}_4 = \mathbf{W}_4 \mathbf{x}_4 = \begin{bmatrix} 6 \\ -2+2j \\ -2 \\ -2-2j \end{bmatrix}$$

EXAMPLE

1. Given x(n) = {1, 1, 0, 0}, the DFT of this 4-point sequence can be computed using the matrix formulation as

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^1 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 1-j \\ 0 \\ 1+j \end{bmatrix},$$

where we used symmetry and periodicity properties given in Equations (6.16) and (6.17) to obtain $W_4^0 = W_4^4 = 1$, $W_4^1 = W_4^9 = -j$, $W_4^2 = W_4^6 = -1$, and $W_4^3 = j$. The IDFT can be computed as

$$\mathbf{x} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ 1 & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ 1 & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{bmatrix} \mathbf{X} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 2 \\ 1-j \\ 0 \\ 1+j \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}.$$

The DFT coefficients are equally spaced on the unit circle with frequency intervals of f_s/N (or $2\pi/N$). Therefore, the frequency resolution of the DFT is $\Delta = f_s/N$. The frequency sample $X(k)$ represents discrete frequency

DFT-Properties

1. Periodicity:

If x(n) and X(k) are N-point DFT pairs

$$x(n+N) = x(n) \quad \text{for all } n$$

$$X(k+N) = X(k) \quad \text{for all } k$$

2. Symmetry:

$$X(k) = X^*(-k)_N = X^*(N-k)$$

3. Linearity:

$$ax_1[n] + bx_2[n] \overset{DFT}{\leftrightarrow} aX_1[k] + bX_2[k]$$

DFT-Properties

4. Circular Shift of a sequence:

- The desired shift, called the **circular shift**, is defined using a modulo operation:

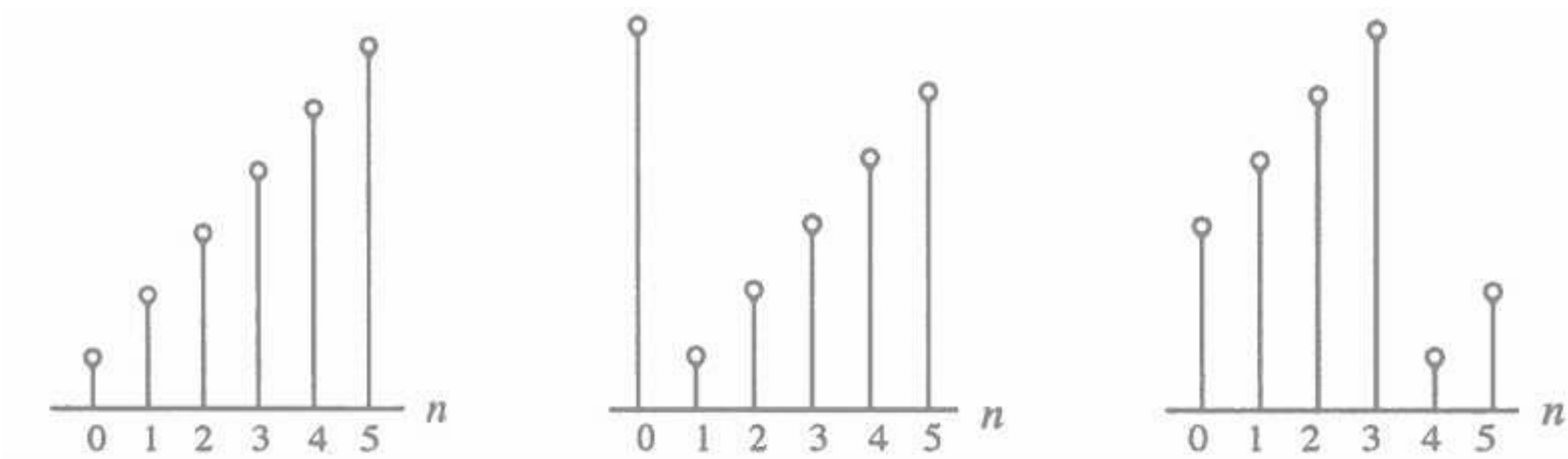
$$x_c[n] = x[\langle n - n_0 \rangle_N]$$

- For $n_0 > 0$ (right circular shift), the above equation implies

$$x_c[n] = \begin{cases} x[n - n_0], & for \;\; n_0 \le n \le N - 1 \\ x[N - n_0 + n], & for \;\; 0 \le n < n_0 \end{cases}$$

DFT-Properties

Concept of a circular shift



$x[n]$

$x[\langle n-1\rangle_6] = x[\langle n+5\rangle_6]$

$x[\langle n-4\rangle_6] = x[\langle n+2\rangle_6]$

DFT-Properties

5. Multiplication: $$DFT\,[x_1(n)\cdot x_2(n)] = \frac{1}{N} X_1(k) \otimes X_2(k)$$

6. Parseval's relation:

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

Energy spectrum $$\frac{|X(k)|^2}{N}$$

Power spectrum $$\left|\frac{X(k)}{N}\right|^2$$

DFT-Properties

7. Circular Convolution:

$$x_1(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X_1(k)$$

$$x_2(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X_2(k)$$

$$x_1(n) \text{ Ⓝ } x_2(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X_1(k)X_2(k)$$

DFT-Properties

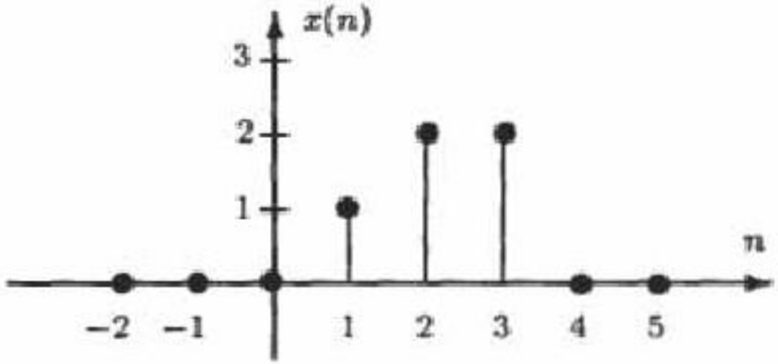
For the two finite-duration sequences of length N, $x_1(n)$ and $x_2(n)$

$$X_3(k) = X_1(k)X_2(k) \qquad k = 0, 1, \ldots, N-1$$

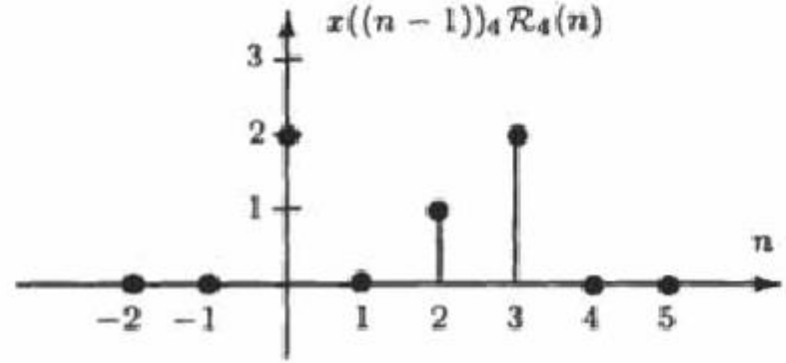
$$x_3(m) = \sum_{n=0}^{N-1} x_1(n)x_2((m-n))_N \qquad m = 0, 1, \ldots, N-1$$

Convolution Sum called Circular Convolution

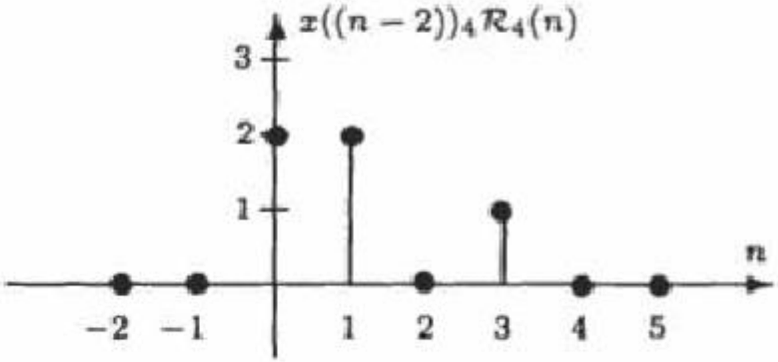
Example:



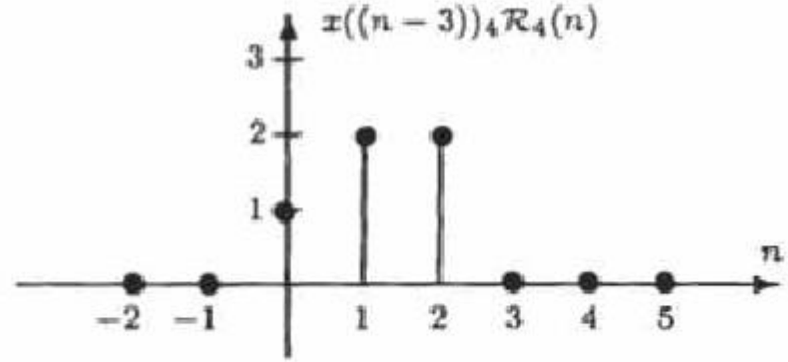
(a) A discrete-time signal of length $N = 4$.



(b) Circular shift by one.



(c) Circular shift by two.



(d) Circular shift by three.

Example:

Perform the circular convolution of the following 2 sequence:

$$x_1(n) = \{\underset{\uparrow}{2}, 1, 2, 1\}$$

$$x_2(n) = \{\underset{\uparrow}{1}, 2, 3, 4\}$$

m=0

$$x_3(0) = \sum_{n=0}^{3} x_1(n)x_2((-n))_N$$

= 14

m=2

$$x_3(2) = \sum_{n=0}^{3} x_1(n)x_2((2-n))_4$$

= 14

m=1

$$x_3(1) = \sum_{n=0}^{3} x_1(n)x_2((1-n))_4$$

= 16

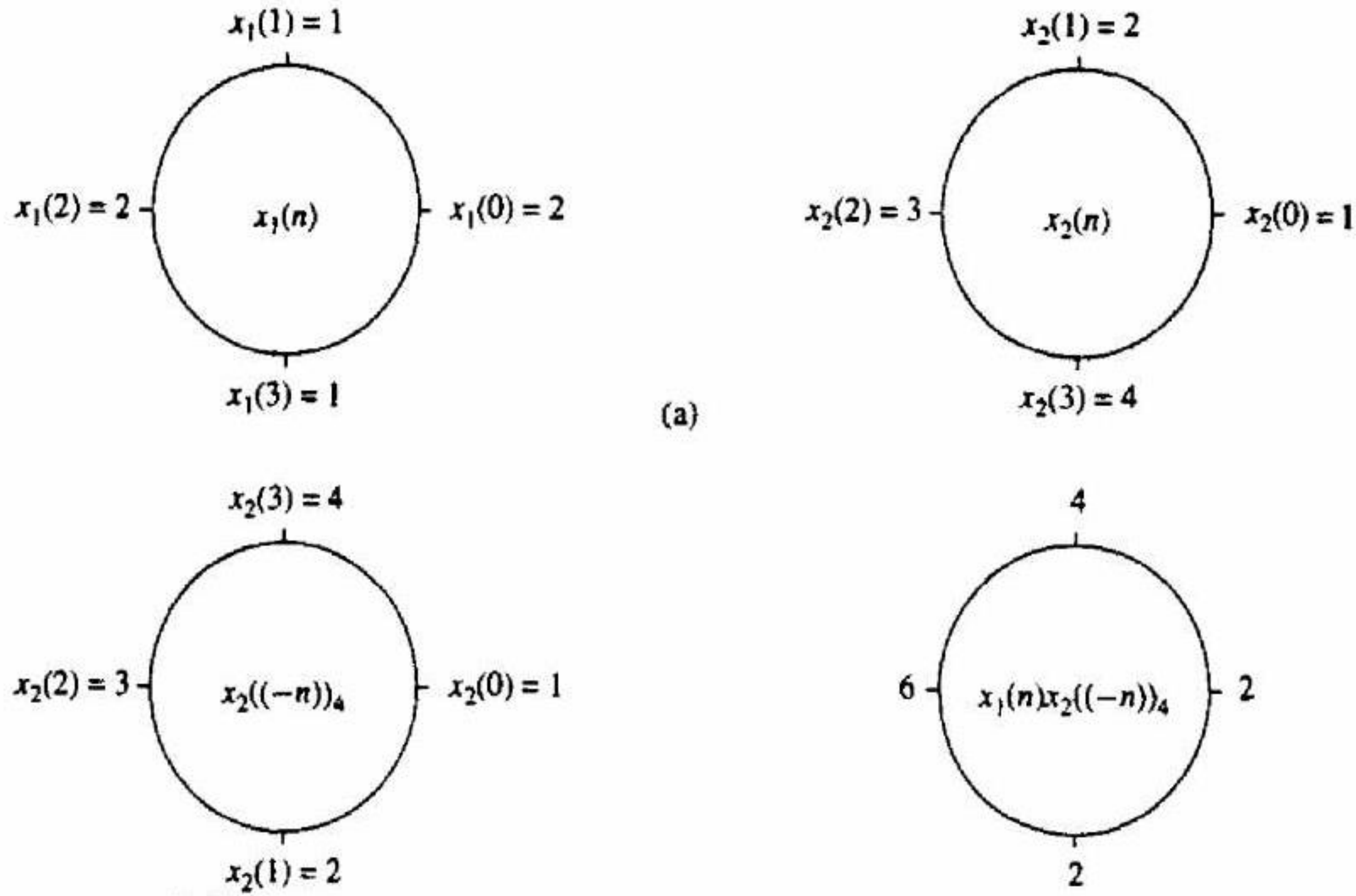
m=3

$$x_3(3) = \sum_{n=0}^{3} x_1(n)x_2((3-n))_4$$

= 16

$$\boxed{x_3(n) = \{\underset{\uparrow}{14}, 16, 14, 16)}$$

Continued…

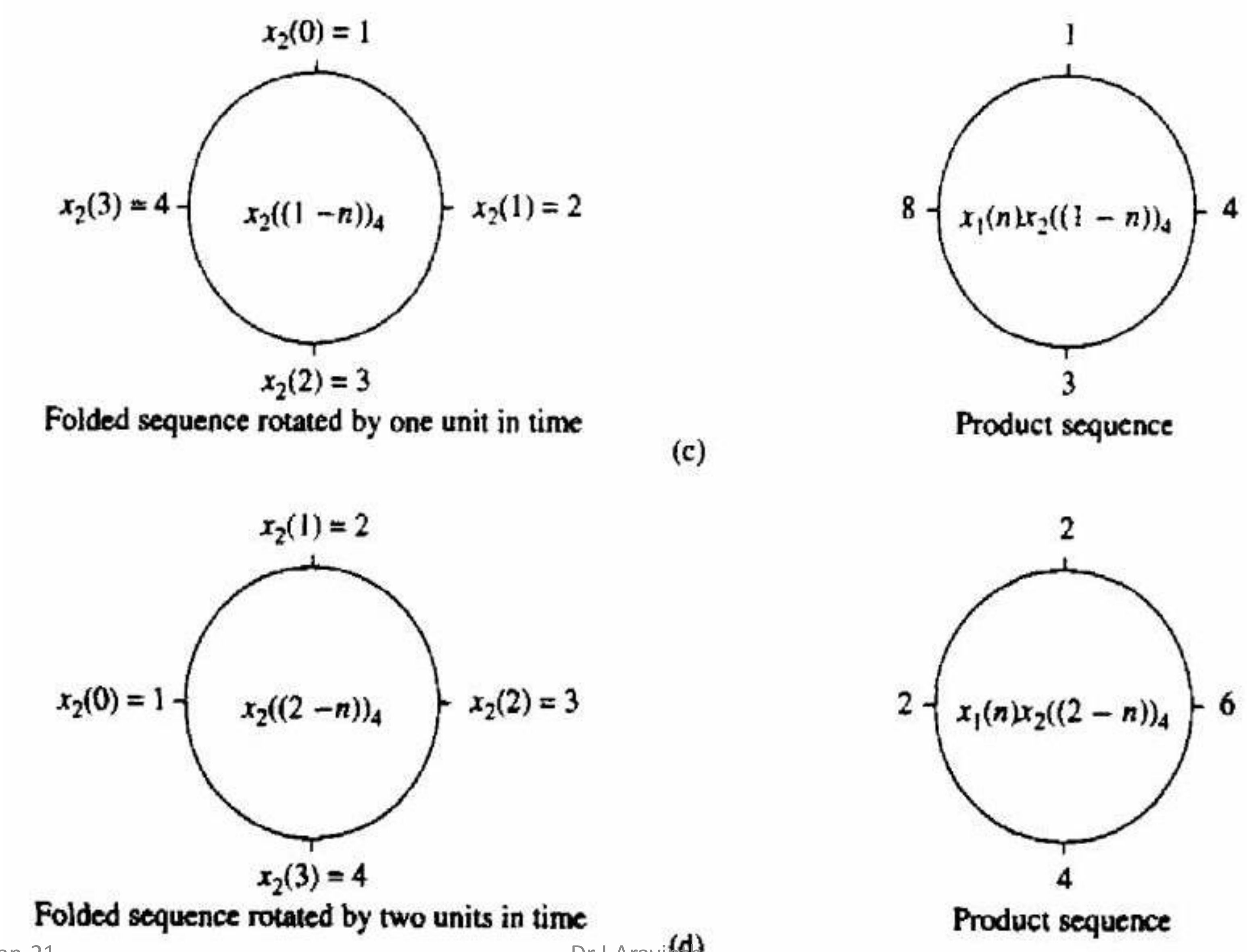


Folded sequence

Product sequence

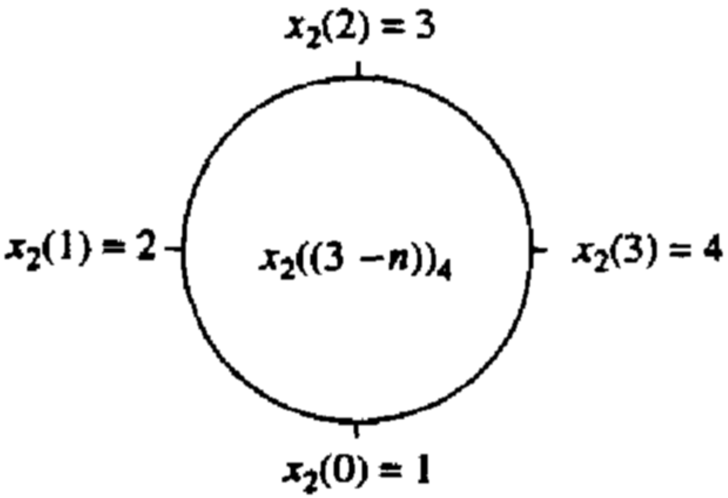
(b)

Continued…

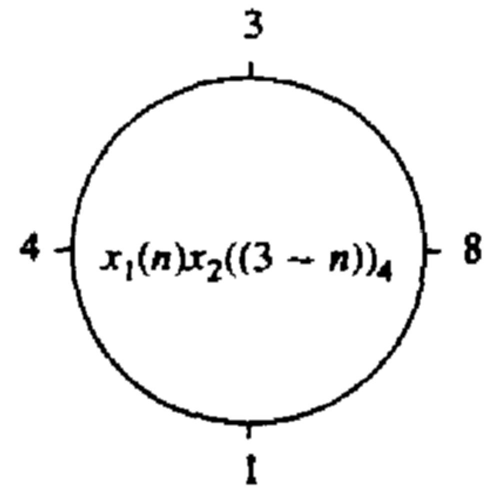


$x_2(0) = 1$
$x_2(3) = 4$
$x_2((1-n))_4$
$x_2(1) = 2$
$x_2(2) = 3$
Folded sequence rotated by one unit in time
1
8
$x_1(n)x_2((1-n))_4$
4
3
Product sequence
(c)
$x_2(1) = 2$
$x_2(0) = 1$
$x_2((2-n))_4$
$x_2(2) = 3$
$x_2(3) = 4$
Folded sequence rotated by two units in time
2
2
$x_1(n)x_2((2-n))_4$
6
4
Product sequence
(d)

Continued…



Folded sequence rotated by three units in time



Product sequence

(e)

Linear filtering methods based on the DFT

- DFT provides a discrete frequency representation of a finite duration sequence in the frequency domain.
- Computational tool for linear system analysis, especially for linear filtering.
- DFT can be used to perform linear filtering in the frequency domain.

Linear filtering methods based on the DFT

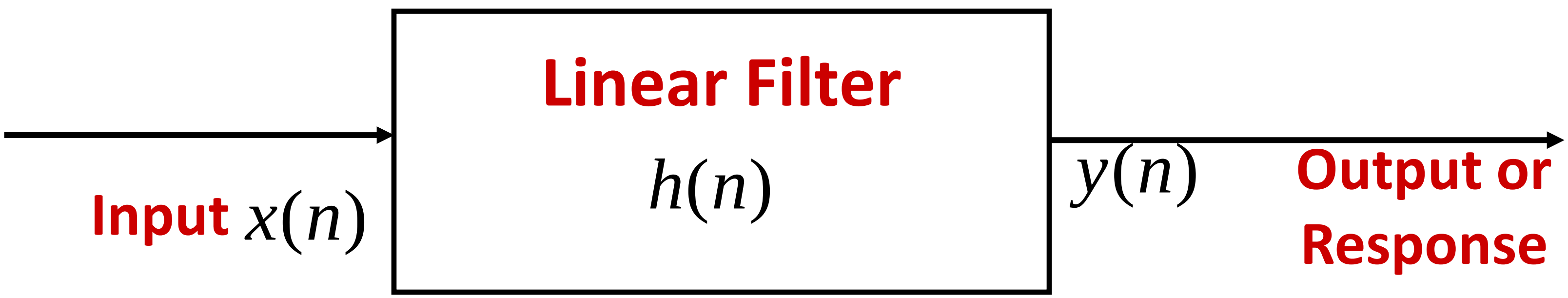
1.Use of the DFT in Linear Filtering

2.Filtering of long data sequence

- **Overlap-save method**
- **Overlap-add method**

Use of the DFT in Linear Filtering

- Our objective is to determine the output of a linear filter to a given input sequence.



- Linear Convolution
- By using DFT and IDFT
- Circular Convolution

Linear Convolution

- A sequence $x(n)$ of length L filtered by an FIR filter $h(n)$ of length M

$$y(n) = \sum_{k=0}^{M-1} h(n)x(n-k)$$

Example $x(n)$ =[1, 2, 2, 1] , $h(n)$ =[1, 2, 3]
L=4,M=3
x(n)=[1 ,2,2,1] ,h(n)=[1, 2,3]

The length of sequence y(n) is N=L+M-1=6

y(n)={1,4,9,11,8,3}

Find the convolution between **X** and **H** (or find **Y**) for the system shown below:

X= [1, 2, 3] ⟶ **H= [4, 5]** ⟶ **Y = [4, 13, 22, 15]**

Solution:

Multiplication	4	5
1	4	5
2	8	10
3	12	15

4

13

22

15

Addition

DFT and IDFT

- Therefore ,a DFT of size N $\geq$ L+M-1 is required to represent y(n) in the frequency domain.
- Using the DFT notation

$$Y(k) = H(k)X(k), k = 0,1,........(N-1)$$

X(*k*) and *H*(*k*) are the DFT (with zero padding) of *x*(*n*) and *h*(*n*), respectively.

- Performing the inverse DFT of Y(K)

$$y(n) = IDFT(Y(k)$$

By means of the DFT and IDFT, determine the response of the FIR filter with impulse response

$$h(n) = \{\underset{\uparrow}{1}, 2, 3\}$$

to the input sequence

$$x(n) = \{\underset{\uparrow}{1}, 2, 2, 1\}$$

$$X(k) = \sum_{n=0}^{7} x(n)e^{-j2\pi kn/8}$$

$$= 1 + 2e^{-j\pi k/4} + 2e^{-j\pi k/2} + e^{-j3\pi k/4}, \qquad k = 0, 1, \ldots, 7$$

- L=4,M=3,N=L+M-1=6
- Eight point DFT of x(n) is
- X(K)= {6,1.707 - 4.121j, -1 – j,0.2929 - j0.121, 0,0.292 + j0.121, -1+j,1.707 + j4.121}
- Eight point DFT of h(n) is
- H(K)={6,2.414 - j4.414, -2-j2,-0.4142+ j1.585,2,-0.414 -j 1.585,-2+j2,2.414 + j4.4142}
- Y(k)=X(k).H(K)={36,-14.0-j17.48,j4,0.07+j0.515,0,0.07-j0.515,-j4,-14.07+j17.48}
- Eight point IDFT of Y(K) is

$$y(n) = \sum_{k=0}^{7} Y(k)e^{j2\pi kn/8}, \qquad n = 0, 1, \ldots, 7$$

$$y(n) = \{\underset{\uparrow}{1}, 4, 9, 11, 8, 3, 0, 0\}$$

Circular Convolution

- Example $x(n)$ =[1, 2, 2, 1] , $h(n)$ =[1, 2, 3]
- L=4,M=3
- The length of sequence y(n) is N=L+M-1=6
- Example $x(n)$=[1 ,2,2,1,0 ,0] ,$h(n)$=[1, 2,3,0,0 0]

- y(n)={1,4,9,11,8,3}

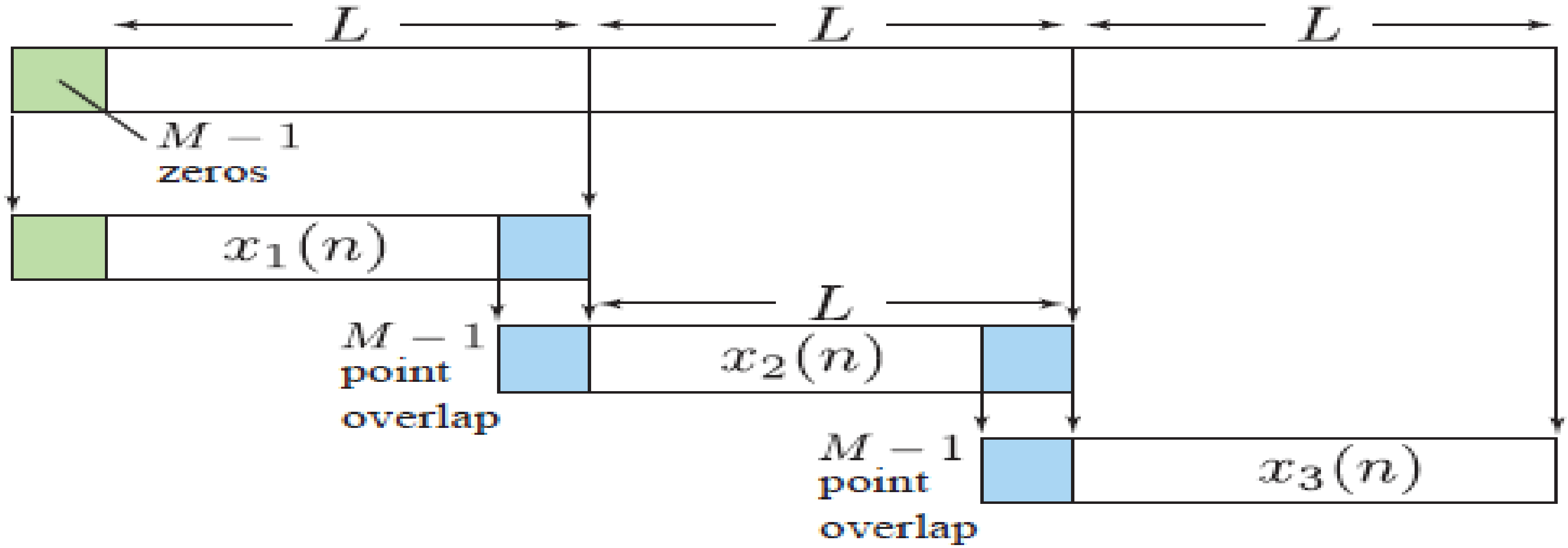
Filtering of Long Data Sequences

- In practical applications involving linear filtering of signals, the input sequence x(n) is often a very long sequence.
- Real-time signal processing applications concerned with signal monitoring and analysis.
- **Overlap-save method**
- **Overlap-add method**

Filtering of Long Data Sequences

- When the DFT is used to implement linear filtering, a signal is processed in blocks. Due to the real-time requirement (low delay) and the limitation of physical memory, the size of the block can not be arbitrarily large.
- The length of the FIR filter is M and the length of on block of data is L ($L>M$)
- Each time a block of data of length $L+M-1$ is filtered by using the DFT method.

Input signal blocks:



$$x_1(n) = \{\underbrace{0, \quad 0, \quad \ldots \quad 0}_{M-1 \text{ zeros}}, \; x(0), x(1), \ldots, x(L-1)\}$$

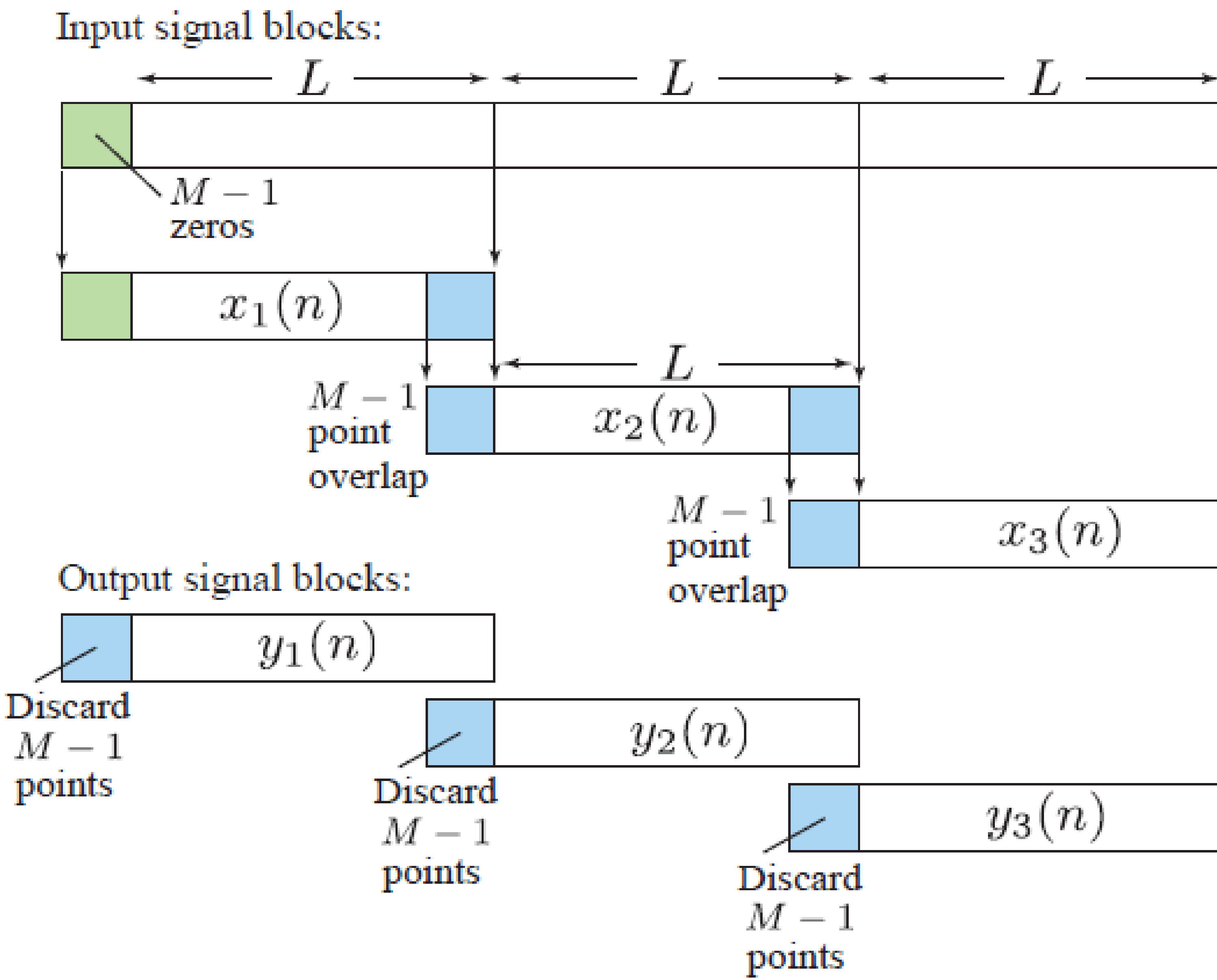
$$x_2(n) = \{\underbrace{x(L-M+1), \ldots x(L-1)}_{\text{last } M-1 \text{ points from } x_1(n)}, x(L), \ldots, x(2L-1)\}$$

$$x_3(n) = \{\underbrace{x(2L-M+1), \ldots x(2L-1)}_{\text{last } M-1 \text{ points from } x_2(n)}, x(2L), \ldots, x(3L-1)\}$$

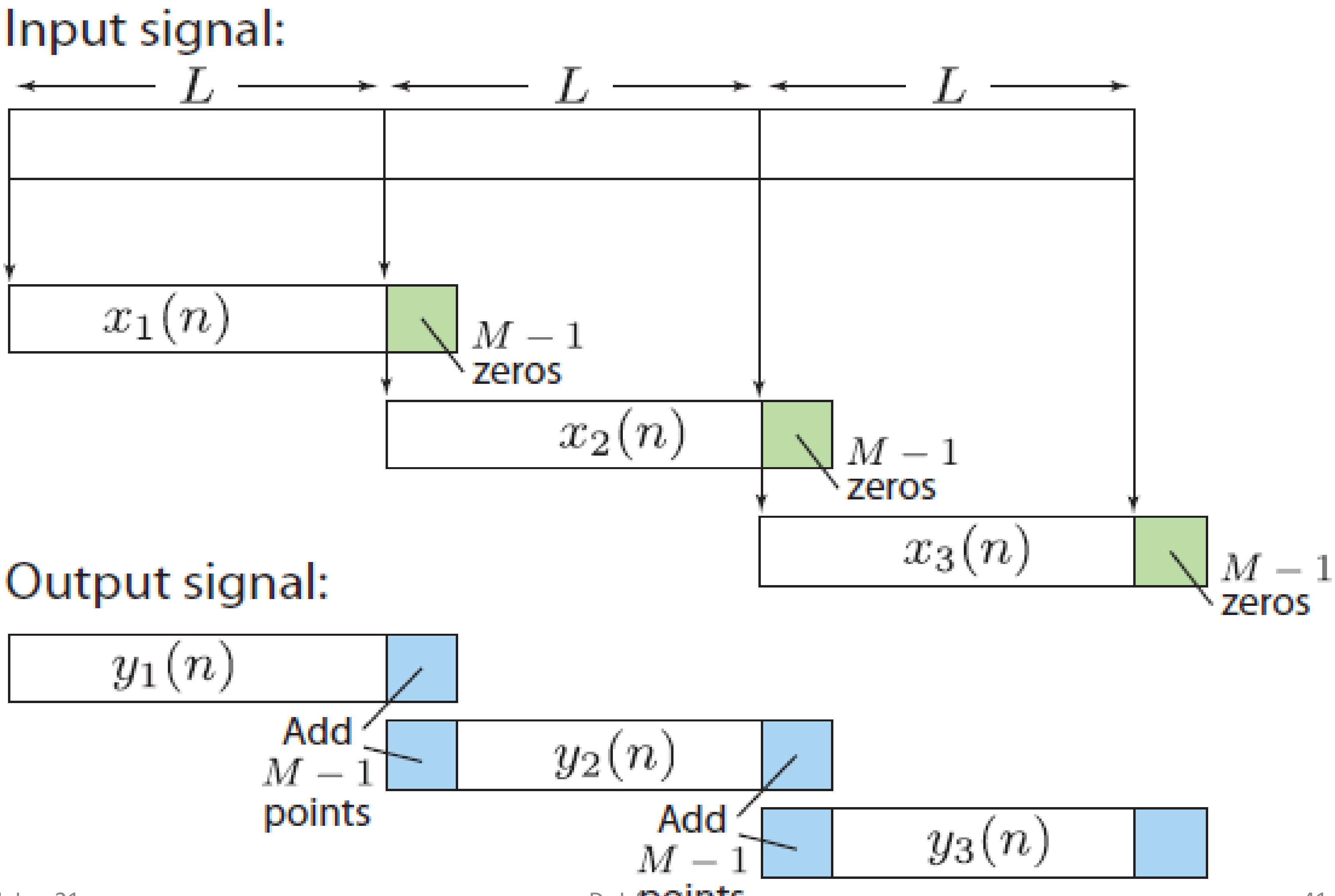
$$\vdots$$

The last $M-1$ points from the previous input block must be saved for use in the current input block.

$y_i(n)=x_i(n) \textcircled{N} h(n)$


Input signal blocks:
L
L
L
M − 1
zeros
x1(n)
L
M − 1
point
overlap
x2(n)
M − 1
point
overlap
x3(n)
Output signal blocks:
y1(n)
Discard
M − 1
points
y2(n)
Discard
M − 1
points
y3(n)
Discard
M − 1
points

Overlap-Add Method



For example ……

- Consider

x(n)={3 9 1 2 3 4 5 6 3 4 5 6 7 8 9 8 7 5 } and h(n)= {1 2 1 1}

let **L=5** and **M=4**, hence **N=L+M-1=8**

therefore,

$$x_1(n)=\{\underbrace{0, 0, 0,}_{\text{M-1 zeros}}\ \underbrace{3, 9, 1, 2, 3}_{\substack{\text{first L points}\\ \text{from input seq.}}}\}$$

$$x_3(n)=\{\underbrace{6, 3, 4,}_{\substack{\text{last M-1 pts}\\ \text{from } x_2(n)}}\ \underbrace{5, 6, 7, 8, 9}_{\substack{\text{next L points}\\ \text{from input seq.}}}\}$$

$$x_2(n)=\{\underbrace{1, 2, 3,}_{\substack{\text{last M-1 pts}\\ \text{from } x_1(n)}}\ \underbrace{4, 5, 6, 3, 4}_{\substack{\text{next L points}\\ \text{from input seq.}}}\}$$

$$x_4(n)=\{\underbrace{7, 8, 9,}_{\substack{\text{last M-1 pts}\\ \text{from } x_3(n)}}\ \underbrace{8, 7, 5, 0, 0}_{\substack{\text{next L points}\\ \text{from input seq.}}}\}$$

$$y_i(n)=x_i(n) \text{ \textcircled{N} } h(n)$$

$$y_1(n)=\{\underbrace{9\ \ 5\ \ 3}_{\text{Discard M-1 pts}}\ \overbrace{3\ \ 15\ \ 22\ \ 16\ \ 17}^{\text{Send to output}}\}$$

$$y_2(n)=\{\underbrace{18\ 11\ 12}_{\text{Discard M-1 pts}}\ \overbrace{13\ 18\ 23\ 24\ 21}^{\text{Send to output}}\}$$

$$y_3(n)=\{\underbrace{12\ 22\ 32}_{\text{Discard M-1 pts}}\ \overbrace{41\ 40\ 36\ 25\ 12}^{\text{Send to output}}\}$$

$$y_4(n)=\{\underbrace{39\ 32\ 25}_{\text{Discard M-1 pts}}\ \overbrace{22\ 23\ 28\ 33\ 38}^{\text{Send to output}}\}$$

$$y(n)=\{3\ 15\ 22\ 16\ 17\ 13\ 18\ 23\ 24\ 21\ 41\ 40\ 36\ 25\ 12\ 22\ 23\ 28\ 33\ 38\}$$

$x(n)=\{3,-1,0,1,3,2,0,1,2,1\}, h(n)=\{1,1,1\}$

- Overlap-Save Method:
- Let L=5,M=3,N=L+M-1=7

$$x_3(m) = x_1(n) \otimes x_2(n) = \sum_{n=0}^{N-1} x_1(n)x_2((n-m))_N$$

$x_1(n)=\{0,0,3,-1,0,1\}$, $x_3(n)=\{0,1,2,1,0,0\}$

$x_2(n)=\{0,1,3,2,0,1\}$

Perform $y_1(n)=x_1(n)$ Ⓝ $h(n)$

$$\boxed{y_1(m) = x_1(n) \otimes h(n) = \sum_{n=0}^{N-1} x_1(n)h((n-m))_N}$$

$y_2(n)=x_2(n)$ Ⓝ $h(n)$

$$\boxed{y_2(m) = x_2(n) \otimes h(n) = \sum_{n=0}^{N-1} x_2(n)h((n-m))_N}$$

$y_3(n)=x_3(n)$ Ⓝ $h(n)$

$$y_3(m) = x_3(n) \otimes h(n) = \sum_{n=0}^{N-1} x_3(n)h((n-m))_N$$

$x(n)=\{3,-1,0,1,3,2,0,1,2,1\}, h(n)=\{1,1,1\}$

- Overlap-Add Method:
- Let L=5, M=3, N=L+M-1=7 $x_3(m) = x_1(n) \otimes x_2(n) = \sum_{n=0}^{N-1} x_1(n) x_2((n-m))_N$

$x_1(n)=\{3,-1,0,1,0,0\}$, $x_3(n)=\{2,1,0,0,0,0\}$

$x_2(n)=\{3,2,0,1,0,0\}$

Perform $y_1(n)=x_1(n)$ Ⓝ $h(n)$ $\boxed{y_1(m) = x_1(n) \otimes h(n) = \sum_{n=0}^{N-1} x_1(n) h((n-m))_N}$

$y_2(n)=x_2(n)$ Ⓝ $h(n)$ $\boxed{y_2(m) = x_2(n) \otimes h(n) = \sum_{n=0}^{N-1} x_2(n) h((n-m))_N}$

$y_3(n)=x_3(n)$ Ⓝ $h(n)$ $y_3(m) = x_3(n) \otimes h(n) = \sum_{n=0}^{N-1} x_3(n) h((n-m))_N$

- $x(n) \xleftrightarrow{\text{DFT}} X(k) \xleftrightarrow{\text{IDFT}} x(n)$
- Properties of DFT

  Circular Convolution

$$x_3(m) = x_1(n) \otimes x_2(n) = \sum_{n=0}^{N-1} x_1(n) x_2((n-m))_N \xleftrightarrow{\text{DFT}} X_3(K) = X_1(K) X_2(K)$$

- Linear Filtering Methods Based on the DFT

1.Use of the DFT in Linear Filtering

- By using DFT and IDFT

$$x(n) \xleftrightarrow{\text{DFT}} X(k) \rightarrow \boxed{H(K)} \rightarrow Y(K) = X(K).H(K) \xrightarrow{\text{IDFT}} y(n)$$

- Linear Convolution
- Circular Convolution

2.Filtering of Long Data Sequence

- **Overlap-save method**
- **Overlap-add method**

COMPUTATIONAL COMPLEXITY

- How ever its implementation in so easy for

$$X(k) = \sum_{n=0}^{N-1} x(n).e^{\frac{-j2\pi kn}{N}} \qquad for...k = 0,1,...,N-1$$

An implementation of DFT involves TWO basic functional blocks,

1. ADDITION

2. MULTIPLICATION

Length of the Sequence (N)	DFT-Complex Function		DFT-Real Function	
	No.of Complex MULs (N*N)	No.of Complex ADDs (N*N-1)	No.of Real MULs 4(N*N)	No.of Real ADDs 4(N*N-1)
2	4	2	16	8
3	9	6	36	24
4	16	12	64	48
:	:	:	:	:
8	64	56	256	224
:	:	:	:	:
16	256	240	1024	960
:	:	:	:	:

LIMITATIONS OF DFT

- **When DFTs are used to process a CT-Signal by sampling, several potential error sources may be important**

  - **Aliasing, Spectral Leakage**

  - **Under suitable restrictions, the DFT closely approximates the spectrum of CT-signal at a Discrete set of frequencies.**

Why do we need FFT?

- DFT properties and its use in system analysis in the numerical computation of long sequences is prohibitively time-consuming.

- The importance of DFT and IDFT in practical applications is due to a large extent on the existences of computationally efficient algorithms, collectively known as FFT algorithms.

- Therefore several algorithms have been developed to efficiently compute the DFT. These are collectively called Fast Fourier transform (or FFT) algorithms.

FAST FOURIER TRANSFORM (FFT)

- FFT is introduced by Cooley-Tukey, (almost a half century ago) is playing historically sustained significant role in the development of DSP.

- It is widely used ―fast algorithm‖ to solve many engineering challenges, designing filters, performing spectral analysis, estimation, noise cancellation and benchmark testing devices and systems, etc.

- It is also very readily useable for computing the inverse transforms.

COMPARISON OF DFT WITH FFT

Length of the Sequence (N)	DFT		FFT	
	No.of Complex MULs (N*N)	No.of Complex ADDs (N*N-1)	No.of Complex MULs $\frac{N}{2}.\log_2^N$	No.of Complex ADDs $N.\log_2^N$
2	4	2	1	2
4	16	12	4	8
8	64	56	12	24
16	256	240	32	64
:	:	:	:	:
128	16384	16256	448	896
:	:	:	:	:
1024	1048576	1047552	5120	10240

FFT PERFORMANCE

- **I. Speed calculation of FFT:**

$$FFT - Speed = \frac{\text{No.of Complex MULs required in direct DFT}}{\text{No.of Complex MULs required in FFT}}$$

- For N=8; $$FFT - Speed = \frac{N * N}{\frac{N}{2}.\log_2^N}$$

$$\frac{64}{12} = 5.33$$

FFT is 5.33 times faster than direct DFT

FFT PERFORMANCE

- **I. Percentage of Computation saved in FFT:**

$$Computation.\% - Saved = 100 - \left\{ \frac{\text{No.of Complex MULs required in FFT}}{\text{No.of Complex MULs required in direct DFT}} \right\} * 100$$

$$Computation.\% - Saved = 100 - \left\{ \frac{\frac{N}{2}.\log_2^N}{N * N} \right\} * 100$$

- For N=8;

$$Computation.\% - Saved = 100 - \left\{ \frac{12}{64} \right\} * 100 \qquad = 81.25\%$$

- For N=16;

$$Computation.\% - Saved = 100 - \left\{ \frac{32}{256} \right\} * 100 \qquad = 87.5\%$$

Fast Fourier Transform

- A large amount of work has been devoted to reducing the computation time of a DFT.
- This has led to efficient algorithms which are known as the Fast Fourier Transform (FFT) algorithms.
- Decimation In Time(DIT) Radix-2 Algorithm
- Decimation In Frequency(DIF) Radix-2 Algorithm

Fast Fourier Transform(FFT)

$$X(k)=\sum_{n=0}^{N-1}x[n]e^{-j\frac{2\pi}{N}nk}$$

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

$$x(n)=\frac{1}{N}\sum_{n=0}^{N-1}X[k]W_N^{-nk}$$

$$e^{-j\frac{2\pi}{N}}=W_N\,(TwiddleFactor(or)PhaseFactor)$$

$$W_N^{k+\frac{N}{2}}=e^{-j\frac{2\pi}{N}k}e^{-j\frac{2\pi}{N}\frac{N}{2}}=e^{-j\frac{2\pi}{N}k}e^{-j\pi}=-e^{-j\frac{2\pi}{N}k}=-W_N^k,\ Symmetry\ Property$$

$$W_{\frac{N}{2}}^{k+\frac{N}{2}}=e^{-j\frac{2\pi}{N/2}k}e^{-j\frac{2\pi}{N/2}\frac{N}{2}}=e^{-j\frac{2\pi}{N/2}k}=W_{\frac{N}{2}}^k,\ Periodicity\ Property$$

N-Point DFT	DFT		FFT	
	N(N-1) complex '+'	N^2 complex '×'	N/2 $\log_2$(N) complex '×'.	N $\log_2$(N) complex '+'.
8	56	64	12	24
16	240	256	32	64
32	992	1024	80	160
64	4032	4096	192	384
128	16256	16384	448	896

DIT Radix-2 FFT Algorithm

- Let us consider the computation of the N-Point DFT

$\mathrm{N} = \mathrm{r}^{\mathrm{m}}$,r-Radix, m- No.of Stages

First Step: x[n] = x[0], x[1], …, x[N-1]

Split the N-point data sequences into two N/2-point data sequences $f_1(n)$ and $f_2(n)$

Lets divide the sequence x[n] into even and odd Sequences

$$\left.\begin{aligned} f_1(n) &= \mathbf{x[2n]} = \mathbf{x[0], x[2], \ldots, x[N-2]} \quad ; n = 0,1,\ldots\left(\frac{N}{2}-1\right) \\ f_2(n) &= \mathbf{x[2n+1]} = \mathbf{x[1], x[3], \ldots, x[N-1]} \quad ; n = 0,1,\ldots\left(\frac{N}{2}-1\right) \end{aligned}\right\} --(1)$$

$$X(k)=\sum_{n=0}^{N-1} x[n] W_N^{nk}; \quad 0 \le k \le N-1$$ N-point DFT

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

$$W_N^{2nk} = e^{-j\frac{2\pi}{N}2nk} = e^{-j\frac{2\pi}{N/2}nk} = W_{\frac{N}{2}}^{nk}$$

$$W_N^{(2n+1)k} = W_N^k \cdot W_{\frac{N}{2}}^{nk}$$

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

$$=\sum_{n=0}^{\frac{N}{2}-1} f_1(n) W_{\frac{N}{2}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} f_2(n) W_{\frac{N}{2}}^{nk}$$

$$X(k)=F_1(k)+W_N^k F_2(k); \quad k=0,\ldots(N-1)---------(2)$$

- **Where F1(k) and F2(k)** N/2-point DFTs

$$F_1(k)=\sum_{n=0}^{\frac{N}{2}-1} f_1(n)W_{\frac{N}{2}}^{nk}$$

$$F_2(k)=\sum_{n=0}^{\frac{N}{2}-1} f_2(n)W_{\frac{N}{2}}^{nk}$$

- Since **F1(k+N/2)= F1(k) and F2(k+N/2)= F2(k),** $W_N^{k+N/2}=-W_N^k$

$$X(k)=F_1(k)+W_N^kF_2(k);\quad k=0,\ldots\left(\frac{N}{2}-1\right)\quad ------(3.1)$$

$$X\left(k+\frac{N}{2}\right)=F_1(k)-W_N^kF_2(k);\quad k=0,\ldots\left(\frac{N}{2}-1\right)\quad ------(3.2)$$

Second Step: Split the N/2-point data sequences into two N/4-point sequences

- $v_{11}(n) = f_1(2n) \quad ; n = 0,1,\ldots\left(\frac{N}{4} - 1\right)$
- $v_{12}(n) = f_1(2n+1)$--------------- (4.1)
- $v_{21}(n) = f_2(2n) \quad ; n = 0,1,\ldots\left(\frac{N}{4} - 1\right)$
- $v_{22}(n) = f_2(2n+1)$----------------(4.2)

$$F_1(k) = \sum_{n=0}^{\frac{N}{2}-1} f_1(n) W_{\frac{N}{2}}^{nk}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} f_1(2n) W_{\frac{N}{2}}^{2nk} + \sum_{n=0}^{\frac{N}{4}-1} f_1(2n+1) W_{\frac{N}{2}}^{(2n+1)k}$$

$$= \sum_{n=0}^{\frac{N}{4}-1} v_{11}(n) W_{\frac{N}{4}}^{nk} + W_{\frac{N}{2}}^{k} \sum_{n=0}^{\frac{N}{4}-1} v_{12}(n) W_{\frac{N}{4}}^{nk} = V_{11}(k) + W_{\frac{N}{2}}^{k} V_{12}(k) --- (5)$$

$$F_1(k) = V_{11}(k) + W_{\frac{N}{2}}^{k} V_{12}(k); k = 0,1,\ldots\ldots\ldots,(\frac{N}{4}-1)$$

$$F_1\left(k+\frac{N}{4}\right) = V_{11}(k) - W_{\frac{N}{2}}^{k} V_{12}(k) --- (6.1)$$

$$F_2(k) = V_{21}(k) + W_{\frac{N}{2}}^{k} V_{22}(k); k = 0,1,\ldots\ldots\ldots,(\frac{N}{4}-1)$$

$$F_2\left(k+\frac{N}{4}\right) = V_{21}(k) - W_{\frac{N}{2}}^{k} V_{22}(k) --- (6.2)$$

- Where $V_{11}(k)$,$V_{12}(k)$,$V_{21}(k)$ and $V_{22}(k)$
- N/4 point DFT of the sequences $v_{11}(n)$,$v_{12}(n)$,$v_{21}(n)$ and $v_{22}(n)$

$$V_{11}(k) = \sum_{n=0}^{\frac{N}{4}-1} v_{11}(n) W_{\frac{N}{4}}^{nk}; V_{12}(k) = \sum_{n=0}^{\frac{N}{4}-1} v_{12}(n) W_{\frac{N}{4}}^{nk}$$

$$V_{21}(k) = \sum_{n=0}^{\frac{N}{4}-1} v_{21}(n) W_{\frac{N}{4}}^{nk}; V_{22}(k) = \sum_{n=0}^{\frac{N}{4}-1} v_{22}(n) W_{\frac{N}{4}}^{nk}$$

- N=8, $N = 2^3$ - Three Stages
- Four 2-point(N/4) DFT's --->Two 4-point(N/2) DFT's --->One 8-point (N)DFT

- $v_{11}(0) = f_1(0) = x(0)$
- $v_{11}(1) = f_1(2) = x(4)$
- $v_{12}(0) = f_1(1) = x(2)$
- $v_{12}(1) = f_1(3) = x(6)$
- $v_{21}(0) = f_2(0) = x(1)$
- $v_{21}(1) = f_2(2) = x(5)$
- $v_{22}(0) = f_2(1) = x(3)$
- $v_{22}(1) = f_2(3) = x(7)$

v11(n) =f1(2n)

v12(n) =f1(2n+1)

v21(n) =f2(2n)

v22(n) =f2(2n+1) ; n=0,1,…(N/4-1)

$f_1(n)$ =x[2n]

$f_2(n)$ =x[2n+1] ; n=0,1,…(N/2-1)

$$Bit\,Reversal - Shuffling\,of\,Data$$

$$x[0] = x[000] \leftrightarrow x[000] = x[0]$$
$$x[1] = x[001] \leftrightarrow x[100] = x[4]$$
$$x[2] = x[010] \leftrightarrow x[010] = x[2]$$
$$x[3] = x[011] \leftrightarrow x[110] = x[6]$$
$$x[4] = x[100] \leftrightarrow x[001] = x[1]$$
$$x[5] = x[101] \leftrightarrow x[101] = x[3]$$
$$x[6] = x[110] \leftrightarrow x[011] = x[5]$$
$$x[7] = x[111] \leftrightarrow x[111] = x[7]$$

- **First Stage**-Four 2-point(N/4) DFT's

$$V_{11}(k)=\sum_{n=0}^{\frac{N}{4}-1} v_{11}(n)W_{\frac{N}{4}}^{nk}=\sum_{n=0}^{\frac{8}{4}-1} v_{11}(n)W_{\frac{8}{4}}^{nk}=\sum_{n=0}^{1} v_{11}(n)W_{2}^{nk}$$

$$=v_{11}(0)W_2^{(0)k}+v_{11}(1)W_2^{(1)k}$$

$$k=0; V_{11}(0)=v_{11}(0)W_2^{(0)0}+v_{11}(1)W_2^{(1)0}$$

$$=v_{11}(0)+W_2^{0}v_{11}(1)=x(0)+W_8^{0}x(4)$$

$$k=1; V_{11}(1)=v_{11}(0)W_2^{(0)1}+v_{11}(1)W_2^{(1)1}$$

$$=v_{11}(0)+v_{11}(1)W_2^{1}=x(0)-W_8^{0}x(4)$$

$$V_{12}(0)=x(2)+W_8^{0}x(6)$$

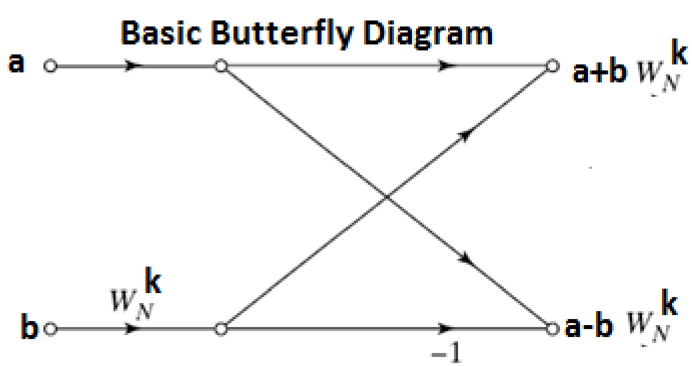
$$V_{12}(1)=x(2)-W_8^{0}x(6)$$

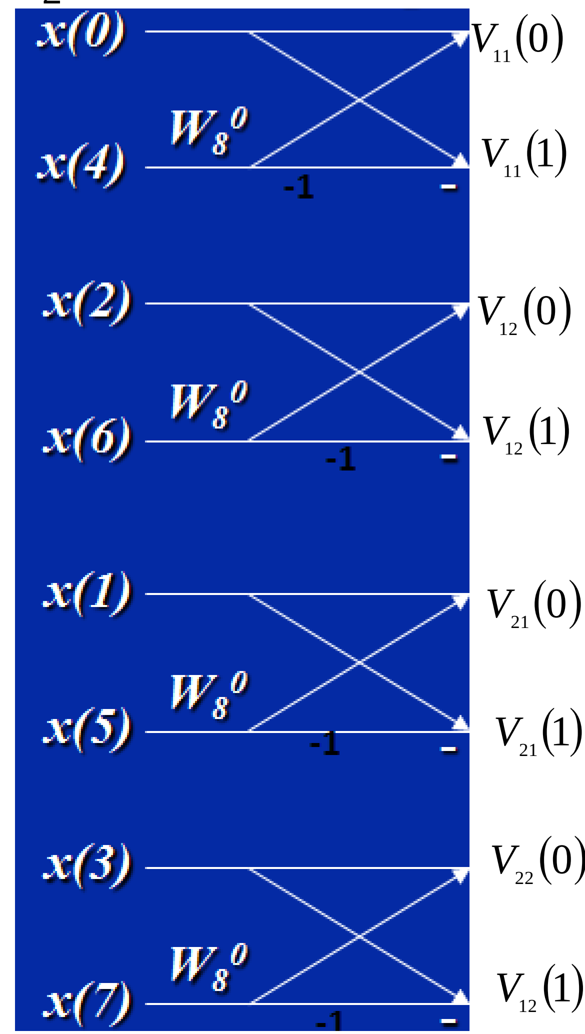
$$V_{21}(0)=x(1)+W_8^{0}x(5)$$

$$V_{21}(1)=x(1)-W_8^{0}x(5)$$

$$V_{22}(0)=x(3)+W_8^{0}x(7)$$

$$V_{22}(1)=x(3)-W_8^{0}x(7)$$





$$W_8^0=e^{-j\frac{2\pi(0)}{8}}=1$$

Second Stage-Two 4-point(N/2) DFT's

$$F_1(k)=V_{11}(k)+W_{\frac{N}{2}}^{k}V_{12}(k)=V_{11}(k)+W_{\frac{8}{2}}^{k}V_{12}(k)$$

$$k=0;F_1(0)=V_{11}(0)+W_4^0V_{12}(0)=V_{11}(0)+W_8^0V_{12}(0)$$

$$k=1;F_1(1)=V_{11}(1)+W_4^1V_{12}(1)=V_{11}(1)+W_8^2V_{12}(1)$$

$$F_1\left(k+\frac{N}{4}\right)=V_{11}(k)-W_{\frac{N}{2}}^{k}V_{12}(k)\rightarrow F_1\left(k+\frac{8}{4}\right)=V_{11}(k)-W_{\frac{8}{2}}^{k}V_{12}(k)$$

$$k=0;F_1(2)=V_{11}(0)-W_4^0V_{12}(k)=V_{11}(0)-W_8^0V_{12}(k)$$

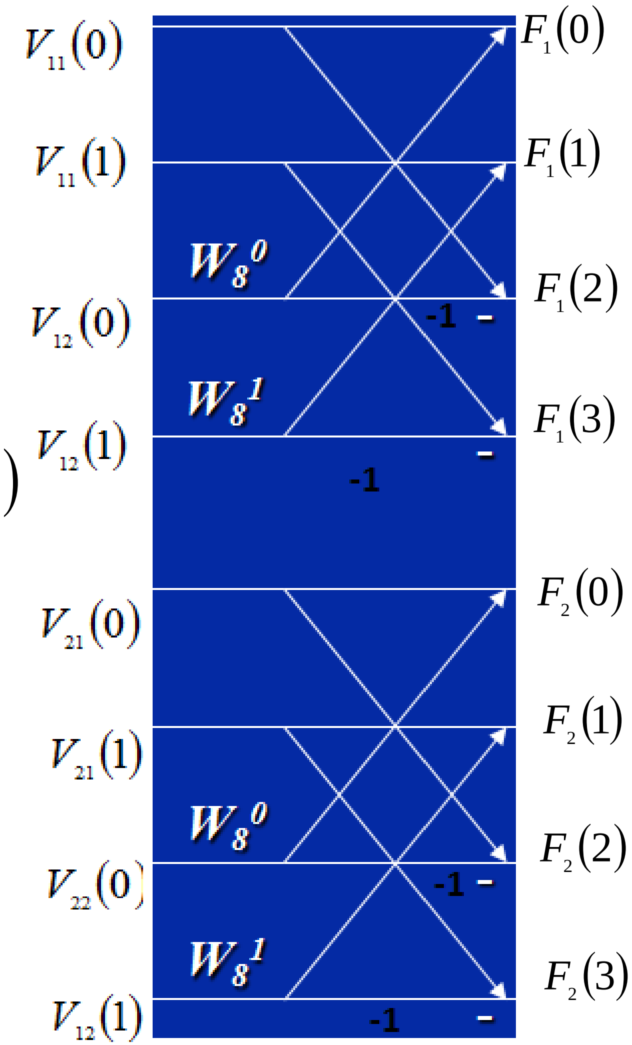
$$k=1;F_1(3)=V_{11}(1)-W_4^1V_{12}(1)=V_{11}(0)-W_8^2V_{12}(k)$$

$$k=0;F_2(0)=V_{21}(0)+W_4^0V_{22}(0)=V_{21}(0)+W_8^0V_{22}(0)$$

$$k=1;F_2(1)=V_{21}(1)+W_4^1V_{22}(1)=V_{21}(1)+W_8^2V_{22}(1)$$

$$k=0;F_2(2)=V_{21}(0)-W_4^0V_{22}(0)=V_{21}(0)-W_8^0V_{22}(0)$$

$$k=1;F_2(3)=V_{21}(1)-W_4^1V_{22}(1)=V_{21}(1)-W_8^2V_{22}(1)$$



$$W_8^0=e^{-j\frac{2\pi(0)}{8}}=1$$

$$W_8^2=e^{-j\frac{2\pi(2)}{8}}=e^{-j\frac{\pi}{2}}=-j$$

Third Stage-One 8-point(N) DFT

$$X(k) = F_1(k) + W_N^k F_2(k)$$

$$k = 0; X(0) = F_1(0) + W_8^0 F_2(0)$$

$$k = 1; X(1) = F_1(1) + W_8^1 F_2(1)$$

$$k = 2; X(2) = F_1(2) + W_8^2 F_2(2)$$

$$k = 3; X(3) = F_1(3) + W_8^3 F_2(3)$$

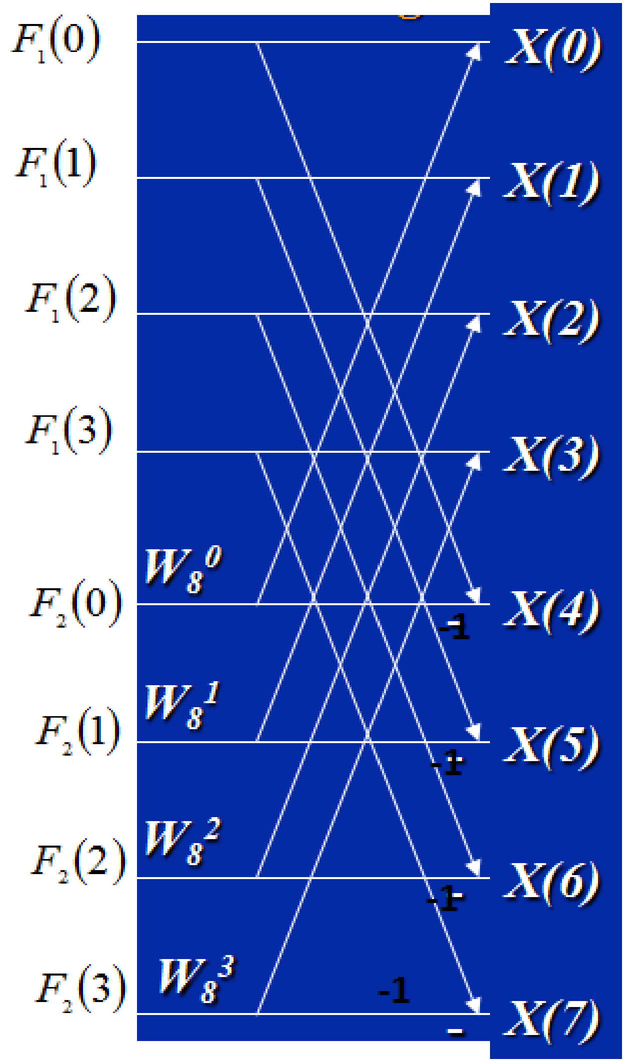
$$X\left(k + \frac{N}{2}\right) = F_1(k) - W_N^k F_2(k)$$

$$k = 0; X(4) = F_1(0) - W_8^0 F_2(0)$$

$$k = 1; X(5) = F_1(1) - W_8^1 F_2(1)$$

$$k = 2; X(6) = F_1(2) + W_8^2 F_2(2)$$

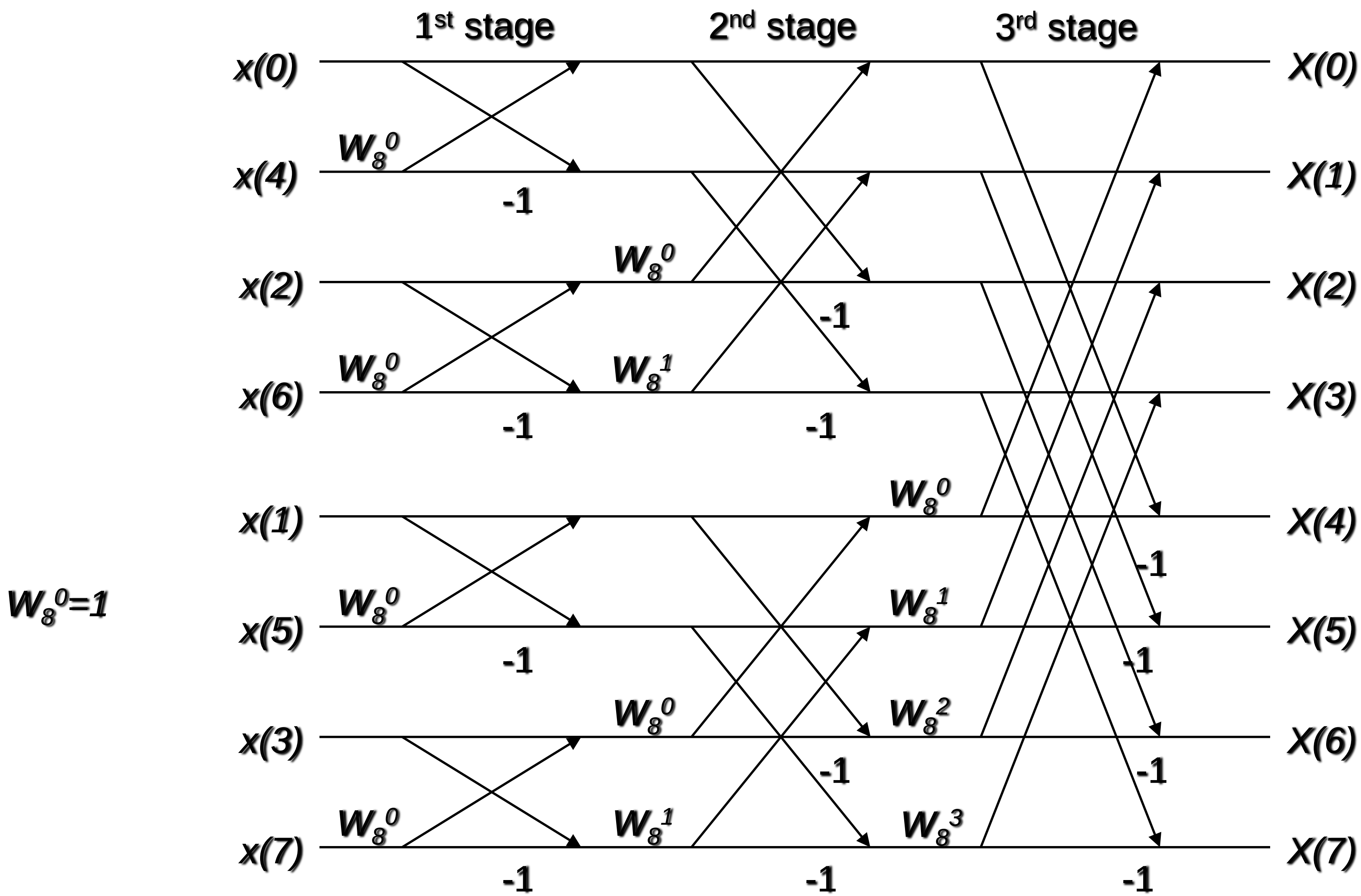
$$k = 3; X(7) = F_1(3) + W_8^3 F_2(3)$$



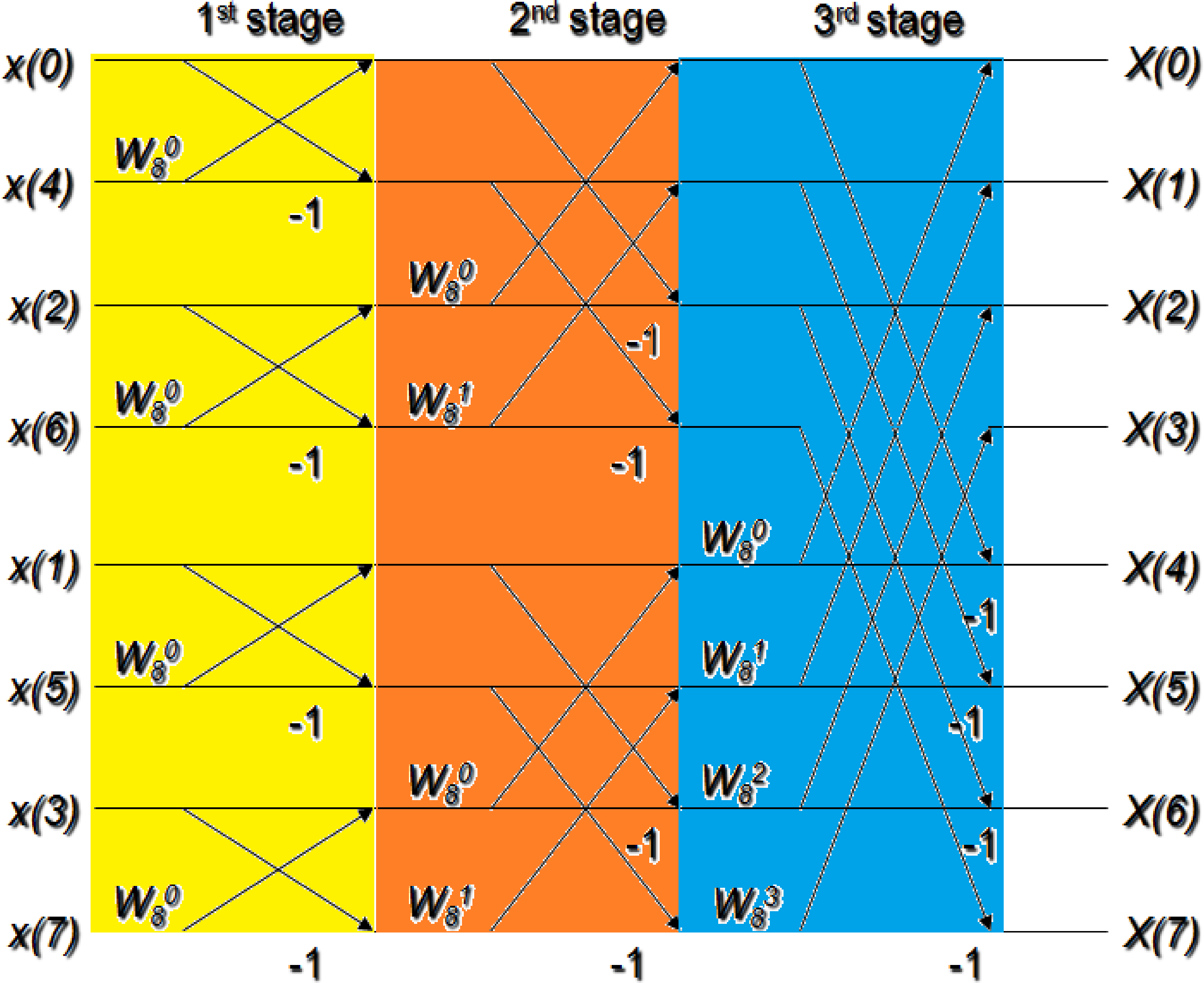
$$W_8^0 = e^{-j\frac{2\pi(0)}{8}} = 1$$

$$W_8^1 = e^{-j\frac{2\pi(1)}{8}} = e^{-j\frac{\pi}{4}} = 0707 - j0.707$$

$$W_8^2 = -j; W_8^3 = e^{-j\frac{2\pi(3)}{8}} = e^{-j\frac{3\pi}{4}} = -0707 - j0.707$$

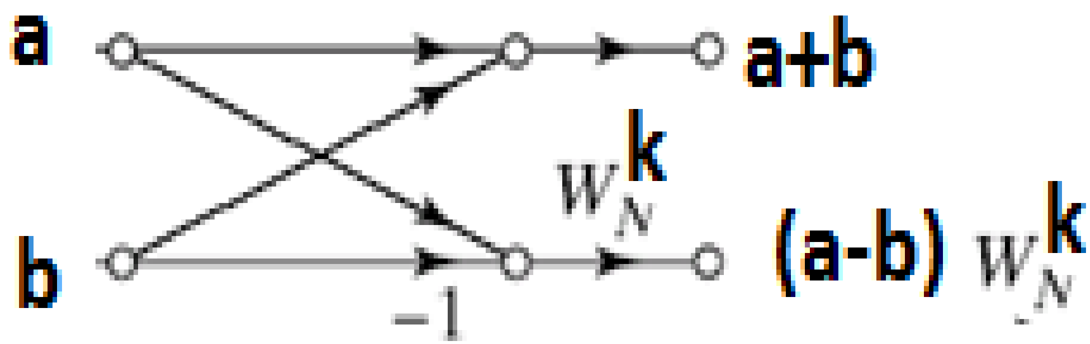


1st stage
2nd stage
3rd stage
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
X(0)
X(1)
X(2)
X(3)
X(4)
X(5)
X(6)
X(7)
W_8^0
W_8^1
W_8^2
W_8^3
-1
$W_8^0=1$

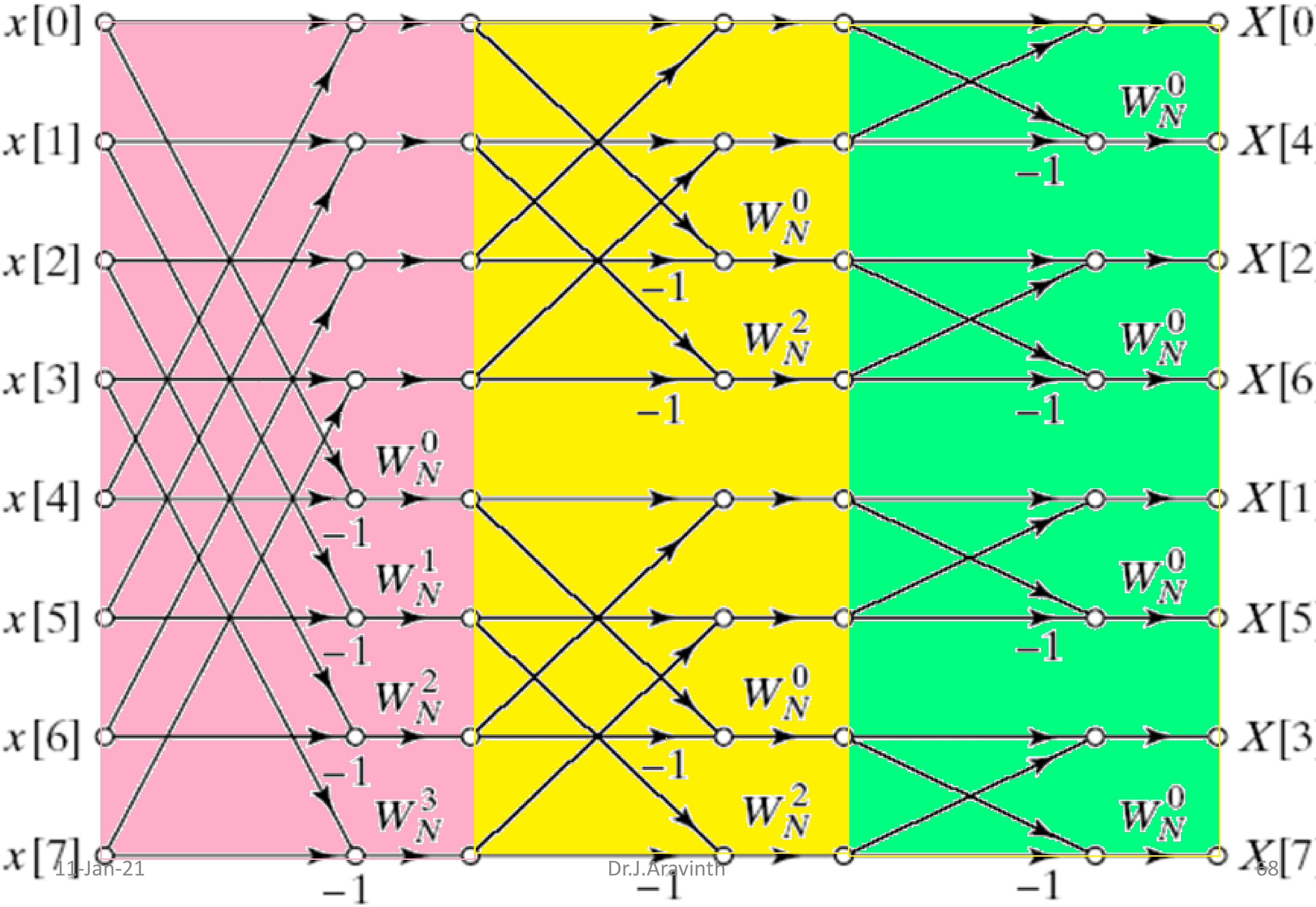

1st stage
2nd stage
3rd stage
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
X(0)
X(1)
X(2)
X(3)
X(4)
X(5)
X(6)
X(7)
W_8^0
W_8^1
W_8^2
W_8^3
-1

Decimation In Frequency(DIF) Radix-2 Algorithm

- Frequency Domain sequence is decimated
- Butterfly Diagram



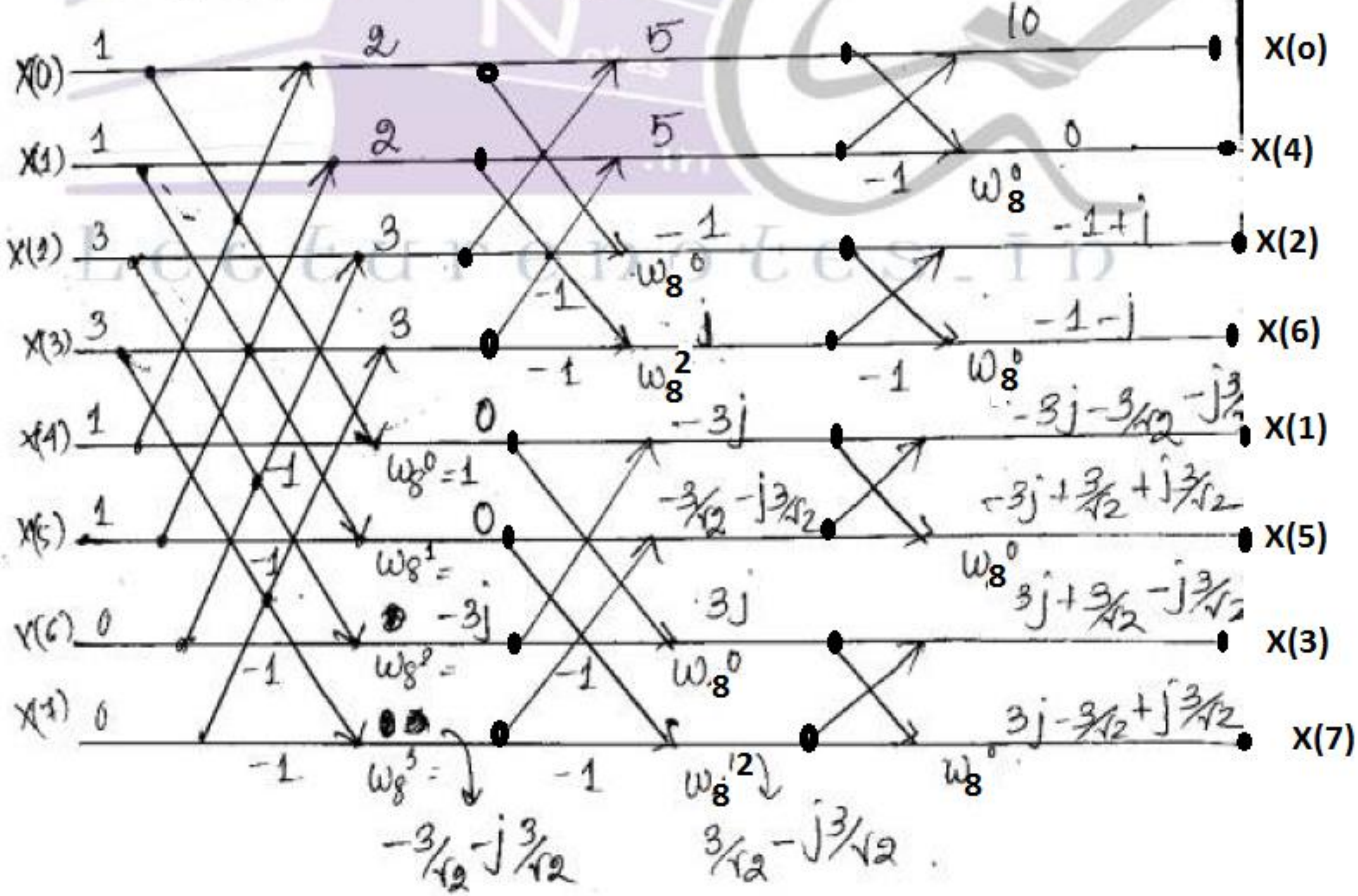
Decimation In Frequency(DIF) Radix-2 Algorithm

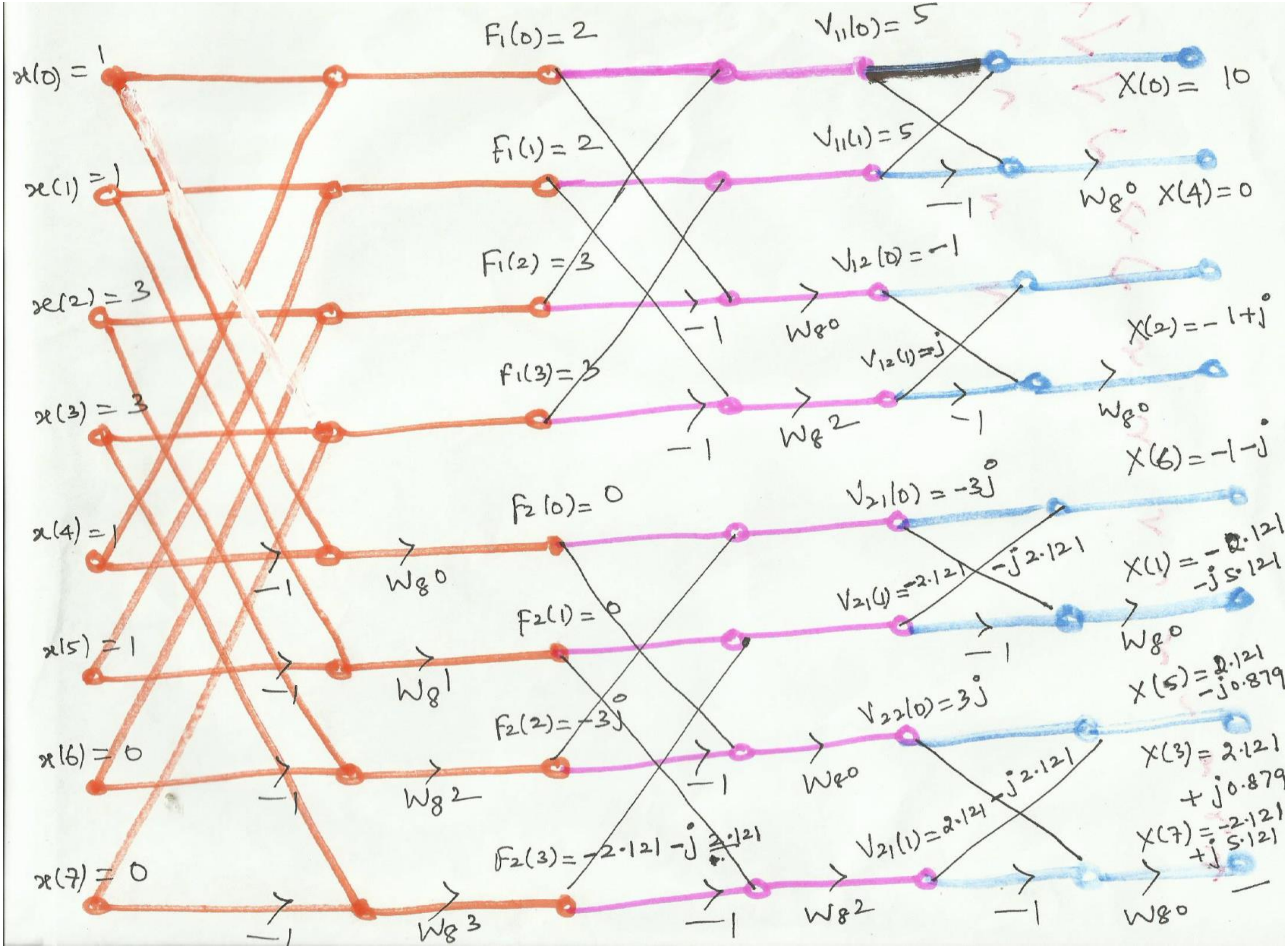


Zoom in (Ctrl+Plus)

Q) $x(n) = \{1, 1, 3, 3, 1, 1, 0, 0\}$.

Find its DFT through DIF-FFT





x(0) = 1
x(1) = 1
x(2) = 3
x(3) = 3
x(4) = 1
x(5) = 1
x(6) = 0
x(7) = 0
-1
-1
-1
-1
W8 0
W8 1
W8 2
W8 3
F1(0) = 2
F1(1) = 2
F1(2) = 3
F1(3) = 3
F2(0) = 0
F2(1) = 0
F2(2) = -3j
F2(3) = -2.121 - j 2.121
-1
-1
-1
-1
W8 0
W8 2
W8 0
W8 2
V11(0) = 5
V11(1) = 5
V12(0) = -1
V12(1) = j
V21(0) = -3j
V21(1) = -2.121 - j 2.121
V22(0) = 3j
V21(1) = 2.121 - j 2.121
-1
-1
-1
-1
W8 0
W8 0
W8 0
W8 0
X(0) = 10
X(4) = 0
X(2) = -1 + j
X(6) = -1 - j
X(1) = -2.121 - j 5.121
X(5) = 2.121 - j 0.879
X(3) = 2.121 + j 0.879
X(7) = -2.121 + j 5.121

$$F_1(0) = x(0) + x(4) = 1 + 1 = 2$$

$$F_1(1) = x(1) + x(5) = 1 + 1 = 2$$

$$F_1(2) = x(2) + x(6) = 3 + 0 = 3$$

$$F_1(3) = x(3) + x(7) = 3 + 0 = 3$$

$$F_2(0) = [x(0) - x(4)]\, W_8^0 = [1-1] \cdot 1 = 0$$

$$F_2(1) = [x(1) - x(5)]\, W_8^1 = [1-1][0.707 - j0.707] = 0$$

$$F_2(2) = [x(2) - x(6)]\, W_8^0 = [3-0](-j) = -3j$$

$$F_2(3) = [x(2) - x(6)]\, W_8^2 = [3-0][0.707 - j0.707]$$

$$= -2.121 - j2.121$$

$$V_{11}(0) = F_1(0) + F_1(2) = 2+3 = 5$$

$$V_{11}(1) = F_1(1) + F_1(3) = 2+3 = 5$$

$$V_{12}(0) = [F_1(0) - F_1(2)]\, W_8^0 = [2-3]\cdot 1 = -1$$

$$V_{12}(1) = [F_1(1) - F_1(3)]\, W_8^2 = [2-3](-j) = j$$

$$V_{21}(0) = F_2(0) + F_2(2) = 0 + (-3j) = -3j$$

$$V_{21}(1) = F_2(1) + F_2(3) = 0 + (-2.121 - j2.121)$$

$$= -2.121 - j\,2.121$$

$$V_{22}(0) = [F_2(0) - F_2(2)]\, W_8^0 = [0 - (-3j)]\cdot 1 = 3j$$

$$V_{22}(1) = [F_2(1) - F_2(3)]\, W_8^2 = [0 - (2.121 - j\,2.121)](-j)$$

$$= +2.121 + j\,2.121$$

$$X(0) = V_{11}(0) + V_{11}(1) = 5 + 5 = 10$$

$$X(4) = [V_{11}(0) - V_{11}(1)] W_8^0 = [5-5] \cdot 1 = 0$$

$$X(2) = V_{12}(0) + V_{12}(1) = -1 + j$$

$$X(6) = [V_{12}(0) - V_{12}(1)] W_8^0 = -1 - j$$

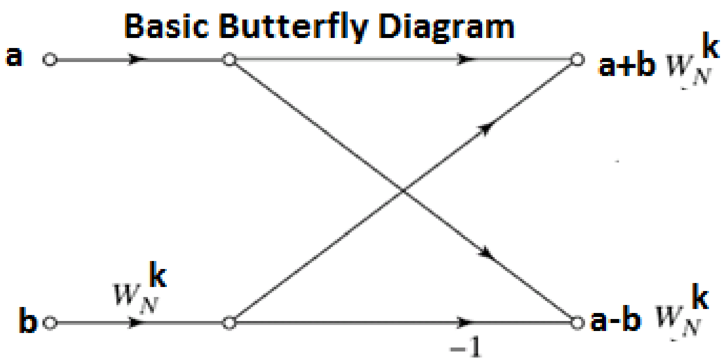
$$X(1) = V_{21}(0) + V_{21}(1) = -3j + (-2.121 - j2.121)$$
$$= -2.121 + j5.121$$

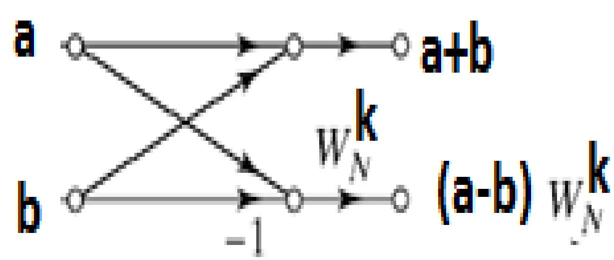
$$X(5) = [V_{21}(0) - V_{21}(1)] W_8^0 = [-3j - (-2.121 - j2.121)] \cdot 1$$
$$= -3j + 2.121 + j2.121$$
$$= 2.121 - j0.879$$

$$X(3) = V_{22}(0) + V_{22}(1) = 3j + 2.121 - j2.121$$
$$= 2.121 + j0.879$$

$$X(7) = [V_{22}(0) - V_{22}(1)] W_8^0 = [3j - (2.121 - j2.121)] \cdot 1$$
$$= -2.121 + j5.121$$

DIT ! DIF

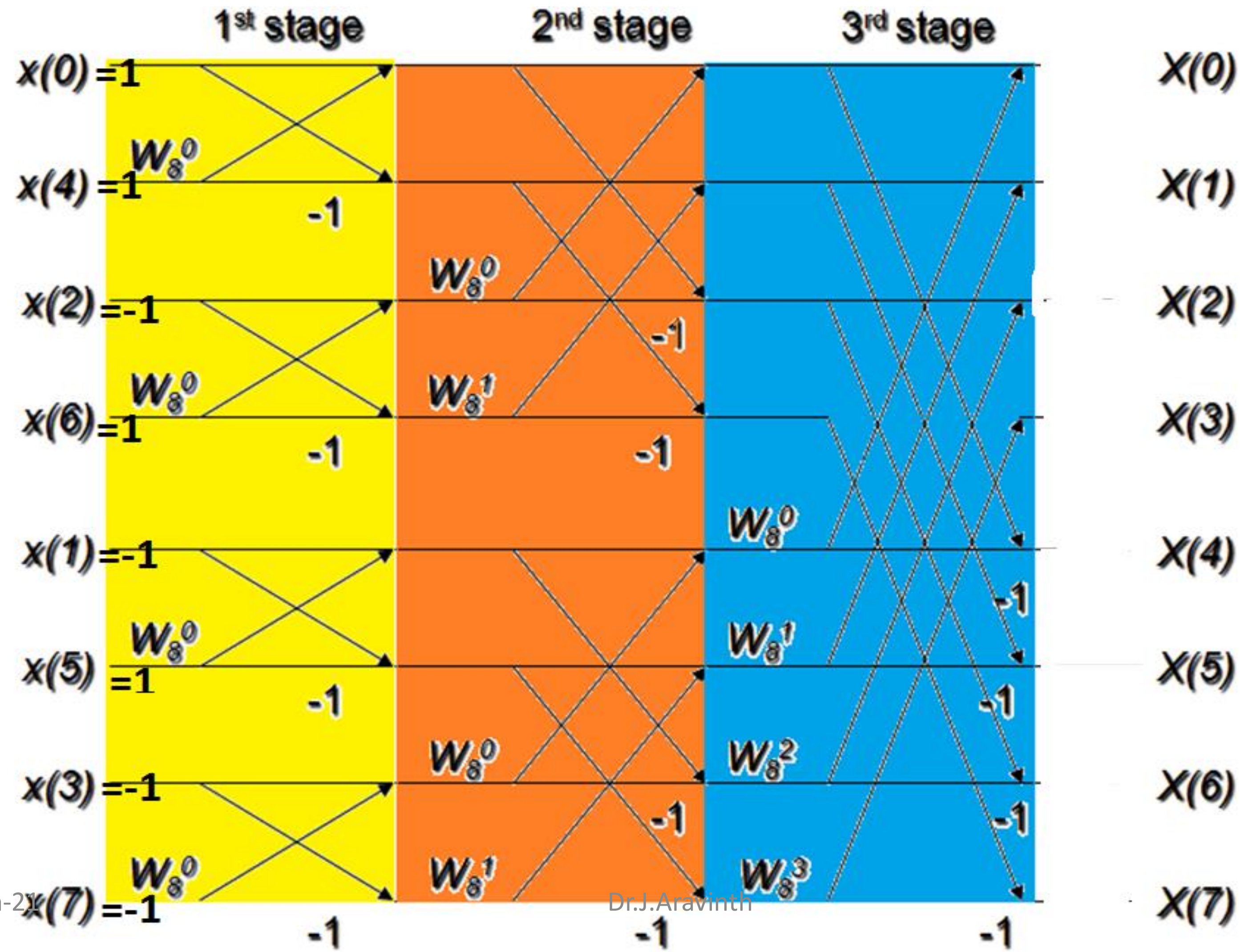




DIT	DIF
• **Time Domain Sequence** decimated	**Frequency Domain Sequence decimated**
• Input Sequence is bit **Reversal order**	**Normal order**
• Output sequence is **Normal order**	**Bit-reversal order**
• Phase factor is Multiplied **Before addition & subtraction**	**After subtraction**

Problems

- x(n)={1,-1,-1,-1,1,1,1,-1}



- **First Stage**

$$V_{11}(0) = x(0) + W_8^{0} x(4) = 1 + 1.1 = 2$$

$$V_{11}(1) == x(0) - W_8^{0} x(4) = 1 - 1.1 = 0$$

$$V_{12}(0) = x(2) + W_8^{0} x(6) = -1 + 1.1 = 0$$

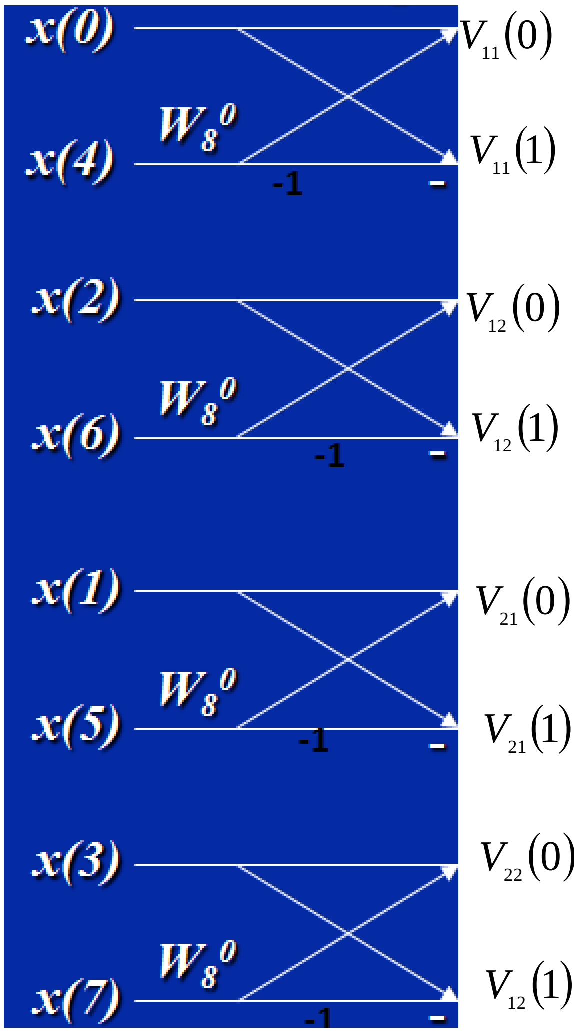
$$V_{12}(1) = x(2) - W_8^{0} x(6) = -1 - 1.1 = -2$$

$$V_{21}(0) = x(1) + W_8^{0} x(5) = -1 + 1.1 = 0$$

$$V_{21}(1) = x(1) - W_8^{0} x(5) = -1 - 1.1 = -2$$

$$V_{22}(0) = x(3) + W_8^{0} x(7) = -1 + 1.(-1) = -2$$

$$V_{22}(1) = x(3) - W_8^{0} x(7) = -1 - 1.(-1) = 0$$



$$W_8^{0} = e^{-j\frac{2\pi(0)}{8}} = 1$$

Second Stage

$$F_1(0)=V_{11}(0)+W_8^0V_{12}(0)=2+1.0=2$$

$$F_1(1)=V_{11}(1)+W_8^2V_{12}(1)=0+(-j)(-2)=2j$$

$$F_1(2)=V_{11}(0)-W_8^0V_{12}(k)=2-1.0=2$$

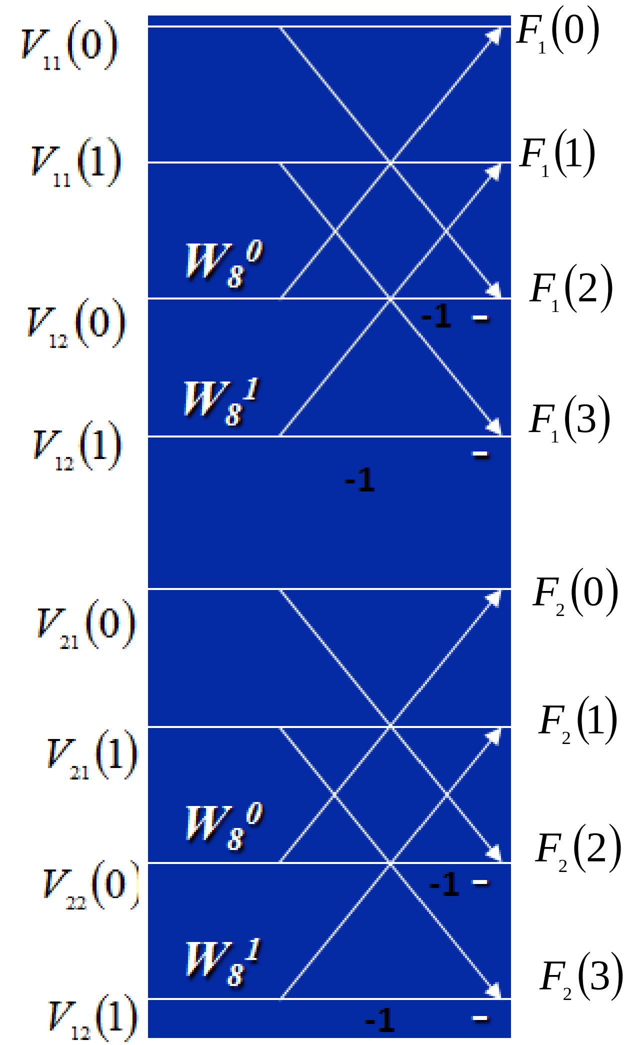
$$F_1(3)=V_{11}(0)-W_8^2V_{12}(k)=0-(-j)(-2)=-2j$$

$$F_2(0)=V_{21}(0)+W_8^0V_{22}(0)=0+1.(-2)=-2$$

$$F_2(1)=V_{21}(1)+W_8^2V_{22}(1)=-2+(-j).0=-2$$

$$F_2(2)=V_{21}(0)-W_8^0V_{22}(0)=0-1(-2)=2$$

$$F_2(3)=V_{21}(1)-W_8^2V_{22}(1)=-2-(-j).0=-2$$



$$W_8^0=e^{-j\frac{2\pi(0)}{8}}=1$$

$$W_8^2=e^{-j\frac{2\pi(2)}{8}}=e^{-j\frac{\pi}{2}}=-j$$

Third Stage

$$X(0) = F_1(0) + W_8^0 F_2(0) = 2 + 1.(-2) = 0$$

$$X(1) = F_1(1) + W_8^1 F_2(1) = 2j + (0.707 - j0.707)(-2)$$

$$= -1.414 + j3.414$$

$$X(2) = F_1(2) + W_8^2 F_2(2) = 2 - j2$$

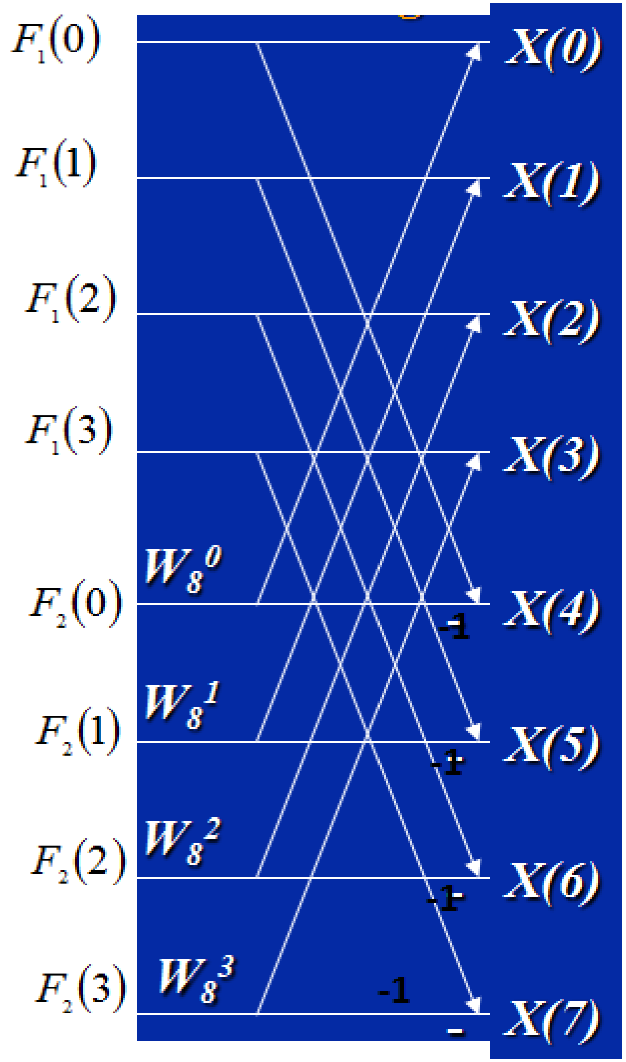
$$X(3) = F_1(3) + W_8^3 F_2(3) = 1.414 - j0.585$$

$$X(4) = F_1(0) - W_8^0 F_2(0) = 4$$

$$X(5) = F_1(1) - W_8^1 F_2(1) = 1.414 + j0.585$$

$$X(6) = F_1(2) + W_8^2 F_2(2) = 2 + j2$$

$$X(7) = F_1(3) + W_8^3 F_2(3) = -1.4142 - j3.414$$



$$W_8^0 = e^{-j\frac{2\pi(0)}{8}} = 1$$

$$W_8^1 = e^{-j\frac{2\pi(1)}{8}} = e^{-j\frac{\pi}{4}} = 0707 - j0.707$$

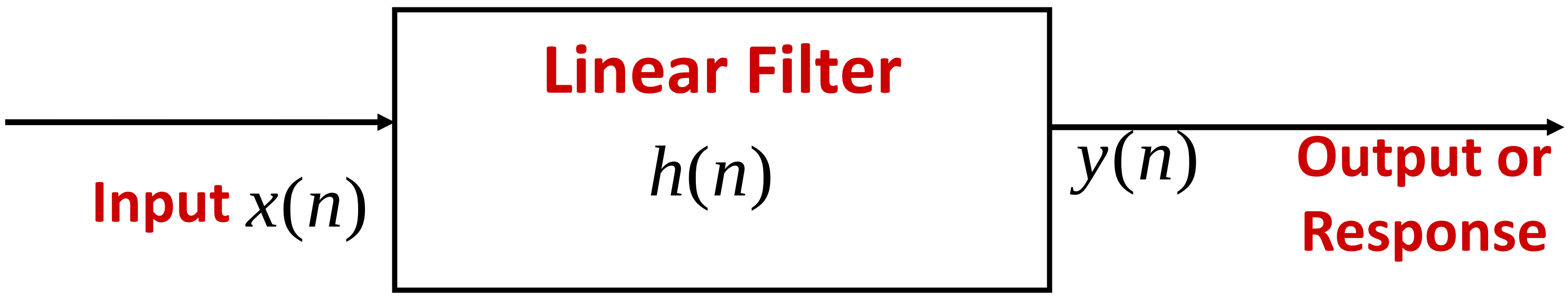
$$W_8^2 = -j; W_8^3 = e^{-j\frac{2\pi(3)}{8}} = e^{-j\frac{3\pi}{4}} = -0707 - j0.707$$

Application of FFT Algorithm

- Linear Filtering
- Correlation
- Spectrum Analysis

Use Of FFT in Linear Filtering

- Let h(n).0≤n≤M-1,be the sample response of the FIR filter
- x(n)-Input data Sequence



- The block size of the FFT Algorithm is N
- N=L+M-1 and L is the number of new data samples being processed by the filter.
- N is a power of 2 . $N = 2^3$

1. The N-point DFT of h(n),which is padded by L-1 zeros, is denoted as H(K) using either DIT or DIF algorithm
2. The N-point DFT of x(n), is denoted as X(K) using either DIT or DIF algorithm.
3. Multiply X(k) and H(K), Y(k)=X(k).H(K)
4. The inverse DFT can be computed by use of an FFT algorithm

$$y(n) = \frac{1}{N}\left[\sum_{n=0}^{N-1} Y^*(k) W_N^{nk}\right]^*$$

- Step 1: Conjugate of Y(K)
- Step 2: DFT of Y*(k) using either DIT or DIF radix-2 Algorithm
- Step 3:conjugate of result of step 2
- Step 4: The result of Step 3 Divided by N, we get y(n).

Determine the response of LTI system when the input sequence x(n)={-1,1,2,1,-1} by radix-2 DIT FFT. The impulse response of the systems is h(n)={-1,1,1,-1}

- L=5,M=4,N=L+M-1=8
- x(n)= {-1,1,2,1,-1,0,0,0}
- h(n)={-1,1,1,-1,0,0,0,0}
- Determine X(K) using DIT or DIF algorithm
- X(K)={ 2,- 3.414j,-4,0.585j,-2,-0.5858j,-4,3.414j
- Determine H(K) using DIT or DIF algorithm
- H(K)={ 0,-1- 0.414j,0,-1-2.414j,-4,-1+2.414j,0,-1+0.414j}
- Y(k)={0,-1.413+3.414j,0,1.412-j0.585,8,1.12+0.585j,0,-1.413-3.414j}
- Y*(k)={0,-1.413-3.414j,0,1.412+j0.585,8,1.12-0.585j,0,-1.413+3.414j}
- DFT of Y*(k) using Either DIT or DIF
- The result of previous step divide by N=8
- y(n)={1,-2,0,-1,1,0,2,-1}

APPLICATIONS OF FOURIER TRANSFORM

- Image Processing and filters
- Transformation, representation, and encoding
- Smoothing and sharpening
- Restoration, blur removal, and Wiener filter
- Data Processing and Analysis
- Seismic arrays and streamers
- Multibeam echo sounder and side scan sonar
- Synthetic Aperture Radar (SAR) and Interferometric SAR (In-SAR)
- High-pass, low-pass, and band-pass filters
- Cross correlation — transfer functions — Coherence
- Signal and noise estimation — encoding time series.

REFERENCES

1. John G. Proakis and Dimitris G. Manolakis, —*Digital Signal Processing: Principles,Algorithms, and Applications*,‖ 4th edition, 2007.

2. http://pws.npru.ac.th/sartthong/data/files/Digital%20Signal%20Processing%20-%20Computer%20Based%20Approach%20-%20Sanjit%20K.%20Mitra.pdf.

3. https://www.scribd.com/doc/180108714/Digital-signal-processing-by-sk-mitra-4th-edition-pdf.

4. —Digital Signal Processing with Matlab Programs‖- Dr.S.Sanjay Sharma,

5. http://downloadingstarted.com/digital-signal-processing-by-sanjay-sharma-pdf-free-download.html

6. John G. Proakis and Dimitris G. Manolakis, Digital Signal Processing: Principles,Algorithms, and Applications, 4th edition, 2007.