METHODIST
COLLEGE OF ENGINEERING AND TECHNOLOGY
Approved by AICTE New Delhi | Affiliated to Osmania University, Hyderabad
Estd : 2008 Address : King Koti Road, Abids, Hyderabad, Telangana, 500001 | Email : principal@methodist.edu.in

DEPARTMENT OF

ELECTRONICS AND COMMUNICATION ENGINEERING

# LECTURE NOTES

# ON

# DIGITAL SIGNAL PROCESSING

## B.E V Semester (PC503 EC)

**Mr. I. SRIKANTH,**
**Associate Professor**
**Department of ECE**

**2018-2019**

# UNIT 1

## DISCRETE FOURIER TRANSFORMS (DFT)

### CONTENTS:-

DISCRETE FOURIER TRANSFORMS (DFT): FREQUENCY DOMAIN SAMPLING AND RECONSTRUCTION OF DISCRETE TIME SIGNALS. DFT AS A LINEAR TRANSFORMATION, ITS RELATIONSHIP WITH OTHER TRANSFORMS. 6 HRS

### RECOMMENDED READINGS


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# UNIT 1

# Discrete Fourier Transform

## 1.1 Introduction:

Before we introduce the DFT we consider the sampling of the Fourier transform of an aperiodic discrete-time sequence. Thus we establish the relation between the sampled Fourier transform and the DFT.A discrete time system may be described by the convolution sum, the Fourier representation and the z transform as seen in the previous chapter. If the signal is periodic in the time domain DTFS representation can be used, in the frequency domain the spectrum is discrete and periodic. If the signal is non-periodic or of finite duration the frequency domain representation is periodic and continuous this is not convenient to implement on the computer. Exploiting the periodicity property of DTFS representation the finite duration sequence can also be represented in the frequency domain, which is referred to as Discrete Fourier Transform 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.

## 1.2 FREQUENCY DOMAIN SAMPLING AND RECONSTRUCTION OF DISCRETE TIME SIGNALS:

Consider an aperiodic discrete time signal x (n) with Fourier transform, an aperiodic finite energy signal has continuous spectra. For an aperiodic signal x[n] the spectrum is:

$$X[w] = \sum_{n=-\infty}^{\infty} x[n]e^{-jwn} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1.1)$$

Suppose we sample X[w] periodically in frequency at a sampling of δw radians between successive samples. We know that DTFT is periodic with 2π, therefore only samples in the fundamental frequency range will be necessary. For convenience we take N equidistant samples in the interval (0<=w<2π ). The spacing between samples will be $\delta w = \frac{2\pi}{N}$ as shown below in Fig.1.1.

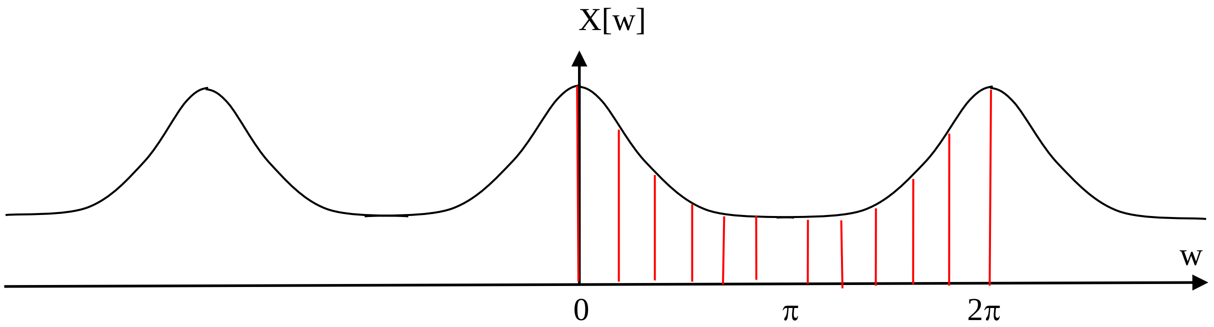



**Fig 1.1 Frequency Domain Sampling**

Let us first consider selection of N, or the number of samples in the frequency domain.

If we evaluate equation (1) at $w = \frac{2\pi k}{N}$

$$X\left[\frac{2\pi k}{N}\right] = \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi kn/N} \qquad k = 0,1,2,......,(N-1) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.2)$$

We can divide the summation in (1) into infinite number of summations where each sum contains N terms.

$$X\left[\frac{2\pi k}{N}\right] = \ldots\ldots + \sum_{n=-N}^{-1} x[n] e^{-j2\pi kn/N} + \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} + \sum_{n=N}^{2N-1} x[n] e^{-j2\pi kn/N}$$

$$= \sum_{l=-\infty}^{\infty} \sum_{n=lN}^{lN+N-1} x[n] e^{-j2\pi kn/N}$$

If we then change the index in the summation from n to n-l N and interchange the order of summations we get:

$$\left[\frac{2\pi k}{N}\right]=\sum_{n=0}^{N-1}\left[\sum_{l=-\infty}^{\infty}x[n-lN]\right]e^{-j2\pi kn/N} \qquad for\ k=0,1,2,......,(N-1) \ldots\ldots.(1.3)$$

Denote the quantity inside the bracket as $x_p[n]$. This is the signal that is a repeating version of x[n] every N samples. Since it is a periodic signal it can be represented by the Fourier series.

$$x_p[n]=\sum_{k=0}^{N-1}c_k e^{j2\pi kn/N} \qquad n=0,1,2,........,(N-1)$$

With FS coefficients:

$$c_k=\frac{1}{N}\sum_{n=0}^{N-1}x_p[n]e^{-j2\pi kn/N} \qquad k=0,1,2,.......,(N-1) \ldots\ldots\ldots\ldots\ldots (1.4)$$

Comparing the expressions in equations (1.4) and (1.3) we conclude the following:

$$c_k=\frac{1}{N}X\left[\frac{2\pi}{N}k\right] \qquad k=0,1,.......,(N-1) \ldots\ldots\ldots\ldots\ldots\ldots (1.5)$$

Therefore it is possible to write the expression $x_p[n]$ as below:

$$x_p[n]=\frac{1}{N}\sum_{k=0}^{N-1}X\left[\frac{2\pi}{N}k\right]e^{j2\pi kn/N} \qquad n=0,1,.....,(N-1) \ldots\ldots\ldots (1.6)$$

The above formula shows the reconstruction of the periodic signal $x_p[n]$ from the samples of the spectrum X[w]. But it does not say if X[w] or x[n] can be recovered from the samples.

Let us have a look at that:

Since $x_p[n]$ is the periodic extension of x[n] it is clear that x[n] can be recovered from $x_p[n]$ if there is no aliasing in the time domain. That is if x[n] is time-limited to less than the period N of $x_p[n]$.This is depicted in Fig. 1.2 below:

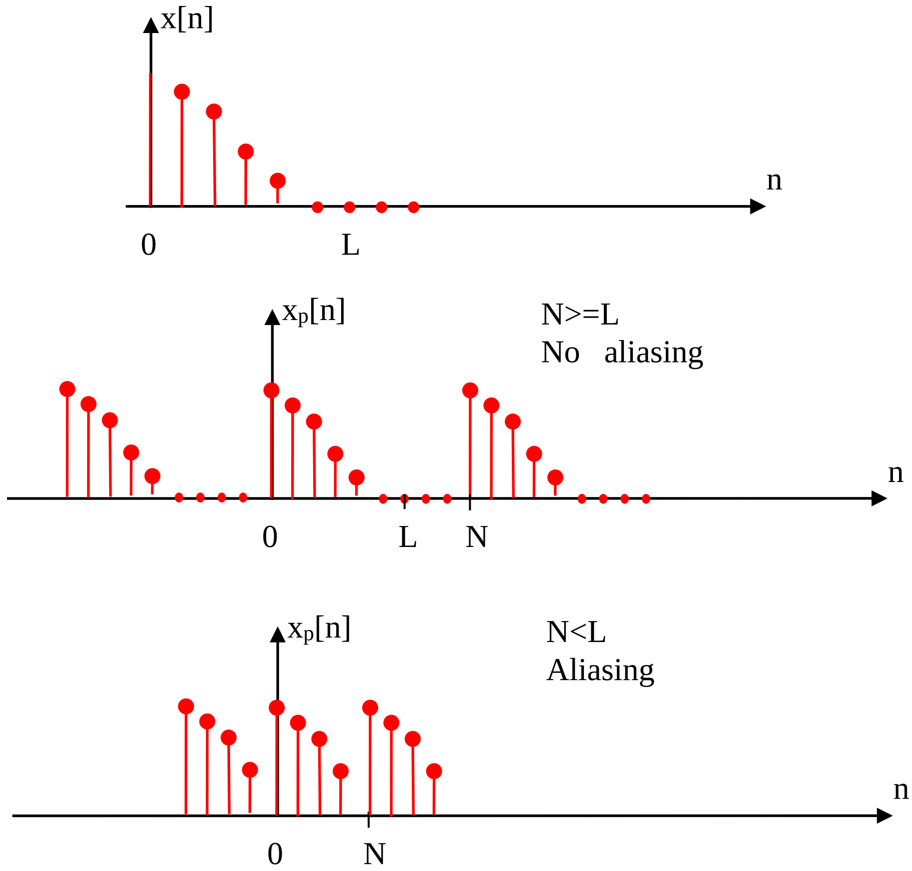



**Fig. 1.2 Signal Reconstruction**

Hence we conclude:
The spectrum of an aperiodic discrete-time signal with finite duration L can be exactly recovered from its samples at frequencies $w_k = \frac{2\pi k}{N}$ if N >= L.

We compute $x_p[n]$ for n=0, 1,....., N-1 using equation (1.6)
Then X[w] can be computed using equation (1.1).

## 1.3 Discrete Fourier Transform:

The DTFT representation for a finite duration sequence is

$$X(j\omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\omega n}$$

$$X(n) = 1/2\pi \int X(j\omega)\, e^{j\omega n}\, d\omega, \quad \text{Where } \omega = 2\pi k/n$$

$2\pi$

Where x(n) is a finite duration sequence, X(jω) is periodic with period 2π.It is convenient sample X(jω) with a sampling frequency equal an integer multiple of its period =m that is taking N uniformly spaced samples between 0 and 2π.

Let $\omega_k = 2\pi k/n$, $0 \le k \le N-1$

$$\text{Therefore } X(j\omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j2\pi kn/N}$$

Since X(jω) is sampled for one period and there are N samples X(jω) can be expressed as

$$X(k) = X(j\omega)\Big|_{\omega=2\pi kn/N} = \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi kn/N} \qquad 0 \le k \le N-1$$

## 1.4 Matrix relation of DFT

The DFT expression can be expressed as

[X] = [x(n)] [WN]

Where $[X] = [X(0), X(1), \ldots\ldots]^T$

[x] is the transpose of the input sequence. WN is a N x N matrix

$$WN = \begin{matrix} 1 & 1 & 1 & 1 & \ldots\ldots\ldots\ldots 1 \\ 1 & wn1 & wn2 & wn3 & \ldots\ldots\ldots\ldots wn\ n-1 \\ 1 & wn2 & wn4 & wn6 & \ldots\ldots\ldots\ldots wn2(n-1) \\ \ldots & & & & \\ \ldots & & & & \\ 1 & \ldots & & & \ldots wN\ (N-1)(N-1) \end{matrix}$$

ex;

4 pt DFT of the sequence 0,1,2,3

$$\begin{matrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{matrix} = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{matrix}$$

Solving the matrix X(K) = 6 , -2+2j, -2 , -2-2j

## 1.5 Relationship of Fourier Transforms with other transforms

### 1.5.1 Relationship of Fourier transform with continuous time signal:

Suppose that $x_a(t)$ is a continuous-time periodic signal with fundamental period $T_p = 1/F_0$.The signal can be expressed in Fourier series as

$$x_a(t) = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi k F_0}$$

Where {ck} are the Fourier coefficients. If we sample $x_a(t)$ at a uniform rate Fs = $N/T_p = 1/T$, we obtain discrete time sequence

$$x(n) \equiv x_a(nT) = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi k F_0 nT} = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi kn/N}$$

$$= \sum_{k=0}^{N-1} \left[ \sum_{l=-\infty}^{\infty} c_{k-lN} \right] e^{j2\pi kn/N}$$

$$X(k) = N \sum_{l=-\infty}^{\infty} c_{k-lN} \equiv N\tilde{c}_k$$

Thus $\{c_{k'}\}$ is the aliasing version of $\{c_k\}$

### 1.5.2 Relationship of Fourier transform with z-transform

Let us consider a sequence x(n) having the z-transform

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

With ROC that includes unit circle. If X(z) is sampled at the N equally spaced points on the unit circle $Z_k = e^{j2\pi k/N}$ for K= 0,1,2,………..N-1 we obtain

$$X(k) \equiv X(z)|_{z=e^{j2\pi k/N}} \qquad k = 0, 1, \ldots, N-1$$

$$= \sum_{n=-\infty}^{\infty} x(n) e^{-j2\pi nk/N}$$

The above expression is identical to Fourier transform X(ω) evaluated at N equally spaced frequencies $\omega_k = 2\pi k/N$ for K= 0,1,2,………..N-1.

If the sequence x(n) has a finite duration of length N or less. The sequence can be recovered from its N-point DFT. Consequently X(z) can be expressed as a function of DFT as

$$X(z) = \sum_{n=0}^{N-1} x(n) z^{-n}$$

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

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

$$X(z) = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{j2\pi k/N} z^{-1}}$$

Fourier transform of a continuous time signal can be obtained from DFT as

$$X(\omega) = \frac{1 - e^{-j\omega N}}{N} \sum_{k=0}^{N-1} \frac{X(k)}{1 - e^{-j(\omega - 2\pi k/N)}}$$

# Recommended Questions with solutions

## Question 1

The first five points of the 8-point DFT of a real valued sequence are {0.25, 0.125-j0.318, 0, 0.125-j0.0518, 0}. Determine the remaining three points

**Ans:** Since x(n) is real, the real part of the DFT is even, imaginary part odd. Thus the remaining points are {0.125+j0.0518,0,0, 0.125+j0.318}.

## Question 2

Compute the eight-point DFT circular convolution for the following sequences.
$x_2(n) = \sin 3\pi n/8$

**Ans:**

(a)

$$\begin{aligned}
\tilde{x}_2(l) &= x_2(l), \quad 0 \le l \le N-1 \\
&= x_2(l+N), \quad -(N-1) \le l \le -1 \\
\tilde{x}_2(l) &= sin(\frac{3\pi}{8}l), \quad 0 \le l \le 7 \\
&= sin(\frac{3\pi}{8}(l+8)), \quad -7 \le l \le -1 \\
&= sin(\frac{3\pi}{8}|l|), \quad |l| \le 7 \\
\text{Therefore, } x_1(n) \circledast_8 x_2(n) &= \sum_{m=0}^{3} \tilde{x}^2(n-m) \\
&= sin(\frac{3\pi}{8}|n|) + sin(\frac{3\pi}{8}|n-1|) + \ldots + sin(\frac{3\pi}{8}|n-3|) \\
&= \{1.25, 2.55, 2.55, 1.25, 0.25, -1.06, -1.06, 0.25\}
\end{aligned}$$

## Question 3

Compute the eight-point DFT circular convolution for the following sequence
$X_3(n) = \cos 3\pi n/8$

$$\begin{aligned}
\tilde{x}_2(n) &= cos(\frac{3\pi}{8}n), \quad 0 \le l \le 7 \\
&= -cos(\frac{3\pi}{8}n), \quad -7 \le l \le -1 \\
&= [2u(n)-1]\,cos(\frac{3\pi}{8}n), \quad |n| \le 7 \\
\text{Therefore, } x_1(n) \circledast_8 x_2(n) &= \sum_{m=0}^{3} \left(\frac{1}{4}\right)^m \tilde{x}^2(n-m) \\
&= \{0.96, 0.62, -0.55, -1.06, -0.26, -0.86, 0.92, -0.15\}
\end{aligned}$$

## Question 4

Define DFT. Establish a relation between the Fourier series coefficients of a continuous time signal and DFT

## Solution

The DTFT representation for a finite duration sequence is

$$X(j\omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\omega n}$$

$$X(n) = 1/2\pi \int_{2\pi} X(j\omega)\, e^{j\omega n}\, d\omega, \quad \text{Where } \omega = 2\pi k/n$$

Where x(n) is a finite duration sequence, X(jω) is periodic with period 2π.It is convenient sample X(jω) with a sampling frequency equal an integer multiple of its period =m that is taking N uniformly spaced samples between 0 and 2π.

Let $\omega_k = 2\pi k/n$, $0 \le k \le N$

$$\text{Therefore } X(j\omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j2\pi kn/N}$$

Since X(jω) is sampled for one period and there are N samples X(jω) can be expressed as

$$X(k) = X(j\omega)\Big|_{\omega=2\pi kn/N} = \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi kn/N} \quad 0 \le k \le N-1$$

## Question 5

**5.7** If $X(k)$ is the DFT of the sequence $x(n)$, determine the $N$-point DFTs of the sequences

$$x_c(n) = x(n)\cos\frac{2\pi kn}{N} \qquad 0 \le n \le N-1$$

and

$$x_s(n) = x(n)\sin\frac{2\pi kn}{N} \qquad 0 \le n \le N-1$$

in terms of $X(k)$.

**Solution:-**

$$\begin{aligned}
X_c(k) &= \sum_{n=0}^{N-1} \frac{1}{2} x(n)\left(e^{j\frac{2\pi k_0 n}{N}} + e^{-j\frac{2\pi k_0 n}{N}}\right) e^{-\frac{2\pi kn}{N}} \\
&= \frac{1}{2}\sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi(k-k_0)n}{N}} + \frac{1}{2}\sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi(k+k_0)n}{N}} \\
&= \frac{1}{2}X(k-k_0)_{\text{mod}N} + \frac{1}{2}X(k+k_0)_{\text{mod}N}
\end{aligned}$$

$$\text{similarly, } X_s(k) = \frac{1}{2j}X(k-k_0)_{\text{mod}N} - \frac{1}{2j}X(k+k_0)_{\text{mod}N}$$

## Question 6

Find the 4-point DFT of sequence x(n) = 6+ sin(2πn/N), n= 0,1,………N-1

**Solution :-**

$$\text{Here } x(n) = 6+\sin\left(\frac{2\pi n}{N}\right), \quad \textit{with } N=4$$

$$x(n) = 6+\sin\left(\frac{2\pi n}{4}\right), \; n=0,1,2,3.$$

$$= 6+\sin\left(\frac{\pi}{2}n\right), \; n=0,1,2,3.$$

$$= \{6, 7, 6, 5\}.$$

The N-point DFT is given as,

$$X_N = [W_N]x_N$$

$$\therefore \quad X_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 6 \\ 7 \\ 6 \\ 5 \end{bmatrix}$$

$$= \begin{bmatrix} 6+7+6+5 \\ 6-j7-6+j5 \\ 6-7+6-5 \\ 6+j7-6-j5 \end{bmatrix} = \begin{bmatrix} 24 \\ -j2 \\ 0 \\ +j2 \end{bmatrix}$$

## Question 7

Determine the eight-point DFT of the signal

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

and sketch its magnitude and phase.

## Solution

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{7} x(n) e^{-j\frac{2\pi}{8}kn} \\
&= \{6, -0.7071 - j1.7071, 1 - j, 0.7071 + j0.2929, 0, 0.7071 - j0.2929, 1 + j, \\
&\quad -0.7071 + j1.7071\} \\
|X(k)| &= \{6, 1.8478, 1.4142, 0.7654, 0, 0.7654, 1.4142, 1.8478\} \\
\angle X(k) &= \left\{0, -1.9635, \frac{-\pi}{4}, 0.3927, 0, -0.3927, \frac{\pi}{4}, 1.9635\right\}
\end{aligned}
$$

## Question 8

Compute the $N$-point DFTs of the signal

$$x(n) = \cos \frac{2\pi}{N} k_0 n \quad 0 \leq n \leq N-1$$

## Solution

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{N-1} e^{j\frac{2\pi}{N}nk_0} e^{-j\frac{2\pi}{N}kn} \\
&= \sum_{n=0}^{N-1} e^{-j\frac{2\pi}{N}(k-k_0)n} \\
&= N\delta(k - k_0)
\end{aligned}
$$

$$
\begin{aligned}
x(n) &= \frac{1}{2} e^{j\frac{2\pi}{N}nk_0} + \frac{1}{2} e^{-j\frac{2\pi}{N}nk_0} \\
\text{From (e) we obtain } X(k) &= \frac{N}{2}\left[\delta(k - k_0) + \delta(k - N + k_0)\right]
\end{aligned}
$$

# UNIT 2

## PROPERTIES OF DISCRETE FOURIER TRANSFORMS (DFT)

### CONTENTS:-

PROPERTIES OF DFT, MULTIPLICATION OF TWO DFTS- THE CIRCULAR CONVOLUTION, ADDITIONAL DFT PROPERTIES. 6 HRS

### RECOMMENDED READINGS


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# Unit 2
# Properties of DFT

## 2.1 Properties:-

The DFT and IDFT for an N-point sequence x(n) are given as

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

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

where $W_N$ is defined as

$$W_N = e^{-j2\pi/N}$$

In this section we discuss about the important properties of the DFT. These properties are helpful in the application of the DFT to practical problems.

The notation used below to denote the $N$-point DFT pair $x(n)$ and $X(k)$ is

$$x(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} X(k)$$

### Periodicity:-

If $x(n)$ and X(k) are an $N$-point DFT pair, then

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

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

### 2.1.2 Linearity: If

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

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

Then A x1 (n) + b x2 (n) ←→ a X1(k) + b X2(k)

## 2.1.3 Circular shift:

In linear shift, when a sequence is shifted the sequence gets extended. In circular shift the number of elements in a sequence remains the same. Given a sequence x (n) the shifted version x (n-m) indicates a shift of m. With DFTs the sequences are defined for 0 to N-1.

If x (n) = x (0), x (1), x (2), x (3)

X (n-1) = x (3), x (0), x (1).x (2)

X (n-2) = x (2), x (3), x (0), x (1)

## 2.1.4 Time shift:

If x (n) ←→ X (k)

Then x (n-m) ←→ $W_N^{mk}$ X (k)

## 2.1.5 Frequency shift

If x(n)←→ X(k)

$W_n^{+nok}$ x(n) ←→X(k+no)

Consider x(k) $=\sum_{n=0}^{N-1} x(n) W_n^{kn}$

$$X(k+no)=\sum_{n=0}^{N-1} x(n)\, W_N^{(k+no)n}$$

$$= \sum x(n)\, W_N^{kn}\, W_N^{non}$$

∴ X(k+no)←→x(n) $W_N^{non}$

## 2.1.6 Symmetry:

For a real sequence, if $x(n) \longleftrightarrow X(k)$

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

For a complex sequence
$DFT(x^*(n)) = X^*(N-K)$

| If x(n) | then X(k) |
|---|---|
| Real and even | real and even |
| Real and odd | imaginary and odd |
| Odd and imaginary | real odd |
| Even and imaginary | imaginary and even |

## 2.2 Convolution theorem;

Circular convolution in time domain corresponds to multiplication of the DFTs

If $y(n) = x(n) \otimes h(n)$ then $Y(k) = X(k) H(k)$

Ex let x(n) = 1,2,2,1 and h(n) = 1,2,2,1
Then $y(n) = x(n) \otimes h(n)$

Y(n) = 9,10,9,8

N pt DFTs of 2 real sequences can be found using a single DFT

If g(n) & h(n) are two sequences then let $x(n) = g(n) + j\,h(n)$

$G(k) = ½ (X(k) + X^*(k))$

$H(k) = 1/2j (X(K) + X^*(k))$

2N pt DFT of a real sequence using a single N pt DFT

Let x(n) be a real sequence of length 2N with y(n) and g(n) denoting its N pt DFT

Let $y(n) = x(2n)$ and $g(2n+1)$

$X(k) = Y(k) + W_N^k\, G(k)$

Using DFT to find IDFT

The DFT expression can be used to find IDFT

X(n) = 1/N [DFT(X*(k)]*

# Recommended Questions with solutions

## Question 1

State and Prove the Time shifting Property of DFT

## Solution

The DFT and IDFT for an N-point sequence x(n) are given as

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

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

where $W_N$ is defined as

$$W_N = e^{-j2\pi/N}$$

Time shift:

If x (n) ←→ X (k)

Then x (n-m) ←→ $W_N^{mk}$ X (k)

## Question 2

State and Prove the: (i) Circular convolution property of DFT; (ii) DFT of Real and even sequence.

## Solution

***(i) Convolution theorem***

Circular convolution in time domain corresponds to multiplication of the DFTs
If y(n) = x(n) ⊗ h(n) then Y(k) = X(k) H(k)
Ex let x(n) = 1,2,2,1 and h(n) = 1,2,2,1 Then y (n) = x(n) ⊗ h(n)

Y(n) = 9,10,9,8
N pt DFTs of 2 real sequences can be found using a single DFT

If g(n) & h(n) are two sequences then let x(n) = g(n) +j h(n)
G(k) = ½ (X(k) + X*(k))
H(k) = 1/2j (X(K) +X*(k))
2N pt DFT of a real sequence using a single N pt DFT
Let x(n) be a real sequence of length 2N with y(n) and g(n) denoting its N pt DFT
Let y(n) = x(2n) and g(2n+1)

$X (k) = Y (k) + W_N^K \; G (k)$
Using DFT to find IDFT
The DFT expression can be used to find IDFT
X(n) = 1/N [DFT(X*(k)]*

***(ii)DFT of Real and even sequence.***
For a real sequence, if x(n)←→ X(k)

X (N-K) = X* (k)

For a complex sequence
DFT(x*(n)) = X*(N-K)

| If x(n) then | X(k) |
|---|---|
| Real and even | real and even |
| Real and odd | imaginary and odd |
| Odd and imaginary | real odd |
| Even and imaginary | imaginary and even |

**<u>Question 3</u>**

Distinguish between circular and linear convolution

**<u>Solution</u>**

1) Circular convolution is used for periodic and finite signals while linear convolution is used for aperiodic and infinite signals.
2) In linear convolution we convolved one signal with another signal where as in circular convolution the same convolution is done but in circular pattern depending upon the samples of the signal
3) Shifts are linear in linear in linear convolution, whereas it is circular in circular convolution.

**<u>Question 4</u>**

For the sequences

$$x_1(n) = \cos\frac{2\pi}{N}n \qquad x_2(n) = \sin\frac{2\pi}{N}n \qquad 0 \le n \le N-1$$

determine the $N$-point:

(a) Circular convolution $x_1(n) \, (N) \, x_2(n)$

(b) Circular correlation of $x_1(n)$ and $x_2(n)$

(c) Circular autocorrelation of $x_1(n)$

(d) Circular autocorrelation of $x_2(n)$

**Solution(a)**

$$\begin{aligned}
x_1(n) &= \frac{1}{2}\left(e^{j\frac{2\pi}{N}n} + e^{-j\frac{2\pi}{N}n}\right) \\
X_1(k) &= \frac{N}{2}\left[\delta(k-1) + \delta(k+1)\right] \\
\text{also } X_2(k) &= \frac{N}{2j}\left[\delta(k-1) - \delta(k+1)\right] \\
\text{So } X_3(k) &= X_1(k)X_2(k) \\
&= \frac{N^2}{4j}\left[\delta(k-1) - \delta(k+1)\right] \\
\text{and } x_3(n) &= \frac{N}{2}sin(\frac{2\pi}{N}n)
\end{aligned}$$

**Solution(b)**

$$\begin{aligned}
\tilde{R}_{xy}(k) &= X_1(k)X_2^*(k) \\
&= \frac{N^2}{4j}\left[\delta(k-1) - \delta(k+1)\right] \\
\Rightarrow \quad \tilde{r}_{xy}(n) &= -\frac{N}{2}sin(\frac{2\pi}{N}n)
\end{aligned}$$

**Solution(c)**

$$\begin{aligned}
\tilde{R}_{xx}(k) &= X_1(k)X_1^*(k) \\
&= \frac{N^2}{4}\left[\delta(k-1) + \delta(k+1)\right] \\
\Rightarrow \quad \tilde{r}_{xx}(n) &= \frac{N}{2}cos(\frac{2\pi}{N}n)
\end{aligned}$$

**Solution(d)**

$$
\begin{aligned}
\tilde{R}_{yy}(k) &= X_2(k)X_2^*(k) \\
&= \frac{N^2}{4}[\delta(k-1)+\delta(k+1)] \\
\Rightarrow \quad \tilde{r}_{yy}(n) &= \frac{N}{2}cos(\frac{2\pi}{N}n)
\end{aligned}
$$

## Question 5

Use the four-point DFT and IDFT to determine the sequence

$$x_3(n) = x_1(n) \, (N) \, x_2(n)$$

where $x_1(n)$ and $x_2(n)$ are the sequence given

$$x_1(n) = \{\underset{\uparrow}{1}, 2, 3, 1\}$$

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

## Solution

$$
\begin{aligned}
y(n) &= x_1(n)4x_2(n) \\
&= \sum_{m=0}^{3} x_1(m)_{\text{mod}4} x_2(n-m)_{\text{mod}4} \\
&= \{17, 19, 22, 19\}
\end{aligned}
$$

$$
\begin{aligned}
X_1(k) &= \{7, -2-j, 1, -2+j\} \\
X_2(k) &= \{11, 2-j, 1, 2+j\} \\
\Rightarrow X_3(k) &= X_1(k)X_2(k) \\
&= \{17, 19, 22, 19\}
\end{aligned}
$$

## Question 6

A linear time-invariant system with frequency response $H(\omega)$ is excited with the periodic input

$$x(n) = \sum_{k=-\infty}^{\infty} \delta(n-kN)$$

Suppose that we compute the $N$-point DFT $Y(k)$ of the samples $y(n)$, $0 \le n \le N-1$ of the output sequence. How is $Y(k)$ related to $H(\omega)$?

## Solution

$$
\begin{aligned}
x(n) &= \sum_{i=-\infty}^{\infty} \delta(n-iN) \\
y(n) &= \sum_{m} h(m)x(n-m) \\
&= \sum_{m} h(m)\left[\sum_{i} \delta(n-m-iN)\right] \\
&= \sum_{i} h(n-iN)
\end{aligned}
$$

Therefore, $y(.)$ is a periodic sequence with period N. So

$$
\begin{aligned}
Y(k) &= \sum_{n=0}^{N-1} y(n)W_N^{kn} \\
&= H(w)|_{w=\frac{2\pi}{N}k}
\end{aligned}
$$

$$
Y(k) = H(\frac{2\pi k}{N}) \quad k = 0, 1, \ldots, N-1
$$

# UNIT 3

## FAST FOURIER TRANSFORMS (FFT) ALOGORITHMS

## CONTENTS:-

USE OF DFT IN LINEAR FILTERING : OVERLAP-SAVE AND OVERLAP-ADD METHOD. DIRECT COMPUTATION OF DFT, NEED FOR EFFICIENT COMPUTATION OF THE DFT (FFT ALGORITHMS).

7 HRS

## RECOMMENDED READINGS


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# UNIT 3
# FAST-FOURIER-TRANSFORM (FFT) ALGORITHMS

## 3.1 Digital filtering using DFT

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.

### 3.1.1 Overlap and add

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

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

$$x_2(n) = \{x(L), x(L+1), \ldots, x(2L-1), \underbrace{0, 0, \ldots, 0}_{M-1 \text{ zeros}}\}$$

$$x_3(n) = \{x(2L), \ldots, x(3L-1), \underbrace{0, 0, \ldots, 0}_{M-1 \text{ zeros}}\}$$

The two $N$-point DFTs are multiplied together to form

$$Y_m(k) = H(k)X_m(k) \qquad k = 0, 1, \ldots, N-1$$

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.

$$y(n) = \{y_1(0), y_1(1), \ldots, y_1(L-1), y_1(L) + y_2(0), y_1(L+1) + y_2(1), \ldots, y_1(N-1) + y_2(M-1), y_2(M), \ldots\}$$

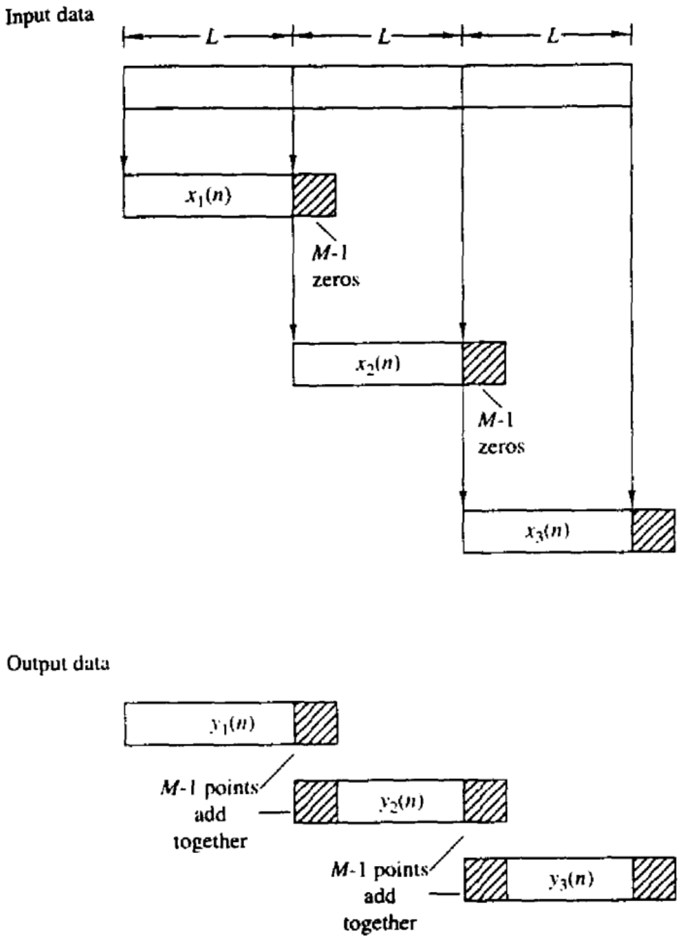



**Figure 5.11** Linear FIR filtering by the overlap-add method.

## 2.1.2 Overlap and 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

$$\hat{Y}_m(k) = H(k)X_m(k) \qquad k = 0, 1, \ldots, N-1$$

**Then the $N$-point IDFT yields the result**

$$\hat{Y}_m(n) = \{\hat{y}_m(0)\hat{y}_m(1)\cdots\hat{y}_m(M-1)\hat{y}_m(M)\cdots\hat{y}_m(N-1)\}$$

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

$$\hat{y}_m(n) = y_m(n), n = M, M+1, \ldots, N-1$$

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

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

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

and so forth. The resulting data sequences from the IDFT are given by (5.3.8), where the first $M-1$ points are discarded due to aliasing and the remaining $L$ points constitute the desired result from linear convolution. This segmentation of the input data and the fitting of the output data blocks together to form the output sequence are graphically illustrated in Fig. 5.10.

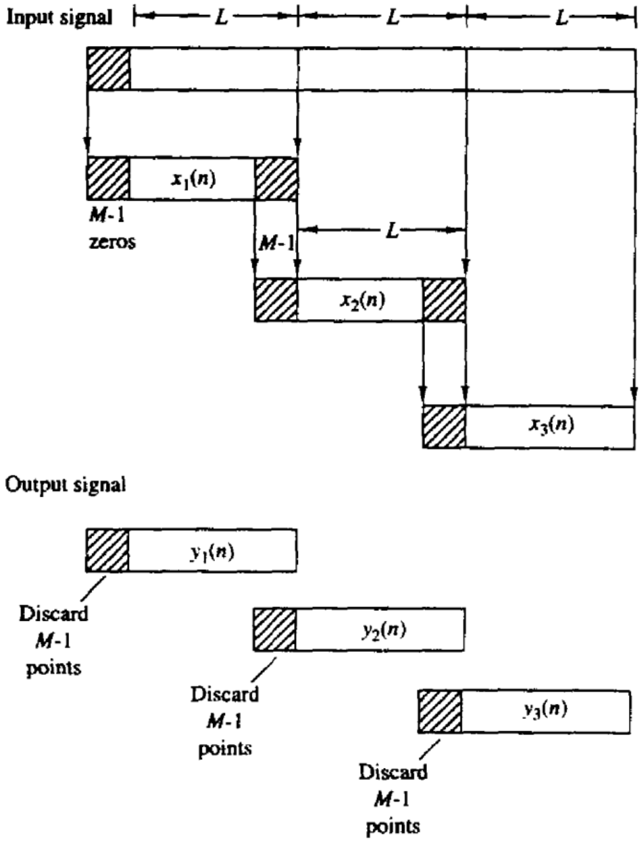



**Figure 5.10** Linear FIR filtering by the overlap-save method.

## 3.2 Direct Computation of DFT

The problem:
Given signal samples: x[0], . . . , x[N - 1] (some of which may be zero), develop a procedure to compute

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

for k = 0, . . . , N - 1 where

$$W_N = e^{-\jmath \frac{2\pi}{N}}.$$

We would like the procedure to be fast, simple, and accurate. Fast is the most important, so we will sacrifice simplicity for speed, hopefully with minimal loss of accuracy

## 3.3 Need for efficient computation of DFT (FFT Algorithms)

Let us start with the simple way. Assume that $W_N^{kn}$ has been precompiled and stored in a table for the N of interest. How big should the table be? $W_N^{m}$ is periodic in m with period N, so we just need to tabulate the N values:

$$W_N^m = \cos\left(\frac{2\pi}{N}m\right) - \jmath \sin\left(\frac{2\pi}{N}m\right)$$

(Possibly even less since Sin is just Cos shifted by a quarter periods, so we could save just Cos when N is a multiple of 4.)

Why tabulate? To avoid repeated function calls to Cos and sin when computing the DFT. Now we can compute each X[k] directly form the formula as follows

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

For each value of k, there are N complex multiplications, and (N-1) complex additions. There are N values of k, so the total number of complex operations is

$$N \cdot N + N(N-1) = 2N^2 - N \equiv O(N^2).$$

Complex multiplies require 4 real multiplies and 2 real additions, whereas complex additions require just 2 real additions $N^2$ complex multiplies are the primary concern.

$N^2$ increases *rapidly* with N, so how can we reduce the amount of computation? By exploiting the following properties of W:

- Symmetry property: $W_N^{k+N/2} = -W_N^k = e^{\jmath\pi}\, W_N^k$
- Periodicity property: $W_N^{k+N} = W_N^k$
- Recursion property: $W_N^2 = W_{N/2}$

The first and third properties hold for even N, *i.e.*, when 2 is one of the prime factors of N. There are related properties for other prime factors of N.

## Divide and conquer approach

We have seen in the preceding sections that the DFT is a very computationally intensive operation. In 1965, Cooley and Tukey published an algorithm that could be used to compute the DFT much more efficiently. Various forms of their algorithm, which came to be known as the Fast Fourier Transform (FFT), had actually been developed much earlier by other mathematicians (even dating back to Gauss). It was their paper, however, which stimulated a revolution in the field of signal processing.

It is important to keep in mind at the outset that the FFT is *not* a new transform. It is simply a very efficient way to compute an existing transform, namely the DFT. As we saw, a straight forward implementation of the DFT can be computationally expensive because the number of multiplies grows as the square of the input length (i.e. $N^2$ for an $N$ point DFT). The FFT reduces this computation using two simple but important concepts. The first concept, known as divide-and-conquer, splits the problem into two smaller problems. The second concept, known as recursion, applies this divide-and-conquer method repeatedly until the problem is solved.

## Recommended Questions with solutions

### Question1

A designer has available a number of eight-point FFT chips. Show explicitly how he should interconnect three such chips in order to compute a 24-point DFT.

**Solution:-**

Create three subsequences of 8-pts each

$$
\begin{aligned}
Y(k) &= \sum_{n=0,3,6,\ldots}^{21} y(n)W_N^{kn} + \sum_{n=1,4,7,\ldots}^{22} y(n)W_N^{kn} + \sum_{n=2,5,\ldots}^{23} y(n)W_N^{kn} \\
&= \sum_{i=0}^{7} y(3i)W_{\frac{N}{3}}^{ki} + \sum_{i=0}^{7} y(3i+1)W_{\frac{N}{3}}^{ki}W_N^{k} + \sum_{i=0}^{7} y(3i+2)W_{\frac{N}{3}}^{ki}W_N^{2k} \\
&\triangleq Y_1(k) + W_N^k Y_2(k) + W_N^{2k} Y_3(k)
\end{aligned}
$$

where $Y_1, Y_2, Y_3$ represent the 8-pt DFTs of the subsequences.

### Question 2

Let $x(n)$ be a real-valued $N$-point ($N = 2^{\nu}$) sequence. Develop a method to compute an $N$-point DFT $X'(k)$, which contains only the odd harmonics [i.e., $X'(k) = 0$ if $k$ is even] by using only a real $N/2$-spoint DFT.

**Solution:-**

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{N-1} x(n)W_N^{kn} \qquad 0 \le k \le N-1 \\
&= \sum_{n=0}^{\frac{N}{2}-1} x(n)W_N^{kn} + \sum_{n=\frac{N}{2}}^{N-1} x(n)W_N^{kn} \\
&= \sum_{n=0}^{\frac{N}{2}-1} x(n)W_N^{kn} + \sum_{r=0}^{\frac{N}{2}-1} x(r+\frac{N}{2})W_N^{(r+\frac{N}{2})k} \\
\text{Let} X'(k') &= X(2k+1), \quad 0 \le k' \le \frac{N}{2}-1
\end{aligned}
$$

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

$$
\text{Using the fact that } W_N^{2k'n} = W_{\frac{N}{2}}^{k'n}, \quad W_N^N = 1
$$

$$
\begin{aligned}
X'(k') &= \sum_{n=0}^{\frac{N}{2}-1} \left[ x(n)W_N^n W_{\frac{N}{2}}^{k'n} + x(n+\frac{N}{2})W_{\frac{N}{2}}^{k'n} W_N^n W_N^{\frac{N}{2}} \right] \\
&= \sum_{n=0}^{\frac{N}{2}-1} \left[ x(n) - x(n+\frac{N}{2}) \right] W_N^n W_{\frac{N}{2}}^{k'n}
\end{aligned}
$$

## Question 3

The $z$-transform of the sequence $x(n) = u(n) - u(n-7)$ is sampled at five points on the unit circle as follows

$$
x(k) = X(z)|_{z} = e^{j2\pi k/5} \qquad k = 0, 1, 2, 3, 4
$$

**Solution:-**

$$
\begin{aligned}
X(z) &= 1 + z^{-1} + \ldots + z^{-6} \\
X(k) &= X(z)|_{z=e^{j\frac{2\pi}{5}}} \\
&= 1 + e^{-j\frac{2\pi}{5}} + e^{-j\frac{4\pi}{5}} + \ldots + e^{-j\frac{12\pi}{5}} \\
&= 2 + 2e^{-j\frac{2\pi}{5}} + e^{-j\frac{4\pi}{5}} + \ldots + e^{-j\frac{8\pi}{5}} \\
x'(n) &= \{2, 2, 1, 1, 1\} \\
x'(n) &= \sum_m x(n+7m), \qquad n = 0, 1, \ldots, 4
\end{aligned}
$$

Temporal aliasing occurs in first two points of $x'(n)$ because $X(z)$ is not sampled at sufficiently small spacing on the unit circle.

## Question 4

Consider a finite-duration sequence $x(n)$, $0 \le n \le 7$, with $z$-transform $X(z)$. We wish to compute $X(z)$ at the following set of values:

$$z_k = 0.8e^{j[(2\pi k/8)+(\pi/8)]} \qquad 0 \le k \le 7$$

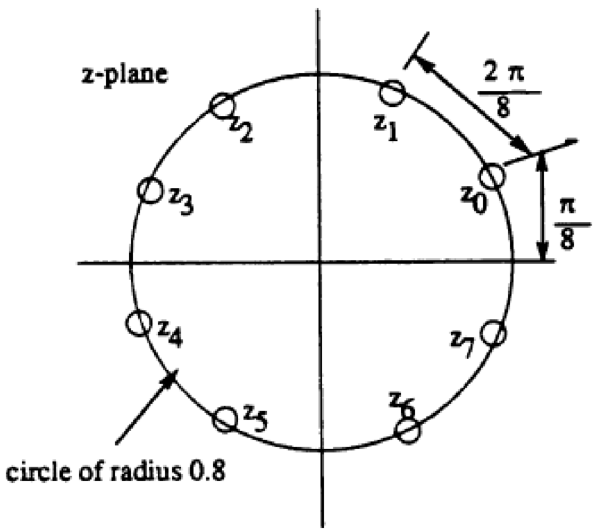
**(a)** Sketch the points $\{z_k\}$ in the complex plane.

**(b)** Determine a sequence $s(n)$ such that its DFT provides the desired samples of $X(z)$.

**Solution:- (a)**

$$Z_k = 0.8e^{j\left[\frac{2\pi k}{8}+\frac{\pi}{8}\right]}$$

**(b)**




$$\begin{aligned}
X(k) &= X(z)|_{z=z_k} \\
&= \sum_{n=0}^{7} x(n)\left[0.8e^{j\left[\frac{2\pi k}{8}+\frac{\pi}{8}\right]}\right]^{-n} \\
s(n) &= x(n)\, 0.8e^{-j\frac{\pi}{8}n}
\end{aligned}$$

# UNIT 4

## FAST FOURIER TRANSFORMS (FFT) ALGORITHMS

## CONTENTS:-

**RADIX-2 FFT ALGORITHM FOR THE COMPUTATION OF DFT AND IDFT**

**DECIMATION IN- TIME AND DECIMATION-IN-FREQUENCY ALGORITHMS. GOERTZEL ALGORITHM, AND CHIRP-Z TRANSFORM.** 7 HRS

## RECOMMENDED READINGS


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# UNIT 4

# RADIX-2 FFT ALGORITHM FOR THE COMPUTATION OF DFT AND IDFT

## 4.1 Introduction:

Standard frequency analysis requires transforming time-domain signal to frequency domain and studying Spectrum of the signal. This is done through DFT computation. N-point DFT computation results in N frequency components. We know that DFT computation through FFT requires N/2 $\log_2 N$ complex multiplications and N $\log_2 N$ additions. In certain applications not all N frequency components need to be computed (an application will be discussed). If the desired number of values of the DFT is less than 2 $\log_2 N$ than direct computation of the desired values is more efficient that FFT based computation.

## 4.2 Radix-2 FFT

Useful when N is a power of 2: N = $r^v$ for integers r and v. ‘r’ is called the **radix**, which comes from the Latin word meaning .a root, and has the same origins as the word radish.

When N is a power of r = 2, this is called **radix-2**, and the natural .divide and conquer approach. is to split the sequence into two sequences of length N=2. This is a very clever trick that goes back many years.

### 4.2.1 Decimation in time

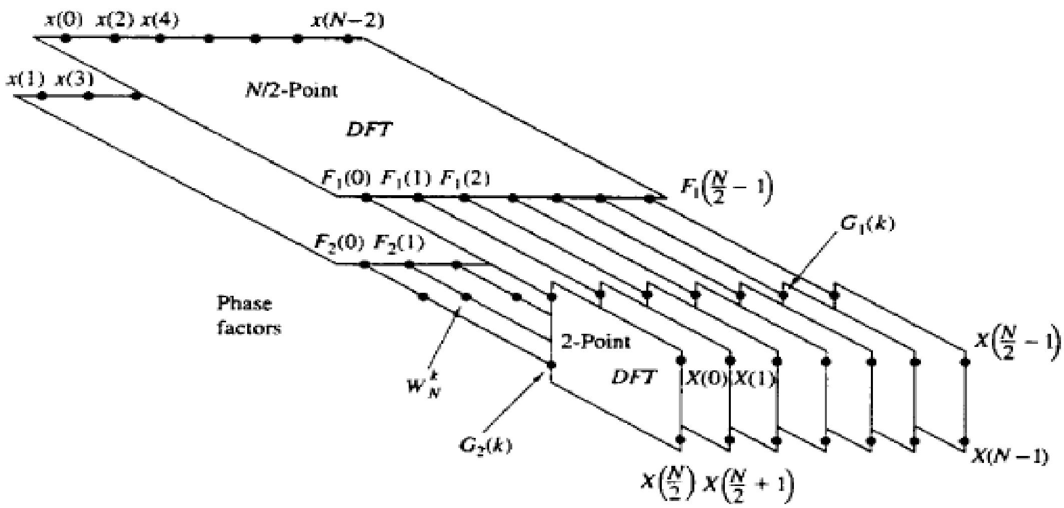
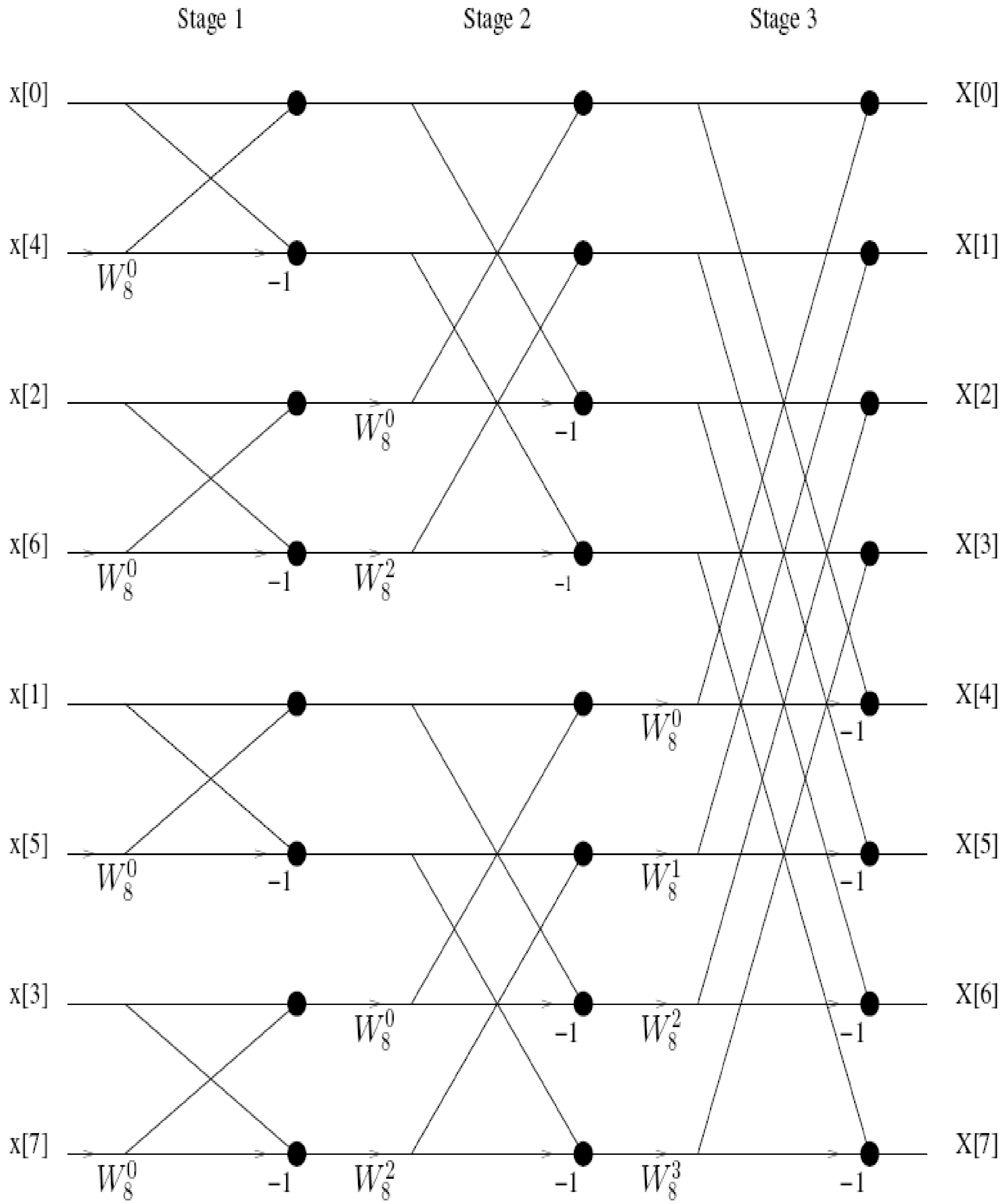



**Fig 4.1 First step in Decimation-in-time domain Algorithm**

$N = 8$-point decimation-in-time FFT algorithm




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

Each dot represents a complex addition.

Each arrow represents a complex multiplication.

### 4.2.2 Decimation-in-frequency Domain

Another important radix-2 FFT algorithm, called decimation-in-frequency algorithm is obtained by using divide-and-conquer approach with the choice of M=2 and L= N/2.This choice of data implies a column-wise storage of the input data sequence. To derive the algorithm, we begin by splitting the DFT formula into two summations, one of which involves the sum over the first N/2 data points and the second sum involves the last N/2 data points. Thus we obtain

$$\begin{aligned} X(k) &= \sum_{n=0}^{(N/2)-1} x(n) W_N^{kn} + \sum_{n=N/2}^{N-1} x(n) W_N^{kn} \\ &= \sum_{n=0}^{(N/2)-1} x(n) W_N^{kn} + W_N^{Nk/2} \sum_{n=0}^{(N/2)-1} x\left(n+\frac{N}{2}\right) W_N^{kn} \end{aligned}$$

Since $W_N^{kN/2} = (-1)^k$, the expression (6.1.33) can be rewritten as

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

Now, let us split X(k) into the even and odd-numbered samples. Thus we obtain

$$X(2k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + x\left(n+\frac{N}{2}\right)\right] W_{N/2}^{kn} \qquad k = 0, 1, \ldots, \frac{N}{2}-1 \qquad (6.1.35)$$

and

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} \left\{\left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^n\right\} W_{N/2}^{kn} \qquad k = 0, 1, \ldots, \frac{N}{2}-1 \qquad (6.1.36)$$

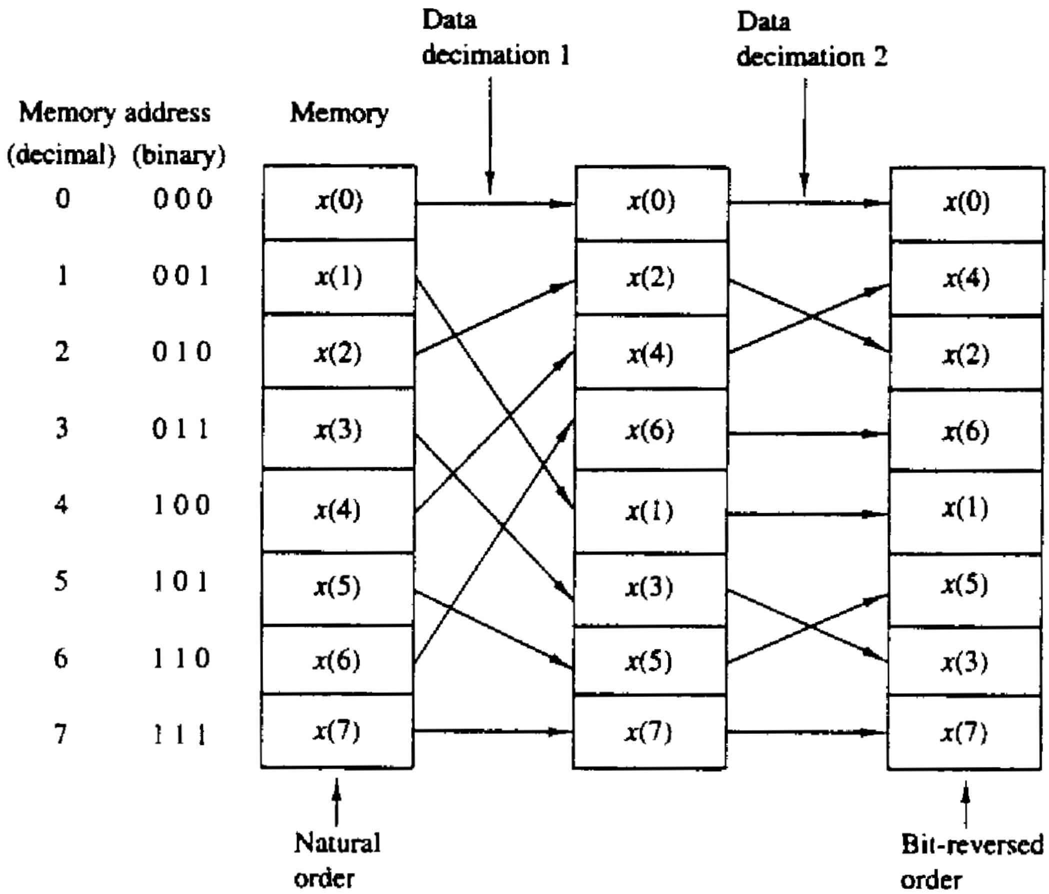
where we have used the fact that $W_N^2 = W_{N/2}$.

If we define the $N/2$-point sequences $g_1(n)$ and $g_2(n)$ as

$$\begin{aligned} g_1(n) &= x(n) + x\left(n+\frac{N}{2}\right) \\ g_2(n) &= \left[x(n) - x\left(n+\frac{N}{2}\right)\right] W_N^n \qquad n = 0, 1, 2, \ldots, \frac{N}{2}-1 \end{aligned}$$

then

$$\begin{aligned} X(2k) &= \sum_{n=0}^{(N/2)-1} g_1(n) W_{N/2}^{kn} \\ X(2k+1) &= \sum_{n=0}^{(N/2)-1} g_2(n) W_{N/2}^{kn} \end{aligned}$$




(a)

| $(n_2 n_1 n_0)$ | → | $(n_0 n_2 n_1)$ | → | $(n_0 n_1 n_2)$ |
|---|---|---|---|---|
| (0 0 0) | → | (0 0 0) | → | (0 0 0) |
| (0 0 1) | → | (1 0 0) | → | (1 0 0) |
| (0 1 0) | → | (0 0 1) | → | (0 1 0) |
| (0 1 1) | → | (1 0 1) | → | (1 1 0) |
| (1 0 0) | → | (0 1 0) | → | (0 0 1) |
| (1 0 1) | → | (1 1 0) | → | (1 0 1) |
| (1 1 0) | → | (0 1 1) | → | (0 1 1) |
| (1 1 1) | → | (1 1 1) | → | (1 1 1) |

(b)

**Fig 4.2 Shuffling of Data and Bit reversal**

The computation of the sequences g1 (n) and g2 (n) and subsequent use of these sequences to compute the N/2-point DFTs depicted in fig we observe that the basic computation in this figure involves the butterfly operation.

The computation procedure can be repeated through decimation of the N/2-point DFTs, X(2k) and X(2k+1). The entire process involves $v = \log_2 N$ of decimation, where each stage involves N/2 butterflies of the type shown in figure 4.3.

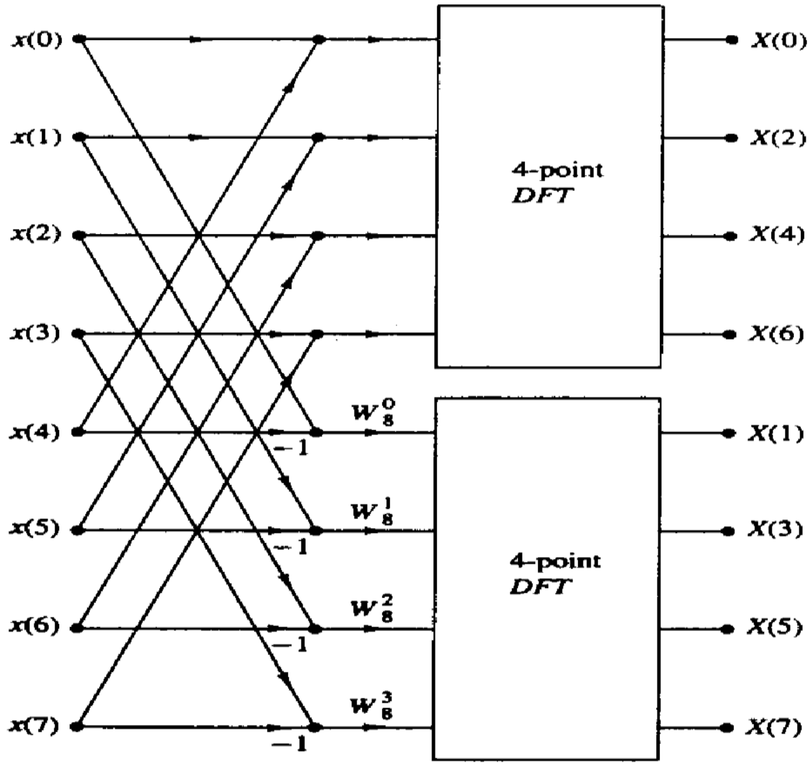



**Fig 4.3 First step in Decimation-in-time domain Algorithm**

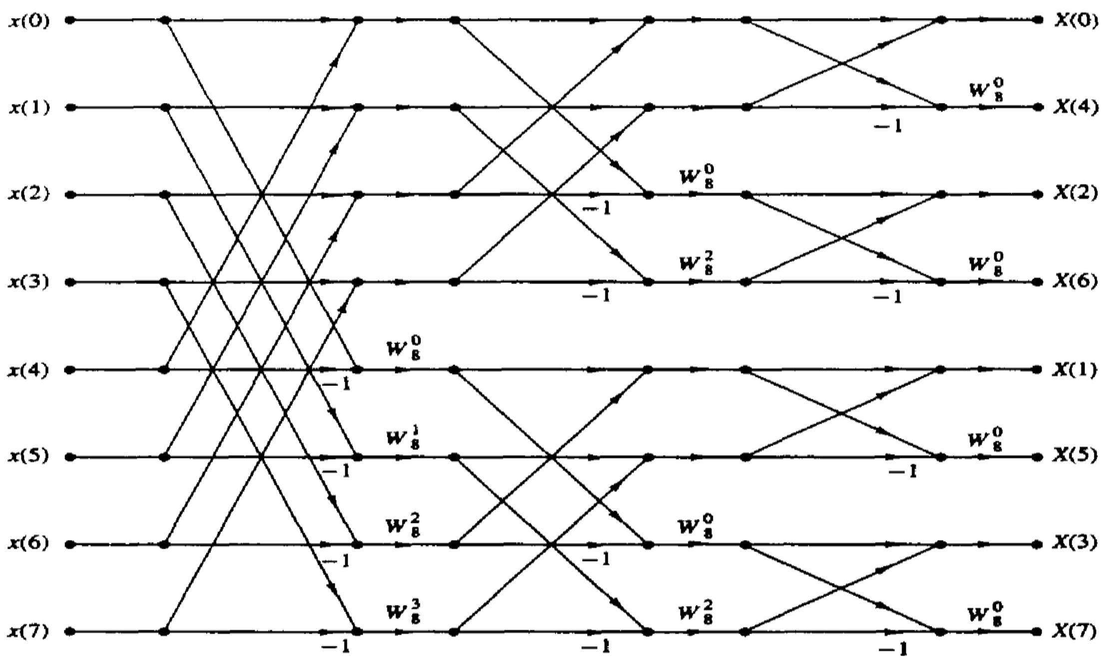
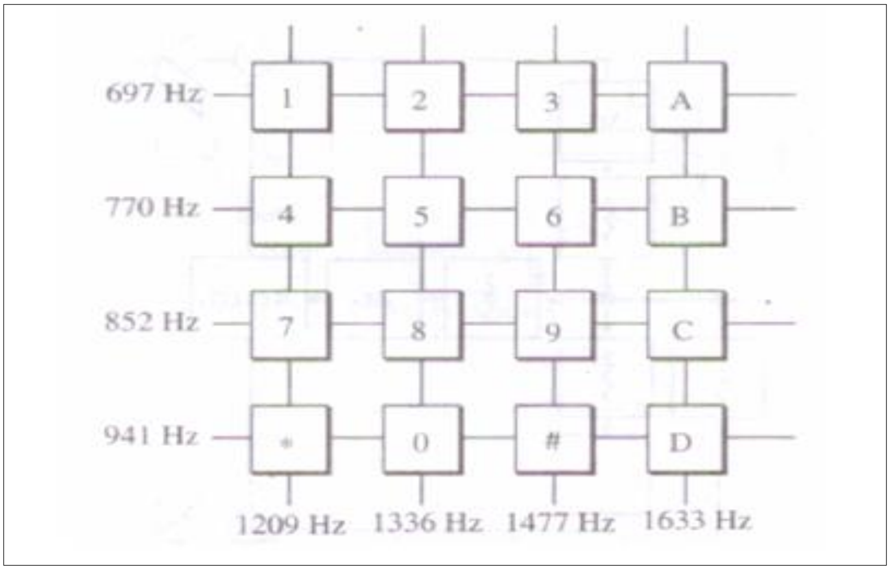



Fig 4.4 N=8 point Decimation-in-frequency domain Algorithm

## 4.2 Example: DTMF – Dual Tone Multi frequency

This is known as touch-tone/speed/electronic dialing, pressing of each button generates a unique set of two-tone signals, called DTMF signals. These signals are processed at exchange to identify the number pressed by determining the two associated tone frequencies. Seven frequencies are used to code the 10 decimal digits and two special characters (4x3 array)



In this application frequency analysis requires determination of possible seven (eight) DTMF fundamental tones and their respective second harmonics .For an 8 kHz sampling freq, the best value of the DFT length N to detect the eight fundamental DTMF tones has been found to be 205 .Not all 205 freq components are needed here, instead only those corresponding to key frequencies are required. FFT algorithm is not effective and efficient in this application. The direct computation of the DFT which is more effective in this application is formulated as a linear filtering operation on the input data sequence.

**This algorithm is known as Goertzel Algorithm**

This algorithm exploits periodicity property of the phase factor. Consider the DFT definition

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

Since $W_N^{-kN}$ is equal to 1, multiplying both sides of the equation by this results in;

$$X(k) = W_N^{-kN} \sum_{m=0}^{N-1} x(m) W_N^{mk} = \sum_{m=0}^{N-1} x(m) W_N^{-k(N-m)} \qquad (2)$$

This is in the form of a convolution $y_k(n) = x(n) * h_k(n)$

$$y_k(n) = \sum_{m=0}^{N-1} x(m) W_N^{-k(n-m)} \qquad (3)$$

$$h_k(n) = W_N^{-kn} u(n) \qquad (4)$$

Where $y_k(n)$ is the out put of a filter which has impulse response of $h_k(n)$ and input x(n).

The output of the filter at n = N yields the value of the DFT at the freq $\omega_k = 2\pi k/N$

The filter has frequency response given by

$$H_k(z) = \frac{1}{1 - W_N^{-k} z^{-1}} \qquad (6)$$

The above form of filter response shows it has a pole on the unit circle at the frequency $\omega_k = 2\pi k/N$.

Entire DFT can be computed by passing the block of input data into a parallel bank of N single-pole filters (resonators)

The above form of filter response shows it has a pole on the unit circle at the frequency $\omega_k = 2\pi k/N$.

Entire DFT can be computed by passing the block of input data into a parallel bank of N single-pole filters (resonators)

### 1.3 Difference Equation implementation of filter:

From the frequency response of the filter (eq 6) we can write the following difference equation relating input and output;

$$H_k(z) = \frac{Y_k(z)}{X(z)} = \frac{1}{1 - W_N^{-k} z^{-1}}$$

$$y_k(n) = W_N^{-k} y_k(n-1) + x(n) \qquad y_k(-1) = 0 \qquad (7)$$

The desired output is $X(k) = y_k(n)$ for k = 0,1,…N-1. The phase factor appearing in the difference equation can be computed once and stored.

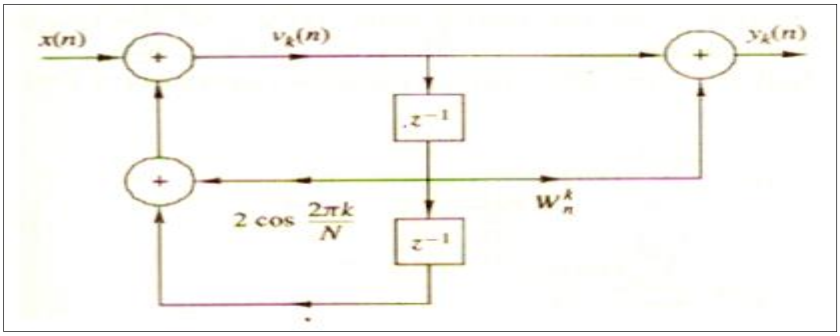
The form shown in eq (7) requires complex multiplications which can be avoided doing suitable modifications (divide and multiply by $1 - W_N^k z^{-1}$). Then frequency response of the filter can be alternatively expressed as

$$H_k(z) = \frac{1 - W_N^k z^{-1}}{1 - 2\cos(2\pi k/N) z^{-1} + z^{-2}} \qquad (8)$$

This is second –order realization of the filter (observe the denominator now is a second-order expression). The direct form realization of the above is given by

$$v_k(n) = 2\cos(2\pi k/N) v_k(n-1) - v_k(n-2) + x(n) \qquad (9)$$

$$y_k(n) = v_k(n) - W_N^k v_k(n-1) \qquad v_k(-1) = v_k(-2) = 0 \qquad (10)$$




The recursive relation in (9) is iterated for $n = 0,1,\ldots\ldots N$, but the equation in (10) is computed only once at time $n = N$. Each iteration requires one real multiplication and two additions. Thus, for a real input sequence x(n) this algorithm requires (N+1) real multiplications to yield X(k) and X(N-k) (this is due to symmetry). Going through the Goertzel algorithm it is clear that this algorithm is useful only when M out of N DFT values need to be computed where $M \leq 2\log_2 N$, Otherwise, the FFT algorithm is more efficient method. The utility of the algorithm completely depends on the application and number of frequency components we are looking for.

## 4.2. Chirp z- Transform

### 4.2.1 Introduction:

Computation of DFT is equivalent to samples of the z-transform of a finite-length sequence at equally spaced points around the unit circle. The spacing between the samples is given by $2\pi/N$. The efficient computation of DFT through FFT requires N to be a highly composite number which is a constraint. Many a times we may need samples of z-transform on contours other than unit circle or we my require dense set of frequency samples over a small region of unit circle. To understand these let us look in to the following situations:

1. Obtain samples of z-transform on a circle of radius ‘a’ which is concentric to unit circle The possible solution is to multiply the input sequence by $a^{-n}$
2. 128 samples needed between frequencies $\omega = -\pi/8$ to $+\pi/8$ from a 128 point sequence

From the given specifications we see that the spacing between the frequency samples is π/512 or 2π/1024. In order to achieve this freq resolution we take 1024- point FFT of the given 128-point seq by appending the sequence with 896 zeros. Since we need only 128 frequencies out of 1024 there will be big wastage of computations in this scheme.

For the above two problems Chirp z-transform is the alternative.

Chirp z- transform is defined as:

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n} \qquad k = 0,1,......L-1 \qquad (11)$$

Where $z_k$ is a generalized contour. $z_k$ is the set of points in the z-plane falling on an arc which begins at some point $z_0$ and spirals either in toward the origin or out away from the origin such that the points $\{z_k\}$are defined as,

$$z_k = r_0 e^{j\theta_0} (R_0 e^{j\phi_0})^k \qquad k = 0,1,....L-1 \qquad (12)$$

<u>Note that,</u>

**a.** if $R_0 < 1$ the points fall on a contour that spirals toward the origin

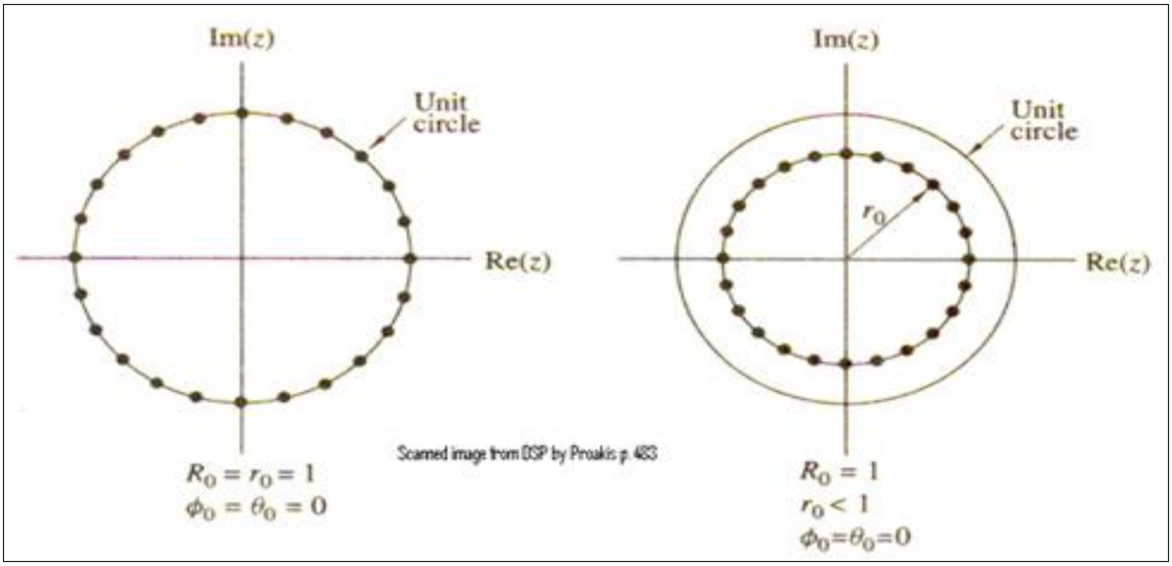
**b**. If $R_0 > 1$ the contour spirals away from the origin

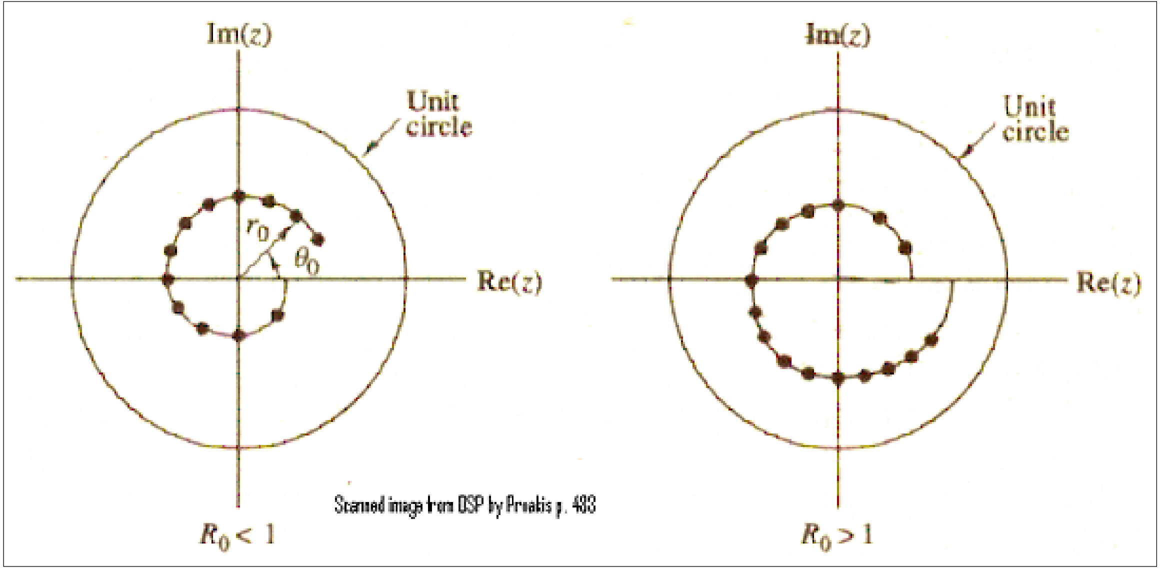
**c**. If $R_0 = 1$ the contour is a circular arc of radius

**d**.If $r_0=1$ and $R_0=1$ the contour is an arc of the unit circle.

(Additionally this contour allows one to compute the freq content of the sequence x(n) at dense set of L frequencies in the range covered by the arc without having to compute a large DFT (i.e., a DFT of the sequence x(n) padded with many zeros to obtain the desired resolution in freq.))

**e.** If $r_0 = R_0 = 1$ and $\theta_0 = 0$ $\Phi_0 = 2\pi/N$ and $L = N$ the contour is the entire unit circle similar to the standard DFT. These conditions are shown in the following diagram.






***Substituting the value of $z_k$ in the expression of $X(z_k)$***

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n} = \sum_{n=0}^{N-1} x(n)(r_0 e^{j\theta_0})^{-n} W^{-nk} \qquad (13)$$

where $\quad W = R_0 e^{j\phi_0} \qquad (14)$

### 4.2.2 Expressing computation of X($z_k$) as linear filtering operation:

By substitution of

$$nk = \frac{1}{2}(n^2 + k^2 - (k-n)^2) \qquad (15)$$

we can express X($z_k$) as

$$X(z_k) = W^{-k^2/2} y(k) = y(k)/h(k) \qquad k = 0,1,\ldots\ldots\ldots L-1 \qquad (16)$$

Where

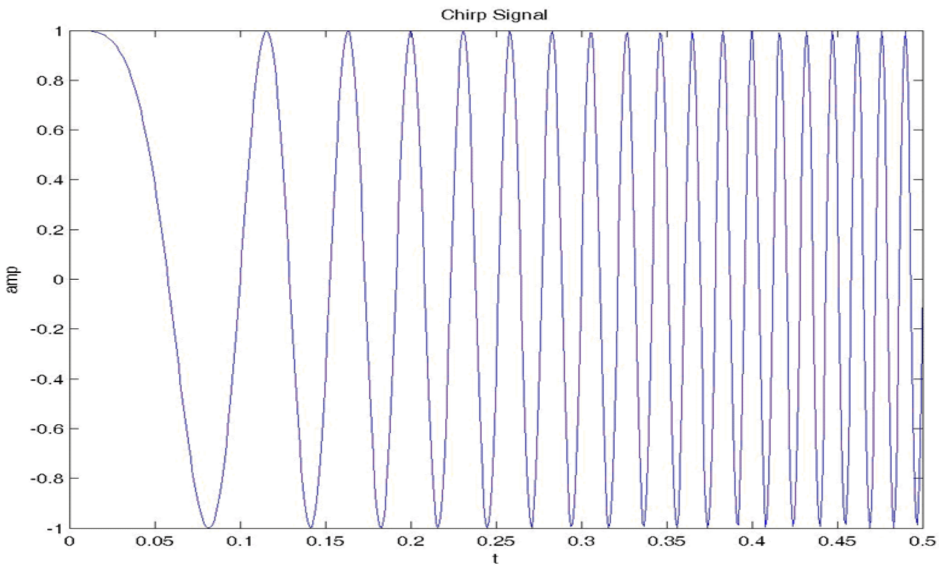
$$h(n) = W^{n^2/2} \qquad g(n) = x(n)(r_0 e^{j\theta_0})^{-n} W^{-n^2/2}$$

$$y(k) = \sum_{n=0}^{N-1} g(n) h(k-n) \qquad (17)$$

both g(n) and h(n) are complex valued sequences

### 4.2.3 Why it is called Chirp z-transform?

If $R_0$ =1, then sequence h(n) has the form of complex exponential with argument ωn = $n^2\Phi_0/2$ = (n $\Phi_0$/2) n. The quantity (n $\Phi_0$/2) represents the freq of the complex exponential signal, which increases linearly with time. Such signals are used in radar systems are called chirp signals. Hence the name chirp z-transform.




**4.2.4 How to Evaluate linear convolution of eq (17)**

1. Can be done efficiently with FFT
2. The two sequences involved are g(n) and h(n). g(n) is finite length seq of length N and h(n) is of infinite duration, but fortunately only a portion of h(n) is required to compute L values of X(z), hence FFT could be still be used.
3. Since convolution is via FFT, it is circular convolution of the N-point seq g(n) with an M- point section of h(n) where M > N

4. The concepts used in overlap –save method can be used

5. While circular convolution is used to compute linear convolution of two sequences we know the initial N-1 points contain aliasing and the remaining points are identical to the result that would be obtained from a linear convolution of h(n) and g(n), In view of this the DFT size selected is M = L+N-1 which would yield L valid points and N-1 points corrupted by aliasing. The section of h(n) considered is for $-(N-1) \le n \le (L-1)$ yielding total length M as defined
6. The portion of h(n) can be defined in many ways, one such way is,

$h_1(n) = h(n-N+1) \quad n = 0,1,\ldots..M-1$

7. Compute $H_1(k)$ and $G(k)$ to obtain

   $Y_1(k) = G(K)H_1(k)$

8. Application of IDFT will give $y_1(n)$, for

   $n = 0,1,\ldots M-1$. The starting N-1 are discarded and desired values are $y_1(n)$ for

   $N-1 \leq n \leq M-1$ which corresponds to the range $0 \leq n \leq L-1$ i.e.,

   $y(n) = y_1(n+N-1) \quad n=0,1,2,\ldots..L-1$

9. Alternatively $h_2(n)$ can be defined as

$$\begin{aligned} h_2(n) &= h(n) & 0 \leq n \leq L-1 \\ &= h(n-(N+L-1)) & L \leq n \leq M-1 \end{aligned}$$

10. Compute $Y_2(k) = G(K)H_2(k)$, The desired values of $y_2(n)$ are in the range

    $0 \leq n \leq L-1$ i.e.,

    $y(n) = y_2(n) \quad n=0,1,\ldots.L-1$

11. Finally, the complex values $X(z_k)$ are computed by dividing $y(k)$ by $h(k)$
    For $k = 0,1,\ldots\ldots L-1$

## 4.3 Computational complexity

In general the computational complexity of CZT is of the order of $M \log_2 M$ complex multiplications. This should be compared with N.L which is required for direct evaluation. If L is small direct evaluation is more efficient otherwise if L is large then CZT is more efficient.

### 4.3.1 Advantages of CZT

**a.** Not necessary to have N =L

**b.**Neither N or L need to be highly composite

**c.**The samples of Z transform are taken on a more general contour that includes the unit circle as a special case.

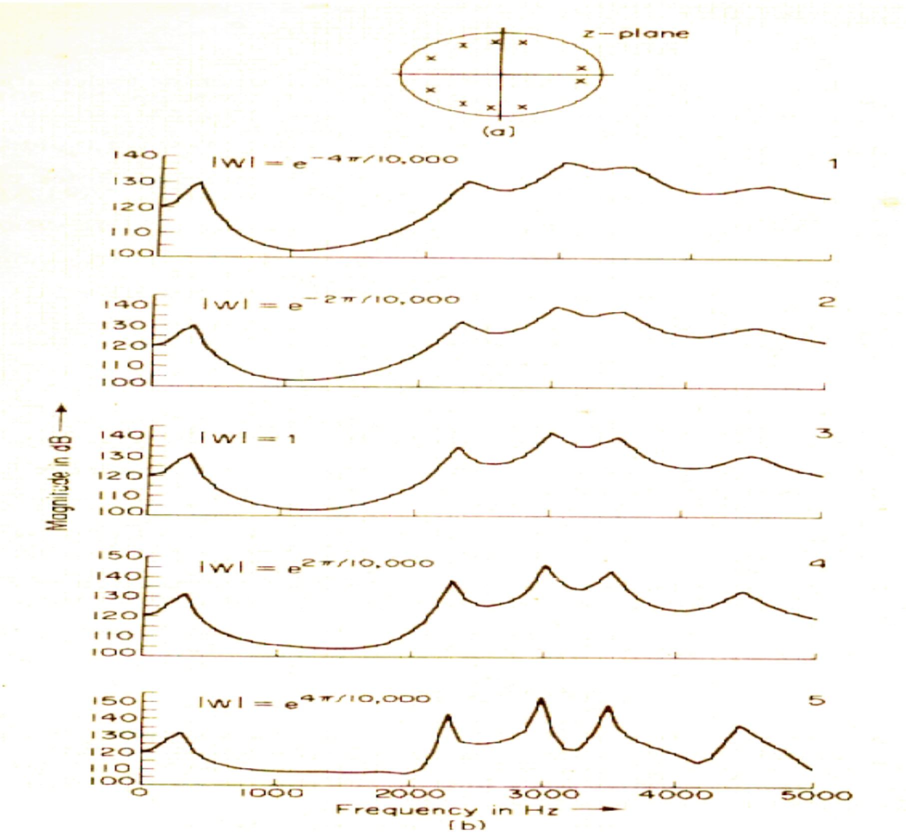
### 4.4 Example to understand utility of CZT algorithm in freq analysis

(ref: DSP by Oppenheim Schaffer)

CZT is used in this application to sharpen the resonances by evaluating the z-transform off the unit circle. Signal to be analyzed is a synthetic speech signal generated by exciting a five-pole system with a periodic impulse train. The system was simulated to correspond to a sampling freq. of 10 kHz. The poles are located at center freqs of 270,2290,3010,3500 & 4500 Hz with bandwidth of 30, 50, 60,87 & 140 Hz respectively.

**Solution**: Observe the pole-zero plots and corresponding magnitude frequency response for different choices of |w|. The following observations are in order:

- The first two spectra correspond to spiral contours outside the unit circle with a resulting broadening of the resonance peaks
- |w| = 1 corresponds to evaluating z-transform on the unit circle
- The last two choices correspond to spiral contours which spirals inside the unit circle and close to the pole locations resulting in a sharpening of resonance peaks.

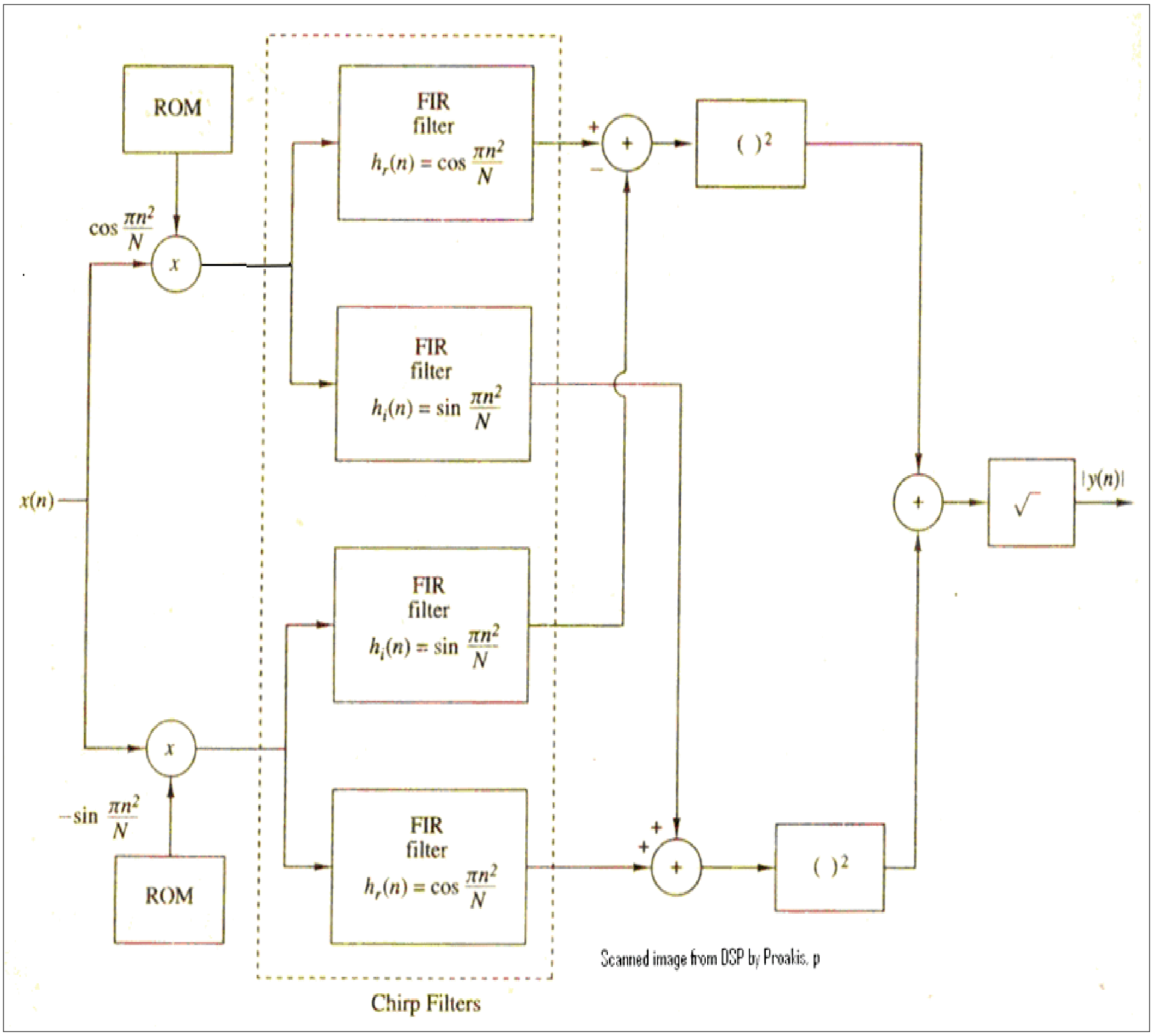


## 4.5 Implementation of CZT in hardware to compute the DFT signals

The block schematic of the CZT hardware is shown in down figure. DFT computation requires $r_0 = R_0 = 1$, $\theta_0 = 0$ $\Phi_0 = 2\pi/N$ and $L = N$.

The cosine and sine sequences in h(n) needed for pre multiplication and post multiplication are usually stored in a ROM. If only magnitude of DFT is desired, the post multiplications are unnecessary,

In this case $|X(z_k)| = |y(k)|$ k =0,1,….N-1



# Recommended Questions with solutions

## Question 1

Compute the 16-point DFT of the sequence

$$x(n) = \cos\frac{\pi}{2}n \qquad 0 \le n \le 15$$

using the radix-4 decimation-in-time algorithm.

**Solution:-**

$$\underline{A} \triangleq \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$\underline{x}_1 \triangleq [\ x(0) \quad x(4) \quad x(8) \quad x(12)\ ]^T$$

$$\underline{x}_2 \triangleq [\ x(1) \quad x(5) \quad x(9) \quad x(13)\ ]^T$$

$$\underline{x}_3 \triangleq [\ x(2) \quad x(6) \quad x(10) \quad x(14)\ ]^T$$

$$\underline{x}_4 \triangleq [\ x(3) \quad x(7) \quad x(11) \quad x(15)\ ]^T$$

$$\begin{bmatrix} F(0) \\ F(4) \\ F(8) \\ F(12) \end{bmatrix} = \underline{A}\underline{x}_1 = \begin{bmatrix} 4 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} F(1) \\ F(5) \\ F(9) \\ F(13) \end{bmatrix} = \underline{A}\underline{x}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} F(2) \\ F(6) \\ F(10) \\ F(14) \end{bmatrix} = \underline{A}\underline{x}_3 = \begin{bmatrix} -4 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} F(3) \\ F(7) \\ F(11) \\ F(15) \end{bmatrix} = \underline{A}\underline{x}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

As every $F(i) = 0$ except $F(0) = -F(2) = 4$,

$$\begin{bmatrix} x(0) \\ x(7) \\ x(8) \\ x(12) \end{bmatrix} = \underline{Ax}_4 \begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \begin{bmatrix} 0 \\ 8 \\ 0 \\ 8 \end{bmatrix}$$

which means that $X(4) = X(12) = 8$. $X(k) = 0$ for other $K$.

## Question 2

Draw the flow graph for the decimation-in-frequency (DIF) SRFFT algorithm for $N = 16$. What is the number of nontrivial multiplications?

**Solution :- There are 20 real , non trial multiplications**

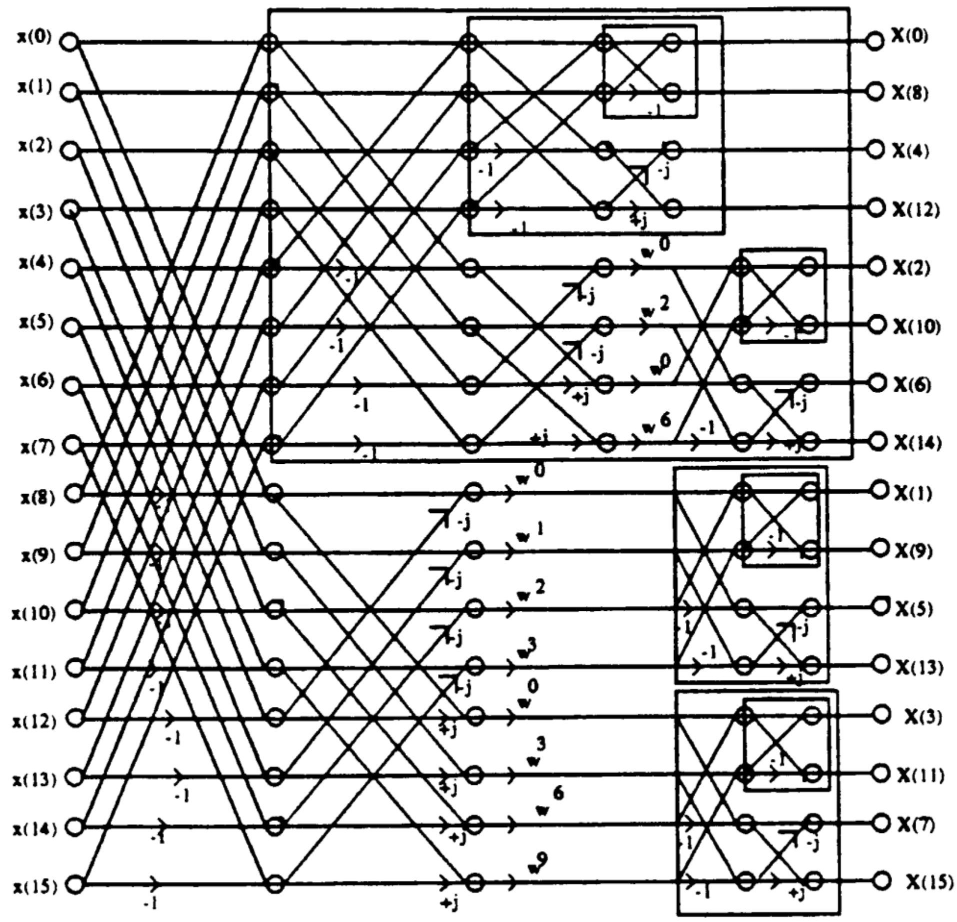



**Figure 4.1 DIF Algorithm for N=16**

## Question 3

Explain how the DFT can be used to compute $N$ equispaced samples of the $z$-transform, of an $N$-point sequence, on a circle of radius $r$.

**Solution:-**

$$X(z) = \sum_{n=0}^{N-1} x(n) z^{-n}$$

$$\text{Hence, } X(z_k) = \sum_{n=0}^{N-1} x(n) r^{-n} e^{-j\frac{2\pi}{N}kn}$$

where $z_k = re^{-j\frac{2\pi}{N}k}, k = 0, 1, \ldots, N-1$ are the N sample points. It is clear that $X(z_k), k = 0, 1, \ldots, N-1$ is equivalent to the DFT (N-pt) of the sequence $x(n)r^{-n}, n \in [0, N-1]$.

## Question 4

Let $X(k)$ be the $N$-point DFT of the sequence $x(n)$, $0 \leq n \leq N-1$. What is the $N$-point DFT of the sequence $s(n) = X(n)$, $0 \leq n \leq N-1$?

**Solution:-**

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

Let $F(t)$, $t = 0, 1, \ldots, N-1$ be the DFT of the sequence on $k$ $X(k)$.

$$\begin{aligned}
F(t) &= \sum_{k=0}^{N-1} X(k) W_N^{tk} \\
&= \sum_{k=0}^{N-1} \left[ \sum_{n=0}^{N-1} x(n) W_N^{kn} \right] W_N^{tk} \\
&= \sum_{n=0}^{N-1} x(n) \left[ \sum_{k=0}^{N-1} W_N^{k(n+t)} \right] \\
&= \sum_{n=0}^{N-1} x(n) \delta(n+t)_{\text{mod N}} \\
&= \sum_{n=0}^{N-1} x(n) \delta(N-1-n-t) \qquad t = 0, 1, \ldots, N-1 \\
&= \{x(N-1), x(N-2), \ldots, x(1), x(0)\}
\end{aligned}$$

## Question 5

Develop a radix-3 decimation-in-time FFT algorithm for $N = 3^v$ and draw the corresponding flow graph for $N = 9$. What is the number of required complex multiplications? Can the operations be performed in place?

**Solution:-**

$$
\begin{aligned}
Y(k) &= \sum_{n=0}^{8} y(n)W_9^{nk} \\
&= \sum_{n=0,3,6} y(n)W_9^{nk} + \sum_{n=1,4,7} y(n)W_9^{nk} + \sum_{n=2,5,8} y(n)W_9^{nk} \\
&= \sum_{m=0}^{2} y(3m)W_9^{3km} + \sum_{m=0}^{2} y(3m+1)W_9^{(3m+1)k} + \sum_{m=0}^{2} y(3m+2)W_9^{(3m+2)k} \\
&= \sum_{m=0}^{2} y(3m)W_3^{km} + \sum_{m=0}^{2} y(3m+1)W_3^{mk}W_9^{k} + \sum_{m=0}^{2} y(3m+2)W_3^{mk}W_9^{2k}
\end{aligned}
$$

## Question 6

Determine the system function $H(z)$ and the difference equation for the system that uses the Goertzel algorithm to compute the DFT value $X(N-k)$.

**Solution:-**

$$
\begin{aligned}
X(k) &= \sum_{m=0}^{N-1} x(m)W_N^{km} \\
&= \sum_{m=0}^{N-1} x(m)W_N^{km}W_N^{-kN} \quad \text{since } W_N^{-kN} = 1 \\
&= \sum_{m=0}^{N-1} x(m)W_N^{-k(N-m)}
\end{aligned}
$$

**This can be viewed as the convolution of the N-length sequence x(n) with implulse response of a linear filter**

$$
\begin{aligned}
h_k(n) &\triangleq W_N^{kn}u(n), \text{ evaluated at time N} \\
H_k(z) &= \sum_{n=0}^{\infty} W_N^{kn}z^{-n}
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{1 - W_N^k z^{-1}} \\
&= \frac{Y_u(z)}{X(z)} \\
y_k(n) &= W_N^k y_k(n-1) + x(n), \qquad y_k(-1) = 0 \\
y_k(N) &= X(k)
\end{aligned}
$$

# UNIT 5

## IIR FILTER DESIGN

## CONTENTS:-

**IIR FILTER DESIGN: CHARACTERISTICS OF COMMONLY USED ANALOG FILTERS – BUTTERWORTH AND CHEBYSHEVE FILTERS, ANALOG TO ANALOG FREQUENCY TRANSFORMATIONS.** **6** HRS

## RECOMMENDED READINGS


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# Unit 5
# Design of IIR Filters

## 5.1 Introduction

A digital filter is a linear shift-invariant discrete-time system that is realized using finite precision arithmetic. The design of digital filters involves three basic steps:

- ❖ The specification of the desired properties of the system.
- ❖ The approximation of these specifications using a causal discrete-time system.
- ❖ The realization of these specifications using finite precision arithmetic.

These three steps are independent; here we focus our attention on the second step. The desired digital filter is to be used to filter a digital signal that is derived from an analog signal by means of periodic sampling. The specifications for both analog and digital filters are often given in the frequency domain, as for example in the design of low pass, high pass, band pass and band elimination filters.

Given the sampling rate, it is straight forward to convert from frequency specifications on an analog filter to frequency specifications on the corresponding digital filter, the analog frequencies being in terms of Hertz and digital frequencies being in terms of radian frequency or angle around the unit circle with the point Z=-1 corresponding to half the sampling frequency. The least confusing point of view toward digital filter design is to consider the filter as being specified in terms of angle around the unit circle rather than in terms of analog frequencies.

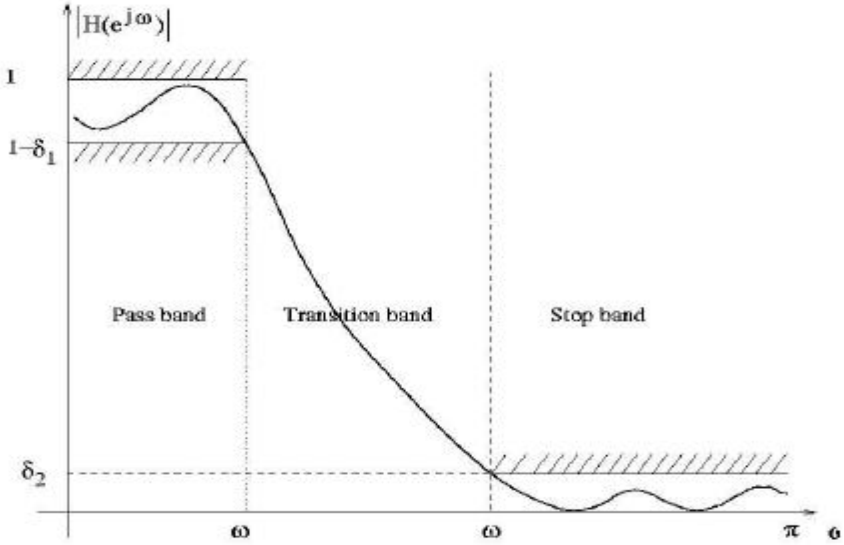



Figure 5.1: Tolerance limits for approximation of ideal low-pass filter

A separate problem is that of determining an appropriate set of specifications on the digital filter. In the case of a low pass filter, for example, the specifications often take the form of a tolerance scheme, as shown in Fig. 5.1.

$$1 - \delta_1 \ \leq | \ H(e^{j\omega}) \ | \leq 1, \quad | \ \omega \ | \leq \omega_p$$

$$| \ H(e^{j\omega}) \ | \leq \delta_2, \ \ \omega_s \leq | \ \omega \ | \leq \pi$$

Many of the filters used in practice are specified by such a tolerance scheme, with no constraints on the phase response other than those imposed by stability and causality requirements; i.e., the poles of the system function must lie inside the unit circle. Given a set of specifications in the form of Fig. 5.1, the next step is to and a discrete time linear system whose frequency response falls within the prescribed tolerances. At this point the filter design problem becomes a problem in approximation. In the case of infinite impulse response (IIR) filters, we must approximate the desired frequency response by a rational function, while in the finite impulse response (FIR) filters case we are concerned with polynomial approximation.

## 5.1 Design of IIR Filters from Analog Filters:

The traditional approach to the design of IIR digital filters involves the transformation of an analog filter into a digital filter meeting prescribed specifications. This is a reasonable approach because:

- ❖ The art of analog filter design is highly advanced and since useful results can be achieved, it is advantageous to utilize the design procedures already developed for analog filters.
- ❖ Many useful analog design methods have relatively simple closed-form design formulas.

Therefore, digital filter design methods based on analog design formulas are rather simple to implement. An analog system can be described by the differential equation

$$\sum_{k=0}^{N} c_k \frac{d^k y_a(t)}{dt^k} = \sum_{k=0}^{M} d_k \frac{d^k x_a(t)}{dt^k}$$

And the corresponding rational function is

$$H_a(s) = \frac{\sum_{k=0}^{M} d_k s^k}{\sum_{k=0}^{N} c_k s^k} = \frac{y_a(s)}{x_a(s)}$$

The corresponding description for digital filters has the form

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k)$$

and the rational function

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}} = \frac{Y(z)}{X(z)}$$

In transforming an analog filter to a digital filter we must therefore obtain either H(z) or h(n) (inverse Z-transform of H(z) i.e., impulse response) from the analog filter design. In such transformations, we want the imaginary axis of the S-plane to map into the nit circle of the Z-plane, a stable analog filter should be transformed to a stable digital filter. That is, if the analog filter has poles only in the left-half of S-plane, then the digital filter must have poles only inside the unit circle. These constraints are basic to all the techniques discussed here.

## 5.2 Characteristics of Commonly Used Analog Filters:

From the previous discussion it is clear that, IIT digital filters can be obtained by beginning with an analog filter. Thus the design of a digital filter is reduced to designing an appropriate analog filter and then performing the conversion from Ha(s) to H (z). Analog filter design is a well - developed field, many approximation techniques, viz., Butterworth, Chebyshev, Elliptic, etc., have been developed for the design of analog low pass filters. Our discussion is limited to low pass filters, since, frequency transformation can be applied to transform a designed low pass filter into a desired high pass, band pass and band stop filters.

### 5.2.1 Butterworth Filters:

Low pass Butterworth filters are all - pole filters with monotonic frequency response in both pass band and stop band, characterized by the magnitude - squared frequency response

$$|H_a(\Omega)|^2 = \frac{1}{1+(\Omega/\Omega_c)^{2N}} = \frac{1}{1+\epsilon^2(\Omega/\Omega_p)^{2N}}$$

Where, N is the order of the filter, Ώc is the -3dB frequency, i.e., cutoff frequency, Ώp is the pass band edge frequency and 1= (1 /1+ε$^2$ ) is the band edge value of $|Ha(\Omega)|^2$. Since the product Ha(s) Ha(-s) and evaluated at s = jΩ is simply equal to $|Ha(\Omega)|^2$, it follows that

$$H_a(s)H_a(-s) = \frac{1}{1+(\frac{-s^2}{\Omega_c^2})^N}$$

The poles of Ha(s)Ha(-s) occur on a circle of radius Ώc at equally spaced points. From Eq. (5.29), we find the pole positions as the solution of

$$\frac{-s^2}{\Omega_c^2} = (-1)^{1/N} = e^{j(2k+1)\pi/N}, \quad k = 0, 1, , \cdots, N-1$$

And hence, the N poles in the left half of the s-plane are

$$\begin{aligned} s_k &= \Omega_c e^{j\pi/2} e^{(2k+1)\pi/2N}, \quad k = 0, 1, \cdots, N-1 \\ &= \sigma_k + j\Omega_k \end{aligned}$$

Note that, there are no poles on the imaginary axis of s-plane, and for N odd there will be a pole on real axis of s-plane, for N even there are no poles even on real axis of s-plane. Also note that all the poles are having conjugate symmetry. Thus the design methodology to design a Butterworth low pass filter with δ2 attenuation at a specified frequency Ωs is Find N,

$$N = \frac{\log[(1/\delta_2^2) - 1]}{2\log(\Omega_s/\Omega_c)} = \frac{\log(\delta/\epsilon)}{\log(\Omega_s/\Omega_p)}$$

Where by definition, δ2 = $1/\sqrt{1+\delta^2}$. Thus the Butterworth filter is completely characterized by the parameters N, δ2, ε and the ratio Ωs/Ωp or Ωc.Then, from Eq. (5.31) find the pole positions Sk; k = 0,1, 2,……..(N-1). Finally the analog filter is given by

$$H_a(s) = \prod_{k==1}^{N} \frac{1}{(s - s_k)}$$

## 5.2.2 Chebyshev Filters:

There are two types of Chebyshev filters. Type I Chebyshev filters are all-pole filters that exhibit equiripple behavior in the pass band and a monotonic characteristic in the stop band. On the other hand, type II Chebyshev filters contain both poles and zeros and exhibit a monotonic behavior in the pass band and an equiripple behavior in the stop band. The zeros of this class of filters lie on the imaginary axis in the s-plane. The magnitude squared of the frequency response characteristic of type I Chebyshev filter is given as

$$| H_a(\Omega) |^2 = \frac{1}{1 + \epsilon^2 T_N^2(\Omega/\Omega_p)}$$

Where ε is a parameter of the filter related to the ripple in the pass band as shown in Fig. (5.7), and $T_N$ is the Nth order Chebyshev polynomial defined as

$$T_N(x) = \begin{cases} \cos(N\cos^{-1}x), & |x| \leq 1 \\ \cosh(N\cosh^{-1}x), & |x| > 1 \end{cases}$$

The Chebyshev polynomials can be generated by the recursive equation

$$T_{N+1}(x) == 2xT_N(x) - T_{N-1}(x), \quad N = 1, 2, \cdots$$

Where $T_0(x) = 1$ and $T_1(x) = x$.

At the band edge frequency $\Omega = \Omega_p$, we have

$$\frac{1}{\sqrt{1+\epsilon^2}} = 1 - \delta_1$$

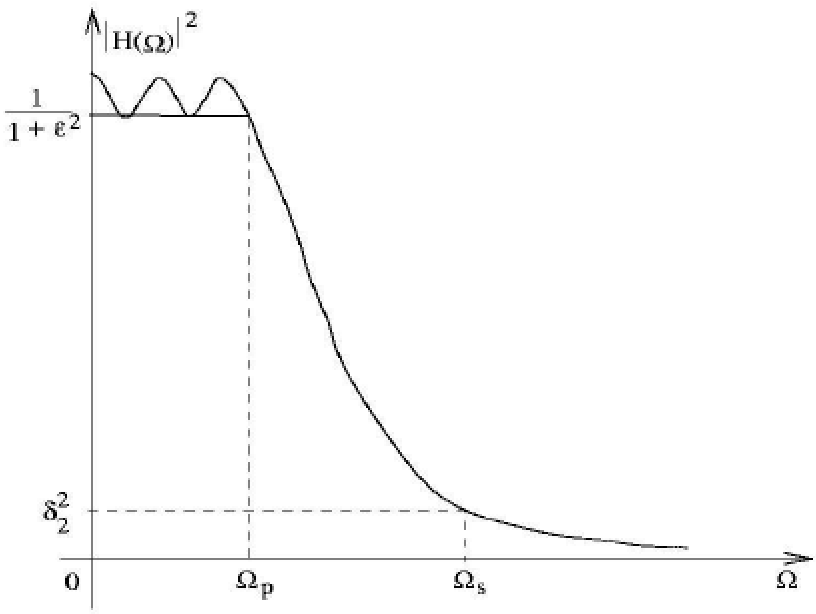



**Figure 5.2**: Type I Chebysehev filter characteristic

Or equivalently

$$\epsilon^2 = \frac{1}{(1-\delta_1)^2} - 1$$

Where $\delta_1$ is the value of the pass band ripple.

The poles of Type I Chebyshev filter lie on an ellipse in the s-plane with major axis

$$r_1 = \Omega_p \frac{\beta^2 + 1}{2\beta}$$

And minor axis

$$r_2 = \Omega_p \frac{\beta^2 - 1}{2\beta}$$

Where β is related to ε according to the equation

$$\beta = [\frac{\sqrt{1+\epsilon^2}+1}{\epsilon}]^{1/N}$$

The angular positions of the left half s-plane poles are given by

$$\phi_k = \frac{\pi}{2} + \frac{(2k+1)\pi}{2N}, \quad k = 0, 1, \cdots, N-1$$

Then the positions of the left half s-plane poles are given by

$$s_k = \sigma_k + j\Omega_k, \quad k = 0, 1, \cdots, N-1$$

Where $\sigma_k = r_2 \text{ Cos } \varphi k$ and $\Omega_k = r_1 \text{ Sin}\varphi k$. The order of the filter is obtained from

$$\begin{aligned} N &= \frac{\log[(\sqrt{1-\delta_2^2}+\sqrt{1-\delta_2^2(1+\epsilon^2)})/\epsilon\delta_2]}{\log[\frac{\Omega_s}{\Omega_p}+\sqrt{(\frac{\Omega_s}{\Omega_p})^2-1}]} \\ &= \frac{\cosh^{-1}(\frac{\delta}{\epsilon})}{\cosh^{-1}(\frac{\Omega_s}{\Omega_p})} \end{aligned}$$

Where, by definition $\delta_2 = 1/\sqrt{1+\delta^2}$.

Finally, the Type I Chebyshev filter is given by

$$H_a(s) = \prod_{k=1}^{N} \frac{1}{(s-s_k)}$$

A Type II Chebyshev filter contains zero as well as poles. The magnitude squared response is given as

$$| H_a(-\Omega) |^2 = \frac{1}{1+\epsilon^2[T_N^2(\frac{\Omega_s}{\Omega_p})/T_N^2(\frac{\Omega_s}{\Omega})]}$$

Where $T_N(x)$ is the N-order Chebyshev polynomial. The zeros are located on the imaginary axis at the points

$$z_k = j\frac{\Omega_s}{\sin\phi_k}, \ k = 0, 1, \ldots, N-1$$

and the left-half s-plane poles are given

$$s_k = \sigma_k + j\Omega_k, \ k = 0, 1, \ldots, N-1$$

Where

$$\sigma_k = \frac{\Omega_s r_2 \cos\phi_k}{\sqrt{r_2^2 \cos^2\phi_k + r_1^2 \sin^2\phi_k}}$$

and

$$\Omega_k = \frac{\Omega_s r_1 \sin\phi_k}{\sqrt{r_2^2 \cos^2\phi_k + r_1^2 \sin^2\phi_k}}$$

Finally, the Type II Chebyshev filter is given by

$$H_a(s) = \prod_{k=1}^{N} \frac{s - z_k}{s - s_k}$$

The other approximation techniques are elliptic (equiripple in both passband and stopband) and Bessel (monotonic in both passband and stopband).

## 5.3 Analog to Analog Frequency Transforms

Frequency transforms are used to transform lowpass prototype filter to other filters like highpass or bandpass or bandstop filters. One possibility is to perform frequency transform in the analog domain and then convert the analog filter into a corresponding digital filter by a mapping of the s-plane into z-plane. An alternative approach is to convert the analog lowpass filter into a lowpass digital filter and then to transform the lowpass digital filter into the desired digital filter by a digital transformation.

Suppose we have a lowpass filter with pass edge $\Omega_P$ and if we want convert that into another lowpass filter with pass band edge $\Omega'_P$ then the transformation used is

$$s \longrightarrow \frac{\Omega_p}{\Omega'_p} s \qquad \text{(lowpass to lowpass)}$$

Thus we obtain a lowpass filter with system function $H_l(s) = H_p[(\Omega_p/\Omega'_p)s]$, where $H_p(s)$ is the system function of the prototype filter with passband edge frequency $\Omega_p$.

To convert low pass filter into highpass filter the transformation used is

$$s \longrightarrow \frac{\Omega_p \Omega'_p}{s} \qquad \text{(lowpass to highpass)}$$

$s$

The system function of the highpass filter is $H_h(s) = H_p(\Omega_p \Omega'_p / s)$.

The transformation for converting a lowpass analog filter with passband edge frequency $\Omega_p$ into a band filter, having a lower band edge frequency $\Omega_l$ and an upper band edge frequency $\Omega_u$, can be accomplished by first converting the lowpass

$$s \longrightarrow \frac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)} \qquad \text{(lowpass to bandpass)}$$

Thus we obtain

$$H_b(s) = H_p\left(\Omega_p \frac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)}\right)$$

Finally, if we wish to convert a lowpass analog filter with band edge frequency $\Omega_p$ into a bandstop filter, the transformation is simply the inverse of (8.4.3) with the additional factor $\Omega_p$ serving to normalize for the band edge frequency of the lowpass filter. Thus the transformation is

$$s \longrightarrow \Omega_p \frac{s(\Omega_u - \Omega_l)}{s^2 + \Omega_u \Omega_l} \qquad \text{(lowpass to bandstop)}$$

The filter function is

$$H_{bs}(s) = H_p\left(\Omega_p \frac{s(\Omega_u - \Omega_l)}{s^2 + \Omega_u \Omega_l}\right)$$

# Recommended Questions with answers

**Question 1**

I Design a digital filter to satisfy the following characteristics.

- -3dB cutoff frequency of 0:5_ rad.
- Magnitude down at least 15dB at 0:75_ rad.
- Monotonic stop band and pass band Using
- Impulse invariant technique
- Approximation of derivatives
- Bilinear transformation technique

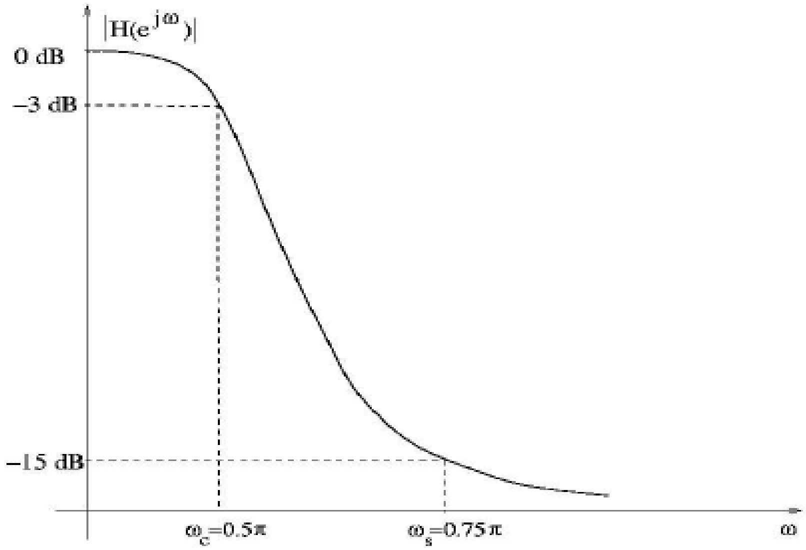



**Figure 5.8: Frequency response plot of the example**

**Solution:-**

a) Impulse Invariant Technique

From the given digital domain frequency, _nd the corresponding analog domain frequencies.

$\Omega_c = \frac{\omega_c}{T}$ and $\Omega_s = \frac{\omega_s}{T}$

Where T is the sampling period and 1/T is the sampling frequency and it always corresponds to 2Π radians in the digital domain. In this problem, let us assume T = 1sec.

Then Ώc = 0:5Π and Ώs = 0:75Π

Let us find the order of the desired filter using

$$N = \frac{(\frac{1}{\delta_2^2} - 1)}{2\log(\frac{\Omega_s}{\Omega_c})}$$

Where $\delta_2$ is the gain at the stop band edge frequency ωs.

$$-15 \text{ dB} = 20 \log \delta_2$$

$$\delta_2 = 20 \log \delta_2$$

$$\delta_2 = 10^{-\frac{\pi}{20}} = 0.1778$$

$$N = \frac{\log(\frac{1}{(0.1778)^2} - 1}{2\log(\frac{0.75\pi}{0.5\pi})} = 4.219 \simeq 5$$

Order of filter N =5.

Then the 5 poles on the Butterworth circle of radius Ώc = 0:5 Π are given by

$$\begin{aligned}
s_0 &= 0.5\pi e^{j(\frac{\pi}{2}+\frac{\pi}{10})} = -0.485 + j1.493 \\
s_1 &= 0.5\pi e^{j(\frac{\pi}{2}+\frac{3\pi}{10})} = -1.27 + j0.923 \\
s_2 &= 0.5\pi e^{j(\frac{\pi}{2}+\frac{5\pi}{10})} = -1.57 + j0.0 \\
s_3 &= 0.5\pi e^{j(\frac{\pi}{2}+\frac{7\pi}{10})} = -1.27 - j0.923 \\
s_5 &= 0.5\pi e^{j(\frac{\pi}{2}+\frac{9\pi}{10})} = -0.485 - j1.493
\end{aligned}$$

Then the filter transfer function in the analog domain is

$$\begin{aligned}
H_a(s) &= \frac{1}{(s+0.485-j1.493)(s+1.27-j0.923)(s+1.57)(s+1.27+j0.923)(s+0.485+j0.923)} \\
&= \sum_{k=1}^{5} \frac{A_k}{(s-s_k)}
\end{aligned}$$

where $A_k$'s are partial fractions coefficients of Ha(s).

Finally, the transfer function of the digital filter is

$$H(z) = \sum_{k=1}^{5} \frac{A_k}{(1-e^{s_k}z^{-1})}, \text{ where } s_k\text{'s are the poles on the Butterworth circle}$$

**b)**

$$H(z) = H_a(s)\,|_{s=\frac{1-z^{-1}}{T}=1-z^{-1}}$$

$$H(z) = \sum_{k=1}^{5} \frac{1}{(1-z^{-1}-s_k)}$$

**c)** For the bilinear transformation technique, we need to pre-warp the digital frequencies into corresponding analog frequencies.

i.e., $\Omega = \frac{2}{T}\tan(\frac{\omega}{2})$

$$\Omega_c = 2\tan(\frac{0.5\pi}{2}) = 2 \text{ rad.}$$

and

$$\Omega_s = 2\tan(\frac{0.75\pi}{2}) = 4.828 \text{ rad.}$$

Then the order of the filter

$$N = \frac{\log(\frac{1}{(0.1778)^2} - 1)}{2\log(\frac{4.828}{2})}$$

The pole locations on the Butterworth circle with radius $\Omega$c = 2 are

$$s_0 = 2e^{j(\frac{\pi}{2}+\frac{\pi}{4})} = -1.414 + j1.414$$

$$s_1 = 2e^{j(\frac{\pi}{2}+\frac{3\pi}{4})} = -1.414 - j1.414$$

Then the filter transfer function in the analog domain is

$$H_a(s) = \frac{1}{(s+1.414-j1.414)(s+1.414+j1.414)}$$

Finally, the transfer function of the digital filter is

$$H(z) = H_a(s)\,\Big|_{s=\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}=2\frac{1-z^{-1}}{1+z^{-1}}}$$

$$H(z) = \frac{1}{(2\frac{1-z^{-1}}{1+z^{-1}} + 1.414 - j1.414)(2\frac{1-z^{-1}}{1+z^{-1}} + 1.414 + j1.414)}$$

**<u>Question 2</u>**

**Design a digital filter using impulse invariant technique to satisfy following characteristics**

(i) Equiripple in pass band and monotonic in stop band

(ii) -3dB ripple with pass band edge frequency at 0:5Π radians.

(iii) Magnitude down at least 15dB at 0:75 Π radians.

**Solution**: Assuming T=1, Ώ= 0:5 Π and s = 0:75 Π

The order of desired filter is

$$N = \frac{\log[(\sqrt{1-\delta_2^2}) + \sqrt{1-\delta_2^2(1-\epsilon^2)})/\epsilon\delta_2]}{\log[\frac{\Omega_s}{\Omega_p} + \sqrt{(\frac{\Omega_s}{\Omega_p})^2 - 1}]}$$

when

$$20\log\frac{1}{\sqrt{1+\epsilon^2}} = -3|mboxdB$$

i.e.,

$$10\log(1+\epsilon^2) = 3dB$$

$$\epsilon^2 = 10^{0.3} - 1 = 0.9952$$

$$\epsilon = 0.9976$$

and

$$20\log\delta_2 = -15 \text{ dB}$$

$$\delta_2 = 10^{-0.75} = 0.1778$$

Hence

$$\begin{aligned} N &= \frac{[(\sqrt{1-(0.1778)^2}+\sqrt{1-(0.1778)^2(1+0.9952)})/0.9976\times 0.1778]}{\log[\frac{0.75\pi}{0.5\pi}+\sqrt{(\frac{0.75\pi}{0.5\pi})^2-1}]} \\ &= 2.48 \\ &\simeq 3 \end{aligned}$$

The order of filter, N = 3.

The 3 poles on the ellipse are determined by

$$\beta = [\frac{\sqrt{1+\epsilon^2}+1}{\epsilon}]^{\frac{1}{N}} = [\frac{\sqrt{1+0.9976^2}+1}{0.9976}]^{\frac{1}{3}} = 1.342$$

$$\begin{aligned} r_1 &= \Omega_p \frac{\beta^2+1}{2\beta} \\ &= 0.5\pi \times \frac{(1.341)^2+1}{2\times 1.341} \\ &= 1.639 \end{aligned}$$

$$\begin{aligned} r_2 &= \Omega_p \frac{\beta^2-1}{2\beta} \\ &= 0.5\pi \times \frac{(1.341)^2-1}{2\times 1.341} \\ &= 0.469 \end{aligned}$$

The angles,

$$\phi_k = \frac{\pi}{2} + \frac{(2k+1)\pi}{2N},\ \text{k = 0,1 ,2}$$

The poles are at

$$s_k = r_2\cos\phi_k + jr_1\sin\phi_k$$

$$
\begin{aligned}
s_0 &= 0.469\cos(\frac{4\pi}{6}) + j1.639\sin(\frac{4\pi}{6}) \\
&= -0.2345 + j1.419
\end{aligned}
$$

$$
\begin{aligned}
s_1 &= 0.469\cos(\pi) + j1.639\sin(\pi) \\
&= -0.469 + j0.0
\end{aligned}
$$

$$
\begin{aligned}
s_2 &= 0.469\cos(\frac{8\pi}{6}) + j1.6939\sin(\frac{8\pi}{6}) \\
&= -0.2345 - j1.419
\end{aligned}
$$

The analog filter transfer function is given by

$$
\begin{aligned}
H_a(s) &= \frac{1}{s + 0.2345 - j1.419)(s + 0.469) + (s + 0.2345 + j1.419)} \\
&= \sum_{k=1}^{3} \frac{A_k}{(s - s_k)}
\end{aligned}
$$

where $A_k$'s are the partial fraction coefficients.

Finally, the digital filter transfer function is given by

$$
H(z) = \sum_{k=1}^{3} \frac{A_k}{(1 - e^{s_k} z^{-1})}
$$

**Question 3**

An IIR digital low-pass filter is required to meet the following specifications:

Passband ripple (or peak-to-peak ripple): $\leq 0.5$ dB

Passband edge: 1.2 kHz

Stopband attenuation: $\geq 40$ dB

Stopband edge: 2.0 kHz

Sample rate: 8.0 kHz

Use the design formulas in the book to determine the required filter order for

**(a)** A digital Butterworth filter

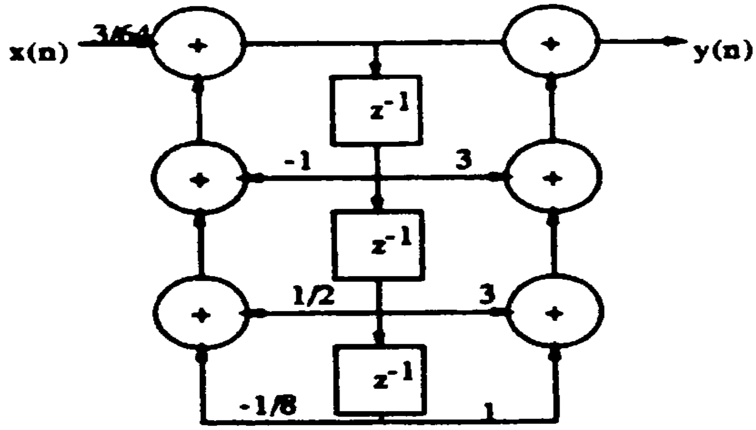
**(b)** A digital Chebyshev filter

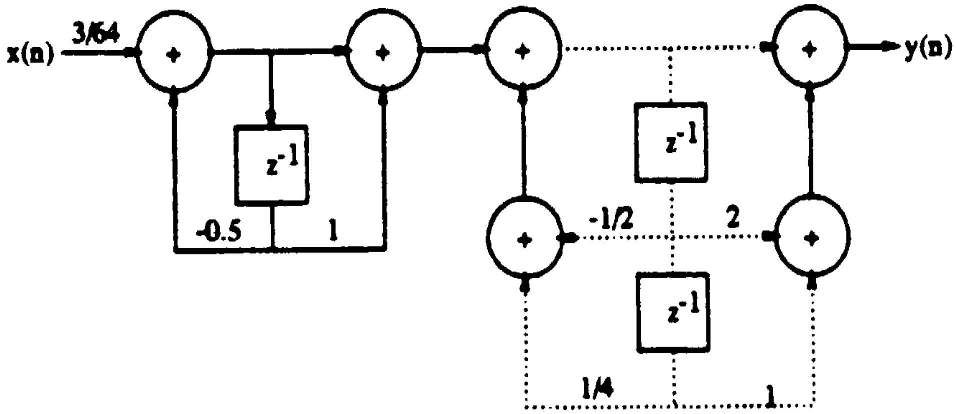
**(c)** A digital elliptic filter

**Solution:-**

**For the design specifications we have**

$$\begin{aligned}
\epsilon &= 0.349 \\
\delta &= 99.995 \\
f_p &= \frac{1.2}{8} = 0.15 \\
f_s &= \frac{2}{8} = 0.25 \\
\Omega_p &= 2tan\frac{w_p}{2} = 1.019 \\
\Omega_s &= 2tan\frac{w_s}{2} = 2 \\
\eta &= \frac{\delta}{\epsilon} = 286.5 \\
k &= \frac{\Omega_s}{\Omega_p} = 1.963
\end{aligned}$$






$$\text{Butterworth filter: } N_{\min} \geq \frac{\log\eta}{\log k} = 8.393 \Rightarrow N = 9$$

$$\text{Chebyshev filter: } N_{\min} \geq \frac{\cosh^{-1}\eta}{\cosh^{-1}k} = 4.90 \Rightarrow N = 5$$

$$\text{Elliptic filter: } N_{\min} \geq \frac{k(\frac{1}{k})}{k(\sqrt{1-\frac{1}{k^2}})} \cdot \frac{k(\sqrt{1-\frac{1}{\eta^2}})}{k(\frac{1}{\eta})} \Rightarrow N = 4$$

## Question 4

Determine the system function $H(z)$ of the lowest-order Chebyshev digital filter that meets the following specifications:

**(a)** $\frac{1}{2}$-dB ripple in the passband $0 \leq |\omega| \leq 0.24\pi$.

**(b)** At least 50-dB attenuation in the stopband $0.35\pi \leq |\omega| \leq \pi$. Use the bilinear transformation.

**Solution:-**

Passband ripple = 0.5dB $\Rightarrow \epsilon = 0.349$
Stopband attenuation = 50dB

$$\begin{aligned}
w_p &= 0.24\pi \\
w_s &= 0.35\pi \\
\Omega_p &= 2\tan\frac{w_p}{2} = 0.792 \\
\Omega_s &= 2\tan\frac{w_s}{2} = 1.226 \\
\eta &= \frac{\delta}{\epsilon} = 906.1 \\
k &= \frac{\Omega_s}{\Omega_p} = 1.547 \\
N_{\min} &\geq \frac{\cosh^{-1}\eta}{\cosh^{-1}k} = \frac{7.502}{1.003} = 7.48 \Rightarrow N = 8
\end{aligned}$$

# UNIT 6
# Implementation of Discrete time systems

## CONTENTS:-

**IMPLEMENTATION OF DISCRETE-TIME SYSTEMS: STRUCTURES FOR IIR AND FIR SYSTEMS DIRECT FORM I AND DIRECT FORM II SYSTEMS, CASCADE, LATTICE AND PARALLEL REALIZATION.** 7 HRS

## RECOMMENDED READINGS:-


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# UNIT 6
# Implementation of Discrete-Time Systems

## 6.1 Introduction

The two important forms of expressing system leading to different realizations of FIR & IIR filters are

a) Difference equation form

$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + \sum_{k=1}^{M} b_k x(n-k)$$

b) Ration of polynomials

$$H(Z) = \frac{\sum_{k=0}^{M} b_k Z^{-k}}{1 + \sum_{k=1}^{N} a_k Z^{-k}}$$

The following factors influence choice of a specific realization,

- Computational complexity
- Memory requirements
- Finite-word-length
- Pipeline / parallel processing

### 6.1.1 Computation Complexity

This is do with number of arithmetic operations i.e. multiplication, addition & divisions. If the realization can have less of these then it will be less complex computationally.

In the recent processors the fetch time from memory & number of times a comparison between two numbers is performed per output sample is also considered and found to be important from the point of view of computational complexity.

### 6.1.2 Memory requirements

This is basically number of memory locations required to store the system parameters, past inputs, past outputs, and any intermediate computed values. Any realization requiring less of these is preferred.

### 6.1.3 Finite-word-length effects

These effects refer to the quantization effects that are inherent in any digital implementation of the system, either in hardware or in software. No computing system has infinite precision. With finite precision there is bound to be errors. These effects are basically to do with truncation & rounding-off of samples. The extent of this effect varies with type of arithmetic used(fixed or floating). The serious issue is that the effects have influence on system characteristics. A structure which is less sensitive to this effect need to be chosen.

### 6.1.4 Pipeline / Parallel Processing

This is to do with suitability of the structure for pipelining & parallel processing. The parallel processing can be in software or hardware. Longer pipelining make the system more efficient.

## 6.2 Structure for FIR Systems:

FIR system is described by,

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

Or equivalently, the system function

$$H(Z) = \sum_{k=0}^{M-1} b_k Z^{-k}$$

Where we can identify $h(n) = \begin{cases} b_n & 0 \le n \le n-1 \\ 0 & otherwise \end{cases}$

Different FIR Structures used in practice are,

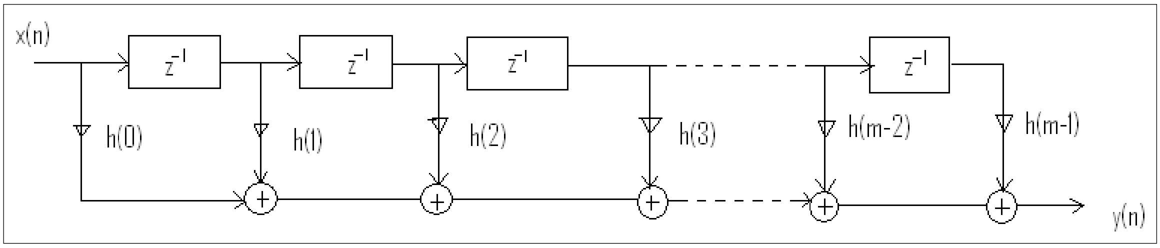
1. Direct form
2. Cascade form
3. Frequency-sampling realization
4. Lattice realization

### 6.2.1 Direct – Form Structure

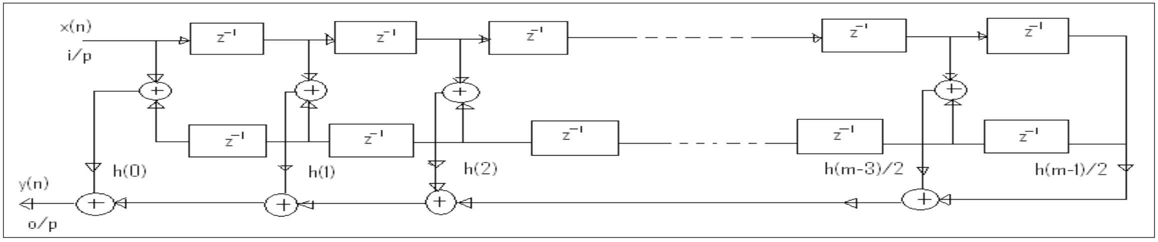
Convolution formula is used to express FIR system given by,

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

- It is Non recursive in structure




- As can be seen from the above implementation it requires M-1 memory locations for storing the M-1 previous inputs
- It requires computationally M multiplications and M-1 additions per output point
- It is more popularly referred to as tapped delay line or transversal system
- Efficient structure with linear phase characteristics are possible where $h(n) = \pm h(M-1-n)$




**Prob:**

**Realize the following system function using minimum number of multiplication**

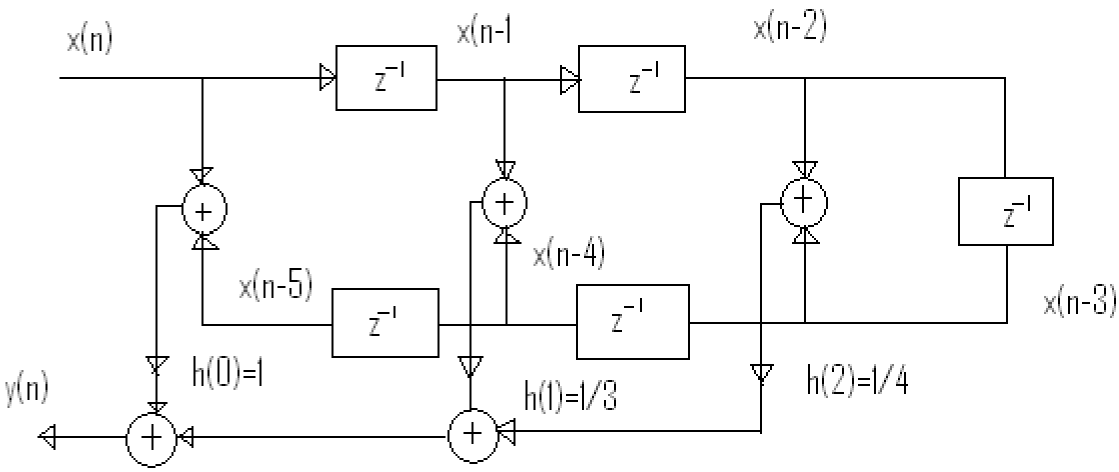
(1) $H(Z) = 1 + \frac{1}{3}Z^{-1} + \frac{1}{4}Z^{-2} + \frac{1}{4}Z^{-3} + \frac{1}{3}Z^{-4} + Z^{-5}$

We recognize $h(n) = \left[1, \frac{1}{3}, \frac{1}{4}, \frac{1}{4}, \frac{1}{3}, 1\right]$

M is even = 6, and we observe h(n) = h(M-1-n)        h(n) = h(5-n)

i.e h(0) = h(5)        h(1) = h(4)        h(2) = h(3)

Direct form structure for Linear phase FIR can be realized




Exercise: Realize the following using system function using minimum number of multiplication.

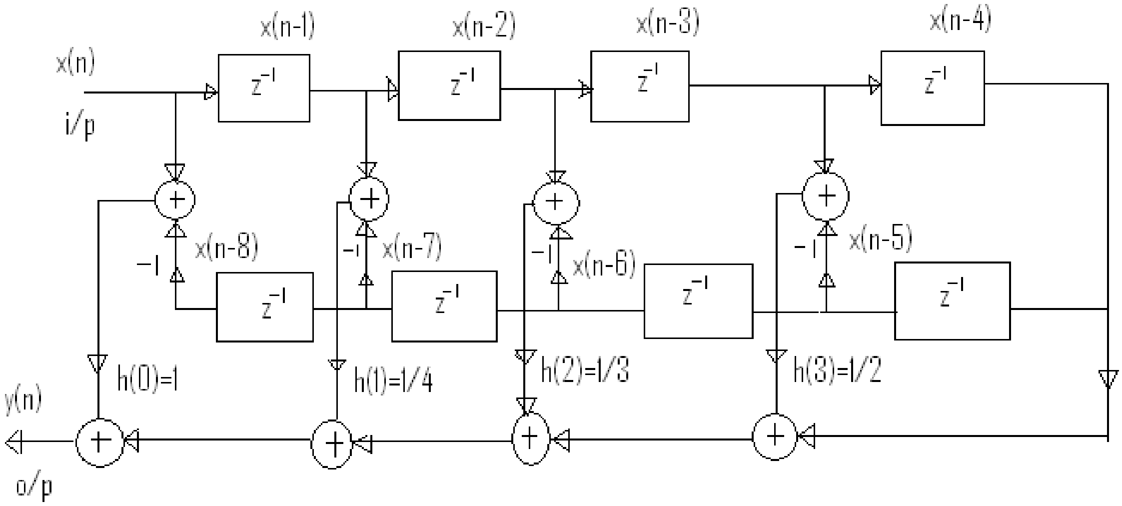
$$H(Z) = 1 + \frac{1}{4}Z^{-1} + \frac{1}{3}Z^{-2} + \frac{1}{2}Z^{-3} - \frac{1}{2}Z^{-5} - \frac{1}{3}Z^{-6} - \frac{1}{4}Z^{-7} - Z^{-8}$$

m=9        $h(n) = \left[1, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{3}, -\frac{1}{4}, -1\right]$

odd symmetry

h(n) = -h(M-1-n);        h(n) = -h(8-n);        h(m-1/2) = h(4) = 0

h(0) = -h(8);        h(1) = -h(7);    h(2) = -h(6);    h(3) = -h(5)




## 6.2.2 Cascade – Form Structure

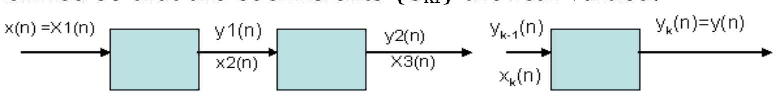
The system function H(Z) is factored into product of second – order FIR system

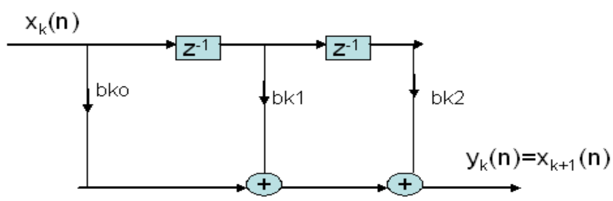
$$H(Z) = \prod_{k=1}^{K} H_k(Z)$$

Where $H_k(Z) = b_{k0} + b_{k1}Z^{-1} + b_{k2}Z^{-2}$ k = 1, 2, ….. K

and K = integer part of (M+1) / 2

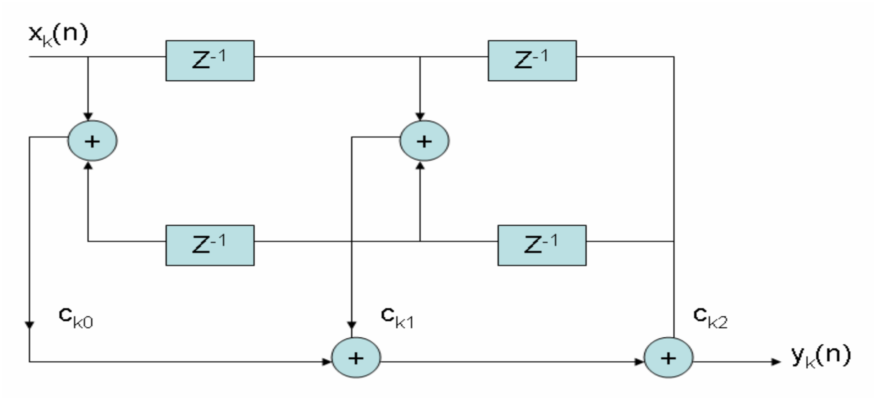
The filter parameter $b_0$ may be equally distributed among the K filter section, such that $b_0 = b_{10}\, b_{20} \ldots.\, b_{k0}$ or it may be assigned to a single filter section. The zeros of H(z) are grouped in pairs to produce the second – order FIR system. Pairs of complex-conjugate roots are formed so that the coefficients $\{b_{ki}\}$ are real valued.



In case of linear –phase FIR filter, the symmetry in h(n) implies that the zeros of H(z) also exhibit a form of symmetry. If zk and zk* are pair of complex – conjugate zeros then 1/zk and 1/zk* are also a pair complex –conjugate zeros. Thus simplified fourth order sections are formed. This is shown below,

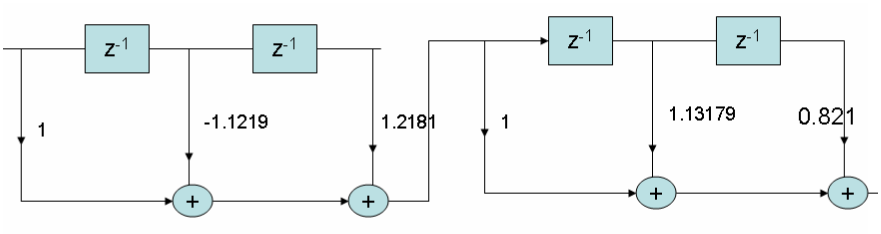
$$H_k(z) = C_{k0}(1 - z_k z^{-1})(1 - z_k * z^{-1})(1 - z^{-1}/z_k)(1 - z^{-1}/z_{k*})$$
$$= C_{k0} + C_{k1}z^{-1} + C_{k2}z^{-2} + C_{k1}z^{-3} + z^{-4}$$




**Problem: Realize the difference equation**
$y(n) = x(n) + 0.25x(n-1) + 0.5x(n-2) + 0.75x(n-3) + x(n-4)$
in cascade form.

Soln:

$$Y(z) = X(z)\{1 + 0.25z^{-1} + 0.5z^{-2} + 0.75z^{-3} + z^{-4})$$
$$H(z) = 1 + 0.25z^{-1} + 0.5z^{-2} + 0.75z^{-3} + z^{-4}$$
$$H(z) = (1 - 1.1219z^{-1} + 1.2181z^{-2})(1 + 1.3719z^{-1} + 0.821z^{-2})$$
$$H(z) = H_1(z)H_2(z)$$



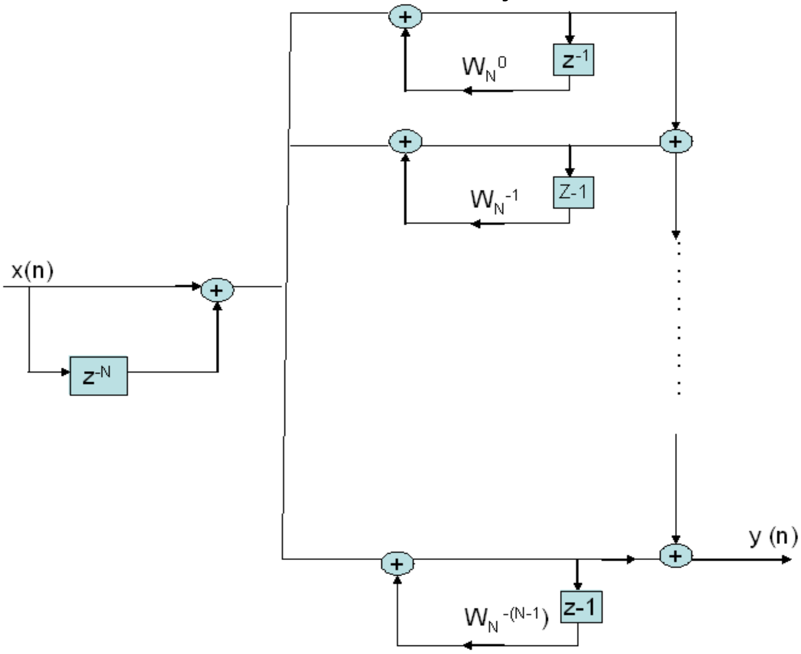

## 6.3 Frequency sampling realization:

We can express system function H(z) in terms of DFT samples H(k) which is given by

$$H(z) = (1 - z^{-N})\frac{1}{N}\sum_{k=0}^{N-1}\frac{H(k)}{1 - W_N^{-k}z^{-1}}$$

This form can be realized with cascade of FIR and IIR structures. The term $(1-z^{-N})$ is realized as FIR and the term $\frac{1}{N}\sum_{k=0}^{N-1}\frac{H(k)}{1-W_N^{-k}z^{-1}}$ as IIR structure.

The realization of the above freq sampling form shows necessity of complex arithmetic. Incorporating symmetry in h(n) and symmetry properties of DFT of real sequences the realization can be modified to have only real coefficients.




## 6.4 Lattice structures

Lattice structures offer many interesting features:

1. Upgrading filter orders is simple. Only additional stages need to be added instead of redesigning the whole filter and recalculating the filter coefficients.
2. These filters are computationally very efficient than other filter structures in a filter bank applications (eg. Wavelet Transform)
3. Lattice filters are less sensitive to finite word length effects.

Consider

$$H(z)=\frac{Y(z)}{X(z)}=1+\sum_{i=1}^{m}a_m(i)z^{-i}$$

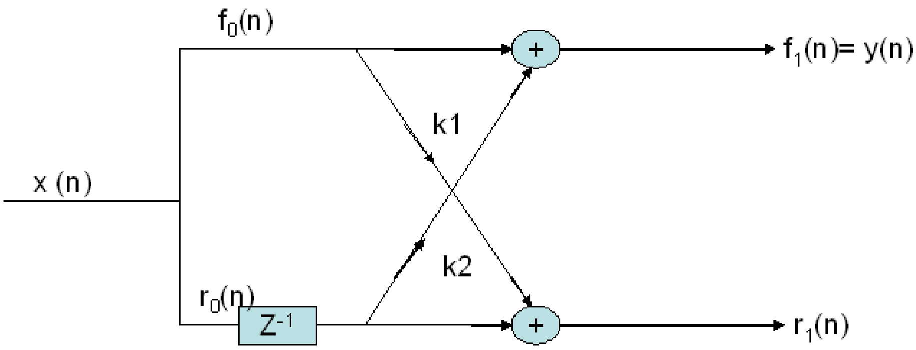
m is the order of the FIR filter and $a_m(0)=1$

when m = 1 $Y(z)/X(z) = 1+ a_1(1)\, z^{-1}$

$y(n)= x(n)+ a_1(1)x(n-1)$

f1(n) is known as upper channel output and r1(n)as lower channel output.

$f_0(n)= r_0(n)=x(n)$




The outputs are

$$f_1(n) = f_0(n) + k_1 r_0(n-1) \qquad 1a$$
$$r_1(n) = k_1 f_0(n) + r_0(n-1) \qquad 1b$$
$$if \quad k_1 = a_1(1), then \quad f_1(n) = y(n)$$

If m=2

$$\frac{Y(z)}{X(z)} = 1 + a_2(1)z^{-1} + a_2(2)z^{-2}$$
$$y(n) = x(n) + a_2(1)x(n-1) + a_2(2)x(n-2)$$
$$y(n) = f_1(n) + k_2 r_1(n-1) \qquad (2)$$

Substituting 1a and 1b in (2)

$$y(n) = f_0(n) + k_1 r_0(n-1) + k_2[k_1 f_0(n-1) + r_0(n-2)]$$
$$= f_0(n) + k_1 r_0(n-1) + k_2 k_1 f_0(n-1) + k_2 r_0(n-2)]$$
$$\sin ce \quad f_0(n) = r_0(n) = x(n)$$
$$y(n) = x(n) + k_1 x(n-1) + k_2 k_1 x(n-1) + k_2 x(n-2)]$$
$$= x(n) + (k_1 + k_1 k_2) x(n-1) + k_2 x(n-2)$$

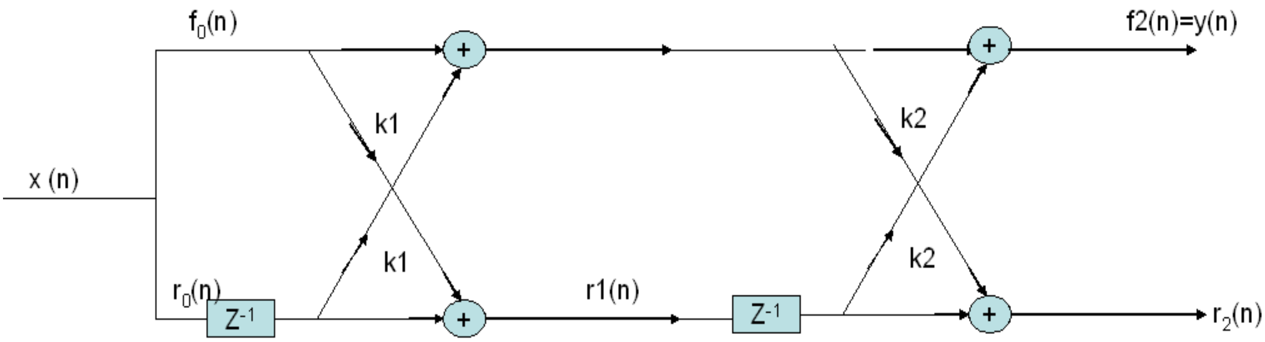
We recognize

$$a_2(1) = k_1 + k_1 k_2$$
$$a_2(1) = k_2$$

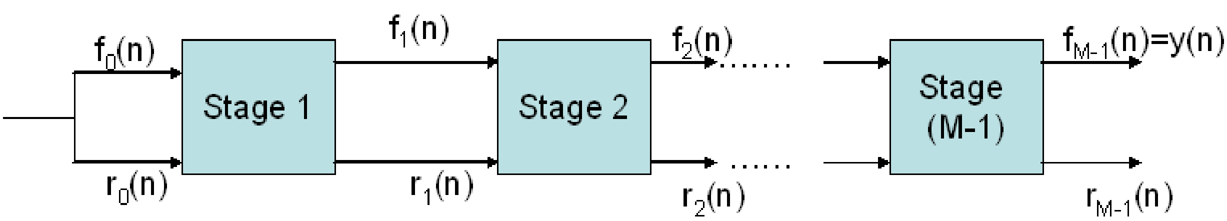
Solving the above equation we get

$$k_1 = \frac{a_2(1)}{1 + a_2(2)} \quad and \quad k_2 = a_2(2) \qquad (4)$$

Equation (3) means that, the lattice structure for a second-order filter is simply a cascade of two first-order filters with k1 and k2 as defined in eq (4)




Similar to above, an Mth order FIR filter can be implemented by lattice structures with M – stages




## 8.4.1 Direct Form –I to lattice structure

For m = M, M-1, ………..2, 1 do

$$k_m = a_m(m)$$

$$a_{m-1}(i) = \frac{a_m(i) - a_m(m)a_m(m-i)}{1 - k_m^2} \qquad 1 \le i \le m-1$$

- The above expression fails if $k_m$=1. This is an indication that there isa zero on the unit circle. If $k_m$=1, factor out this root from A(z) and the recursive formula can be applied for reduced order system.

$$for\ m = 2\ and\ m = 1$$
$$k_2 = a_2(2) \quad \& \quad k_1 = a_1(1)$$

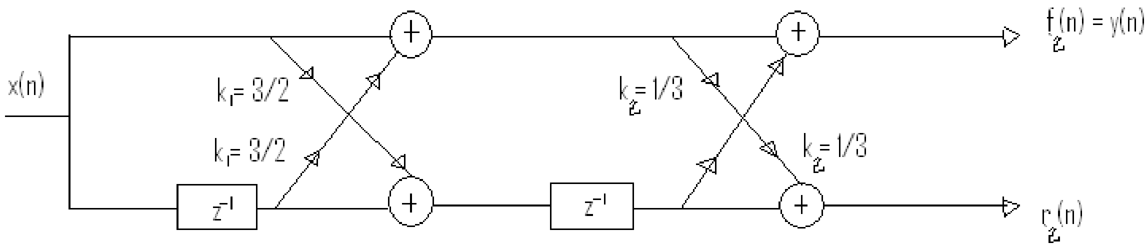
$$for\ m = 2\ \&\ i = 1$$
$$a_1(1) = \frac{a_2(1) - a_2(2)a_2(1)}{1 - k_2^2} = \frac{a_2(1)[1 - a_2(2)]}{1 - a_2^2(2)} = \frac{a_2(1)}{1 + a_2(2)}$$
$$Thus \quad k_1 = \frac{a_2(1)}{1 + a_2(2)}$$

8.4.2 Lattice to direct form –I

For m = 1,2,…….M-1

$$a_m(0) = 1$$
$$a_m(m) = k_m$$
$$a_m(i) = a_{m-1}(i) + a_m(m)a_{m-1}(m-i) \quad 1 \le i \le m-1$$

Problem:
Given FIR filter $H(Z) = 1 + 2Z^{-1} + \frac{1}{3}Z^{-2}$ obtain lattice structure for the same
Given $a_1(1) = 2$, $a_2(2) = 1/3$
Using the recursive equation for
m = M, M-1, ……, 2, 1
here M=2 therefore m = 2, 1
if m=2 $k_2 = a_2(2) = 1/3$
if m=1 $k_1 = a_1(1)$
also, when m=2 and i=1
$$a_1(1) = \frac{a_2(1)}{1 + a_2(2)} = \frac{2}{1 + 1/3} = \frac{3}{2}$$
Hence $k_1 = a_1(1) = 3/2$



# Recommended questions with solution

**Problem:1**

Consider an FIR lattice filter with co-efficients $k_1 = \frac{1}{2}$, $k_2 = \frac{1}{3}$, $k_3 = \frac{1}{4}$. Determine the FIR filter co-efficient for the direct form structure ($H(Z) = a_3(0) + a_3(1)Z^{-1} + a_3(2)Z^{-2} + a_3(3)Z^{-3}$)

$a_3(0) = 1$ $\quad a_3(3) = k_3 = 1/4$ $\quad a_2(2) = k_2 = \frac{1}{3}$

$a_1(1) = k_1 = \frac{1}{2}$

for m=2, i=1

$$a_2(1) = a_1(1) + a_2(2)a_1(1)$$

$$= a_1(1)[1 + a_2(2)] = \frac{1}{2}\left[1 + \frac{1}{3}\right]$$

$$= \frac{4}{6} = \frac{2}{3}$$

for m=3, i=1

$$a_3(1) = a_2(1) + a_3(3)a_2(2)$$

$$= \frac{2}{3} + \frac{1}{4}.\frac{1}{3}$$

$$= \frac{2}{3} + \frac{1}{12} = \frac{8+1}{12}$$

$$= \frac{9}{12} = \frac{3}{4}$$

for m=3 & i=2

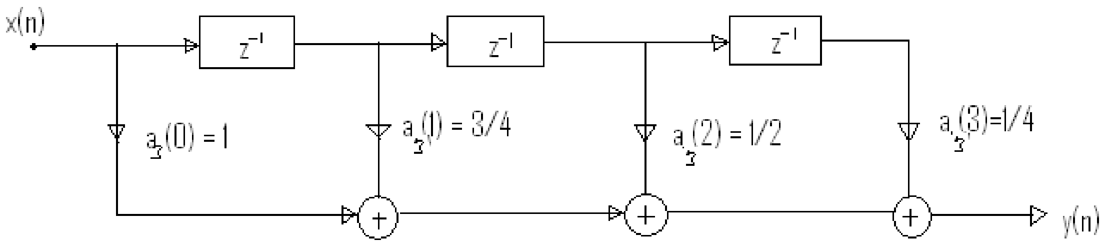
$$a_3(2) = a_2(2) + a_3(3)a_2(1)$$

$$= \frac{1}{3} + \frac{1}{4}.\frac{2}{3}$$

$$= \frac{1}{3} + \frac{1}{6} = \frac{2+1}{6}$$

$$= \frac{3}{6} = \frac{1}{2}$$

$$a_3(0) = 1, \qquad a_3(1) = \frac{3}{4}, \qquad a_3(2) = \frac{1}{2}, \qquad a_3(3) = \frac{1}{4}$$




## 6.5 Structures for IIR Filters

The IIR filters are represented by system function;

$$H(Z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}}$$

and corresponding difference equation given by,

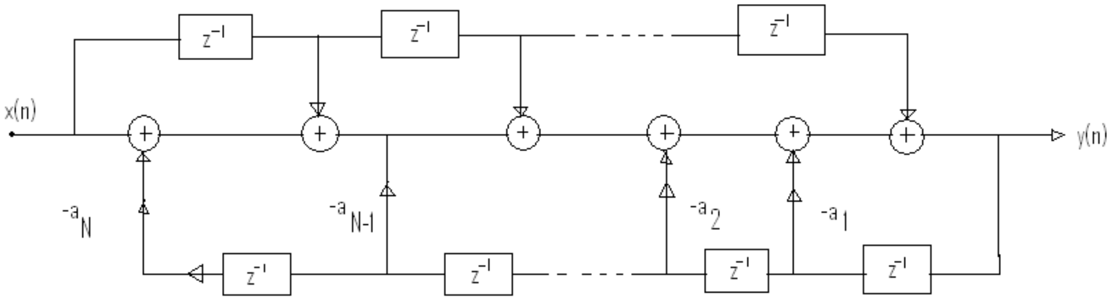
$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + \sum_{k=0}^{N} b_k x(n-k)$$

Different realizations for IIR filters are,

1. Direct form-I
2. Direct form-II
3. Cascade form
4. Parallel form
5. Lattice form

### 6.5.1 Direct form-I

This is a straight forward implementation of difference equation which is very simple. Typical Direct form – I realization is shown below . The upper branch is forward path and lower branch is feedback path. The number of delays depends on presence of most previous input and output samples in the difference equation.




## 6.5.2 Direct form-II

The given transfer function H(z) can be expressed as,

$$H(z) = \frac{Y(z)}{X(z)} = \frac{V(z)}{X(z)} \cdot \frac{Y(z)}{V(z)}$$

where V(z) is an intermediate term. We identify,

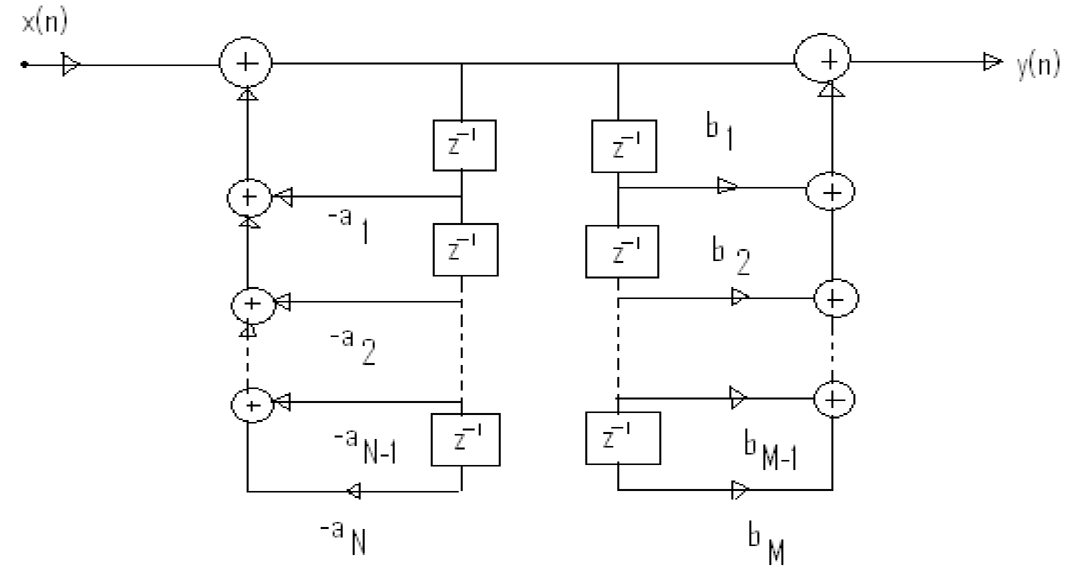
$$\frac{V(z)}{X(z)} = \frac{1}{1+\sum_{k=1}^{N} a_k z^{-k}}$$ -------------------all poles

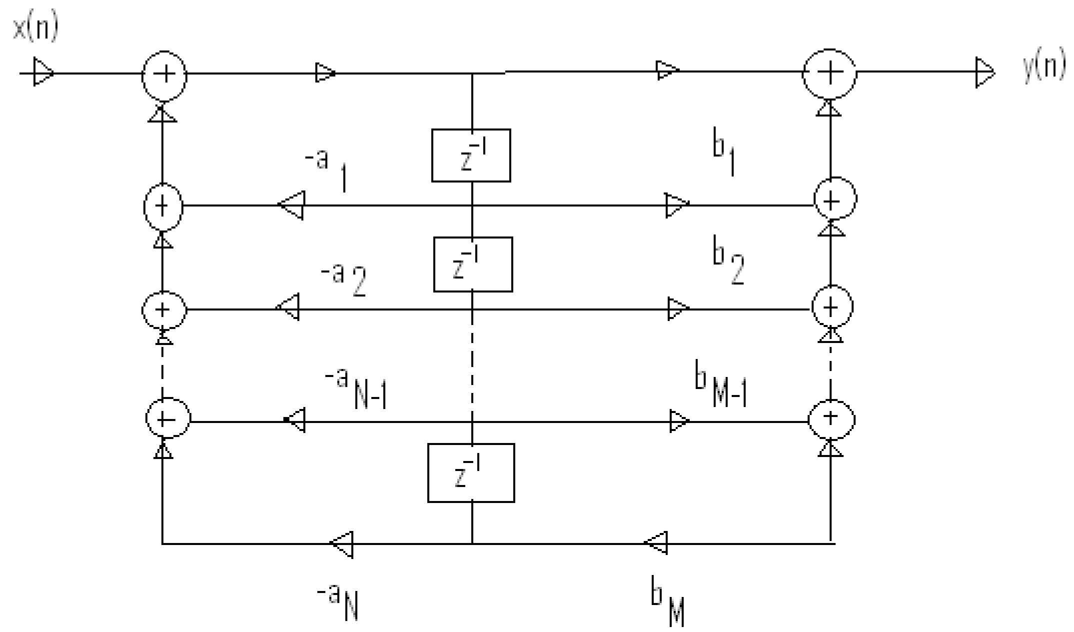
$$\frac{Y(z)}{V(z)} = \left(1+\sum_{k=1}^{M} b_k z^{-k}\right)$$ -------------------all zeros

The corresponding difference equations are,

$$v(n) = x(n) - \sum_{k=1}^{N} a_k v(n-k)$$

$$y(n) = v(n) + \sum_{k=1}^{M} b_k v(n-1)$$




This realization requires M+N+! multiplications, M+N addition and the maximum of {M, N} memory location

## 6.5.3 Cascade Form

The transfer function of a system can be expressed as,

$H(z) = H_1(z)H_2(z)....H_k(z)$

Where $H_k(Z)$ could be first order or second order section realized in Direct form – II form i.e.,

$$H_k(Z) = \frac{b_{k0} + b_{k1}Z^{-1} + b_{k2}Z^{-2}}{1 + a_{k1}Z^{-1} + a_{k2}Z^{-2}}$$

where K is the integer part of (N+1)/2

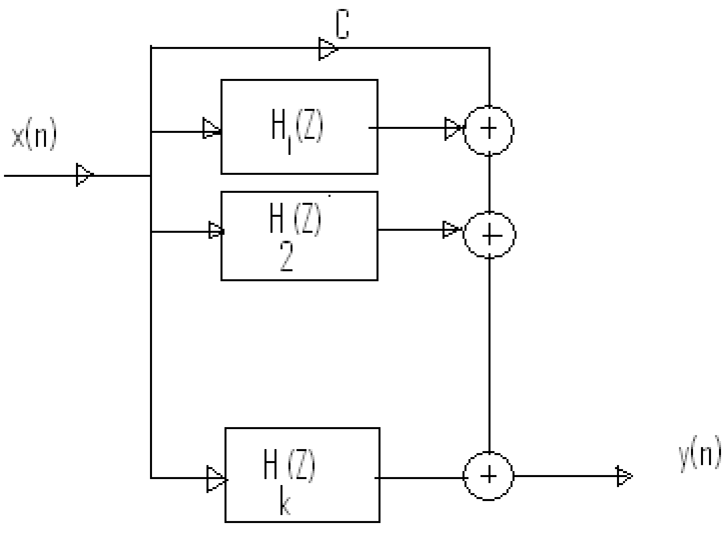
Similar to FIR cascade realization, the parameter $b_0$ can be distributed equally among the k filter section $B_0$ that $b_0 = b_{10}b_{20}.....b_{k0}$. The second order sections are required to realize section which has complex-conjugate poles with real co-efficients. Pairing the two complex-conjugate poles with a pair of complex-conjugate zeros or real-valued zeros to form a subsystem of the type shown above is done arbitrarily. There is no specific rule used in the combination. Although all cascade realizations are equivalent for infinite precision arithmetic, the various realizations may differ significantly when implemented with finite precision arithmetic.

### 6.5.4 Parallel form structure

In the expression of transfer function, if $N \geq M$ we can express system function

$$H(Z) = C + \sum_{k=1}^{N} \frac{A_k}{1 - p_k Z^{-1}} \qquad = C + \sum_{k=1}^{N} H_k(Z)$$

Where $\{p_k\}$ are the poles, $\{A_k\}$ are the coefficients in the partial fraction expansion, and the constant C is defined as $C = b_N / a_N$ , The system realization of above form is shown below.




Where $H_k(Z) = \dfrac{b_{k0} + b_{k1}Z^{-1}}{1 + a_{k1}Z^{-1} + a_{k2}Z^{-2}}$

Once again choice of using first- order or second-order sections depends on poles of the denominator polynomial. If there are complex set of poles which are conjugative in nature then a second order section is a must to have real coefficients.

**Problem 2**

Determine the

(i)Direct form-I (ii) Direct form-II (iii) Cascade &

(iv)Parallel form realization of the system function

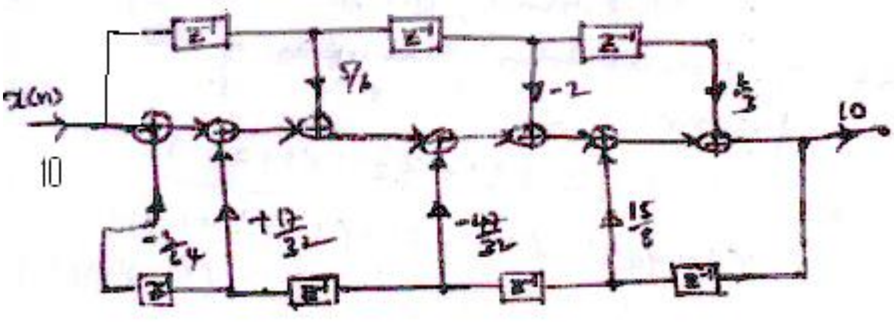
$$H(Z) = \frac{10\left(1-\frac{1}{2}Z^{-1}\right)\left(1-\frac{2}{3}Z^{-1}\right)\left(1+2Z^{-1}\right)}{\left(1-\frac{3}{4}Z^{-1}\right)\left(1-\frac{1}{8}Z^{-1}\right)\left(1-\left(\frac{1}{2}+j\frac{1}{2}\right)Z^{-1}\right)\left(1-\left(\frac{1}{2}-j\frac{1}{2}\right)Z^{-1}\right)}$$

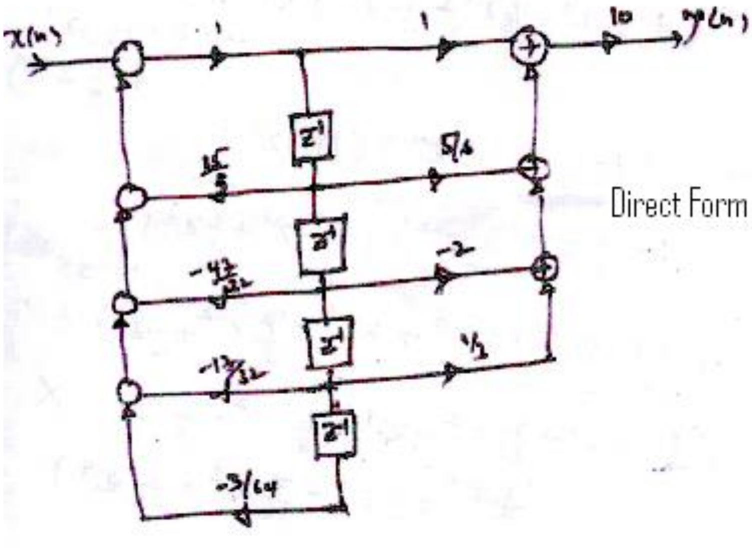
$$= \frac{10\left(1-\frac{7}{6}Z^{-1}+\frac{1}{3}Z^{-2}\right)\left(1+2Z^{-1}\right)}{\left(1+\frac{7}{8}Z^{-1}+\frac{3}{32}Z^{-2}\right)\left(1-Z^{-1}+\frac{1}{2}Z^{-2}\right)}$$

$$H(Z) = \frac{10\left(1+\frac{5}{6}Z^{-1}-2Z^{-2}+\frac{2}{3}Z^{-3}\right)}{\left(1-\frac{15}{8}Z^{-1}+\frac{47}{32}Z^{-2}-\frac{17}{32}Z^{-3}+\frac{3}{64}Z^{-4}\right)}$$

$$H(z) = \frac{(-14.75-12.90z^{-1})}{(1+\frac{7}{8}z^{-1}+\frac{3}{32}z^{-2})} + \frac{(24.50+26.82z^{-1})}{(1-z^{-1}+\frac{1}{2}z^{-2})}$$

Direct Form I






Direct Form II

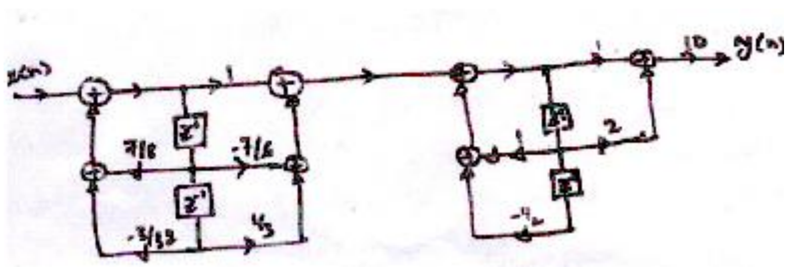
Cascade Form

$H(z) = H_1(z)\, H_2(z)$

Where

$$H_1(z) = \frac{1 - \frac{7}{6} z^{-1} + \frac{1}{3} z^{-2}}{1 - \frac{7}{8} z^{-1} + \frac{3}{32} z^{-2}}$$

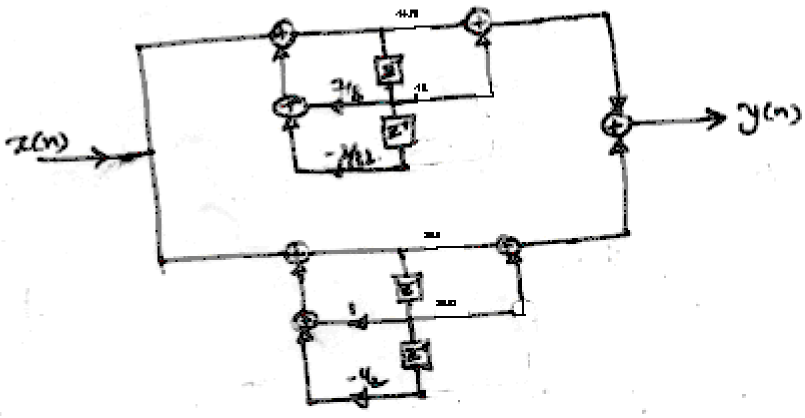
$$H_1(z) = \frac{10(1 + 2z^{-1})}{1 - z^{-1} + \frac{1}{2} z^{-2}}$$



Parallel Form

H(z) = $H_1(z) + H_2(z)$

$$H(z) = \frac{(-14.75 - 12.90z^{-1})}{(1 + \frac{7}{8}z^{-1} + \frac{3}{32}z^{-2})} + \frac{(24.50 + 26.82z^{-1})}{(1 - z^{-1} + \frac{1}{2}z^{-2})}$$




**Problem: 3**

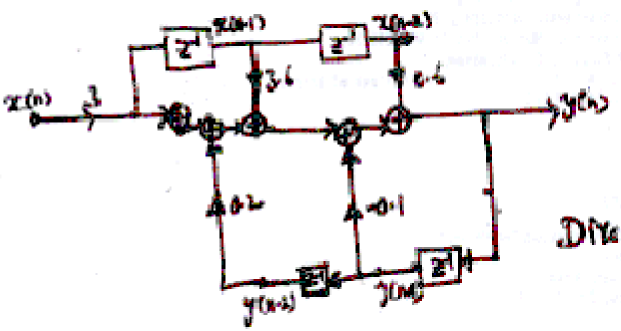
Obtain the direct form – I, direct form-II

Cascade and parallel form realization for the following system,

y(n)= -0.1 y(n-1)+0.2y(n-2)+3x(n)+3.6 x(n-1)+0.6 x(n-2)

Solution:

The Direct form realization is done directly from the given i/p – o/p equation, show in below diagram

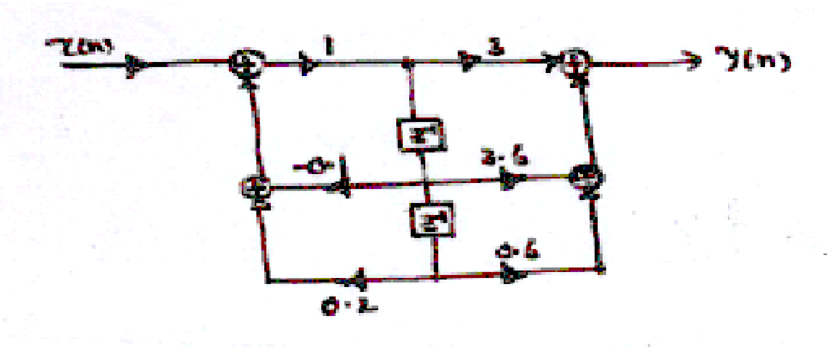



Direct form I

Direct form –II realization

Taking ZT on both sides and finding H(z)

$$H(z) = \frac{Y(z)}{X(z)} = \frac{3 + 3.6z^{-1} + 0.6z^{-2}}{1 + 0.1z^{-1} - 0.2z^{-2}}$$

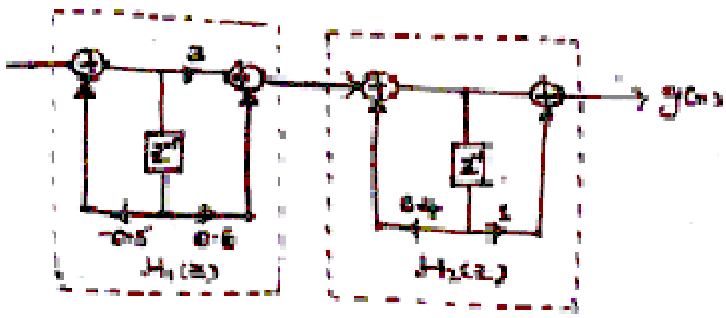



Cascade form realization
The transformer function can be expressed as:

$$H(z) = \frac{(3+0.6z^{-1})(1+z^{-1})}{(1+0.5z^{-1})(1-0.4z^{-1})}$$

which can be re written as

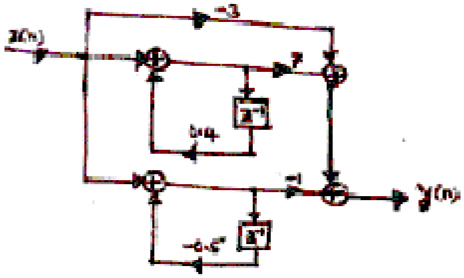
where $H_1(z) = \dfrac{3+0.6z^{-1}}{1+0.5z^{-1}}$ and $H_2(z) = \dfrac{1+z^{-1}}{1-0.4z^{-1}}$




**Parallel Form realization**

The transfer function can be expressed as
$H(z) = C + H_1(z) + H_2(z)$ where $H_1(z)$ & $H_2(z)$ is given by,

$$H(z) = -3 + \frac{7}{1-0.4z^{-1}} - \frac{1}{1+0.5z^{-1}}$$




## 6.6 Lattice Structure for IIR System:

Consider an All-pole system with system function.

$$H(Z) = \frac{1}{1+\sum_{k=1}^{N} a_N(k) Z^{-k}} = \frac{1}{A_N(Z)}$$

The corresponding difference equation for this IIR system is,

$$y(n) = -\sum_{k=1}^{N} a_N(k) y(n-k) + x(n)$$

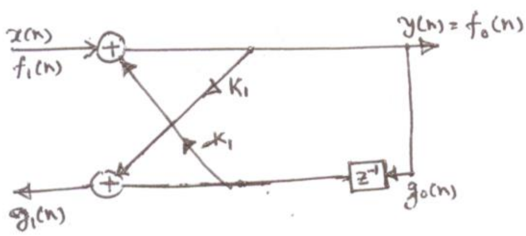
OR

$$x(n) = y(n) + \sum_{k=1}^{N} a_N(k) y(n-k)$$

For N=1

$$x(n) = y(n) + a_1(1) y(n-1)$$

Which can realized as,



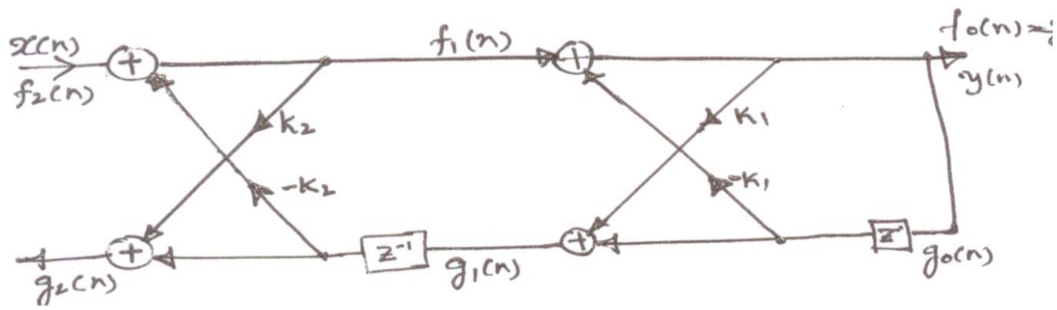

We observe

$$x(n) = f_1(n)$$
$$y(n) = f_0(n) = f_1(n) - k_1 g_0(n-1)$$
$$= x(n) - k_1 y(n-1) \qquad\qquad k_1 = a_1(1)$$
$$g_1(n) = k_1 f_0(n) + g_0(n-1) = k_1 y(n) + y(n-1)$$

For N=2, then

$$y(n) = x(n) - a_2(1) y(n-1) - a_2(2) y(n-2)$$

This output can be obtained from a two-stage lattice filter as shown in below fig




$$f_2(n) = x(n)$$
$$f_1(n) = f_2(n) - k_2 g_1(n-1)$$
$$g_2(n) = k_2 f_1(n) + g_1(n-1)$$
$$f_0(n) = f_1(n) - k_1 g_0(n-1)$$
$$g_1(n) = k_1 f_0(n) + g_0(n-1)$$

$$\begin{aligned}
y(n) &= f_0(n) = g_0(n) = f_1(n) - k_1 g_0(n-1) \\
&= f_2(n) - k_2 g_1(n-1) - k_1 g_0(n-1) \\
&= f_2(n) - k_2\left[k_1 f_0(n-1) + g_0(n-2)\right] - k_1 g_0(n-1) \\
&= x(n) - k_2\left[k_1 y(n-1) + y(n-2)\right] - k_1 y(n-1) \\
&= x(n) - k_1(1+k_2) y(n-1) - k_2 y(n-2)
\end{aligned}$$

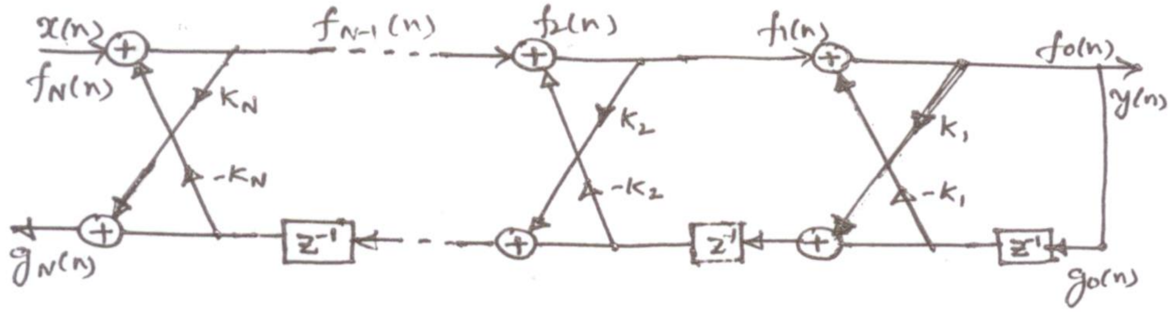
Similarly

$$g_2(n) = k_2 y(n) + k_1(1+k_2) y(n-1) + y(n-2)$$

We observe

$$a_2(0) = 1; a_2(1) = k_1(1+k_2); a_2(2) = k_2$$

N-stage IIR filter realized in lattice structure is,




$$f_N(n) = x(n)$$

$$f_{m-1}(n) = f_m(n) - k_m g_{m-1}(n-1) \qquad \text{m=N, N-1,---1}$$

$$g_m(n) = k_m f_{m-1}(n) + g_{m-1}(n-1) \qquad \text{m=N, N-1,---1}$$

$$y(n) = f_0(n) = g_0(n)$$

8.6.1 Conversion from lattice structure to direct form:

$$a_m(m) = k_m; \qquad a_m(0) = 1$$
$$a_m(k) = a_{m-1}(k) + a_m(m) a_{m-1}(m-k)$$

Conversion from direct form to lattice structure

$$a_{m-1}(0) = 1 \qquad\qquad k_m = a_m(m)$$

$$a_{m-1}(k) = \frac{a_m(k) - a_m(m)\, a_m(m-k)}{1 - a_m^2(m)}$$

**6.6.2 Lattice – Ladder Structure:**

A general IIR filter containing both poles and zeros can be realized using an all pole lattice as the basic building block.

If,

$$H(Z) = \frac{B_M(Z)}{A_N(Z)} = \frac{\sum_{k=0}^{M} b_M(k) Z^{-k}}{1 + \sum_{k=1}^{N} a_N(k) Z^{-k}}$$

Where $N \geq M$

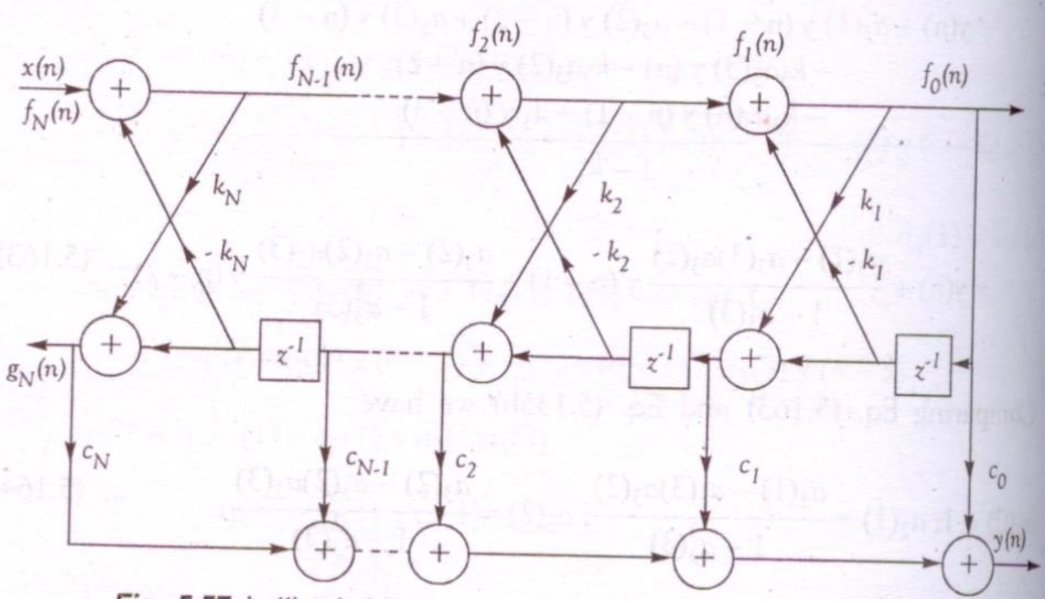
A lattice structure can be constructed by first realizing an all-pole lattice co-efficients $k_m, \ 1 \leq m \leq N$ for the denominator $A_N(Z)$, and then adding a ladder part for M=N. The output of the ladder part can be expressed as a weighted linear combination of $\{g_m(n)\}$.

Now the output is given by

$$y(n) = \sum_{m=0}^{M} C_m g_m(n)$$

Where $\{C_m\}$ are called the ladder co-efficient and can be obtained using the recursive relation,

$$C_m = b_m - \sum_{i=m+1}^{M} C_i a_i(i-m); \qquad \text{m=M, M-1, ….0}$$




**Problem:4**

Convert the following pole-zero IIR filter into a lattice ladder structure,

$$H(Z) = \frac{1+2Z^{-1}+2Z^{-2}+Z^{-3}}{1+\frac{13}{24}Z^{-1}+\frac{5}{8}Z^{-2}+\frac{1}{3}Z^{-3}}$$

Solution:

Given $b_M(Z) = 1+2Z^{-1}+2Z^{-2}+Z^{-3}$

And $A_N(Z) = 1+\frac{13}{24}Z^{-1}+\frac{5}{8}Z^{-2}+\frac{1}{3}Z^{-3}$

$a_3(0)=1; \quad a_3(1)=\frac{13}{24}; \quad a_3(2)=\frac{5}{8}; \quad a_3(3)=\frac{1}{3}$

$$k_3 = a_3(3) = \tfrac{1}{3}$$

Using the equation

$$a_{m-1}(k) = \frac{a_m(k)-a_m(m)a_m(m-k)}{1-a^2m(m)}$$

for m=3, k=1

$$a_2(1) = \frac{a_3(1)-a_3(3)a_3(2)}{1-a_3^2(3)} = \frac{\frac{13}{24}-\frac{1}{3}\cdot\frac{5}{8}}{1-\left(\frac{1}{3}\right)^2} = \frac{3}{8}$$

for m=3, & k=2

$$a_2(2) = k_2 = \frac{a_3(2)-a_3(3)a_3(1)}{1-a_3^2(3)}$$

$$\frac{\frac{5}{8}-\frac{1}{3}\cdot\frac{13}{24}}{1-\frac{1}{9}} = \frac{\frac{45-13}{72}}{\frac{8}{9}} = \frac{1}{2}$$

for m=2, & k=1

$$a_1(1) = k_1 = \frac{a_2(1)-a_2(2)a_2(1)}{1-a_2^2(2)}$$

$$\frac{\frac{3}{8}-\frac{1}{2}\cdot\frac{3}{8}}{1-(\frac{1}{2})^2} = \frac{\frac{3}{8}-\frac{3}{16}}{1-\frac{1}{4}} = \frac{1}{4}$$

for lattice structure $k_1 = \frac{1}{4}, \quad k_2 = \frac{1}{2}, \quad k_3 = \frac{1}{3}$

For ladder structure

$$C_m = b_m - \sum_{i=m+1}^{M} C_1.a_1(1-m) \qquad \text{m=M, M-1,1,0}$$

M=3 $\quad C_3 = b_3 = 1; \quad C_2 = b_2 - C_3 a_3(1)$

$$= 2 - 1.(\tfrac{13}{24}) = 1.4583$$

$$C_1 = b_1 - \sum_{i=2}^{3} c_1 a_1 (i-m) \qquad \text{m=1}$$

$$= b_1 - \left[c_2 a_2(1) + c_3 a_{3(2)}\right]$$

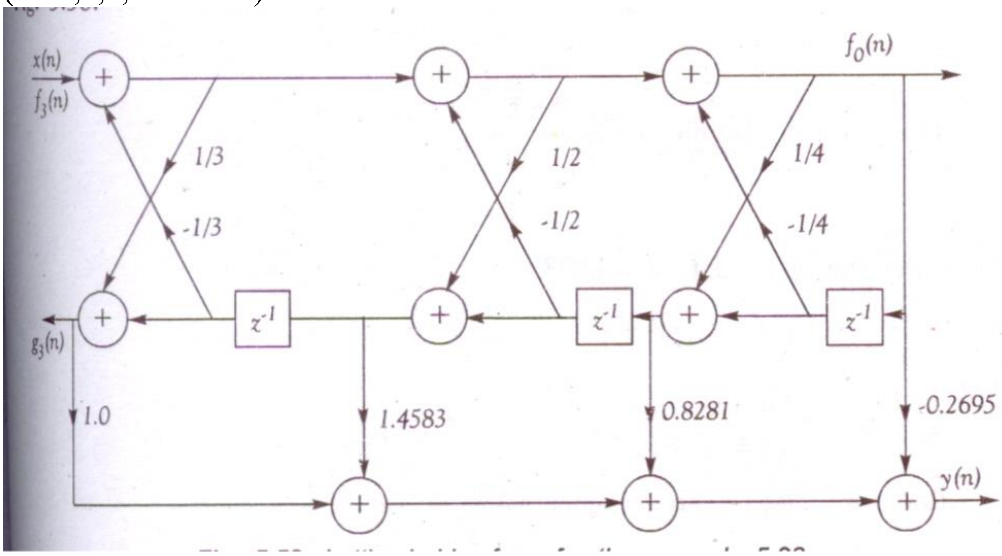
$$= 2 - \left[(1.4583)(\tfrac{3}{8}) + \tfrac{5}{8}\right] = 0.8281$$

$$c_0 = b_0 - \sum_{i=1}^{3} c_1 a_1 (i-m)$$

$$= b_0 - \left[c_1 a_1(1) + c_2 a_2(2) + c_3 a_3(3)\right]$$

$$= 1 - \left[08281(\tfrac{1}{4}) + 1.4583(\tfrac{1}{2}) + \tfrac{1}{3}\right] = -02695$$

To convert a lattice- ladder form into a direct form, we find an equation to obtain $a_N(k)$ from $k_m$ (m=1,2,………N) then equation for $c_m$ is recursively used to compute $b_m$ (m=0,1,2,………M).




**Problem 5**

*A z-plane pole-zero plot for a certain digital filter is shown in figure 7. The filter has unity gain at DC. Determine the system function in the form*

$$H(z) = A\left[\frac{(1+a_1 z^{-1})(1+b_1 z^{-2}+b_2 z^{-2})}{(1+c_1 z^{-1})(1+d_1 z^{-1}+d_2 z^{-2})}\right]$$ *giving the numerical values for parameters*

*A,* $a_1, b_1, b_2, c_1, d_1$ *and* $d_2$*. Sketch the direct form-II and cascade realizations of the system.* **[9]**

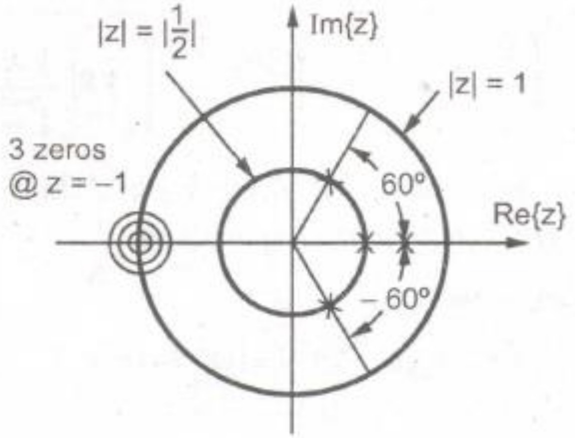



**Fig. 7**

**Sol. :**

$$H(z) = A\cdot\frac{(1+z^{-1})^3}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-1}\right)}$$

$$H(z) = A\cdot\frac{(1+z^{-1})(1+2z^{-1}+z^{-2})}{\left(1-\frac{1}{2}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-1}\right)}$$

$$H(z)\big|_{z=1} = 1$$

$$\Rightarrow \quad A = \frac{3}{64}, a_1 = 1, b_1 = 2, b_2 = 1, c_1 = -\frac{1}{2}, d_1 = -\frac{1}{2} \text{ and } d_2 = \frac{1}{4}$$

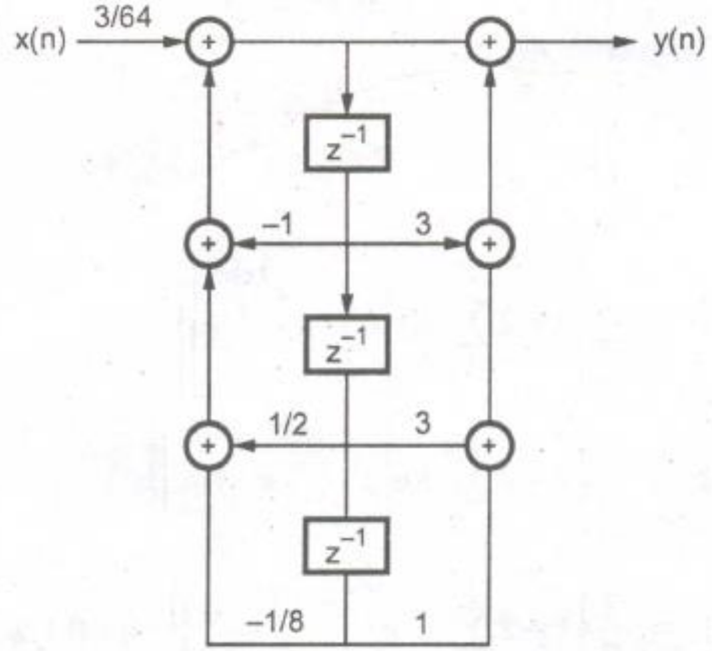



**Fig. 8 (a)**

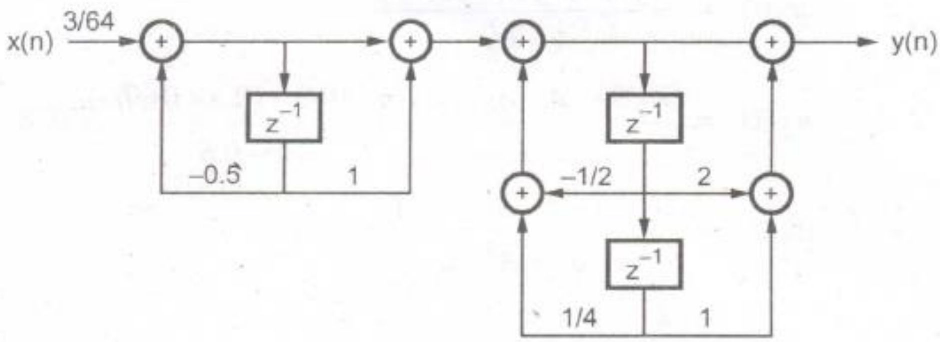



**Fig. 8 (b)**

## <u>Question 6</u>

Consider a FIR filter with system function:

$H(z) = 1+2.82\, Z^{-1} + 3.4048z^{-2} + 1.74z^{-3}$. Sketch the direct form and lattice realizations of the filter.

**Sol. :** $A_3(z) = H(z) = 1 + 2.82z^{-1} + 3.4048z^{-2} + 1.74z^{-3}$

$$B_3(z) = 1.74 + 3.4048z^{-1} + 2.82z^{-2} + z^{-3}$$

Hence $k_3 = 1.74$

$$A_2(z) = \frac{A_3(z) - k_3 B_3(z)}{1 - k_4^2}$$

$$= \frac{1 + 2.82z^{-1} + 3.4048z^{-2} + 1.74z^{-3} - 3.0276 - 5.9243z^{-1} - 4.9068z^{-2} - 1.74z^{-3}}{(-2.0276)}$$

$$= \frac{-2.0276 - 3.1043z^{-1} - 1.502z^{-2}}{(-2.0276)} = 1 + 1.531z^{-1} + 0.7407z^{-2}$$

$$B_2(z) = 0.7407 + 1.531z^{-1} + z^{-2}$$

$$k_2 = 0.7407$$

$$A_1(z) = \frac{A_2(z) - k_2 B_2(z)}{1 - k_2^2}$$

$$= \frac{1 + 1.531z^{-1} + 0.7407z^{-2} - 0.5486 - 1.134z^{-1} - 0.7407z^{-2}}{0.4514}$$

$$= 1 + 0.8795z^{-1}$$

$$k_1 = 0.8795$$

**Direct form realization :**

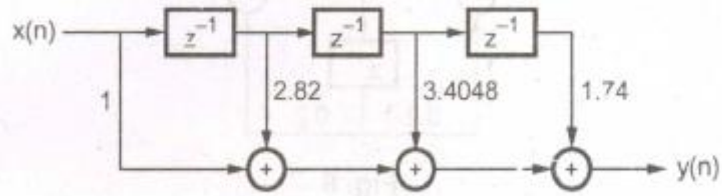



**Fig. 5**

**Lattice realization :**

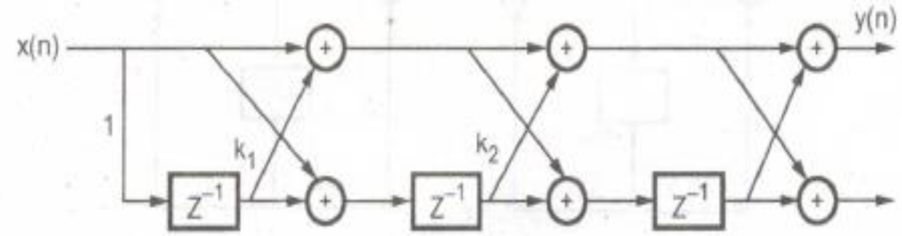



Fig. 6

# UNIT 7

## FIR FILTER DESIGN

## CONTENTS:-

**FIR FILTER DESIGN: INTRODUCTION TO FIR FILTERS, DESIGN OF FIR FILTERS USING - RECTANGULAR, HAMMING, BARTLET AND KAISER WINDOWS, FIR FILTER DESIGN USING FREQUENCY SAMPLING TECHNIQUE.** 6 HRS

## RECOMMENDED READINGS


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# UNIT 7
# Design of FIR Filters

## 7.1 Introduction:

Two important classes of digital filters based on impulse response type are

Finite Impulse Response (FIR)

Infinite Impulse Response (IIR)

The filter can be expressed in two important forms as:

1 ) System function representation;

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}} \qquad (1)$$

2) Difference Equation representation;

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k) \qquad (2)$$

Each of this form allows various methods of implementation. The eq (2) can be viewed as a computational procedure (an algorithm) for determining the output sequence y(n) of the system from the input sequence x(n). Different realizations are possible with different arrangements of eq (2)

The major issues considered while designing a digital filters are :

- Realiability (causal or non causal)
- Stability (filter output will not saturate)
- Sharp Cutoff Characteristics
- Order of the filter need to be minimum (this leads to less delay)
- Generalized procedure (having single procedure for all kinds of filters)
- Linear phase characteristics

The factors considered with filter implementation are ,

a. It must be a simple design

b. There must be modularity in the implementation so that any order filter can be obtained with lower order modules.

c. Designs must be as general as possible. Having different design procedures for different types of filters( high pass, low pass,…) is cumbersome and complex.

d. Cost of implementation must be as low as possible

e. The choice of Software/Hardware realization

## 7.2 Features of IIR:

The important features of this class of filters can be listed as:

- Out put is a function of past o/p, present and past i/p's
- It is recursive in nature
- It has at least one Pole (in general poles and zeros)
- Sharp cutoff chas. is achievable with minimum order
- Difficult to have linear phase chas over full range of freq.
- Typical design procedure is analog design then conversion from analog to digital

**7.3 Features of FIR : The main features of FIR filter are,**

- They are inherently stable
- Filters with linear phase characteristics can be designed
- Simple implementation – both recursive and nonrecursive structures possible
- Free of limit cycle oscillations when implemented on a finite-word length digital system

**7.3.1 Disadvantages:**

- Sharp cutoff at the cost of higher order
- Higher order leading to more delay, more memory and higher cost of implementation

## 7.4 Importance of Linear Phase:

The group delay is defined as

$$\tau_g = -\frac{d\theta(\omega)}{d\omega}$$

which is negative differential of phase function.

Nonlinear phase results in different frequencies experiencing different delay and arriving at different time at the receiver. This creates problems with speech processing and data communication applications. Having linear phase ensures constant group delay for all frequencies.

The further discussions are focused on FIR filter.

6.5 Examples of simple FIR filtering operations: ***1.Unity Gain Filter***

$y(n)=x(n)$

2. Constant gain filter

$y(n)=Kx(n)$

3. Unit delay filter

$y(n)=x(n-1)$

4.Two - term Difference filter

$y(n) = x(n)-x(n-1)$

5. Two-term average filter

$y(n) = 0.5(x(n)+x(n-1))$

6. Three-term average filter (3-point moving average filter)

$y(n) = 1/3[x(n)+x(n-1)+x(n-2)]$

7. Central Difference filter

$y(n)= 1/2[ x(n) – x(n-2)]$

When we say Order of the filter it is the number of previous inputs used to compute the current output and Filter coefficients are the numbers associated with each of the terms x(n), x(n-1),.. etc

The table below shows order and filter coefficients of above simple filter types:

| Ex. | order | a0 | a1 | a2 |
|---|---|---|---|---|
| 1 | 0 | 1 | - | - |
| 2 | 0 | K | - | - |
| 3 | 1 | 0 | 1 | - |
| 4(HP) | 1 | 1 | -1 | - |
| 5(LP) | 1 | 1/2 | 1/2 | - |
| 6(LP) | 2 | 1/3 | 1/3 | 1/3 |
| 7(HP) | 2 | 1/2 | 0 | -1/2 |

## 7.6 Design of FIR filters:

The section to follow will discuss on design of FIR filter. Since linear phase can be achieved with FIR filter we will discuss the conditions required to achieve this.

### 7.6.1 Symmetric and Antisymmetric FIR filters giving out Linear Phase characteristics:

Symmetry in filter impulse response will ensure linear phase

An FIR filter of length M with i/p x(n) & o/p y(n) is described by the difference equation:

$$y(n)= b_0\, x(n) + b_1\, x(n-1)+\ldots\ldots+b_{M-1}\, x(n-(M-1)) = \sum_{k=0}^{M-1} b_k x(n-k) \qquad -(1)$$

Alternatively. it can be expressed in convolution form

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \qquad - (2)$$

i.e $b_k = h(k)$, k=0,1,…..M-1

Filter is also characterized by

$H(z)=\sum_{k=0}^{M-1} h(k) z^{-k}$ -(3) polynomial of degree M-1 in the variable $z^{-1}$. The roots of this polynomial constitute zeros of the filter.

An FIR filter has linear phase if its unit sample response satisfies the condition
h(n)= ± h(M-1-n) n=0,1,…….M-1 -(4)

Incorporating this symmetry & anti symmetry condition in eq 3 we can show linear phase chas of FIR filters

$$H(z)=h(0)+h(1)z^{-1}+h(2)z^{-2}+\dots\dots\dots\dots+h(M-2)z^{-(M-2)}+h(M-1)z^{-(M-1)}$$

If M is odd

$$H(z)=h(0)+h(1)z^{-1}+\dots\dots\dots+h(\frac{M-1}{2})z^{-(\frac{M-1}{2})}+h(\frac{M+1}{2})z^{-(\frac{M+1}{2})}+h(\frac{M+3}{2})z^{-(\frac{M+3}{2})}+\dots\dots\dots\dots$$
$$+h(M-2)z^{-(M-2)}+h(M-1)z^{-(M-1)}$$

$$=z^{-(\frac{M-1}{2})}\left[h(0)z^{(\frac{M-1}{2})}+h(1)z^{(\frac{M-3}{2})}+\dots\dots\dots\dots+h(\frac{M-1}{2})+h(\frac{M+1}{2})z^{-1}+h(\frac{M+3}{2})z^{-2}+\dots\dots h(M-1)z^{-(\frac{M-1}{2})}\right]$$

Applying symmetry conditions for M odd

$$h(0)=\pm h(M-1)$$
$$h(1)=\pm h(M-2)$$
.
.
$$h(\frac{M-1}{2})=\pm h(\frac{M-1}{2})$$
$$h(\frac{M+1}{2})=\pm h(\frac{M-3}{2})$$
.
.
$$h(M-1)=\pm h(0)$$

$$H(z) = z^{-(\frac{M-1}{2})}\left[ h(\frac{M-1}{2}) + \sum_{n=0}^{\frac{M-3}{2}} h(n)\{z^{(M-1-2n)/2} \pm z^{-(M-1-2n)/2}\}\right]$$

*similarly for M even*

$$H(z) = z^{-(\frac{M-1}{2})}\left[ \sum_{n=0}^{\frac{M}{2}-1} h(n)\{z^{(M-1-2n)/2} \pm z^{-(M-1-2n)/2}\}\right]$$

### 7.6.2 Frequency response:

If the system impulse response has symmetry property (i.e.,h(n)=h(M-1-n)) and M is odd $H(e^{j\omega}) = e^{j\theta(\omega)} \mid H_r(e^{j\omega}) \mid$ where

$$H_r(e^{j\omega}) = \left[ h(\frac{M-1}{2}) + 2\sum_{n=0}^{\frac{M-3}{2}} h(n)\cos\omega(\frac{M-1}{2} - n)\right]$$

$$\theta(\omega) = -(\frac{M-1}{2})\omega \quad \text{if } \mid H_r(e^{j\omega}) \mid \geq 0$$

$$= -(\frac{M-1}{2})\omega + \pi \quad \text{if } \mid H_r(e^{j\omega}) \mid \leq 0$$

In case of M even the phase response remains the same with magnitude response expressed as

$$H_r(e^{j\omega}) = \left[ 2\sum_{n=0}^{\frac{M}{2}-1} h(n)\cos\omega(\frac{M-1}{2} - n)\right]$$

If the impulse response satisfies anti symmetry property (i.e., h(n)=-h(M-1-n))then for M odd we will have

$$h(\frac{M-1}{2}) = -h(\frac{M-1}{2}) \text{ } i.e., \text{ } h(\frac{M-1}{2}) = 0$$

$$H_r(e^{j\omega}) = \left[ 2\sum_{n=0}^{\frac{M-3}{2}} h(n)\sin\omega(\frac{M-1}{2} - n)\right]$$

If M is even then,

$$H_r(e^{j\omega}) = \left[ 2\sum_{n=0}^{\frac{M}{2}-1} h(n)\sin\omega(\frac{M-1}{2}-n) \right]$$

In both cases the phase response is given by

$$\theta(\omega) = -(\frac{M-1}{2})\omega + \pi/2 \quad if\ |H_r(e^{j\omega})| \geq 0$$
$$= -(\frac{M-1}{2})\omega + 3\pi/2 \quad if\ |H_r(e^{j\omega})| \leq 0$$

Which clearly shows presence of Linear Phase characteristics.

### 7.6.3 Comments on filter coefficients:

- The number of filter coefficients that specify the frequency response is (M+1)/2 when is M odd and M/2 when M is even in case of symmetric conditions
- In case of impulse response antisymmetric we have h(M-1/2)=0 so that there are (M-1/2) filter coefficients when M is odd and M/2 coefficients when M is even

### 7.6.5 Choice of Symmetric and antisymmetric unit sample response

When we have a choice between different symmetric properties, the particular one is picked up based on application for which the filter is used. The following points give an insight to this issue.

- If h(n)=-h(M-1-n) and M is odd, $H_r(w)$ implies that Hr(0)=0 & $H_r(\pi)=0$, consequently not suited for lowpass and highpass filter. This condition is suited in Band Pass filter design.
- Similarly if M is even $H_r(0)=0$ hence not used for low pass filter
- Symmetry condition h(n)=h(M-1-n) yields a linear-phase FIR filter with non zero response at w = 0 if desired.

Looking at these points, antisymmetric properties are not generally preferred.

### 7.6.6 Zeros of Linear Phase FIR Filters:

Consider the filter system function

$$H(z)=\sum_{n=o}^{M-1}h(n)z^{-n}$$

Expanding this equation

$$H(z)=h(0)+h(1)z^{-1}+h(2)z^{-2}+\ldots\ldots+h(M-2)z^{-(M-2)}+h(M-1)z^{-(M-1)}$$

$$\sin ce\ \ for\ Linear-phase\ we\ need$$

$$h(n)=h(M-1-n)\quad i.e.,$$

$$h(0)=h(M-1);h(1)=h(M-2);\ldots\ldots h(M-1)=h(0);$$

$$then$$

$$H(z)=h(M-1)+h(M-2)z^{-1}+\ldots\ldots..+h(1)z^{-(M-2)}+h(0)z^{-(M-1)}$$

$$H(z)=z^{-(M-1)}[h(M-1)z^{(M-1)}+h(M-2)z^{(M-2)}+\ldots..+h(1)z+h(0)]$$

$$H(z)=z^{-(M-1)}[\sum_{n=0}^{M-1}h(n)(z^{-1})^{-n}]=z^{-(M-1)}H(z^{-1})$$

This shows that if $z = z_1$ is a zero then $z=z_1^{-1}$ is also a zero

The different possibilities:

1. If $z_1 = 1$ then $z_1 = z_1^{-1} =1$ is also a zero implying it is one zero
2. If the zero is real and $|z|<1$ then we have pair of zeros
3. If zero is complex and $|z|=1$then and we again have pair of complex zeros.
4. If zero is complex and $|z|\neq1$ then and we have two pairs of complex zeros

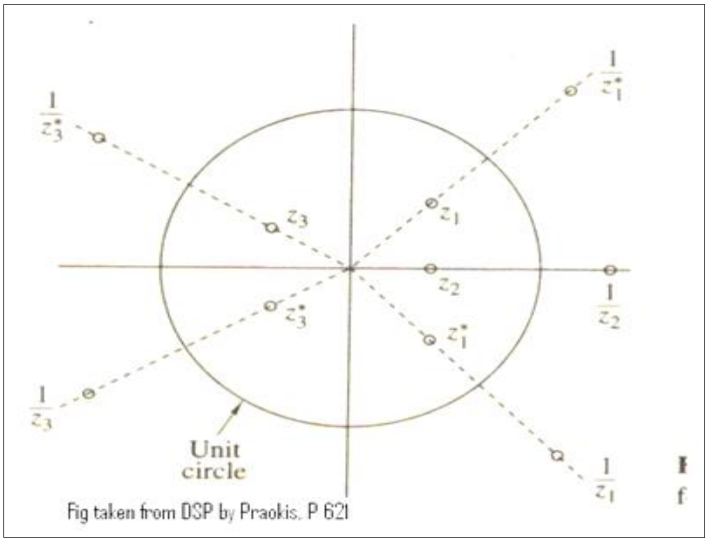


Fig taken from DSP by Praokis, P 621

The plot above shows distribution of zeros for a Linear – phase FIR filter. As it can be seen there is pattern in distribution of these zeros.

### 7.7 Methods of designing FIR filters:

The standard methods of designing FIR filter can be listed as:

1. Fourier series based method
2. Window based method
3. Frequency sampling method

### 7.7.1 Design of Linear Phase FIR filter based on Fourier Series method:

Motivation: Since the desired freq response $H_d(e^{j\omega})$ is a periodic function in $\omega$ with period $2\pi$, it can be expressed as Fourier series expansion

$$H_d(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_d(n) e^{-j\omega n}$$

$$where \quad h_d(n) \quad are \quad fourier\ series\ coefficients$$

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(e^{j\omega}) e^{j\omega n} d\omega$$

This expansion results in impulse response coefficients which are infinite in duration and non causal. It can be made finite duration by truncating the infinite length. The linear phase can be obtained by introducing symmetric property in the filter impulse response, i.e., h(n) = h(-n). It can be made causal by introducing sufficient delay (depends on filter length)

### 7.7.2 Stepwise procedure:

1. From the desired freq response using inverse FT relation obtain $h_d(n)$
2. Truncate the infinite length of the impulse response to finite length with ( assuming M odd)

$$h(n) = h_d(n) \quad for -(M-1)/2 \le n \le (M-1)/2$$
$$= 0 \quad otherwise$$

3. Introduce h(n) = h(-n) for linear phase characteristics
4. Write the expression for H(z); this is non-causal realization
5. To obtain causal realization $H'(z) = z^{-(M-1)/2} H(z)$

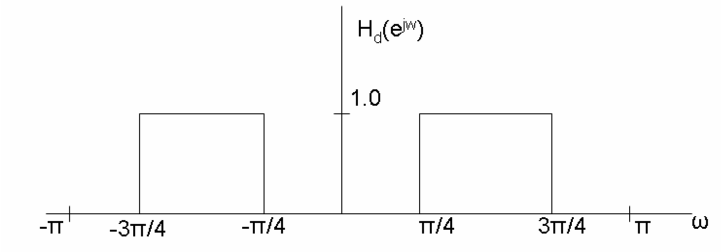
# Exercise Problems

**Problem 1 : Design an ideal bandpass filter with a frequency response:**

$$H_d(e^{j\omega}) = 1 \qquad for\, \frac{\pi}{4} \le |\omega| \le \frac{3\pi}{4}$$

$$= 0 \qquad otherwise$$

Find the values of h(n) for M = 11 and plot the frequency response.




$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}d\omega$$

$$= \frac{1}{2\pi}\left[\int_{-3\pi/4}^{-\pi/4} e^{j\omega n}d\omega + \int_{\pi/4}^{3\pi/4} e^{j\omega n}d\omega\right]$$

$$= \frac{1}{\pi n}\left[\sin\frac{3\pi}{4}n - \sin\frac{\pi}{4}n\right] \quad -\infty \le n \le \infty$$

$$truncating\ to\ 11\ samples\ we\ have\ h(n) = h_d(n)\ \ for\,|n| \le 5$$

$$= 0 \quad otherwise$$

For n = 0 the value of h(n) is separately evaluated from the basic integration

h(0) = 0.5

Other values of h(n) are evaluated from h(n) expression

h(1)=h(-1)=0

h(2)=h(-2)=-0.3183

h(3)=h(-3)=0

h(4)=h(-4)=0

h(5)=h(-5)=0

The transfer function of the filter is

$$H(z) = h(0) + \sum_{n=1}^{(N-1)/2} [h(n)\{z^n + z^{-n}\}]$$

$$= 0.5 - 0.3183(z^2 + z^{-2})$$

*the transfer function of the realizable filter is*

$$H^{'}(z) = z^{-5}[0.5 - 0.3183(z^2 + z^{-2})]$$

$$= -0.3183z^{-3} + 0.5z^{-5} - 0.3183z^{-7}$$

*the filter coeff are*

$$h^{'}(0) = h'(10) = h'(1) = h'(9) = h'(2) = h'(8) = h'(4) = h'(6) = 0$$

$$h'(3) = h'(7) = -0.3183$$

$$h'(5) = 0.5$$

The magnitude response can be expressed as

$$|H(e^{j\omega})| = \sum_{n=1}^{(N-1)/2} a(n)\cos\omega n$$

*comparing this* exp *with*

$$|H(e^{j\omega})| = |z^{-5}[h(0) + 2\sum_{n=1}^{5} h(n)\cos\omega n]|$$

We have

a(0)=h(0)

a(1)=2h(1)=0

a(2)=2h(2)=-0.6366

a(3)=2h(3)=0

a(4)=2h(4)=0

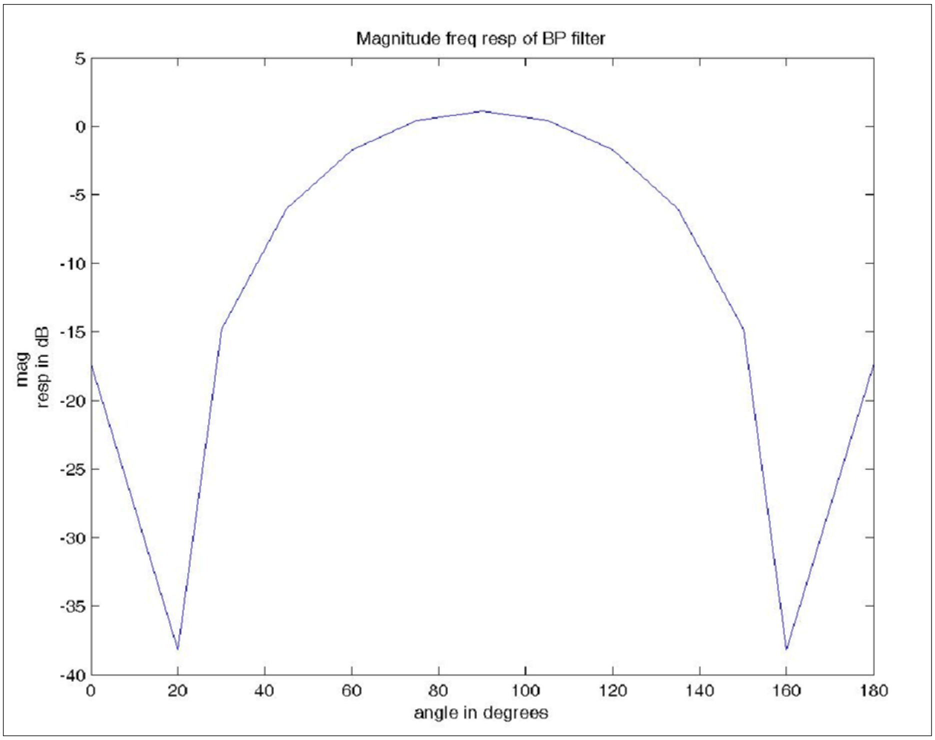
a(5)=2h(5)=0

The magnitude response function is

$|H(e^{j\omega})| = 0.5 – 0.6366 \cos 2\omega$ which can plotted for various values of ω

ω in degrees =[0 20 30 45 60 75 90 105 120 135 150 160 180];

$|H(e^{j\omega})|$ in dBs= [-17.3 -38.17 -14.8 -6.02 -1.74 0.4346 1.11 0.4346 -1.74 -6.02 -14.8 -38.17 -17.3];




**Problem 2: Design an ideal lowpass filter with a freq response**

$$H_d(e^{j\omega}) = 1 \qquad for -\frac{\pi}{2} \le \omega \le \frac{\pi}{2}$$
$$= 0 \qquad for \ \frac{\pi}{2} \le |\omega| \le \pi$$

Find the values of h(n) for N =11. Find H(z). Plot the magnitude response

From the freq response we can determine $h_d(n)$,

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2} e^{j\omega n} d\omega = \frac{\sin\frac{\pi n}{2}}{\pi n} \qquad -\infty \le n \le \infty \quad and \quad n \ne 0$$

Truncating $h_d(n)$ to 11 samples

h(0) = 1/2
h(1)=h(-1)=0.3183
h(2)=h(-2)=0
h(3)=h(-3)=-0.106

h(4)=h(-4)=0
h(5)=h(-5)=0.06366

The realizable filter can be obtained by shifting h(n) by 5 samples to right h'(n)=h(n-5)

h'(n)= [0.06366, 0, -0.106, 0, 0.3183, 0.5, 0.3183, 0, -0.106, 0, 0.06366];

$$H'(z) = 0.06366 - 0.106z^{-2} + 0.3183z^{-4} + 0.5z^{-5} + 0.3183z^{-6} - 0.106z^{-8} + 0.06366z^{-10}$$

Using the result of magnitude response for M odd and symmetry

$$H_r(e^{j\omega}) = [h(\frac{M-1}{2}) + \sum_{n=0}^{\frac{M-3}{2}} h(n)\cos\omega(\frac{M-1}{2} - n)]$$

$$|H_r(e^{j\omega})| = |[0.5 + 0.6366\cos w - 0.212\cos 3w + 0.127\cos 5w]|$$

**Problem 3 :**

Design an ideal band reject filter with a frequency response:

$$\begin{aligned} H_d(e^{j\omega}) &= 1 \qquad for|\omega| \le \frac{\pi}{3} \; and \; |\omega| \ge \frac{2\pi}{3} \\ &= 0 \qquad otherwise \end{aligned}$$

Find the values of h(n) for M = 11 and plot the frequency response

Ans:h(n)= [0 -0.1378 0 0.2757 0 0.667 0 0.2757 0 -0.1378 0];

**7.8 Window based Linear Phase FIR filter design**

The other important method of designing FIR filter is by making use of windows. The arbitrary truncation of impulse response obtained through inverse Fourier relation can lead to distortions in the final frequency response.The arbitrary truncation is equivalent to multiplying infinite length function with finite length rectangular window, i.e.,

$h(n) = h_d(n)\, w(n)$ where $w(n) = 1$ for $n = \pm(M-1)/2$

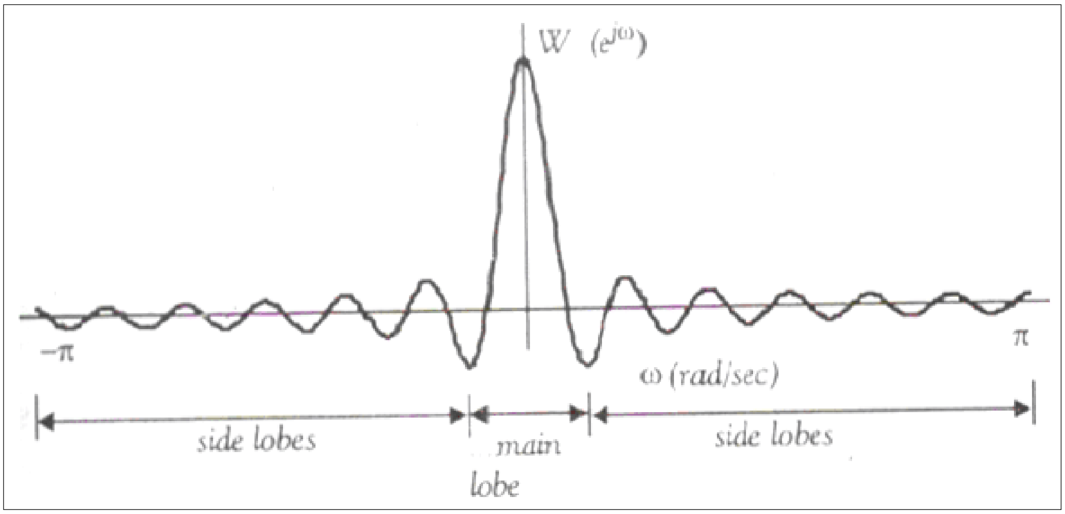
The above multiplication in time domain corresponds to convolution in freq domain, i.e.,

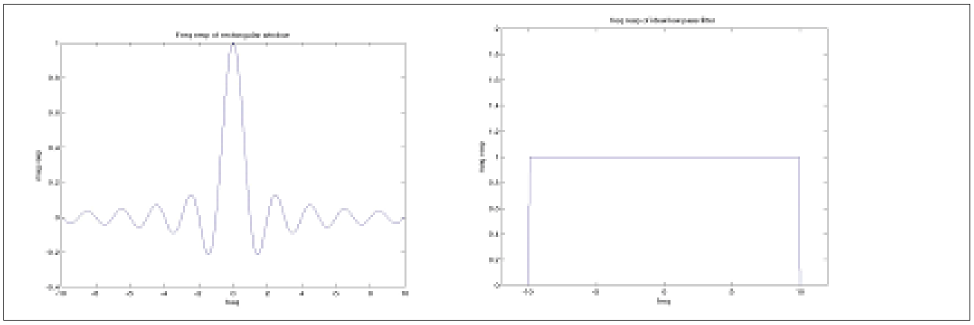
$H(e^{j\omega}) = H_d(e^{j\omega}) * W(e^{j\omega})$ where $W(e^{j\omega})$ is the FT of window function w(n).

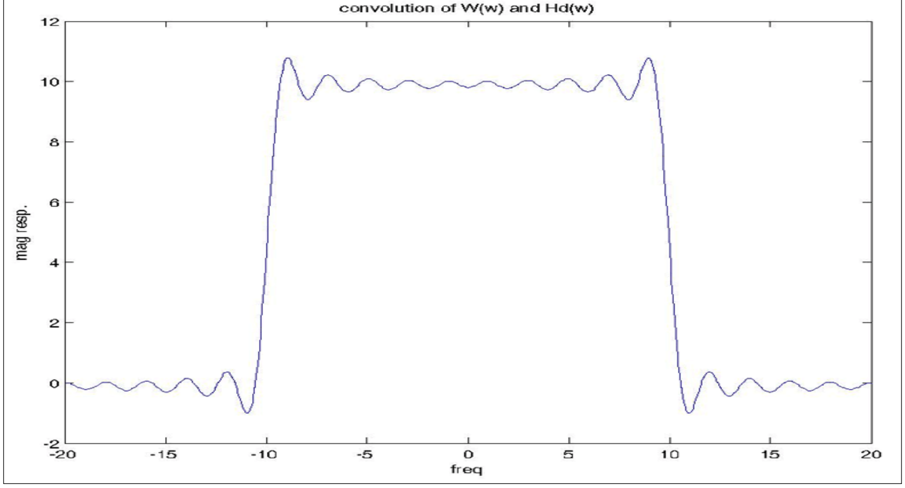
The FT of w(n) is given by

$$W(e^{j\omega}) = \frac{\sin(\omega M/2)}{\sin(\omega/2)}$$

The whole process of multiplying h(n) by a window function and its effect in freq domain are shown in below set of figures.






Suppose the filter to be designed is Low pass filter then the convolution of ideal filter freq response and window function freq response results in distortion in the resultant filter freq response. The ideal sharp cutoff chars are lost and presence of ringing effect is seen at the band edges which is referred to Gibbs Phenomena. This is due to main lobe width and side lobes of the window function freq response.The main lobe width introduces transition band and side lobes results in rippling characters in pass band and stop band. Smaller the main lobe width smaller will be the transition band. The ripples will be of low amplitude if the peak of the first side lobe is far below the main lobe peak.

### 7.8.1 How to reduce the distortions?

1. Increase length of the window

- as M increases the main lob width becomes narrower, hence the transition band width is decreased

-With increase in length the side lobe width is decreased but height of each side lobe increases in such a manner that the area under each sidelobe remains invariant to changes in M. Thus ripples and ringing effect in pass-band and stop-band are not changed.

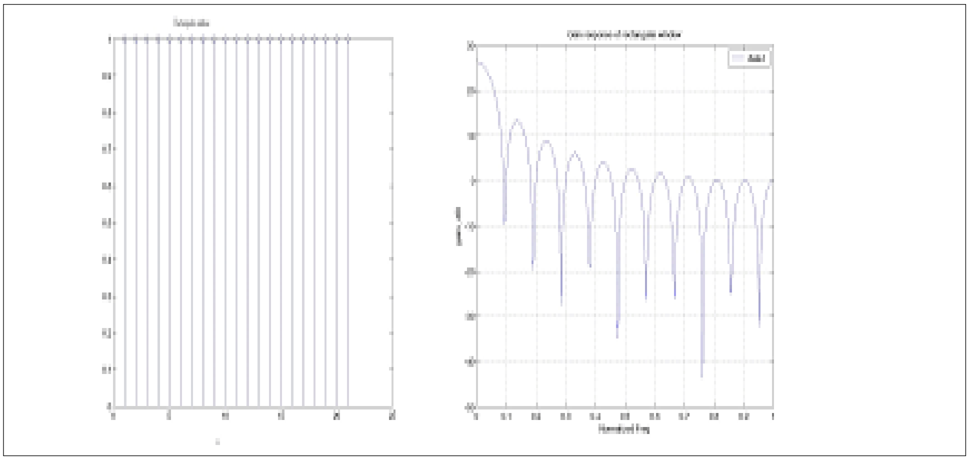
2. Choose windows which tapers off slowly rather than ending abruptly - Slow tapering reduces ringing and ripples but generally increases transition width since main lobe width of these kind of windows are larger.

### 7.8.2 What is ideal window characteristics?

Window having very small main lobe width with most of the energy contained with it (i.e.,ideal window freq response must be impulsive).Window design is a mathematical problem, more complex the window lesser are the distortions. Rectangular window is one of the simplest window in terms of computational complexity. Windows better than rectangular window are, Hamming, Hanning, Blackman, Bartlett, Traingular,Kaiser. The different window functions are discussed in the following sention.

### 7.8.3 Rectangular window: The mathematical description is given by,

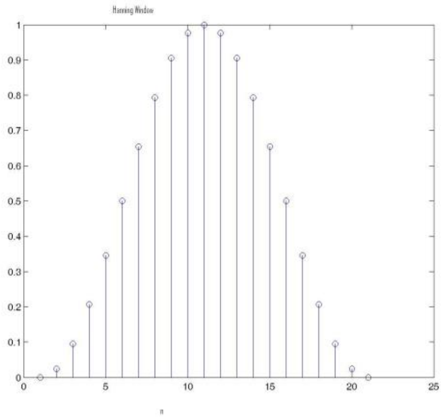
$$w_r(n) = 1 \ for \ 0 \le n \le M-1$$

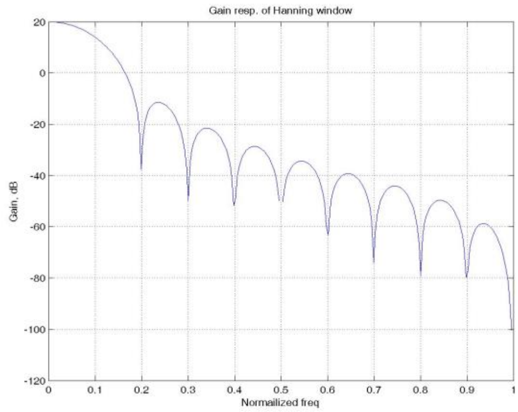


### 7.8.4 Hanning windows:

It is defined mathematically by,

$$w_{han}(n) = 0.5(1 - \cos\frac{2\pi n}{M-1}) \ for \ 0 \le n \le M-1$$

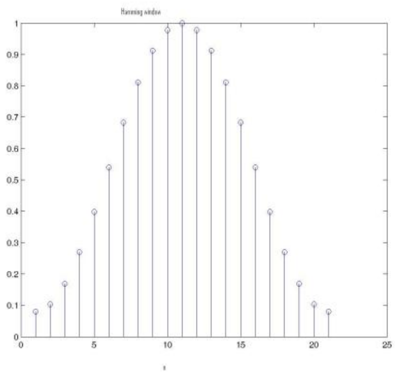


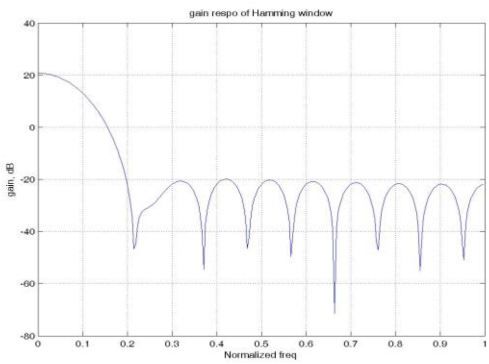



### 7.8.5 Hamming windows:

This window function is given by,

$$w_{ham}(n) = 0.54 - 0.46\cos\frac{2\pi n}{M-1} \quad for\ 0 \le n \le M-1$$

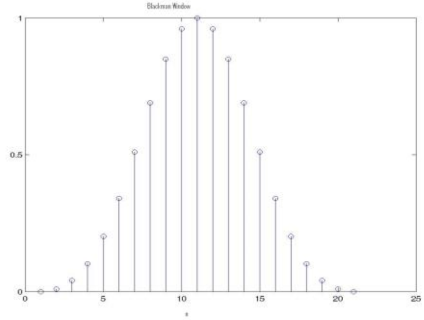


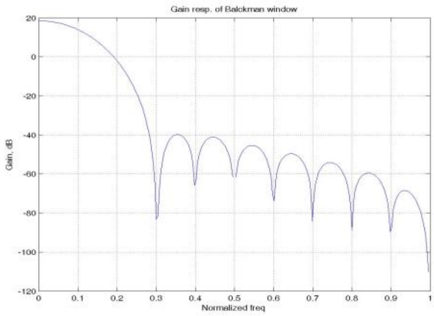



### 7.8.6 Blackman windows:

This window function is given by,

$$w_{blk}(n) = 0.42 - 0.5\cos\frac{2\pi n}{M-1} + 0.08\cos\frac{4\pi n}{M-1} \quad for\ 0 \le n \le M-1$$

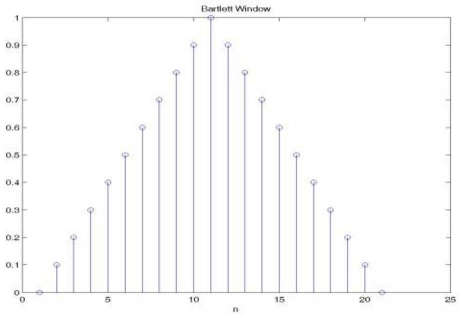


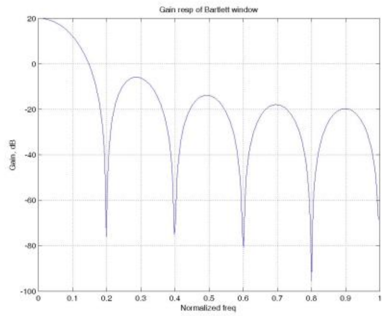



### 7.8.7 Bartlett (Triangular) windows:

The mathematical description is given by,

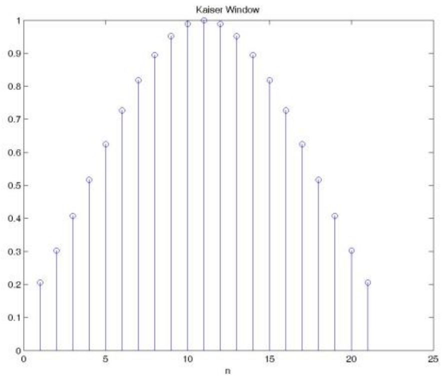
$$w_{bart}(n) = 1 - \frac{2\,|\,n - \frac{M-1}{2}\,|}{M-1} \qquad for\ 0 \le n \le M-1$$

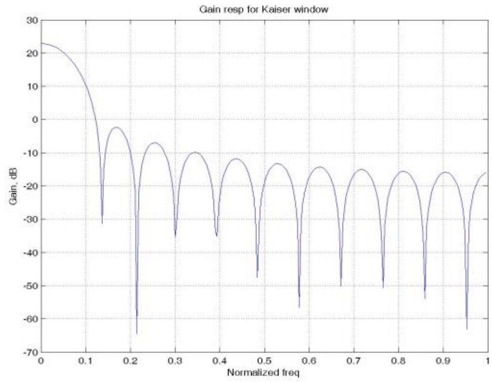





### 7.8.8 Kaiser windows: The mathematical description is given by,

$$w_k(n) = \frac{I_0\left[\alpha\sqrt{\left(\frac{M-1}{2}\right)^2 - \left(n - \frac{M-1}{2}\right)^2}\right]}{I_0\left[\alpha\left(\frac{M-1}{2}\right)\right]} \qquad for\ 0 \le n \le M-1$$




| Type of window | Appr. Transition width of the main lobe | Peak sidelobe (dB) |
|---|---|---|
| Rectangular | $4\pi/M$ | -13 |
| Bartlett | $8\pi/M$ | -27 |
| Hanning | $8\pi/M$ | -32 |
| Hamming | $8\pi/M$ | -43 |
| Blackman | $12\pi/M$ | -58 |

Looking at the above table we observe filters which are mathematically simple do not offer best characteristics. Among the window functions discussed Kaiser is the most complex one in terms of functional description whereas it is the one which offers maximum flexibility in the design.

**7.8.9 Procedure for designing linear-phase FIR filters using windows:**

1. Obtain $h_d(n)$ from the desired freq response using inverse FT relation
2. Truncate the infinite length of the impulse response to finite length with

( assuming M odd) choosing proper window

$$h(n) = h_d(n)w(n) \quad where$$
$$w(n) \; is \; the \; window \; function \; defined \; for \; -(M-1)/2 \le n \le (M-1)/2$$

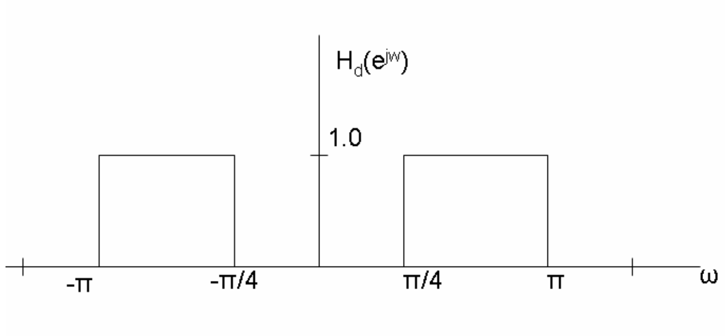
3. Introduce h(n) = h(-n) for linear phase characteristics
4. Write the expression for H(z); this is non-causal realization
5. To obtain causal realization H'(z) = z $^{-(M-1)/2}$ H(z)

# Exercise Problems

**Prob 1: Design an ideal highpass filter with a frequency response:**

$$H_d(e^{j\omega}) = 1 \qquad for \frac{\pi}{4} \le |\omega| \le \pi$$
$$= 0 \qquad |\omega| < \frac{\pi}{4}$$

using a hanning window with M = 11 and plot the frequency response.




$$h_d(n) = \frac{1}{2\pi}[\int_{-\pi}^{-\pi/4} e^{j\omega n} d\omega + \int_{\pi/4}^{\pi} e^{j\omega n} d\omega]$$

$$h_d(n) = \frac{1}{\pi n}[\sin \pi n - \sin \frac{\pi n}{4}] \quad for \quad -\infty \le n \le \infty \quad and \quad n \ne 0$$

$$h_d(0) = \frac{1}{2\pi}[\int_{-\pi}^{-\pi/4} d\omega + \int_{\pi/4}^{\pi} d\omega] = \frac{3}{4} = 0.75$$

$h_d(1) = h_d(-1)=-0.225$
$h_d(2) = h_d(-2)= -0.159$
$h_d(3) = h_d(-3)= -0.075$
$h_d(4) = h_d(-4)= 0$
$h_d(5) = h_d(-5) = 0.045$

The hamming window function is given by

$$w_{hn}(n) = 0.5 + 0.5\cos\frac{2\pi n}{M-1} \qquad -(\frac{M-1}{2}) \le n \le (\frac{M-1}{2})$$
$$= 0 \qquad otherwise$$

$$for \quad N = 11$$
$$w_{hn}(n) = 0.5 + 0.5\cos\frac{\pi n}{5} \qquad -5 \le n \le 5$$

$w_{hn}(0) = 1$
$w_{hn}(1) = w_{hn}(-1)=0.9045$
$w_{hn}(2)= w_{hn}(-2)=0.655$
$w_{hn}(3)= w_{hn}(-3)= 0.345$
$w_{hn}(4)= w_{hn}(-4)=0.0945$
$w_{hn}(5)= w_{hn}(-5)=0$

$h(n)= w_{hn}(n)h_d(n)$

h(n)=[0  0 -0.026  -0.104  -0.204  0.75  -0.204  -0.104  -0.026  0  0]

$$h'(n) = h(n-5)$$
$$H'(z) = -0.026z^{-2} - 0.104z^{-3} - 0.204z^{-4} + 0.75z^{-5} - 0.204z^{-6} - 0.104z^{-7} - 0.026z^{-8}$$

Using the equation

$$H_r(e^{jw}) = [h(\frac{M-1}{2}) + 2\sum_{n=0}^{\frac{M-3}{2}} h(n)\cos\omega(\frac{M-1}{2} - n)$$

$$H_r(e^{jw}) = 0.75) + 2\sum_{n=0}^{4} h(n)\cos\omega(5-n)$$

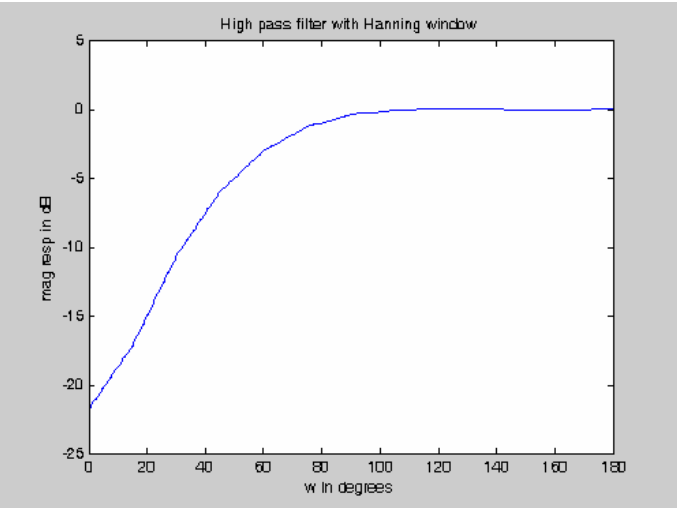
The magnitude response is given by,

$|Hr(e^{j\omega})| = |0.75 - 0.408\cos\omega - 0.208\cos 2\omega - 0.052\cos 3\omega|$

$\omega$ in degrees = [0 15 30 45 60 75 90 105 120 135 150 165 180]

$|H(e^{j\omega})|$ in dBs = [-21.72 -17.14 -10.67 -6.05 -3.07 -1.297 -0.3726

-0.0087 0.052 0.015 0 0 0.017]



## Prob 2 : Design a filter with a frequency response:

$$H_d(e^{j\omega}) = e^{-j3\omega} \qquad for -\frac{\pi}{4} \le \omega \le \frac{\pi}{4}$$

$$= 0 \quad \frac{\pi}{4} < |\omega| \le \pi$$

using a Hanning window with M = 7

**Soln:**

The freq resp is having a term e $^{-j\omega(M-1)/2}$ which gives h(n) symmetrical about n = M-1/2 = 3 i.e we get a causal sequence.

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi/4}^{\pi/4} e^{-j3\omega} e^{j\omega n} d\omega$$

$$= \frac{\sin \frac{\pi}{4}(n-3)}{\pi(n-3)}$$

$$\textit{this gives } h_d(0) = h_d(6) = 0.075$$

$$h_d(1) = h_d(5) = 0.159$$

$$h_d(2) = h_d(4) = 0.22$$

$$h_d(3) = 0.25$$

The Hanning window function values are given by

$w_{hn}(0) = w_{hn}(6) = 0$

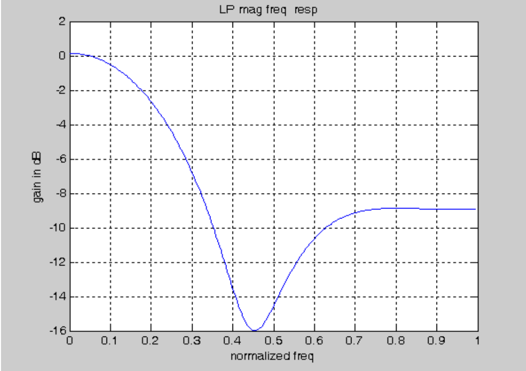
$w_{hn}(1) = w_{hn}(5) = 0.25$

$w_{hn}(2) = w_{hn}(4) = 0.75$

$w_{hn}(3) = 1$

$h(n) = h_d(n)\, w_{hn}(n)$

h(n)=[0  0.03975  0.165  0.25  0.165  0.3975  0]




## 7.9 Design of Linear Phase FIR filters using Frequency Sampling method

7.9.1 Motivation: We know that DFT of a finite duration DT sequence is obtained by sampling FT of the sequence then DFT samples can be used in reconstructing original time domain samples if frequency domain sampling was done correctly. The samples of FT of h(n) i.e., H(k) are sufficient to recover h(n).

Since the designed filter has to be realizable then h(n) has to be real, hence even symmetry properties for mag response |H(k)| and odd symmetry properties for phase response can be applied. Also, symmetry for h(n) is applied to obtain linear phase chas.

Fro DFT relationship we have

$$h(n) = \frac{1}{N}\sum_{k=0}^{N-1} H(k) e^{j2\pi kn/N} \quad for \quad n = 0,1,......N-1$$

$$H(k) = \sum_{n=0}^{N-1} h(n) e^{-j2\pi kn/N} \quad for \quad k = 0,1,.........N-1$$

Also we know $H(k) = H(z)|_{z=e^{j2\pi kn/N}}$

The system function H(z) is given by

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n}$$

Substituting for h(n) from IDFT relationship

$$H(z) = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{H(k)}{1 - e^{j2\pi kn/N} z^{-1}}$$

Since H(k) is obtained by sampling $H(e^{j\omega})$ hence the method is called Frequency Sampling Technique.

Since the impulse response samples or coefficients of the filter has to be real for filter to be realizable with simple arithmetic operations, properties of DFT of real sequence can be used. The following properties of DFT for real sequences are useful:

$H^*(k) = H(N-k)$

$|H(k)|=|H(N-k)|$ - magnitude response is even

$\theta(k) = -\theta(N-k)$ – Phase response is odd

$h(n) = \frac{1}{N}\sum_{k=0}^{N-1} H(k)e^{j2\pi kn/N}$ can be rewritten as (for N odd)

$$h(n) = \frac{1}{N}\left[H(0) + \sum_{k=1}^{N-1} H(k)e^{j2\pi kn/N}\right]$$

$$h(n) = \frac{1}{N}\left[H(0) + \sum_{k=1}^{N-1/2} H(k)e^{j2\pi kn/N} + \sum_{k=N-1/2}^{N-1} H(k)e^{j2\pi kn/N}\right]$$

Using substitution k = N – r or r = N- k in the second substitution
with r going from now (N- 1)/2 to 1 as k goes from 1 to (N-1)/2

$$h(n) = \frac{1}{N}\left[H(0) + \sum_{k=1}^{(N-1)/2} H(k)e^{j2\pi kn/N} + \sum_{k=1}^{(N-1)/2} H(N-k)e^{-j2\pi kn/N}\right]$$

$$h(n) = \frac{1}{N}\left[H(0) + \sum_{k=1}^{(N-1)/2} H(k)e^{j2\pi kn/N} + \sum_{k=1}^{(N-1)/2} H^*(k)e^{-j2\pi kn/N}\right]$$

$$h(n) = \frac{1}{N}\left[H(0) + \sum_{k=1}^{(N-1)/2} H(k)e^{j2\pi kn/N} + \sum_{k=1}^{(N-1)/2} (H(k)e^{j2\pi kn/N})^*\right]$$

$$h(n) = \frac{1}{N}\left[H(0) + \sum_{k=1}^{(N-1)/2} (H(k)e^{j2\pi kn/N} + (H(k)e^{j2\pi kn/N})^*\right]$$

$$h(n) = \frac{1}{N}\left[H(0) + 2\sum_{k=1}^{(N-1)/2} \mathrm{Re}(H(k)e^{j2\pi kn/N}\right]$$

Similarly for N even we have

$$h(n) = \frac{1}{N}\left[H(0) + 2\sum_{k=1}^{(N-1)/2} \mathrm{Re}(H(k)e^{j2\pi kn/N}\right]$$

Using the symmetry property h(n)= h (N-1-n) we can obtain Linear phase FIR filters using the frequency sampling technique.

## Exercise problems

**Prob 1 : Design a LP FIR filter using Freq sampling technique having cutoff freq of π/2 rad/sample. The filter should have linear phase and length of 17.**

The desired response can be expressed as

$$
\begin{aligned}
H_d(e^{j\omega}) &= e^{-j\omega(\frac{M-1}{2})} \quad for \quad |\omega| \le \omega c \\
&= 0 \quad otherwise \\
with \quad M &= 17 \quad and \quad \omega c = \pi/2
\end{aligned}
$$

$$
\begin{aligned}
H_d(e^{j\omega}) &= e^{-j\omega 8} \quad for \quad 0 \le \omega \le \pi/2 \\
&= 0 \quad for \quad \pi/2 \le \omega \le \pi
\end{aligned}
$$

Selecting $\omega_k = \dfrac{2\pi k}{M} = \dfrac{2\pi k}{17} \quad for \quad k = 0,1,......16$

$$
H(k) = H_d(e^{j\omega})|_{\omega=\frac{2\pi k}{17}}
$$

$$
\begin{aligned}
H(k) &= e^{-j\frac{2\pi k}{17}8} \quad for \quad 0 \le \frac{2\pi k}{17} \le \frac{\pi}{2} \\
&= 0 \quad for \quad \pi/2 \le \frac{2\pi k}{17} \le \pi \\
H(k) &= e^{-j\frac{16\pi k}{17}} \quad for \quad 0 \le k \le \frac{17}{4} \\
&= 0 \quad for \quad \frac{17}{4} \le k \le \frac{17}{2}
\end{aligned}
$$

The range for “k” can be adjusted to be an integer such as

$$
\begin{aligned}
&0 \le k \le 4 \\
&and \quad 5 \le k \le 8
\end{aligned}
$$

The freq response is given by

$$H(k) = e^{-j\frac{2\pi k}{17}8} \qquad for \quad 0 \le k \le 4$$
$$= 0 \qquad for \qquad 5 \le k \le 8$$

Using these value of H(k) we obtain h(n) from the equation

$$h(n) = \frac{1}{M}(H(0) + 2\sum_{k=1}^{(M-1)/2} \mathrm{Re}(H(k)e^{j2\pi kn/M}))$$

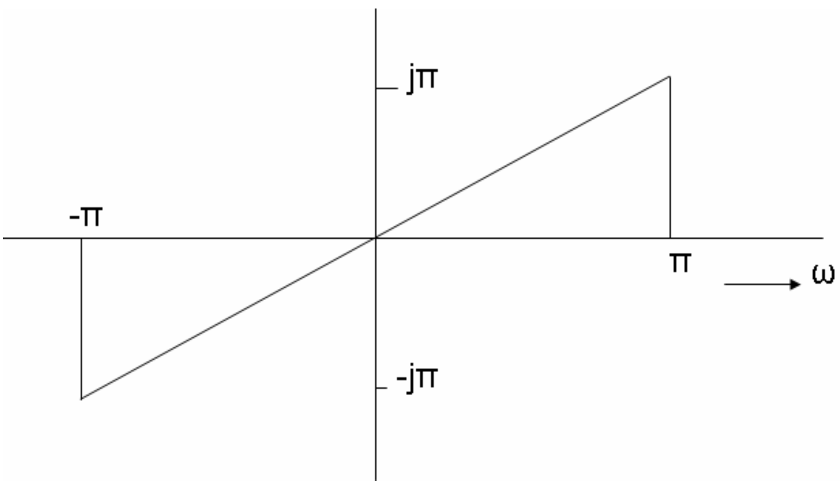
i.e., $h(n) = \frac{1}{17}(1 + 2\sum_{k=1}^{4} \mathrm{Re}(e^{-j16\pi k/17}e^{j2\pi kn/17}))$

$$h(n) = \frac{1}{17}(H(0) + 2\sum_{k=1}^{4} \cos(\frac{2\pi k(8-n)}{17}) \qquad for \quad n = 0,1,........16$$

- Even though k varies from 0 to 16 since we considered ω varying between 0 and π/2 only k values from 0 to 8 are considered
- While finding h(n) we observe symmetry in h(n) such that n varying 0 to 7 and 9 to 16 have same set of h(n)

### 7.10 Design of FIR Differentiator

Differentiators are widely used in Digital and Analog systems whenever a derivative of the signal is needed. Ideal differentiator has pure linear magnitude response in the freq range –π to +π. The typical frequency response characteristics is as shown in the below figure.



**Problem 2: Design an Ideal Differentiator using a) rectangular window and b)Hamming window with length of the system = 7.**

**Solution:**
As seen from differentiator frequency chars. It is defined as

$H(e^{j\omega}) = j\omega$ between $-\pi$ to $+\pi$

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} j\omega\, e^{j\omega n}\, d\omega = \frac{\cos \pi n}{n} \qquad -\infty \le n \le \infty \quad and \quad n \ne 0$$

The $h_d(n)$ is an add function with $h_d(n)=-h_d(-n)$ and $h_d(0)=0$

a) rectangular window

$h(n)=h_d(n)w_r(n)$

h(1)=-h(-1)=hd(1)=-1
h(2)=-h(-2)=hd(2)=0.5
h(3)=-h(-3)=hd(3)=-0.33

h'(n)=h(n-3) for causal system
thus,

$$H'(z) = 0.33 - 0.5z^{-1} + z^{-2} - z^{-4} + 0.5z^{-5} - 0.33z^{-6}$$

Also from the equation

$$H_r(e^{j\omega}) = 2\sum_{n=0}^{(M-3)/2} h(n)\sin \omega\left(\frac{M-1}{2} - n\right)$$

For M=7 and h'(n) as found above we obtain this as

$$H_r(e^{j\omega}) = 0.66\sin 3\omega - \sin 2\omega + 2\sin \omega$$

$$H(e^{j\omega}) = jH_r(e^{j\omega}) = j(0.66\sin 3\omega - \sin 2\omega + 2\sin \omega)$$

b) Hamming window
$h(n)=h_d(n)w_h(n)$

where $w_h(n)$ is given by

$$w_h(n) = 0.54 + 0.46\cos\frac{2\pi n}{(M-1)} \quad -(M-1)/2 \le n \le (M-1)/2$$
$$= 0 \quad otherwise$$

For the present problem

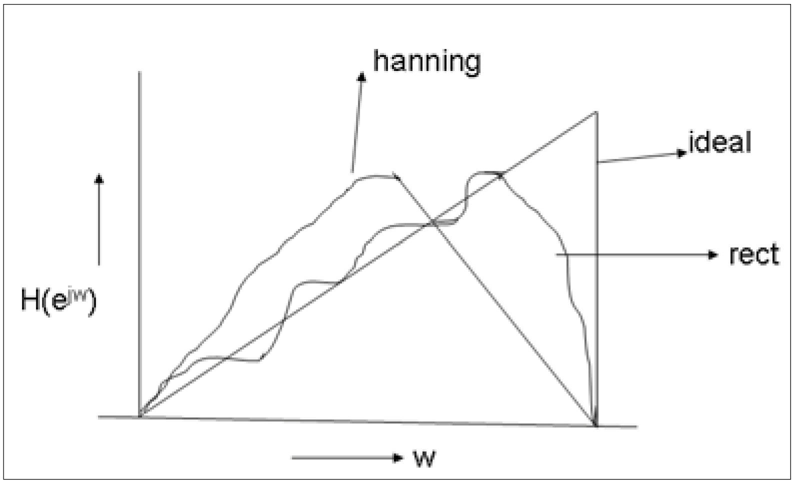
$$w_h(n) = 0.54 + 0.46\cos\frac{\pi n}{3} \quad -3 \le n \le 3$$

The window function coefficients are given by for n=-3 to +3
Wh(n)= [0.08 0.31 0.77 1 0.77 0.31 0.08]

Thus h'(n) = h(n-5) = [0.0267, -0.155, 0.77, 0, -0.77, 0.155, -0.0267]

Similar to the earlier case of rectangular window we can write the freq response of differentiator as

$$H(e^{j\omega}) = jH_r(e^{j\omega}) = j(0.0534\sin 3\omega - 0.31\sin 2\omega + 1.54\sin\omega)$$



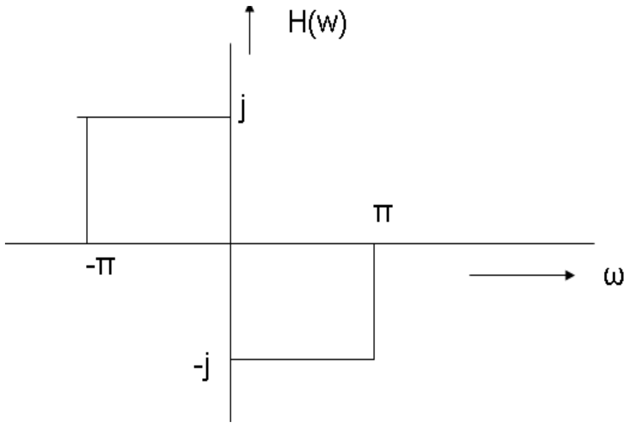

We observe

- With rectangular window, the effect of ripple is more and transition band width is small compared with hamming window
- With hamming window, effect of ripple is less whereas transition band is more

## 7.11 Design of FIR Hilbert transformer:

Hilbert transformers are used to obtain phase shift of 90 degree. They are also called j operators. They are typically required in quadrature signal processing. The Hilbert transformer is very useful when out of phase component (or imaginary part) need to be generated from available real component of the signal.

**Problem 3: Design an ideal Hilbert transformer using a) rectangular window and b) Blackman Window with M = 11**




**Solution:**

As seen from freq chars it is defined as

$$
\begin{aligned}
H_d(e^{j\omega}) &= j \quad -\pi \le \omega \le 0 \\
&= -j \qquad 0 \le \omega \le \pi
\end{aligned}
$$

The impulse response is given by

$$h_d(n) = \frac{1}{2\pi}\left[\int_{-\pi}^{0} je^{j\omega n}\,d\omega + \int_{0}^{\pi} -je^{j\omega n}\,d\omega\right] = \frac{(1-\cos \pi n)}{\pi n} \quad -\infty \le n \le \infty \quad except \quad n = 0$$

At n = 0 it is hd(0) = 0 and hd(n) is an odd function

a) Rectangular window

$h(n) = h_d(n)\, w_r(n) = h_d(n)$ for $-5 \ge n \ge 5$

h'(n)=h(n-5)

h(n)= [-0.127, 0, -0.212, 0, -0.636, 0, 0.636, 0, 0.212, 0, 0.127]

$$H_r(e^{j\omega}) = 2\sum_{n=0}^{4} h(n)\sin\omega(5-n)$$

$$H(e^{j\omega}) = j\,|H_r(e^{j\omega})| = j\{0.254\sin 5\omega + 0.424\sin 3\omega + 1.272\sin\omega\}$$

b) Blackman Window
window function is defined as

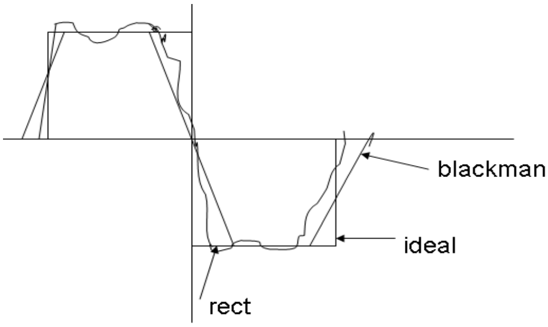
$$w_b(n) = 0.42 + 0.5\cos\frac{\pi n}{5} + 0.08\cos\frac{2\pi n}{5} \qquad -5 \le n \le 5$$

$$= 0 \qquad \textit{otherwise}$$

$W_b(n)$ = [0, 0.04, 0.2, 0.509,0.849,1,0.849, 0.509, 0.2, 0.04,0] for -5≥n≥5

h’(n) = h(n-5) = [0, 0, -0.0424, 0, -0.5405, 0, 0.5405, 0, 0.0424, 0, 0]

$$H(e^{j\omega}) = -j[0.0848\sin 3\omega + 1.0810\sin\omega]$$



# Recommended questions with solution

## Question1

Design an FIR linear phase, digital filter approximating the ideal frequency response

$$H_d(\omega) = \begin{cases} 1, & \text{for } |\omega| \le \dfrac{\pi}{6} \\ 0, & \text{for } \dfrac{\pi}{6} < |\omega| \le \pi \end{cases}$$

**(a)** Determine the coefficients of a 25-tap filter based on the window method with a rectangular window.
**(b)** Determine and plot the magnitude and phase response of the filter.
**(c)** Repeat parts (a) and (b) using the Hamming window.
**(d)** Repeat parts (a) and (b) using a Bartlett window.

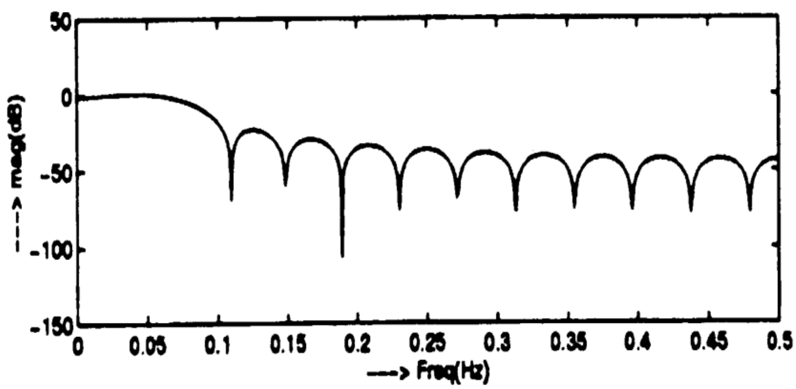
**Solution:-**

(a) To obtain the desired length of 25, a delay of $\frac{25-1}{2} = 12$ is incorporated into $H_d(w)$. Hence,

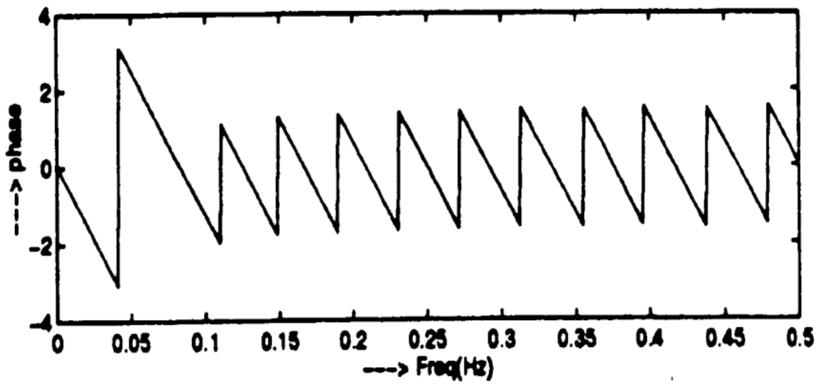
$$\begin{aligned} H_d(w) &= 1e^{-j12w}, \quad 0 \le |w| \le \frac{\pi}{6} \\ &= 0, \quad \text{otherwise} \\ h_d(n) &= \frac{1}{2\pi}\int_{-\frac{\pi}{6}}^{\frac{\pi}{6}} H_d(w)e^{-jwn}dw \\ &= \frac{\sin\frac{\pi}{6}(n-12)}{\pi(n-12)} \\ \text{Then, } h(n) &= h_d(n)w(n) \end{aligned}$$

where $w(n)$ is a rectangular window of length $N = 25$.

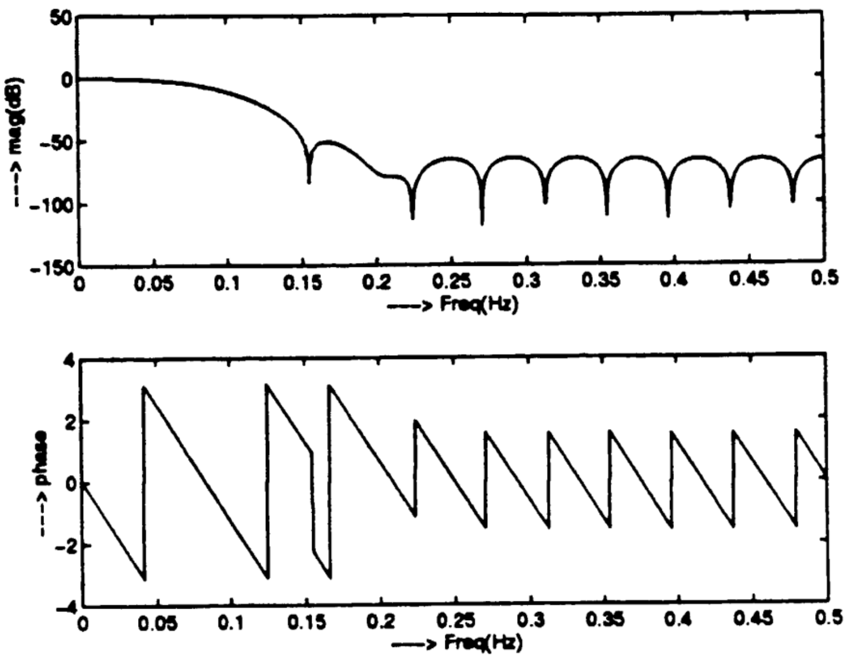
**(b) Magnitude plot**



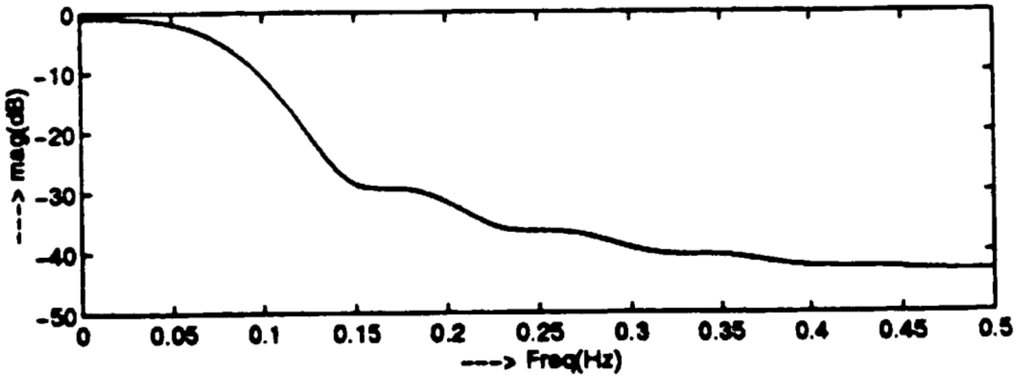
**Phase plot**

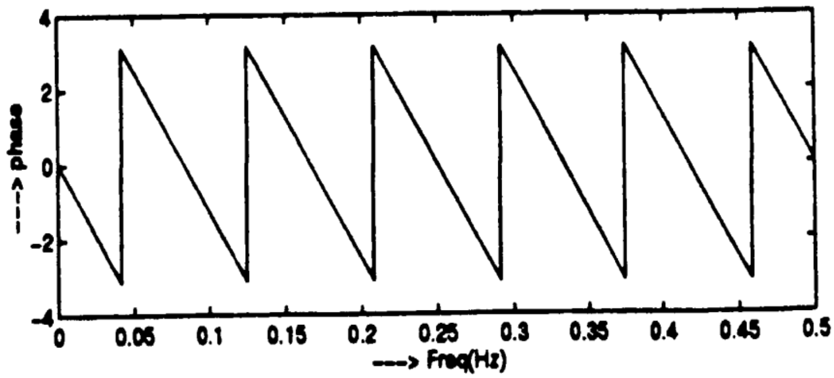



**(c) Hamming window**




**(d) Bartlett window**




## Question 2

Determine the unit sample response $\{h(n)\}$ of a linear-phase FIR filter of length $M = 4$ for which the frequency response at $\omega = 0$ and $\omega = \pi/2$ is specified as

$$H_r(0) = 1 \qquad H_r\left(\frac{\pi}{2}\right) = \frac{1}{2}$$

**Solution:-**

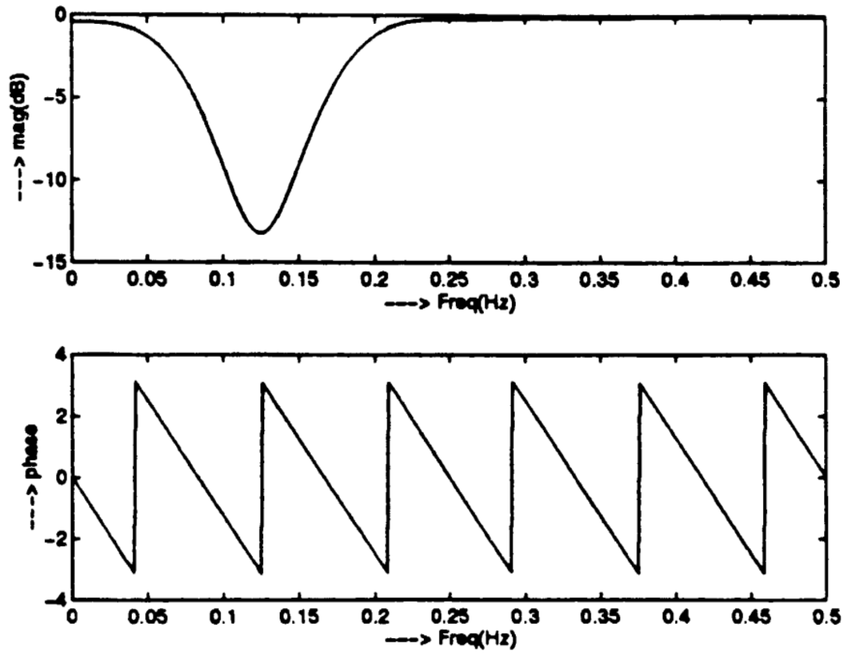
$$\begin{aligned}
M &= 4, \quad H_r(0) = 1, \quad H_r(\frac{\pi}{2}) = \frac{1}{2} \\
H_r(w) &= 2\sum_{n=0}^{\frac{M}{2}-1} h(n)cos[w(\frac{M-1}{2} - n)] \\
&= 2\sum_{n=0}^{1} h(n)cos[w(\frac{3}{2} - n)] \\
\text{At } w = 0, H_r(0) = 1 &= 2\sum_{n=0}^{1} h(n)cos[0] \\
2[h(0) + h(1)] &= 1 \qquad (1)
\end{aligned}$$

$$\text{At } w = \frac{\pi}{2}, H_r(\frac{\pi}{2}) = \frac{1}{2} = 2\sum_{n=0}^{1} h(n)cos[\frac{\pi}{2}(\frac{3}{2} - n)]$$

$$-h(0) + h(1) = 0.354 \qquad (2)$$

Solving (1) and (2), we get

$$\begin{aligned} h(0) &= 0.073 \text{ and} \\ h(1) &= 0.427 \\ h(2) &= h(1) \\ h(3) &= h(0) \\ \text{Hence, } h(n) &= \{0.073, 0.427, 0.427, 0.073\} \end{aligned}$$



## Question 3

Use the window method with a Hamming window to design a 21-tap differentiator as shown in Fig. P8.9. Compute and plot the magnitude and phase response of the resulting filter.

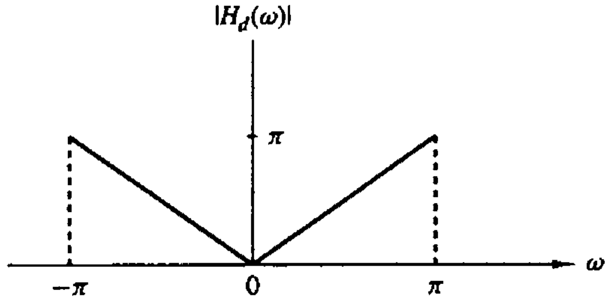
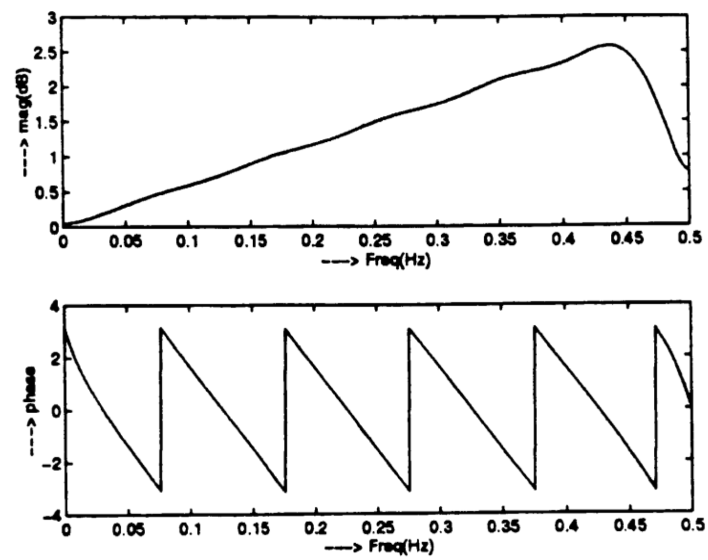



**Figure P8.9**

**Solution:-**

$$
\begin{aligned}
H_d(w) &= we^{-j10w}, \quad 0 \le w \le \pi \\
&= -we^{-j10w}, \quad -\pi \le w \le 0 \\
h_d(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(w)e^{-jwn}dw \\
&= \frac{\cos\pi(n-10)}{(n-10)}, \quad n \neq 10 \\
&= 0, \quad n = 10 \\
h_d(n) &= \frac{\cos\pi(n-10)}{(n-10)}, \quad 0 \le n \le 20, n \neq 10 \\
&= 0, \quad n = 10
\end{aligned}
$$

## Magnitude and phase response



## Question 4

A digital low-pass filter is required to meet the following specifications:

Passband ripple: $\leq 1$ dB
Passband edge: 4 kHz
Stopband attenuation: $\geq 40$ dB
Stopband edge: 6 kHz
Sample rate: 24 kHz

The filter is to be designed by performing a bilinear transformation on an analog system function. Determine what order Butterworth, Chebyshev, and elliptic analog designs must be used to meet the specifications in the digital implementation.

**Solution:-**

From the design specifications we obtain

$$\begin{aligned}
\epsilon &= 0.509 \\
\delta &= 99.995 \\
f_p &= \frac{4}{24} = \frac{1}{6} \\
f_s &= \frac{6}{24} = \frac{1}{4} \\
\text{Assume } t = 1. \text{ Then, } \Omega_p &= 2tan\frac{w_p}{2}
\end{aligned}$$

$$\begin{aligned}
&= 2tan\pi f_p = 1.155 \\
\text{and } \Omega_s &= 2tan\frac{w_s}{2} \\
&= 2tan\pi f_s = 2 \\
\eta &= \frac{\delta}{\epsilon} = 196.5
\end{aligned}$$

# UNIT 8

# Design of IIR Filters from Analog Filters

## CONTENTS:-

DESIGN OF IIR FILTERS FROM ANALOG FILTERS (BUTTERWORTH AND CHEBYSHEV) - IMPULSE INVARIANCE METHOD. MAPPING OF TRANSFER FUNCTIONS: APPROXIMATION OF DERIVATIVE (BACKWARD DIFFERENCE AND BILINEAR TRANSFORMATION) METHOD, MATCHED Z TRANSFORMS, VERIFICATION FOR STABILITY AND LINEARITY DURING MAPPING 7 HRS

## RECOMMENDED READINGS:-


1. DIGITAL SIGNAL PROCESSING – PRINCIPLES ALGORITHMS & APPLICATIONS, PROAKIS & MONALAKIS, PEARSON EDUCATION, 4TH EDITION, NEW DELHI, 2007.
2. DISCRETE TIME SIGNAL PROCESSING, OPPENHEIM & SCHAFFER, PHI, 2003.
3. DIGITAL SIGNAL PROCESSING, S. K. MITRA, TATA MC-GRAW HILL, 2ND EDITION, 2004.

# UNIT - 8

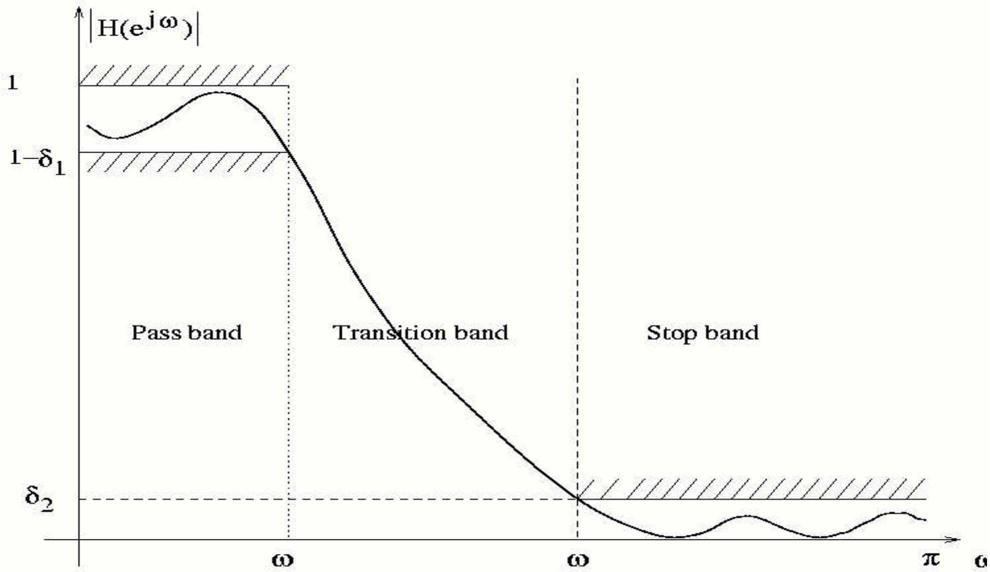
## DESIGN OF IIR FILTERS FROM ANALOG FILTERS (BUTTERWORTH AND CHEBYSHEV)

### 8.1 Introduction

A digital filter is a linear shift-invariant discrete-time system that is realized using finite precision arithmetic. The design of digital filters involves three basic steps:

- The specification of the desired properties of the system.
- The approximation of these specifications using a causal discrete-time system.
- The realization of these specifications using _nite precision arithmetic.

These three steps are independent; here we focus our attention on the second step. The desired digital filter is to be used to filter a digital signal that is derived from an analog signal by means of periodic sampling. The speci_cations for both analog and digital filters are often given in the frequency domain, as for example in the design of low
pass, high pass, band pass and band elimination filters. Given the sampling rate, it is straight




forward to convert from frequency specifications on an analog _lter to frequency speci_cations on the corresponding digital filter, the analog frequencies being in terms of Hertz and digital frequencies being in terms of radian frequency or angle around the unit circle with the point Z=-1 corresponding to half the sampling frequency. The least confusing point of view toward digital filter design is to consider the filter as being specified in terms of angle around the unit circle rather than in terms of analog frequencies.

Figure 7.1: Tolerance limits for approximation of ideal low-pass filter

A separate problem is that of determining an appropriate set of specifications on the digital filter. In the case of a low pass filter, for example, the specifications often take the form of a tolerance scheme, as shown in Fig. 4.1

$$1-\delta_1 \leq |H(e^{j\omega})| \leq 1, \quad |\omega| \leq \omega_p$$

$$|H(e^{j\omega})| \leq \delta_2, \quad \omega_s \leq |\omega| \leq \pi$$

Many of the filters used in practice are specified by such a tolerance scheme, with no constraints on the phase response other than those imposed by stability and causality requirements; i.e., the poles of the system function must lie inside the unit circle. Given a set of specifications in the form of Fig. 7.1, the next step is to and a discrete time linear system whose frequency response falls within the prescribed tolerances. At this point the filter design problem becomes a problem in approximation. In the case of infinite impulse response (IIR) filters, we must approximate the desired frequency response by a rational function, while in the finite impulse response (FIR) filters case we are concerned with polynomial approximation.

## 7.2 Design of IIR Filters from Analog Filters:

The traditional approach to the design of IIR digital filters involves the transformation of an analog filter into a digital filter meeting prescribed specifications. This is a reasonable approach because:

- The art of analog filter design is highly advanced and since useful results can be achieved, it is advantageous to utilize the design procedures already developed for analog filters.
- Many useful analog design methods have relatively simple closed-form design formulas.

Therefore, digital filter design methods based on analog design formulas are rather simple to implement.
An analog system can be described by the differential equation

$$\sum_{k=0}^{N} c_k \frac{d^k y_a(t)}{dt^k} = \sum_{k=0}^{M} d_k \frac{d^k x_a(t)}{dt^k}$$ ---------------------------------------------------------------7.1

And the corresponding rational function is

$$H_a(s) = \frac{\sum_{k=0}^{M} d_k s^k}{\sum_{k=0}^{N} c_k s^k} = \frac{y_a(s)}{x_a(s)}$$ ---------------------------------------------------------7.2

The corresponding description for digital filters has the form

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k)$$ -------------------------------------------------7.3

and the rational function

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}} = \frac{Y(z)}{X(z)}$$ ------------------------------------------------------7.4

In transforming an analog filter to a digital filter we must therefore obtain either H(z)or h(n) (inverse Z-transform of H(z) i.e., impulse response) from the analog filter design. In such transformations, we want the imaginary axis of the S-plane to map into the finite circle of the Z-plane, a stable analog filter should be transformed to a stable digital filter. That is, if the analog filter has poles only in the left-half of S-plane, then the digital filter must have poles only inside the unit circle. These constraints are basic to all the techniques discussed

## 7.3 IIR Filter Design by Impulse Invariance:

This technique of transforming an analog filter design to a digital filter design corresponds to choosing the unit-sample response of the digital filter as equally spaced samples of the impulse response of the analog filter. That is,

$$h(n) = h_a(nT)$$

-------------------------------------------------------------------------7.5

Where T is the sampling period. Because of uniform sampling, we have

$$H(e^{j\omega T}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} H_a(j\Omega + j\frac{2\pi}{T}k)$$

---------------------------------------------7.6

Or

$$H(z)\mid_{z=e^{sT}} = \frac{1}{T}\sum_{k=-\infty}^{\infty} H_a(s + j\frac{2\pi}{T}k)$$

---------------------------------------------7.7

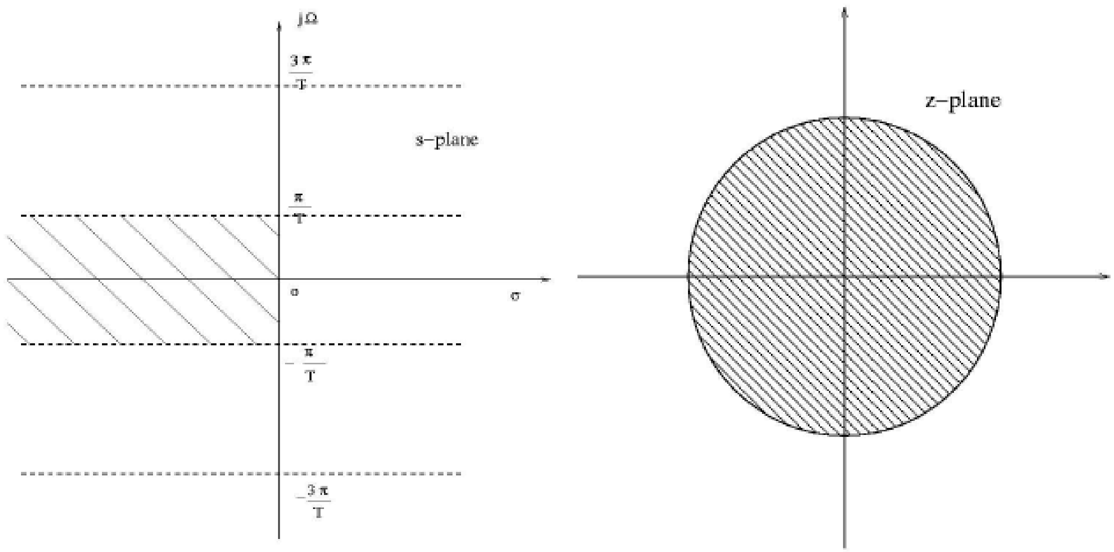



Figure 7.2: Mapping of s-plane into z-plane

Where s = jω and Ω=ω/T, is the frequency in analog domain and ω is the frequency in digital domain.

From the relationship Z = $e^{ST}$ it is seen that strips of width 2π/T in the S-plane map into the entire Z-plane as shown in Fig. 7.2. The left half of each S-plane strip maps into interior of the unit circle, the right half of each S-plane strip maps into the exterior of the unit circle, and the imaginary axis of length 2π/T of S-plane maps on to once round the unit circle of Z-plane. Each horizontal strip of the S-plane is overlaid onto the Z-plane to form the digital filter function from analog filter function. The frequency response of the digital filter is related to the frequency response of the

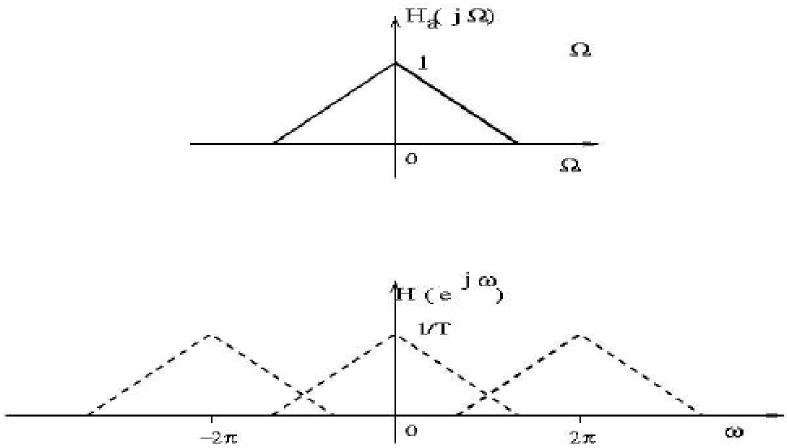



Figure 7.3: Illustration of the effects of aliasing in the impulse invariance technique

analog filter as

$$H(e^{j\omega}) = \frac{1}{T}\sum_{k=-\infty}^{\infty} H_a(j\frac{\omega}{T} + j\frac{2\pi}{T}k) \qquad \text{-----------------------------------------------7.8}$$

From the discussion of the sampling theorem it is clear that if and only if

$$H_a(j\Omega) = 0, \quad |\,\Omega\,| \geq \frac{\pi}{T}$$

Then

$$H(e^{j\omega}) = \frac{1}{T}H_a(j\frac{\omega}{T}), \quad |\,\omega\,| \leq \pi$$

Unfortunately, any practical analog filter will not be band limited, and consequently there is interference between successive terms in Eq. (7.8) as illustrated in Fig. 7.3. Because of the aliasing that occurs in the sampling process, the frequency response of the resulting digital filter will not be identical to the original analog frequency response. To get the filter design procedure, let us consider the system function of the analog filter expressed in terms of a partial-fraction expansion

$$H_a(s) = \sum_{k=1}^{N} \frac{A_k}{s - s_k} \qquad \text{----------------------------------------------------------------------7.9}$$

The corresponding impulse response is

$$h_a(t) = \sum_{k=1}^{N} A_k e^{s_k t} U(t)$$ ------------------------------------------------------------- 7.10

And the unit-sample response of the digital filter is then

$$h(n) = h_a(nT) = \sum_{k=1}^{N} A_k e^{s_k nT} u(n) = \sum_{k=1}^{N} A_k (e^{s_k T})^n U(n)$$ --------------7.11

The system function of the digital filter H(z) is given by

$$H(z) = \sum_{k=1}^{N} \frac{A_k}{(1 - \exp^{s_k T} z^{-1})}$$ ----------------------------------------------------------7.12

In comparing Eqs. (7.9) and (7.12) we observe that a pole at s=sk in the S-plane transforms to a pole at $\exp^{skT}$ in the Z-plane. It is important to recognize that the impulse invariant design procedure does not correspond to a mapping of the S-plane to the Z-plane.

## 8.4 IIR Filter Design By Approximation Of Derivatives:

A second approach to design of a digital filter is to approximate the derivatives in Eq. (4.1) by finite differences. If the samples are closer together, the approximation to the derivative would be increasingly accurate. For example, suppose that the first derivative is approximated by the first backward difference

$$\frac{dy_a(t)}{dt} \bigg|_{t=nT} \longrightarrow \nabla^{(1)}[y(n)] = \frac{y(n) - y(n-1)}{T}$$ -------------------------7.13

Where y(n)=y(nT). Approximation to higher-order derivatives are obtained by repeated application of Eq. (7.13); i.e.,

$$\frac{d^k y_a(t)}{dt^k} \bigg|_{t=nT} = \frac{d}{dt}\left(\frac{d^{k-1} y_a(t)}{dt^{k-1}}\right) \bigg|_{t=nT} \longrightarrow \nabla^{(k)}[y(n)] = \nabla^{(1)}[\nabla^{(k-1)}[y(n)]]$$ ------------------------- 7.14

For convenience we define

$$\nabla^{(0)}[y(n)] = y(n)$$ ----------------------------------------------------------------7.15

Applying Eqs. (7.13), (7.14) and (7.15) to (7.1), we obtain

$$\sum_{k==0}^{N} c_k \nabla^{(k)}[y(n)] = \sum_{k=0}^{M} d_k \nabla^{(k)}[x(n)]$$ --------------------------------------------7.16

Where y(n) = ya(nT) and x(n) = xa(nT). We note that the operation $\Delta^{(1)}[\ ]$ is a linear shift-invariant operator and that $\Delta^{(k)}[\ ]$ can be viewed as a cascade of (k) operators $\Delta^{(1)}[\ ]$. In particular

$$Z[\nabla^{(1)}[x(n)]] = [\frac{1-z^{-1}}{T}]X(z)$$

And

$$Z[\nabla^{(1)}[x(n)]] = [\frac{1-z^{-1}}{T}]^k X(z)$$

Thus taking the Z-transform of each side in Eq. (7.16), we obtain

$$H(z) = \frac{\sum_{k=0}^{M} d_k[\frac{1-z^{-1}}{T}]^k}{\sum_{k=0} N c_k[\frac{1-z^{-1}}{T}]^k}$$ ----------------------------------------------------------7.17

Comparing Eq. (7.17) to (7.2), we observe that the digital transfer function can be obtained directly from the analog transfer function by means of a substitution of variables

$$s = \frac{1-z^{-1}}{T}$$ -------------------------------------------------------------------------------7.18

So that, this technique does indeed truly correspond to a mapping of the S-plane to the Z-plane, according to Eq. (7.18). To investigate the properties of this mapping, we must express z as a function of s, obtaining

$$z = \frac{1}{1-sT}$$

Substituting s = jΩ, i.e., imaginary axis in S-plane

$$
\begin{aligned}
z &= \frac{1}{1-j\Omega T} \\
&= \frac{1}{1-j\Omega T}+\frac{1}{2}-\frac{1}{2} \\
&= \frac{1}{2}+\frac{1}{2}[\frac{1+j\Omega T}{1-j\Omega T}] \\
&= \frac{1}{2}[1+\frac{1+j\Omega T}{1-j\Omega T}] \\
&= \frac{1}{2}[1+e^{j2\tan^{-1}(\Omega T)}]
\end{aligned}
$$

-----------------------------------------------------7.19

Which corresponds to a circle whose center is at z =1/2 and radius is 1/2, as shown in Fig. 7.4. It is easily verified that the left half of the S-plane maps into the inside of the small circle and the right half of the S-plane maps onto the outside of the small circle. Therefore, although the requirement of mapping the jΩ-axis to the unit circle is not satisfied, this mapping does satisfy the stability condition.

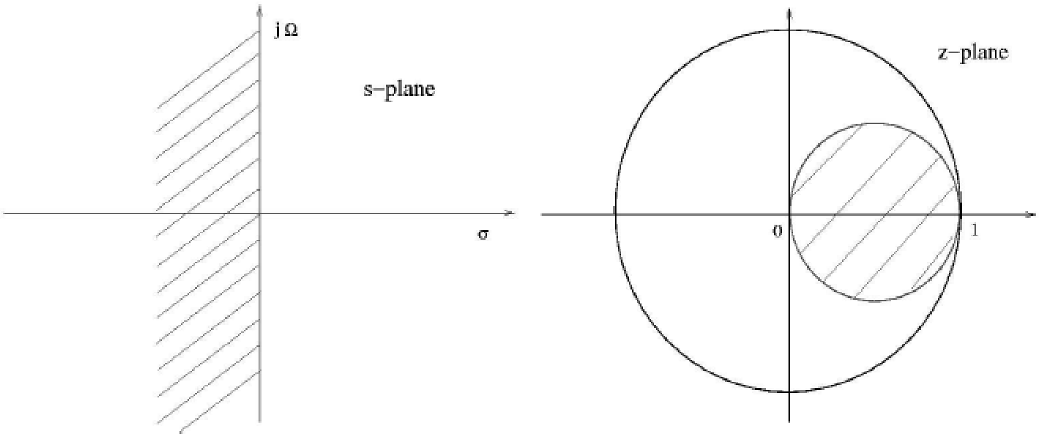



Figure 4.4: Mapping of s-plane to z-plane corresponding to first backward-difference approximation to the derivative

In contrast to the impulse invariance technique, decreasing the sampling period T, theoretically produces a better filter since the spectrum tends to be concentrated in a very small region of the unit circle. These two procedures are highly unsatisfactory for anything but low pass filters. An alternative approximation to the derivative is a forward difference and it provides a mapping into the unstable digital filters.

## 8.5 IIR Filter Design By The Bilinear Transformation:

In the previous section a digital filter was derived by approximating derivatives by differences. An alternative procedure is based on integrating the differential equation and then using a numerical approximation to the integral. Consider the first - order equation

$$c_1 y_a'(t) + c_0 y_a(t) = d_0 x_a(t)$$ -----------------------------------------------------------7.20

Where $y'_a(t)$ is the first derivative of ya(t). The corresponding analog system function is

$$H_a(s) = \frac{d_0}{c_0 + c_1 s}$$

We can write ya(t) as an integral of $y'_a(t)$, as in

$$y_a(t) = \int_{t_0}^{t} y_a'(t)dt + y_a(t_0)$$

In particular, if t = nT and $t_0$ = (n - 1)T,

$$y_a(nT) = \int_{(n-1)T}^{nT} y_a'(\tau)d\tau + y_a((n-1)T)$$

If this integral is approximated by a trapezoidal rule, we can write

$$y_a(nT) = y_a((n-1)T) + \frac{T}{2}[y_a'(nT) + y_a'((n-1)T)]$$ -----------------------7.21

However, from Eq. (7.20),

$$y_a'(nT) = -\frac{c_0}{c_1} y_a(nT) + \frac{d_0}{c_1} x_a(nT)$$

Substituting into Eq. (4.21) we obtain

$$[y(n) - y(n-1)] = \frac{T}{2}[-\frac{c_0}{c_1}(y(n) + y(n-1)) + \frac{d_0}{c_1}(x(n) + x(n-1))]$$

Where y(n) = y(nT) and x(n) = x(nT). Taking the Z-transform and solving for H(z) gives

$$H(z) = \frac{Y(z)}{X(z)} = \frac{d_0}{c_0 + c_1 \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}}$$ -------------------------------------------7.22

From Eq. (7.22) it is clear that H(z) is obtained from Ha(s) by the substitution

$$s = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}$$
-----------------------------------------------------------------7.23

That is,

$$H(z) = H_a(s)\,|_{s=\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}}$$
-----------------------------------------------------------------7.24

This can be shown to hold in general since an $N^{th}$ - order differential equation of the form of Eq. (7.1) can be written as a set of N first-order equations of the form of Eq. (7.20). Solving Eq. (7.23) for z gives

$$z = \frac{1+\frac{T}{2}s}{1-\frac{T}{2}s}$$
-----------------------------------------------------------------------7.25

The invertible transformation of Eq. (7.23) is recognized as a bilinear transformation. To see that this mapping has the property that the imaginary axis in the s-plane maps onto the unit circle in the z-plane, consider $z = e^{j\omega}$, then from Eq. (7.23), s is given by

$$\begin{aligned} s &= \frac{2}{T}\frac{1-e^{-j\omega}}{1+e^{-j\omega}} \\ &= \frac{2}{T}\frac{j\sin(\omega/2)}{\cos(\omega/2)} \\ &= \frac{2}{T}j\tan(\omega/2) \\ &= \sigma + j\Omega \end{aligned}$$

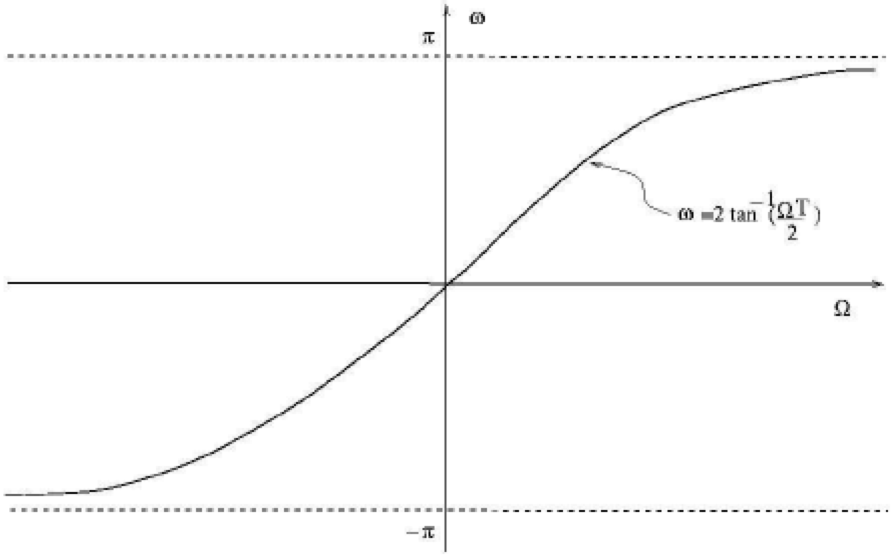



Figure 7.5: Mapping of analog frequency axis onto the unit circle using the bilinear Transformation

Thus for z on the unit circle, $\sigma = 0$ and $\Omega$ and $\omega$ are related by

$$T\,\Omega/2 = \tan(\omega/2)$$
or
$$\omega = 2\tan^{-1}(T\,\Omega/2)$$

This relationship is plotted in Fig. (7.5), and it is referred as frequency warping. From the _gure it is clear that the positive and negative imaginary axis of the s-plane are mapped, respectively, into the upper and lower halves of the unit circle in the z-plane. In addition to the fact that the imaginary axis in the s-plane maps into the unit circle in the z-plane, the left half of the s-plane maps to the inside of the unit circle and the right half of the s-plane maps to the outside of the unit circle, as shown in Fig. (7.6). Thus we see that the use of the bilinear transformation yields stable digital filter from analog filter. Also this transformation avoids the problem of aliasing encountered with the use of impulse invariance, because it maps the entire imaginary axis in the s-plane onto the unit circle in the z-plane. The price paid for this, however, is the introduction of a distortion in the frequency axis.

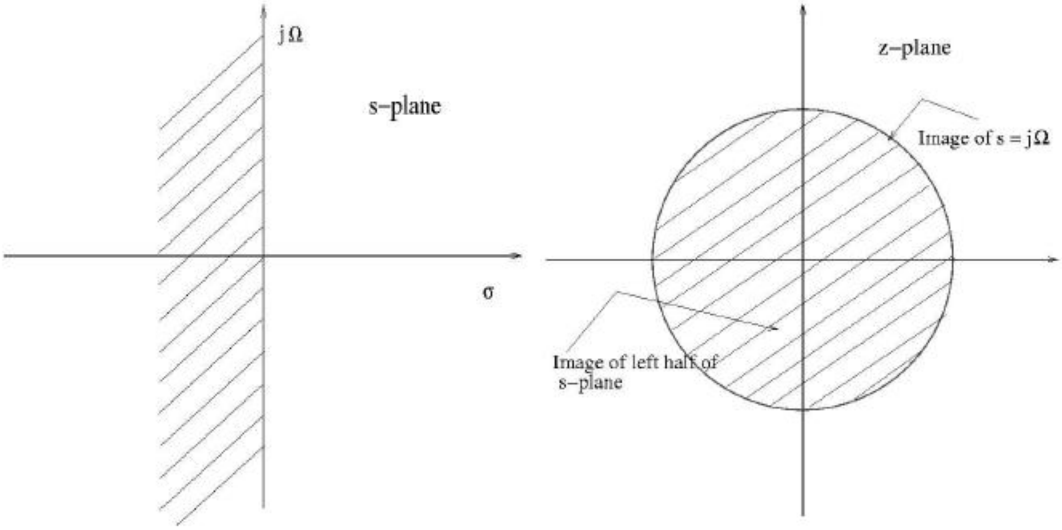



Figure 4.6: Mapping of the s-plane into the z-plane using the bilinear transformation

## 8.6 The Matched-Z Transform:

Another method for converting an analog filter into an equivalent digital filter is to map the poles and zeros of Ha(s) directly into poles and zeros in the z-plane. For analog filter

$$H_a(s) = \frac{\prod_{k=1}^{M}(s - s_k)}{\prod_{k=1}^{N}(s - p_k)}$$ -----------------------------------------------------------------7.26

the corresponding digital filter is

$$H(z) = \frac{\prod_{k=1}^{M}(1 - e^{z_k T} z^{-1})}{\prod_{k=1}^{N}(1 - e^{p_k T} z^{-1})}$$ ----------------------------------------------------------7.27

Where T is the sampling interval. Thus each factor of the form (s-a) in Ha(s) is mapped into the factor $(1- e^{aT} z^{-1})$.

# Recommended questions with solution

**Question 1**

*Design a digital band pass filter from a $2^{nd}$ order analog low pass Butterworth prototype filter using bilinear transformation. The lower and upper cut-off frequencies for band pass filter are $5\pi/12$ and $7\pi/12$. Assume T = 2 sec.* **[12]**

**Sol. :**

$$\omega_l = \frac{5\pi}{12}$$

$$\Omega_l = \frac{2}{T}\tan\frac{\omega_l}{2}$$

For

$$T = 2$$

$$\Omega_l = \tan\frac{\omega_l}{2}$$

$$\omega_u = \frac{7\pi}{12}$$

$$\therefore \quad \Omega_u = \tan\frac{\omega_u}{2}$$

Analog low pass to band pass

$$s \rightarrow \frac{s^2+\Omega_l\Omega_u}{s(\Omega_u-\Omega_l)} \qquad \text{... (1)}$$

Analog prototype is,

$$H(s) = \frac{1}{s^2+\sqrt{2}s+1} \qquad \text{... (2)}$$

Putting equation (1) in equation (2) and then to get bilinear analog to digital

$$s \rightarrow \frac{z-1}{z+1} = \frac{1-z^{-1}}{1+z^{-1}}$$

Combining above two steps we get

$$s \rightarrow \frac{\left(\frac{1-z^{-1}}{1+z^{-1}}\right)^2+\Omega_l\Omega_u}{\left(\frac{1-z^{-1}}{1+z^{-1}}\right)(\Omega_u-\Omega_l)} = \frac{(1-z^{-1})^2+\Omega_l\,\Omega_u(1+z^{-1})^2}{(1-z^{-2})(\Omega_u-\Omega_l)}$$

$$\therefore \quad H(z) = \frac{1}{\left[\dfrac{(1-z^{-1})^2+\Omega_l\Omega_u(1+z^{-1})^2}{(1-z^{-2})(\Omega_u-\Omega_l)}\right]^2+\sqrt{2}\left[\dfrac{(1-z^{-1})^2+\Omega_l\Omega_u(1+z^{-1})^2}{(1-z^{-2})(\Omega_u-\Omega_l)}\right]+1}$$

**Question 2**

*Show that the bilinear transformation maps.*

*i) The $j\Omega$ axis in s-plane onto the unit circle, $|z| = 1$.*

*ii) The left half s-plane, Re (s) < 0 inside the unit circle, $|z < 1|$.*

**Sol. :**

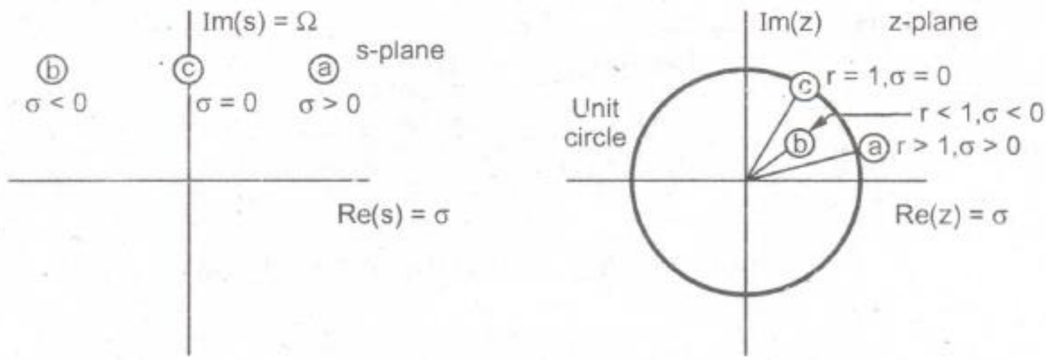



**Fig. 3**

Here $\quad s = \sigma + j\Omega \quad$ and $z = re^{j\omega}$

**Question 3**

*Fig. 4 shows the frequency response of an infinite-length ideal multi-band real filter. Find. h(n), impulse response of this filter. Present the sketch of implementation of $\omega(n)\,h(n)$ (Truncated impulse response of this filter) via block diagram. Where $\omega(n)$ is a finite length window sequence ?* **[12]**

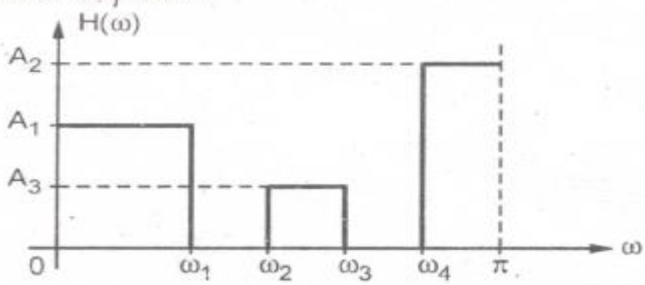



**Fig. 4**

**Question 4**

*We are interested to design an FIR filter with a stopband attenuation of 64 dB and $\Delta\omega = 0.05\pi$ using windows. Provide the means to achieve precisely this attenuation using suitable window function.* **[3]**

**Sol. :** Hamming window will satisfy the stopband attenuation requirement i.e. 64 dB. Because it has lower transition width.

Hamming window function is given by :

$$\omega(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right) \qquad 0 \le n \le N-1$$

**Question 5**

*The transfer function of analog low pass filter is given by* $H(s) = \dfrac{(s-1)}{(s^2-1)(s^2+s+1)}$.

*Find H(z) using impulse invariance method. Take T = 1 sec.* **[6]**

**Sol. :**

$$H(s) = \frac{(s-1)}{(s^2-1)(s^2+s+1)}$$

$$= \frac{(s-1)}{(s+1)(s-1)(s^2+s+1)}$$

$$= \frac{1}{(s+1)(s^2+s+1)}$$

$$= \frac{1}{(s+1)(s+0.5-j\,0.866)(s+0.5+j\,0.866)}$$

$$= \frac{C_1}{s+1} + \frac{C_2}{s+0.5-j\,0.866} + \frac{C_2^*}{s+0.5+j\,0.866}$$

Using practical fraction expansion, we get

$$C_1 = 1, \quad C_2 = 0.577e^{-j2.62} \quad \text{and} \quad C_2^* = 0.577e^{j2.62}$$

$$\therefore \qquad H(s) = \frac{1}{s+1} + \frac{0.577e^{-j2.62}}{s+0.5-j\,0.866} + \frac{0.577e^{j2.62}}{s+0.5+j\,0.866}$$

The three poles are :

$$s_1 = -1, \quad s_2 = -0.5 + j\,0.866 \quad \text{and} \quad s_3 = -0.5 - j\,0.866$$

We know that,

$$H(z) = \sum_{i=1}^{3} \frac{C_i}{1-e^{S_i T} z^{-1}}$$

$$= \frac{C_1}{1-e^{s_1 T} z^{-1}} + \frac{C_2}{1-e^{s_2 T} z^{-1}} + \frac{C_3}{1-e^{s_3 T} z^{-1}}$$

Here $C_3 = C_2^*$

$$\therefore \quad H(z) = \frac{1}{1-e^{-T} z^{-1}} + \frac{0.577 e^{-j2.62}}{1-e^{(-0.5+j0.866)T} z^{-1}} + \frac{0.577 e^{j2.62}}{1-e^{(0.5-j0.866)T} z^{-1}}$$

$$= \frac{1}{1-e^{-T} z^{-1}} + \frac{0.577 e^{-j2.62}}{1-e^{-0.5T} e^{j0.866T} z^{-1}} + \frac{0.577 e^{j2.62}}{1-e^{-0.5T} e^{-j0.866T} z^{-1}}$$

$$= \frac{1}{1-e^{-T} z^{-1}} + \frac{2(0.577)\cos(-2.62) - 2(0.577) e^{-0.5T} z^{-1} \cos(-2.62-0.866T)}{1-2e^{-0.5T} \cos(0.866T) z^{-1} + e^{-T} z^{-2}}$$

Multiplying the numerator and denominator of first term on RHS by z and by $\tau^2$ for second term on RHS, above equation becomes,

$$H(z) = \frac{z}{z-e^{-T}} + \frac{-z^2 - 1.154 e^{-0.5T} \cos\left(\frac{5\pi}{6} + 0.866T\right) z}{z^2 - 2e^{-0.5T} \cos(0.866T) z + e^{-T}}$$

In terms of sampling interval T = 1, transfer function is,

$$H(z) = \frac{b_0 z^{-1} + b_1 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2} - a_3 z^{-3}}$$

Where $b_0 = -2e^{-0.5T} \cos(0.866T) + e^{-T} + 1.154 e^{-0.5T} \cos\left(\frac{5\pi}{6} + 0.866T\right) = 1.0773$

$$b_1 = e^{-T} + 1.154 e^{-1.5T} \cos\left(\frac{5\pi}{6} + 0.866T\right) = 0.1254$$

$$a_1 = e^{-T} + 2e^{-0.5T} \cos(0.866T) = 1.1538$$

$$a_2 = -e^{-T} - 2e^{-1.5T} \cos(0.866T) = -0.657$$

$$a_3 = e^{-2T} = 0.1353$$

**Question 6**

*Design a linear phase high pass filter using the Hamming window for the following desired frequency response.*

$$H_d(\omega) = \begin{cases} e^{-j3\omega} & \frac{\pi}{6} \le |\omega| \le \pi \\ 0 & |\omega| < \frac{\pi}{6} \end{cases}$$

$\omega(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right)$, *where N is the length of the Hamming window.*

**Sol. :**

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(\omega)e^{j\omega n}\,d\omega$$

$$= \frac{1}{2\pi}\int_{-\pi}^{-\pi/6} e^{-j3\omega}e^{j\omega n}\,d\omega + \frac{1}{2\pi}\int_{\pi/6}^{\pi} e^{-j3\omega}e^{j\omega n}\,d\omega$$

$$= \frac{1}{\pi(n-3)}\left[\sin[\pi(n-3)] - \sin\left[\frac{\pi}{6}(n-3)\right]\right] \quad n \neq 3$$

Also,

$$h_d(3) = \frac{1}{2\pi}\left(\frac{5\pi}{6} + \frac{5\pi}{6}\right) \frac{5}{6}$$

Let

$$N = 7$$

Impulse response of FIR filter is :

$$h(n) = h_d(n)\omega(n)$$

$$= \begin{cases} \left\{\dfrac{1}{\pi(n-3)}\left[\sin[\pi(n-3)] - \sin\left[\dfrac{\pi}{6}(n-3)\right]\right]\right\}\left\{0.54 - 0.46\cos\left(\dfrac{2\pi n}{6}\right)\right\} & n \neq 3 \\ \dfrac{5}{6}\left[0.54 - 0.46\cos\left(\dfrac{2\pi n}{6}\right)\right] & n = 3 \end{cases}$$

| n | $h_d(n)$ | $\omega(n)$ | h(n) |
|---|---|---|---|
| 0 | – 0.1061 | 0.08 | 0.0085 |
| 1 | – 0.1378 | 0.31 | 0.0427 |
| 2 | – 0.1592 | 0.77 | 0.1226 |
| 3 | 0.8333 | 1 | 0.8333 |
| 4 | 0.1592 | 0.77 | 0.1226 |
| 5 | 0.1378 | 0.31 | 0.0427 |
| 6 | 0.1061 | 0.08 | 0.0085 |

## Question 7

*Design a digital lowpass Butterworth filter using bilinear transformation method to meet the following specifications. Take T = 2 sec.*

*Passband ripple ≤ 1.25 dB*

*Passband edge = 200 Hz*

*Stopband attenuation ≥ 15 dB*

*Stopband edge = 400 Hz*

*Sampling frequency = 2 kHz* **[12]**

**Sol. :**

$$\Omega_p = 2\pi\times 200 = 400\,\pi\, rad/\sec$$

$$\Omega_s = 2\pi\times 400 = 800\,\pi\, rad/\sec$$

$$T_s = \frac{1}{f_s} = \frac{1}{2000}\sec$$

$$\omega_p = \Omega_p T_s = 400\pi\times\frac{1}{2000} = 0.2\,\pi \text{ rad}$$

$$\omega_s = \Omega_p T_s = 800\pi\times\frac{1}{2000} = 0.4\,\pi \text{ rad}$$

**Given :** T = 2 sec.

$$\Omega'_P = \frac{2}{T}\tan\left(\frac{\omega_P}{2}\right) = \tan\left(\frac{0.2\pi}{2}\right) = 0.3249$$

$$\Omega'_s = \frac{2}{T}\tan\left(\frac{\omega_s}{2}\right) = \tan\left(\frac{0.4\pi}{2}\right) = 0.7265$$

$$N = \frac{\log\left[(10^{-k_P/10}-1)/(10^{-k_s/10}-1)\right]}{2\log(\Omega'_P/\Omega'_s)}$$

$$= \frac{\log(0.3335/30.6228)}{2\log(0.3249/0.7265)} = 2.8083 \cong 3$$

$$\Omega_c = \frac{\Omega'_P}{(10^{-k_P/10}-1)^{1/2N}} = 1.7688$$

Referring to normalized lowpass butterworth filter tables

$$H_3(s) = \frac{1}{(s^2+s+1)(s+1)}$$

The required prewarped analog filter is obtained by applying lowpass to lowpass transformation.

$$H_a(s) = H_3(s)\Big|_{s \to \frac{s}{2}}$$

$$= \frac{1}{(s^2+s+1)(s+1)}\Bigg|_{s \to \frac{s}{2}}$$

$$= \frac{1}{\left(\frac{s^2}{4}+\frac{s}{2}+1\right)\left(\frac{s}{2}+1\right)}$$

$$= \frac{1}{\left(\frac{s^2}{4}+\frac{s+2}{2}\right)\left(\frac{s+2}{2}\right)}$$

$$= \frac{1}{\left(\frac{2s^2+4s+8}{8}\right)\left(\frac{s+2}{2}\right)}$$

$$= \frac{1}{\left(\frac{s^2+2s+4}{4}\right)\left(\frac{s+2}{2}\right)}$$

$$= \frac{8}{(s^2+2s+4)(s+2)} = \frac{8}{s^3+2s^2+2s^2+4s+8}$$

$$= \frac{8}{s^3+4s^2+8s+8}$$

Applying bilinear transformation to $H_a(s)$

$$H(z) = H_3(s)\Big|_{s \to \frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)}$$

$$\therefore \qquad H(z) = \frac{8}{\left(\frac{1-z^{-1}}{1+z^{-1}}\right)^3 + 4\left(\frac{1-z^{-1}}{1+z^{-1}}\right)^2 + 8\left(\frac{1-z^{-1}}{1+z^{-1}}\right) + 8}$$

# Architectures for Programmable Digital Signal Processing Devices

## Basic Architectural Features

A programmable DSP device should provide instructions similar to a conventional microprocessor. The instruction set of a typical DSP device should include the following,

a. Arithmetic operations such as ADD, SUBTRACT, MULTIPLY etc
b. Logical operations such as AND, OR, NOT, XOR etc
c. Multiply and Accumulate (MAC) operation
d. Signal scaling operation

In addition to the above provisions, the architecture should also include,

a. On chip registers to store immediate results
b. On chip memories to store signal samples (RAM)
c. On chip memories to store filter coefficients (ROM)

## DSP Computational Building Blocks

Each computational block of the DSP should be optimized for functionality and speed and in the meanwhile the design should be sufficiently general so that it can be easily integrated with other blocks to implement overall DSP systems.

### *Multipliers*

The advent of single chip multipliers paved the way for implementing DSP functions on a VLSI chip. Parallel multipliers replaced the traditional shift and add multipliers now days. Parallel multipliers take a single processor cycle to fetch and execute the instruction and to store the result. They are also called as Array multipliers. The key features to be considered for a multiplier are:

a. Accuracy
b. Dynamic range
c. Speed

The number of bits used to represent the operands decides the accuracy and the dynamic range of the multiplier. Whereas speed is decided by the architecture employed. If the multipliers are implemented using hardware, the speed of execution will be very high but the circuit complexity will also increases considerably. Thus there should be a tradeoff between the speed of execution and the circuit complexity. Hence the choice of the architecture normally depends on the application.

### *Parallel Multipliers*

Consider the multiplication of two unsigned numbers A and B. Let A be represented using m bits as (Am-1 Am-2 …….. A1 A0) and B be represented using n bits as (Bn-1 Bn-2 …….. B1 B0). Then the product of these two numbers is given by,

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | $A_3$ | $A_2$ | $A_1$ | $A_0$ |
| | | | | $B_3$ | $B_2$ | $B_1$ | $B_0$ |
| | | | | $A_3B_0$ | $A_2B_0$ | $A_1B_0$ | $A_0B_0$ |
| | | | $A_3B_1$ | $A_2B_1$ | $A_1B_1$ | $A_0B_1$ | |
| | | $A_3B_2$ | $A_2B_2$ | $A_1B_2$ | $A_0B_2$ | | |
| | $A_3B_3$ | $A_2B_3$ | $A_1B_3$ | $A_0B_3$ | | | |
| **P7** | **P6** | **P5** | **P4** | **P3** | **P2** | **P1** | **P0** |

This operation can be implemented paralleling using Braun multiplier whose hardware structure is as shown in the figure 2.1.

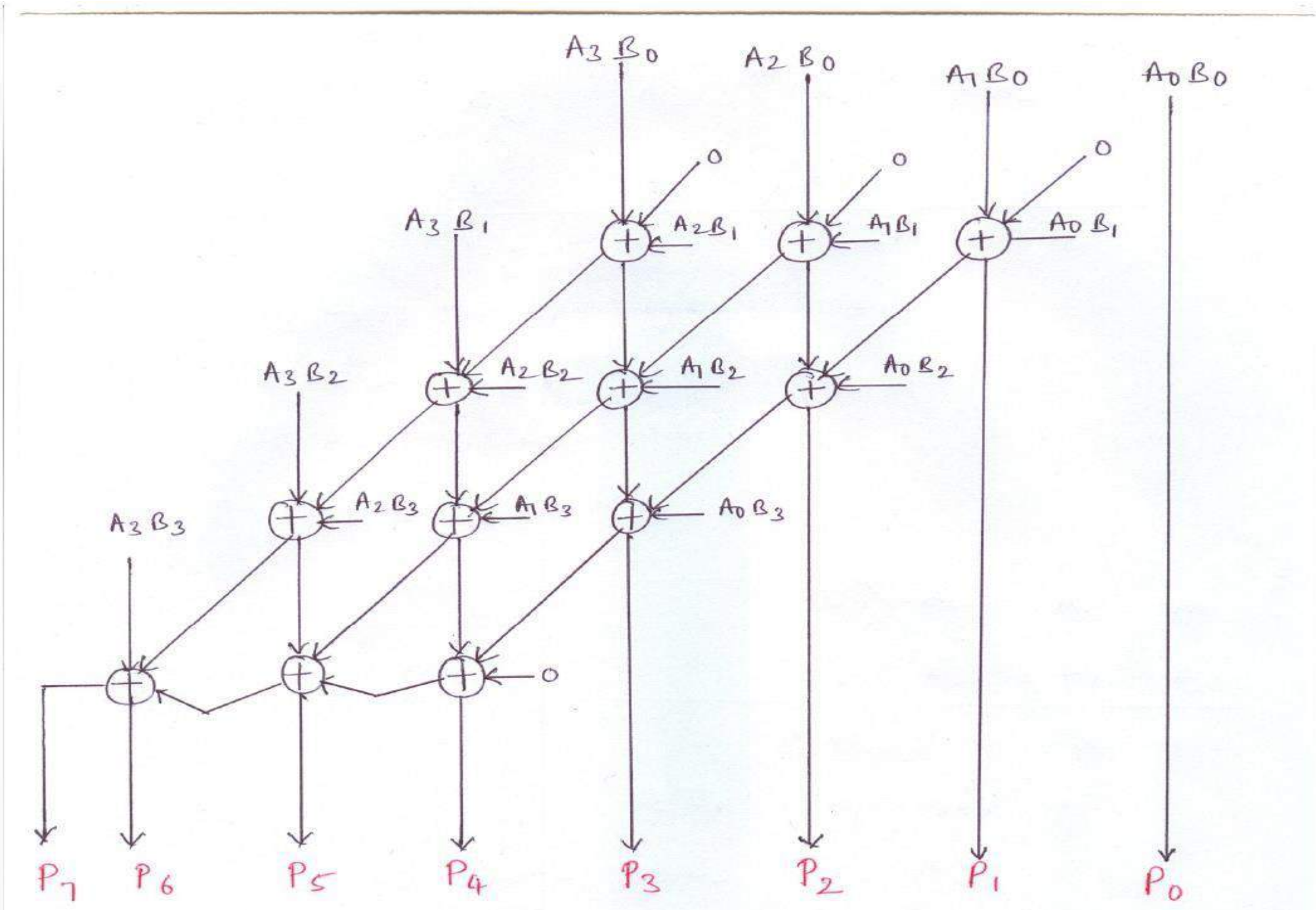



Fig 2.1 Braun Multiplier for a 4X4 Multiplication

### *Multipliers for Signed Numbers*

In the Braun multiplier the sign of the numbers are not considered into account. In order to implement a multiplier for signed numbers, additional hardware is required to modify the Braun multiplier. The modified multiplier is called as Baugh-Wooley multiplier.

Consider two signed numbers A and B,

$$A = -A_{m-1}2^{m-1} + \sum_{i=0}^{m-2} A_i 2^i$$

$$B = -B_{n-1}2^{n-1} + \sum_{j=0}^{n-2} B_j 2^j$$

$$\text{Product } P = P_{m+n-1} \ldots\ldots P_1 P_0$$

$$P = A_{m-1}B_{n-1}2^{m+n-2} + \sum_{i=0}^{m-2}\sum_{j=0}^{n-2} A_i B_j 2^{i+j} - \sum_{i=0}^{m-2} A_i B_{n-1} 2^{n-1+I} - \sum_{j=0}^{n-2} A_{m-1} B_j 2^{m-1+j}$$

### *Speed*

Conventional Shift and Add technique of multiplication requires n cycles to perform the multiplication of two n bit numbers. Whereas in parallel multipliers the time required will be the longest path delay in the combinational circuit used. As DSP applications generally require very high speed, it is desirable to have multipliers operating at the highest possible speed by having parallel implementation.

### *Bus Widths*

Consider the multiplication of two n bit numbers X and Y. The product Z can be at most 2n bits long. In order to perform the whole operation in a single execution cycle, we require two buses of width n bits each to fetch the operands X and Y and a bus of width 2n bits to store the result Z to the memory. Although this performs the operation faster, it is not an efficient way of implementation as it is expensive. Many alternatives for the above method have been proposed. One such method is to use the program bus itself to fetch one of the operands after fetching the instruction, thus requiring only one bus to fetch the operands. And the result Z can be stored back to the memory using the same operand bus. But the problem with this is the result Z is 2n bits long whereas the operand bus is just n bits long. We have two alternatives to solve this problem, a. Use the n bits operand bus and save Z at two successive memory locations. Although it stores the exact value of Z in the memory, it takes two cycles to store the result.

b. Discard the lower n bits of the result Z and store only the higher order n bits into the memory. It is not applicable for the applications where accurate result is required. Another alternative can be used for the applications where speed is not a major concern. In which latches are used for inputs and outputs thus requiring a single bus to fetch the operands and to store the result (Fig 2.2).

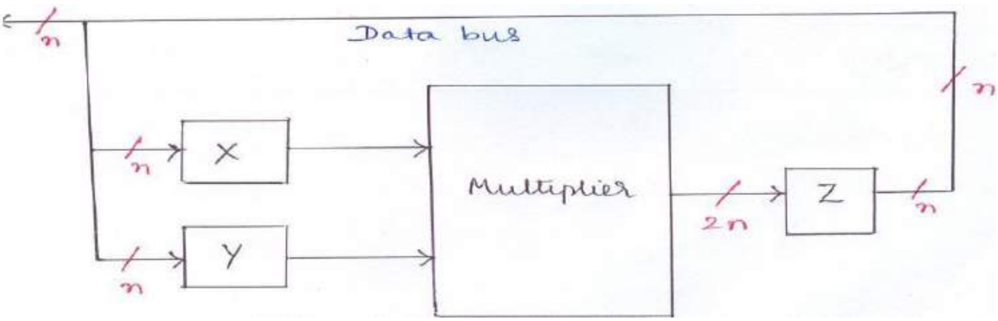



Fig 2.2: A Multiplier with Input and Output Latches

***Shifters***

Shifters are used to either scale down or scale up operands or the results. The following scenarios give the necessity of a shifter
a. While performing the addition of N numbers each of n bits long, the sum can grow up to n+log2 N bits long. If the accumulator is of n bits long, then an overflow error will occur. This can be overcome by using a shifter to scale down the operand by an amount of log2N.
b. Similarly while calculating the product of two n bit numbers, the product can grow up to 2n bits long. Generally the lower n bits get neglected and the sign bit is shifted to save the sign of the product.
c. Finally in case of addition of two floating-point numbers, one of the operands has to be shifted appropriately to make the exponents of two numbers equal.
From the above cases it is clear that, a shifter is required in the architecture of a DSP.

***Barrel Shifters***

In conventional microprocessors, normal shift registers are used for shift operation. As it requires one clock cycle for each shift, it is not desirable for DSP applications, which generally involves more shifts. In other words, for DSP applications as speed is the crucial issue, several shifts are to be accomplished in a single execution cycle. This can be accomplished using a barrel shifter, which connects the input lines representing a word to a group of output lines with the required shifts determined by its control inputs. For an input of length n, log2 n control lines are required. And an dditional control line is required to indicate the direction of the shift.
The block diagram of a typical barrel shifter is as shown in figure 2.3.

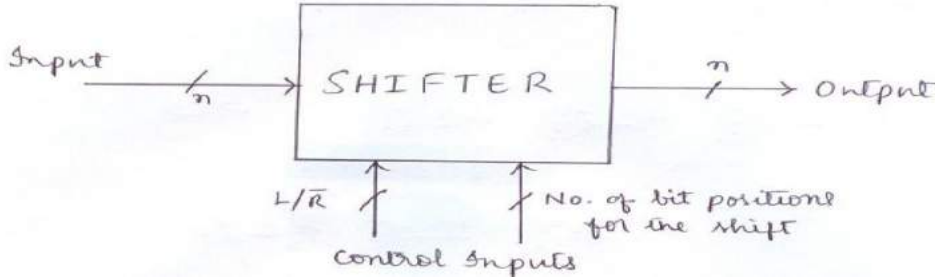
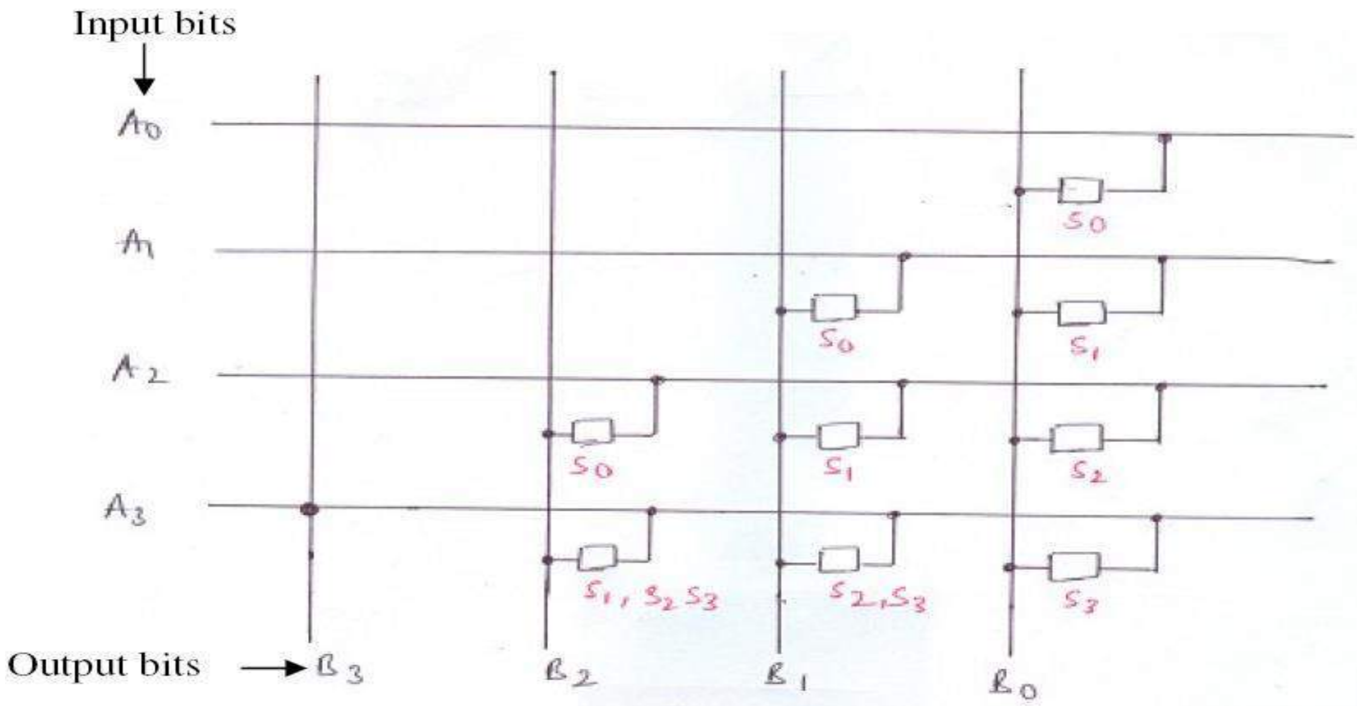



Fig 2.3 A Barrel Shifter




| INPUT | | | | SHIFT (SWITCH) | OUTPUT ($B_3$ $B_2$ $B_1$ $B_0$) | | | |
|---|---|---|---|---|---|---|---|---|
| $A_3$ | $A_2$ | $A_1$ | $A_0$ | 0 ($S_0$) | $A_3$ | $A_2$ | $A_1$ | $A_0$ |
| $A_3$ | $A_2$ | $A_1$ | $A_0$ | 1 ($S_1$) | $A_3$ | $A_3$ | $A_2$ | $A_1$ |
| $A_3$ | $A_2$ | $A_1$ | $A_0$ | 2 ($S_2$) | $A_3$ | $A_3$ | $A_3$ | $A_2$ |
| $A_3$ | $A_2$ | $A_1$ | $A_0$ | 3 ($S_3$) | $A_3$ | $A_3$ | $A_3$ | $A_3$ |

Fig 2.4 Implementation of a 4 bit Shift Right Barrel Shifter

Figure 2.4 depicts the implementation of a 4 bit shift right barrel shifter. Shift to right by 0, 1, 2 or 3 bit positions can be controlled by setting the control inputs appropriately.

### 2.3 Multiply and Accumulate Unit

Most of the DSP applications require the computation of the sum of the products of a series of successive multiplications. In order to implement such functions a special unit called a multiply and Accumulate (MAC) unit is required. A MAC consists of a multiplier and a special register called Accumulator. MACs are used to implement the functions of the type A+BC. A typical MAC unit is as shown in the figure 2.5.

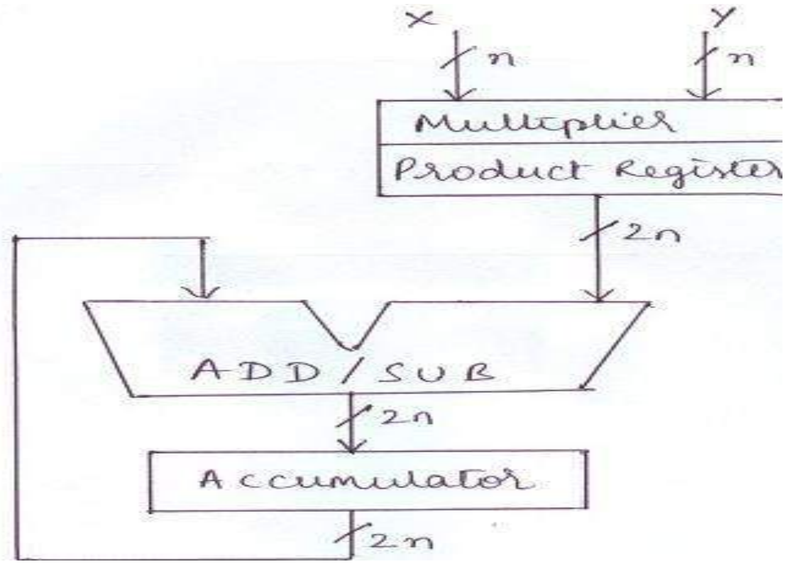



Fig 2.5 A MAC Unit

Although addition and multiplication are two different operations, they can be performed in parallel. By the time the multiplier is computing the product, accumulator can accumulate the product of the previous multiplications. Thus if N products are to be accumulated, N-1 multiplications can overlap with N-1 additions. During the very first multiplication, accumulator will be idle and during the last accumulation, multiplier will be idle. Thus N+1 clock cycles are required to compute the sum of N products.

### *2.3.1 Overflow and Underflow*

While designing a MAC unit, attention has to be paid to the word sizes encountered at the input of the multiplier and the sizes of the add/subtract unit and the accumulator, as there is a possibility of overflow and underflows. Overflow/underflow can be avoided by using any of the following methods viz

a. Using shifters at the input and the output of the MAC

b. Providing guard bits in the accumulator

c. Using saturation logic

**Shifters**

Shifters can be provided at the input of the MAC to normalize the data and at the output to de normalize the same.

**Guard bits**

As the normalization process does not yield accurate result, it is not desirable for some applications. In such cases we have another alternative by providing additional bits called guard bits in the accumulator so that there will not be any overflow error. Here the add/subtract unit also has to be modified appropriately to manage the additional bits of the accumulator.

**Saturation Logic**

Overflow/ underflow will occur if the result goes beyond the most positive number or below the least negative number the accumulator can handle. Thus the overflow/underflow error can be resolved by loading the accumulator with the most positive number which it can handle at the time of overflow and the least negative number that it can handle at the time of underflow. This method is called as saturation logic. A schematic diagram of saturation logic is as shown in figure 2.7. In saturation logic, as soon as an overflow or underflow condition is satisfied the accumulator will be loaded with the most positive or least negative number overriding the result computed by the MAC unit.

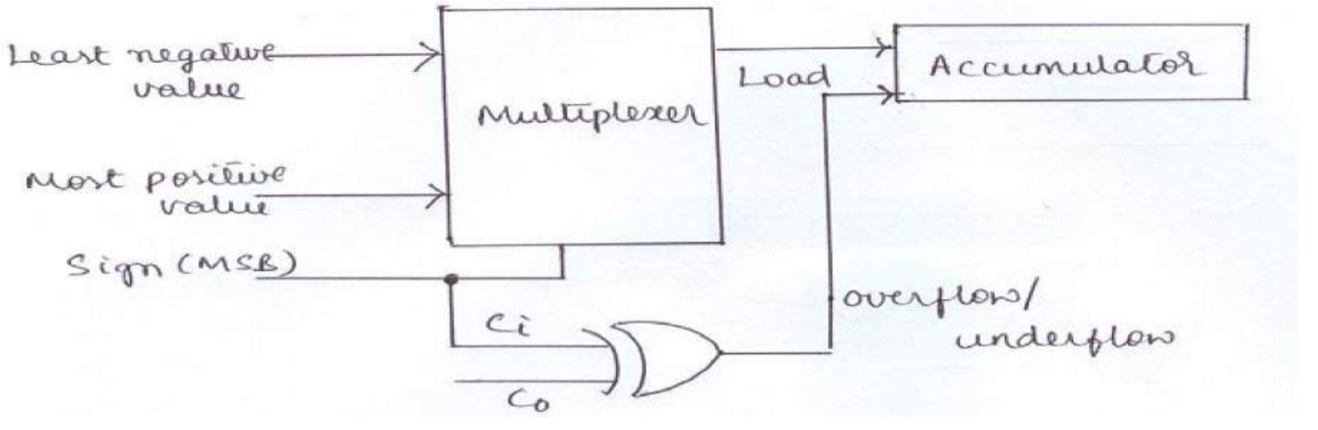



Fig 2.7: Schematic Diagram of the Saturation Logic

**Arithmetic and Logic Unit**

A typical DSP device should be capable of handling arithmetic instructions like ADD, SUB, INC, DEC etc and logical operations like AND, OR , NOT, XOR etc. The block diagram of a typical ALU for a DSP is as shown in the figure 2.8.

It consists of status flag register, register file and multiplexers.

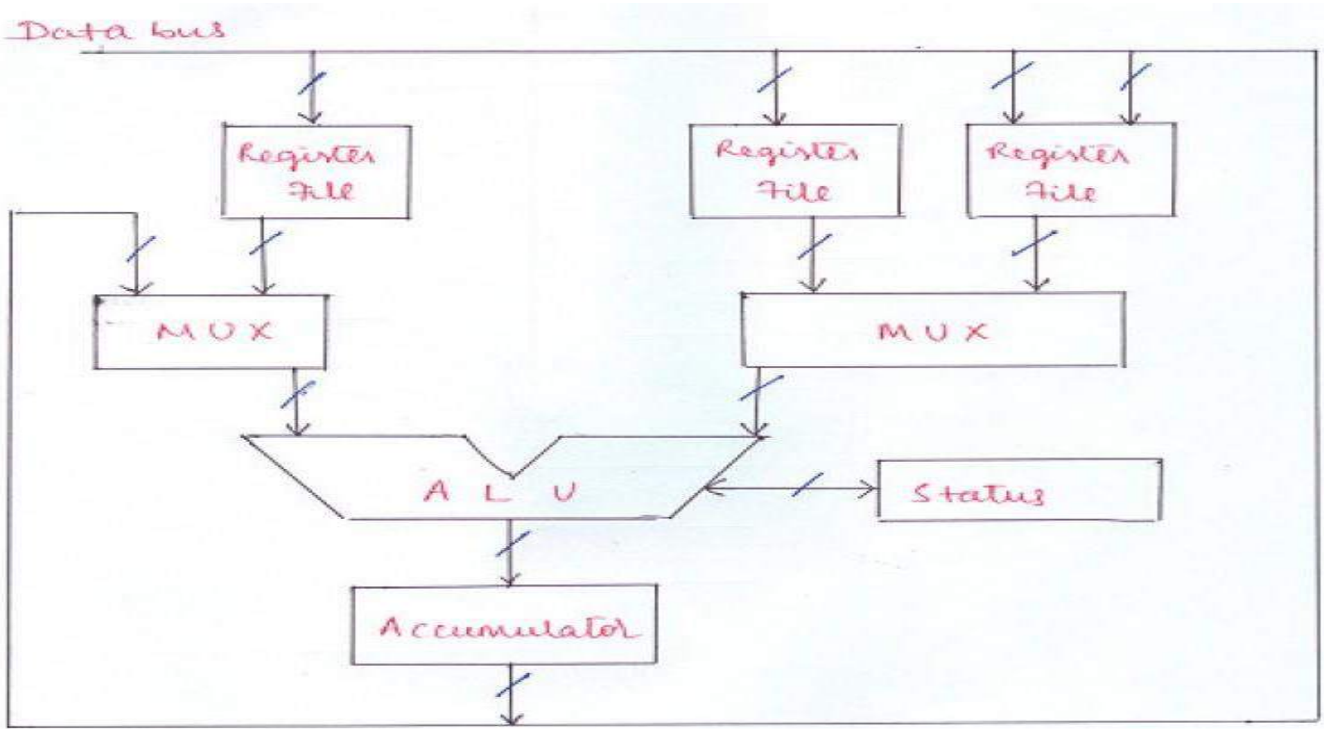



Fig 2.8 Arithmetic Logic Unit of a DSP

**Status Flags**

ALU includes circuitry to generate status flags after arithmetic and logic operations. These flags include sign, zero, carry and overflow.

**Overflow Management**

Depending on the status of overflow and sign flags, the saturation logic can be used to limit the accumulator content.

**Register File**

Instead of moving data in and out of the memory during the operation, for better speed, a large set of general purpose registers are provided to store the intermediate results.

**Bus Architecture and Memory**

Conventional microprocessors use Von Neumann architecture for memory management wherein the same memory is used to store both the program and data (Fig 2.9). Although this architecture is simple, it takes more number of processor cycles for the execution of a single instruction as the same bus is used for both data and program.

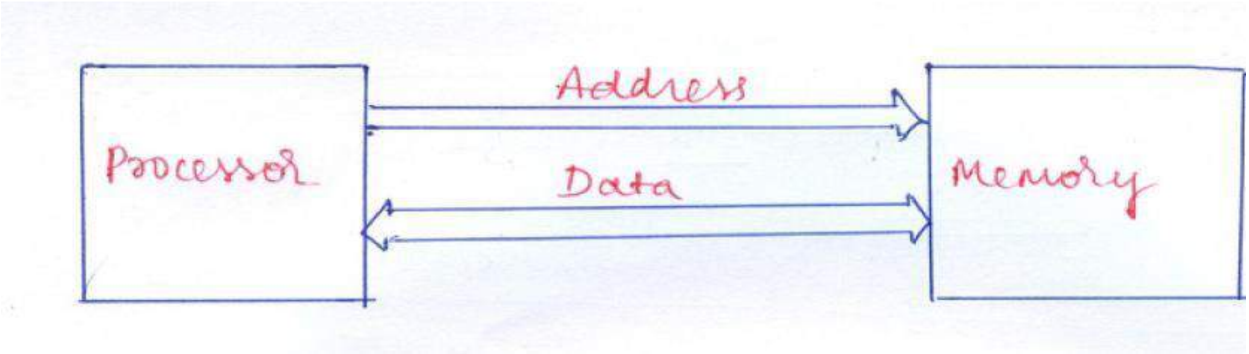



Fig 2.9 Von Neumann Architecture

In order to increase the speed of operation, separate memories were used to store program and data and a separate set of data and address buses have been given to both memories, the architecture called as Harvard Architecture. It is as shown in figure 2.10.

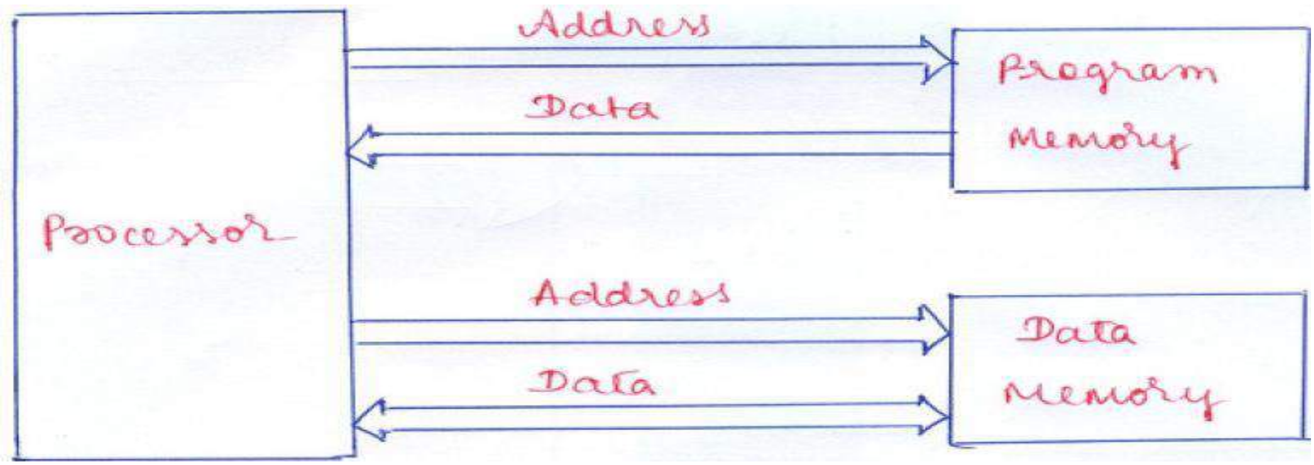



Fig 2.10 Harvard Architecture

Although the usage of separate memories for data and the instruction speeds up the processing, it will not completely solve the problem. As many of the DSP instructions require more than one operand, use of a single data memory leads to the fetch the operands one after the other, thus increasing the delay of processing. This problem can be overcome by using two separate data memories for storing operands separately, thus in a single clock cycle both the operands can be fetched together (Figure 2.11).

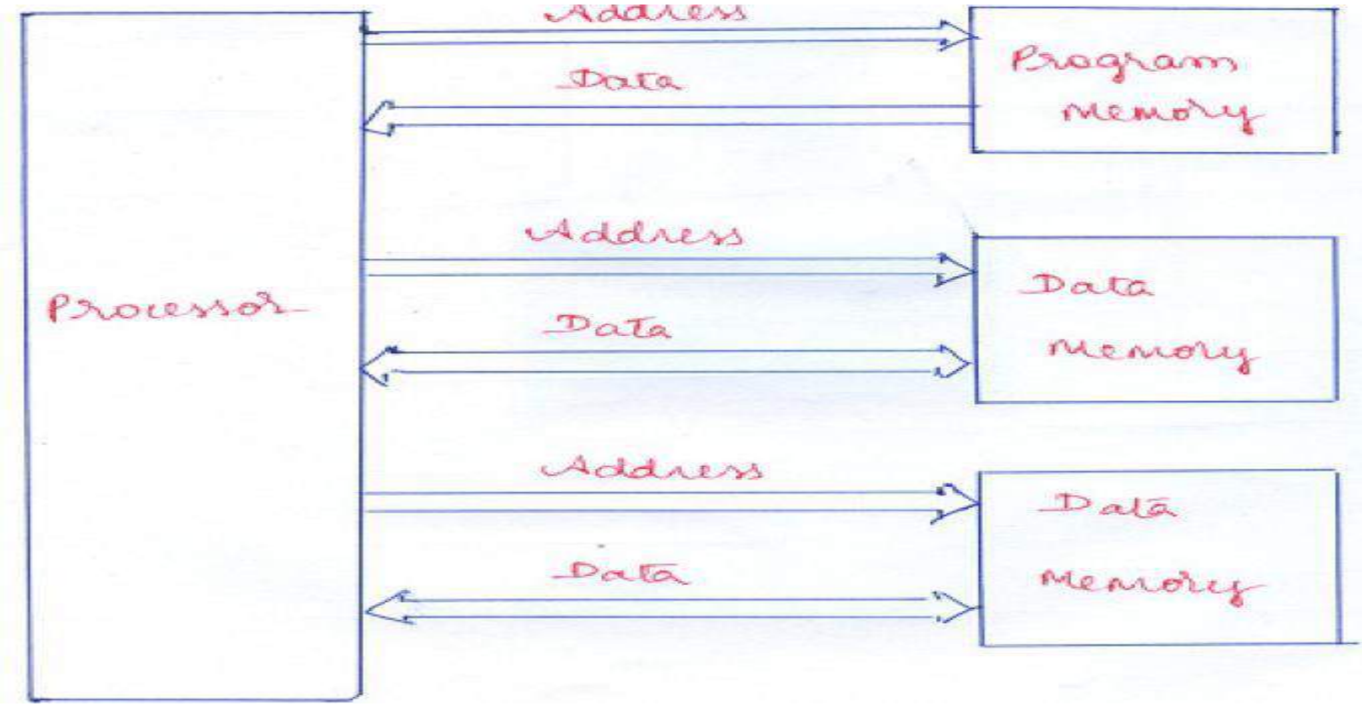



Fig 2.11 Harvard Architecture with Dual Data Memory

Although the above architecture improves the speed of operation, it requires more hardware and interconnections, thus increasing the cost and complexity of the system. Therefore there should be a trade off between the cost and speed while selecting memory architecture for a DSP.

***On-chip Memories***

In order to have a faster execution of the DSP functions, it is desirable to have some memory located on chip. As dedicated buses are used to access the memory, on chip memories are faster. Speed and size are the two key parameters to be considered with respect to the on-chip memories.

**Speed**

On-chip memories should match the speeds of the ALU operations in order to maintain the single cycle instruction execution of the DSP.

**Size**

In a given area of the DSP chip, it is desirable to implement as many DSP functions as possible. Thus the area occupied by the on-chip memory should be minimum so that there will be a scope for implementing more number of DSP functions on- chip.

***Organization of On-chip Memories***

Ideally whole memory required for the implementation of any DSP algorithm has to reside on-chip so that the whole processing can be completed in a single execution cycle. Although it looks as a better solution, it consumes more space on chip, reducing the scope for implementing any functional block on-chip, which in turn reduces the speed of execution. Hence some other alternatives have to be thought of. The following are some other ways in which the on-chip memory can be organized.

a. As many DSP algorithms require instructions to be executed repeatedly, the instruction can be stored in the external memory, once it is fetched can reside in the instruction cache.

b. The access times for memories on-chip should be sufficiently small so that it can be accessed more than once in every execution cycle.

c. On-chip memories can be configured dynamically so that they can serve different purpose at different times.

**Data Addressing Capabilities**

Data accessing capability of a programmable DSP device is configured by means of its addressing modes. The summary of the addressing modes used in DSP is as shown in the table below.

Table 2.1 DSP Addressing Modes

| Addressing Mode | Operand | Sample Format | Operation |
|---|---|---|---|
| Immediate | Immediate Value | ADD #imm | #imm +A → A |
| Register | Register Contents | ADD reg | reg +A → A |
| Direct | Memory Address Register | ADD mem | mem+A → A |
| Indirect | Memory contents with address in the register | ADD *addreg | *addreg +A → A |

***Immediate Addressing Mode***

In this addressing mode, data is included in the instruction itself.

***Register Addressing Mode***

In this mode, one of the registers will be holding the data and the register has to be specified in the instruction.

***Direct Addressing Mode***

In this addressing mode, instruction holds the memory location of the operand.

***Indirect Addressing Mode***

In this addressing mode, the operand is accessed using a pointer. A pointer is generally a register, which holds the address of the location where the operands resides. Indirect addressing mode can be extended to inculcate automatic increment or decrement capabilities, which has lead to the following addressing modes.

Table 2.2 Indirect Addressing Modes

| Addressing Mode | Sample Format | Operation |
|---|---|---|
| Post Increment | ADD *addreg+ | A $\longrightarrow$ A + *addreg<br>addreg $\longrightarrow$ addreg+1 |
| Post Decrement | ADD *addreg- | A $\longrightarrow$ A + *addreg<br>addreg $\longrightarrow$ addreg-1 |
| Pre Increment | ADD +*addreg | addreg $\longrightarrow$ addreg+1<br>A $\longrightarrow$ A + *addreg |
| Pre Decrement | ADD -*addreg | addreg $\longrightarrow$ addreg-1<br>A $\longrightarrow$ A + *addreg |
| Post_Add_Offset | ADD *addreg, offsetreg+ | A $\longrightarrow$ A + *addreg<br>addreg $\longrightarrow$ addreg+offsetreg |
| Post_Sub_Offset | ADD *addreg, offsetreg- | A $\longrightarrow$ A + *addreg<br>addreg $\longrightarrow$ addreg-offsetreg |
| Pre_Add_Offset | ADD offsetreg+,*addreg | addreg $\longrightarrow$ addreg+offsetreg<br>A $\longrightarrow$ A + *addreg |
| Pre_Sub_Offset | ADD offsetreg-,*addreg | addreg $\longrightarrow$ addreg-offsetreg<br>A $\longrightarrow$ A + *addreg |

**Special Addressing Modes**

For the implementation of some real time applications in DSP, normal addressing modes will not completely serve the purpose. Thus some special addressing modes are required for such applications.

***Circular Addressing Mode***

While processing the data samples coming continuously in a sequential manner, circular buffers are used. In a circular buffer the data samples are stored sequentially from the initial location till the buffer gets filled up. Once the buffer gets filled up, the next data samples will get stored once again from the initial location. This process can go forever as long as the data samples are processed in a rate faster than the incoming data rate.

Circular Addressing mode requires three registers viz

a. Pointer register to hold the current location (PNTR)

b. Start Address Register to hold the starting address of the buffer (SAR)

c. End Address Register to hold the ending address of the buffer (EAR)

There are four special cases in this addressing mode. They are

a. SAR < EAR & updated PNTR > EAR
b. SAR < EAR & updated PNTR < SAR
c. SAR >EAR & updated PNTR > SAR
d. SAR > EAR & updated PNTR < EAR
The buffer length in the first two case will be (EAR-SAR+1) whereas for the next tow cases (SAR-EAR+1)
The pointer updating algorithm for the circular addressing mode is as shown below.

```
; Pointer Updating Algorithm

Updated PNTR ←—— PNTR ± increment

If SAR < EAR
        And if Updated PNTR > EAR then
                New PNTR ←—— Updated PNTR – Buffer size
        And if Updated PNTR < SAR then
                New PNTR         Updated PNTR + Buffer size

If SAR > EAR
        And if Updated PNTR > SAR then
                New PNTR ←—— Updated PNTR – Buffer size
        And if Updated PNTR < EAR then
                New PNTR ←—— Updated PNTR + Buffer size

Else
        New PNTR ←—— Updated PNTR
```

Four cases explained earlier are as shown in the figure 2.12.

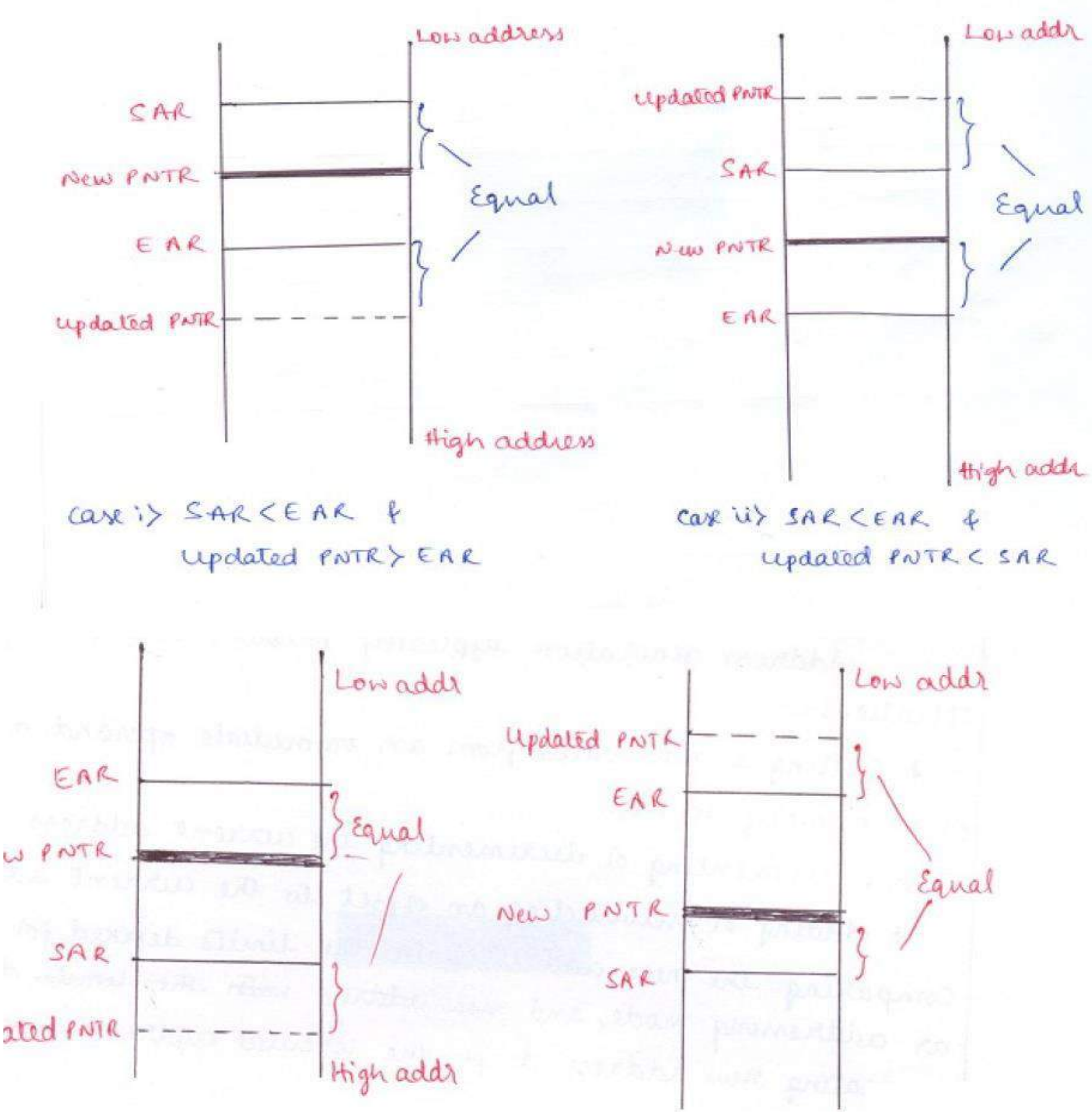



Fig 2.12 Special Cases in Circular Addressing Mode

***Bit Reversed Addressing Mode***

To implement FFT algorithms we need to access the data in a bit reversed manner. Hence a special addressing mode called bit reversed addressing mode is used to calculate the index of the next data to be fetched. It works as follows. Start with index 0. The present index can be calculated by adding half the FFT length to the previous index in a bit reversed manner, carry being propagated from MSB to LSB.

**Current index= Previous index+ B (1/2(FFT Size))**

**Address Generation Unit**

The main job of the Address Generation Unit is to generate the address of the operands required to carry out the operation. They have to work fast in order to satisfy the timing constraints. As the address generation unit has to perform some mathematical operations in order to calculate the operand address, it is provided with a separate ALU.

Address generation typically involves one of the following operations.

a. Getting value from immediate operand, register or a memory location

b. Incrementing/ decrementing the current address

c. Adding/subtracting the offset from the current address

d. Adding/subtracting the offset from the current address and generating new address according to circular addressing mode

e. Generating new address using bit reversed addressing mode

The block diagram of a typical address generation unit is as shown in figure 2.13.

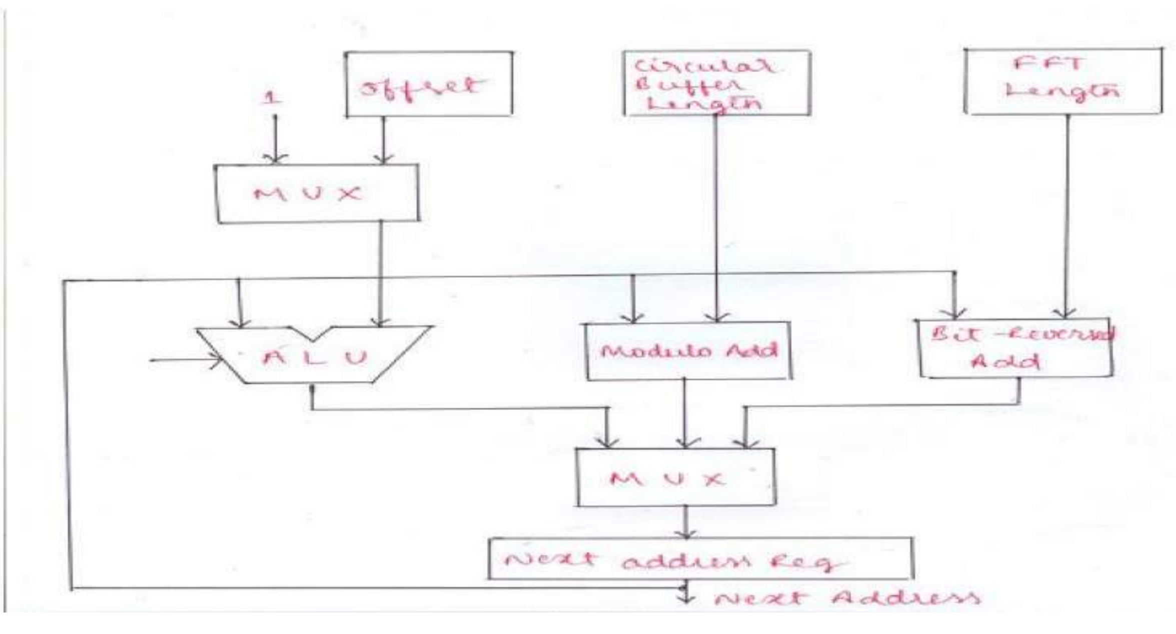



**Fig 2.13 Address generation unit**

### Programmability and program Execution

A programmable DSP device should provide the programming capability involving branching, looping and subroutines. The implementation of repeat capability should be hardware based so that it can be programmed with minimal or zero overhead. A dedicated register can be used as a counter. In a normal subroutine call, return address has to be stored in a stack thus requiring memory access for storing and retrieving the return address, which in turn reduces the speed of operation. Hence a LIFO memory can be directly interfaced with the program counter.

### *Program Control*

Like microprocessors, DSP also requires a control unit to provide necessary control and timing signals for the proper execution of the instructions. In microprocessors, the controlling is micro coded based where each instruction is divided into microinstructions stored in micro memory. As this mechanism is slower, it is not applicable for DSP applications. Hence in DSP the controlling is hardwired base where the Control unit is designed as a single, comprehensive, hardware unit. Although it is more complex it is faster.

# Review Questions

**Question 1**: Investigate the basic features that should be provided in the DSP architecture to be used to implement the following $N^{th}$ order FIR filter.

**Solution:-**

**$y(n) = \sum h(i)\, x(n-i)$ n=0,1,2…**

In order to implement the above operation in a DSP, the architecture requires the following features

i. A RAM to store the signal samples x (n)
ii. A ROM to store the filter coefficients h (n)
iii. An MAC unit to perform Multiply and Accumulate operation
iv. An accumulator to store the result immediately
v. A signal pointer to point the signal sample in the memory
vi. A coefficient pointer to point the filter coefficient in the memory
vii. A counter to keep track of the count
viii. A shifter to shift the input samples appropriately

1). It is required to find the sum of 64, 16 bit numbers. How many bits should the accumulator have so that the sum can be computed without the occurrence of overflow error or loss of accuracy?

The sum of 64, 16 bit numbers can grow up to (16+ log2 64 )=22 bits long. Hence the accumulator should be 22 bits long in order to avoid overflow error from occurring.

1. In the previous problem, it is decided to have an accumulator with only 16 bits but shift the numbers before the addition to prevent overflow, by how many bits should each number be shifted?

As the length of the accumulator is fixed, the operands have to be shifted by an amount of log2 64 = 6 bits prior to addition operation, in order to avoid the condition of overflow.

2. If all the numbers in the previous problem are fixed point integers, what is the actual sum of the numbers?

The actual sum can be obtained by shifting the result by 6 bits towards left side after the sum being computed. Therefore

Actual Sum= Accumulator content X $2^{6}$

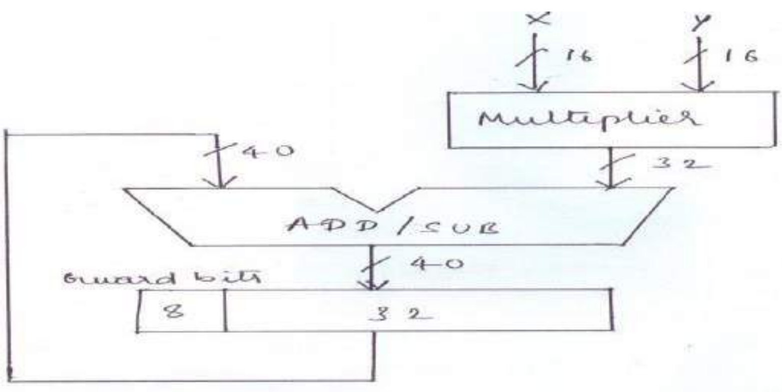
3. If a sum of 256 products is to be computed using a pipelined MAC unit, and if the MAC execution time of the unit is 100nsec, what will be the total time required to complete the operation?

As N=256 in this case, MAC unit requires N+1=257execution cycles. As the single MAC execution time is 100nsec, the total time required will be, (257*100nsec)=25.7usec

4. Consider a MAC unit whose inputs are 16 bit numbers. If 256 products are to be summed up in this MAC, how many guard bits should be provided for the accumulator to prevent overflow condition from occurring?

As it is required to calculate the sum of 256, 16 bit numbers, the sum can be as long as (16+ log2 256)=24 bits. Hence the accumulator should be capable of handling these 22 bits. Thus the guard bits required will be (24-16)= 8 bits.

The block diagram of the modified MAC after considering the guard or extention bits is as shown in the figure




**Question 2**: What are the memory addresses of the operands in each of the following cases of indirect addressing modes? In each case, what will be the content of the *addreg* after the memory access? Assume that the initial contents of the *addreg* and the *offsetreg* are 0200h and 0010h, respectively.

**a. ADD **addreg***

**b.ADD +**addreg***

***c.* ADD *offsetreg+,*addreg***

***d.* ADD **addreg,offsetreg-***

| Instruction | Addressing Mode | Operand Address | addreg Content after Access |
|---|---|---|---|
| ADD **addreg-* | Post Decrement | 0200h | 0200-01=01FFh |
| ADD +**addreg* | Pre Increment | 0200+01=0201h | 0201h |
| ADD *offsetreg+,*addreg* | Pre_Add_Offset | 0200+0010=0210h | 0210h |
| ADD **addreg,offsetreg-* | Post_Sub_Offset | 0200h | 0200-0010=01F0h |

**Question 3**: A DSP has a circular buffer with the start and the end addresses as 0200h and 020Fh respectively. What would be the new values of the address pointer of the buffer if, in the course of address computation, it gets updated to

a. 0212h
b. 01FCh

Buffer Length= (EAR-SAR+1) = 020F-0200+1=10h

a. New Address Pointer= Updated Pointer-buffer length = 0212-10=0202h
b. New Address Pointer= Updated Pointer+ buffer length = 01FC+10=020Ch

**Question 4**: Repeat the previous problem for SAR= 0210h and EAR=0201h Buffer Length= (SAR-EAR+1)= 0210-0201+1=10h

c. New Address Pointer= Updated Pointer- buffer length = 0212-10=0202h
d. New Address Pointer= Updated Pointer+ buffer length = 01FC+10=020Ch

**Question 5**: Compute the indices for an 8-point FFT using Bit reversed Addressing Mode Start with index 0. Therefore the first index would be (000)

Next index can be calculated by adding half the FFT length, in this case it is (100) to the previous index. i.e. Present Index= (000)+B (100)= (100)

Similarly the next index can be calculated as

Present Index= (100)+B (100)= (010)

The process continues till all the indices are calculated. The following table summarizes the calculation.

| Index in Binary | BCD value | Bit reversed index | BCD value |
|---|---|---|---|
| 000 | 0 | 000 | 0 |
| 001 | 1 | 100 | 4 |
| 010 | 2 | 010 | 2 |
| 011 | 3 | 110 | 6 |
| 100 | 4 | 001 | 1 |
| 101 | 5 | 101 | 5 |
| 110 | 6 | 011 | 3 |
| 111 | 7 | 111 | 7 |

# UNIT IV:Programmable Digital Signal Processors

**Introduction:**

Leading manufacturers of integrated circuits such as Texas Instruments (TI), Analog devices & Motorola manufacture the digital signal processor (DSP) chips. These manufacturers have developed a range of DSP chips with varied complexity.

The TMS320 family consists of two types of single chips DSPs: 16-bit fixed point &32-bit floating-point. These DSPs possess the operational flexibility of high-speed controllers and the numerical capability of array processors

**Commercial Digital Signal-Processing Devices:**

There are several families of commercial DSP devices. Right from the early eighties, when these devices began to appear in the market, they have been used in numerous applications, such as communication, control, computers, Instrumentation, and consumer electronics. The architectural features and the processing power of these devices have been constantly upgraded based on the advances in technology and the application needs. However, their basic versions, most of them have Harvard architecture, a single-cycle hardware multiplier, an address generation unit with dedicated address registers, special addressing modes, on-chip peripherals interfaces. Of the various families of programmable DSP devices that are commercially available, the three most popular ones are those from Texas Instruments, Motorola, and Analog Devices. Texas Instruments was one of the first to come out with a commercial programmable DSP with the introduction of its TMS32010 in 1982.

**Summary of the Architectural Features of three fixed-Points DSPs**

| Architectural Feature | TMS320C25 | DSP 56000 | ADSP2100 |
|---|---|---|---|
| Data representation format | 16-bit fixed | 24-bit fixed point | 16-bit fixed point |
| Hardware multiplier | 16 x 16 | 24 x 24 | 16 x 16 |
| ALU | 32 bits | 56 bits | 40 bits |
| Internal buses | 16-bit program bus | 24-bit program bus | 24-bit program bus |
| | 16-bit data bus | 2 x 24-bit data buses | 16-bit data bus |
| | | 24-bit global | 16-bit result |

| | | | |
|---|---|---|---|
| | | databus | bus |
| External buses | 16-bit program/data bus | 24-bit program/data bus | 24-bit program bus<br>16-bit data bus |
| On-chip Memory | 544 words RAM<br>4K words ROM | 512 words PROM<br>2 x 256 words data RAM<br>2 x 256 words data ROM | - |
| Off-chip memory | 64 K words program<br>64k words data | 64K words program<br>2 x 64K words data | 16K words program<br>16K words data |
| Cache memory | - | - | 16 words program |
| Instruction cycle time | 100 nsec | 97.5 nsec. | 125 nsecc. |
| Special addressing modes | Bit reversed | Modulo<br>Bit reversed | Modulo<br>Bit reversed |
| Data address generators | 1 | 2 | 2 |
| Interfacing features | Synchronous serial I/O<br>DMA | Synchronous and Asynchronous serial I/O DMA | DMA |

## The architecture of TMS320C54xx digital signal processors:

TMS320C54xx processors retain in the basic Harvard architecture of their predecessor, TMS320C25, but have several additional features, which improve their performance over it. Figure 4.1 shows a functional block diagram of TMS320C54xx processors. They have one program and three data memory spaces with separate buses, which provide simultaneous accesses to program instruction and two data operands and enables writing of result at the same time. Part of the memory is implemented on-chip and consists of combinations of ROM, dual-access RAM, and single-access RAM. Transfers between the memory spaces are also possible.

The central processing unit (CPU) of TMS320C54xx processors consists of a 40- bit arithmetic logic unit (ALU), two 40-bit accumulators, a barrel shifter, a 17x17 multiplier, a 40-bit adder, data address generation logic (DAGEN) with its own arithmetic unit, and program address generation logic (PAGEN). These major functional units are supported by a number of registers and logic in the architecture. A powerful instruction set with a hardware-supported, single-instruction repeat and block repeat operations, block memory move instructions, instructions that pack two or three simultaneous reads, and arithmetic instructions with parallel store and load make these devices very efficient for running high-speed DSP algorithms.

Several peripherals, such as a clock generator, a hardware timer, a wait state generator, parallel I/O ports, and serial I/O ports, are also provided on-chip. These peripherals make it convenient to interface the signal processors to the outside world. In these following sections, we examine in detail the various architectural features of the TMS320C54xx family of processors.

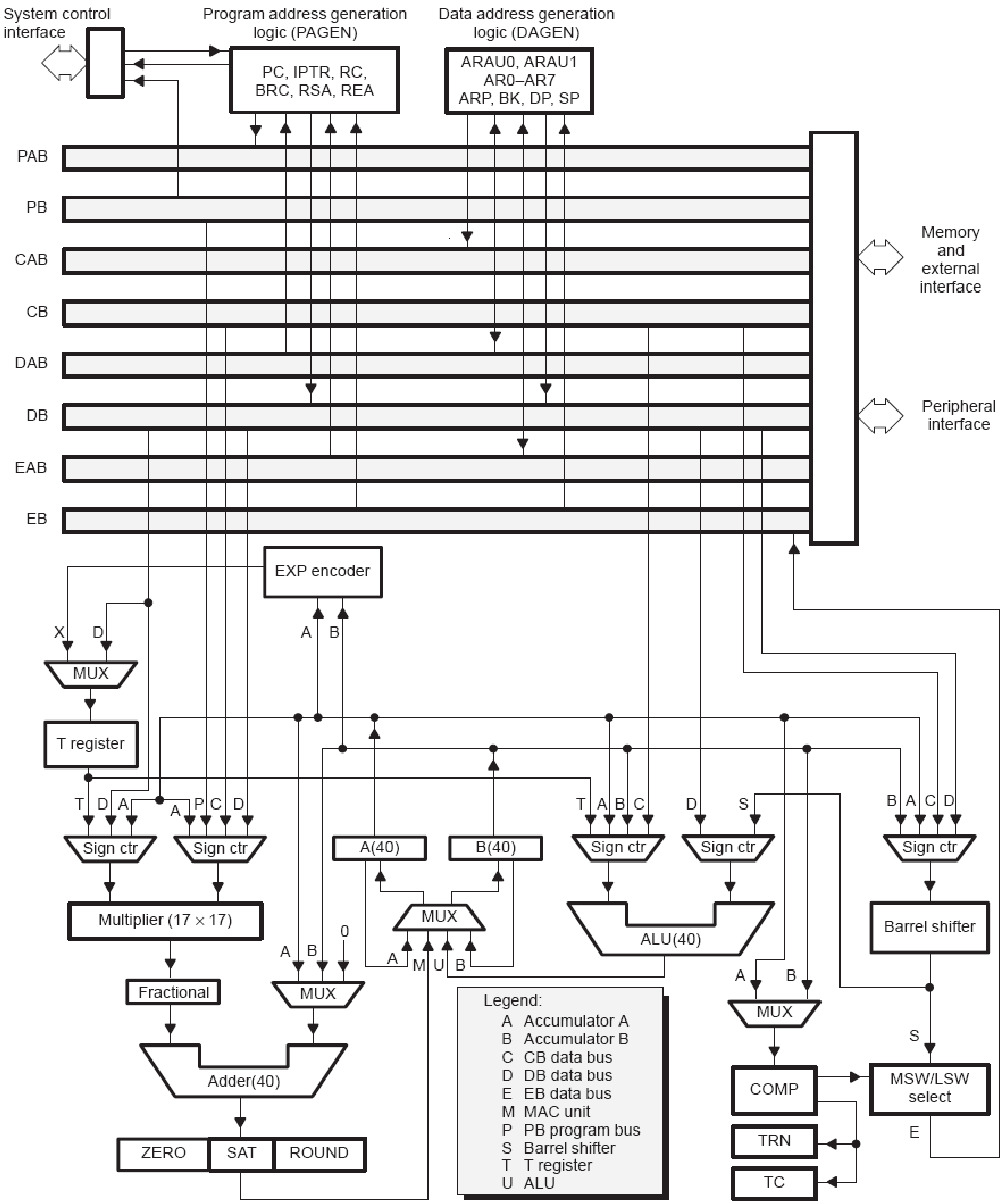



**Figure 4.1**.Functional architecture for TMS320C54xx processors.

**Bus Structure:**

The performance of a processor gets enhanced with the provision of multiple buses to provide simultaneous access to various parts of memory or peripherals. The 54xx architecture is built around four pairs of 16-bit buses with each pair consisting of an address bus and a data bus. As shown in Figure 4.1, these are The program bus pair (**PAB, PB**); which carries the instruction code from the program memory. Three data bus pairs (**CAB, CB; DAB, DB**; and **EAB, EB**); which interconnected the various units within the CPU. In Addition the pair CAB, CB and DAB, DB are used to read from the data memory, while The pair **EAB, EB**; carries the data to be written to the memory. The ‘54xx can generate up to two data-memory addresses per cycle using the two auxiliary register arithmetic unit (ARAU0 and ARAU1) in the DAGEN block. This enables accessing two operands simultaneously.

**Central Processing Unit (CPU):**

The ‘54xx CPU is common to all the ‘54xx devices. The ’54xx CPU contains a 40-bit arithmetic logic unit (**ALU**); two 40-bit accumulators (**A** and **B**); a barrel shifter; a 17 x 17-bit multiplier; a 40-bit adder; a compare, select and store unit (**CSSU**); an exponent encoder(**EXP**); a data address generation unit (**DAGEN**); and a program address generation unit (**PAGEN**).

The ALU performs 2’s complement arithmetic operations and bit-level Boolean operations on 16, 32, and 40-bit words. It can also function as two separate 16-bit ALUs and perform two 16-bit operations simultaneously. Figure 3.2 show the functional diagram of the ALU of the TMS320C54xx family of devices.

**Accumulators A and B** store the output from the ALU or the multiplier/adder block and provide a second input to the ALU. Each accumulators is divided into three parts: guards bits (bits 39-32), high-order word (bits-31-16), and low-order word (bits 15- 0), which can be stored and retrieved individually. Each accumulator is memory-mapped and partitioned. It can be configured as the destination registers. The guard bits are used as a head margin for computations.

| | | |
|---|---|---|
| AG(39-32) | AH(31-16) | AL(15-0) |

| | | |
|---|---|---|
| BG(39-32) | BH(31-16) | BL(15-0) |

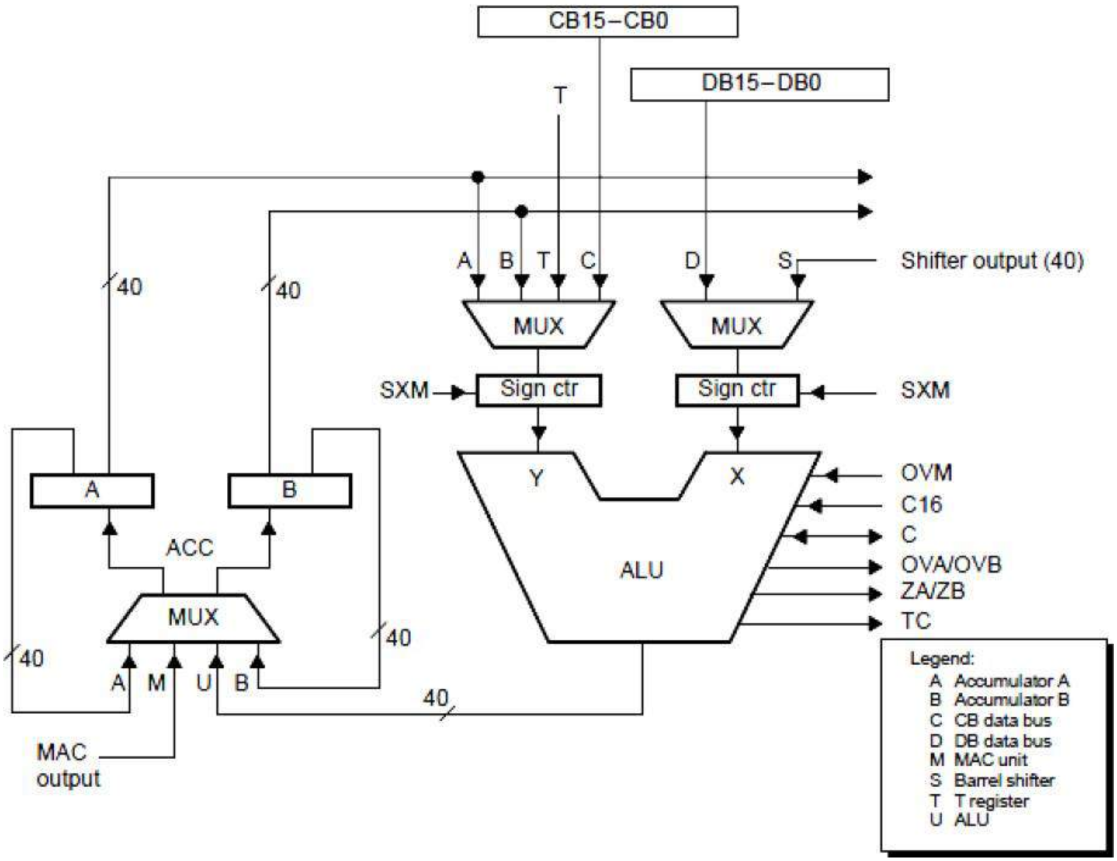



**Figure 4.2**.Functional diagram of the central processing unit of the TMS320C54xx processors.

**Barrel shifter:** provides the capability to scale the data during an operand read or write. No overhead is required to implement the shift needed for the scaling operations. The'54xx barrel shifter can produce a left shift of 0 to 31 bits or a right shift of 0 to 16 bits on the input data. The shift count field of status registers ST1, or in the temporary register T. Figure 4.3 shows the functional diagram of the barrel shifter of TMS320C54xx processors. The barrel shifter and the exponent encoder normalize the values in an accumulator in a single cycle. The LSBs of the output are filled with0s, and the MSBs can be either zero filled or sign extended, depending on the state of the sign-extension mode bit in the status register ST1. An additional shift capability enables the processor to perform numerical scaling, bit extraction, extended arithmetic, and overflow prevention operations.

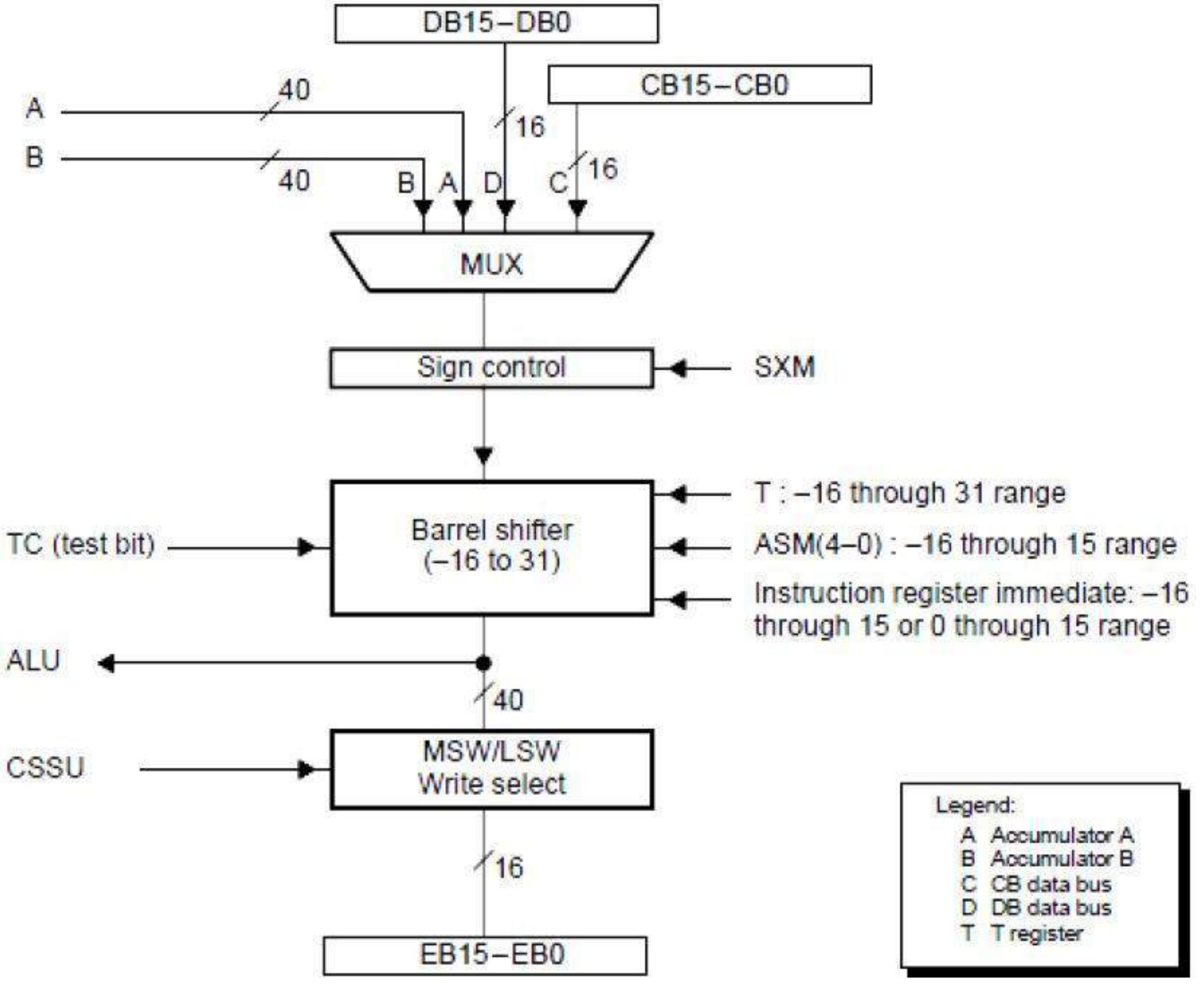



**Figure 4.3**.Functional diagram of the barrel shifter

**Multiplier/adder unit:** The kernel of the DSP device architecture is multiplier/adder unit. The multiplier/adder unit of TMS320C54xx devices performs 17 x 17 2's complement multiplication with a 40-bit addition effectively in a single instruction cycle.

In addition to the multiplier and adder, the unit consists of control logic for integer and fractional computations and a 16-bit temporary storage register, T. Figure 4.4 show the functional diagram of the multiplier/adder unit of TMS320C54xx processors. The compare, select, and store unit (CSSU) is a hardware unit specifically incorporated to accelerate the add/compare/select operation. This operation is essential to implement the *Viterbi* algorithm used in many signal-processing applications. The exponent encoder unit supports the EXP instructions, which stores in the T register the number of leading redundant bits of the accumulator content. This information is useful while shifting the accumulator content for the purpose of scaling.

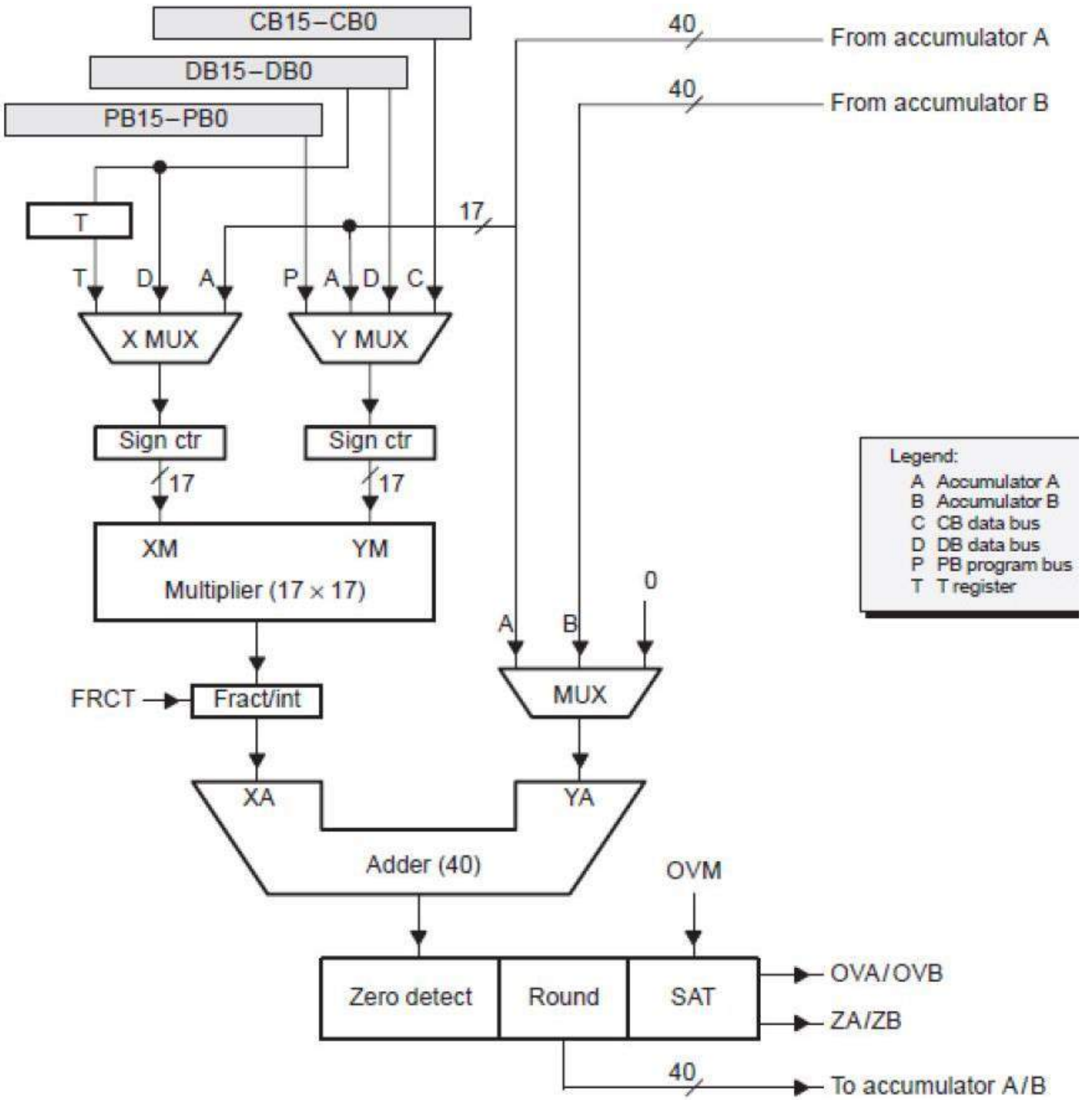



**Figure 4.4.** Functional diagram of the multiplier/adder unit of TMS320C54xx processors.

**Internal Memory and Memory-Mapped Registers:**

The amount and the types of memory of a processor have direct relevance to the efficiency and performance obtainable in implementations with the processors. The '54xx memory is organized into three individually selectable spaces: program, data, and I/O spaces. All '54xx devices contain both RAM and ROM. RAM can be either dual-access type (DARAM) or single-access type (SARAM). The on-chip RAM for these processors is organized in pages having 128 word locations on each page.

The '54xx processors have a number of CPU registers to support operand addressing and computations. The CPU registers and peripherals registers are all located on page 0 of the data

memory. Figure 4.5(a) and (b) shows the internal CPU registers and peripheral registers with their addresses. The processors mode status (PMST) registers
that is used to configure the processor. It is a memory-mapped register located at address 1Dh on page 0 of the RAM. A part of on-chip ROM may contain a boot loader and look-up tables for function such as sine, cosine, $\mu$- *law, and A*- law.

| NAME | DEC | HEX | DESCRIPTION |
|---|---|---|---|
| IMR | 0 | 0 | Interrupt mask register |
| IFR | 1 | 1 | Interrupt flag register |
| — | 2–5 | 2–5 | Reserved for testing |
| ST0 | 6 | 6 | Status register 0 |
| ST1 | 7 | 7 | Status register 1 |
| AL | 8 | 8 | Accumulator A low word (15–0) |
| AH | 9 | 9 | Accumulator A high word (31–16) |
| AG | 10 | A | Accumulator A guard bits (39–32) |
| BL | 11 | B | Accumulator B low word (15–0) |
| BH | 12 | C | Accumulator B high word (31–16) |
| BG | 13 | D | Accumulator B guard bits (39–32) |
| TREG | 14 | E | Temporary register |
| TRN | 15 | F | Transition register |
| AR0 | 16 | 10 | Auxiliary register 0 |
| AR1 | 17 | 11 | Auxiliary register 1 |
| AR2 | 18 | 12 | Auxiliary register 2 |
| AR3 | 19 | 13 | Auxiliary register 3 |
| AR4 | 20 | 14 | Auxiliary register 4 |
| AR5 | 21 | 15 | Auxiliary register 5 |
| AR6 | 22 | 16 | Auxiliary register 6 |
| AR7 | 23 | 17 | Auxiliary register 7 |
| SP | 24 | 18 | Stack pointer register |
| BK | 25 | 19 | Circular buffer size register |
| BRC | 26 | 1A | Block repeat counter |
| RSA | 27 | 1B | Block repeat start address |
| REA | 28 | 1C | Block repeat end address |
| PMST | 29 | 1D | Processor mode status (PMST) register |
| XPC | 30 | 1E | Extended program page register |
| — | 31 | 1F | Reserved |

**Figure 4.5(a)** Internal memory-mapped registers of TMS320C54xx processors.

| NAME | ADDRESS | | DESCRIPTION |
|---|---|---|---|
| | DEC | HEX | |
| DRR20 | 32 | 20 | McBSP 0 Data Receive Register 2 |
| DRR10 | 33 | 21 | McBSP 0 Data Receive Register 1 |
| DXR20 | 34 | 22 | McBSP 0 Data Transmit Register 2 |
| DXR10 | 35 | 23 | McBSP 0 Data Transmit Register 1 |
| TIM | 36 | 24 | Timer Register |
| PRD | 37 | 25 | Timer Period Register |
| TCR | 38 | 26 | Timer Control Register |
| — | 39 | 27 | Reserved |
| SWWSR | 40 | 28 | Software Watt-State Register |
| BSCR | 41 | 29 | Bank-Switching Control Register |
| — | 42 | 2A | Reserved |
| SWCR | 43 | 2B | Software Watt-State Control Register |
| HPIC | 44 | 2C | HPI Control Register (HMODE = 0 only) |
| — | 45–47 | 2D–2F | Reserved |
| DRR22 | 48 | 30 | McBSP 2 Data Receive Register 2 |
| DRR12 | 49 | 31 | McBSP 2 Data Receive Register 1 |
| DXR22 | 50 | 32 | McBSP 2 Data Transmit Register 2 |
| DXR12 | 51 | 33 | McBSP 2 Data Transmit Register 1 |
| SPSA2 | 52 | 34 | McBSP 2 Subbank Address Register |
| SPSD2 | 53 | 35 | McBSP 2 Subbank Data Register |
| — | 54–55 | 36–37 | Reserved |
| SPSA0 | 56 | 38 | McBSP 0 Subbank Address Register |
| SPSD0 | 57 | 39 | McBSP 0 Subbank Data Register |
| — | 58–59 | 3A–3B | Reserved |
| GPIOCR | 60 | 3C | General-Purpose I/O Control Register |
| GPIOSR | 61 | 3D | General-Purpose I/O Status Register |
| CSIDR | 62 | 3E | Device ID Register |
| — | 63 | 3F | Reserved |
| DRR21 | 64 | 40 | McBSP 1 Data Receive Register 2 |
| DRR11 | 65 | 41 | McBSP 1 Data Receive Register 1 |
| DXR21 | 66 | 42 | McBSP 1 Data Transmit Register 2 |
| DXR11 | 67 | 43 | McBSP 1 Data Transmit Register 1 |
| — | 68–71 | 44–47 | Reserved |
| SPSA1 | 72 | 48 | McBSP 1 Subbank Address Register |
| SPSD1 | 73 | 49 | McBSP 1 Subbank Data Register |
| — | 74–83 | 4A–53 | Reserved |
| DMPREC | 84 | 54 | DMA Priority and Enable Control Register |
| DMSA | 85 | 55 | DMA Subbank Address Register |

**Figure 4.5(b).**peripheral registers for the TMS320C54xx processors

**Status registers (ST0,ST1):**

**ST0:** Contains the status of flags (OVA, OVB, C, TC) produced by arithmetic operations & bit manipulations.

**ST1:** Contain the status of various conditions & modes. Bits of ST0&ST1registers can be set or clear with the SSBX & RSBX instructions.

**PMST:** Contains memory-setup status & control information.

**Status register0 diagram:**

| ARP (15-13) | TC (12) | C (11) | OVA (10) | OVB (9) | DP (8-0) |
|---|---|---|---|---|---|

Figure 4.6(a). ST0 diagram

ARP: Auxiliary register pointer.
TC: Test/control flag.
C: Carry bit.
OVA: Overflow flag for accumulator A.
OVB: Overflow flag for accumulator B.
DP: Data-memory page pointer.

**Status register1 diagram:**

| BRAF(15) | CPL (14) | XF (13) | HM (12) | INTM (11) | 0 (10) | OVM (9) | SXM (8) | C16 (7) | FRCT(6) | CMPT(5) | ASM (4-0) |
|---|---|---|---|---|---|---|---|---|---|---|---|

Figure 4.6(b). ST1 diagram

**BRAF**: **Block repeat active flag**
BRAF=0, the block repeat is deactivated.
BRAF=1, the block repeat is activated.

**CPL: Compiler mode**
CPL=0, the relative direct addressing mode using data page pointer is selected.
CPL=1, the relative direct addressing mode using stack pointer is selected.

**HM:** Hold mode, indicates whether the processor continues internal execution or acknowledge for external interface.

**INTM: Interrupt mode, it globally masks or enables all interrupts.**
INTM=0_all unmasked interrupts are enabled.
INTM=1_all masked interrupts are disabled.
0: Always read as 0

**OVM: Overflow mode.**
OVM=1_the destination accumulator is set either the most positive value or the most negative value.
OVM=0_the overflowed result is in destination accumulator.

**SXM: Sign extension mode.**
SXM=0 _Sign extension is suppressed.

SXM=1_Data is sign extended

**C16: Dual 16 bit/double-Precision arithmetic mode.**
C16=0_ALU operates in double-Precision arithmetic mode.
C16=1_ALU operates in dual 16-bit arithmetic mode.

**FRCT: Fractional mode**.
FRCT=1_the multiplier output is left-shifted by 1bit to compensate an extra sign bit.

**CMPT: Compatibility mode.**
CMPT=0_ ARP is not updated in the indirect addressing mode.
CMPT=1_ARP is updated in the indirect addressing mode.

**ASM: Accumulator Shift Mode.**
5 bit field, & specifies the Shift value within -16 to 15 range.

**Processor Mode Status Register (PMST):**

| IPTR(15-7) | MP/MC(6) | OVLY(5) | AVIS(4) | DROM(3) | CLKOFF(2) | SMUL(1) | SST(0) |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Fig 4.6(c) PMST Register block diagram

**INTR: Interrupt vector pointer**, point to the 128-word program page where the interrupt vectors reside.
MP/MC: Microprocessor/Microcomputer mode,
MP/MC=0, the on chip ROM is enabled.
MP/MC=1, the on chip ROM is enabled.

**OVLY: RAM OVERLAY,** OVLY enables on chip dual access data RAM blocks to be mapped into program space.

**AVIS:** It enables/disables the internal program address to be visible at the address pins.
**DROM: Data ROM**, DROM enables on-chip ROM to be mapped into data space.
CLKOFF: CLOCKOUT off.

**SMUL: Saturation on multiplication**.

**SST: Saturation on store.**

## Data Addressing Modes of TMS320C54X Processors:

Data addressing modes provide various ways to access operands to execute instructions and place results in the memory or the registers. The 54XX devices offer seven basic addressing modes

1. Immediate addressing.
2. Absolute addressing.
3. Accumulator addressing.
4. Direct addressing.
5. Indirect addressing.
6. Memory mapped addressing
7. Stack addressing.

Immediate addressing:

The instruction contains the specific value of the operand. The operand can be short (3,5,8 or 9 bit in length) or long (16 bits in length). The instruction syntax for short operands occupies one memory location,

Example: LD #20, DP.

RPT #0FFFFh.

Absolute Addressing:

The instruction contains a specified address in the operand.

1. Dmad addressing. MVDK Smem,dmad, MVDM dmad,MMR
2. Pmad addressing. MVDP Smem,pmad, MVPD pmem,Smad
3. PA addressing. PORTR PA, Smem,
4. *(lk) addressing .

Accumulator Addressing:

Accumulator content is used as address to transfer data between Program and Data memory.

Ex: READA *AR2

Direct Addressing:

Base address + 7 bits of value contained in instruction = 16 bit address. A page of 128 locations can be accessed without change in DP or SP.Compiler mode bit (CPL) in ST1 register is used.

If CPL =0 selects DP

CPL = 1 selects SP,

It should be remembered that when SP is used instead of DP, the effective address is computed by adding the 7-bit offset to SP.

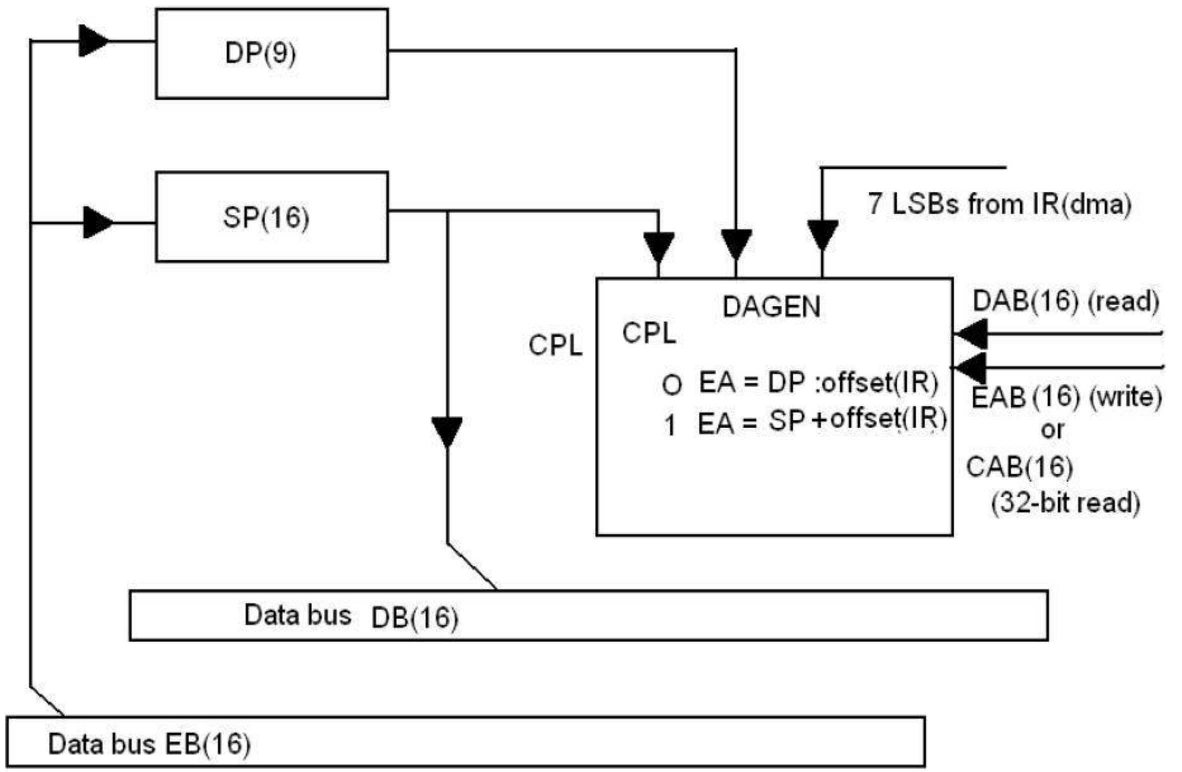



Figure 4.7 Block diagram of the direct addressing mode for TMS320C54xx Processors.

**Indirect Addressing:**

□Data space is accessed by address present in an auxiliary register.
TMS320C54xx have 8, 16 bit auxiliary register (AR0 – AR 7). Two auxiliary register arithmetic units (ARAU0 & ARAU1)
Used to access memory location in fixed step size. AR0 register is used for indexed and bit reverse addressing modes.
□For single – operand addressing
MOD _ type of indirect addressing
ARF _ AR used for addressing
ARP depends on (CMPT) bit in ST1
CMPT = 0, Standard mode, ARP set to zero
CMPT = 1, Compatibility mode, Particularly AR selected by ARP

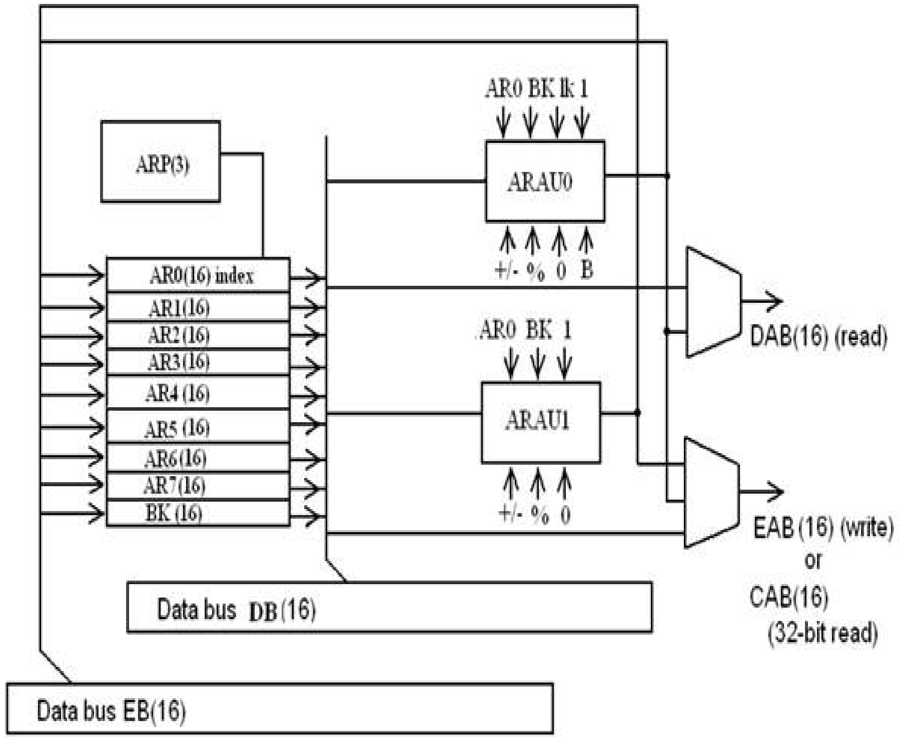



Figure 4.8Block diagram of the indirect addressing mode for TMS320C54xx Processors.

| Operand syntax | Function |
|---|---|
| *ARx | Addr = ARx; |
| *ARx - | Addr = ARx ; ARx = ARx -1 |
| *ARx + | Addr = ARx; ARx = ARx +1 |
| *+ARx | Addr = ARx+1; ARx = ARx +1 |
| *ARx - 0B | Addr = ARx ; ARx = B(ARx – AR0) |
| *ARx - 0 | Addr = Arx ; ARx = ARx – AR0 |
| *ARx + 0 | Addr = Arx ; ARx = ARx +AR0 |
| *ARx + 0B | Addr = ARx ; ARx = B(ARx + AR0) |
| *ARx - % | Addr = ARx ; ARx = circ(ARx – 1) |
| *+AR – 0% | Addr = Arx; ARx = circ(ARx - AR0) |
| *ARx + % | Addr = ARx ; ARx = circ (ARx + 1) |

Table 4.2 Indirect addressing options with a single data –memory operand.

Circular Addressing;

- Used in convolution, correlation and FIR filters.
- A circular buffer is a sliding window contains most recent data. Circular buffer of size R must start on a N-bit boundary, where 2N > R .
- The circular buffer size register (BK): specifies the size of circular buffer.
- Effective base address (EFB): By zeroing the N LSBs of a user selected AR (ARx).
- End of buffer address (EOB) : By repalcing the N LSBs of ARx with the N LSBs of BK.

  If 0 _ index + step < BK ; index = index +step;

  else if index + step _ BK ; index = index + step - BK;

  else if index + step < 0; index + step + BK

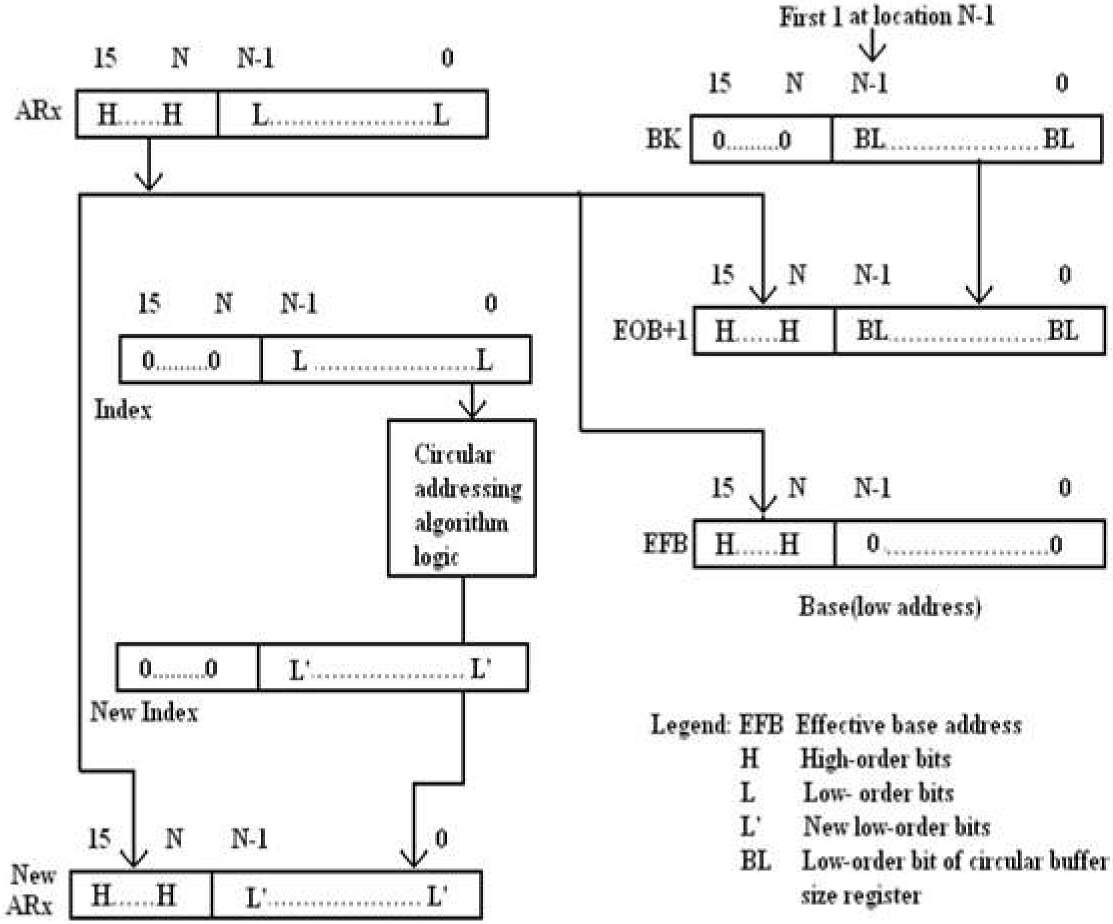



Figure 4.9 Block diagram of the circular addressing mode for TMS320C54xx Processors.

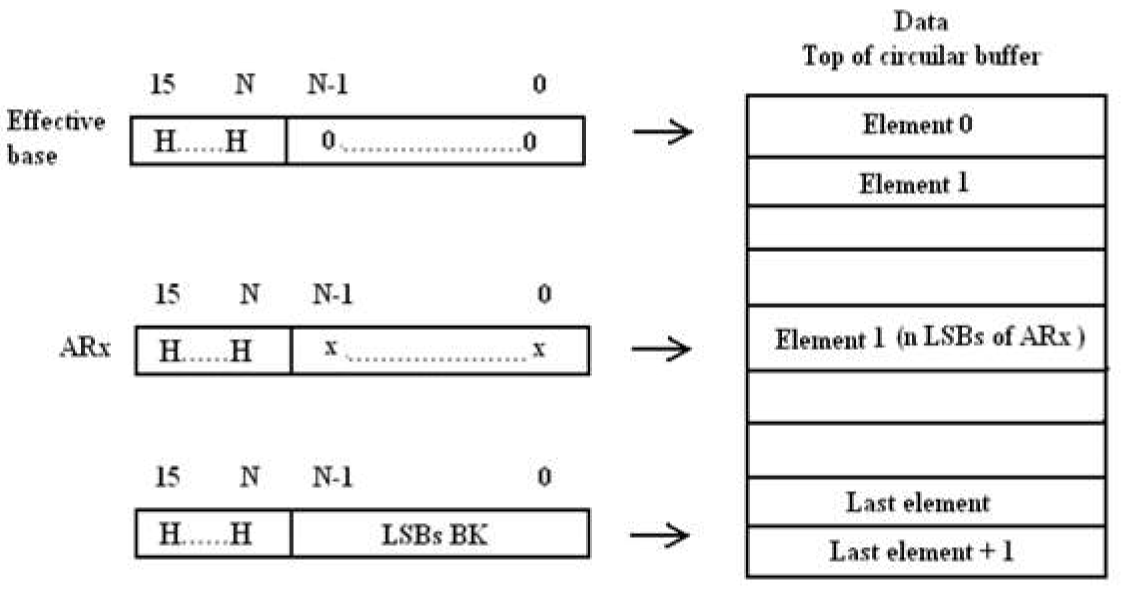



Figure 4.10 circular addressing mode implementation for TMS320C54xx Processors.

**Bit-Reversed Addressing:**

- Used for FFT algorithms.
- AR0 specifies one half of the size of the FFT.
- The value of AR0 = 2N-1: N = integer FFT size = 2N
- AR0 + AR (selected register) = bit reverse addressing.
- The carry bit propagating from left to right.

**Dual-Operand Addressing:**

Dual data-memory operand addressing is used for instruction that simultaneously perform two reads (32-bit read) or a single read (16-bit read) and a parallel store (16-bit store) indicated by two vertical bars, II. These instructions access operands using indirect addressing mode.

If in an instruction with a parallel store the source operand the destination operand point to the same location, the source is read before writing to the destination. Only 2 bits are available in the instruction code for selecting each auxiliary register in this mode. Thus, just four of the auxiliary registers, AR2-AR5, can be used, The ARAUs together with these registers, provide capability to access two operands in a single cycle. Figure 4.11 shows how an address is generated using dual data-memory operand addressing.

| 15 - 8 | 7 - 6 | 5 - 4 | 3 - 2 | 1 - 0 |
|---|---|---|---|---|
| Opcode | Xmod | Xar | Ymod | Yar |

| Name | Function |
|---|---|
| Opcode | This field contains the operation code for the instruction |
| Xmod | Defined the type of indirect addressing mode used for accessing the Xmem operand |
| XAR | Xmem AR selection field defines the AR that contains the address of Xmem |
| Ymod | Defies the type of inderect addressing mode used for accessing the Ymem operand |
| Yar | Ymem AR selection field defines the AR that contains the address of Ymem |

Table 4.3 Function of the different field in dual data memory operand addressing

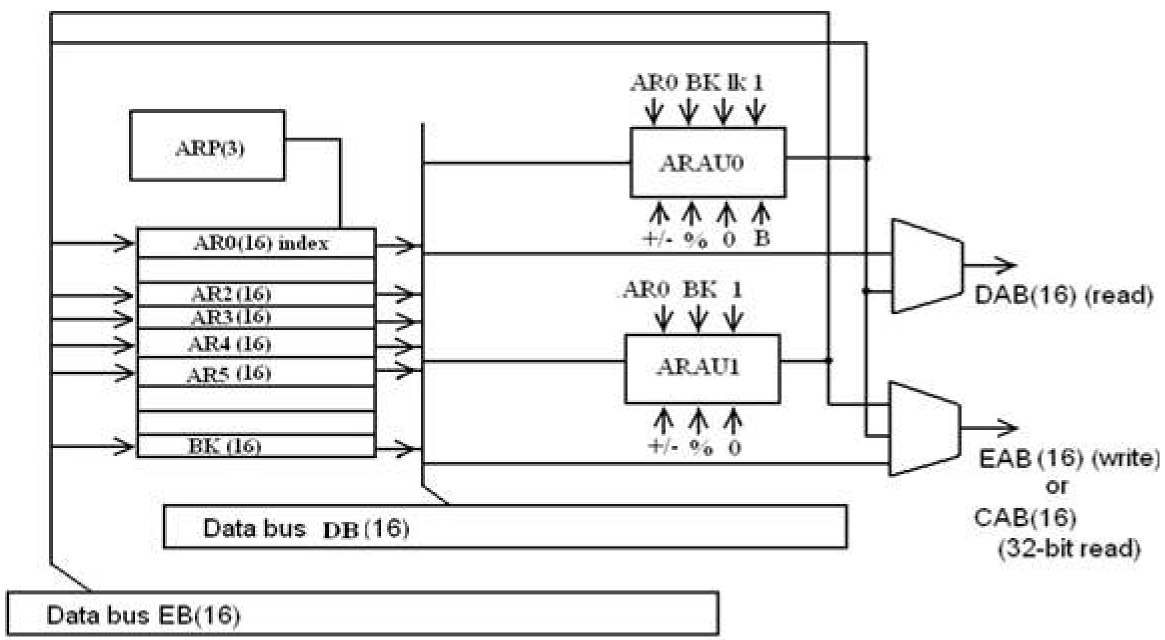



Figure 4.11 Block diagram of the Indirect addressing options with a dual data –memory operand.

**Memory-Mapped Register Addressing:**

- Used to modify the memory-mapped registers without affecting the current data page
- pointer (DP) or stack-pointer (SP)
  - Overhead for writing to a register is minimal
  - Works for direct and indirect addressing
  - Scratch –pad RAM located on data PAGE0 can be modified
- STM #x, DIRECT
- STM #tbl, AR1

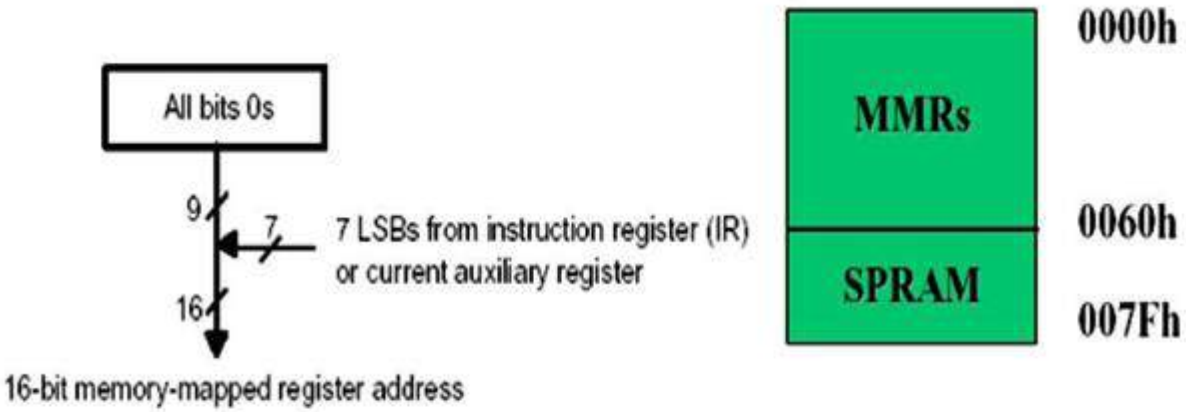



Figure 4.12. 16 bit memory mapped register address generation.

**4.4.7 Stack Addressing:**

- Used to automatically store the program counter during interrupts and subroutines.
- Can be used to store additional items of context or to pass data values.
- Uses a 16-bit memory-mapped register, the stack pointer (SP).
- PSHD X2

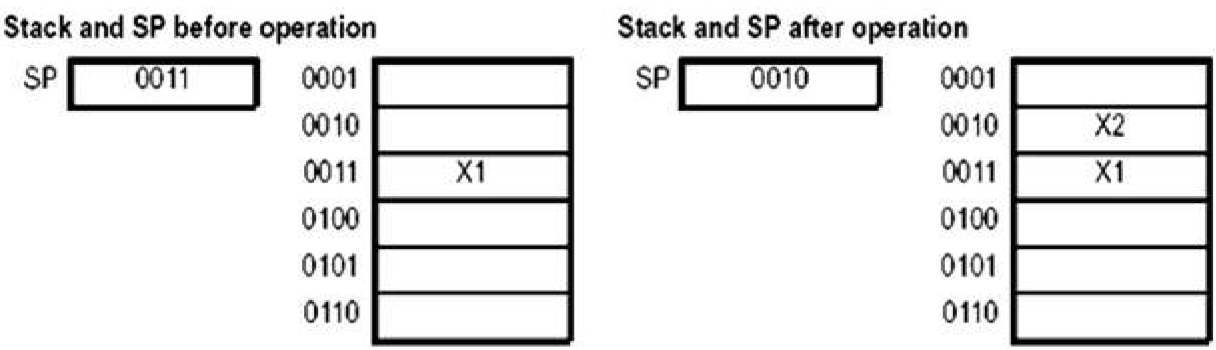
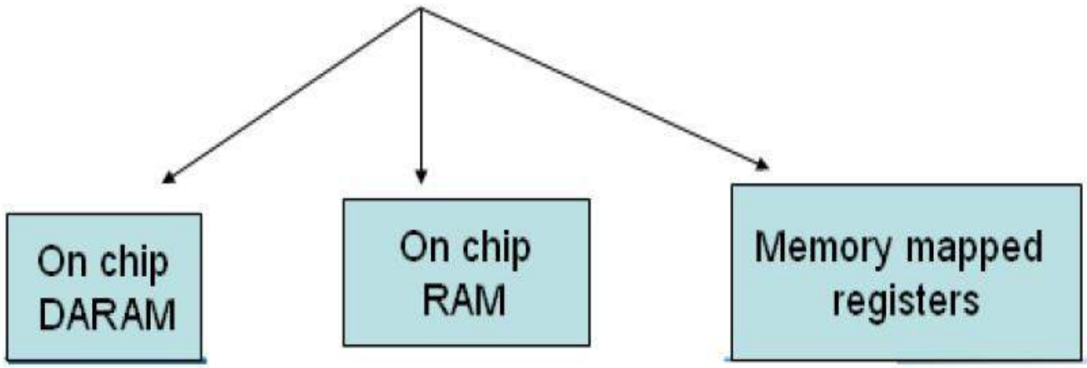



Figure 4.13 Values of stack &SP before and after operation.

**Memory Space of TMS320C54xx Processors**

- A total of 128k words extendable up to 8192k words.
- Total memory includes RAM, ROM, EPROM, EEPROM or Memory mapped peripherals.
- Data memory: To store data required to run programs & for external memory mapped registers.

Size 64k words




Program memory: To store program instructions &tables used in the execution of programs.

Organized into 128 pages, each of 64k word size

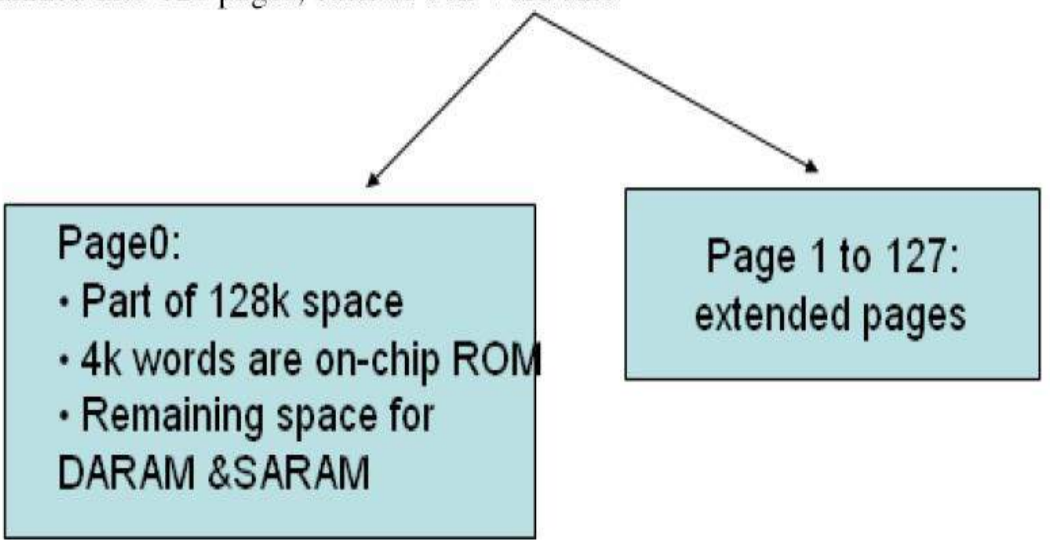
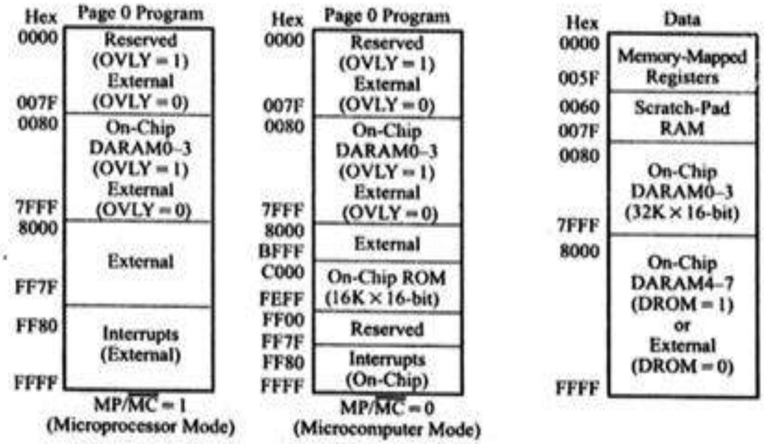


Table 4.4 Function of different pin PMST register

| PMST bit | Logic | On-chip memory configuration |
|---|---|---|
| MP/MC | 0<br>1 | ROM enabled<br>ROM not available |
| OVLY | 0<br>1 | RAM in data space<br>RAM in program space |
| DROM | 0<br>1 | ROM not in data space<br>ROM in data space |




Address ranges for on-chip DARAM in data memory are:

DARAM0: 0080h–1FFFh; DARAM1: 2000h–3FFFh
DARAM2: 4000h–5FFFh; DARAM3: 6000h–7FFFh
DARAM4: 8000h–9FFFh; DARAM5: A000h–BFFFh
DARAM6: C000h–DFFFh; DARAM7: E000h–FFFFh

Figure 3.14 Memory map for the TMS320C5416 Processor.

**Program Control**

- It contains program counter (PC), the program counter related H/W, hard stack, repeat counters &status registers.
- PC addresses memory in several ways namely:
- Branch: The PC is loaded with the immediate value following the branch instruction
- Subroutine call: The PC is loaded with the immediate value following the call instruction
- Interrupt: The PC is loaded with the address of the appropriate interrupt vector.
- Instructions such as BACC, CALA, etc ;The PC is loaded with the contents of the accumulator low word
- End of a block repeat loop: The PC is loaded with the contents of the block repeat program address start register.
- Return: The PC is loaded from the top of the stack.

**Problems:**

1. Assuming the current content of AR3 to be 200h, what will be its contents after each of the following TMS320C54xx addressing modes is used? Assume that the contents of AR0 are 20h.

a. *AR3+0
b. *AR3-0
c. *AR3+
d. *AR3
e. *AR3
f. *+AR3 (40h)
g. *+AR3 (-40h)

**Solution:**

a. AR3 ← AR3 + AR0;
AR3 = 200h + 20h = 220h
b. AR3← AR3 - AR0;
AR3 = 200h - 20h = 1E0h
c. AR3 ← AR3 + 1;
AR3 = 200h + 1 = 201h
d. AR3 ← AR3 - 1;
AR3 = 200h - 1 = 1FFh
e. AR3 is not modified.
AR3 = 200h
f. AR3 ← AR3 + 40h;
AR3 = 200 + 40h = 240h
g. AR3 ← AR3 - 40h;
AR3 = 200 - 40h = 1C0h